Angular Constraint Embedding via SpherePair Loss for Constrained Clustering

Shaojie Zhang **Ke Chen**
Department of Computer Science, The University of Manchester, Manchester M13 9PL, U.K.
{shaojie.zhang,ke.chen}@manchester.ac.uk

Abstract

Constrained clustering integrates domain knowledge through pairwise constraints. However, existing deep constrained clustering (DCC) methods are either limited by anchors inherent in end-to-end modeling or struggle with learning discriminative Euclidean embedding, restricting their scalability and real-world applicability. To avoid their respective pitfalls, we propose a novel angular constraint embedding approach for DCC, termed SpherePair. Using the SpherePair loss with a geometric formulation, our method faithfully encodes pairwise constraints and leads to embeddings that are clustering-friendly in angular space, effectively separating representation learning from clustering. SpherePair preserves pairwise relations without conflict, removes the need to specify the exact number of clusters, generalizes to unseen data, enables rapid inference of the number of clusters, and is supported by rigorous theoretical guarantees. Comparative evaluations with state-of-the-art DCC methods on diverse benchmarks, along with empirical validation of theoretical insights, confirm its superior performance, scalability, and overall real-world effectiveness. Code is available at our repository.

1 Introduction

Clustering is pivotal in machine learning and data mining. Unsupervised clustering methods, being fundamentally ill-posed, often partition data based solely on instance similarities or connections, which may misalign with domain knowledge [1]. To address this issue, integrating domain knowledge through weakly supervised methods like Constrained Clustering [2, 3, 4] has gained attention. These methods enforce both positive and negative instance-level pairwise constraints, significantly boosting clustering accuracy. Moreover, they offer a cost-effective solution in scenarios where obtaining pairwise relations is easier than acquiring class labels [5].

Most early pairwise constrained clustering (CC) methods adapt traditional unsupervised clustering by introducing constraints through modified similarity metrics or penalty functions [3, 6, 7, 8, 9]. Recent advances in deep clustering have led to the emergence of deep constrained clustering (DCC) paradigms, which outperform traditional methods, particularly on high-dimensional and complex data across various data types, and generalize well to unseen instances. Broadly, based on whether constraints are enforced at the level of cluster assignments or instance embeddings, we propose categorizing DCC methods into two paradigms: *end-to-end DCC* and *deep constraint embedding*.

As the dominant paradigm, end-to-end DCC methods (e.g., [10, 11, 12, 13, 14]) reformulate clustering as a pseudo-classification task by introducing anchors to represent classes, learning representations and cluster assignments jointly. However, the absence of global supervision hinders the proper alignment of anchors with cluster centers, resulting in a mismatch between local instance-level similarities and global cluster-level decisions. Moreover, these methods require prior knowledge of the number of clusters in the data to formulate the pseudo-classification task. These weaknesses, along with other technical issues reviewed in the next section, limit their practical usability in real

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

applications. In contrast, deep constraint embedding methods [15, 16] transform CC into traditional clustering through learned representations that encode constraint information using deep learning models. Nevertheless, these methods still struggle to maintain appropriate distances between positive and negative pairs in Euclidean space during representation learning.

In this paper, we propose the novel *SpherePair* loss function within a deep constraint embedding approach, addressing the limitations of existing DCC methods. As illustrated in Fig. 1, unlike existing DCC methods that either require anchors or rely on pairwise loss based on Euclidean distance, our SpherePair loss employs cosine similarity to learn a latent representation in angular space without relying on any anchors. This effectively balances pairwise relationships, resulting in a representation that accurately encodes constraint information towards minimizing intra-cluster distances and maximizing inter-cluster distances. Furthermore, our approach is supported by a theoretical foundation that ensures optimal performance under certain conditions, and does not require knowing the exact number of clusters, thereby reducing the effort required for hyper-parameter tuning. These strengths make our approach more scalable and practically applicable. As demonstrated, our approach outperforms state-of-the-art DCC baselines on various benchmark datasets, even when a simple K-means algorithm is applied to its learned representations.

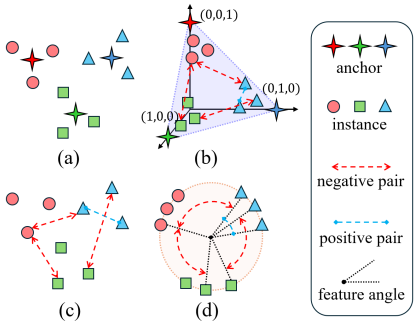


Figure 1: Different pairwise learning approaches. End-to-end DCC introduces anchors to transform features in (a) into soft cluster assignments in (b) for pairwise losses. Deep constraint embedding in (c) focuses on the Euclidean distances between features, while ours in (d) operates in angular space.

Our main contributions are summarized as follows: (i) We propose an angular constraint embedding method using the *SpherePair* loss, offering a scalable and practical solution for real-world CC tasks. (ii) We establish a rigorous theoretical foundation for our approach. (iii) We demonstrate that our approach can handle unknown number of clusters, rapidly infer the number of clusters and generalize effectively to unseen instances. (iv) We conduct extensive evaluations, demonstrating that our approach outperforms state-of-the-art DCC methods across diverse benchmark datasets.

2 Related work

End-to-end DCC. End-to-end DCC methods utilize a predefined number of anchors to connect representations (see Fig. 1(a)) and clustering assignments (see Fig. 1(b)), serving as key elements in structuring clustering (See Appendix A for a preliminary with the formal formulation of end-to-end DCC). Key differences arise from anchor configurations and pairwise losses: (i) *Anchors* include class weights from neural network classification layers (e.g., [14]), centroids of temporary clusters in embedding-based models (e.g., [11, 17, 12]), and components of Gaussian mixture distributions in generative models (e.g., [13]). (ii) *Pairwise losses* constrain clustering assignments, using measures such as Kullback–Leibler (KL) divergence [18, 19, 20] or inner products [10, 21, 13]. The Meta Classification Likelihood (MCL) loss [21] is theoretically validated and has inspired numerous extensions [22, 12, 23, 24, 14]. Despite their utility, anchors have notable limitations: anchors often struggle with data misaligned to explicit clustering centers [25] and miss local relationships due to their emphasis on a global perspective [26, 27]. In end-to-end DCC, this can further leads to potential mismatches, as global assignments can only be inferred indirectly from local pairwise relationships, especially when inappropriate anchors propagate errors during iterations. These issues are exacerbated by imbalanced constraints (i.e., many constraints arising from a small number of clusters), which fail to capture nuanced data structures and ultimately degrade clustering performance. Finally, these methods require the exact number of clusters, limiting practical usability and scalability.

Deep constraint embedding. Deep constraint embedding methods [15, 16] address CC problems by learning latent representations with deep learning models (see a preliminary in Appendix A for the formal formulation), typically using anchor-free, pairwise Euclidean distance-based losses (see Fig. 1(c)). However, AutoEmbedder [16] requires manually setting a margin due to the unbounded range of Euclidean distances $[0, +\infty)$. While CPAC [15] avoids margin tuning, its excessive expansion of negative pairs leads to non-convex, poorly distinguishable clusters, and its reliance on

connectivity graphs limits generalization to unseen instances. In contrast, our approach leverages angular distances between feature vectors (see Fig. 1(d)) to overcome the challenges of anchor-based and Euclidean distance-based methods. The angular space ensures equal, definitive inter-cluster distances without anchors, satisfying both positive and negative constraints while enabling inference of the true cluster number from this geometric configuration. Moreover, our approach is supported by a theoretical framework that guarantees optimal performance under specific conditions, eliminates the need for manual tuning, and determines the embedding space dimension without trial and error.

Deep angular learning. Deep learning methods in angular space have been extensively studied for supervised classification tasks [28, 29, 30, 31, 32] like face recognition, where each instance has a label. In contrast, CC involves defining pairwise relationships with sparse constraints applied only to subsets of data, presenting a unique challenge of learning representations while satisfying incomplete constraints without explicit labels. Some supervised prototype-based methods [33, 34, 35, 36, 37, 38] have explored angular output spaces by guiding instances to converge around equidistant prototypes, benefiting aspects of supervised tasks such as boundary discriminability for imbalanced classes and the alignment of Euclidean and cosine metrics [37]. However, these approaches rely on distances to class centers, making them unsuitable for instance-level pairwise learning in CC. Moreover, enforcing class margins does not help resolve complex pairwise constraints that cause conflicts during embedding learning. In contrast, our proposed method is the first to apply deep angular learning to CC. By focusing exclusively on the angles between feature vectors (see Fig. 1(d)), we establish equal inter-cluster distances without using anchors, effectively satisfying both positive and negative constraints while also generalizing well to unseen instances. Our approach leverages the closed nature of angular space to prevent constraint conflicts and is supported by a theoretical foundation.

3 SpherePair constraint clustering

CC aims to partition a dataset $\mathcal{X} = \{\boldsymbol{x}_j\}_{j=1}^{|\mathcal{X}|}$ into K clusters $\mathcal{S} = \{\mathcal{S}_k\}_{k=1}^{K}$ while satisfying pairwise constraints $\mathcal{C} = \{(a_i, b_i, y_i)\}_{i=1}^{|\mathcal{C}|}$, where each constraint (a_i, b_i, y_i) requires that instances $\boldsymbol{x}_{a_i}$ and $\boldsymbol{x}_{b_i}$ be in the same cluster if $y_i = 1$, or in different clusters if $y_i = 0$. To avoid the reliance on anchors inherent to end-to-end DCC, we learn a constrained yet clustering-friendly representation $\mathcal{Z} \subset \mathbb{R}^D$ to determine $\mathcal{S}$, constituting deep constraint embedding. Distinct from existing approaches in Euclidean space, we propose angular embedding to effectively preserve pairwise distances and eliminate the need for complex hyperparameter tuning. To facilitate the learning of $\mathcal{Z}$, we adopt a deep autoencoder with encoder $f_{\boldsymbol{\phi}} : \mathcal{X} \to \mathcal{Z}$ and decoder $g_{\boldsymbol{\phi}'} : \mathcal{Z} \to \mathcal{X}$, parameterized by $\boldsymbol{\phi}$ and $\boldsymbol{\phi}'$, respectively.

SpherePair loss. We formulate an anchor-free pairwise loss based on angular distance, optimizing the encoder $f_{\boldsymbol{\phi}}$ to generate latent representations $\mathcal{Z}$ aligned with constraints $\mathcal{C}$ in angular space. For each constrained pair $\boldsymbol{z}_{a_i}, \boldsymbol{z}_{b_i} \in \mathcal{Z}$, the angle $\theta_{z_{a_i}, z_{b_i}} \in [0, \pi]$ is normalized to a similarity score $\mathrm{Sim}(a_i, b_i) \in [0, 1]$ for constraint embedding using logistic loss, promoting angular similarity and separation for positive and negative pairs. The resulting SpherePair loss, $\mathcal{L}_{\text{ang}}$, is defined as:

$$\mathcal{L}_{\text{ang}} = -\frac{1}{|\mathcal{C}|} \sum_{i=1}^{|\mathcal{C}|} \Big(y_i \log \mathrm{Sim}(a_i, b_i) + (1 - y_i) \log(1 - \mathrm{Sim}(a_i, b_i)) \Big), \tag{1}$$

$$\mathrm{Sim}(a_i, b_i) = \frac{1}{2} \begin{cases} \cos(\theta_{\boldsymbol{z}_{a_i}, \boldsymbol{z}_{b_i}}) + 1, & \text{if } y_i = 1, \\ \cos(\min(\omega \theta_{\boldsymbol{z}_{a_i}, \boldsymbol{z}_{b_i}}, \pi)) + 1, & \text{if } y_i = 0. \end{cases}$$

Here, ω is an angular factor that ensures sufficient separation between clusters in the embedding space. Our proposed loss promotes constrained embedding learning in the angular space by enforcing the following: (i) for $(a_i, b_i, 1) \in \mathcal{C}$, smaller angles $\theta_{\boldsymbol{z}_{a_i}, \boldsymbol{z}_{b_i}}$ are favored to emphasize similarity; and (ii) for $(a_i, b_i, 0) \in \mathcal{C}$, a *negative zone* of angular size $\frac{\pi}{\omega}$ regulates the spacing of dissimilar pairs. The optimal negative-zone factor ω, theoretically determined in Section 4, mitigates conflicts among negative pairs in the embedding while ensuring sufficient separation. Notably, our SpherePair loss ensures a bounded angular distance $[0, \pi]$, providing stable similarity mapping and avoiding normalization issues associated with unbounded Euclidean distances $[0, +\infty)$ [11, 15].

Regularization and learning. While the SpherePair loss $\mathcal{L}_{\text{ang}}$ aligns $\mathcal{Z}$ with constraints $\mathcal{C}$, minimizing it directly may lead to degenerate representations that fail to capture intrinsic cluster structures. Inspired by deep clustering methods [39, 40], we incorporate a reconstruction loss:

$$\mathcal{L}_{\text{recon}} = \frac{1}{|\mathcal{X}|} \sum_{j=1}^{|\mathcal{X}|} \|\boldsymbol{x}_j - \hat{\boldsymbol{x}}_j\|_2^2. \tag{2}$$

Unlike DCC methods that directly construct latent representations [12, 41, 23], our autoencoder enforces instance reconstruction from normalized angular latent embeddings. To preserve angular properties during regularization, latent embeddings are normalized prior to decoding:

$$\hat{\boldsymbol{x}}_j = g_{\boldsymbol{\phi}'}(\text{Norm}(f_{\boldsymbol{\phi}}(\boldsymbol{x}_j))), \tag{3}$$

where $\text{Norm}(\cdot)$ ensures unit-length embeddings, preserving angular information. Thus, our overall objective for deep constraint embedding is:

$$\mathcal{L} = \mathcal{L}_{\text{ang}} + \lambda \mathcal{L}_{\text{recon}}, \tag{4}$$

where trade-off factor λ balances the angular loss in Eq. 1 and reconstruction loss in Eq. 2.

Minimizing the overall loss in Eq. 4 yields the optimal angular embeddings, $\mathcal{Z}^*$. As illustrated in Fig. 2, our angular constraint embedding learning compacts instances within the same cluster while separating different clusters using the negative zone in a (hyper)sphere. This structure simplifies subsequent clustering. Consequently, we name our deep constraint embedding framework *SpherePair*, and the resulting embeddings are spherical representations:

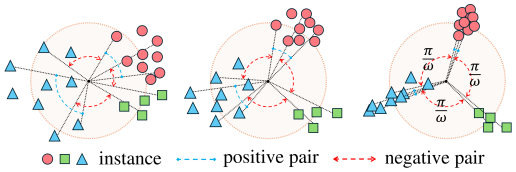


Figure 2: $\mathcal{Z}$ change in the SpherePair embedding learning (from left to right): the angular distances of positive pairs decrease, while those of negative pairs gradually adhere to the negative zone $\frac{\pi}{\omega}$.

$$\mathcal{Z}_{\text{sphere}} = \{\text{Norm}(\boldsymbol{z}_j^*) \mid \boldsymbol{z}_j^* \in \mathcal{Z}^*\}, \qquad \boldsymbol{z}_j^* = f_{\boldsymbol{\phi}^*}(\boldsymbol{x}_j). \tag{5}$$

Deployment. Applying an unsupervised clustering method to these spherical representations completes the CC process and enables generalization to unseen data. Note that for normalized features in $\mathcal{Z}_{\text{sphere}}$, the cosine and Euclidean distances can be shown to be equivalent [42], so either metric can be used for clustering. The SpherePair CC algorithm is outlined in Algorithm 1 in Appendix B.

Inferring cluster numbers. When the number of clusters is unknown, it can be inferred from the pre-learned $\mathcal{Z}_{\text{sphere}}$ via its geometric properties, thereby avoiding both retraining of the latent embedding as required by end-to-end methods and cumbersome post-clustering validation [43]. By applying principal component analysis (PCA) [44] to the training data involved in the negative constraints in $\mathcal{Z}_{\text{sphere}}$, we obtain all top-$d$ subspace projections. In each subspace, we compute inter-cluster angles, then select the smallest ρ-fraction $(0 < \rho \ll 1)$ of these angles and compute their tail average. With the theoretical justification presented in Section 4, we infer the number of clusters by identifying the onset of the plateau in the sequence of mean inter-cluster angles across subspaces. The cluster number inference algorithm is presented as Algorithm 2 in Appendix B.

4 Theoretical foundation

We establish a theoretical foundation for SpherePair to rigorously determine the negative-zone factor ω and embedding dimension D, which are indispensable for the unique properties of our angular constraint embedding, and further enable our inference of the unknown cluster number K. All proofs of the theortical results are provided in Appendix C.

To fix the negative-zone factor ω and prevent conflicts in negative pair embedding, we first establish the *conflict-free condition* for an optimal angular representation:

Proposition 4.1 (Conflict-free). *Let $\mathcal{S}^* = \{\mathcal{S}_k^*\}_{k=1}^K$ be the ground-truth partition of $\mathcal{X} = \{\boldsymbol{x}_j\}_{j=1}^{|\mathcal{X}|}$. An optimal angular representation $\mathcal{Z}^* = \{\boldsymbol{z}_j^*\}_{j=1}^{|\mathcal{X}|} \subset \mathbb{R}^D$ achieves $\mathcal{L}_{\text{ang}} = 0$. To ensure no conflicts in negative pairs, for any $\boldsymbol{x}_j \in \mathcal{S}_k^*$ and $\boldsymbol{x}_{j'} \in \mathcal{S}_{k'}^*$, $\theta_{\boldsymbol{z}_j^*, \boldsymbol{z}_{j'}^*} = 0$ if $k = k'$, and $\theta_{\boldsymbol{z}_j^*, \boldsymbol{z}_{j'}^*} \geq \frac{\pi}{\omega}$ if $k \neq k'$.*

Proof of Proposition 4.1 is given in Appendix C.1, based on which we further constrain ω as follows:

Proposition 4.2 (Equidistance). *Given $\mathcal{X} = \{\boldsymbol{x}_j\}_{j=1}^{|\mathcal{X}|}$ with ground-truth partition $\mathcal{S}^* = \{\mathcal{S}_k^*\}_{k=1}^K$, and a factor ω satisfying the conflict-free condition of $\mathcal{Z}^* = \{\boldsymbol{z}_j^*\}_{j=1}^{|\mathcal{X}|} \subset \mathbb{R}^D$, then for each constraint (a_i, b_i, y_i) from $\mathcal{S}^*$, the angle $\theta_{\boldsymbol{z}_{a_i}^*, \boldsymbol{z}_{b_i}^*}$ is uniquely determined by (a_i, b_i, y_i) if and only if: (i) $\mathcal{Z}^*$ is equidistant among clusters, i.e., all cross-cluster angles $\{\theta_{\boldsymbol{z}_j^*, \boldsymbol{z}_{j'}^*}\}$ are the same for any $\boldsymbol{x}_j \in \mathcal{S}_k^*$ and $\boldsymbol{x}_{j'} \in \mathcal{S}_{k' \neq k}^*$; (ii) ω matches this unique angular separation in (i), i.e. $\omega = \omega^*$, where $\theta_{\boldsymbol{z}_j^*, \boldsymbol{z}_{j'}^*} = \frac{\pi}{\omega^*}$.*

The proof is given in Appendix C.2. An ω satisfying Proposition 4.2 also meets the conflict-free condition of Proposition 4.1, ensuring each local pairwise constraint. Moreover, it enforces equidistant cluster embeddings, reflecting the fairness of all negative relationships $\{(a_i, b_i, 0)\} \subseteq \mathcal{C}$ and thus balancing pairwise relations. While Propositions 4.1 and 4.2 describe the ideal geometric configuration under $\mathcal{L}_{\text{ang}} = 0$, the following corollary complements them via perturbation analysis:

Corollary 4.3 (Geometric Deviations under Near-zero Residual Loss). *Given a set of $|\mathcal{C}|$ constraints $\mathcal{C} = \{(a_i, b_i, y_i)\}_{i=1}^{|\mathcal{C}|}$, and average angular loss $\mathcal{L}_{\text{ang}} \leq \varepsilon$ for some $0 < \varepsilon \ll 1$, then: (i) For each positive constraint $(a_i, b_i, 1) \in \mathcal{C}$ with $\theta_{z_{a_i}, z_{b_i}}$, we have $0 \leq \theta_{z_{a_i}, z_{b_i}} \leq \Delta^+(\varepsilon) := \arccos\big(2e^{-|\mathcal{C}|\varepsilon} - 1\big) \approx 2\sqrt{|\mathcal{C}|\varepsilon}$; (ii) For each negative constraint $(a_i, b_i, 0) \in \mathcal{C}$ with $\theta_{z_{a_i}, z_{b_i}} \leq \frac{\pi}{\omega}$, we have $0 \leq \frac{\pi}{\omega} - \theta_{z_{a_i}, z_{b_i}} \leq \Delta^-(\varepsilon) := \frac{1}{\omega}\arccos\big(1 - 2e^{-|\mathcal{C}|\varepsilon}\big) \approx 2\sqrt{|\mathcal{C}|\varepsilon}/\omega$.*

The proof of Corollary 4.3 is given in Appendix C.3. Corollary 4.3 shows that the geometric deviations degrade gracefully as $\mathcal{O}(\sqrt{\varepsilon})$ in the limit $\varepsilon \to 0$, approximately preserving the ideal configuration of Propositions 4.1 and 4.2 under small residual loss.

Despite only such tiny deviations from the ideal embedding, a valid ω required by Proposition 4.2 is not always guaranteed, since an equidistant cluster arrangement $\mathcal{Z}^*$ may fail in arbitrary dimension D. The feasibility and bounds of such ω depend on D and the number of clusters K. The following theorem provides conditions for the existence of a valid ω meeting Proposition 4.2:

Theorem 4.4 (Existence of Valid ω). *Given $\mathcal{X}$ with the ground truth partition $\mathcal{S}^* = \{\mathcal{S}_k^*\}_{k=1}^K$ containing K clusters. Suppose we seek an ω that matches a $\mathcal{Z}^* \subset \mathbb{R}^D$ with equidistant clusters, as formalized in Proposition 4.2. Then we have: (i) When $D < K - 1$, such a valid ω does not exist; (ii) When $D = K - 1$, the unique valid ω is $\pi / \arccos(-\frac{1}{K-1})$; (iii) When $D \geq K$, the range of valid ω values is relaxed to $\omega \geq \pi / \arccos(-\frac{1}{K-1})$.*

Proofs of Theorem 4.4 are in Appendix C.4. For each D, Theorem 4.4 establishes the existence conditions and bounds of ω that satisfy Proposition 4.2. With such ω, our SpherePair embedding learning drives $\mathcal{Z}$ to converge to $\mathcal{Z}_{\text{sphere}}$, where K clusters form a regular simplex in a $(K-1)$-dimensional subspace when $D \geq K - 1$ (Appendix D offers a 3D visualization of this convergence based on a real-world dataset). While any ω within the range specified by Theorem 4.4 is admissible, we further restrict the optimal setting through the following corollary:

Corollary 4.5 (Minimal Admissible ω). *Given $\mathcal{X}$ with the ground truth partition $\mathcal{S}^* = \{\mathcal{S}_k^*\}_{k=1}^K$ containing K clusters, and an embedding space $\mathbb{R}^D$ with sufficiently large $D \geq K$, the minimal admissible ω satisfying the validity condition of Theorem 4.4, denoted by $\omega_{\min}^*(K)$, is bounded as $1 \leq \omega_{\min}^*(K) < 2$ for all $K > 1$, and is monotone increasing in K with $\lim_{K\to\infty} \omega_{\min}^*(K) = 2$.*

The proof of Corollary 4.5 is provided in Appendix C.5. In pursuit of a larger negative-zone (given by $\frac{\pi}{\omega}$) for enhanced inter-cluster separability via a smaller ω, Corollary 4.5 prescribes $\omega = 2$ as the optimal setting for sufficiently large D, universally valid for any K. This theoretically fixes the choice of ω, leaving the attainment of conflict-free embeddings governed solely by D, with the lenient bound $D \geq K$ readily satisfied even when K is not precisely known. Therefore, our theoretical guarantees for ω and D in SpherePair CC eliminate the need for manual Euclidean margin tuning and strict embedding-dimension calibration [16], ensuring consistent scaling.

When the cluster number K is unknown, given the optimal $\mathcal{Z}^*$ (or $\mathcal{Z}_{\text{sphere}}$) learned by Algorithm 1 with valid ω and D (cf. Theorem 4.4), K can be inferred from its theoretically supported geometry: $\mathcal{Z}_{\text{sphere}}$ converges to a K-vertex regular simplex in a $(K-1)$-dimensional subspace, and its projection onto this subspace preserves the same configuration. This implies that the intrinsic embedding dimension $d^* = K-1$ is directly linked to the true cluster number. To exploit this link, we employ PCA and have the following theorem:

Theorem 4.6 (Pairwise-angle Invariance). *Given $\mathcal{X} = \{\boldsymbol{x}_j\}_{j=1}^{|\mathcal{X}|}$ with ground-truth partition $\mathcal{S}^* = \{\mathcal{S}_k^*\}_{k=1}^K$ and an optimal $\mathcal{Z}^* = \{\boldsymbol{z}_j^*\}_{j=1}^{|\mathcal{X}|} \subset \mathbb{R}^D$ with equidistant clusters satisfying Propositions 4.1 and 4.2. For $d \in \{1, \ldots, D\}$, let $\mathcal{Z}_{\text{pca}}^{(d)} = \{\tilde{\boldsymbol{z}}_j^{(d)}\}_{j=1}^{|\mathcal{X}|} \subset \mathbb{R}^d$ be the d-dimensional PCA projection of $\mathcal{Z}_{\text{sphere}} = \{\text{Norm}(\boldsymbol{z}_j^*)\}_{j=1}^{|\mathcal{X}|}$, and denote by $\theta_{\tilde{\boldsymbol{z}}_j^{(d)}, \tilde{\boldsymbol{z}}_{j'}^{(d)}}$ the pairwise angle between $\tilde{\boldsymbol{z}}_j^{(d)} \neq \mathbf{0}$ and $\tilde{\boldsymbol{z}}_{j'}^{(d)} \neq \mathbf{0}$. Then: (i) For every $d, d' \geq K-1$ and all such pairs (j, j'), $\theta_{\tilde{\boldsymbol{z}}_j^{(d)}, \tilde{\boldsymbol{z}}_{j'}^{(d)}} = \theta_{\tilde{\boldsymbol{z}}_j^{(d')}, \tilde{\boldsymbol{z}}_{j'}^{(d')}}$; (ii) For any $d, d' < K-1$, the cross-dimensional invariance in (i) cannot hold for arbitrary pairs (j, j').*

The proof is in Appendix C.6. Theorem 4.6 shows that the cross-d invariance of pairwise angles $\theta_{\tilde{\boldsymbol{z}}_j^{(d)}, \tilde{\boldsymbol{z}}_{j'}^{(d)}}$ determines d^*. To monitor this invariance, we consider the *minimal inter-cluster angle*,

$$\delta_d = \min\{\theta_{\tilde{\boldsymbol{z}}_j^{(d)}, \tilde{\boldsymbol{z}}_{j'}^{(d)}} \mid \boldsymbol{x}_j \in \mathcal{S}_k^*,\ \boldsymbol{x}_{j'} \in \mathcal{S}_{k' \neq k}^*\},$$

for which we have the following corollary:

Corollary 4.7 (δ_d Invariance). *Under the same conditions as in Theorem 4.6, and define the cluster frequencies $p_k = \frac{|\mathcal{S}_k^*|}{|\mathcal{X}|} > 0$ with $\sum_{k=1}^K p_k = 1$. Then: (i) If $K = 2$, the minimal inter-cluster angle $\delta_d = \pi,\ \forall d \geq 1$; (ii) If $K > 2$, then $\delta_1 = 0$, and there exists a constant $\delta_\star \in (\frac{\pi}{3}, \arccos(-\frac{1}{K-1})]$ such that $\delta_d = \delta_\star$ always holds when $d \geq K-1$. The upper bound $\arccos(-\frac{1}{K-1})$ of $\delta_\star$ is attained when $p_1 = p_2 = \cdots = p_K$, while the lower bound $\frac{\pi}{3}$ is approached when some $p_k \to 1$.*

Its proof is in Appendix C.7. Corollary 4.7 offers a practical reflection of the invariance in Theorem 4.6 through δ_d, which helps determine d^*, and hence K. Concretely, for $d = 1, \ldots, K-1$, δ_d increases from 0 (when $K > 2$) and reaches some $\delta_\star > \frac{\pi}{3}$ at $d = K-1$; for $d \geq K-1$, δ_d stabilizes around this $\delta_\star$. Thus, the onset of the plateau of sequence $\{\delta_d\}_{d=1}^D$ identifies $d^* = K-1$.

In practice, under the assumption that the training negative constraints $\mathcal{C}^-$ cover all true clusters, δ_d can be readily computed from $\mathcal{C}^-$, and due to small deviations (cf. Corollary 4.3), we replace δ_d by a tail-averaged variant $\bar{\delta}_d$ for stability. The sequence $\{\bar{\delta}_d\}_{d=1}^D$ is then used to locate the plateau entry d^* and estimate $\widehat{K} = d^* + 1$, which provides a theoretical foundation for Algorithm 2.

5 Experiments

5.1 Experimental settings

Our experimental settings aim to address the following questions: (i) How does our SpherePair perform compared to state-of-the-art DCC methods? (ii) How well do our SpherePair and baseline methods capture consistent instance relations under imbalanced constraint distributions? (iii) How effectively does our approach handle unknown cluster numbers? (iv) How are our theoretical insights empirically supported, and how sensitive is our approach to the introduced hyperparameters?

