Comparative Analysis of Clustering Algorithms for Song lyrics Clustering

Arya Lesmana*
Faculty of Computer Science
Universitas Indonesia
Depok, Indonesia
arya.lesmana21@ui.ac.id

Cyrilus Yodha Maheswara*
Faculty of Computer Science
Universitas Indonesia
Depok, Indonesia
cyrilus.yodha@ui.ac.id

Mariano Gerardus Senduk*
Faculty of Computer Science
Universitas Indonesia
Depok, Indonesia
mariano.gerardus@ui.ac.id

Narendra Dzulqarnain*
Faculty of Computer Science
Universitas Indonesia
Depok, Indonesia
narendra.dzulqarnain@ui.ac.id

Jaycent Gunawan Ongris
Faculty of Computer Science
Universitas Indonesia
Depok, Indonesia
jaycent.gunawan@ui.ac.id

Timothy Orvin Edwardo
Faculty of Computer Science
Universitas Indonesia
Depok, Indonesia
timothy.orvin@ui.ac.id

Calista Vimalametta Heryadi
Faculty of Computer Science
Universitas Indonesia
Depok, Indonesia
calista.vimalametta@ui.ac.id

Andi Pujo Rahadi
Faculty of Computer Science
Universitas Indonesia
Depok, Indonesia
ap.rahadi@gmail.com

Abstract—**The rise of the internet has significantly increased access to music from different artists, genres, and cultural backgrounds. Consequently, understanding patterns and similarities in song lyrics has become an increasingly relevant task, particularly for applications such as music recommendation, genre classification, and thematic organization. This paper explores the use of unsupervised machine learning techniques to cluster song lyrics based on linguistic similarities. We conduct clustering analysis on a dataset of English-language song lyrics obtained from Kaggle, which is based on the most-viewed songs on Genius. After preprocessing the lyrics using text mining techniques including contraction expansion, tokenization, special character and digit removal, stopword and filler word removal, and lemmatization, we convert the text into vector representations using Word2Vec. We then applied three clustering algorithms, namely K-Means, Agglomerative, and DBSCAN. Based on evaluations using the Silhouette Score, we determine that two is the optimal number of clusters. Due to the high dimensionality of the data, we apply Principal Component Analysis (PCA) to reduce the data into three dimensional for easier visualization. Based on our visualization using 3-dimensional and wordcloud visualizer, we conclude that the resulting clusters can be classified as Rap/Hip-Hop and Romantic Pop songs. This study demonstrates a comparative and practical approach to clustering song lyrics and provides insights into selecting suitable unsupervised methods for textual musical data.**

Index Terms—**Unsupervised Learning, Text Clustering, lyricsal Analysis, Machine Learning, Natural Language Processing, Word2vec, Principal Component Analysis, K-Means Clustering, Agglomerative Clustering, DBSCAN, Text Mining, Data Preprocessing, and Digital Music Archives.**

I. Introduction

This introduction outlines the motivation behind analyzing song lyrics through clustering, discusses the relevance and challenges of the task, highlights related research, and frames the business understanding that guides this study within the CRISP-DM framework.

A. Background & Motivation

Music is one of the human traditions that is estimated to have existed since 40,000 years ago, it has long been a powerful medium of cultural expression and emotional communication[1]. With the widespread availability of digital music and textual lyrics, analyzing these resources can provide insights into cultural trends and dominant themes in popular music. However, large-scale lyrics analysis is challenging due to the inherent ambiguity, contextuality, and subjective nature of lyrical content, which often includes metaphors, slang, and nuanced emotional expressions.

To address these complexities, this study applies unsupervised learning methods, specifically clustering, to group songs based on lyrical similarity. This approach can reveal hidden thematic structures and relationships without requiring manual annotation. Such analysis has the potential to enhance various applications, including improving song recommendation systems, enriching streaming service experiences, and providing valuable insights for cultural studies and digital archive systems.

This study investigates the effectiveness of K-Means (a partitioning approach), Agglomerative Clustering (a hierarchical method), and DBSCAN (a density-based technique) in grouping English song lyrics. The dataset comprises 9.000 songs scraped from Genius.com.

