Project 2: Clustering Algorithms

Alif Al Hasan
Case Western Reserve University
axh1218@case.edu

1 INTRODUCTION

In this project, I implemented and compared three clustering algorithms: K-Means, Hierarchical Agglomerative Clustering, and DBSCAN on two gene expression datasets. I evaluated the algorithms using both external indices (ARI and Jaccard Coefficient) and internal indices (Silhouette Score), and visualized the results using PCA and t-SNE.

2 ALGORITHM DESCRIPTIONS

2.1 K-Means Clustering

K-Means is a partition-based algorithm that divides data into k clusters by minimizing the within-cluster sum of squares, which is the total distance between each point and its cluster center. This creates compact clusters where points are close to their centroids. The algorithm works as follows:

(1) Initialize k random centroids
(2) Assign each point to the nearest centroid
(3) Update centroids as the mean of assigned points
(4) Repeat steps 2-3 until convergence

My Implementation: I used silhouette analysis to automatically find the best number of clusters (k). The algorithm runs with 10 random initializations to avoid getting stuck in local optima and stops after a maximum of 300 iterations.

2.2 Hierarchical Agglomerative Clustering

Hierarchical clustering builds clusters in a tree-like structure using a bottom-up approach. It starts with each data point as its own cluster and repeatedly merges the two closest clusters until only one cluster remains. We can then cut this tree at any level to get the desired number of clusters.

My Implementation: I tested two different ways to measure distance between clusters:

- **Ward Linkage:** Merges clusters that cause the smallest increase in total within-cluster variance. This method tends to create balanced cluster sizes.
- **Complete Linkage:** Uses the maximum distance between any two points in different clusters. This creates compact clusters but is sensitive to outliers.

For this project, I set the number of clusters to match the ground truth labels (5 clusters for cho.txt and 10 clusters for iyer.txt).

2.3 DBSCAN

DBSCAN finds clusters by looking at the density of data points. It groups together points that form dense regions and marks isolated points as noise (outliers). The algorithm needs two parameters:

- ϵ (eps): How close points need to be to be considered neighbors
- *minPts*: Minimum number of points needed to form a dense region

DBSCAN classifies points into three types:

- **Core points:** Points with at least *minPts* neighbors within distance ϵ
- **Border points:** Points near core points but don't have enough neighbors
- **Noise points:** Isolated points that don't belong to any cluster

My Implementation: I tested different combinations of ϵ values (0.3, 0.5, 0.7, 1.0, 1.5, 2.0) and *minPts* values (3, 5, 7). I selected the parameter combination that gave the best Adjusted Rand Index (ARI) score when compared to ground truth labels.

3 EXPERIMENTAL SETUP

3.1 Datasets

cho.txt: Contains 386 genes with 16 features and 5 true clusters.
iyer.txt: Contains 517 genes with 12 features and 10 true clusters (including 33 outliers).
Data Preprocessing: Before clustering, I:

(1) Removed columns with constant values
(2) Filled missing values with column means
(3) Standardized all features to have mean 0 and standard deviation 1

3.2 Evaluation Metrics

I used both external indices (which compare results to ground truth) and internal indices (which measure cluster quality without ground truth).

External Indices:

- **Adjusted Rand Index (ARI):** Measures how similar the predicted clusters are to true clusters, with correction for random chances. Values range from -1 to 1, where 1 means perfect match.
- **Jaccard Coefficient:** Measures agreement between predicted and true clusters by counting pairs of points. Computed as:

$$J = \frac{a}{a+b+c}$$

where a = pairs clustered together in both, b = pairs together only in ground truth, c = pairs together only in prediction.

Internal Index:

- **Silhouette Score:** Measures how well-separated and cohesive clusters are. Values range from -1 to 1, where higher values indicate better cluster quality.

3.3 Visualization

I used two methods to visualize the high-dimensional data in 2D:

- **PCA (Principal Component Analysis):** A linear method that reduces dimensions while keeping the most important variations in the data. It shows the overall spread and structure of clusters.
- **t-SNE:** A non-linear method that focuses on keeping nearby points together. It is better at showing tight, well-separated clusters and local patterns.