Datasets. We adopt eight benchmarks with diverse class counts and class balance: CIFAR-100-20 and CIFAR-10 [45], FashionMNIST [46], ImageNet-10 [47], MNIST [48], STL-10 [49], together with two imbalanced text datasets, Reuters subset [50] and RCV1-10 (see Appendix E.1 for details).

Baselines. We evaluate our SpherePair against three categories of DCC methods: (i) state-of-the-art end-to-end approaches, including VanillaDCC [21], VolMaxDCC [14], DCGMM [13], and CIDEC [12]; (ii) SDEC [11], which integrates Euclidean constraint embedding loss into end-to-end deep clustering; and (iii) AutoEmbedder [16], a fully anchor-free Eulidean constraint embedding method. For AutoEmbedder and SpherePair, K-means is applied to their learned representations for clustering unless otherwise specified. These baselines encompass the key advancements in DCC research.

Protocol. For FashionMNIST, MNIST, and the Reuters subset, we use the original pre-split training and test data settings. For the remaining benchmarks, we randomly split the data into 80% training and 20% test sets. Consistent with [14], we reserve a validation set of 1,000 instances from the training data to optimise the hyperparameters for baselines requiring such tuning. Constraints are generated based on the ground-truth labels of pairs sampled within the training sets. For a comparative study,

performance with different constraint set sizes (1k/5k/10k) is evaluated on training and test sets over five trials using three standard clustering metrics: *Accuracy* (ACC), *Normalised Mutual Information* (NMI), and *Adjusted Rand Index* (ARI). To assess performance under imbalanced constraints, we create a balanced set, `IMB0`, via uniform sampling and gradually introduce additional constraints linked to fewer clusters to form two imbalanced sets, `IMB1` and `IMB2`, where `IMB0` $\subset$ `IMB1` $\subset$ `IMB2` (see Appendix E.3 for detailed constraint generation procedure). To examine the effectiveness of cluster number inference, we repeat the procedure using embeddings pre-learned from five random initializations. To empirically validate our theoretical insights and assess the robustness of our approach, we explore a wide range of D, λ, and ρ settings.

Implementation. We strictly follow the same fully connected architectures from baseline papers [11, 12, 13, 14] for fair comparison and compatibility with both image and non-image datasets. For DCGMM, CIDEC, SDEC, AutoEmbedder, and SpherePair, we use a fully connected encoder with hidden layers of size 500–500–2000 (and a symmetric decoder when required), and an embedding layer of $D = 20$ for CIFAR-100-20 and $D = 10$ for the remaining datasets, unless stated otherwise. For VanillaDCC and VolMaxDCC, we use a fully connected network with two hidden layers of size 512–512 and a classification layer matching the number of clusters, K, as recommended in [14]. ReLU activations are used across all networks. Pretrained autoencoders are employed for model initialisation, except for VanillaDCC and VolMaxDCC. Specifically, a variational autoencoder is pretrained for DCGMM, while stacked denoising autoencoders are pretrained layer-wise for other models. Pretraining is performed unsupervised on the entire training set. In SpherePair and cluster number inference, ω is theoretically fixed at 2 as per Sect. 4, while $\lambda = 0.02$ and $\rho = 0.05$ are used by default unless varied for hyperparameter robustness evaluation. For baselines, we adopt reported optimal hyperparameters (VanillaDCC, DCGMM, CIDEC, SDEC) or follow the search procedures in VolMaxDCC and AutoEmbedder. SpherePair and all baselines (except DCGMM, where we use the authors' source code) are implemented in PyTorch 1.5.1. Training is conducted using the Adam optimizer, except for SDEC and VolMaxDCC, which employs SGD as suggested by their authors.

More details of our experimental settings are provided in Appendix E to ensure full replicability.

5.2 Experimental results

Using the experimental setup outlined in Sect. 5.1, we present and analyze our results to address the four motivating questions that guided our study. Additional results are provided in Appendix F.

5.2.1 Comparative performance

Overall comparison. Table 1 reports SpherePair's results against six baselines on all eight datasets under three constraint levels (1k/5k/10k). Out of 72 total comparisons (8 datasets $\times$ 3 constraint levels $\times$ 3 metrics), SpherePair ranks first in over 60 cases and second in nearly all others. It is notably dominant on CIFAR-100-20, FMNIST, REUTERS, and RCV1-10, achieving the top result in every metric at every constraint level (9/9 each), surpassing the second-best method by up to 4-16% in absolute ACC. Even when second-best, the performance gap is within 1-2%. Additional comparisons with AutoEmbedder using hierarchical clustering (replacing K-means) can be found in Appendix F.1. These findings showcase SpherePair's state-of-the-art DCC performance across both image and text datasets, along with the robustness of its geometric formulation under varying supervision levels and data domains.

Comparison without pretraining. We further compared two strong models without pretraining (random initialization), CIDEC† and SpherePair†. As shown in Table 1, SpherePair† exceeds CIDEC† by 10-20% on nearly all datasets except MNIST at 1k constraints; at 5k/10k it remains superior in most cases. Except for the highly class-imbalanced RCV1-10, pretrained SpherePair consistently yields a modest improvement (around 1%) over its unpretrained version. In contrast, CIDEC benefits substantially from pretraining only at 1k constraints, but shows inconsistent gains at higher constraint levels and even a drop on FMNIST. Hence, end-to-end DCC rely heavily on sufficiently good initial clustering to bootstrap reliable iterations, but our anchor-free SpherePair loss demonstrates robustness to initialization, particularly when pretraining is unavailable or leads to degraded performance.

Table 1: Comparative performance (%) (ACC, NMI, ARI) across datasets for models with 1k/5k/10k constraints. Blue and black represent training and test results, respectively. Best results are in **bold**, second-best are underlined, and † indicates models without pretraining.

			Vanilla-DCC	VolMax-DCC	CIDEC†	CIDEC	DCGMM	SDEC	Auto-Embedder	SpherePair† (Ours)	SpherePair (Ours)
CIFAR100-20	1k	ACC	34.2, 34.3	20.1, 20.3	32.8, 33.0	46.6, 46.2	44.5, 44.2	45.7, 45.4	21.5, 21.6	45.1, 45.1	**48.3**, **48.2**
		NMI	36.0, 36.3	21.4, 21.6	35.1, 35.6	47.3, 47.9	44.9, 45.4	47.0, 47.5	23.1, 23.4	44.9, 45.4	**47.7**, **48.0**
		ARI	19.3, 19.3	7.1, 7.2	19.7, 19.6	30.0, 29.9	28.7, 28.7	29.0, 29.2	7.1, 7.1	29.4, 29.5	**32.2**, **32.4**
	5k	ACC	47.4, 47.4	42.8, 42.8	42.3, 42.1	46.7, 46.1	48.1, 47.9	45.6, 45.1	13.8, 14.2	55.4, 55.7	**59.0**, **58.8**
		NMI	46.7, 47.1	41.9, 42.1	42.3, 42.5	45.4, 45.7	46.7, 47.1	47.0, 47.5	13.5, 13.8	51.1, 51.7	**52.6**, **53.0**
		ARI	32.2, 32.2	22.8, 22.8	27.1, 26.8	30.3, 29.6	32.2, 32.2	29.2, 29.3	4.7, 4.7	39.1, 39.4	**41.0**, **40.9**
	10k	ACC	54.6, 54.5	51.2, 51.0	49.8, 49.8	50.9, 50.1	52.3, 52.1	45.7, 45.2	31.3, 31.3	60.5, 60.4	**62.8**, **62.6**
		NMI	50.2, 50.3	48.5, 48.7	47.4, 47.6	48.5, 48.	49.2, 49.6	47.1, 47.7	36.6, 36.9	53.9, 54.3	**55.1**, **55.5**
		ARI	37.9, 37.6	33.4, 33.3	33.4, 33.2	34.0, 33.0	36.7, 36.7	29.3, 29.5	20.6, 20.4	43.4, 43.4	**45.3**, **45.2**
CIFAR10	1k	ACC	70.2, 70.1	65.2, 64.9	64.9, 65.1	**86.5**, **86.5**	82.1, 82.1	84.0, 84.1	58.2, 58.5	84.3, 84.2	85.7, 85.6
		NMI	67.0, 66.9	62.5, 62.4	60.2, 60.3	**78.8**, **78.9**	75.0, 74.9	76.5, 76.5	57.8, 58.1	75.6, 75.4	77.3, 77.1
		ARI	57.8, 57.6	48.6, 48.3	50.1, 50.3	**75.2**, **75.1**	69.7, 69.6	70.5, 70.6	43.1, 43.3	71.7, 71.5	74.0, 73.7
	5k	ACC	87.6, 87.3	84.9, 84.6	86.4, 86.2	88.9, 88.7	88.3, 88.0	85.4, 85.5	85.9, 85.8	88.9, 88.7	**89.2**, **88.9**
		NMI	79.3, 79.0	79.0, 78.6	78.3, 78.0	80.9, 80.8	80.2, 79.8	78.1, 78.2	79.2, 79.3	80.7, 80.3	**81.2**, **80.9**
		ARI	76.9, 76.4	75.2, 74.5	75.0, 74.5	79.0, 78.5	78.0, 77.4	73.4, 73.4	75.7, 75.6	79.1, 78.6	**79.6**, **79.1**
	10k	ACC	90.0, 89.5	90.0, 89.5	88.8, 88.4	90.1, 89.8	89.9, 89.7	85.6, 85.6	87.7, 87.4	**90.5**, **90.0**	**90.5**, 89.9
		NMI	81.6, 80.9	81.3, 80.6	81.3, 80.6	82.0, **81.8**	81.9, 81.6	78.2, 78.3	80.7, 80.4	**82.3**, 81.6	**82.3**, 81.6
		ARI	80.4, 79.5	80.3, 79.5	79.4, 78.5	80.7, 80.1	80.4, 80.0	73.8, 73.8	78.2, 77.7	81.3, **80.4**	**81.4**, **80.4**
FMNIST	1k	ACC	56.6, 56.3	50.9, 50.6	52.7, 52.0	58.0, 57.6	64.7, 63.5	56.7, 56.6	39.6, 39.4	62.8, 62.1	**70.3**, **69.8**
		NMI	56.4, 56.1	49.5, 49.1	53.4, 52.9	61.1, 60.3	62.0, 61.1	62.0, 61.2	41.1, 41.0	60.1, 59.5	**62.3**, **61.7**
		ARI	43.9, 43.3	33.3, 32.7	38.2, 37.5	44.9, 43.9	49.6, 48.4	44.7, 43.5	25.0, 24.7	49.8, 49.1	**52.7**, **51.9**
	5k	ACC	76.2, 75.2	76.0, 75.3	71.7, 71.2	64.6, 63.8	78.5, 77.3	57.3, 57.3	59.0, 58.6	80.1, 79.0	**81.0**, **79.9**
		NMI	67.4, 66.5	67.3, 66.6	65.7, 65.1	61.2, 60.4	70.7, 69.8	62.8, 62.0	57.9, 57.5	70.8, 69.7	**72.0**, **70.9**
		ARI	61.4, 60.0	60.8, 59.8	57.1, 56.3	48.7, 47.5	64.5, 63.1	45.6, 44.5	45.6, 45.0	65.2, 63.7	**66.8**, **65.3**
	10k	ACC	80.3, 79.0	80.2, 78.7	77.7, 76.7	74.7, 74.0	81.5, 80.3	58.1, 58.2	68.1, 67.5	83.6, 82.3	**84.8**, **83.6**
		NMI	71.3, 70.1	71.2, 69.6	71.3, 70.3	68.9, 68.2	73.8, 72.3	63.1, 62.4	65.0, 64.3	73.8, 72.4	**75.6**, **74.2**
		ARI	66.4, 64.6	66.2, 63.9	65.0, 63.7	61.4, 60.2	68.6, 66.8	46.1, 45.2	55.0, 53.9	69.8, 67.9	**72.0**, **70.1**
ImageNet10	1k	ACC	83.4, 83.6	84.0, 83.9	83.9, 84.1	92.2, 92.7	94.3, 94.4	89.0, 88.9	61.2, 60.7	**95.9**, 95.6	**95.9**, **95.9**
		NMI	83.1, 83.7	81.7, 82.9	81.0, 82.2	88.3, 88.8	89.1, 89.4	84.8, 84.4	55.7, 55.4	90.6, **91.1**	**90.7**, **91.1**
		ARI	77.0, 76.9	75.9, 76.1	75.2, 74.9	85.8, 86.3	88.9, 88.8	80.7, 80.1	39.5, 38.3	**91.2**, 91.1	**91.2**, **91.2**
	5k	ACC	96.8, 96.3	96.8, 96.4	96.3, 96.2	96.8, 96.5	96.6, 96.3	89.5, 89.5	96.4, 96.3	96.8, 96.4	**96.9**, **96.6**
		NMI	92.5, 92.3	**92.6**, **92.7**	91.6, 92.1	92.4, 92.2	91.8, 91.5	86.1, 86.0	91.6, 91.6	92.2, 92.5	92.4, 92.2
		ARI	**93.4**, 92.1	93.2, 92.3	92.2, 91.8	93.1, 92.5	92.6, 91.9	82.2, 82.0	92.3, 92.0	93.2, 92.3	93.2, **92.7**
	10k	ACC	97.0, 96.2	96.9, 96.2	97.1, 96.5	97.2, 96.6	97.0, 96.5	89.6, 89.6	96.8, 96.6	97.0, 96.2	**97.3**, **96.7**
		NMI	**93.2**, 92.2	92.8, 92.4	92.9, 92.3	93.0, 92.4	92.7, 92.0	86.3, 86.4	92.2, 92.2	**93.2**, **92.6**	**93.2**, 92.5
		ARI	**94.1**, 91.9	93.7, 91.9	93.7, 92.4	93.9, 92.7	93.6, 92.5	82.4, 82.4	93.0, 92.7	94.0, 92.4	**94.1**, **93.0**
MNIST	1k	ACC	54.4, 54.9	57.4, 57.3	65.1, 65.8	88.6, 88.0	84.4, 84.6	84.4, 84.7	43.2, 43.4	69.8, 71.0	**91.6**, **91.7**
		NMI	48.1, 49.5	50.8, 51.6	62.7, 63.0	**86.8**, **86.0**	80.4, 80.8	79.6, 80.5	35.5, 36.6	59.8, 61.5	82.5, 82.8
		ARI	39.0, 40.0	40.8, 41.2	51.9, 52.1	**83.5**, 82.3	75.4, 75.7	75.9, 76.5	22.9, 23.4	54.2, 55.8	82.6, **82.8**
	5k	ACC	82.2, 82.6	76.5, 77.0	90.2, 89.5	95.8, 95.4	94.0, 94.1	86.2, 86.1	58.5, 58.9	93.2, 93.3	**96.1**, **95.9**
		NMI	75.1, 76.2	68.0, 69.1	85.5, 84.3	**91.6**, **90.7**	88.7, 88.7	82.6, 83.0	57.5, 58.8	84.5, 84.9	90.0, 89.9
		ARI	72.4, 73.2	64.2, 65.1	83.9, 82.4	**92.0**, 91.0	88.6, 88.6	79.4, 79.4	46.9, 47.5	85.7, 86.0	91.5, **91.2**
	10k	ACC	92.6, 92.6	91.4, 91.3	96.5, 95.6	**97.5**, **97.0**	95.6, 95.3	86.3, 86.1	81.5, 81.8	95.6, 95.6	97.1, **97.0**
		NMI	84.1, 84.3	82.9, 83.2	91.3, 89.7	**93.5**, **92.6**	91.2, 90.8	83.0, 83.3	77.7, 78.7	89.0, 89.1	92.1, 92.1
		ARI	85.3, 85.3	83.6, 83.5	92.5, 90.7	**94.7**, **93.7**	91.7, 91.2	79.6, 79.5	74.1, 74.9	90.7, 90.6	93.7, **93.7**
REUTERS	1k	ACC	71.1, 70.8	66.7, 66.7	70.7, 71.9	77.0, 76.9	85.1, 87.1	72.6, 71.2	49.5, 48.8	87.1, 86.0	**91.2**, **91.4**
		NMI	42.3, 43.0	38.8, 39.8	48.2, 51.1	59.3, 60.5	68.8, 68.2	52.4, 51.5	15.2, 14.7	64.1, 62.5	**72.9**, **74.1**
		ARI	49.1, 49.0	45.7, 45.4	51.3, 54.6	66.1, 66.9	77.7, 77.0	57.3, 55.8	14.6, 13.7	74.1, 71.9	**81.0**, **81.5**
	5k	ACC	96.1, **94.8**	95.5, 94.3	82.0, 82.3	93.0, 92.7	95.7, 94.4	72.1, 71.6	85.6, 84.9	95.6, 93.9	**96.2**, **94.8**
		NMI	84.6, **81.5**	83.0, 79.6	60.5, 61.4	79.4, 78.3	83.6, 80.2	54.6, 54.6	63.7, 62.9	83.1, 78.2	**84.9**, 80.7
		ARI	90.9, **88.1**	89.7, 86.6	68.0, 68.2	86.6, 85.3	90.3, 87.1	57.1, 57.7	73.7, 71.8	89.7, 85.3	**91.0**, 87.2
	10k	ACC	97.4, 94.9	97.1, 94.4	88.1, 86.3	97.6, **95.9**	97.5, 95.6	71.6, 71.2	92.8, 91.0	**97.8**, 95.2	**97.8**, 95.8
		NMI	89.1, 80.9	88.2, 79.6	70.8, 68.7	89.8, **84.5**	89.3, 83.5	54.0, 53.9	78.6, 75.4	**90.3**, 81.9	90.2, 83.9
		ARI	94.2, 87.5	93.6, 86.3	80.3, 76.0	94.5, **90.1**	94.3, 89.3	56.1, 56.8	86.6, 82.7	**95.0**, 87.9	**95.0**, 89.8
STL10	1k	ACC	69.9, 70.0	74.4, 74.5	69.8, 71.1	88.0, 87.8	77.6, 77.2	85.5, 85.6	50.6, 51.0	86.4, 87.2	**88.1**, **88.1**
		NMI	64.9, 65.6	69.4, 69.5	65.4, 67.3	**79.5**, **79.2**	70.0, 70.1	77.2, 77.1	49.9, 50.5	76.6, 78.4	78.7, 79.1
		ARI	54.5, 54.7	61.9, 61.8	57.0, 58.9	76.5, 76.1	63.7, 63.2	71.8, 72.1	33.6, 33.9	74.2, 75.7	**76.7**, **76.8**
	5k	ACC	88.1, 87.1	88.9, 87.9	87.2, 86.9	90.3, 89.4	88.0, 87.1	87.3, 87.6	84.0, 83.7	91.2, 90.1	**91.4**, **90.3**
		NMI	80.6, 79.5	80.2, 79.4	78.6, 79.1	81.9, 80.9	78.7, 78.1	79.1, 79.3	77.8, 77.4	83.0, 81.7	**83.1**, **81.9**
		ARI	78.3, 76.5	78.0, 76.5	76.0, 76.0	80.4, 78.9	77.0, 75.7	75.0, 75.6	73.7, 72.8	82.2, 80.3	**82.5**, **80.5**
	10k	ACC	92.6, 89.9	92.4, 90.1	90.7, 89.7	91.3, 89.9	91.4, 89.5	87.8, 88.0	90.9, 89.6	**93.0**, **91.0**	**93.0**, **91.0**
		NMI	84.9, 81.8	84.2, 81.8	82.3, 81.7	83.2, 81.5	82.7, 80.7	79.4, 79.6	82.5, 81.0	**85.4**, **83.2**	**85.4**, **83.2**
		ARI	84.7, 80.0	83.9, 80.0	81.3, 79.7	82.4, 79.8	82.3, 79.1	75.9, 76.3	81.5, 79.2	85.4, **81.9**	**85.5**, 81.7
RCV1-10	1k	ACC	38.1, 38.1	50.0, 49.9	41.7, 42.3	39.8, 39.2	34.7, 34.6	39.5, 39.4	33.4, 33.4	52.1, 52.2	**65.8**, **65.8**
		NMI	12.3, 12.3	40.1, 40.3	32.6, 33.6	45.1, 46.2	17.8, 17.9	49.6, 49.9	10.4, 10.4	47.6, 47.8	**60.8**, **61.1**
		ARI	12.9, 12.8	39.5, 39.4	27.2, 28.4	31.0, 31.1	12.3, 12.2	32.5, 32.4	7.8, 7.8	42.5, 42.5	**55.8**, **55.9**
	5k	ACC	78.1, 78.1	80.7, 80.6	57.6, 55.1	61.4, 61.7	54.1, 53.5	39.5, 39.4	49.4, 49.4	86.7, 86.5	**89.8**, **89.6**
		NMI	61.1, 60.9	61.9, 61.8	44.8, 47.5	53.7, 54.4	38.2, 38.0	50.9, 51.2	31.6, 31.5	69.2, 68.9	**74.2**, **73.9**
		ARI	70.7, 70.5	75.3, 75.1	46.4, 46.2	53.4, 54.0	40.2, 39.3	33.0, 33.0	35.9, 35.8	81.3, 80.9	**83.8**, **83.4**
	10k	ACC	84.3, 84.1	87.8, 87.5	58.2, 54.5	80.0, 79.4	81.0, 80.0	38.8, 38.8	77.3, 77.3	89.9, 89.8	**91.8**, **91.5**
		NMI	70.0, 69.6	70.8, 70.3	47.3, 51.0	66.9, 67.7	63.4, 62.7	50.5, 50.8	58.4, 58.2	74.9, 74.6	**78.0**, **77.4**
		ARI	81.6, 81.2	83.0, 82.4	47.9, 47.1	74.5, 73.9	75.4, 74.4	32.8, 32.7	68.9, 68.6	85.5, 85.2	**87.0**, **86.4**

5.2.2 Imbalanced constraints

Fig. 3 presents the test ACC under the imbalanced constraint setting outlined in Sect. 5.1. As the imbalance increases, all DCC baselines experience significant performance degradation across most datasets. In contrast, our SpherePair demonstrates remarkable stability and consistently outperforms

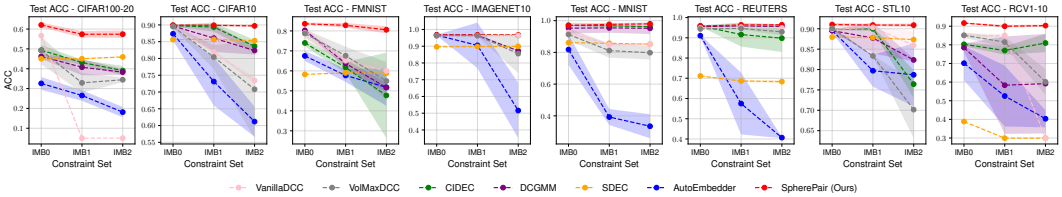


Figure 3: Test ACC performance (mean±std over 5 runs) of all models across datasets under the balanced vs. imbalanced constraints setting where ($|\texttt{IMB0}|$, $|\texttt{IMB1}|$, $|\texttt{IMB2}|$) = (10k, 50k, 100k).

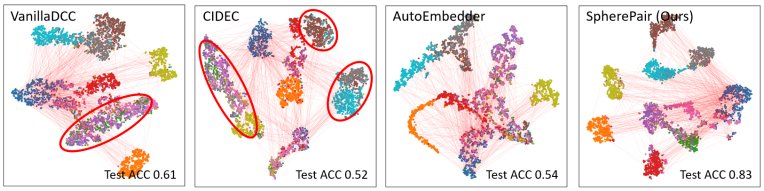


Figure 4: t-SNE visualizations of learned FMNIST embeddings under the IMB2 setting in Fig. 3. Marker colors denote ground-truth categories, and dashed lines represent pairwise constraints. The red circles highlight the misclustered instances.

all baselines across the tasks. Figure 4 visualizes the learned FMNIST embeddings, illustrating how SpherePair preserves coherent structures while forming more discriminative clusters compared to representative baselines. Notably, while the imbalanced sets contain more constraints than the balanced set, the performance of baselines worsens with greater imbalance, posing an open question for further investigation. Additional results and visualizations are in Appendix F.2, further showcasing the robustness of our approach in real-world scenarios with prevalent imbalanced constraints.

5.2.3 Unknown cluster number

We evaluate our cluster-number inference on SpherePair embeddings, simulating the unknown-K case with large embedding dimensions ($D = 50$ for CIFAR-100-20 and $D = 20$ for others). As shown in Fig. 5, the tail-averaged minimal inter-cluster angle $\overline{\delta}_d$ rises from 0 as the PCA subspace dimension d increases and, for most datasets, reaches a plateau at $\overline{\delta}_{K-1}$, which clearly reveals the true cluster number K. On CIFAR-100-20, the plateau is less distinct, yet the estimated K falls within 18-22 around the true $K = 20$. The only notable deviation is observed on RCV1-10, where strong class imbalance results in only six dominant clusters being identified, underscoring the inherent difficulty of K-inference in such settings. Nevertheless, our approach demonstrates overall robustness and efficiency, as further validated in Appendix F.3 with additional results under varied constraint levels, comparisons to alternative K-inference methods, and evaluations against other DCC methods.

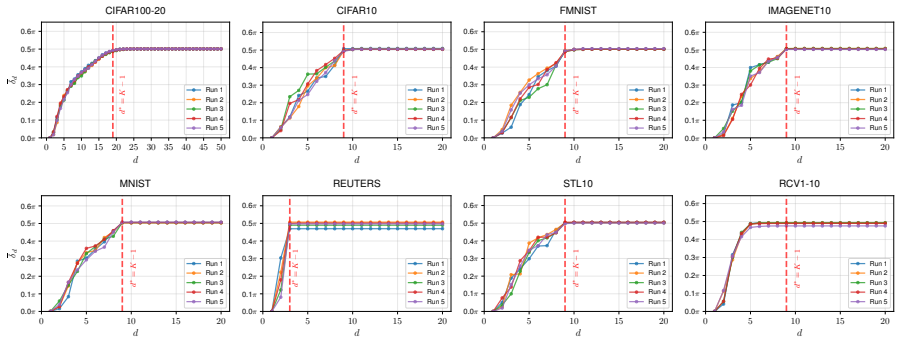


Figure 5: Tail-averaged minimal inter-cluster angle $\overline{\delta}_d$ vs. PCA subspace dimension d, obtained from SpherePair embeddings learned with 10k constraints across 5 runs. The red lines indicate the ground-truth intrinsic dimensions $d^* = K-1$.

5.2.4 Empirical validation and hyperparameter sensitivity analysis

Embedding dimension D. We study the impact of embedding dimension D, the only hyperparameter to be specified to obtain a conflict-free angular embedding, to empirically validate our

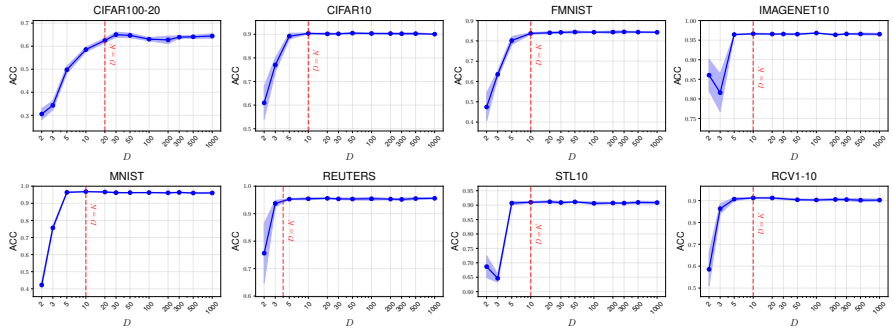


Figure 6: SpherePair test ACC (mean±std, 5 runs) vs. embedding dimension D across datasets (10k constraints). The red lines indicate the theoretical boundary between insufficient and sufficient D.

theoretical insights. Fig. 6 presents performance with respect to D under 10k balanced constraints (see Appendix F.4.1 for results under more constraint levels and clustering metrics), showing that: (i) SpherePair is robust to choices of $D \geq K$, even up to $D = 1000$, which corresponds to 50–250$\times$ the cluster number K across different datasets. This provides clear and easily satisfied practical guidance. (ii) Even selecting D slightly below the theoretical threshold minimally impacts performance, offering flexibility when K is unavailable. Additional results in Appendix F.4.1 further support this D-flexibility through comparisons with baselines on CIFAR-100-20. (iii) Ablation-like comparisons between theoretically sufficient and insufficient D illustrate the effectiveness of our conflict-free constraint embedding in angular space, and empirically validate our theoretical insights in Section 4.

Other hyperparameters. We further conduct sensitivity analysis on the regularization strength λ over $[0, 1]$ and the tail ratio ρ over $[0.01, 0.2]$ (see Appendices F.4.2 and F.4.3). Key observations are: (i) SpherePair embedding is broadly robust to λ, though an appropriate λ can be beneficial, particularly under limited constraints or random initialization. We recommend a default of $\lambda = 0.02$ for consistent performance in most cases; (ii) in cluster-number inference, smaller ρ sharpens rises before $\overline{\delta}_{K-1}$, while larger ρ stabilizes the subsequent plateau; $\rho \in [0.03, 0.1]$ generally offers a favorable trade-off for clearer K estimation.[1]

In summary, the experimental findings provide robust validation of our contributions. Notably, our method also demonstrates significant potential for real-world applications in terms of its learning efficiency (see Appendix G) and insensitivity to model structures (see Appendix H).