We adopted the Cross-Industry Standard Process for Data Mining (CRISP-DM) methodology to ensure a systematic and structured approach to the research. The CRISP-DM framework consists of six stages, starting from understanding the business and data to evaluating and deploying the model. By following these stages, we hope that the research can be carried out comprehensively, measurably, and replicability.

B. Related Work

Clustering techniques have been widely applied to various forms of text data, including news articles, social media posts,

*These authors contributed equally to this work and websites. In [2], Majid et al. explained and summarized methods for short text clustering, detailing preprocessing techniques, feature transformation strategies such as TF-IDF and word embeddings (Word2Vec, Doc2Vec, GloVe), and clustering algorithms including K-Means, Agglomerative, and DBSCAN. Their work provides a solid foundation and methodology for similar research. However, the study remains theoretical and does not include empirical evaluation using real-world datasets.

Other related studies have tried to apply clustering method to the analysis of song lyrics clustering or classification. Rarasati [3] utilized the K-Means clustering algorithm to classify 400 songs into four predefined groups: love, friendship, religion, and fighting, achieving an accuracy of 93.25%. The study most closely resembling our proposed experiment is that of Gupta et al. [4], who developed a song recommendation system based on context-based semantic similarity between lyrics. Gupta et al. [4] use K-Means algorithm for clustering the songs and identified an optimal cluster configuration of ten clusters, yielding a Silhouette Score of 0.243.

To build on these works, our study investigates multiple clustering algorithms for unsupervised grouping of song lyrics based on linguistic similarity, without relying on predefined labels. These methods are chosen based on their effectiveness in text data clustering, each offering different strengths.

1) Word2Vec: Before applying the dataset to the model, the data need to get processed first by changing it to numerical representation. One way that can be done is by using Word2Vec, the technique uses shallow, two-layer neural networks trained to reconstruct linguistic contexts of words, producing dense vector embeddings where semantically similar words are positioned closer together in the vector space[7].

Word2Vec employs two primary architectures: Continuous Bag of Words (CBOW) and Skip-gram. The CBOW model predicts a target word based on surrounding context words, functioning as a "fill in the blank" task where word embeddings represent how words influence the relative probabilities of other words in the context window. In contrast, the Skip-gram architecture uses the current word to predict the surrounding context words, weighing nearby context words more heavily than distant ones[7].

2) K-Means Clustering: K-Means is a partitioning-based clustering algorithm that divides data points (in this case, song lyrics) into k distinct clusters where each data point belongs to the cluster with the nearest centroid. This algorithm was selected for our lyrics clustering task due to its computational efficiency, implementation simplicity, and proven effectiveness in text data organization.

The computational complexity of K-Means is $O(tkn)$ [3], where n represents the number of lyrics, k the number of clusters, and t the number of iterations. This relatively low complexity makes it suitable for large lyrics datasets, as the values for k and t are typically much smaller than n, and the algorithm stops under local optimum conditions

3) Agglomerative Clustering: Agglomerative is a type of hierarchical clustering algorithm, which is a cluster algorithm that will produces nested groups in the form of a hierarchy by progressively merging smaller clusters into larger ones [2]. Agglomerative uses a bottom-up approach with each data point as its own individual cluster and then iteratively combining them based on their similarity, forming a tree-like structure known as a dendrogram. The complexity for Agglomerative is $O(n^3)$[5], where n is the number of lyrics.

The process allows for the exploration of data at various levels of granularity, making it a useful method for clustering tasks such as song lyrics analysis, where relationships between data points may not be immediately apparent.

4) DBSCAN Clustering: DBSCAN is a density type clustering algorithm, which works by analyzing spatial density of the data objects to find identify the clusters. A cluster is defined as a densely linked component which grows in any direction to increase density [2]. By using this density based approach, This approach enables DBSCAN to discover clusters of arbitrary shape and to effectively identify noise or outliers—points that are not sufficiently close to any dense region. The complexity of DBSCAN is $O(n * \log n)$ where n is the number of lyrics [6].

This makes DBSCAN particularly useful where the structure of the data may not be well-defined, and there might be some isolated points that don't belong to any cluster.

II. Methodology

This section details the methodology adopted for clustering song lyrics, including business understanding, data understanding, and data preparation.