4 RESULTS

4.1 Quantitative Results

Table 1: Clustering Performance on cho.txt

Algorithm	ARI	Jaccard	Silhouette
K-Means	**0.4192**	**0.4076**	**0.2462**
Hier. Ward	0.4174	0.3864	0.2007
Hier. Complete	0.2947	0.3362	N/A
DBSCAN	0.0629	0.2054	0.1391

Table 2: Clustering Performance on iyer.txt

Algorithm	ARI	Jaccard	Silhouette
K-Means	0.0213	0.1785	**0.7459**
Hier. Ward	**0.3790**	**0.3587**	N/A
Hier. Complete	0.0714	0.1993	N/A
DBSCAN	0.3113	0.3186	0.4133

Table 3: Average Performance Across Both Datasets

Algorithm	Avg ARI	Avg Jaccard	Avg Silh.
K-Means	0.2203	0.2931	**0.4961**
Hier. Ward	**0.3982**	**0.3725**	0.2007
Hier. Complete	0.1830	0.2678	N/A
DBSCAN	0.1871	0.2620	0.2762

Tables 1-3 show the clustering performance on both datasets.

4.2 Detailed Performance Analysis

Results on cho.txt: K-Means achieved the best scores on this dataset (ARI: 0.4192, Jaccard: 0.4076), just slightly better than Hierarchical Ward (ARI: 0.4174). K-Means found 3 clusters and had the highest silhouette score (0.2462), indicating good cluster quality.

However, DBSCAN performed poorly (ARI: 0.0629) because it marked 180 out of 386 points (46.6%) as noise using parameters $\epsilon = 2.0$ and $minPts = 5$. This aggressive noise detection caused it to miss the true cluster structure.

Hierarchical Complete linkage performed moderately (ARI: 0.2947) but had undefined silhouette score, suggesting problems with cluster separation.

Results on iyer.txt: Hierarchical Ward was the best performer on this dataset (ARI: 0.3790, Jaccard: 0.3587). It correctly identified all 10 true clusters and produced well-defined groupings.

K-Means failed dramatically on this dataset (ARI: 0.0213) because the automatic cluster selection chose only 2 clusters instead of 10. Interestingly, K-Means achieved the highest silhouette score (0.7459), which means that the 2 clusters it found were very compact and well-separated. They just didn't match the true structure.

DBSCAN performed reasonably well (ARI: 0.3113) with parameters $\epsilon = 0.7$ and $minPts = 3$, but it still labeled 199 out of 517 points (38.5%) as noise, which hurt its overall score.

4.3 Visualization Results

Figures 1-4 show the clustering results in 2D space.

For cho.txt: Both PCA and t-SNE show moderate cluster separation. K-Means and Hierarchical Ward produce similar, well-separated clusters. DBSCAN creates one main dense region but marks many points as noise. The t-SNE plots reveal tighter, more distinct clusters compared to PCA.

For iyer.txt: The visualizations reveal substantial overlap between the 10 true clusters when projected to 2D. K-Means groups all points into just 2 large clusters, completely missing the true structure. Hierarchical Ward does better in capturing the true cluster boundaries. DBSCAN's noise points appear as isolated scattered points throughout both visualizations. The t-SNE projections preserve local structure better than PCA, showing tighter sub-clusters.

5 ALGORITHM COMPARISON

5.1 K-Means

Advantages:

- Very fast and efficient
- Easy to understand and implement
- Best internal cluster quality (average silhouette: 0.4961)
- Best performance on cho.txt dataset

Disadvantages:

- Must specify number of clusters k in advance
- Automatic k selection can fail badly (chose 2 instead of 10 on iyer.txt)
- Assumes clusters are spherical and similarly sized
- Sensitive to initialization and outliers
- Cannot find irregular-shaped clusters

5.2 Hierarchical Clustering

Advantages:

- Don't need to specify number of clusters beforehand
- Always produces the same result
- Most consistent performance across both datasets
- Ward linkage gives balanced cluster sizes
- Can choose different numbers of clusters by cutting the tree at different levels

Disadvantages:

- Very slow for large datasets
- Sensitive to noise and outliers
- Choice of linkage method matters significantly (Ward: 0.3982 ARI vs Complete: 0.1830 ARI)
- Complete linkage produced negative silhouette scores

