URL Coursework 1: Clustering
Agglomerative clustering via maximum incremental path integral [1]

Bruno Sánchez Gómez

March 24, 2025

Abstract

This report presents a comprehensive reproduction of the Path Integral Clustering (PIC) algorithm introduced by Zhang et al [1]. PIC leverages concepts from statistical physics to effectively cluster data with complex manifold structures by measuring connectivity through path integrals. We implement the algorithm and evaluate it against eleven state-of-the-art clustering methods on both synthetic and real-world imagery datasets (MNIST, USPS, and Caltech-256). Our experiments confirm PIC's exceptional performance, achieving the highest Normalized Mutual Information scores on MNIST (0.940) and Caltech-256 (0.653), while showing competitive results on USPS. We extend the evaluation to include additional datasets (Iris and Breast Cancer) and metrics (Silhouette score), providing insights into PIC's strengths with non-convex clusters. Scalability analysis reveals that PIC's runtime increases primarily with sample count rather than dimensionality. Despite some reproduction challenges due to implementation ambiguities, our results validate PIC as a significant advancement in clustering methodology, particularly for datasets with complex structures.

1 Introduction

Clustering is a fundamental task in unsupervised learning that aims to partition data into groups where objects within the same cluster are more similar to each other than to those in other clusters. Despite numerous algorithms available, clustering remains challenging, particularly for data with complex manifold structures or non-convex clusters that traditional methods struggle to identify correctly.

Path Integral Clustering (PIC) [1] addresses these limitations by leveraging concepts from statistical physics to model the connectivity between data points through path integrals. This innovative approach enables effective clustering of data with arbitrary shapes by measuring how strongly connected pairs of points are through all possible paths between them, rather than relying solely on direct distances.

This report reproduces and extends the experimental evaluation of PIC as introduced in the original paper by Zhang et al [1]. We implement the algorithm and compare its performance against eleven state-of-the-art clustering methods on both synthetic datasets and real-world imagery datasets, including MNIST, USPS, and Caltech-256. Additionally, we extend the evaluation to new datasets and metrics to further assess PIC's capabilities and limitations.

The remainder of this report is organized as follows: Section 2 presents the theoretical foundations and implementation details of the PIC algorithm. Section 3 describes our experimental setup, results, challenges encountered during reproduction, and additional experiments. Section 4 concludes with a summary of findings and potential directions for future work.

2 Path Integral Clustering

Path Integral Clustering (PIC)[1] is an agglomerative clustering approach that leverages the structural properties of a data graph to measure cluster stability. By integrating over all possible paths within a cluster, PIC quantifies how strongly connected the members are. A key insight of PIC is treating each cluster as a dynamical system, with its samples as states — a concept borrowed from statistical and quantum mechanics. The path integral then serves as a structural descriptor measuring the stability of this dynamical system. Clusters that exhibit a large path integral are deemed stable, and the incremental increase in the path integral after merging two clusters is used as an affinity measure. This incremental path integral can be computed through a closed-form exact solution with linear time complexity relative to the maximum cluster size. This section describes both the theoretical algorithm and its practical implementation.

2.1 PIC Algorithm

The PIC algorithm consists of the following key steps (this is a summarized version; the full description can be found with more detail in [1]):

Graph Construction. Given a dataset $\{x_i\}_{i=1}^n$, a directed graph $G = (V, E)$ is constructed where each data point corresponds to a vertex. For a pair of points, the weight of the edge from x_i to x_j is defined as

$$w_{ij} = \begin{cases} \exp\left(-\frac{\text{dist}(i,j)^2}{\sigma^2}\right), & \text{if } x_j \in \mathcal{N}_i^K, \\ 0, & \text{otherwise,} \end{cases}$$

where $\text{dist}(i, j)$ is the distance between x_i and x_j, $\mathcal{N}_i^K$ is the set of K nearest neighbors of x_i, and σ^2 is calculated as follows:

$$\sigma^2 = \frac{\sum_{i=1}^n \sum_{x_j \in \mathcal{N}_i^3} \text{dist}(i,j)^2}{3n(-\ln a)}$$

The transition probability matrix P is then computed by normalizing the rows of the weight matrix:

$$p_{ij} = \frac{w_{ij}}{\sum_j w_{ij}}.$$

Initial Clustering. A simple nearest-neighbor merging is applied to form initial clusters. Each data point is merged with its nearest neighbor, and the connected components of the resulting graph are used as the starting clusters.