A. Business Understanding

The primary objective of this study is to evaluate and compare the effectiveness of various clustering algorithms for grouping English-language song lyrics based on their linguistic features. This research aligns with the growing importance of content-based recommendation systems in the music industry, where lyricsal content analysis represents an untapped dimension for categorization beyond traditional genres [4]. Key stakeholders include the academic community seeking advancements in text mining techniques [2], music streaming services that could enhance content organization, and musicologists who might gain insights into lyricsal patterns across different periods [1].

Expected benefits include enhanced recommendation systems based on lyricsal themes, improved content discovery for users, and more nuanced understanding of music categorization. However, challenges remain. Clustering outputs may not align with human-perceived similarity, production deployment can be computationally demanding, and focusing solely on English lyrics limits the cross-cultural applicability of the findings.

Our technical goals include implementing three distinct clustering algorithms—hierarchical, partitioned, and density-based—to compare their effectiveness in grouping songs based on lyrical similarity. We aim to determine optimal cluster counts, identify representative features defining each cluster, and analyze relationships between clusters and metadata such as release year and view count.

B. Data Understanding

Our dataset combines structured and unstructured data sourced primarily from Kaggle and supplemented via web scraping. The original dataset, titled `songs_train.csv`, contained 9,000 records representing the most-viewed songs on Genius.com. Each record included metadata fields such as `title` (string), `artist` (string), `year` (integer), `views` (integer), and `url` (string). However, the critical `lyrics` field was initially absent.

To enrich the dataset, we implemented a web scraping pipeline using Python's `BeautifulSoup` and `requests` libraries. The scraper navigated to the Genius URLs provided in the dataset and extracted only the lyricsal content, avoiding extraneous webpage elements.

After preprocessing, the resulting dataset consisted of 7,315 fully populated records (approx. 81% of the original) and included the following six attributes:

- **title** (string): The title of the song.
- **artist** (string): The name of the performing artist.
- **year** (integer): The release year of the song.
- **views** (integer): The view count of the song on Genius.com.
- **url** (string): The Genius.com URL for the song.
- **lyrics** (string): The song's lyrics, extracted via web scraping.

The lyrics field was approximately 81% complete due to missing or inaccessible pages. All other attributes are fully populated with no missing values. The dataset thus contains a mix of structured data (e.g., `year`, `views`) and unstructured text data (`lyrics`), which will be central to the clustering process.

Inspection of the `lyrics` field reveals formatting artifacts such as section headers (e.g., `[Verse 1]`, `[Intro]`, `[Chorus]`), which will need to be cleaned during preprocessing to ensure accurate text representation and clustering performance.

C. Data Preparation

We need to sanitize and normalize the raw lyrics, ensuring consistency and reducing lexical variability before applying them to machine learning algorithms. We utilzed python libraries such as NLTK [9] in the process. The process was composed of the following sequential operations:

- Missing Value Removal: Removing missing value entries was done for convenience, as this is a clustering task and the missing values would not contribute to the analysis.
- Lowercasing: All words were converted to lowercase to standardize the vocabulary and prevent case-sensitive discrepancies (e.g. Love and love).
- Bracket Removal: Square and round brackets were removed in their entirety, as they provide metastructural cues rather than meaningful lyricsal content.
- Colloquial Form Normalization: Nonstandard word forms ending in 'in" (e.g., goin', runnin') were normalized to their standard equivalents (going, running). This improves lexical alignment and reduce sparsity in feature space.
- Contraction Expansion: Contractions in the lyrics were automatically expanded to their full forms using the `contractions` Python library (e.g., *don't* $\rightarrow$ *do not*, *I'm* $\rightarrow$ *I am*). This step helps standardize the textual data and improves the performance of downstream tokenization and linguistic analysis.
- Tokenization: Texts were tokenized into individual word units, enabling word-level processing and facilitating further lexical operations such as filtering and transformation.
- Character and Digit Filtering: All nonalphabetic characters and numerical digits were removed to eliminate syntactic noise and preserve only linguistically relevant content.
- Stopword Removal: This operation reduced redundancy and emphasized informative content.
- Filler Word and Single Char Removal: A domain-specific list of filler expressions frequently found in song lyrics (e.g. ohh, yeah, uhh, hmm) was compiled and removed to clean the further. Also removed the rest of single character word since it provides small meaning [8].
- Lemmatization: This step helped consolidate different inflected forms under a unified representation. We chose lemmatization over stemming because we wanted to ensure that the resulting words exist in the dictionary thereby preserving the correct context and meaning, and make the words in the clustering result more interpretable compared to using stemmed tokens, which can often be incomplete or distorted word forms.