6 Conclusion

In this paper, we propose SpherePair, a novel representation learning approach for constrained clustering. It learns effective representations from pairwise constraints in an angular space, supported by theoretical guarantees. Extensive experiments on real-world and benchmark datasets demonstrate that SpherePair, when integrated with a simple clustering algorithm such as K-means, consistently outperforms various state-of-the-art DCC baselines. Furthermore, SpherePair is anchor-free, requires minimal hyperparameter tuning, offers robustness with theoretical guarantees, and readily handles an unknown number of clusters while rapidly inferring their quantity, making it highly applicable to real-world constrained clustering tasks.

Our approach has two limitations: (i) It currently supports only single-view unstructured data; (ii) It does not address incomplete or noisy constraint annotations. To address these challenges, we are extending SpherePair to handle semi-structured, structured, and multi-view data, as well as introducing mechanisms to manage noisy or incomplete annotations. We also aim to combine end-to-end and deep constraint embedding frameworks to capture higher-order correlations and improve scalability, particularly for applications requiring a deeper understanding of local and global relationships. We anticipate that addressing these challenges will result in more robust SpherePair models, improving the applicability and performance across diverse real-world data.

[1] Robustness can be enhanced by evaluating consistency across multiple ρ values within this range.

Acknowledgment

We are grateful to the anonymous reviewers for their comments, which improved the presentation of this paper. S.J. Zhang's work was supported by the UoM-CSC scholarship.

References

[1] Ian Davidson and Sugato Basu. A survey of clustering with instance level constraints. *ACM Transactions on Knowledge Discovery from data*, 1(1-41):2–42, 2007. 1

[2] Kiri Wagstaff and Claire Cardie. Clustering with instance-level constraints. *AAAI/IAAI*, 1097(577-584):197, 2000. 1

[3] Kiri Wagstaff, Claire Cardie, Seth Rogers, Stefan Schrödl, et al. Constrained k-means clustering with background knowledge. In *Icml*, volume 1, pages 577–584, 2001. 1

[4] Germán González-Almagro, Daniel Peralta, Eli De Poorter, José-Ramón Cano, and Salvador García. Semi-supervised constrained clustering: An in-depth overview, ranked taxonomy and future research directions. *arXiv preprint arXiv:2303.00522*, 2023. 1

[5] Yucen Luo, Tian Tian, Jiaxin Shi, Jun Zhu, and Bo Zhang. Semi-crowdsourced clustering with deep generative models. *Advances in Neural Information Processing Systems*, 31, 2018. 1

[6] Zhengdong Lu and Todd Leen. Semi-supervised learning with penalized probabilistic clustering. *Advances in neural information processing systems*, 17, 2004. 1

[7] Mikhail Bilenko, Sugato Basu, and Raymond J Mooney. Integrating constraints and metric learning in semi-supervised clustering. In *Proceedings of the twenty-first international conference on Machine learning*, page 11, 2004. 1

[8] Sugato Basu, Arindam Banerjee, and Raymond J Mooney. Active semi-supervision for pairwise constrained clustering. In *Proceedings of the 2004 SIAM international conference on data mining*, pages 333–344. SIAM, 2004. 1

[9] Brian Kulis, Sugato Basu, Inderjit Dhillon, and Raymond Mooney. Semi-supervised graph clustering: a kernel approach. In *Proceedings of the 22nd international conference on machine learning*, pages 457–464, 2005. 1

[10] Yen-Chang Hsu, Zhaoyang Lv, Joel Schlosser, Phillip Odom, and Zsolt Kira. A probabilistic constrained clustering for transfer learning and image category discovery. *arXiv preprint arXiv:1806.11078*, 2018. 1, 2

[11] Yazhou Ren, Kangrong Hu, Xinyi Dai, Lili Pan, Steven CH Hoi, and Zenglin Xu. Semi-supervised deep embedded clustering. *Neurocomputing*, 325:121–130, 2019. 1, 2, 3, 6, 7, 31, 32, 33, 36, 37

[12] Hongjing Zhang, Tianyang Zhan, Sugato Basu, and Ian Davidson. A framework for deep constrained clustering. *Data Mining and Knowledge Discovery*, 35:593–620, 2021. 1, 2, 4, 6, 7, 31, 32, 36, 37, 55

[13] Laura Manduchi, Kieran Chin-Cheong, Holger Michel, Sven Wellmann, and Julia Vogt. Deep conditional gaussian mixture model for constrained clustering. *Advances in Neural Information Processing Systems*, 34:11303–11314, 2021. 1, 2, 6, 7, 31, 32, 33, 36, 37

[14] Tri Nguyen, Shahana Ibrahim, and Xiao Fu. Deep clustering with incomplete noisy pairwise annotations: A geometric regularization approach. In *International Conference on Machine Learning*, pages 25980–26007. PMLR, 2023. 1, 2, 6, 7, 31, 32, 34, 37, 38

[15] Sharon Fogel, Hadar Averbuch-Elor, Daniel Cohen-Or, and Jacob Goldberger. Clustering-driven deep embedding with pairwise constraints. *IEEE computer graphics and applications*, 39(4):16–27, 2019. 2, 3

[16] Abu Quwsar Ohi, Muhammad Firoz Mridha, Farisa Benta Safir, Md Abdul Hamid, and Muhammad Mostafa Monowar. Autoembedder: A semi-supervised dnn embedding system for clustering. *Knowledge-Based Systems*, 204:106190, 2020. 2, 5, 6, 31, 32, 33, 34, 37, 38

[17] Hongjing Zhang, Sugato Basu, and Ian Davidson. A framework for deep constrained clustering-algorithms and advances. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2019, Würzburg, Germany, September 16–20, 2019, Proceedings, Part I*, pages 57–72. Springer, 2020. 2

[18] Yen-Chang Hsu and Zsolt Kira. Neural network-based clustering using pairwise constraints. *arXiv preprint arXiv:1511.06321*, 2015. 2

[19] Ankita Shukla, Gullal S Cheema, and Saket Anand. Semi-supervised clustering with neural networks. In *2020 IEEE Sixth International Conference on Multimedia Big Data (BigMM)*, pages 152–161. IEEE, 2020. 2

[20] Elham Amirizadeh and Reza Boostani. CDEC: A constrained deep embedded clustering. *International Journal of Intelligent Computing and Cybernetics*, 14(4):686–701, 2021. 2

[21] Yen-Chang Hsu, Zhaoyang Lv, Joel Schlosser, Phillip Odom, and Zsolt Kira. Multi-class classification without multi-class labels. *arXiv preprint arXiv:1901.00544*, 2019. 2, 6, 32

[22] Marek Śmieja, Łukasz Struski, and Mário AT Figueiredo. A classification-based approach to semi-supervised clustering with pairwise constraints. *Neural Networks*, 127:193–203, 2020. 2

[23] Tian Tian, Jie Zhang, Xiang Lin, Zhi Wei, and Hakon Hakonarson. Model-based deep embedding for constrained clustering analysis of single cell rna-seq data. *Nature communications*, 12(1):1873, 2021. 2, 4

[24] Jann Goschenhofer, Bernd Bischl, and Zsolt Kira. Constraintmatch for semi-constrained clustering. In *2023 International Joint Conference on Neural Networks (IJCNN)*, pages 1–10. IEEE, 2023. 2

[25] Yi Wen, Suyuan Liu, Xinhang Wan, Siwei Wang, Ke Liang, Xinwang Liu, Xihong Yang, and Pei Zhang. Efficient multi-view graph clustering with local and global structure preservation. In *Proceedings of the 31st ACM International Conference on Multimedia*, pages 3021–3030, 2023. 2

[26] Jian Dai, Zhenwen Ren, Yunzhi Luo, Hong Song, and Jian Yang. Tensorized anchor graph learning for large-scale multi-view clustering. *Cognitive Computation*, 15(5):1581–1592, 2023. 2

[27] Suyuan Liu, Qing Liao, Siwei Wang, Xinwang Liu, and En Zhu. Robust and consistent anchor graph learning for multi-view clustering. *IEEE Transactions on Knowledge and Data Engineering*, 2024. 2

[28] Weiyang Liu, Yandong Wen, Zhiding Yu, and Meng Yang. Large-margin softmax loss for convolutional neural networks. In *International Conference on Machine Learning*, pages 507–516. PMLR, 2016. 3

[29] Weiyang Liu, Yandong Wen, Zhiding Yu, Ming Li, Bhiksha Raj, and Le Song. Sphereface: Deep hypersphere embedding for face recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 212–220, 2017. 3

[30] Weiyang Liu, Yan-Ming Zhang, Xingguo Li, Zhiding Yu, Bo Dai, Tuo Zhao, and Le Song. Deep hyperspherical learning. *Advances in neural information processing systems*, 30, 2017. 3

[31] Hao Wang, Yitong Wang, Zheng Zhou, Xing Ji, Dihong Gong, Jingchao Zhou, Zhifeng Li, and Wei Liu. Cosface: Large margin cosine loss for deep face recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5265–5274, 2018. 3

[32] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4690–4699, 2019. 3

[33] Pascal Mettes, Elise Van der Pol, and Cees Snoek. Hyperspherical prototype networks. *Advances in neural information processing systems*, 32, 2019. 3

[34] Yueqi Duan, Jiwen Lu, and Jie Zhou. Uniformface: Learning deep equidistributed representation for face recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3415–3424, 2019. 3

[35] Florian Graf, Christoph Hofer, Marc Niethammer, and Roland Kwitt. Dissecting supervised contrastive learning. In *International Conference on Machine Learning*, pages 3821–3830. PMLR, 2021. 3

[36] Tianhong Li, Peng Cao, Yuan Yuan, Lijie Fan, Yuzhe Yang, Rogerio S Feris, Piotr Indyk, and Dina Katabi. Targeted supervised contrastive learning for long-tailed recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6918–6928, 2022. 3

[37] Hakan Cevikalp, Hasan Saribas, and Bedirhan Uzun. Reaching nirvana: Maximizing the margin in both euclidean and angular spaces for deep neural network classification. *IEEE Transactions on Neural Networks and Learning Systems*, 2024. 3

[38] Hakan Cevikalp, Hasan Serhan Yavuz, and Hasan Saribas. Deep uniformly distributed centers on a hypersphere for open set recognition. In *Asian Conference on Machine Learning*, pages 217–230. PMLR, 2024. 3

[39] Bo Yang, Xiao Fu, Nicholas D Sidiropoulos, and Mingyi Hong. Towards k-means-friendly spaces: Simultaneous deep learning and clustering. In *international conference on machine learning*, pages 3861–3870. PMLR, 2017. 4

[40] Xifeng Guo, Long Gao, Xinwang Liu, and Jianping Yin. Improved deep embedded clustering with local structure preservation. In *Ijcai*, volume 17, pages 1753–1759, 2017. 4, 33, 37

[41] Yi Cui, Xianchao Zhang, Linlin Zong, and Jie Mu. Maintaining consistency with constraints: A constrained deep clustering method. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 219–230. Springer, 2021. 4

[42] David Barber. *Bayesian Reasoning and Machine Learning*, page 341. Cambridge University Press, 2012. 4

[43] Brian S Everitt, Sabine Landau, Morven Leese, and Daniel Stahl. *Cluster Analysis*. Wiley, Chichester, UK, 5th edition, 2011. 4, 36

[44] Ian Jolliffe. Principal component analysis. In *International encyclopedia of statistical science*, pages 1094–1096. Springer, 2011. 4, 36

[45] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 6, 31

[46] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017. 6, 31

[47] Jianlong Chang, Lingfeng Wang, Gaofeng Meng, Shiming Xiang, and Chunhong Pan. Deep adaptive image clustering. In *Proceedings of the IEEE international conference on computer vision*, pages 5879–5887, 2017. 6, 31

[48] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 6, 31

[49] Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 215–223. JMLR Workshop and Conference Proceedings, 2011. 6, 32

[50] Junyuan Xie, Ross Girshick, and Ali Farhadi. Unsupervised deep embedding for clustering analysis. In *International conference on machine learning*, pages 478–487. PMLR, 2016. 6, 30, 31, 33, 37

[51] David D Lewis, Yiming Yang, Tony Russell-Rose, and Fan Li. Rcv1: A new benchmark collection for text categorization research. *Journal of machine learning research*, 5(Apr):361–397, 2004. 30, 31

[52] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 31

[53] Yunfan Li, Peng Hu, Zitao Liu, Dezhong Peng, Joey Tianyi Zhou, and Xi Peng. Contrastive clustering. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 8547–8555, 2021. 31, 37

[54] Zhuxi Jiang, Yin Zheng, Huachun Tan, Bangsheng Tang, and Hanning Zhou. Variational deep embedding: An unsupervised and generative approach to clustering. *arXiv preprint arXiv:1611.05148*, 2016. 33

[55] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 37

[56] Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006. 37

[57] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pages 1096–1103. ACM, 2008. 37

[58] Weiwei Gu, Aditya Tandon, Yong-Yeol Ahn, and Filippo Radicchi. Principled approach to the selection of the embedding dimension of networks. *Nature Communications*, 12(1):3772, 2021. 51

[59] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. *arXiv preprint arXiv:1611.03530*, 2016. 51

[60] Devansh Arpit, Stanisław Jastrzębski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S Kanwal, Tegan Maharaj, Asja Fischer, Aaron Courville, Yoshua Bengio, et al. A closer look at memorization in deep networks. In *International conference on machine learning*, pages 233–242. PMLR, 2017. 51

[61] Zeyuan Allen-Zhu, Yuanzhi Li, and Yingyu Liang. Learning and generalization in overparameterized neural networks, going beyond two layers. *Advances in neural information processing systems*, 32, 2019. 51

[62] Peter L Bartlett, Philip M Long, Gábor Lugosi, and Alexander Tsigler. Benign overfitting in linear regression. *Proceedings of the National Academy of Sciences*, 117(48):30063–30070, 2020. 51

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The main contributions and scope of the paper are detailed in the abstract and Sect. 1. Theoretical foundations are in Sect. 4 and Appendix C, while Sect. 5 and Appendices F to H provide solid empirical evidence.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Yes, please see Sect. 6 for limitations.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: We provide the assumptions and complete proofs of the theoretical results in Sect. 4 and Appendix C.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: Detailed descriptions of all the information necessary to reproduce the results are provided in Sect. 5.1 and Appendix E. We also provide code and instructions to reproduce the results in our repository.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The sources of the publicly available datasets we use are detailed in Appendix E.1, and the external codes used in experiments are listed in Appendix E.4. We also provide code and instructions to reproduce the results in our repository.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: The key experimental settings and details are presented in Sect. 5.1, and the full details are provided in Appendix E.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Shaded regions (in the imbalanced constraints results and hyperparameter sensitivity plots, i.e., Figs. 3, 6, 10 to 12 and 19 to 22) and error bars (in the network structure effect bar chart, i.e., Fig. 23) indicate mean$\pm$standard deviation over 5 runs. While the main comparison results in Table 1 omit statistical significance due to space constraints, they share identical experimental settings with the `IMBO` results in the imbalanced constraints plots (Figs. 3 and 10 to 12), where standard deviations are reported.

   Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification The compute resources used for the experiments are described in Appendix E.4, and we also provide information on learning efficiency in Appendix G.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: We have strictly followed the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

Justification: This paper aims to contribute to the advancement of the field of Machine Learning. While our work may have various societal implications, we do not believe any particular aspect requires explicit emphasis at this stage.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: This work does not pose misuse risks. It focuses on fundamental research without deploying any models or assets that require safeguards.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: Yes, we credited them in appropriate ways.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.

- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, paperswithcode.com/datasets has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: The paper does not release new assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (https://neurips.cc/Conferences/2025/LLM) for what should or should not be described.

A Deep constraint clustering formulations

The goal of constrained clustering (CC) is to partition a dataset $\mathcal{X} = \{\boldsymbol{x}_j\}_{j=1}^{|\mathcal{X}|}$ into K clusters $\mathcal{S} = \{\mathcal{S}_k\}_{k=1}^{K}$ while satisfying a set of pairwise constraints $\mathcal{C} = \{(a_i, b_i, y_i)\}_{i=1}^{|\mathcal{C}|}$. Each constraint (a_i, b_i, y_i) requires that instances $\boldsymbol{x}_{a_i}$ and $\boldsymbol{x}_{b_i}$ be assigned to the same cluster if $y_i = 1$, or to different clusters if $y_i = 0$. To solve this task, deep constraint clustering (DCC) methods employ a deep encoder $f_{\boldsymbol{\phi}} : \mathcal{X} \to \mathcal{Z} \subset \mathbb{R}^D$, parameterized by $\boldsymbol{\phi}$, to embed each instance into a latent representation $\boldsymbol{z}_j = f_{\boldsymbol{\phi}}(\boldsymbol{x}_j)$, thereby forming the representation set $\mathcal{Z} = \{\boldsymbol{z}_j\}_{j=1}^{|\mathcal{X}|}$. Depending on how the latent representations $\mathcal{Z}$ are utilized to satisfy constraints and perform clustering, existing DCC methods can be categorized into two paradigms: *end-to-end DCC* and *deep constraint embedding*.

End-to-end DCC. End-to-end DCC introduces additional anchors $\mathcal{A}$ to structure $\mathcal{Z}$, enabling soft cluster assignment $\mathcal{Q}$ that satisfy constraints $\mathcal{C}$. An activation function $\sigma_{\mathcal{A}} : \mathcal{Z} \to \mathcal{Q}$ maps each $\boldsymbol{z}_j$ to a soft assignment $\boldsymbol{q}_j = \sigma_{\mathcal{A}}(\boldsymbol{z}_j) \in \Delta^{K-1}$, where Δ^{K-1} is the probability simplex. Typically, $\mathcal{A}$ consists of K class weight vectors in a classification layer, and is combined with $f_{\boldsymbol{\phi}}$ to form the classifier $h_{\boldsymbol{\phi},\mathcal{A}}(\boldsymbol{x}) = \sigma_{\mathcal{A}}(f_{\boldsymbol{\phi}}(\boldsymbol{x}))$. Alternatively, $\mathcal{A}$ can be independent learnable parameters requiring specific initialization. A generic anchor-based pairwise loss function used in end-to-end DCC is defined as:

$$\mathcal{L}(\sigma_{\mathcal{A}}, f_{\boldsymbol{\phi}}; \mathcal{C}) = \frac{1}{|\mathcal{C}|} \sum_{i=1}^{|\mathcal{C}|} \ell\left(\sigma_{\mathcal{A}}(f_{\boldsymbol{\phi}}(\boldsymbol{x}_{a_i})), \sigma_{\mathcal{A}}(f_{\boldsymbol{\phi}}(\boldsymbol{x}_{b_i})), y_i\right).$$

This loss function measures how well $\mathcal{Q}$ satisfies $\mathcal{C}$, enabling joint optimization of $\boldsymbol{\phi}$ and $\mathcal{A}$.

Deep constraint embedding. Deep constraint embedding methods independently trains $f_{\boldsymbol{\phi}}$ to learn a latent embedding representation $\mathcal{Z}$ that satisfies constraints $\mathcal{C}$, by minimizing the distances between positive pairs and maximizing those between negative pairs. Such representation learning operates without anchors and is driven by a generic anchor-free pairwise loss function, defined as:

$$\mathcal{L}(f_{\boldsymbol{\phi}}; \mathcal{C}) = \frac{1}{|\mathcal{C}|} \sum_{i=1}^{|\mathcal{C}|} \ell\left(f_{\boldsymbol{\phi}}(\boldsymbol{x}_{a_i}), f_{\boldsymbol{\phi}}(\boldsymbol{x}_{b_i}), y_i\right)$$

Optimizing this loss encourages $\mathcal{Z}$ to faithfully encode all pairwise constraints, thereby enabling CC to be treated as an unsupervised clustering task to determine $\mathcal{S}$.

In both DCC paradigms, the encoder $f_{\boldsymbol{\phi}}$ can be paired with a decoder $g_{\boldsymbol{\phi}'} : \mathcal{Z} \to \mathcal{X}$, parameterized by $\boldsymbol{\phi}'$, to form a deep autoencoder. This configuration leverages unconstrained instances in $\mathcal{X}$ to enrich the representation in $\mathcal{Z}$.

B Algorithms

In this appendix, we present our SpherePair CC algorithm and the PCA-based cluster number inference algorithm introduced in Section 3, as detailed in Algorithms 1 and 2.

Algorithm 1 SpherePair constraint clustering

I. Angular constraint embedding learning

Input: Training dataset $\mathcal{X}$, constraints $\mathcal{C}$, training epochs T, batch sizes $|\mathcal{B}_c|$ and $|\mathcal{B}_x|$, embedding dimension D, trade-off factor λ, parametrized autoencoder with encoder $f_{\boldsymbol{\phi}}$ and decoder $g_{\boldsymbol{\phi}'}$

Initialize the autoencoder parameters: $\boldsymbol{\phi}_0$ and $\boldsymbol{\phi}'_0$

for $t = 1, 2, \ldots, T$ **do**

Sample mini-batches $\mathcal{B}_c = \{(a_i, b_i, y_i)\}_{i=1}^{|\mathcal{B}_c|}$ from $\mathcal{C}$, and $\mathcal{B}_c = \{\boldsymbol{x}_j\}_{j=1}^{|\mathcal{B}_x|}$ from $\mathcal{X}$

Obtain latent embeddings: $\boldsymbol{z}_j = f_{\boldsymbol{\phi}}(\boldsymbol{x}_j), \forall \boldsymbol{x}_j \in \mathcal{X}$

Obtain reconstructions $\hat{\boldsymbol{x}}_j, \forall \boldsymbol{x}_j \in \mathcal{X}$ by Eq. 3

Compute gradients $\frac{\partial \mathcal{L}}{\partial \boldsymbol{\phi}}$ and $\frac{\partial \mathcal{L}}{\partial \boldsymbol{\phi}'}$ with $\mathcal{L}$ (in Eq. 4) via $\mathcal{L}_{\text{ang}}$ (in Eq. 1) and $\mathcal{L}_{\text{recon}}$ (in Eq. 2)

Update $\boldsymbol{\phi}$, $\boldsymbol{\phi}'$ using stochastic gradient descent (SGD): $\boldsymbol{\phi}_t, \boldsymbol{\phi}'_t \leftarrow \texttt{SGD}(\boldsymbol{\phi}_{t-1}, \boldsymbol{\phi}'_{t-1}, \frac{\partial \mathcal{L}}{\partial \boldsymbol{\phi}}, \frac{\partial \mathcal{L}}{\partial \boldsymbol{\phi}'})$

end for

Output: Optimal parameters, $\boldsymbol{\phi}^*$ and $\boldsymbol{\phi}'^*$

II. Clustering on spherical representations

Obtain the optimal representations $\mathcal{Z}^* = \{f_{\boldsymbol{\phi}^*}(\boldsymbol{x}_j)\}_{j=1}^{|\mathcal{X}|}$

Obtain the spherical representations $\mathcal{Z}_{\text{sphere}}$ by Eq. 5

Clustering with a chosen algorithm $\texttt{Clustering}(\cdot)$: $\mathcal{S} \leftarrow \texttt{Clustering}(\mathcal{Z}_{\text{sphere}})$, $\mathcal{S} = \{\mathcal{S}_k\}_{k=1}^{K}$

III. Prediction of unseen instances

Input: Test dataset $\tilde{\mathcal{X}}$ ($\tilde{\mathcal{X}} \cap \mathcal{X} = \varnothing$)

Compute latent centroids $\{\boldsymbol{\mu}_k\}_{k=1}^{K}$ of $\mathcal{S}$ based on $\mathcal{Z}_{\text{sphere}}$

for $\forall \tilde{\boldsymbol{x}} \in \tilde{\mathcal{X}}$ **do**

Obtain the spherical representation $\tilde{\boldsymbol{z}}_{\text{sphere}} = \text{Norm}\left(f_{\boldsymbol{\phi}^*}(\tilde{\boldsymbol{x}})\right)$

Assign $\tilde{\boldsymbol{z}}$ to cluster k^* where $k^* = \arg\min_k \theta_{\tilde{\boldsymbol{z}}_{\text{sphere}}, \boldsymbol{\mu}_k}$

end for

Algorithm 2 PCA-based cluster-number inference

Input: Spherical representations $\mathcal{Z}_{\text{sphere}} \subset \mathbb{R}^D$, training negative constraint subset $\mathcal{C}^- = \{(a_i, b_i, 0)\} \subseteq \mathcal{C}$ covering all true clusters, tail ratio hyperparameter ρ with $0 < \rho \ll 1$

Obtain subset of $\mathcal{Z}_{\text{sphere}}$ involved by $\mathcal{C}^-$: $\mathcal{Z}^-_{\text{sphere}} = \{\boldsymbol{z}_{a_i}, \boldsymbol{z}_{b_i} \mid (a_i, b_i, 0) \in \mathcal{C}^-\} \subseteq \mathcal{Z}_{\text{sphere}}$

Conduct PCA on $\mathcal{Z}^-_{\text{sphere}}$ to obtain all top-d subspace projections $\{\mathcal{Z}^{-,(d)}_{\text{pca}}\}_{d=1}^{D}$

for $d = 1$ to D **do**

Compute inter-cluster angles within $\mathcal{Z}^{-,(d)}_{\text{pca}}$, obtain $\Theta^{(d)} = \{\theta_{\tilde{\boldsymbol{z}}^{(d)}_{a_i}, \tilde{\boldsymbol{z}}^{(d)}_{b_i}} \mid (a_i, b_i, 0) \in \mathcal{C}^-\}$

Select the ρ-fraction smallest inter-cluster angles $\mathcal{M}^{(d)} = \texttt{Sort}_{\rho}(\Theta^{(d)})$, where $\texttt{Sort}_{\rho}(\Theta^{(d)})$ is an operation that sorts all members of the set $\Theta^{(d)}$, retaining only the $\lceil \rho \times |\mathcal{C}^-| \rceil$ minimal elements.

Compute tail average $\overline{\delta}_d = \text{mean}(\mathcal{M}^{(d)})$

end for

Identify the onset of the plateau in sequence $\{\overline{\delta}_d\}_{d=1}^{D}$ as d^*

Output: Estimated cluster number $\widehat{K} = d^* + 1$

C Proofs

In this appendix, we provide proofs for Proposition 4.1, Proposition 4.2, Corollary 4.3, Theorem 4.4, Corollary 4.5, Theorem 4.6, and Corollary 4.7.

C.1 Proof of Proposition 4.1

Proof. Recall that $\mathcal{L}_{\text{ang}}$ in Eq. 1 takes the form

$$\mathcal{L}_{\text{ang}} = -\frac{1}{|\mathcal{C}|} \sum_{i=1}^{|\mathcal{C}|} \Big(y_i \, \log \text{Sim}(a_i, b_i) \; + \; \big(1 - y_i\big) \, \log\big(1 - \text{Sim}(a_i, b_i)\big) \Big),$$

where

$$\mathrm{Sim}(a_i, b_i) \;=\; \frac{1}{2}\begin{cases}\cos\big(\theta_{\boldsymbol{z}_{a_i},\boldsymbol{z}_{b_i}}\big)+1, & \text{if } y_i=1,\\ \cos\big(\min\big(\omega\,\theta_{\boldsymbol{z}_{a_i},\boldsymbol{z}_{b_i}},\,\pi\big)\big)+1, & \text{if } y_i=0.\end{cases}$$

Suppose there is an $\mathcal{Z}^* = \{\boldsymbol{z}_j^*\}_{j=1}^{|\mathcal{X}|} \subset \mathbb{R}^D$ such that $\mathcal{L}_{\text{ang}} = 0$ thus satisfies all pairwise relationships $\{(a_i, b_i, y_i)\}$ derived from the ground truth partition $\mathcal{S}^*$ without causing conflicts. Consider two cases:

1. $\boldsymbol{x}_{a_i}, \boldsymbol{x}_{b_i} \in \mathcal{S}_k^*$ with $y_i = 1$. To achieve zero loss for the corresponding positive pair term in $\mathcal{L}_{\text{ang}}$, we must have $\mathrm{Sim}(a_i, b_i) = 1$, and this implies $\cos\big(\theta_{\boldsymbol{z}_{a_i}^*,\boldsymbol{z}_{b_i}^*}\big) \;=\; 1$. Hence,

$$\theta_{\boldsymbol{z}_{a_i}^*,\boldsymbol{z}_{b_i}^*} \;=\; 0,$$

   implying that any two instances in the same cluster $\mathcal{S}_k^*$ must lie at angle of zero degree in the optimal angular representation $\mathcal{Z}^*$.

2. $\boldsymbol{x}_{a_i} \in \mathcal{S}_k^*$, $\boldsymbol{x}_{b_i} \in \mathcal{S}_{k'}^*$ with $k \neq k'$ and $y_i = 0$. To achieve zero loss for this negative pair, we must have $\mathrm{Sim}(a_i, b_i) = 0$. By definition, this simplifies to $\cos\big(\min(\omega\,\theta_{\boldsymbol{z}_{a_i}^*,\boldsymbol{z}_{b_i}^*},\,\pi)\big) = -1$. Thus, $\min(\omega\,\theta_{\boldsymbol{z}_{a_i}^*,\boldsymbol{z}_{b_i}^*},\,\pi) = \pi$. Consequently, $\omega\,\theta_{\boldsymbol{z}_{a_i}^*,\boldsymbol{z}_{b_i}^*} \geq \pi$, leading to

$$\theta_{\boldsymbol{z}_{a_i}^*,\boldsymbol{z}_{b_i}^*} \;\geq\; \frac{\pi}{\omega}.$$

   Therefore, any two instances that belong to different ground-truth clusters must have their feature vectors separated by an angle of at least $\frac{\pi}{\omega}$ in $\mathcal{Z}^*$.