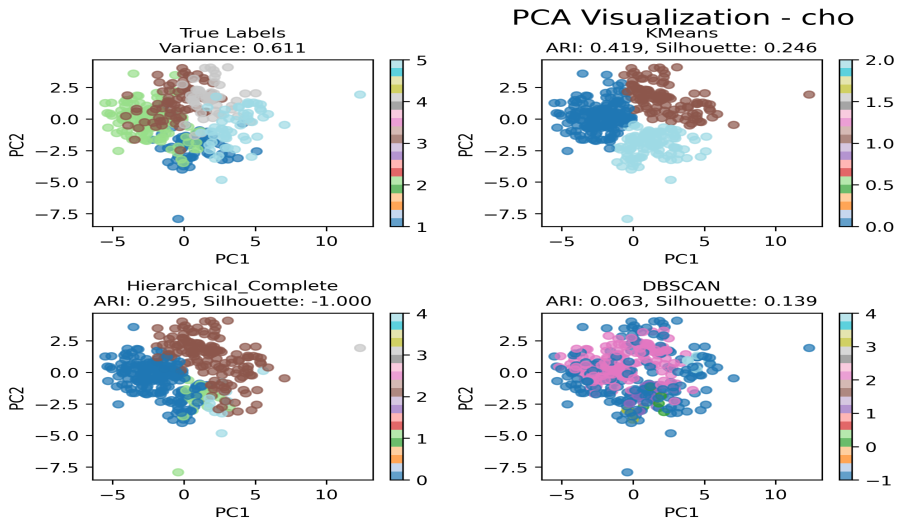

Figure 1: PCA visualization of cho.txt dataset.

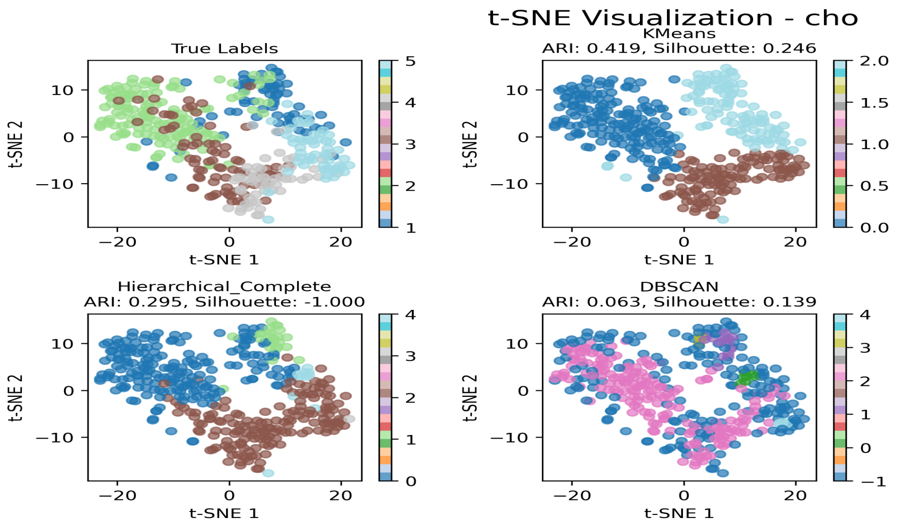

Figure 2: t-SNE visualization of cho.txt dataset.

5.3 DBSCAN

Advantages:

- Can find clusters of any shape
- Automatically determines number of clusters
- Identifies outliers as noise points (found 180-199 noise points)
- No assumptions about cluster shape or size
- Performed reasonably well on iyer.txt (ARI: 0.3113)

Disadvantages:

- Very sensitive to parameters ϵ and $minPts$
- Struggles when clusters have different densities
- Marked too many points as noise (up to 46.6% on cho.txt)
- Poor performance on cho.txt (ARI: 0.0629)
- Doesn't work well in high-dimensional spaces
- Requires extensive parameter tuning

6 DISCUSSION AND KEY FINDINGS

1. No single best algorithm: Different algorithms work better on different datasets. Hierarchical Ward was most consistent overall (average ARI: 0.3982), but K-Means was better on cho.txt (ARI: 0.4192). The best choice depends on the specific data characteristics.

2. Automatic parameter selection is risky: K-Means' automatic cluster selection failed completely on iyer.txt, choosing 2 clusters instead of 10. We need to validate results using domain knowledge and multiple metrics.