III. Modeling

In this section, we detail the modeling process undertaken to cluster song lyrics based on their semantic and linguistic features. We employ three distinct clustering algorithms—K-Means, Agglomerative Clustering, and DBSCAN—to provide a comprehensive comparison of different clustering paradigms: partition-based, hierarchical, and density-based approaches.

A. Test Design

In the test design phase, we established an evaluation framework combining both quantitative and qualitative assessments. A random cluster assignment was used as a baseline. The primary quantitative metric is the silhouette score, which measures how well-separated and cohesive the clusters are. For qualitative evaluation, we analyzed word clouds and term frequency distributions to assess thematic coherence. A clustering method is considered successful if it outperforms the random baseline in silhouette score and produces interpretable, thematically consistent clusters.

B. Model Building

1) Baseline Model: The song lyrics were assigned to one of two clusters randomly. We used a fixed random seed for reproducibility. The number of clusters used in this baseline (i.e., two) was chosen to match the optimal number of clusters determined through evaluation of the more advanced clustering methods, which is discussed in detail in the evaluation section.

The random clustering yielded a Silhouette Score of -0.0001. Qualitative analysis also shows a lot of similarity in both cluster by the most common words. This result confirms the absence of meaningful cluster structure. It serves as a lower bound for comparison with more structured clustering approaches.

2) K-Means Clustering: Our implementation of K-Means for lyrics clustering involves hyperparameter tuning to get the optimum cluster configuration. Specifically, we explored the number of clusters (k) in the range of 2 to 50. For each value of k, the K-Means model was trained with `n_init` set to 10 to ensure stable clustering outcomes, and a `random_state` of 42 was used for reproducibility.

In each iteration, the *silhouette score* was calculated, which is a metric that measures the quality of clustering by calculating a coefficient that reflects how similar each data point is to its own cluster compared to the nearest other cluster. The optimal value of k was determined as the value that maximizes the *silhouette score*

3) Agglomerative Clustering: The modeling pipeline consists of hierarchical linkage computation, dendrogram visualization, and silhouette-based evaluation. We examined four linkage strategies: *ward*, *complete*, *average*, and *single*. This modeling approach enabled a comparative evaluation of hierarchical clustering behavior under different structural assumptions, contributing to the selection of an appropriate linkage method for the song lyrics corpus.

4) DBSCAN Clustering: DBSCAN clustering requires careful selection of two critical parameters: epsilon (ε) and minimum points ($minPts$), which significantly influence cluster formation in high-dimensional embedding space. We implemented a systematic grid search methodology to identify optimal parameter combinations.

The search space encompassed epsilon values ranging from 0.1 to 0.45 with increments of 0.05, and minimum sample values from 3 to 19, generating 136 distinct parameter configurations. For each configuration, we recorded multiple evaluation metrics including: number of formed clusters, proportion of noise points, and silhouette score for valid configurations.

Noise points (labeled as -1 by the algorithm) were tracked separately throughout the analysis process. For evaluation purposes, we excluded these points from silhouette score calculations to prevent artificial score deflation, while still maintaining their importance in the overall clustering assessment.

IV. Evaluation

This section presents the evaluation of each clustering algorithm, focusing on optimal cluster configuration, silhouette score analysis, and content interpretation.

A. K-Means Clustering

Based on the hyperparameter tuning process, the final model configuration was selected with `n_cluster=2`, `n_init=10`, and `random_state=42` which provided the highest silhouette score of 0.1684. Although this silhouette score is relatively low, it is consistent with the inherent complexity of text data clustering, where high overlap between clusters is common. We applied PCA with 3 components so that we can visualize the cluster in 3 dimensional axis. We also visualize wordcloud to see which word that belongs to the cluster.