Since these two conditions hold for every pair derived from $\mathcal{S}^*$ and collectively yield zero loss, it follows that for each $\boldsymbol{x}_j \in \mathcal{S}_k^*$ and $\boldsymbol{x}_{j'} \in \mathcal{S}_{k'}^*$,

$$\theta_{\boldsymbol{z}_j^*,\boldsymbol{z}_{j'}^*}\begin{cases}=0, & \text{if } k=k',\\ \geq \frac{\pi}{\omega}, & \text{if } k\neq k'.\end{cases}$$

□

C.2 Proof of Proposition 4.2

Proof. We establish the proof of conditions (i) and (ii) by showing both necessity and sufficiency.

Necessity. Assume that there exists some $\mathcal{Z}^* = \{\boldsymbol{z}_j^*\}_{j=1}^{|\mathcal{X}|} \subset \mathbb{R}^D$ satisfying the conflict-free condition in Proposition 4.1 and that, for each constraint (a_i, b_i, y_i) from $\mathcal{S}^*$, the angle $\theta_{\boldsymbol{z}_{a_i}^*,\boldsymbol{z}_{b_i}^*}$ can be uniquely determined by (a_i, b_i, y_i). We show that all cross-cluster angles $\{\theta_{\boldsymbol{z}_j^*,\boldsymbol{z}_{j'}^*} \mid \boldsymbol{x}_j \in \mathcal{S}_k^*, \boldsymbol{x}_{j'} \in \mathcal{S}_{k'\neq k}^*\}$ must be the same.

By Proposition 4.1, any negative pair $(\boldsymbol{x}_j, \boldsymbol{x}_{j'}, 0)$ from different clusters satisfies $\theta_{\boldsymbol{z}_j^*,\boldsymbol{z}_{j'}^*} \geq \frac{\pi}{\omega}$. Suppose, for contradiction, that $\mathcal{Z}^*$ is not equidistant among clusters. Then there exist two distinct cross-cluster pairs whose angles differ; let

$$\theta_{\boldsymbol{z}_p^*,\boldsymbol{z}_q^*} \;=\; \min\Big\{\theta_{\boldsymbol{z}_j^*,\boldsymbol{z}_{j'}^*} : \boldsymbol{x}_j \in \mathcal{S}_k^*,\, \boldsymbol{x}_{j'} \in \mathcal{S}_{k'\neq k}^*\Big\}$$

be the smallest cross-cluster angle, and let $\theta_{\boldsymbol{z}_u^*,\boldsymbol{z}_v^*} > \theta_{\boldsymbol{z}_p^*,\boldsymbol{z}_q^*}$ be a strictly larger cross-cluster angle. By conflict-free condition, $\theta_{\boldsymbol{z}_p^*,\boldsymbol{z}_q^*} \geq \frac{\pi}{\omega}$, so certainly $\theta_{\boldsymbol{z}_u^*,\boldsymbol{z}_v^*} > \frac{\pi}{\omega}$. For the negative pair $(u, v, 0)$, the constraint alone enforces an angular separation at least $\frac{\pi}{\omega}$ but does not expand the angle to $\theta_{\boldsymbol{z}_u^*,\boldsymbol{z}_v^*}$. Hence the separation $\theta_{\boldsymbol{z}_u^*,\boldsymbol{z}_v^*}$ in $\mathcal{Z}^*$ cannot be uniquely determined by the negative constraint $(u, v, 0)$, contradicting the hypothesis. Therefore, it is necessary that $\mathcal{Z}^*$ be equidistant among clusters.

Sufficiency. Conversely, assume that $\mathcal{Z}^*$ is equidistant among clusters: every cross-cluster pair has the same angle $\theta^* > 0$. Then: (i) For each negative pair $(j, j', 0)$, we have $\theta_{\boldsymbol{z}_j^*, \boldsymbol{z}_{j'}^*} = \theta^*$. If we set $\omega = \omega^* = \frac{\pi}{\theta^*}$, then by Proposition 4.1, every cross-cluster angle satisfies a separation $\frac{\pi}{\omega} = \theta^*$. Hence each negative constraint $(j, j', 0)$ can uniquely determines each angular separation $\theta_{\boldsymbol{z}_j^*, \boldsymbol{z}_{j'}^*}$ in this equidistant $\mathcal{Z}^*$; (ii) As for any positive constraint $(j, j', 1)$, Proposition 4.1 forces each intra-cluster angle to be 0. Thus each positive constraint $(j, j', 1)$ can also uniquely determines each intra-cluster angle in $\mathcal{Z}^*$.

Hence, the necessity and sufficiency of conditions (i) and (ii) in Proposition 4.2 are shown.

□

C.3 Proof of Corollary 4.3

Proof. Recall from Eq. 1 that the average angular loss is

$$\mathcal{L}_{\text{ang}} = -\frac{1}{|\mathcal{C}|} \sum_{i=1}^{|\mathcal{C}|} \Big(y_i \, \log \text{Sim}(a_i, b_i) \; + \; (1 - y_i) \, \log\big(1 - \text{Sim}(a_i, b_i)\big) \Big),$$

where

$$\text{Sim}(a_i, b_i) = \frac{1}{2} \begin{cases} \cos\big(\theta_{z_{a_i}, z_{b_i}}\big) + 1, & \text{if } y_i = 1, \\ \cos\big(\min(\omega \, \theta_{z_{a_i}, z_{b_i}}, \pi)\big) + 1, & \text{if } y_i = 0. \end{cases}$$

Given $\mathcal{L}_{\text{ang}} \leq \varepsilon$ with $0 < \varepsilon \ll 1$.

Positive constraints. For any $(a_i, b_i, 1) \in \mathcal{C}$, its individual loss term is

$$\ell_i^+ \;=\; -\log\left(\frac{\cos(\theta_{z_{a_i}, z_{b_i}}) + 1}{2} \right).$$

Since the average loss is at most ε, each term satisfies $\ell_i^+ \leq |\mathcal{C}| \, \varepsilon$, hence

$$\frac{\cos(\theta_{z_{a_i}, z_{b_i}}) + 1}{2} \;\geq\; e^{-|\mathcal{C}|\varepsilon} \quad \Longrightarrow \quad \cos(\theta_{z_{a_i}, z_{b_i}}) \;\geq\; 2e^{-|\mathcal{C}|\varepsilon} - 1.$$

Therefore

$$0 \;\leq\; \theta_{z_{a_i}, z_{b_i}} \;\leq\; \arccos\big(2e^{-|\mathcal{C}|\varepsilon} - 1\big) \;=:\; \Delta^+(\varepsilon).$$

For small $|\mathcal{C}|\varepsilon$, a Taylor expansion of $e^{-|\mathcal{C}|\varepsilon}$ and arccos gives $\Delta^+(\varepsilon) \approx 2\sqrt{|\mathcal{C}|\varepsilon}$.

Negative constraints. For any $(a_i, b_i, 0) \in \mathcal{C}$ with $\theta_{z_{a_i}, z_{b_i}} < \frac{\pi}{\omega}$, the loss term is

$$\ell_i^- \;=\; -\log\left(1 - \frac{\cos(\omega \, \theta_{z_{a_i}, z_{b_i}}) + 1}{2} \right).$$

The bound $\ell_i^- \leq |\mathcal{C}| \, \varepsilon$ implies

$$1 - \frac{\cos(\omega \, \theta_{z_{a_i}, z_{b_i}}) + 1}{2} \;\geq\; e^{-|\mathcal{C}|\varepsilon} \quad \Longrightarrow \quad \cos(\omega \, \theta_{z_{a_i}, z_{b_i}}) \;\leq\; 1 - 2e^{-|\mathcal{C}|\varepsilon}.$$

Since $\theta_{z_{a_i}, z_{b_i}}$ is close to $\frac{\pi}{\omega}$, we write

$$\omega \, \theta_{z_{a_i}, z_{b_i}} \;\geq\; \pi - \arccos\big(1 - 2e^{-|\mathcal{C}|\varepsilon}\big),$$

which rearranges to

$$0 \;\leq\; \frac{\pi}{\omega} - \theta_{z_{a_i}, z_{b_i}} \;\leq\; \frac{1}{\omega} \arccos\big(1 - 2e^{-|\mathcal{C}|\varepsilon}\big) \;=:\; \Delta^-(\varepsilon).$$

Again, for small $|\mathcal{C}|\varepsilon$, Taylor expansion yields $\Delta^-(\varepsilon) \approx \frac{2\sqrt{|\mathcal{C}|\varepsilon}}{\omega}$.

Combining the two cases gives the desired bounds in Corollary 4.3.

□

C.4 Proof of Theorem 4.4

Proof. We reformulate the statement as a geometric problem of arranging K distinct unit vectors in $\mathbb{R}^D$, ensuring that the angle between any two distinct vectors remains constant. This arrangement satisfies the "equidistant clusters" condition described in Proposition 4.2.

Let us denote these K vectors as

$$\{\boldsymbol{u}_1, \boldsymbol{u}_2, \ldots, \boldsymbol{u}_K\} \subset \mathbb{R}^D, \quad \|\boldsymbol{u}_k\| = 1,$$

with the property that

$$\theta_{\boldsymbol{u}_k,\, \boldsymbol{u}_{k'}} = \theta^* \quad \text{for all } 1 \le k \ne k' \le K.$$

We show that such a uniform arrangement exists if and only if $D \ge K - 1$, and derive the bounds of θ^* for different D.

Case (i): $D < K - 1$. Assume, for contradiction, that there exist K unit vectors $\{\boldsymbol{u}_1, \ldots, \boldsymbol{u}_K\} \subset \mathbb{R}^D$ with $D < K - 1$ and a common angle $\theta^* > 0$. Let $c = \cos\theta^*$. Consider the $K \times K$ Gram matrix G of these vectors, where

$$G_{ij} = \boldsymbol{u}_i \cdot \boldsymbol{u}_j = \begin{cases} 1, & \text{if } i = j, \\ c, & \text{if } i \ne j. \end{cases}$$

We may rewrite G as:

$$G = (1 - c)\, I + c\, J,$$

where I is the $K \times K$ identity matrix and J is the $K \times K$ matrix of all ones.

To identify the eigenvalues of G, note that J has one eigenvalue K (with eigenvector $\mathbf{1} = (1, \ldots, 1)^\top$) and $K - 1$ eigenvalues equal to 0. Hence G inherits:

- One eigenvalue $(1 - c) \cdot 1 + c \cdot K = 1 + c\,(K - 1)$, associated with $\mathbf{1}$.
- $K - 1$ eigenvalues $(1 - c) \cdot 1 + c \cdot 0 = 1 - c$, corresponding to any vector orthogonal to $\mathbf{1}$.

Since $c = \cos\theta^* < 1$ (ensuring the K vectors are pairwise distinct), it follows that $1 - c > 0$. Thus G has at least $K - 1$ strictly positive eigenvalues, which implies

$$\operatorname{rank}(G) \ge K - 1.$$

On the other hand, because all $\boldsymbol{u}_i$ lie in $\mathbb{R}^D$, the dimension of the subspace spanned by them is at most D, so $\operatorname{rank}(G) \le D$. Therefore we must have

$$K - 1 \le \operatorname{rank}(G) \le D,$$

contradicting $D < K - 1$. Hence, there can be no such K unit vectors in $\mathbb{R}^D$ whose pairwise angles are all equal, and consequently, no valid ω (i.e. no angle θ^*) realizes the equidistant configuration when $D < K - 1$.

Case (ii): $D = K - 1$. In a space of dimension exactly $K - 1$, it is both necessary and sufficient that the K vectors form a regular simplex. The same Gram matrix argument above now forces $\operatorname{rank}(G) = K - 1$, hence the unique way for K vectors to remain all equiangular is

$$1 + c(K - 1) = 0 \quad \Longrightarrow \quad c = -\frac{1}{K - 1}, \quad \theta^* = \arccos\!\Big(-\frac{1}{K - 1}\Big).$$

In terms of Proposition 4.2, the uniform cross-cluster angle is θ^*, so $\omega = \omega^*$ must satisfy

$$\theta^* = \frac{\pi}{\omega^*} \quad \Longrightarrow \quad \omega^* = \frac{\pi}{\theta^*} = \frac{\pi}{\arccos\!\big(-\frac{1}{K-1}\big)}.$$

This ω^* is unique for $D = K - 1$.

Case (iii): $D \geq K$**.** Suppose we wish to place K unit vectors $\{\boldsymbol{u}_1, \ldots, \boldsymbol{u}_K\} \subset \mathbb{R}^D$ so that every pair of distinct vectors has a common angle $\theta^* > 0$. Let $c = \cos\theta^*$, and form the corresponding $K \times K$ Gram matrix as before. As in the preceding cases, we may rewrite G as $G = (1-c)\, I + c\, J$, where I is the $K \times K$ identity and J is the $K \times K$ all-ones matrix. Its eigenvalues are

$$\lambda_1 = 1 + c\,(K-1), \quad \lambda_2 = \cdots = \lambda_K = 1 - c.$$

For G to be a valid Gram matrix of real vectors, all eigenvalues must be nonnegative:

$$1 - c \;\geq\; 0 \quad \Longrightarrow \quad c \;\leq\; 1, \qquad 1 + c\,(K-1) \;\geq\; 0 \quad \Longrightarrow \quad c \;\geq\; -\frac{1}{K-1}.$$

Since $c < 1$ when the vectors are mutually distinct, we combine these to conclude

$$-\frac{1}{K-1} \;\leq\; c \;<\; 1 \quad \Longrightarrow \quad 0 \;<\; \theta^* \;\leq\; \arccos\!\Big(-\frac{1}{K-1}\Big).$$

Thus, whenever $D \geq K$, any common angle θ^* up to $\arccos(-\frac{1}{K-1})$ is feasible. Equivalently, in terms of $\omega = \frac{\pi}{\theta^*}$, we obtain

$$\omega \;=\; \frac{\pi}{\theta^*} \;\;\geq\;\; \frac{\pi}{\arccos\!\big(-\frac{1}{K-1}\big)}.$$

Geometrically, this reflects the fact that we only need a subspace of dimension $K-1$ to place K equiangular vectors, and having additional dimensions ($D \geq K$) does not tighten the required angle—it simply allows the same arrangement (or more flexible ones) to fit in higher-dimensional ambient space. Hence, any ω above the threshold $\pi \big/ \arccos(-\frac{1}{K-1})$ remains valid when $D \geq K$.

Combining all three cases leads to the desired conclusions:

(i) When $D < K-1$, such a valid ω does not exist.

(ii) When $D = K-1$, the unique valid ω is $\pi \big/ \arccos(-\frac{1}{K-1})$.

(iii) When $D \geq K$, the range of valid ω values is relaxed to $\omega \geq \pi \big/ \arccos(-\frac{1}{K-1})$.

□

C.5 Proof of Corollary 4.5

Proof. Given $D \geq K$, by Theorem 4.4, the admissible set is $\omega \geq \pi \big/ \arccos\!\big(-\frac{1}{K-1}\big)$. Hence the minimal admissible value is

$$\omega^*_{\min}(K) \;=\; \frac{\pi}{\arccos\!\big(-\frac{1}{K-1}\big)}.$$

We now establish the stated properties:

Bounds. For $K = 2$, $\arccos(-1) = \pi$, thus $\omega^*_{\min}(2) = \pi/\pi = 1$. For every $K > 2$, $-\frac{1}{K-1} \in (-1, 0)$, hence

$$\arccos\!\Big(-\tfrac{1}{K-1}\Big) \in \Big(\tfrac{\pi}{2},\, \pi\Big), \quad \Rightarrow \quad 1 \;<\; \omega^*_{\min}(K) \;<\; 2.$$

Therefore $1 \leq \omega^*_{\min}(K) < 2$ for all $K > 1$.

Monotonicity. If $2 \leq K_1 < K_2$, then $-\frac{1}{K_1-1} < -\frac{1}{K_2-1}$. Since $\arccos(\cdot)$ is strictly decreasing on $[-1, 1]$,

$$\arccos\!\Big(-\tfrac{1}{K_1-1}\Big) \;>\; \arccos\!\Big(-\tfrac{1}{K_2-1}\Big),$$

and thus

$$\omega^*_{\min}(K_1) = \frac{\pi}{\arccos(-\frac{1}{K_1-1})} \;<\; \frac{\pi}{\arccos(-\frac{1}{K_2-1})} = \omega^*_{\min}(K_2).$$

Hence $\omega^*_{\min}(K)$ is strictly increasing in K.

Limit. As $K \to \infty$, we have $-\frac{1}{K-1} \to 0$, so by continuity

$$\arccos\Big(-\tfrac{1}{K-1}\Big) \to \arccos(0) = \tfrac{\pi}{2}, \qquad \Rightarrow \qquad \omega^*_{\min}(K) = \frac{\pi}{\arccos(-\frac{1}{K-1})} \to 2.$$

Combining the above completes the proof.

□

C.6 Proof of Theorem 4.6

Proof. For notation, let $\{\boldsymbol{u}_k\}_{k=1}^K \subset \mathbb{R}^D$ be the K cluster-representative unit vectors on the sphere (i.e., the common location of each cluster in $\mathcal{Z}_{\text{sphere}}$), let $p_k := |\mathcal{S}^*_k|/|\mathcal{X}| > 0$ with $\sum_{k=1}^K p_k = 1$ be the cluster frequencies, and set the sample mean

$$\boldsymbol{m} := \sum_{k=1}^K p_k \, \boldsymbol{u}_k.$$

After standard centering by $\boldsymbol{m}$, the k-th cluster maps to the common centered vector

$$\boldsymbol{v}_k := \boldsymbol{u}_k - \boldsymbol{m} \quad (k = 1, \dots, K).$$

For an optimal spherical embedding $\mathcal{Z}_{\text{sphere}}$, all instances in cluster $\mathcal{S}^*_k$ coincide at $\boldsymbol{u}_k$, therefore after centering they all coincide at $\boldsymbol{v}_k$. Denote

$$U := \text{span}\{\boldsymbol{v}_1, \dots, \boldsymbol{v}_K\} \subset \mathbb{R}^D.$$

Step 1 (Centered data lie in a $(K-1)$–dimensional subspace and $\text{Im}(\Sigma) = U$). Since $\sum_{k=1}^K p_k \boldsymbol{v}_k = \sum_k p_k(\boldsymbol{u}_k - \boldsymbol{m}) = \boldsymbol{0}$, the family $\{\boldsymbol{v}_k\}$ is linearly dependent and hence $\dim U \le K-1$. On the other hand, $\{\boldsymbol{u}_k\}$ are the K affinely independent vertices of a regular simplex, whose affine hull has dimension $K-1$; translating this affine hull by $-\boldsymbol{m}$ yields the linear subspace U through the origin. Hence $\dim U = K-1$.

Let the covariance after centering be

$$\Sigma = \frac{1}{|\mathcal{X}|} \sum_{j=1}^{|\mathcal{X}|} (\text{Norm}(\boldsymbol{z}^*_j) - \boldsymbol{m})(\text{Norm}(\boldsymbol{z}^*_j) - \boldsymbol{m})^\top = \sum_{k=1}^K p_k \, \boldsymbol{v}_k \boldsymbol{v}_k^\top.$$

For any $\boldsymbol{w} \perp U$ we have $\boldsymbol{v}_k^\top \boldsymbol{w} = 0$ and thus $\Sigma \boldsymbol{w} = \boldsymbol{0}$, so $\text{Im}(\Sigma) \subseteq U$, where $\text{Im}(\Sigma)$ is the column space of Σ . Conversely, for any $\boldsymbol{u} \in U \setminus \{\boldsymbol{0}\}$, since $\{\boldsymbol{v}_k\}$ span U, there exists k with $\boldsymbol{v}_k^\top \boldsymbol{u} \neq 0$, and therefore

$$\boldsymbol{u}^\top \Sigma \boldsymbol{u} = \sum_{k=1}^K p_k \, (\boldsymbol{v}_k^\top \boldsymbol{u})^2 > 0.$$

Thus Σ is positive definite on U and vanishes on $U^\perp$, which implies $\text{Im}(\Sigma) = U$. This conclusion does not depend on the specific values of $\{p_k\}$ beyond $p_k > 0$.

Step 2 (For $d \ge K-1$, PCA projection preserves all pairwise angles). Let W_d be the d–dimensional PCA subspace spanned by the top d eigenvectors of Σ. By Step 1, Σ is positive definite on U and zero on $U^\perp$, hence its nonzero eigenspace is exactly U. Therefore, for any $d \ge K-1 = \dim U$,

$$U \subseteq W_d \subseteq U \oplus U^\perp = \mathbb{R}^D,$$

where $\oplus$ denotes the direct sum of subspaces. $\{\text{Norm}(\boldsymbol{z}^*_j) - \boldsymbol{m}\}$ lie in U and each $\text{Norm}(\boldsymbol{z}^*_j) - \boldsymbol{m}$ equals some $\boldsymbol{v}_k$. Consequently, for any $\boldsymbol{v}_k$, the orthogonal projection onto W_d satisfies $P_d \boldsymbol{v}_k = \boldsymbol{v}_k$ because $\boldsymbol{v}_k \in U \subseteq W_d$. PCA then applies an orthogonal change of basis inside W_d (and, when $d = D$, an orthogonal transform in the full space). Orthogonal transforms preserve inner products, norms, and hence angles between any nonzero pair of vectors. Therefore, for any pair (j, j') with nonzero projections and any $d, d' \ge K-1$,

$$\theta_{\tilde{\boldsymbol{z}}_j^{(d)}, \tilde{\boldsymbol{z}}_{j'}^{(d)}} = \theta_{\tilde{\boldsymbol{z}}_j^{(d')}, \tilde{\boldsymbol{z}}_{j'}^{(d')}},$$

proving part (i).

Step 3 (For $d < K-1$, global invariance cannot hold). We work in the nondegenerate regime where the pairs compared satisfy $\tilde{\boldsymbol{z}}_j^{(d)} \neq \mathbf{0} \neq \tilde{\boldsymbol{z}}_{j'}^{(d)}$, i.e., $P_d \boldsymbol{v}_k \neq \mathbf{0}$ for the involved clusters. Since $\dim U = K-1 > d$, one can choose indices $\mathcal{I} = \{k_1, \ldots, k_{d+1}\}$ so that $P_d \boldsymbol{v}_{k_\tau} \neq \mathbf{0}$ for all τ, and $\{\boldsymbol{v}_{k_\tau}\}_{\tau=1}^{d+1}$ are linearly independent in U. Form the matrix $X = [\, \boldsymbol{v}_{k_1} \ \cdots \ \boldsymbol{v}_{k_{d+1}} \,] \in \mathbb{R}^{D \times (d+1)}$ with Gram matrix $G := X^\top X \in \mathbb{R}^{(d+1)\times(d+1)}$, which is positive definite (hence $\operatorname{rank}(G) = d+1$). Let $\widetilde{G} := (P_d X)^\top (P_d X) = X^\top P_d X$, whose rank is at most d because $P_d X$ has columns in the d–dimensional subspace W_d.

Assume, for contradiction, that the angle invariance of part (i) holds for *all* admissible pairs among the set $\{\boldsymbol{v}_{k_\tau}\}_{\tau=1}^{d+1}$, i.e., for every $\tau \neq \tau'$ with $P_d \boldsymbol{v}_{k_\tau}, P_d \boldsymbol{v}_{k_{\tau'}} \neq \mathbf{0}$,

$$\frac{\langle P_d \boldsymbol{v}_{k_\tau}, P_d \boldsymbol{v}_{k_{\tau'}} \rangle}{\|P_d \boldsymbol{v}_{k_\tau}\| \, \|P_d \boldsymbol{v}_{k_{\tau'}}\|} = \frac{\langle \boldsymbol{v}_{k_\tau}, \boldsymbol{v}_{k_{\tau'}} \rangle}{\|\boldsymbol{v}_{k_\tau}\| \, \|\boldsymbol{v}_{k_{\tau'}}\|}.$$

Setting $h_\tau := \|P_d \boldsymbol{v}_{k_\tau}\| > 0$ and $H := \operatorname{diag}(h_1, \ldots, h_{d+1})$, the above identities imply

$$\widetilde{G}_{\tau\tau'} = \langle P_d \boldsymbol{v}_{k_\tau}, P_d \boldsymbol{v}_{k_{\tau'}} \rangle = h_\tau h_{\tau'} \langle \boldsymbol{v}_{k_\tau}, \boldsymbol{v}_{k_{\tau'}} \rangle, \quad \text{hence} \quad \widetilde{G} = H\, G\, H.$$

Since H is invertible, $\operatorname{rank}(\widetilde{G}) = \operatorname{rank}(G) = d+1$, which contradicts $\operatorname{rank}(\widetilde{G}) \leq d$. Therefore, when $d < K-1$, the invariance across d in (i) cannot hold for all admissible pairs, proving part (ii).

□

C.7 Proof of Corollary 4.7

Proof. We follow the notation in Theorem 4.6: let $\{\boldsymbol{u}_k\}_{k=1}^K \subset \mathbb{R}^D$ be the K cluster-representative unit vectors on the sphere y, let $p_k = |\mathcal{S}_k^*|/|\mathcal{X}| > 0$ with $\sum_{k=1}^K p_k = 1$ be the cluster frequencies, and let the sample mean be $\boldsymbol{m} := \sum_{k=1}^K p_k \, \boldsymbol{u}_k$. After centering by $\boldsymbol{m}$, each cluster $\mathcal{S}_k^*$ collapses to a common vector $\boldsymbol{v}_k := \boldsymbol{u}_k - \boldsymbol{m}$ where $k = 1, \ldots, K$. Let $U = \operatorname{span}\{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_K\} \subset \mathbb{R}^D$.

From Appendix C.6, we know that $\dim U = K-1$ and the image space of the covariance matrix after centering is exactly U. Hence when $d \geq K-1$, the PCA subspace W_d contains U and therefore $P_d \boldsymbol{v}_k = \boldsymbol{v}_k$. Subsequent orthogonal transformations preserve all pairwise angles.

(i) Case $K = 2$. Here $\dim U = 1$. Since $\boldsymbol{m} = p_1 \boldsymbol{u}_1 + p_2 \boldsymbol{u}_2$ with $p_2 = 1 - p_1$, we obtain

$$\boldsymbol{v}_1 = (1-p_1)(\boldsymbol{u}_1 - \boldsymbol{u}_2), \qquad \boldsymbol{v}_2 = -p_1(\boldsymbol{u}_1 - \boldsymbol{u}_2).$$

Thus $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ are collinear in opposite directions, so their angle is always π. Any PCA projection with $d \geq 1 = K-1$ only applies an isometry within this line (and possibly an orthogonal transform in the full space), leaving the angle unchanged. Hence $\delta_d = \pi$ for all $d \geq 1$.

(ii) Case $K > 2$: proving $\delta_1 = 0$ and invariance for $d \geq K-1$.

1. $\delta_1 = 0$: In the one-dimensional PCA projection, all vectors $\boldsymbol{v}_k$ are mapped to a single line. Since $K > 2$ and each $p_k > 0$, there must exist two distinct clusters whose projections lie on the same side and are nonzero, yielding an angle of 0. Hence $\delta_1 = 0$.
2. *Invariance for $d \geq K-1$*: By Theorem 4.6, for any admissible pair (j, j'), when $d \geq K-1$ the angle $\theta_{\tilde{\boldsymbol{z}}_j^{(d)}, \tilde{\boldsymbol{z}}_{j'}^{(d)}}$ equals its counterpart at $d = K-1$ (or D). Therefore the minimal cross-cluster angle δ_d is constant for all $d \geq K-1$; denote this constant by $\delta_\star$.
3. *Upper bound of $\delta_\star$*: Let $c = \cos\theta^*$ be the common inner product among $\{\boldsymbol{u}_k\}$. The Gram matrix of $U = [\, \boldsymbol{u}_1 \ \cdots \ \boldsymbol{u}_K \,]$ is $G = (1-c)I + c\,\mathbf{1}\mathbf{1}^\top$. For any $k \neq k'$, consider $\boldsymbol{v}_k = \boldsymbol{u}_k - \boldsymbol{m}$ and $\boldsymbol{v}_{k'} = \boldsymbol{u}_{k'} - \boldsymbol{m}$. Derivable from the Gram structure, angle $\theta_{\boldsymbol{v}_k, \boldsymbol{v}_{k'}}$ depends only on $\{p_i\}$: letting $S = \sum_{i=1}^K p_i^2$, one obtains

$$\cos\theta_{\boldsymbol{v}_k, \boldsymbol{v}_{k'}} = \frac{S - p_k - p_{k'}}{\sqrt{(1 - 2p_k + S)(1 - 2p_{k'} + S)}}. \tag{6}$$

When $p_1 = \cdots = p_K = \frac{1}{K}$, $S = \frac{1}{K}$, and (6) gives $\theta_{\boldsymbol{v}_k, \boldsymbol{v}_{k'}} = \arccos(-\frac{1}{K-1})$. All cross-cluster angles coincide, so $\delta_\star = \arccos(-\frac{1}{K-1})$, yielding the upper bound. This bound is attained when clusters are balanced.