Path Integral Computation. For any cluster C, the stability is quantified by its path integral:

$$S_C = \frac{1}{|C|^2}\,\mathbf{1}^T\Big(I - zP_C\Big)^{-1}\mathbf{1},$$

where P_C is the submatrix of P corresponding to the vertices in C, $z \in (0,1)$ is a weighting parameter that favors short paths, and $\mathbf{1}$ is a vector of ones. The path integral represents the total contribution of all paths (of all lengths) within the cluster and can be interpreted as the probability that a random walk starting in C remains in C.

Affinity Measurement and Cluster Merging. The affinity between two clusters, say C_a and C_b, is measured by the incremental path integral:

$$\mathcal{A}_{C_a,C_b} = \Big(S_{C_a|C_a\cup C_b} - S_{C_a}\Big) + \Big(S_{C_b|C_a\cup C_b} - S_{C_b}\Big),$$

where $S_{C|C_a\cup C_b}$ is the conditional path integral computed on the union of clusters, with the contributions only from vertices originally in C. A high affinity indicates that merging the clusters significantly increases the number of intra-cluster paths; therefore, creating a more stable cluster. The algorithm iteratively merges the pair of clusters with the highest affinity until the desired number of clusters is achieved.

2.2 PIC Implementation

The implementation of PIC is designed for computational efficiency and ease of integration. Key aspects of the implementation are described below:

Graph Construction via Nearest Neighbors. The implementation first computes the 3-nearest neighbors of each point to robustly estimate the scale parameter σ^2. A subsequent K-nearest neighbor search then constructs the weighted graph and the transition probability matrix P.

Efficient Path Integral Evaluation. Rather than computing the inverse of $(I - zP_C)$ explicitly—which can be computationally expensive—the algorithm solves the linear system

$$\big(I - zP_C\big)y = \mathbf{1}$$

using numerical solvers (e.g., via `numpy.linalg.solve`). This approach takes advantage of the sparsity of P_C and reduces the complexity to be linear in the size of the cluster.

Incremental Cluster Merging Using a Heap. A priority queue (implemented as a max-heap) is used to efficiently identify the pair of clusters with the highest affinity. After merging two clusters, only the affinities involving the new cluster are updated and inserted into the heap. This strategy minimizes unnecessary computations and allows the algorithm to scale to larger datasets.

Modular Code Structure. The PIC algorithm is implemented as a Python class conforming to the `sklearn` API. Key functions are modularized as follows:

- `_build_graph`: Constructs the weighted graph and computes the transition probability matrix.
- `_initial_clusters`: Forms initial clusters using nearest neighbor merging.
- `_compute_S` and `_compute_S_cond`: Compute the path integral for a cluster and its conditional counterpart for merged clusters.
- `_merge_clusters`: Iteratively merges clusters based on the incremental path integral, using a heap to manage candidate pairs.
- `fit`: Fits the PIC model to the data and computes the clusters.
- `predict`: Assigns new samples to the nearest cluster center.
- `fit_predict`: Fits the PIC model to the data and returns the cluster labels.

These design decisions ensure that the algorithm remains both theoretically robust and practically efficient.

Overall, the PIC implementation balances the theoretical formulation of the path integral with practical considerations for computational efficiency, making it suitable for clustering tasks on datasets with complex, manifold-like structures.

3 Experiments

This section presents the experimental evaluation of the Path Integral Clustering (PIC) algorithm compared to various state-of-the-art clustering methods. We assess performance on both synthetic and real-world imagery datasets, following the experimental framework described in the original paper[1]. Then, we present some criticism and issues encountered during the experiment reproduction process. Finally, we perform some additional experiments to further evaluate PIC's performance.