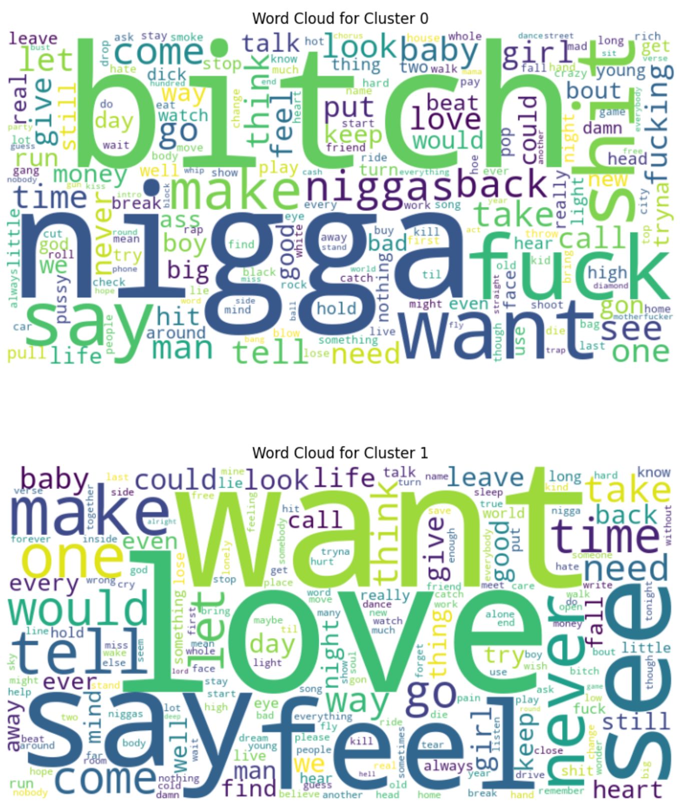


Fig. 1. Word cloud of two K-Means clusters

An analysis of the two clusters reveals distinct thematic differences. Cluster 0 is characterized by frequent occurrences of words such as 'bitch', 'nigga', 'fuck', and 'shit', which are commonly associated with more aggressive or explicit lyricsal content. Cluster 1 is dominated by words like 'love', 'feel', 'never', and 'say', which are generally related to themes of affection, emotion, and relationships. This suggests that the model effectively differentiates lyrics based on their thematic focus.

Overall, despite the moderate silhouette score, the model demonstrates a reasonable ability to separate lyrics into two meaningful categories

B. Agglomerative Clustering

Based on the experiment we conduct, both the average and single linkage methods yielded relatively high Silhouette

Scores compared to other models. However, further qualitative analysis and visualization revealed significant shortcomings in the clustering results.

Specifically, in the case of single linkage, we observed a chaining effect where one large cluster included nearly all the points, and the remaining points were assigned to small or singleton clusters. This led to visually incoherent groupings where clusters did not reflect meaningful thematic separation in the song lyrics. Similarly, average linkage often produced unbalanced clusters, with one cluster containing only a single point and the rest grouped into another, making the silhouette score misleadingly high.

On the other hand, ward linkage, while not producing the highest silhouette score, demonstrated the most interpretable and thematically coherent clusters upon visualization and manual inspection. The resulting clusters showed more balanced distributions and clearer separation in reduced PCA space. Therefore, we selected Ward linkage for our final Agglomerative Clustering configuration, as it provided a more reliable structure in capturing meaningful patterns in the data despite lower silhouette score of 0.1348.