4. *Lower bound of* $\delta_\star$: For a given pair (k, k'), write $r = 1 - p_k - p_{k'} = \sum_{i \neq k,k'} p_i > 0$ and $s = \sum_{i \neq k,k'} p_i^2$, so $0 < s \leq r^2$. Equation (6) can be rewritten as a function $f(s)$. One checks that $f(s)$ is nondecreasing in s on the relevant interval. Hence the maximal value of $\cos\theta_{\boldsymbol{v}_k, \boldsymbol{v}_{k'}}$ (i.e., the minimal angle) occurs at $s = r^2$, which corresponds to the case when all remaining probability mass r concentrates in a single cluster. In this regime, one can show $\cos\theta_{\boldsymbol{v}_k, \boldsymbol{v}_{k'}} \leq \frac{1}{2}$, thus $\cos\theta_{\boldsymbol{v}_k, \boldsymbol{v}_{k'}} \geq \arccos(\frac{1}{2}) = \frac{\pi}{3}$. Equality is never attained when all $p_k > 0$, but as some $p_\ell \to 1$ (the others tending to 0), $\theta_{\boldsymbol{v}_k, \boldsymbol{v}_{k'}}$ approaches $\frac{\pi}{3}$. Therefore $\delta_\star > \frac{\pi}{3}$, with the infimum $\frac{\pi}{3}$ approached when one cluster dominates.

Thus, for $K = 2$, we proved $\delta_d = \pi$ for all $d \geq 1$. For $K > 2$, we established $\delta_1 = 0$; for $d \geq K - 1$, δ_d takes a constant value $\delta_\star$; and

$$\delta_\star \in (\tfrac{\pi}{3},\ \arccos(-\tfrac{1}{K-1})],$$

with the upper bound $\arccos(-\frac{1}{K-1})$ attained in the balanced case and the lower bound $\frac{\pi}{3}$ approached in the highly imbalanced case.

□

D Visualization of SpherePair representation learning in angular space

To demonstrate how SpherePair learns equidistant spherical embeddings for different numbers of clusters in angular space, we perform a 3D visualisation experiment using subsets of the Reuters dataset [51], following the preprocessing steps outlined in the work [50]. Specifically, the original training partition consists of 10,000 instances from four root categories: *Corporate/Industrial* (CCAT), *Economics* (ECAT), *Government/Social* (GCAT), and *Markets* (MCAT). Their respective instance counts in this training portion are 4,066 (CCAT), 2,888 (ECAT), 2,202 (GCAT), and 844 (MCAT). We form three data subsets by taking the first 2, 3, and all 4 categories, thus yielding scenarios with $K = 2, 3, 4$. For each subset, we randomly sample 10k instance pairs and derive pairwise constraints based on whether the two instances belong to the same ground-truth category. Naturally, the constraint distribution varies across different K due to the differing subset compositions.

Using the three data subsets and their respective 10k constraints, we train a SpherePair autoencoder comprising a fully connected encoder with hidden layers of sizes 500, 500, and 2000 (mirrored by a symmetric decoder) and a 3-dimensional embedding layer ($D = 3$). As guaranteed by Theorem 4.4, we set the negative-zone factor $\omega = \pi / \arccos(-\frac{1}{K-1})$ as it is always valid when $D \geq K - 1$, to ensure that the learned embeddings can form an equidistant arrangement on the unit sphere.

Fig. 7 illustrates the evolution of the resulting spherical representations $\mathcal{Z}_{\text{sphere}}$ over the course of training. As the training progresses, the data points gradually separate on the 3D sphere, and the final embeddings form nearly regular simplex configurations in the $(K-1)$-dimensional subspace of $\mathbb{R}^3$, consistent with our theoretical insights in Sect. 4. Worth noting, despite the imbalance in instance counts among different categories, the formation of equidistant spherical clusters is not contingent on ground truth clusters being of equal size. Thus, this 3D visualisation vividly illustrates and validates the theoretical insights into our proposed SpherePair representation learning framework.

E Details of experimental settings

In this appendix, we provide a detailed description of our experimental settings to ensure that all necessary information is included and our results are fully replicable.

E.1 Datasets

We provide detailed descriptions of the benchmark datasets employed for evaluation in experiments.

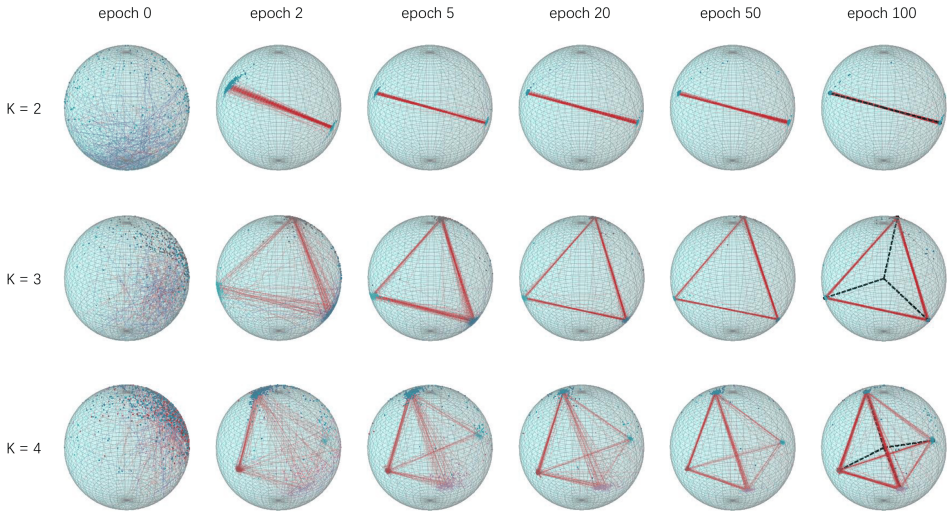


Figure 7: Evolution of SpherePair embeddings on a 3D unit sphere for datasets with $K = 2$, 3, and 4 clusters. Each row corresponds to one of the three subsets derived from the Reuters dataset, and each column shows a snapshot of the embeddings at a particular epoch. Different marker colors denote different ground-truth categories, and dashed lines represent randomly sampled pairwise constraints (blue for positive, red for negative). In the final column of each row, we illustrate the converged embeddings and highlight, in black dashed lines, the connections from the origin to each cluster centroid obtained by running K-means on $\mathcal{Z}_{\text{sphere}}$.

- **CIFAR-10** [2] [45]: Consists of 60,000 real-world 32×32 color images spanning 10 classes, each class containing 6,000 images. CIFAR-10 has been widely used as a benchmark in DCC research [11, 16, 14].
- **CIFAR-100-20** [3] [45]: A more complex extension of CIFAR-10, also having 60,000 real-world 32×32 color images. CIFAR-100 originally contains 100 fine-grained classes, which can be grouped into 20 superclasses. In our experiments, we use these 20 superclasses (each containing 3,000 images) as the ground truth rather than the 100 classes.
- **FashionMNIST** [4] [46]: Contains 70,000 grayscale images of Zalando fashion products (10 categories) with a size of 28×28 each. It is pre-split into 60,000 training images (6,000 per class) and 10,000 test images (1,000 per class). FashionMNIST also serves as a benchmark for DCC evaluation such as the methods [16, 12, 13].
- **ImageNet-10** [47]: A subset of ImageNet [5] [52] with 10 classes, each containing 1,300 randomly selected color images (13,000 in total). The chosen classes are `n02056570`, `n02085936`, `n02128757`, `n02690373`, `n02692877`, `n03095699`, `n04254680`, `n04285008`, `n04467665`, `n07747607`. This ImageNet-10 is commonly used as a benchmark in both DCC [14] and unsupervised deep clustering [47, 53].
- **MNIST** [6] [48]: Contains 70,000 grayscale images of handwritten digits (0–9), each of size 28×28. It is pre-split into 60,000 training images (6,000 per digit) and 10,000 test images (1,000 per digit). MNIST serves as a benchmark in DCC evaluation such as the methods [11, 16, 12, 13].
- **Reuters** [51]: A subset of the RCV1[7] corpus (a large-scale newswire collection of 804,414 English stories). We use the preprocessed version[8] from [50], which provides tf–idf features

[2] CIFAR-10 webpage: https://www.cs.toronto.edu/~kriz/cifar.html
[3] CIFAR-100-20 webpage: https://www.cs.toronto.edu/~kriz/cifar.html
[4] FashionMNIST repository: https://github.com/zalandoresearch/fashion-mnist
[5] ImageNet webpage: https://image-net.org/
[6] MNIST webpage: http://yann.lecun.com/exdb/mnist/
[7] RCV1 webpage: https://trec.nist.gov/data/reuters/reuters.html
[8] Preprocessed Reuters repository: https://github.com/piiswrong/dec

on the 2,000 most frequent words across documents sampled from four root categories: *Corporate/Industrial* (CCAT), *Economics* (ECAT), *Government/Social* (GCAT), and *Markets* (MCAT). The dataset is pre-split into 10,000 training samples and 2,000 test samples, with the four categories containing 4,066/2,888/2,202/844 training and 774/582/471/173 test samples, respectively. We treat these four categories as the ground-truth clusters in our experiments. Reuters has been widely adopted as a benchmark in DCC research [16, 12, 13].

- **STL-10** [9] [49]: Comprises 13,000 96×96 color images from 10 classes, each class having 1,300 images. STL-10 is adopted by various DCC works [11, 13, 14].
- **RCV1-10** [10]: Another subset of RCV1[7] with highly imbalanced class distribution. We randomly selected 10 categories (C14, C18, C313, C42, E21, E311, GDEF, GODD, GWELF, M13) from the 103 available topics and removed documents carrying multiple labels within this set, yielding 177,669 single-label articles with the following class counts: 6,634 (C14), 51,145 (C18), 1,042 (C313), 10,954 (C42), 40,950 (E21), 1,679 (E311), 8,492 (GDEF), 2,743 (GODD), 903 (GWELF), and 53,127 (M13). We treat these 10 categories as the ground-truth clusters in our experiments. Tf–idf features were computed on the 2,000 most frequent word stems, yielding the RCV1-10 subset as a realistic benchmark for imbalanced constrained clustering.

E.2 Baselines

We provide the detailed information on the state-of-the-art deep constrained clustering baselines used in our comparative study.

VanillaDCC [21]: A straightforward end-to-end anchor-based DCC model grounded in the Meta Classification Likelihood (MCL) loss:

$$\mathcal{L}_{\text{MCL}} = -\frac{1}{|\mathcal{C}|}\sum_{i=1}^{|\mathcal{C}|}\Big(y_i\,\log P_{\text{co}_i} + (1-y_i)\,\log\big(1-P_{\text{co}_i}\big)\Big),$$

where $P_{\text{co}_i} = \boldsymbol{q}_{a_i}\boldsymbol{q}_{b_i}^\top \in [0,1]$ is a pairwise co-occurrence likelihood integrated into a logistic loss. Here, $\boldsymbol{q}_{a_i}$ and $\boldsymbol{q}_{b_i}$ are the soft assignments of constrained data pair $(\boldsymbol{x}_{a_i}, \boldsymbol{x}_{b_i})$ to K clusters. By minimizing this loss, VanillaDCC learns a cluster assignment matrix $\mathcal{Q} = \{\boldsymbol{q}_j\}_{j=1}^N$ that respects the constraint set $\mathcal{C}$.

VolMaxDCC [14]: An extension of VanillaDCC that modifies the MCL-based loss to handle confused memberships and incorporates an additional geometric regularization term controlled by a trade-off factor λ. Specifically, VolMaxDCC introduces a matrix B (treated as an optimization variable and derived from a confusion matrix) into the pairwise co-occurrence likelihood and adds a volume maximization regularization term:

$$\mathcal{L}_{\text{VolMax}} = -\frac{1}{|\mathcal{C}|}\sum_{i=1}^{|\mathcal{C}|}\Big(y_i \log P'_{\text{co}_i} + (1-y_i)\log\big(1-P'_{\text{co}_i}\big)\Big) - \lambda \log\det\big(\mathcal{Q}^\top\mathcal{Q}\big).$$

Here $P'_{\text{co}_i} = \boldsymbol{q}_{a_i} B\, \boldsymbol{q}_{b_i}^\top \in [0,1]$ is the modified pairwise co-occurrence likelihood, and $\mathcal{Q}$ is treated as the cluster assignment matrix. The first term in $\mathcal{L}_{\text{VolMax}}$ adjusts the similarity measure to account for membership confusion, while the second term serves as a geometric regularization that encourages maximization of the volume of the Gram matrix $\mathcal{Q}^\top\mathcal{Q}$, thereby enhancing the separability and distinguishability of clusters. Both B and $\mathcal{Q}$ are optimized during training. The choice of the trade-off factor λ requires tuning to balance these effects. Overall, $\mathcal{L}_{\text{VolMax}}$ enables VolMaxDCC to handle noisy constraints caused by annotation confusion while promoting well-separated and distinguishable cluster assignments.

CIDEC [12]: An end-to-end method that balances unsupervised representation learning and constrained clustering within a multi-task joint optimization framework. Specifically, CIDEC first encodes data into a latent space via a deep autoencoder and initializes K learnable cluster anchors

[9] STL-10 webpage: https://cs.stanford.edu/~acoates/stl10/

[10] We provide the preprocessed RCV1-10 subset: https://github.com/spherepaircc/SpherePairCC/tree/main

using K-means, where K corresponds to the known number of ground-truth classes. It then alternates between supervised and unsupervised training phases during each epoch. In each iteration, the model jointly refines the autoencoder parameters and cluster assignments by combining the the following learning objectives to form a multi-objective loss function:

(i) The deep embedding clustering objective from [50, 40], which minimizes the KL divergence

$$\mathcal{L}_{\text{DEC}} = \sum_{j=1}^{|\mathcal{X}|} \sum_{k=1}^{K} p_{jk} \log \frac{p_{jk}}{q_{jk}},$$

between the soft assignment distributions $\boldsymbol{q}_j = (q_{j1}, q_{j2}, \ldots, q_{jK})$ and a target distribution $\boldsymbol{p}_j = (p_{j1}, p_{j2}, \ldots, p_{jK})$. Specifically, for each sample j, the target distribution is calculated as

$$p_{jk} = \frac{q_{jk}^2 / f_k}{\sum_{k'=1}^{K} q_{jk'}^2 / f_{k'}}, \quad \text{where } f_k = \sum_{j=1}^{|\mathcal{X}|} q_{jk},$$

enhancing the influence of assignments with higher confidence,

(ii) $\mathcal{L}_{\text{MCL}}$ for incorporating pairwise constraints, and

(iii) A reconstruction loss to preserve the intrinsic data structure.

This process employs two hyperparameters: λ_1 to balance the clustering and reconstruction losses, and λ_2 to weight the contributions of positive and negative constraints within the MCL-based loss. When no constraints are available, CIDEC reduces to the unsupervised clustering model IDEC [40].

DCGMM [13]: DCGMM combines a deep generative model (i.e., a variational autoencoder-like architecture) with a conditional Gaussian mixture framework to handle pairwise constraints. Specifically, DCGMM models each cluster as a Gaussian mixture component, where the number of components is set to the known ground-truth class count K. It incorporates constraint information by conditioning on positive and negative pairs, thereby reshaping the latent variable distributions. Additionally, DCGMM assigns a weight $|W_{i,j}|$ to each pairwise constraint between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, reflecting the degree of certainty in that constraint. During training, it jointly optimizes (i) the variational likelihood of the autoencoder and (ii) a constraint-based term weighted by $|W_{i,j}|$ that pushes instances from positive pairs into the same mixture component and instances from negative pairs into different components. When no constraints are given, DCGMM reduces to VaDE [54], an unsupervised clustering model.

SDEC [11]: SDEC combines an anchor-free constraint loss with the anchor-based deep embedding clustering objective from [50] to learn cluster assignments that satisfy constraints. Specifically, SDEC minimizes a weighted sum of the unsupervised clustering loss $\mathcal{L}_{\text{DEC}}$, and a Euclidean distance-based constraint loss. The latter encourages positive pairs to be close in the latent space while pushing negative pairs apart:

$$\mathcal{L}_{\text{Euclidean}} = \frac{1}{|\mathcal{C}|} \sum_{i=1}^{|\mathcal{C}|} (2y_i - 1)\, d(\boldsymbol{z}_{a_i}, \boldsymbol{z}_{b_i})^2,$$

where $d(\boldsymbol{z}_{a_i}, \boldsymbol{z}_{b_i}) = \|\boldsymbol{z}_{a_i} - \boldsymbol{z}_{b_i}\|_2$. SDEC introduces a trade-off factor λ to balance these two objectives, optimizing the combined loss

$$\mathcal{L}_{\text{SDEC}} = \mathcal{L}_{\text{DEC}} + \lambda \mathcal{L}_{\text{Euclidean}}.$$

When no constraints are available, SDEC degenerates to the unsupervised clustering model DEC [50].

AutoEmbedder [16]: AutoEmbedder also learns pairwise embeddings in Euclidean space but does not include any unsupervised clustering loss, making it purely anchor-free for deep constraint embedding. AutoEmbedder uses an MSE loss based on a truncated Euclidean margin:

$$\mathcal{L}_{\text{MSE}} = \frac{1}{|\mathcal{C}|} \sum_{i=1}^{|\mathcal{C}|} \Big(\min\{\max(0, d(\boldsymbol{z}_{a_i}, \boldsymbol{z}_{b_i})), \alpha\} \; - \; \alpha(1 - y_i) \Big)^2,$$

where $d(\boldsymbol{z}_{a_i}, \boldsymbol{z}_{b_i}) = \|\boldsymbol{z}_{a_i} - \boldsymbol{z}_{b_i}\|_2$, and α is a manually chosen margin for the Euclidean distance. With $\mathcal{L}_{\text{MSE}}$, AutoEmbedder learn representations that pull positive pairs together while ensuring that negative pairs remain at least a margin α apart.

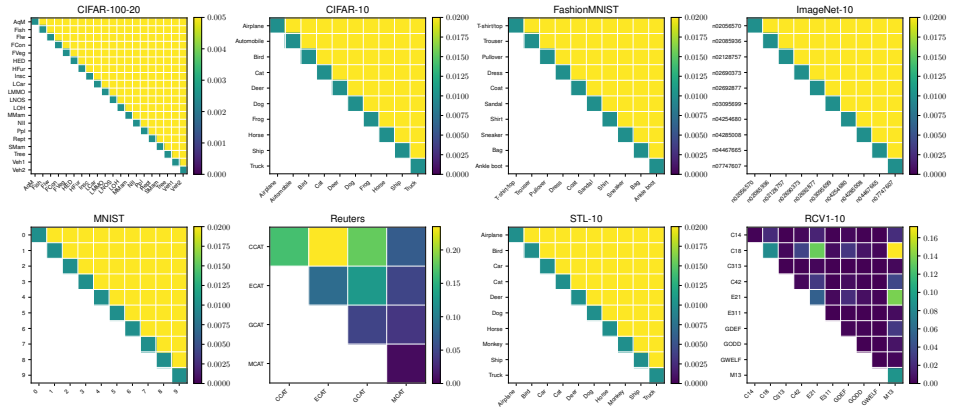


Figure 8: Distribution of randomly sampled constraints across ground-truth clusters. Each heatmap is a $K \times K$ symmetric matrix (represented by its upper triangular part) that illustrates the fraction of constraints originating from each corresponding pair of clusters (k, k'). The fractions in all matrix entries sum to 1 for each dataset.

E.3 Protocol

We provide the details of the experimental protocol used in our experiments.

Data splitting. In accordance with the protocol specified in the main text (Sect. 5.1), we adhere to the original pre-split training/test partitions for FashionMNIST, MNIST, and the Reuters subset. For the remaining benchmarks (CIFAR-10, CIFAR-100-20, STL-10, ImageNet-10, and RCV1-10), we randomly split each dataset into 80% for training and 20% for testing, resulting in 48,000/12,000 samples for training/testing in CIFAR-10 and CIFAR-100-20, 10,400/2,600 in STL-10 and ImageNet-10, and 142,135/35,534 in RCV1-10. Following [14], we then reserve 1,000 samples from each training split to form a validation set, used solely for hyperparameter tuning in baselines that require it (e.g., VolMaxDCC [14], AutoEmbedder [16]).

Constraint set generation. For our standard experiments, we generate pairwise constraints via random sampling of training data pairs according to their ground-truth clusters. Concretely, for a training set with N samples partitioned into K ground-truth clusters $\mathcal{S}^* = \{\mathcal{S}_k^*\}_{k=1}^K$, we randomly select pairs of samples and assign a *positive* constraint $(y_i = 1)$ if both samples lie in the same cluster (i.e., $\boldsymbol{x}_{a_i}, \boldsymbol{x}_{b_i} \in \mathcal{S}_k^*$), or a *negative* constraint $(y_i = 0)$ if they come from different clusters. In principle, there are up to $\binom{|\mathcal{X}|}{2} = \frac{|\mathcal{X}|(|\mathcal{X}|-1)}{2}$ possible constraints, among which each cluster $\mathcal{S}_k^*$ can yield up to $\frac{|\mathcal{S}_k^*|\,(|\mathcal{S}_k^*|-1)}{2}$ positive constraints and each cluster pair $(\mathcal{S}_k^*, \mathcal{S}_{k'}^*)$ with $k' \neq k$ can yield up to $|\mathcal{S}_k^*| \cdot |\mathcal{S}_{k'}^*|$ negative constraints. Consequently, the proportion of positive and negative constraints from different ground-truth clusters reflects the underlying class distribution. Fig. 8 visualizes the distribution of the randomly sampled constraints (of any chosen size, e.g. 1k, 5k, or 10k) in the form of $K \times K$ heatmaps for the eight datasets. Unless otherwise stated, our experiments use this random sampling strategy for constructing constraint sets.

Imbalanced constraint set generation. For each imbalanced-constraint trial, we generate a group of three sets, `IMB0`, `IMB1` and `IMB2`, to evaluate performance under progressively skewed constraint distributions. We fix a size ratio $|\texttt{IMB0}| : |\texttt{IMB1}| : |\texttt{IMB2}| = 1 : 5 : 10$, and designate the first ground-truth class in each dataset as the "IMB cluster," from which additional negative constraints are predominantly sampled. Concretely, we form `IMB0` of size $|\texttt{IMB0}|$ by randomly sampling pairs of training data, labeling them positive if both samples lie in the same ground-truth cluster or negative otherwise (the same procedure as in Fig. 8). We then obtain `IMB1` by adding $\big(|\texttt{IMB1}| - |\texttt{IMB0}|\big)$ extra negative constraints linking the IMB cluster to other clusters, and further enlarge `IMB1` to `IMB2` by appending $\big(|\texttt{IMB2}| - |\texttt{IMB1}|\big)$ similarly imbalanced constraints. As a result, $\texttt{IMB0} \subset \texttt{IMB1} \subset \texttt{IMB2}$, ensuring that any performance decline from `IMB0` to `IMB1` or `IMB2` cannot be attributed to removing earlier constraints.

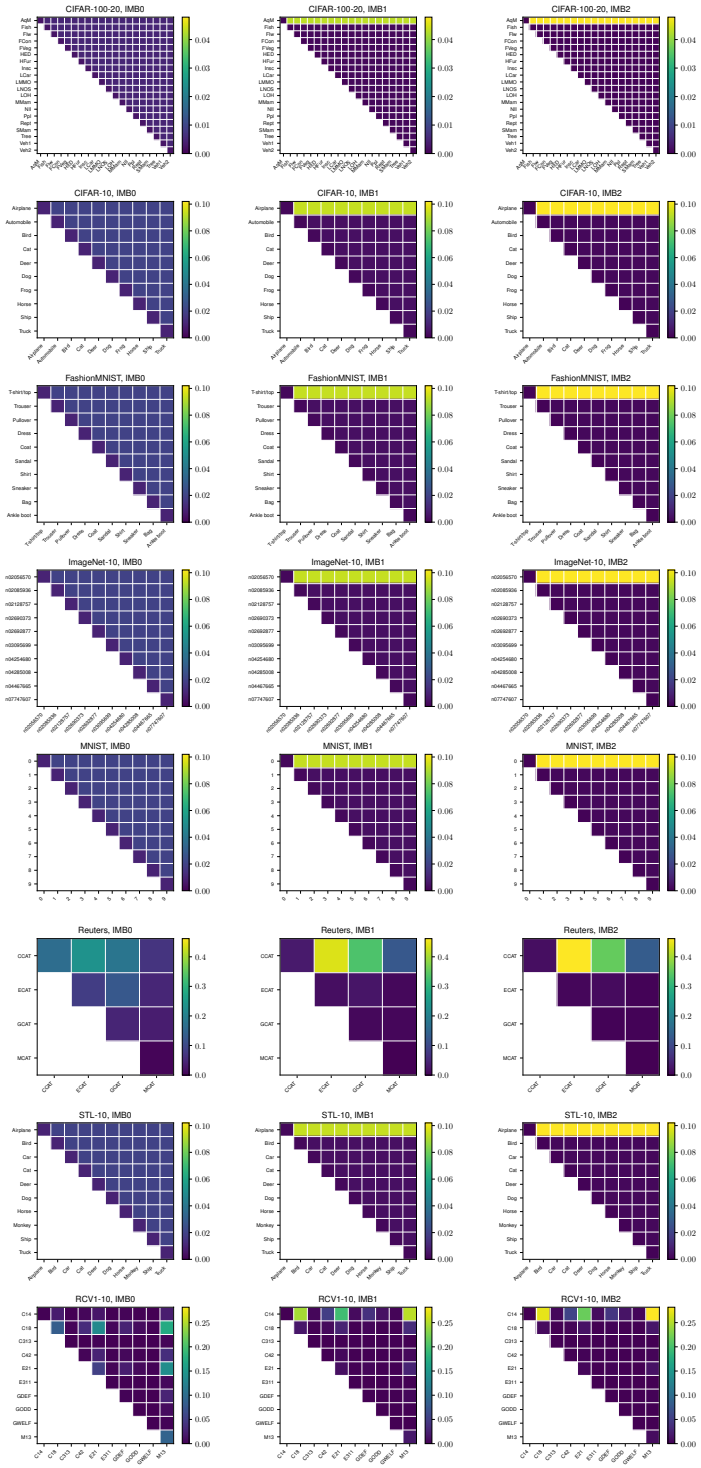


Figure 9: Heatmaps of imbalanced constraint distributions, $K \times K$ symmetric matrices (represented by its upper triangular part), for eight datasets. Rows correspond to datasets, while the three columns represent `IMB0`, `IMB1`, and `IMB2`, with sizes in the ratio of $1 : 5 : 10$. To form imbalanced constraint sets, a single "IMB cluster," corresponding to the first row in each heatmap, is selected to receive additional negative constraints connecting it to other clusters. The heatmap intensity at entry (k, k') indicates the fraction of constraints connecting ground-truth clusters k and k', reflecting how `IMB1` and `IMB2` become increasingly dominated by the IMB cluster's negative constraints.

Since we select the same IMB cluster and preserve the above ratio for each dataset, every group of three sets exhibits a consistent inter-cluster distribution. Fig. 9 illustrates this for the eight datasets, with each row corresponding to one dataset and the three columns showing `IMB0`, `IMB1`, `IMB2` as heatmaps of size $K \times K$; the highlighted area in each `IMB1` and `IMB2` heatmap involves the IMB cluster. We can observe that `IMB1` and `IMB2` gradually become dominated by negative constraints involving the IMB cluster. In Fig. 3, we report empirical results for $\big(|\texttt{IMB0}|, |\texttt{IMB1}|, |\texttt{IMB2}|\big) = (10\text{k}, 50\text{k}, 100\text{k})$, while additional experiments with varying constraint set sizes can be seen in Appendix F.2. These additional constraint sets are generated using the same methodology and exhibit similar cluster-wise constraint distributions.

Cluster number inference. To evaluate our PCA-based cluster-number inference method (Algorithm 2), we apply standard mean-centered PCA [44] to the ℓ_2-normalized embeddings of instances involved in negative pairs. For comparison with alternative K-inference methods and to assess the applicability of different deep constraint embeddings, we adopt two alternative strategies, specifically "K-means + Silhouette Coefficient" and "Agglomerative Clustering + K-cluster lifetime", which are applied to both the SpherePair and AutoEmbedder representations:

- **K-means + Silhouette Coefficient (SC).** For each candidate K', we run K-means clustering 5 times with random initializations and compute the average SC score on a random subset of 5,000 instances. Specifically, for each instance j, we calculate the average intra-cluster distance d_j^{intra} between its embedding $\boldsymbol{z}_j$ (for AutoEmbedder) or $\text{Norm}(\boldsymbol{z}_j)$ (for SpherePair) and the embeddings of all other instances from its assigned cluster $\mathcal{S}_k$. We also determine d_j^{inter}, the minimum average distance between $\boldsymbol{z}_j$ (or $\text{Norm}(\boldsymbol{z}_j)$) and the embeddings of instances from any other assigned cluster $\mathcal{S}_{k'}$, where $k' \neq k$. The SC score s_j for instance j is then computed as $s_j = \big(d_j^{\text{inter}} - d_j^{\text{intra}}\big) \,/\, \max\big(d_j^{\text{intra}}, d_j^{\text{inter}}\big)$. The overall SC score is the mean of s_j across all sampled instances. A higher SC indicates better-defined clusters with greater separation and lower intra-cluster dispersion. We select the number of clusters K' as K^* that corresonds to the maximal SC score.
- **Agglomerative clustering + K-cluster lifetime.** We employ agglomerative clustering with Ward linkage [43], which iteratively merges the closest clusters while recording the corresponding linkage distances. For each candidate number of clusters, K', let $d_{K'}$ denote the linkage distance when the dataset is partitioned into K' clusters. The *K-cluster lifetime* at level K' is defined as $\Delta d_{K'} = d_{K'} - d_{K'+1}$. A larger $\Delta d_{K'}$ indicates a more significant increase in linkage distance, or equivalently, a longer K-cluster lifetime. This suggests that the K'-cluster partition is more stable and well-separated. To determine the optimal number of clusters, K^*, we select $K^* = \text{argmax}_{K'}(\Delta d_{K'})$. In Figs. 16 and 18, each $\Delta d_{K'}$ is normalized using d_2 for readability, resulting in $\Delta \hat{d}_{K'} = \frac{\Delta d_{K'}}{d_2}$.