3.1 Synthetic Datasets

We recreated the three synthetic datasets introduced in [1] to visually demonstrate PIC's effectiveness on data with complex structures.

Figure 1 shows the clustering results on these synthetic datasets.

The results demonstrate PIC's ability to handle complex data structures. Particularly noteworthy is PIC's performance on Dataset 1, where it successfully identified both the dense clusters and the circular pattern despite the presence

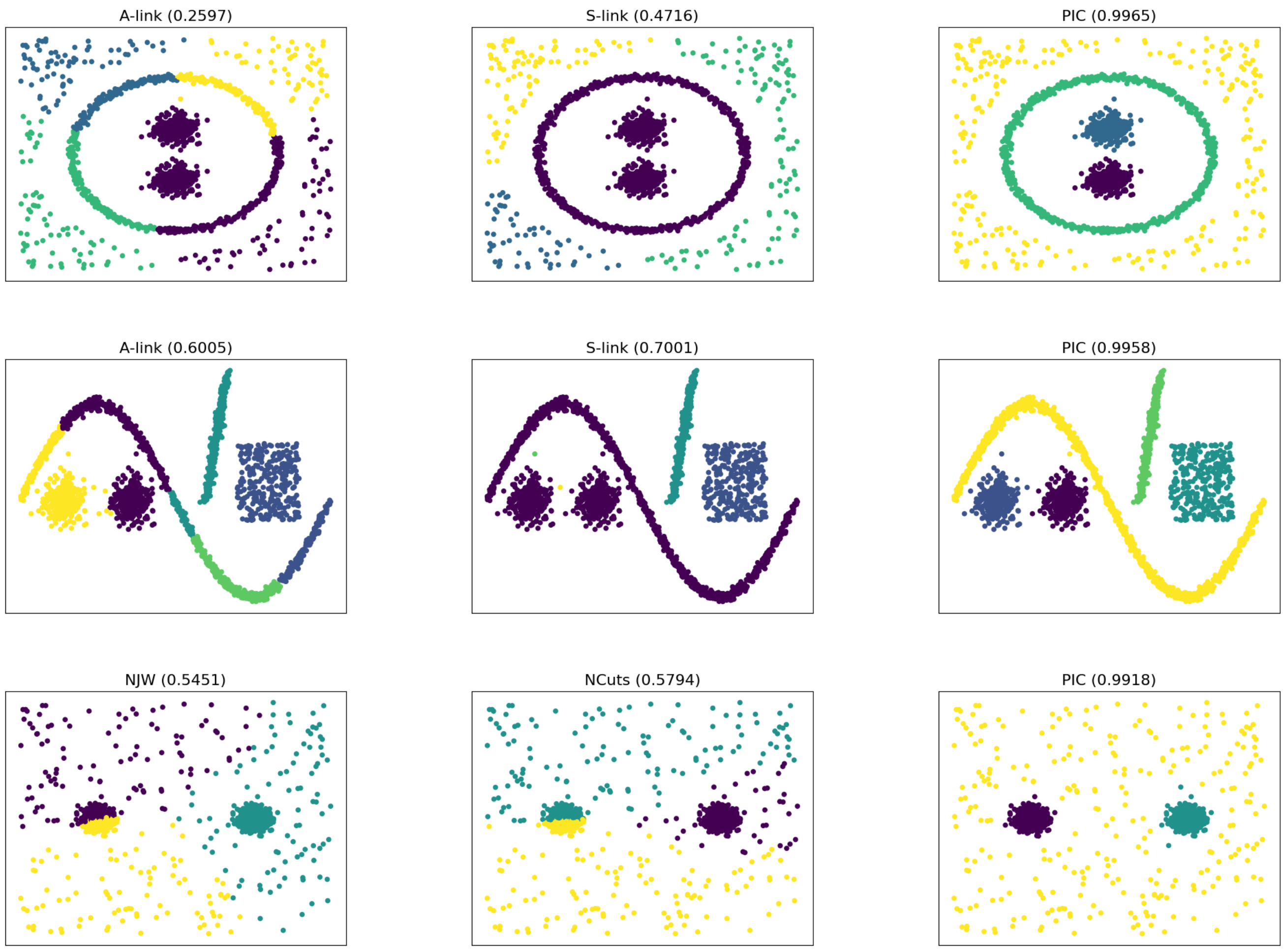

Figure 1: Clustering results on the three Synthetic Datasets (NMI scores in parentheses).