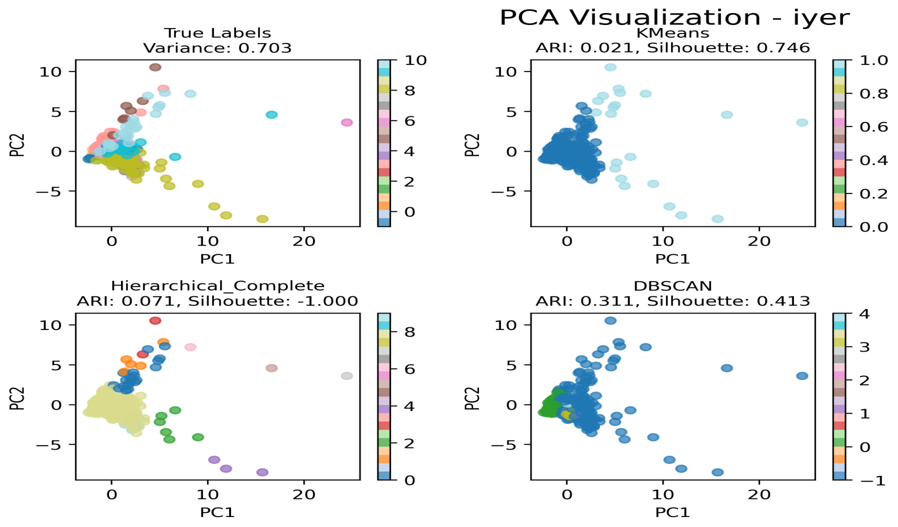

Figure 3: PCA visualization of iyer.txt dataset.

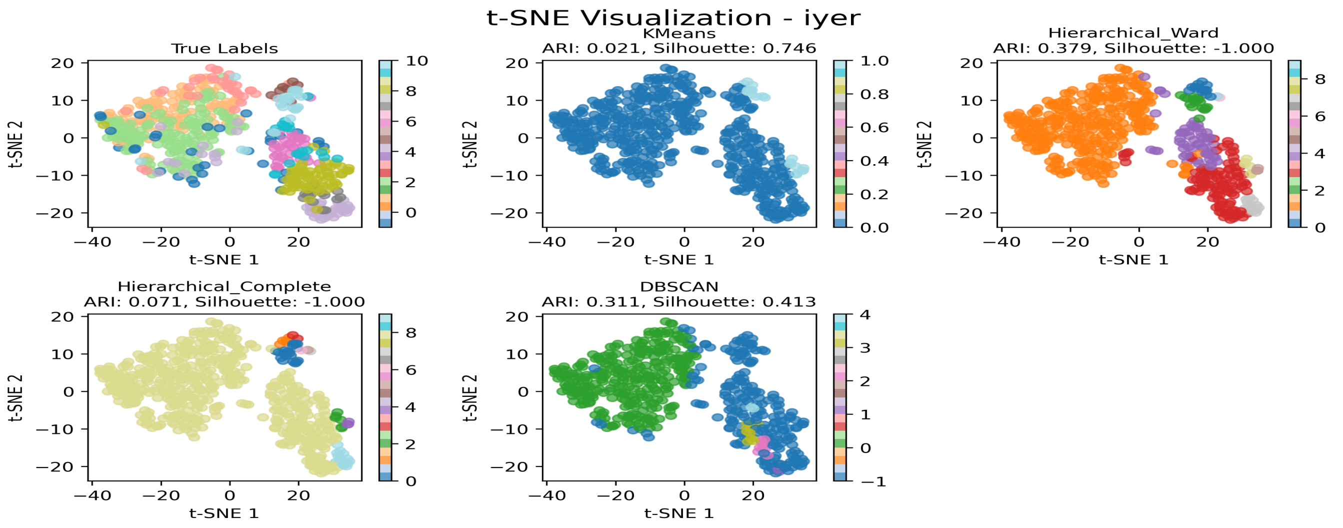

Figure 4: t-SNE visualization of iyer.txt dataset.

3. Trade-off between noise detection and accuracy: DBSCAN marked 38-47% of points as noise. The cho.txt dataset had no true outliers, and iyer.txt had only 6.4% (33/517) outliers, so DBSCAN's aggressive noise detection was excessive and reduced its accuracy scores.

4. Linkage method matters significantly: For hierarchical clustering, Ward linkage (ARI: 0.3982) performed much better than Complete linkage (ARI: 0.1830). The choice of distance measure is quite important to get good results.

5. Internal metrics can be misleading: K-Means achieved the highest silhouette score (0.7459) on iyer.txt but the lowest ARI (0.0213). This means it found 2 very compact, well-separated clusters that completely missed the true structure.

7 CONCLUSION

Hierarchical Ward linkage showed the best overall performance (average ARI: 0.3982, Jaccard: 0.3725) and was most consistent across both datasets. K-Means showed excellent performance on cho.txt (ARI: 0.4192) but failed on iyer.txt (ARI: 0.0213) due to incorrect cluster count. DBSCAN exhibited moderate performance (average ARI: 0.1871) with the advantage of automatic outlier detection. However, it marked too many points as noise (38-47%), which hurt accuracy.

Results show that algorithm selection should consider dataset characteristics and whether the number of clusters is known. We always need to validate results using both internal and external indices (if ground truth is available) and visualization techniques.