E.4 Implementation

We provide the implementation details[11] for our experiments.

Software and hardware. We implement all methods (except DCGMM[12]) in PyTorch 1.5.1 [13] with Python 3.7. For SpherePair and AutoEmbedder, we use scikit-learn's K-means implementation[14] and fastcluster's efficient hierarchical clustering implementation[15] for clustering. Experiments are conducted on Tesla V100 GPU with 16 GB of memory.

Data preprocessing. For Reuters and RCV1-10 subsets, we directly use the preprocessed tf-idf vectors. For all other datasets, we convert images into feature vectors to facilitate fair comparisons using fully connected networks. Specifically, for the 28×28 grayscale images in MNIST and FashionMNIST, we reshape each image into a 784-dimensional vector, mirroring the methods of [11, 12, 13]. For color images in CIFAR-10, CIFAR-100-20, STL-10, and ImageNet-10, we adopt

[11] Our source code is available on GitHub: https://github.com/spherepaircc/SpherePairCC/tree/main
[12] We use the authors' implementation for DCGMM: https://github.com/lauramanduchi/DC-GMM
[13] PyTorch 1.5.1: https://github.com/pytorch/pytorch/releases/tag/v1.5.1
[14] Scikit-learn webpage: https://scikit-learn.org/0.19/documentation.html
[15] fastcluster library: https://pypi.org/project/fastcluster/

the unsupervised feature extraction strategy proposed in [53]; in particular, we train a ResNet-34 [55] model for 1,000 epochs and utilize the resulting 512-dimensional latent representations. This preprocessing is consistent with the method employed in [14]. This vectorization step enables us to apply the same standard fully connected architectures across all datasets, ensuring consistency and fairness in evaluating the baselines.

Network architectures and pretraining. For all methods except VanillaDCC and VolMaxDCC, we employ a fully connected encoder with hidden layers of size 500–500–2000 (following [11, 12, 13]), using an embedding dimension of $D = 20$ for CIFAR-100-20 and $D = 10$ for all other datasets, unless stated otherwise. Notably, while the original AutoEmbedder [16] utilizes a pre-trained MobileNet-based CNN, we adapt it to use our standardized fully connected network to ensure fair comparison across all models. In contrast, VanillaDCC and VolMaxDCC [14] adopt a distinct architecture comprising two hidden layers of size 512–512 followed by a classification layer corresponding to the number of clusters K, as recommended in [14].

Except for the end-to-end VanillaDCC and VolMaxDCC, all other methods undergo unsupervised pretraining on training sets. Specifically, for SpherePair, SDEC, CIDEC, and AutoEmbedder, we utilize a two-stage stacked denoising autoencoder (SDAE) pretraining approach [56, 57], consistent with the works [11, 12]:

(i) *Layer-wise Pretraining*: Each hidden layer is individually pretrained as a single-layer denoising autoencoder for 300 epochs using 20% random masking noise and MSE loss. During this phase, the output of each encoder serves as the input to the subsequent layer, progressively refining the weights of each layer.

(ii) *End-to-End Fine-Tuning*: After completing layer-wise pretraining, the entire network is jointly optimized for an additional 500 epochs. This phase continues to apply 20% masking noise to the inputs.

A key distinction for SpherePair during pretraining is the normalization of latent embeddings before decoding, as specified in Eq. 3. This ensures that the pre-trained autoencoder retains angular information critical for our clustering objectives. For DCGMM [13], we follow the authors' setting by pretraining a variational autoencoder (VAE) for 10 epochs, aligning with their implementation strategy.

Hyperparameters and optimization. We provide detailed hyperparameter settings and optimization configurations for our SpherePair method and all baseline models.

- **SpherePair (Ours).** We fix the negative-zone factor ω at 2 according to our theoretical analysis, and set the reconstruction loss weighting parameter $\lambda = 0.02$ in all experiments unless varied for sensitivity analysis. For optimization, we employ the standard Adam optimizer with a learning rate of 0.001. The constraint mini-batch size is set to $|\mathcal{B}_c| = 256$, and consequently, the instance mini-batch size is determined by $|\mathcal{B}_x| = |\mathcal{X}|/\frac{|\mathcal{C}|}{|\mathcal{B}_c|}$, where $|\mathcal{X}|$ represents the dataset size and $|\mathcal{C}|$ denotes the constraint set size. Training is conducted for a maximum of 300 epochs. An early stopping criterion is applied after the first 100 epochs, terminating training if the relative change in loss $\mathcal{L}$ (Eq. 4) remains less than 0.1 for 5 consecutive epochs. For clustering on the learned spherical representations $\mathcal{Z}_{\text{sphere}}$, we utilize either (i) the K-means algorithm with 20 random initializations or (ii) hierarchical clustering using the Ward linkage method. For cluster-number inference, we set the tail ratio $\rho = 0.05$ when computing the tail-averaged minimal inter-cluster angle $\bar{\delta}_d$.

- **VanillaDCC.** VanillaDCC is implemented straightforwardly by optimizing $\mathcal{L}_{\text{MCL}}$ using the standard Adam optimizer with a learning rate of 0.001. The batch size is set to 256. Training is conducted for a maximum of 300 epochs, with early stopping triggered if the relative change in the soft cluster assignments for training samples falls below 0.001 over 2 consecutive epochs. This early stopping strategy is widely adopted in end-to-end deep clustering [50, 40] and deep constrained clustering [11, 12]

- **VolMaxDCC.** Following the VolMaxDCC paper[16] [14], we parameterize the optimization variable B such that each element $B_{ij} = \frac{1}{1+\exp(-B'_{ij})}$, where B'_{ij} is a trainable parameter initialized to 1 if $i = j$ and to -1 otherwise. We utilize the SGD optimizer with learning rates of 0.5 for the network parameters and 0.1 for B', respectively, and set the batch size to 128. The trade-off factor (geometric regularization weight) λ is selected by searching over the range $\{0, 10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}\}$ based on the model's best accuracy on the validation set. The optimal λ values identified for the datasets CIFAR-100-20, CIFAR-10, FMNIST, ImageNet-10, MNIST, Reuters, STL-10, and RCV1-10 are 10^{-4}, 10^{-4}, 10^{-4}, 10^{-5}, 10^{-2}, 10^{-4}, 10^{-2}, and 10^{-4}, respectively.
- **CIDEC.** We follow the authors' recommendations by setting $\lambda_1 = 1$ to balance the clustering and reconstruction losses and $\lambda_2 = 0.1$ to weight the contributions of positive constraints within the MCL-based loss. The K-means algorithm is executed 20 times to initialize the K cluster anchors. Optimization is performed using the standard Adam optimizer with a learning rate of 0.001 and a batch size of 256. Training proceeds for up to 300 epochs, with early stopping invoked if the soft cluster assignments exhibit a relative change of less than 0.001 over consecutive epochs.
- **DCGMM.** Utilizing the authors' implementation, we set the constraint weights $|W_{ij}| =$ 10,000 and adhere to their optimization configurations. However, we pretrain a variational autoencoder (VAE) for 10 epochs exclusively on the instances in the training set, rather than using all instances from both the training and test sets.[17]
- **SDEC.** We align with the authors' recommendations by setting the constraint loss weight $\lambda = 10^{-5}$. The K-means algorithm is executed 20 times to initialize the K cluster anchors used for unsupervised clustering. Optimization is carried out using the SGD optimizer with a learning rate of 0.01 and a batch size of 256. Training continues for a maximum of 300 epochs, with early stopping triggered if the soft cluster assignments change by less than 0.001 over consecutive epochs.
- **AutoEmbedder.** We implement AutoEmbedder using the same fully connected encoder architecture as SpherePair, ensuring consistency across models. The optimization settings mirror those of SpherePair: an Adam optimizer with a learning rate of 0.001 and a batch size of 256. Training is conducted for up to 300 epochs, with an early stopping criterion applied after the first 100 epochs, terminating training if the relative change in loss remains below 0.1 for 5 consecutive epochs. Unlike the original AutoEmbedder [16], which is based on a pre-trained CNN network with a well-structured embedding space, our fully connected implementation requires modifications to the loss function. Specifically, the MSE loss $\mathcal{L}_{\text{MSE}}$ can lead to scenarios where the embedding distance for positive pairs exceeds the margin α, causing gradient issues. To address this, we introduce separate margins α_1 and α_2 for negative and positive constraints, respectively. We search for α_1 within {1, 10, 50, 100, 500, 1000, 5000} and α_2 within {100, 1000, 10000}, performing hyperparameter tuning based on validation set performance, similar to the approach used for VolMaxDCC. The optimal α_1 values identified for the datasets CIFAR-100-20, CIFAR-10, FMNIST, ImageNet-10, MNIST, Reuters, STL-10, and RCV1-10 are 500, 500, 50, 10, 100, 500, 50, and 10, respectively, while the optimal α_2 is consistently 10,000 across all datasets. This outcome is expected, as the margin for positive constraints does not contribute to the Euclidean clustering objective, and smaller margins render positive constraints ineffective.

F Additional experimental results

In this appendix, we present supplementary experimental results, building upon the experimental settings detailed in Appendix E. Additionally, we provide further insights and findings related to our SpherePair approach.

[16] While we strictly follow the authors' optimal hyperparameter search strategy, which uses ground-truth class label information in the validation data, such information is typically unavailable in constrained clustering tasks.

[17] Based on the authors' source code, we observed that their experiments involved pretraining the VAE on all instances from both the training and test sets. This constitutes a transductive setting, which is unsuitable for scenarios requiring inductive learning, as is the case in all our experiments.

F.1 Comparison of hierarchical clustering results

We present additional results from our comparative study. In Table 1 presented in the main text, we reported the primary clustering analysis results for two anchor-free deep constraint embedding models, AutoEmbedder and our SpherePair, using K-means applied to their learned representations. Here, we extend the analysis by presenting results for both models using Ward's agglomerative hierarchical clustering, as shown in Table 2.

Table 2: Comparative performance (%) of ACC, NMI, and ARI across multiple datasets for AutoEmbedder (AE) and SpherePair (Ours) models using 1k, 5k, and 10k constraints. The results are derived from the hierarchical clustering analysis applied to their learned representations. Consistent with the notation used in Table 1, blue and black indicate training and test performance, respectively. Better results are highlighted in **bold**.

		1k			5k			10k		
		ACC	NMI	ARI	ACC	NMI	ARI	ACC	NMI	ARI
CIFAR100-20	AE	19.7, 19.2	21.4, 20.7	5.9, 5.8	10.5, 10.8	7.9, 8.5	2.1, 2.2	31.2, 31.5	37.3, 38.5	21.2, 22.1
	Ours	**45.9**, **46.4**	**46.4**, **46.3**	**29.4**, **30.5**	**55.0**, **54.9**	**50.9**, **51.2**	**37.4**, **38.7**	**61.6**, **62.6**	**54.8**, **54.9**	**42.6**, **44.1**
CIFAR10	AE	45.8, 44.2	51.3, 50.1	29.4, 28.0	82.1, 83.0	77.4, 78.3	72.3, 73.9	84.8, 85.3	79.4, 80.0	75.2, 76.5
	Ours	**83.7**, **84.3**	**77.0**, **77.1**	**72.3**, **73.0**	**89.5**, **89.4**	**81.5**, **81.1**	**79.8**, **79.6**	**90.5**, **90.2**	**82.4**, **82.0**	**81.3**, **80.8**
FMNIST	AE	40.7, 40.2	42.2, 40.8	25.7, 25.0	60.5, 61.2	59.5, 60.0	46.8, 48.2	66.6, 67.1	65.0, 64.6	52.4, 53.6
	Ours	**66.5**, **67.8**	**64.3**, **63.1**	**51.2**, **52.3**	**79.4**, **79.2**	**71.8**, **70.8**	**65.4**, **65.2**	**84.4**, **83.7**	**75.3**, **74.2**	**71.5**, **70.2**
ImageNet10	AE	63.3, 61.2	62.5, 58.0	47.0, 42.6	96.2, **96.4**	91.4, **91.9**	91.7, 92.1	96.6, 96.5	92.0, **92.1**	92.7, 92.4
	Ours	**95.9**, **96.1**	**91.1**, **91.4**	**91.3**, **91.6**	**96.8**, **96.4**	**92.3**, 91.7	**93.2**, **92.3**	**97.2**, **96.6**	**93.1**, **92.1**	**94.0**, **92.6**
MNIST	AE	44.2, 42.6	41.3, 38.3	27.1, 25.1	60.4, 61.1	57.5, 58.6	47.1, 49.1	87.3, 89.4	79.9, 83.0	77.9, 82.0
	Ours	**95.4**, **92.3**	**89.5**, **83.4**	**90.3**, **83.8**	**96.6**, **96.1**	**91.5**, **90.2**	**92.7**, **91.5**	**97.0**, **96.9**	**92.0**, **91.8**	**93.6**, **93.2**
REUTERS	AE	55.5, 55.1	24.0, 25.8	19.3, 19.4	87.6, 88.7	67.6, 69.1	76.4, 78.1	93.0, 91.9	79.0, 76.2	86.2, 83.6
	Ours	**91.6**, **92.3**	**73.1**, **75.1**	**80.4**, **82.4**	**96.0**, **94.8**	**84.4**, **81.0**	**90.5**, **87.4**	**97.6**, **95.4**	**89.7**, **82.4**	**94.6**, **88.6**
STL10	AE	59.3, 59.4	56.7, 55.9	41.1, 40.7	81.8, 83.1	76.0, 76.8	68.7, 71.2	89.6, 89.7	81.6, 81.3	79.0, 79.4
	Ours	**86.1**, **87.4**	**77.2**, **78.2**	**73.3**, **75.6**	**90.9**, **89.9**	**82.5**, **81.2**	**81.6**, **79.8**	**92.9**, **90.7**	**85.3**, **82.2**	**85.2**, **81.1**
RCV1-10	AE	31.7, 31.6	9.2, 8.6	6.5, 6.7	50.9, 51.9	32.3, 33.7	35.6, 38.1	77.5, 78.6	55.5, 57.4	67.2, 70.5
	Ours	**66.9**, **67.8**	**60.2**, **61.8**	**58.4**, **59.8**	**89.3**, **89.6**	**74.6**, **74.4**	**84.0**, **83.8**	**91.5**, **91.5**	**77.9**, **77.4**	**87.0**, **86.4**

From Table 2, we observe that SpherePair consistently outperforms AutoEmbedder across nearly all dataset-constraint-metric combinations. The only exception is a minor 0.2% gap in test NMI when using 5k constraints from ImageNet10. When comparing the hierarchical clustering results of SpherePair in Table 2 with the K-means results reported in Table 1, we find that SpherePair delivers nearly identical performance, with most deviations under 1% and a maximum difference of less than 4%. This consistency highlights the robustness of SpherePair's embeddings across different clustering algorithms. In contrast, AutoEmbedder exhibits more pronounced performance fluctuations when switching from K-means to hierarchical clustering, with some cases showing gaps as large as 15% (e.g., CIFAR-10 with 1k constraints). These variations suggest that AutoEmbedder struggles to generate well-structured cluster representations, making its downstream partitioning outcomes highly sensitive to the choice of clustering method.

These extended results underscore the versatility of SpherePair's learned embeddings, which deliver stable and high-quality clustering results regardless of the clustering analysis algorithm used. This flexibility is particularly valuable in practice, as hierarchical clustering can reveal dendrogram structures that provide insights into domain-specific phenomena—something K-means and end-to-end anchor-based DCC methods cannot readily achieve. Consequently, users can confidently replace the clustering step in our framework with a more interpretable or domain-specific method, assured that SpherePair's representations will continue to deliver strong performance.

F.2 Imbalanced constraints

We present additional experimental results that analyze model behavior and visualize learned representations in latent embedding spaces under imbalanced constraint conditions. The distribution of imbalanced constraints skews toward specific inter-cluster relationships due to experts' greater familiarity with particular knowledge areas, which is common in real-world scenarios.

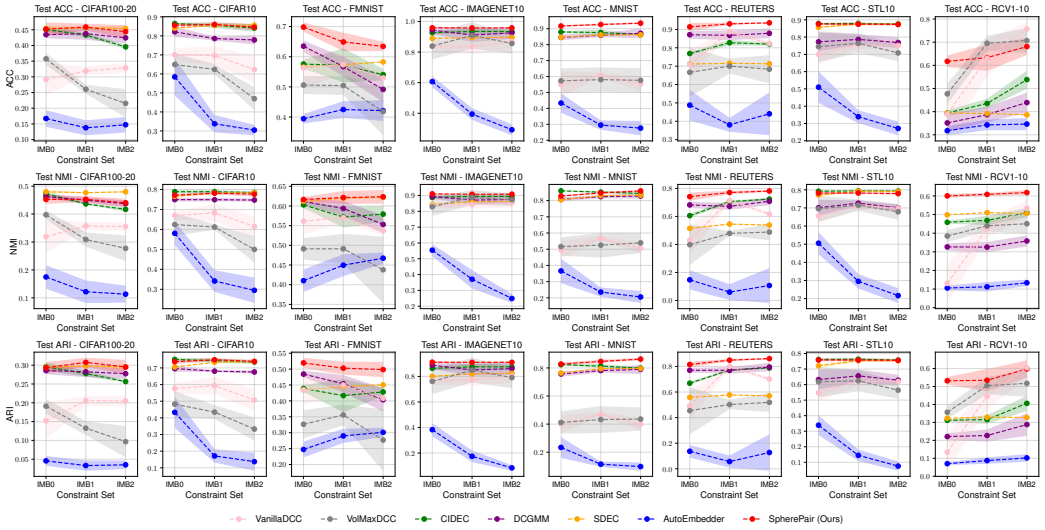


Figure 10: Test ACC/NMI/ARI performance (mean±std over 5 runs) of all models across datasets under the imbalanced constraints setting where ($|\texttt{IMB0}|$, $|\texttt{IMB1}|$, $|\texttt{IMB2}|$) = (1k, 5k, 10k).

F.2.1 Model behavior

To complement Fig. 3 in the main text, we provide additional results in Figs. 10 to 12. Each figure illustrates how performance changes as the imbalance level increases from IMB0 to IMB1 and IMB2, following the generation procedure in Appendix E.3, where the nested relation $\texttt{IMB0} \subset \texttt{IMB1} \subset \texttt{IMB2}$ rules out declines due to reduced constraints. The three figures correspond to different constraint set sizes, namely $(1\text{k}, 5\text{k}, 10\text{k})$, $(5\text{k}, 25\text{k}, 50\text{k})$, and $(10\text{k}, 50\text{k}, 100\text{k})$. Each shows three rows of metrics (ACC, NMI, ARI) across eight datasets, offering a comprehensive view of model behavior under varying imbalance levels.

Baseline behavior. Most baseline models exhibit increasing performance degradation as the imbalance level rises from IMB0 to IMB2, although the severity of this decline varies depending on the initial size of $|\texttt{IMB0}|$.

- When $|\texttt{IMB0}|$ is relatively small, the additional constraints from IMB1 or IMB2 can have a partially positive impact, mitigating some of the adverse effects of the skewed constraint distribution. For example, in Fig. 10, most baselines on the STL10 dataset experience only limited degradation as the imbalance increases, and on RCV1-10 most baselines even exhibit a noticeable performance gain.
- In contrast, when the balanced set size $|\texttt{IMB0}|$ is already large enough for near-saturation performance, introducing imbalanced constraints from IMB1 or IMB2 amplifies negative effects. As shown in Fig. 12, most baselines suffer pronounced drops on both STL10 and RCV1-10 compared to the lower-constraint scenarios.

An exception is the SDEC model, which remains stable under varying levels of imbalance. This stability likely stems from its reliance on an unsupervised clustering objective, as shown in Table 1 presented in the main text, where increasing constraints also leads to minimal performance gains. As a result, SDEC is less susceptible to both the benefits and drawbacks of heavily imbalanced constraint sets, maintaining relatively steady behavior under IMB conditions.

SpherePair behavior. SpherePair remains robust across datasets and imbalance scenarios, consistently ranking among the top-performing methods. The only notable degradations occur on CIFAR-100-20 and FMNIST; however, SpherePair is always noticeably less affected than baselines and consistently outperforms them under nearly all settings. In particular, on FMNIST, increasing $|\texttt{IMB0}|$ enables SpherePair to effectively exploit the richer constraint information to counteract the negative effects of imbalance (see Fig. 12). This resilience can be attributed to its ability to respect and leverage non-dominant local pairwise relationships.

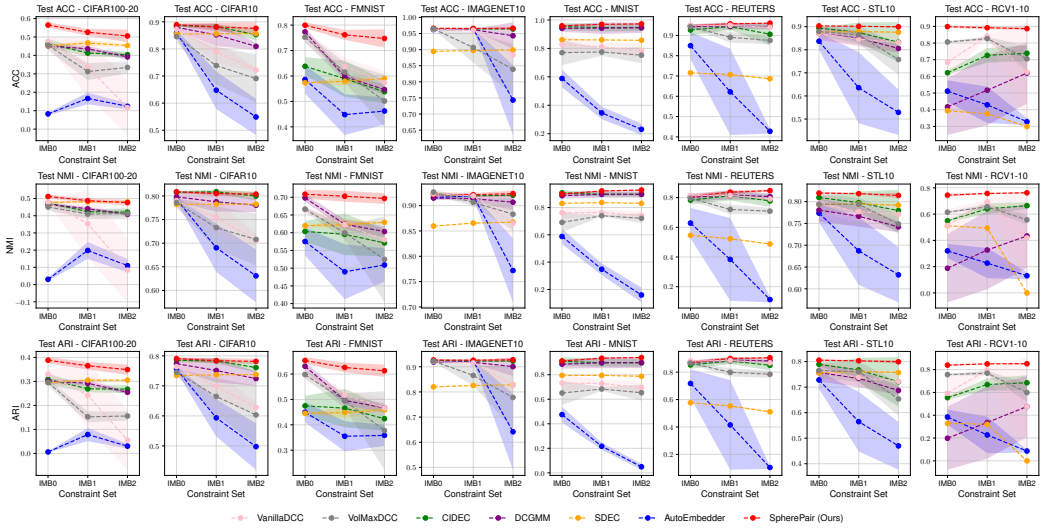


Figure 11: Test ACC/NMI/ARI performance (mean±std over 5 runs) of all models across datasets under the imbalanced constraints setting where ($|\texttt{IMB0}|$, $|\texttt{IMB1}|$, $|\texttt{IMB2}|$) = (5k, 25k, 50k).

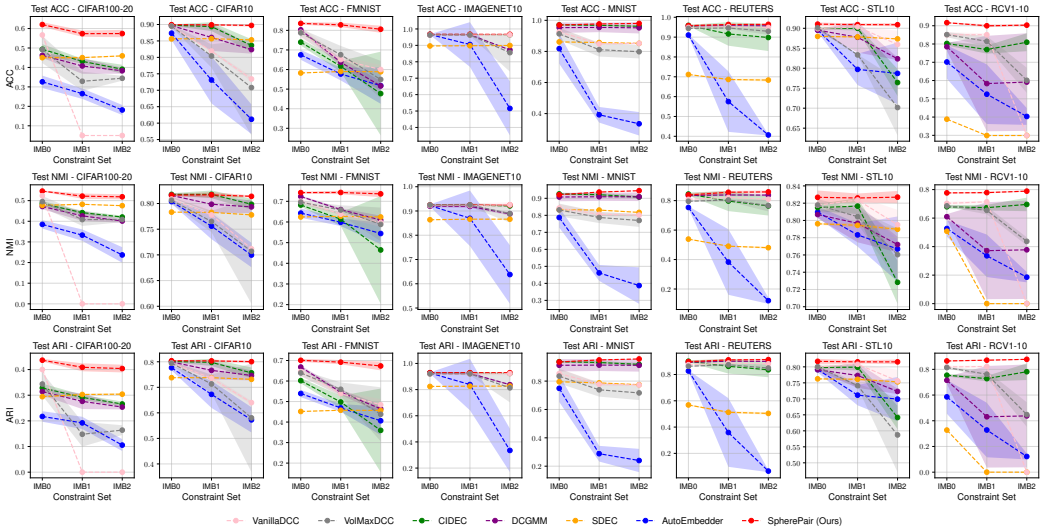


Figure 12: Test ACC/NMI/ARI performance (mean±std over 5 runs) of all models across datasets under the imbalanced constraints setting where ($|\texttt{IMB0}|$, $|\texttt{IMB1}|$, $|\texttt{IMB2}|$) = (10k, 50k, 100k).

SpherePair's robustness highlights its practical advantage in real-world scenarios, where annotators are more likely to label familiar pairwise relationships while neglecting less familiar ones. This adaptability makes SpherePair particularly suited for imbalanced constraint settings.

F.2.2 Latent embedding visualization

To gain deeper insights into how a DCC model is influenced by imbalanced constraints, we visualize the learned representations in the latent embedding space under the IMB2 configuration shown in Fig. 12 (where $|\texttt{IMB2}| = 100\text{k}$). The visualization focuses on four representative models that explicitly learn latent embeddings: **VanillaDCC** (an end-to-end classification approach), **CIDEC** (an end-to-end autoencoder approach), **AutoEmbedder** (a deep constraint embedding method in Euclidean space), and our **SpherePair** (a deep constraint embedding method in angular space).

For each model, the embeddings are visualized as follows: the output of the last hidden layer for VanillaDCC, the autoencoder's latent space for CIDEC and AutoEmbedder, and the unit-normalized

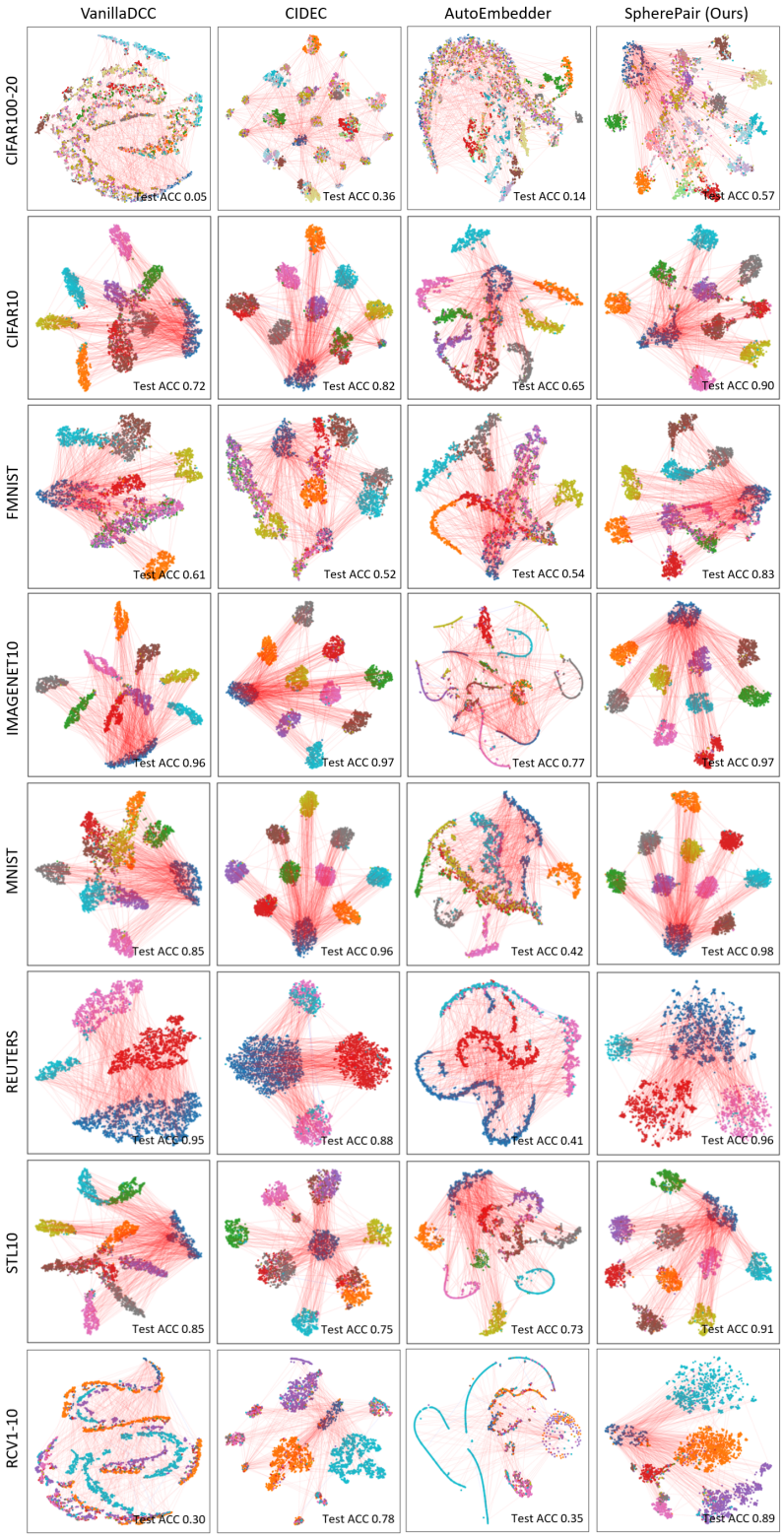


Figure 13: t-SNE visualization of embeddings under imbalanced constraints ($|$ `IMB2` $|= 100$k). Each column corresponds to one model: (a) VanillaDCC, (b) CIDEC, (c) AutoEmbedder, and (d) SpherePair. We visualize the final hidden-layer output (VanillaDCC), latent embedding (CIDEC, AutoEmbedder), and normalized latent embedding (SpherePair). Different marker colors denote different ground-truth categories.

spherical embeddings for SpherePair. Fig. 13 show the t-SNE plots of the resulting representations across different datasets, leading to several key observations:

End-to-end methods. Both VanillaDCC and CIDEC exhibit a tendency to mismatch local similarities and global clustering decisions under imbalanced constraints. This often results in instances from different ground-truth categories being incorrectly embedded into tight, misassigned clusters. These clusters indicate that imbalanced constraints cause the anchors to disproportionately emphasize dominant relationships, neglecting minority local constraints.