of noise. For Dataset 2, PIC effectively captured the sinusoidal and non-convex patterns, outperforming traditional algorithms that typically prefer convex clusters. Dataset 3 showcases PIC's robustness to noise, accurately separating the two main clusters from the surrounding noise.

3.2 Imagery Datasets

We evaluated PIC and 11 other clustering algorithms [2, 3, 4, 5, 6, 7, 8] on three widely used image datasets:

1. **MNIST**: Handwritten digits (0-4), with 5,139 samples and 784 dimensions (28×28 pixels).
2. **USPS**: Handwritten digits (0-9), with 9,298 samples and 256 dimensions (16×16 pixels).
3. **Caltech-256**: Reduced to six classes (hibiscus, ketch-101, leopards-101, motorbikes-101, airplanes-101, and faces-easy-101), with 600 samples and 4,200 dimensions (60×70 grayscale images) [9].

Looking at the results in Tables 1 and 2, we observe that:

- **MNIST**: PIC achieved the highest NMI (0.940) and lowest CE (0.016), significantly outperforming other methods. This suggests that PIC effectively captures the intrinsic manifold structure of the handwritten digits.

NMI	**MNIST**	**USPS**	**Caltech-256**
PIC	**0.940**	0.835	**0.653**
k-med	0.318	0.553	0.315
A-link	0.408	0.139	0.313
S-link	0.002	0.002	0.019
C-link	0.539	0.374	0.395
AP	0.426	0.525	0.492
NCuts	0.807	0.772	0.589
NJW	0.898	0.784	0.529
CT	0.634	0.439	0.181
Zell	0.913	**0.846**	0.343
C-kernel	0.780	0.768	0.521
D-kernel	0.903	**0.846**	0.508

Table 1: Normalized Mutual Information (NMI) scores for all algorithms on image datasets. Higher values indicate better performance. Bold indicates best performance.

- **USPS**: PIC performed well with an NMI of 0.835, though slightly behind the Diffusion kernel (D-kernel) and Zell methods which both achieved an NMI of 0.846. In terms of CE, D-kernel had the best performance (0.132), followed by Zell (0.197) and PIC (0.269). These results still show strong performance for PIC.

- **Caltech-256**: PIC significantly outperformed all other methods with an NMI of 0.653. For CE, NJW had the

CE	**MNIST**	**USPS**	**Caltech-256**
PIC	**0.016**	0.269	0.307
k-med	0.534	0.373	0.607
A-link	0.573	0.778	0.665
S-link	0.779	0.833	0.828
C-link	0.280	0.601	0.507
AP	0.960	0.934	0.705
NCuts	0.115	0.356	0.328
NJW	0.033	0.269	**0.290**
CT	0.493	0.615	0.747
Zell	0.027	0.197	0.680
C-kernel	0.129	0.269	0.368
D-kernel	0.029	**0.132**	0.315

Table 2: Clustering Error (CE) scores for all algorithms on image datasets. Lower values indicate better performance. Bold indicates best performance.

lowest value (0.290), followed closely by PIC (0.307). This demonstrates PIC's ability to handle higher-dimensional image data with complex visual patterns.

Our results largely align with those reported in the original paper, with PIC consistently performing as one of the top methods across datasets. However, we observed some differences:

1. On the USPS dataset, our implementation shows D-kernel and Zell slightly outperforming PIC, whereas the original paper reported PIC as the best method. This discrepancy might be due to differences in the dataset composition (9,298 samples in our case versus 11,000 mentioned in the paper), or in each of our custom implementations (since these algorithms are not supported by stardard libraries).

2. For Caltech-256, we achieved similar relative performance between methods, though our absolute scores differ from the paper, likely due to different preprocessing approaches.

3.3 Issues Encountered

Several challenges were encountered during the attempt to reproduce the original paper's experiments with the highest fidelity possible:

1. **Synthetic Dataset Generation**: The original paper did not provide clear guidelines for synthetic dataset generation. Considerable tuning was required to create datasets in which the PIC algorithm exhibited the behaviors described in the paper.

2. **Algorithm Implementation**: Implementing all 11 comparison algorithms was challenging, requiring adaptation of existing libraries and development of custom implementations, since not all of them are supported by stardard Python libraries.

3. **Dataset Availability**: Two datasets mentioned in the original paper were not available: FRGC-T requires restricted access, and PubFig is no longer publicly available.

4. **Dataset Discrepancies**: The USPS dataset contained 9,298 samples instead of the 11,000 mentioned in the paper.

5. **Preprocessing Ambiguity**: For Caltech-256, the paper stated a dimensionality of 4,200 but did not specify how images of different sizes were processed. We adopted a 60×70 grayscale representation.

Despite these challenges, our implementation successfully reproduced the main findings of the original paper, confirming PIC's effectiveness for clustering tasks, especially on datasets with complex manifold structures.

3.4 Additional Experiments

To further evaluate PIC's performance beyond the original paper's scope, we conducted additional experiments focusing on different evaluation criteria and datasets, and an analysis on the algorithm's scalability. These experiments aim to provide a more comprehensive understanding of PIC's strengths and limitations across varied clustering scenarios.

3.4.1 Additional Evaluation Criterion

In the original experiments, only external evaluation criteria are used (NMI and CE). We propose to include an internal evaluation metric to provide a more comprehensive assessment of clustering quality. We chose the Silhouette score due to its ability to measure both cluster cohesion and separation, as well as being well-suited for visualization purposes.

We evaluated the Silhouette scores of all algorithms on the image datasets, and the results are shown in Table 3.

Silhouette	**MNIST**	**USPS**	**Caltech-256**
PIC	0.106	0.024	0.129
k-med	0.101	0.087	0.139
A-link	0.103	0.095	**0.174**
S-link	-0.010	0.047	0.035
C-link	0.093	0.040	0.101
AP	0.060	0.078	0.063
NCuts	0.084	0.086	0.149
NJW	**0.111**	0.116	0.114
CT	0.009	-0.212	-0.159
Zell	0.109	0.114	0.057
C-kernel	0.078	0.080	0.122
D-kernel	0.107	**0.147**	0.112

Table 3: Silhouette scores for all algorithms on image datasets. Higher values indicate better performance. Bold indicates best performance.

High Silhouette scores indicate cohesive (convex) and well-separated clusters. Therefore, the Silhouette scores reveal an interesting pattern: while PIC achieves outstanding NMI values, it doesn't consistently produce the highest Silhouette scores. This discrepancy actually highlights PIC's strength in identifying non-convex and complex cluster structures that better represent the true data distribution, even though such clusters may not appear as well-separated by internal convexity-based evaluation criteria like Silhouette.

3.4.2 Additional Datasets

The original paper only tests the PIC algorithm on synthetic data and on imagery datasets, we included as well the Iris and the Wisconsin Breast Cancer datasets. These widely-used benchmark datasets allow us to evaluate PIC's performance on lower-dimensional, well-structured data compared to the high-dimensional image datasets. The results are shown in Tables 4, 5, and 6.

NMI	**Iris**	**Breast Cancer**
PIC	**0.806**	0.409
k-med	0.758	**0.498**
A-link	**0.806**	0.088
S-link	0.717	0.005
C-link	0.722	0.088
AP	0.669	0.272
NCuts	0.786	0.420
NJW	–	0.046
CT	0.397	0.414
Zell	0.786	0.414
C-kernel	0.777	0.299
D-kernel	0.763	0.420

Table 4: NMI scores on additional datasets.