AutoEmbedder. As an anchor-free Euclidean embedding method, AutoEmbedder generates non-convex and less discriminative clusters, suggesting that pairwise learning in Euclidean space is particularly sensitive to imbalance. While AutoEmbedder occasionally preserves local groupings for specific categories, its clusters can be challenging to partition accurately. For instance, on the Reuters dataset, K-means applied to AutoEmbedder's features yields an ACC of only 0.41.

SpherePair. In contrast, SpherePair leverages the properties of angular space and its derived *negative zone* to maintain a balance between respecting local relationships and forming sufficiently separable, convex clusters. Even under severe imbalance, SpherePair produces normalized embeddings that form compact, clearly discernible clusters, demonstrating its robustness in representation learning.

Overall, these visualizations highlight that under strong constraint imbalance, anchor-based end-to-end methods like VanillaDCC and CIDEC are prone to misclustering minority classes, while anchor-free Euclidean-based deep constraint embedding methods like AutoEmbedder struggle to form separable clusters. Our SpherePair, by capitalizing on angular distances, preserves coherent local structures and generates stable, well-defined clusters, even under skewed supervision.

F.3 Unknown cluster number

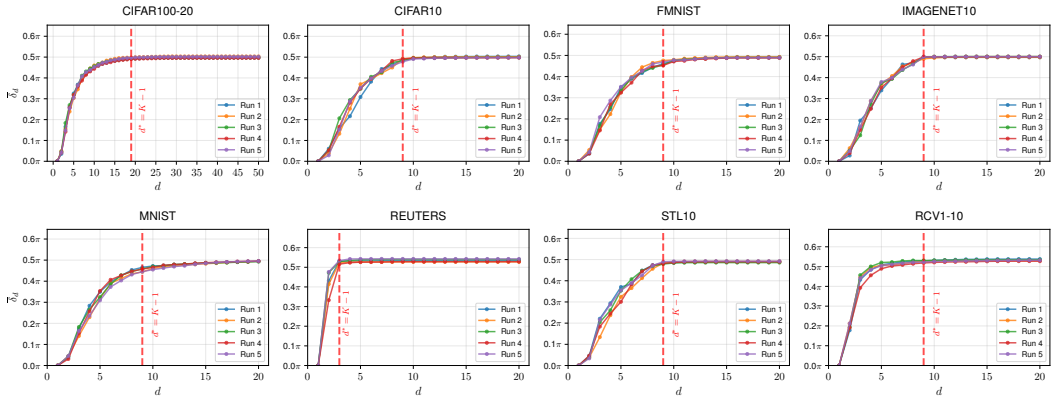
We comprehensively evaluate our PCA-based cluster-number inference combined with SpherePair in terms of its effectiveness across different constraint levels, its comparison with alternative K-inference strategies, and the applicability of SpherePair relative to DCC baselines for K-inference.

F.3.1 PCA-based K-inference under different constraint levels.

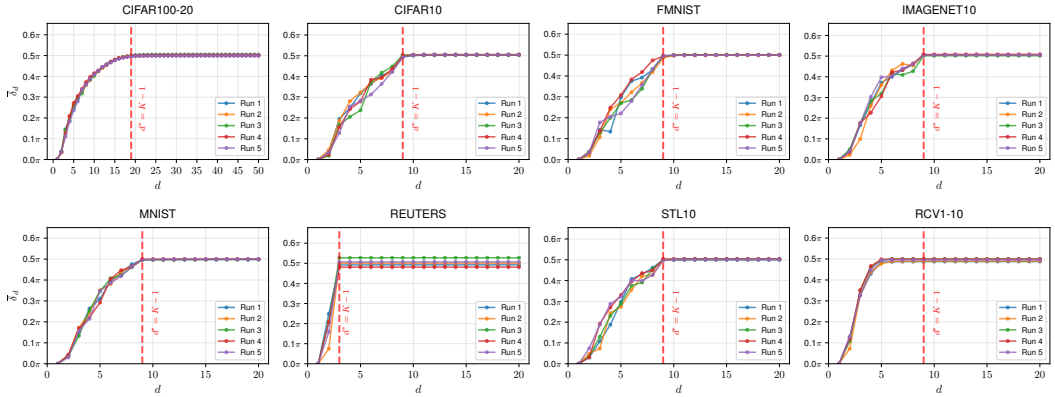
We extend the evaluation of our K-inference from Fig. 5 to additional constraint levels (1k/5k/10k), as shown in Fig. 14. Across most datasets, 10k constraints yield clear plateau entries, while 5k constraints produce slightly less pronounced entries that remain sufficient for correct K estimation. An exception is RCV1-10, where strong class imbalance poses inherent challenges and leads to inaccurate estimates across constraint settings. In the more limited 1k-constraint setting, the plateaus become much less sharp, particularly on CIFAR-100-20, MNIST, and FMNIST, resulting in more frequent inaccuracies. Nevertheless, under the 1k-constraint setting, our method still produces correct K estimates on CIFAR-10, ImageNet-10, Reuters, and STL-10 in nearly all cases, with only two minor deviations on CIFAR-10 where the estimate differed from the ground truth by 1.

F.3.2 Comparison with alternative post-clustering K-inference.

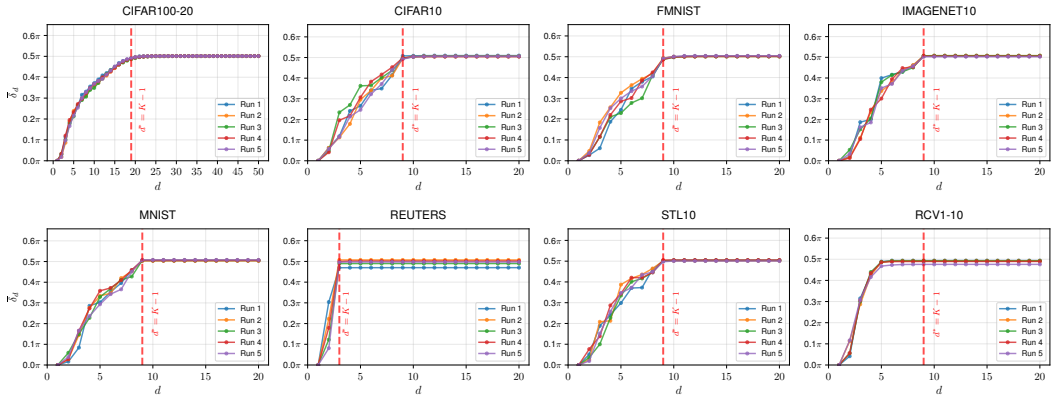
Under the same experimental setup as in Fig. 14, we consider two post-clustering validation strategies as alternatives to our geometric approach. Fig. 15 and Fig. 16 show the curves of the silhouette coefficient (SC) with K-means and the K-cluster lifetime with Agglomerative Clustering, obtained by sweeping candidate K values on SpherePair embeddings learned with 1k/5k/10k constraints. In both cases, the estimated K is given by the curve maximum. Comparing Figs. 15 and 16 with Fig. 14, we find: (i) SC with K-means is slightly less accurate than our method under 10k and 5k constraints, yielding minor misestimations on CIFAR-10, FMNIST, and STL-10, together with a markedly larger discrepancy from the ground-truth K on CIFAR-100-20. With only 1k constraints, while it outperforms our approach on FMNIST and MNIST, it is unstable on other datasets with around half of the estimates incorrect, and exhibits a deviation of up to 15 on CIFAR-100-20. (ii) K-cluster lifetime with Agglomerative Clustering shows stronger sensitivity to the constraint level, yielding almost no correct K estimates under 1k constraints except on ImageNet-10 and STL-10.



(a)

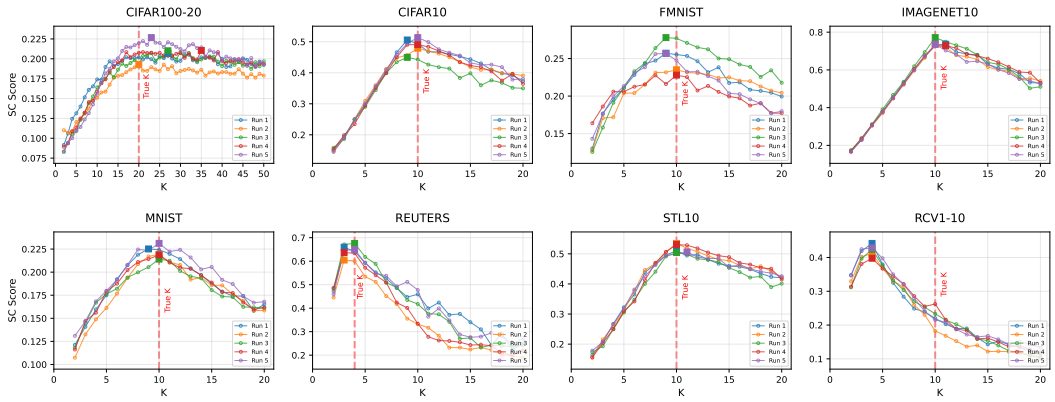


(b)

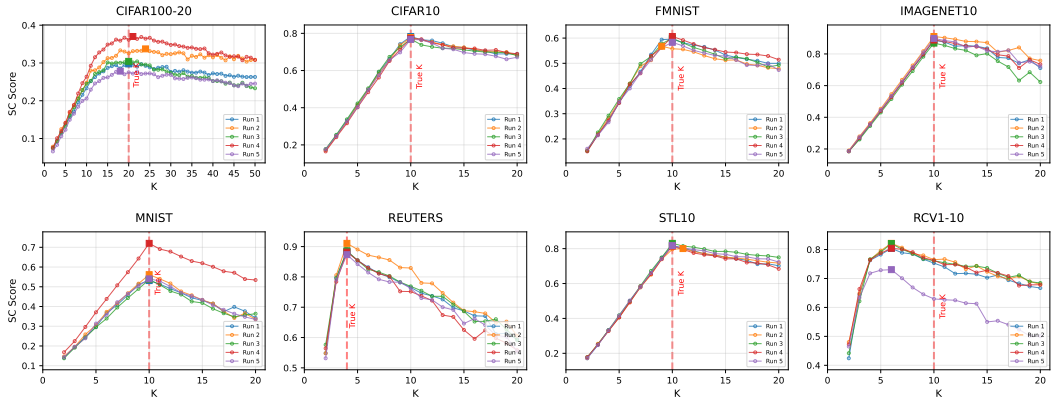


(c)

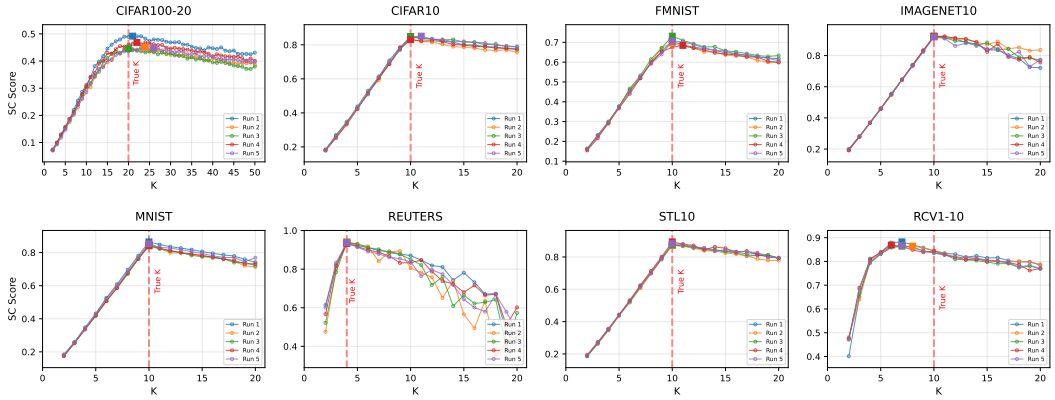
Figure 14: Tail-averaged minimal inter-cluster angle $\overline{\delta}_d$ vs. PCA subspace dimension d, obtained from SpherePair embeddings learned with (a) 1k, (b) 5k, and (c) 10k constraints across five runs. The red lines indicate the ground-truth intrinsic dimensions $d^* = K-1$.



(a)

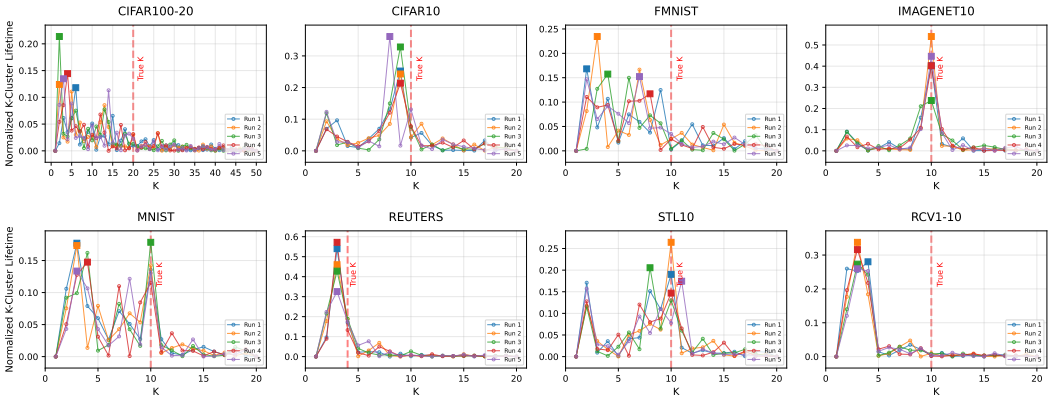


(b)

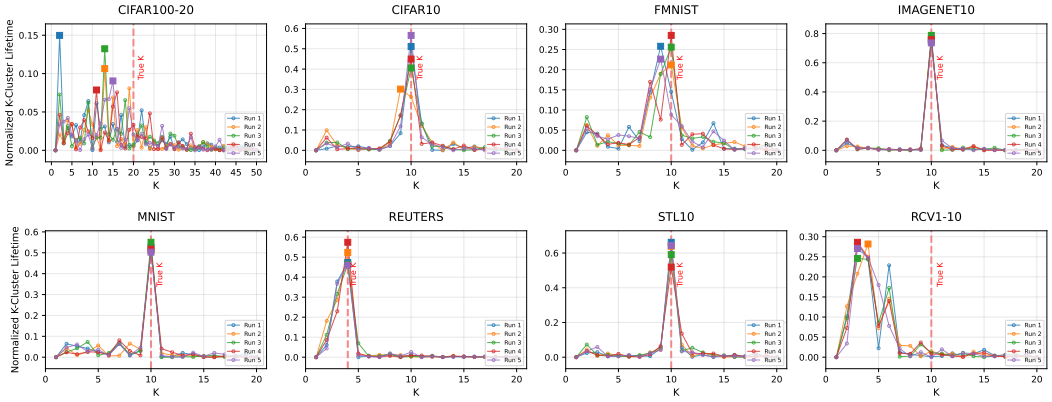


(c)

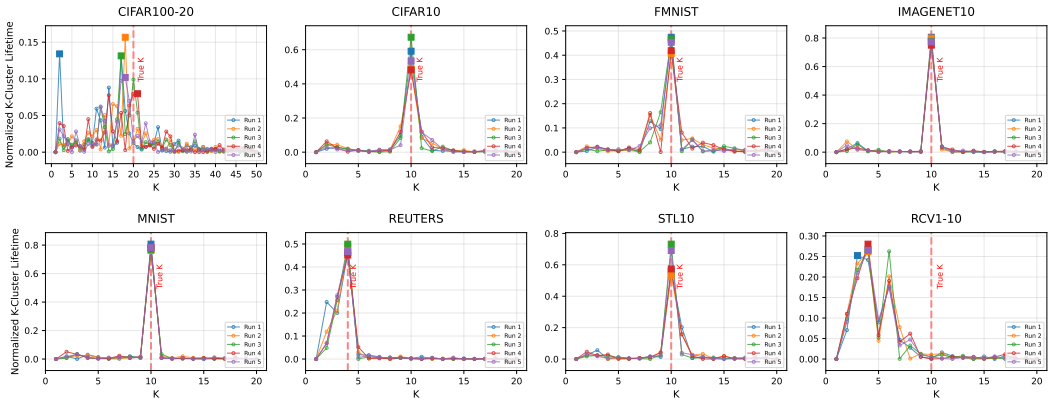
Figure 15: Silhouette coefficient (SC) with K-means for K-inference across five runs, obtained by sweeping candidate K values on SpherePair embeddings learned with (a) 1k, (b) 5k, and (c) 10k constraints. The estimated K corresponds to the maximum SC value (bold solid markers) in each curve. The red lines indicate the ground-truth K.



(a)

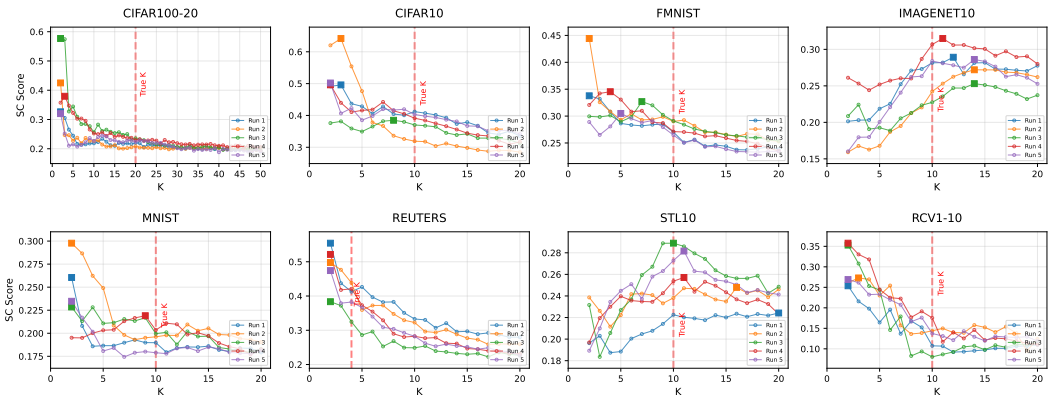


(b)

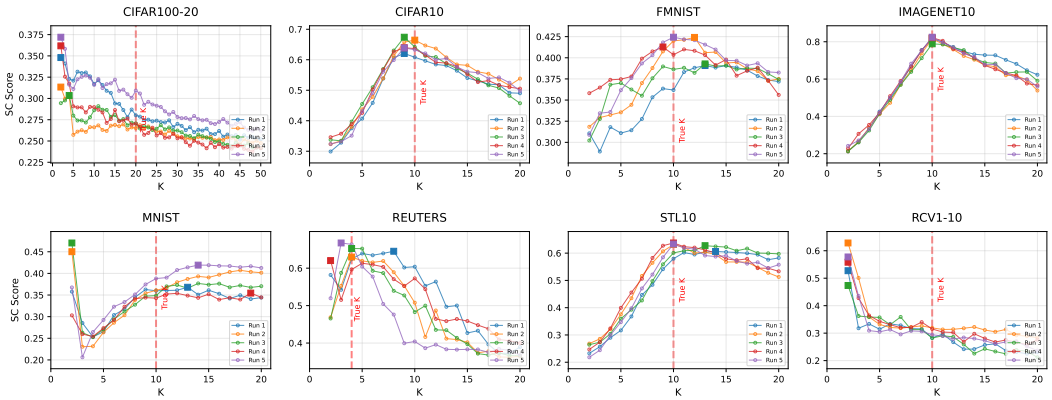


(c)

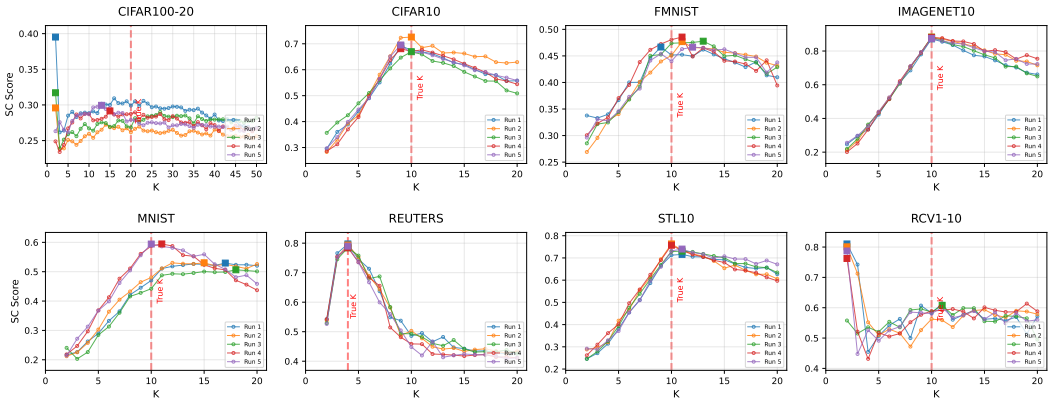
Figure 16: K-cluster lifetime with Agglomerative Clustering for K-inference across five runs, obtained from SpherePair embeddings learned with (a) 1k, (b) 5k, and (c) 10k constraints. The estimated K corresponds to the maximum lifetime value (bold solid markers) in each curve. The red lines indicate the ground-truth K.



(a)

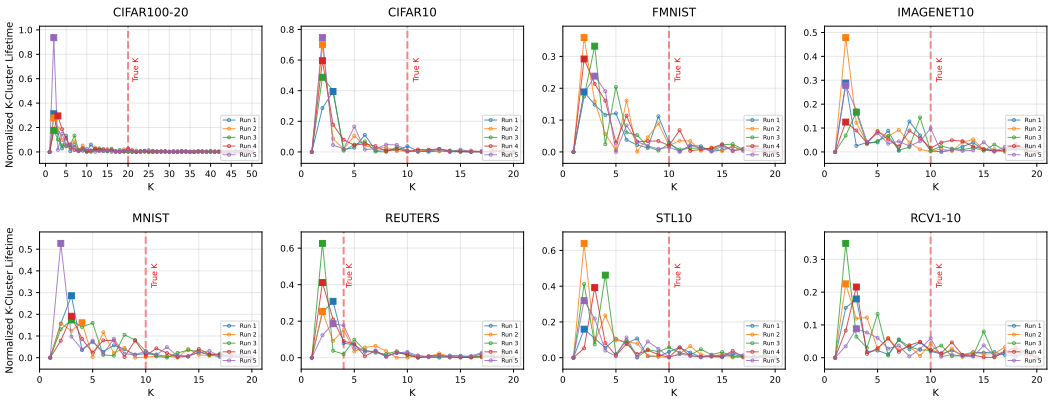


(b)

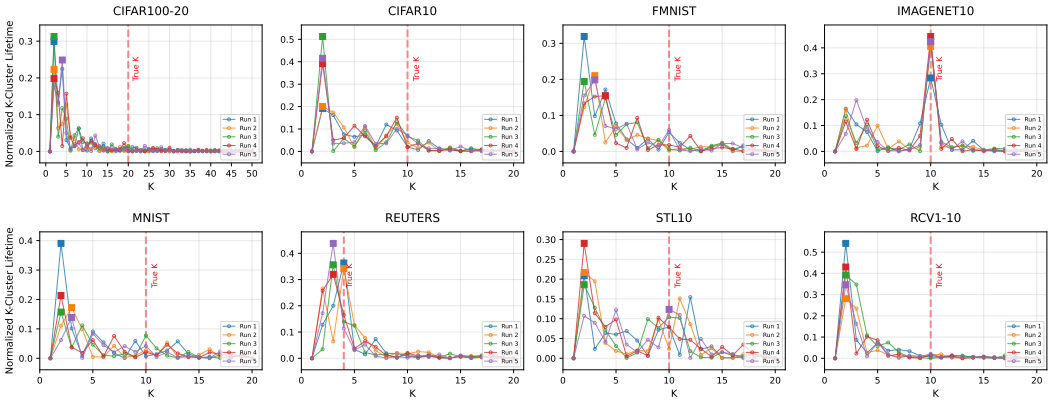


(c)

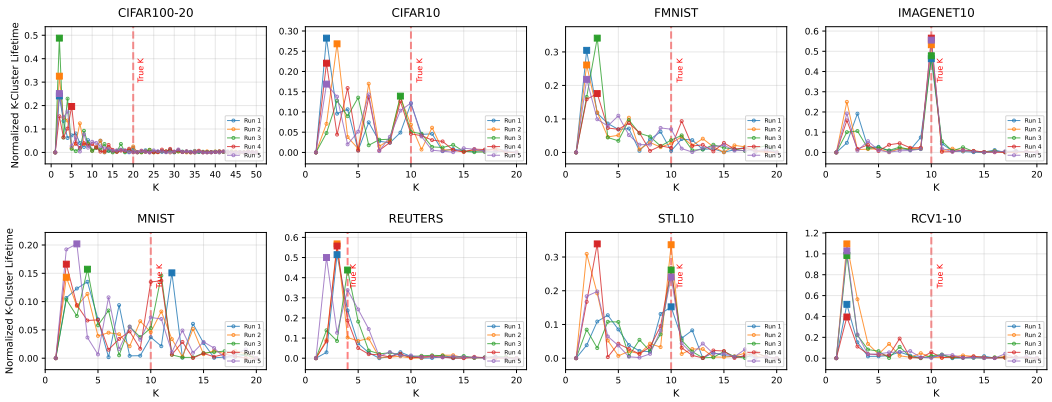
Figure 17: Silhouette coefficient (SC) with K-means for K-inference across five runs, obtained by sweeping candidate K values on AutoEmbedder embeddings learned with (a) 1k, (b) 5k, and (c) 10k constraints. The estimated K corresponds to the maximum SC value (bold solid markers) in each curve. The red lines indicate the ground-truth K.



(a)



(b)



(c)

Figure 18: K-cluster lifetime with Agglomerative Clustering for K-inference across five runs, obtained from AutoEmbedder embeddings learned with (a) 1k, (b) 5k, and (c) 10k constraints. The estimated K corresponds to the maximum lifetime value (bold solid markers) in each curve. The red lines indicate the ground-truth K.

Moreover, its estimates on CIFAR-100-20 are consistently unreliable with deviations up to 18 across all constraint levels, reflecting its failure in complex scenarios.

Overall, these results highlight the superior accuracy and robustness of our PCA-based K-inference on SpherePair embeddings. Appendix G further compares computational overhead, where post-clustering methods incur additional cost from repeated clustering (multiple K-means runs or one Agglomerative clustering), while our PCA-based inference is considerably more efficient as it requires only a single closed-form PCA solution.

F.3.3 Comparison with DCC baselines.

We separately compare SpherePair with end-to-end DCC and Euclidean constraint embedding methods to highlight its advantage in scenarios with unknown K.

Comparison with end-to-end DCC. Unlike our deep constraint embedding approach, which allows direct geometric inference or post-clustering inference via rapid sweeping over K on pre-learned representations, all end-to-end anchor-based DCC baselines require training a new model from scratch for each candidate K. This makes them impractical for such estimation due to the time-intensive nature of retraining (see learning efficiency in Table 4 for training times corresponding to a specific K).

Comparison with Euclidean constraint embedding. Under the same setup as in Figs. 15 and 16, we further apply the two post-clustering K-inference strategies to representations learned by the Euclidean constraint embedding baseline, AutoEmbedder, and report the results in Figs. 17 and 18. Comparing these with Figs. 15 and 16 highlights the applicability of different learned representations to K-inference. Over 40 cases per setting (8 datasets $\times$ 5 runs), AutoEmbedder's representations consistently struggle at all constraint levels, failing under both "K-means + SC" (39/40, 28/40, 25/40 failures for 1k, 5k, 10k constraints, respectively) and "Agglomerative + K-cluster lifetime" (40/40, 32/40, 30/40 failures). This indicates that AutoEmbedder produces suboptimal embeddings that are not sufficiently structured to support reliable cluster-number inference, underscoring the superiority of our SpherePair-based approaches.

In summary, SpherePair proves highly applicable to real-world scenarios with unknown cluster numbers. By separating representation learning from clustering, it avoids the heavy retraining cost required by end-to-end DCC methods. By producing geometrically well-structured representations that remain clustering-friendly, SpherePair enables both reliable PCA-based K-inference and effective post-clustering validation.