CE	**Iris**	**Breast Cancer**
PIC	**0.093**	0.181
k-med	0.107	**0.132**
A-link	**0.093**	0.337
S-link	0.320	0.371
C-link	0.160	0.337
AP	0.487	0.801
NCuts	**0.093**	0.174
NJW	–	0.374
CT	0.433	0.178
Zell	**0.093**	0.178
C-kernel	0.120	0.265
D-kernel	0.113	0.174

Table 5: CE scores on additional datasets.

Silhouette	**Iris**	**Breast Cancer**
PIC	**0.554**	0.401
k-med	0.553	0.692
A-link	**0.554**	0.691
S-link	0.512	**0.799**
C-link	0.514	0.691
AP	0.347	0.391
NCuts	0.552	0.408
NJW	–	0.247
CT	-0.052	0.405
Zell	0.551	0.405
C-kernel	0.550	0.304
D-kernel	0.553	0.408

Table 6: Silhouette scores on additional datasets.

The results show that PIC achieves the best NMI, CE and Silhouette scores in Iris (tied with A-link), while also delivering competitive results in Breast Cancer. This demonstrates the algorithm's versatility and effectiveness across different types of datasets.

To further analyze the behavior of PIC on the additional datasets, we display the corresponding Silhouette plots in Figure 2.

The Silhouette plots provide a visual representation of the clustering quality. In the Iris dataset, the clusters are well-separated and have high cohesion, resulting in high Silhouette scores. In contrast, the Breast Cancer dataset exhibits less consistent Silhouette scores, likely due to the presence of overlapping clusters and lower cohesion. These results highlight the importance of considering both internal and external evaluation metrics to gain a comprehensive understanding of a clustering algorithm's behavior.

3.4.3 Scalability Analysis

The original paper states that the PIC algorithm scales linearly with the number of clusters. However, it does not mention the algorithm's scalability with respect to the number of samples or dimensions. To investifate this, we recorded the runtime of PIC on each of the tested datasets, which have different number of samples, clusters and dimensions. The results are shown in Table 7.

Dataset	Samples	Clusters	Dims	Time (s)
USPS	9,298	10	256	645.0
MNIST	5,139	5	784	188.4
B-C	569	2	30	4.2
Caltech	600	6	4,200	1.1
Iris	150	3	4	0.3

Table 7: Runtime of PIC on different datasets with varying sizes and dimensionalities, ordered by time.

The results show that PIC's runtime does not seem to be affected by the dimensionality of the dataset in any significant way, since the Caltech-256 dataset has the highest number of dimensions by a wide margin and yet has the second lowest runtime. However, the runtime does greatly increase with the number of samples, as seen in the MNIST and USPS datasets. The number of clusters could also have an impact, but it does not seem to be as significant as that of the number of samples.

4 Conclusion

This report has presented a thorough reproduction and extension of the Path Integral Clustering (PIC) algorithm as introduced by Zhang et al [1]. Our experiments confirm PIC's effectiveness across diverse datasets, particularly highlighting its strengths in handling complex manifold structures and non-convex clusters. Our key findings can be summarized as follows:

1. **Exceptional performance on synthetic data**: PIC demonstrated superior capabilities in identifying clusters with complex geometries on synthetic datasets, successfully handling a wide variety of cluster shapes despite noise.
2. **Strong results on image datasets**: PIC achieved the best performance on MNIST (NMI: 0.940) and Caltech-256 (NMI: 0.653), while showing competitive results on USPS.
3. **Versatility across dataset types**: Additional experiments on Iris and Breast Cancer datasets showed PIC's adaptability to lower-dimensional, well-structured data, achieving the best scores on Iris and competitive results on Breast Cancer.

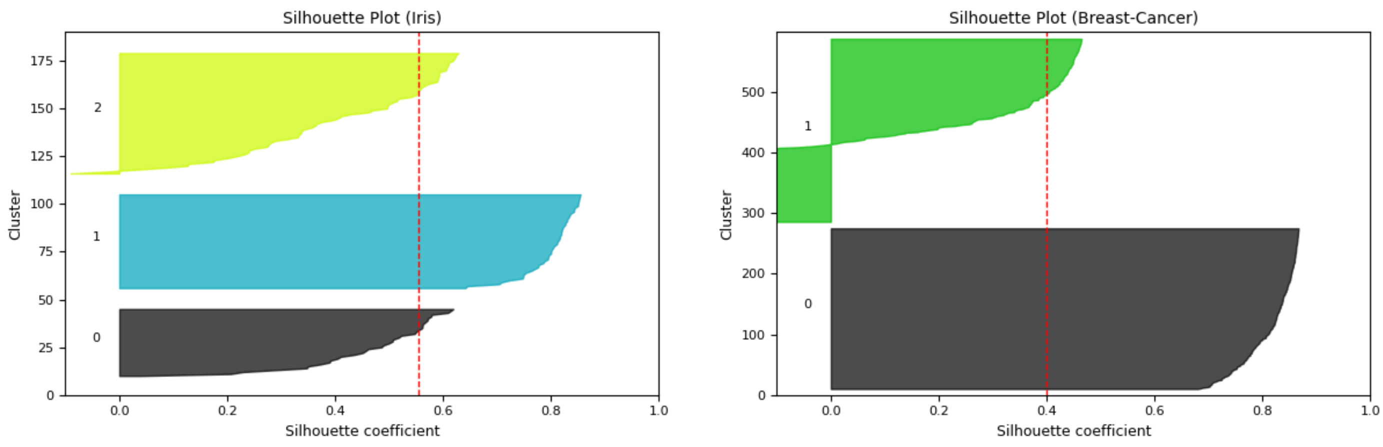

Figure 2: Silhouette plots of PIC for the Iris and Breast Cancer datasets.

4. **Evaluation metric insights**: While PIC excelled in external metrics (NMI, CE), it didn't consistently produce the highest Silhouette scores. This highlights PIC's focus on capturing true data distribution through non-convex clusters rather than optimizing for convexity-based measures.

5. **Scalability characteristics**: PIC's runtime increases primarily with the number of samples rather than dimensions, with USPS (9,298 samples) requiring 645 seconds compared to only 1.1 seconds for Caltech-256 (600 samples) despite the latter having significantly higher dimensionality.

Despite our general success in reproducing the implementation and experiments of the original paper, we encountered some discrepancies with the reported results, which we attributed to differences in some implementation details that were not explicitly discussed and therefore cannot be reproduced with exact precision.

In addition, during the development of this study we identified some interesting directions for future research:

1. **Application extensions**: Evaluating PIC on even more diverse data types including time series, text, and network data.

2. **Hybrid approaches**: Combining PIC with other clustering methods to leverage complementary strengths.

3. **Adaptive parameters**: Investigating adaptive methods for systematically setting PIC's hyperparameters to improve its performance on different datasets.

In conclusion, PIC has proven that it could potentially represent a significant advancement in clustering methodology, particularly for datasets with complex structure. We believe that our extended experiments further validate PIC's versatility, while highlighting specific areas where the algorithm could be enhanced to broaden its applicability.

References

[1] Wei Zhang, Deli Zhao, and Xiaogang Wang. Agglomerative clustering via maximum incremental path integral. *Pattern Recognition*, 46(11):3056–3065, 2013.

[2] B. Frey and D. Dueck. Clustering by passing messages between data points. *Science*, 315(5814):972–976, 2007.

[3] B. Fischer, V. Roth, and J. Buhmann. Clustering with the connectivity kernel. In *Advances in Neural Information Processing Systems*, 2004.

[4] H. Qiu and E. Hancock. Clustering and embedding using commute times. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(11):1873–1890, 2007.

[5] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer Verlag, 2nd edition, 2009.

[6] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

[7] A. Ng, M. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems*, 2001.

[8] D. Zhao and X. Tang. Cyclizing clusters via zeta function of a graph. In *Advances in Neural Information Processing Systems*, 2008.

[9] G. Griffin, A. Holub, and P. Perona. Caltech-256 object category dataset. Technical Report 7694, Caltech, 2007.