F.4 Empirical validation and hyperparameter sensitivity analysis

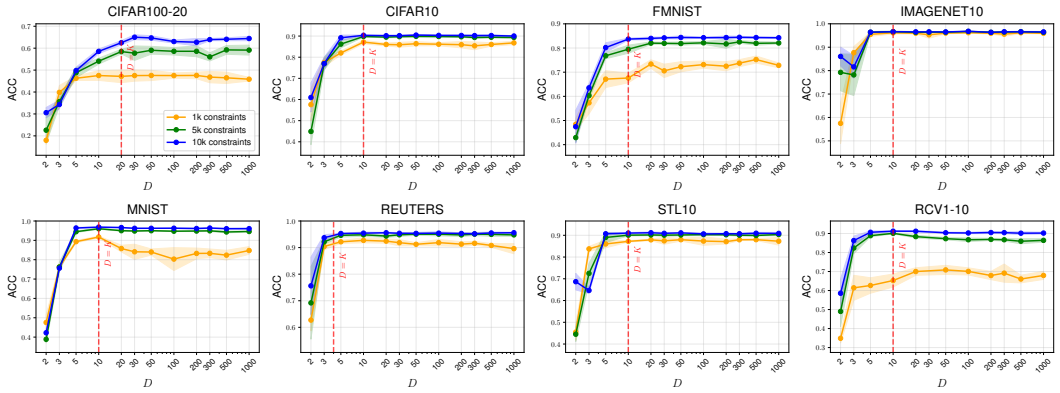
We supplement Sect. 5.2.4 with additional results, providing empirical validation of our theoretical insights and evaluating the robustness of our approach.

F.4.1 Embedding dimension D

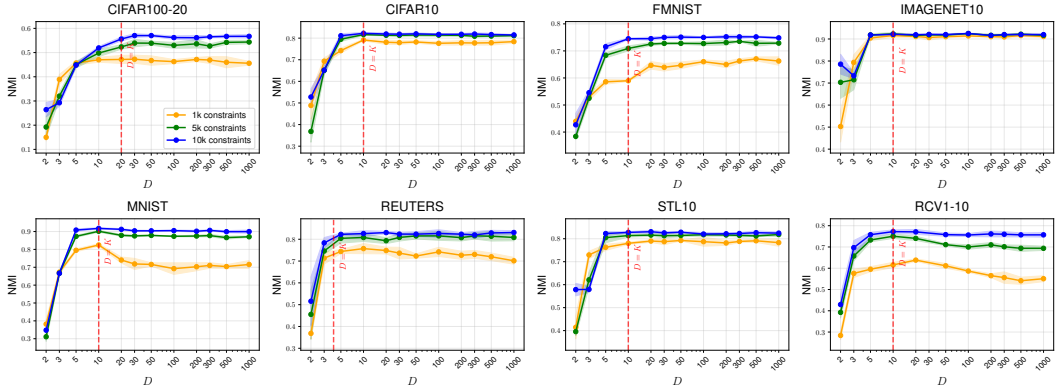
To provide a more comprehensive analysis of the impact of D and support our theoretical findings, we extend our evaluation across varying constraint set sizes (1k, 5k, 10k) and multiple clustering metrics (ACC, NMI, ARI). Fig. 19 display the clustering performance with respect to D for eight datasets under these different settings.

The results consistently demonstrate that ensuring a sufficiently large embedding dimension D achieves near-optimal or fully optimal performance across datasets, metrics, and constraint levels. Notably, the range of D values yielding optimal performance corresponds to the boundary established by our theoretical analysis in Sect. 4 ($D \geq K$, where K is the cluster number), and this correspondence becomes increasingly tight as the constraint set size grows. This alignment underscores the reliability of our theoretical framework for conflict-free constraint embedding in angular space, and provides clear practical guidance for hyperparameter selection.

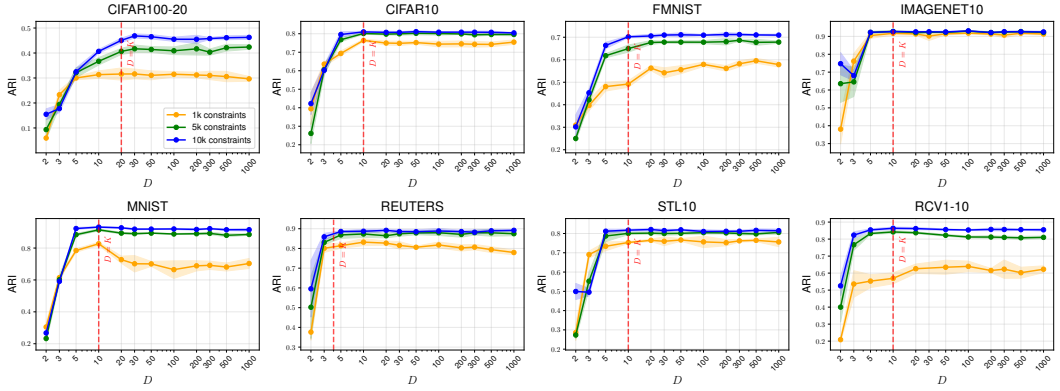
Furthermore, we observe that settings slightly below the theoretical threshold (i.e., $D \leq K - 1$) do not noticeably affect SpherePair's performance, offering useful flexibility when the number of clusters is unknown. This is further supported by Table 3: despite the baselines having no theoretical



(a)



(b)



(c)

Figure 19: Impact of embedding dimension D on SpherePair performance (mean$\pm$std over 5 runs) across datasets under 1k/5k/10k constraints: (a) test ACC, (b) test NMI, and (c) test ARI. The red lines indicate the theoretical boundary between insufficient and sufficient D.

Table 3: Clustering performance (%) (ACC, NMI, ARI) on CIFAR-100-20 for models with varying embedding dimensions and 1k/5k/10k constraints. Blue and black represent training and test performance, respectively. The best results are in **bold**, and the second-best are underlined.

	1k			5k			10k		
	ACC	NMI	ARI	ACC	NMI	ARI	ACC	NMI	ARI
VanillaDCC	34.2, 34.3	36.0, 36.3	19.3, 19.3	47.4, 47.4	46.7, 47.1	32.2, 32.2	54.6, 54.5	50.2, 50.3	37.9, 37.6
VolMaxDCC	20.1, 20.3	21.4, 21.6	7.1, 7.2	42.8, 42.8	41.9, 42.1	22.8, 22.8	51.2, 51.0	48.5, 48.7	33.4, 33.3
CIDEC (10D)	46.2, 45.4	**47.7**, 47.8	29.7, 29.1	48.4, 47.7	48.3, 48.7	32.2, 31.4	49.7, 48.8	49.0, 49.1	33.8, 32.6
CIDEC (20D)	46.6, 46.2	47.3, 47.9	30.0, 29.9	46.7, 46.1	45.4, 45.7	30.3, 29.6	50.9, 50.1	48.5, 48.8	34.0, 33.0
CIDEC (30D)	45.2, 44.5	46.4, 46.9	30.0, 29.8	48.0, 47.4	46.9, 47.2	32.2, 31.6	49.2, 48.7	48.3, 48.8	33.9, 33.6
DCGMM (10D)	44.2, 43.6	45.1, 45.6	28.2, 28.3	46.6, 46.5	46.0, 46.4	31.0, 30.8	49.0, 48.7	47.8, 48.1	33.9, 33.7
DCGMM (20D)	44.5, 44.2	44.9, 45.4	28.7, 28.7	48.1, 47.9	46.7, 47.1	32.2, 32.2	52.3, 52.1	49.2, 49.6	36.7, 36.7
DCGMM (30D)	45.0, 44.9	46.4, 46.8	29.0, 29.1	45.2, 45.2	46.2, 46.8	30.1, 30.1	46.9, 46.7	47.1, 47.4	31.8, 31.6
SDEC (10D)	45.2, 44.9	46.9, 47.2	28.0, 28.2	45.5, 45.6	47.9, 48.5	29.1, 29.3	45.9, 45.6	48.2, 48.8	30.0, 30.1
SDEC (20D)	45.7, 45.4	47.0, 47.5	29.0, 29.2	45.6, 45.1	47.0, 47.5	29.2, 29.3	45.7, 45.2	47.1, 47.7	29.3, 29.5
SDEC (30D)	45.0, 44.3	45.9, 46.3	28.4, 28.5	45.2, 44.7	46.3, 46.8	29.0, 29.2	45.3, 44.7	46.7, 47.2	29.1, 29.2
AutoEmbedder (10D)	29.4, 29.2	31.8, 32.0	12.4, 12.3	29.0, 28.9	35.0, 35.3	18.1, 18.1	39.8, 39.7	42.1, 42.4	27.2, 27.1
AutoEmbedder (20D)	21.5, 21.6	23.1, 23.4	7.1, 7.1	13.8, 14.2	13.5, 13.8	4.7, 4.7	31.3, 31.3	36.6, 36.9	20.6, 20.4
AutoEmbedder (30D)	33.9, 34.0	34.5, 35.2	17.6, 17.8	26.4, 26.2	30.9, 31.3	15.2, 15.1	33.8, 33.8	40.3, 40.3	25.3, 25.1
SpherePair (Ours) (10D)	46.9, 46.5	46.0, 46.3	31.0, 30.9	54.6, 54.2	49.6, 49.8	37.9, 37.5	57.9, 57.5	51.5, 51.6	41.4, 41.1
SpherePair (Ours) (20D)	48.3 , **48.2**	**47.7** , **48.0**	**32.2** , **32.4**	**59.0** , **58.8**	52.6 , 53.0	41.0 , 40.9	62.8 , 62.6	55.1 , 55.5	45.3 , 45.2
SpherePair (Ours) (30D)	**48.4** , **48.2**	46.8 , 47.2	31.4 , 31.5	58.4 , 58.4	**53.3** , **53.8**	**41.7** , **41.9**	**64.4** , **64.3**	**56.6** , **56.9**	**46.8** , **46.5**

restriction on D, SpherePair still outperforms them on CIFAR-100-20 even at $D = 10$ (below the theoretical threshold $K = 20$), underscoring the practical flexibility of using slightly sub-threshold D. Although a sufficiently large D is generally beneficial, we note a minor exception on MNIST under 1,000 constraints: while $D = 10$ yields peak results, increasing D beyond 10 leads to a more pronounced drop in clustering quality. This may reflect the broader impact of large embedding dimensions and the resulting overparameterization on deep neural networks, an effect that carries both negative [58, 59, 60] and positive [61, 62] implications and has been a long-standing subject in deep learning research; with larger constraint sets, however, this adverse effect diminishes, allowing higher-dimensional embeddings to continue enhancing performance.

In summary, these results confirm that respecting the theoretically derived boundary for the embedding dimension D leads to consistently strong clustering performance. In practice, choosing a sufficiently large $D \geq K$ offers a simple yet effective rule, enabling scalable and efficient solutions. This is particularly advantageous in scenarios where the exact number of clusters K is unknown, as the theoretical insights offer robust guidance for parameter selection in diverse real-world applications.

F.4.2 Regularization strength λ

The regularization strength λ governs the trade-off between the reconstruction loss and the angular constraint loss in SpherePair's objective function. We evaluate SpherePair†/SpherePair across a wide range of λ values under varying constraint levels (1k, 5k, and 10k), reporting test ACC, NMI, and ARI. The detailed results are shown in Figs. 20 and 21 for both scenarios, without and with pretraining, respectively.

The results demonstrate that SpherePair is generally robust to changes in λ, with only modest performance variations, except on RCV1-10 where pretraining combined with overly large unsupervised regularization amplifies the negative effect of severe class imbalance. Apart from this exception, we observe that the sensitivity to λ becomes more pronounced when the number of constraints is small, particularly in scenarios with random initialization (i.e., SpherePair† without pretraining), and this effect is most noticeable on CIFAR-100-20, MNIST, and Reuters. In these cases, selecting an inappropriate λ may lead to suboptimal clustering results due to the insufficient supervision provided by the small constraint sets.

Despite this sensitivity, the results suggest using $\lambda = 0.02$ as a default setting when validation information is unavailable, as it consistently provides strong performance across most datasets and constraint sizes. If prior information on class balance is available, λ can be adapted accordingly, with larger values recommended for balanced datasets and smaller values for imbalanced datasets.

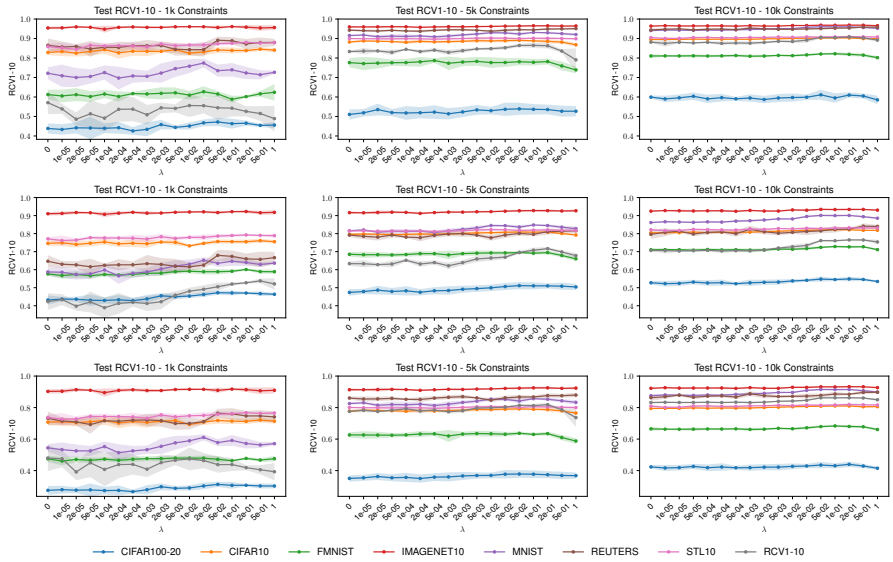


Figure 20: Performance (mean±std over 5 runs) of SpherePair† (without pretraining) across varying λ values (from 0 to 1.0).

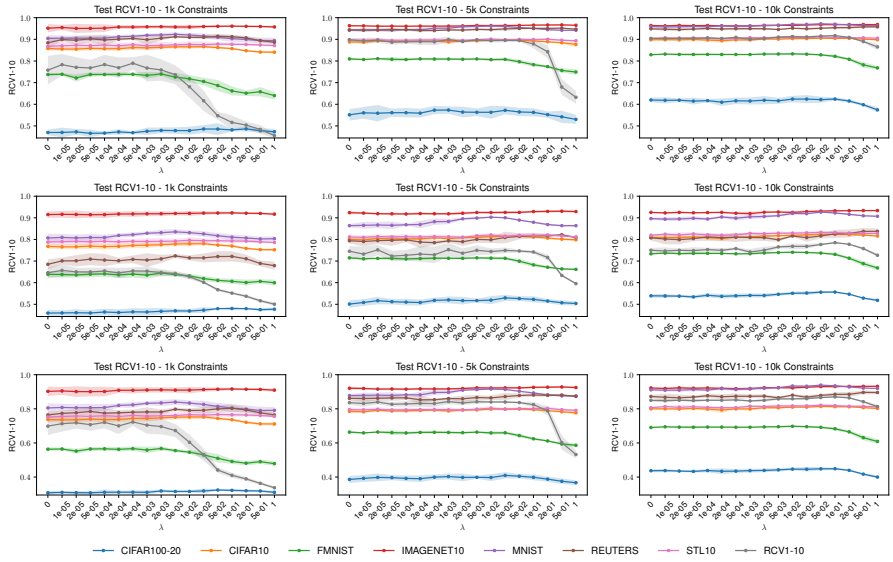
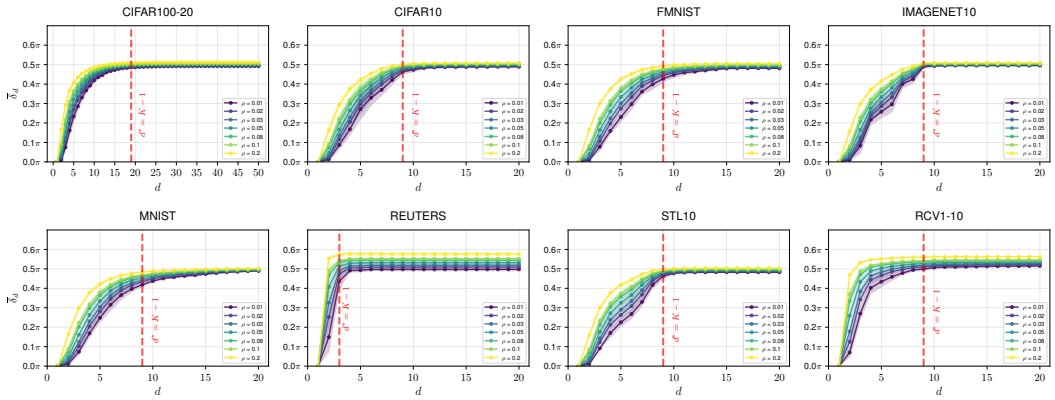


Figure 21: Performance (mean±std over 5 runs) of SpherePair (with pretraining) across varying λ values (from 0 to 1.0).

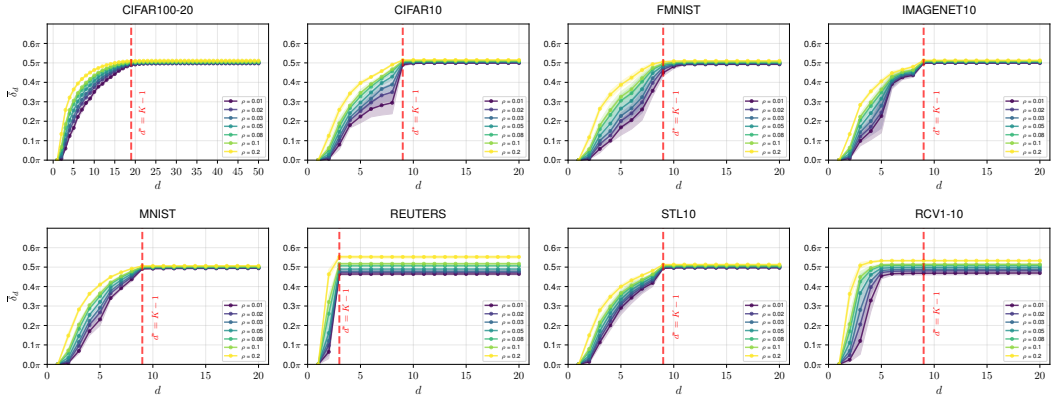
F.4.3 Tail ratio ρ

The tail ratio ρ controls the fraction of negative pairs used to compute the tail-averaged minimal inter-cluster angle in cluster-number inference. We evaluate ρ values in the range $[0.01, 0.2]$ under different constraint levels (1k, 5k, and 10k), and plot the resulting sequences $\{\overline{\delta}_d\}$ in Fig. 22.

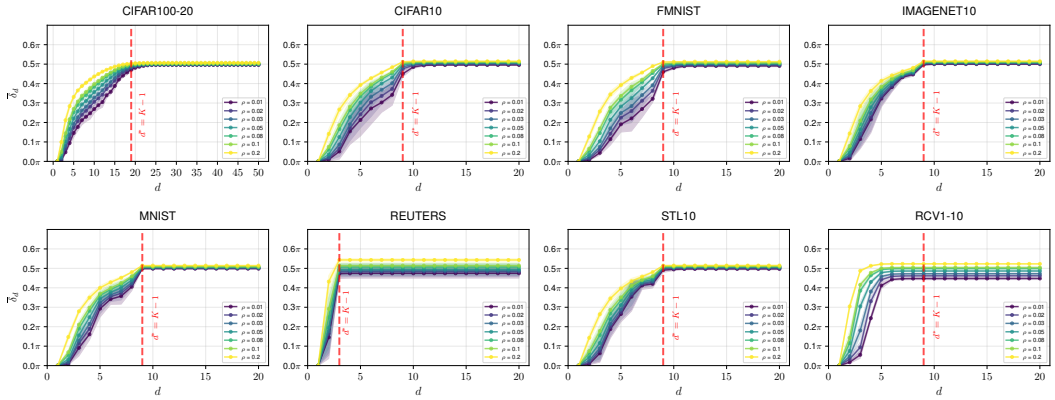
Overall, our method is robust across a broad range of ρ values, although different choices of ρ exhibit characteristic behaviors. Specifically, smaller ρ produces sharper rises before $\overline{\delta}_{K-1}$ but results in values slightly below the plateau levels, whereas larger ρ yields more consistent values when $d \geq K-1$ but makes the plateau entry less steep. Apart from the difficulty of inferring the true cluster number on RCV1-10 due to severe class imbalance, ρ within $[0.03, 0.1]$ provides a good trade-off for highlighting the plateau entry across most datasets. We therefore recommend setting ρ in this range for practical use.



(a)



(b)



(c)

Figure 22: Impact of the tail ratio ρ (from 0.01 to 0.2) on the tail-averaged minimal inter-cluster angle $\overline{\delta}_d$ across PCA subspace dimensions d (mean$\pm$std over five runs), obtained from SpherePair embeddings learned with (a) 1k, (b) 5k, and (c) 10k constraints on 8 datasets. The red vertical dashed lines indicate the ground-truth intrinsic dimension $d^* = K-1$.

G Learning efficiency

We evaluate the learning efficiency of SpherePair and DCC baselines in terms of overall training time, clustering and cluster-number inference overhead, and provide an analysis of computational complexity.

Table 4: Overall training time for different DCC methods on various datasets using 10k constraints. All times are measured on a single Tesla V100 16G GPU. Models marked with * require hyperparameter tuning, and their corresponding times are underlined.

	CIFAR100-20	CIFAR10	FMNIST	ImageNet10	MNIST	REUTERS	STL10	RCV1-10
VanillaDCC	6m23s	6m38s	7m7s	5m23s	7m32s	2m6s	5m47s	1m48s
VolMaxDCC*	32m32s	77m45s	73m57s	61m31s	70m6s	17m31s	22m7s	41m58s
CIDEC	29m13s	26m10s	41m20s	5m56s	36m9s	6m38s	6m2s	71m44s
DCGMM	21m9s	19m21s	27m55s	4m46s	21m12s	5m45s	4m47s	42m25s
SDEC	25m1s	24m33s	33m58s	5m44s	32m53s	6m27s	5m50s	61m12s
AutoEmbedder*	81m32s	77m15s	89m11s	30m11s	93m3s	36m13s	34m29s	82m39s
SpherePair (Ours)	25m31s	25m21s	33m51s	6m36s	34m22s	7m9s	6m21s	64m5s

Table 5: Clustering analysis time for anchor-free deep constraint embedding models using K-means (†) and hierarchical clustering (‡).

	CIFAR100-20	CIFAR10	FMNIST	ImageNet10	MNIST	REUTERS	STL10	RCV1-10
AutoEmbedder	18s†, 45s‡	4s†, 50s‡	6s†, 1m25s‡	1s†, 2s‡	5s†, 1m27s‡	1s†, 2s‡	1s†, 2s‡	9s†, 25s‡
SpherePair (Ours)	10s†, 47s‡	3s†, 52s‡	4s†, 1m24s‡	1s†, 2s‡	2s†, 1m23s‡	1s†, 2s‡	1s†, 2s‡	5s†, 23s‡

Table 6: Comparison of time costs for three K-inference methods across datasets, based on SpherePair embeddings learned with 10k constraints.

	CIFAR100-20	CIFAR10	FMNIST	ImageNet10	MNIST	REUTERS	STL10	RCV1-10
K-means + SC	3m14s	30s	39s	18s	34s	20s	18s	1m7s
Agglomerative + lifetime	47s	52s	1m24s	2s	1m23s	2s	2s	35s
PCA-based (Ours)	3s	1s	1s	1s	1s	1s	1s	2s

Overall training time. Based on our resources and the implementation outlined in Appendix E.4, we report the training durations of various DCC methods using 10,000 constraints across multiple datasets. Table 4 summarizes the average overall training time for each model, including both the hyperparameter tuning and parameter estimation phases: (i) The hyperparameter tuning phase encompasses searching for optimal hyperparameter values and, if necessary, a single pretraining run on the training split (excluding validation samples) prior to the search; (ii) The parameter estimation phase involves training the model with the identified optimal hyperparameters, including any pretraining on the full training split if applicable. It is noteworthy that only VolMaxDCC and AutoEmbedder require hyperparameter tuning, while pretraining is performed for all methods except VanillaDCC and VolMaxDCC.

Clustering overhead. Additionally, we report the clustering analysis time for two deep constraint embedding models, AutoEmbedder and SpherePair, using K-means and Agglomerative clustering. Unlike other end-to-end baselines that embed clustering into the network training, these models produce clustering-friendly representations, and the time required for subsequent clustering is minimal as shown in Table 5.

Cluster-number inference overhead. When the number of clusters K is unknown, clustering validation metrics are typically employed to infer the true K. In this setting, deep constraint embedding models (e.g., AutoEmbedder and SpherePair) incur only modest overhead, as candidate K values can be swept over pre-learned representations via K-means or with a single agglomeration run. Moreover, our SpherePair further benefits from the proposed PCA-based K-inference, achieving even higher efficiency by bypassing post-clustering entirely through a direct PCA solution (see Table 6 for the time costs of different K-inference methods). In contrast, end-to-end DCC baselines must be retrained from scratch for each candidate K (see Table 4 for single-run training costs), leading to far higher computational expense.

Computational complexity analysis. Aside from the empirical results, we analyze the computational complexity of SpherePair's learning, which is theoretically governed by standard DNN operations, as well as our PCA-based K-inference, which relies on a closed-form PCA solution. Let T_{f_ϕ} and $T_{g_{\phi'}}$ denote the forward-pass costs of encoder f_ϕ and decoder $g_{\phi'}$, respectively, $|\mathcal{C}|$ the number of constraints, and $|\mathcal{X}|$ the number of instances. Then the cost of angular pairwise learning (scanning constrained instance pairs) is $\mathcal{O}(|\mathcal{C}|\, T_{f_\phi})$, and that of angular reconstruction (scanning instances) is $\mathcal{O}(|\mathcal{X}|(T_{f_\phi} + T_{g_{\phi'}}))$. The additional cost of PCA-based K-inference comes from running PCA once on the D-dimensional embeddings of instances involved in negative constraints $\mathcal{C}^-$, i.e., $\mathcal{O}(|\mathcal{C}^-|D^2)$. Notably, by avoiding the need to optimize K anchors and clustering assignments—which would incur an additional $\mathcal{O}(KD)$ cost—SpherePair enjoys lower overhead than end-to-end DCC, while its angular reconstruction cost is on par with the standard reconstruction employed in methods such as CIDEC [12], and its K-inference overhead is negligible compared to repeated clustering-based validation.

H Effect of network structure

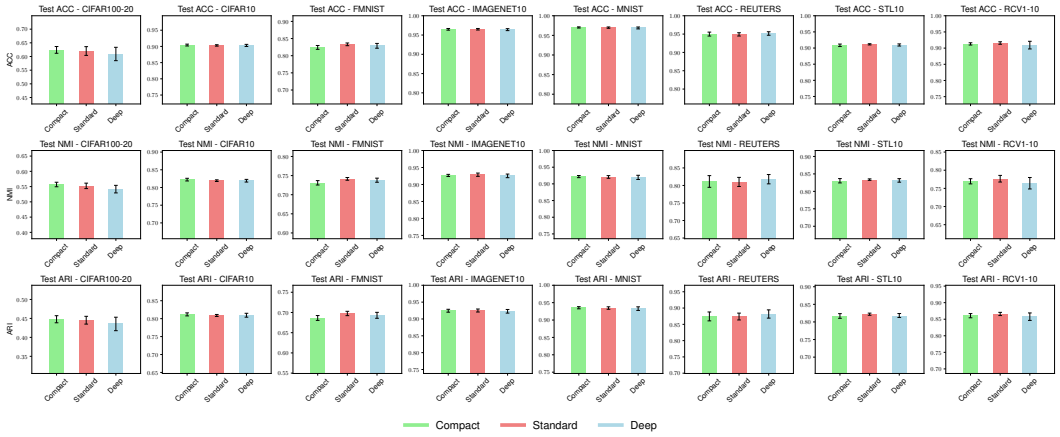


Figure 23: Performance comparison of SpherePair across three encoder-decoder structures: *Compact*, *Standard*, and *Deep*. Results (mean±std over 5 runs) are based on 10k balanced constraints, and metrics include test ACC, NMI, and ARI across multiple datasets.

To evaluate the impact of network structure on SpherePair's performance, we test three different encoder structures (paired with symmetric decoders): *Compact* (256–256–512), *Standard* (500–500–2000), and *Deep* (500–500–500–2000). Using 10k balanced constraints, we measure test ACC, NMI, and ARI on all datasets, with results summarized in Fig. 23.

The results indicate that SpherePair's performance is largely robust to the choice of network structure, with only minor differences observed across datasets. For instance, the *Compact* network performs slightly better on CIFAR-100-20, CIFAR-10, and MNIST, while the *Standard* network achieves the best results on FashionMNIST, ImageNet-10, STL-10, and RCV1-10. The *Deep* network performs marginally better on Reuters. These variations suggest that while specific structures may provide slight advantages for certain datasets, SpherePair maintains high clustering quality across all structures.

Given the observed consistency, we recommend the *Standard* structure (500–500–2000) as a practical default choice due to its balanced performance and moderate complexity. However, for real-world applications, selecting the optimal structure based on the target dataset and computational resources can further enhance performance.