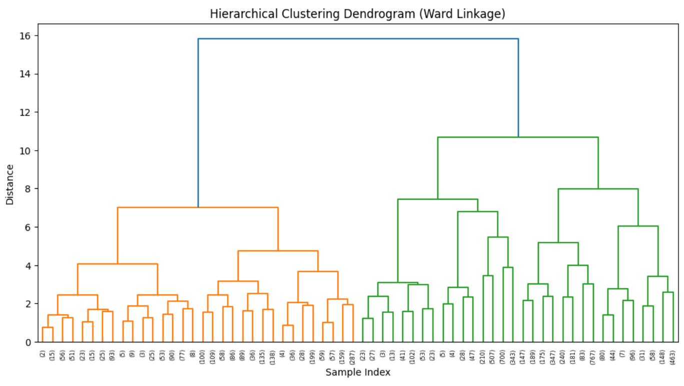
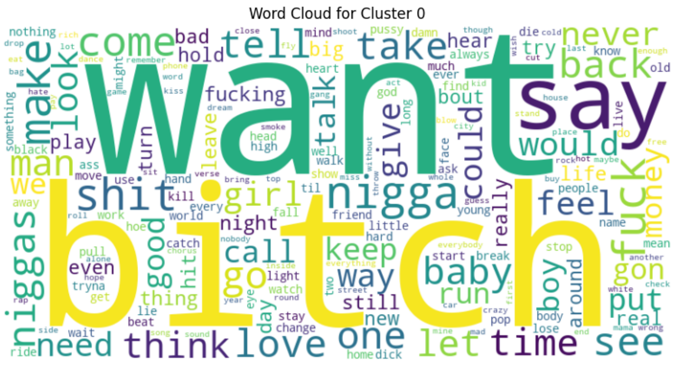


Fig. 2. Dendrogram for ward linkage



The word clouds for each cluster reveal a clear thematic distinction. Cluster 0 prominently features aggressive and explicit terms such as ”bitch”, ”nigga”, and ”shit”, alongside action-oriented words like ”want” and ”take”. This suggests that the songs grouped in this cluster revolve around assertive, confrontational, or street-oriented themes. In contrast, Cluster 1

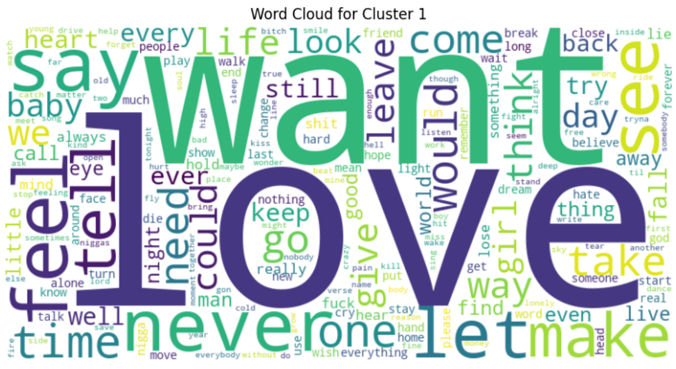


Fig. 3. Word cloud of lyrics words in Agglomerative with ward linkage clusters

emphasizes emotionally expressive and relational vocabulary, with dominant words including ”love”, ”say”, ”never”, and ”feel”. These indicate that the cluster largely contains songs focused on personal emotions, love, and introspection. This contrast demonstrates that the clustering process effectively grouped lyrics based on semantic tone and thematic content.

C. DBSCAN Clustering

Based on comprehensive parameter exploration and sensitivity analysis, we selected the optimal DBSCAN configuration for the song lyrics clustering task. The final parameters were determined to be $\varepsilon = 0.35$ and *minimum samples* = 4, which achieved a silhouette score of 0.674.

Unfortunately, the clustering results obtained from DBSCAN indicate suboptimal performance for the given song lyrics dataset. The distribution of data points across clusters is highly imbalanced, with one dominant cluster containing 7,158 songs, a second very small cluster containing only 4 songs, and 153 songs identified as noise points (labeled as -1). This suggests that DBSCAN failed to identify meaningful structure in the high-dimensional embedding space of the lyrics.

Word frequency analysis reveals that the smaller cluster is dominated by less informative tokens, such as ’oohooh’ and ’oohoohooh’, among its most frequent words. This suggests a need for further data preprocessing to remove such non-lexical artifacts.

D. Discussion

The result shows that the K-Means model achieved a Silhouette Score of 0.1684, while the experiment conducted by Gupta et al., obtained a silhouette score of 0.243[4]. This reinforces the effectiveness of K-Means in unsupervised song lyric analysis, especially when combined with appropriate text preprocessing and feature extraction.

V. Conclusion

This study explored the application of unsupervised machine learning techniques to cluster English-language song lyrics based on their linguistic similarities. We preprocessed a dataset of lyrics from Genius.com using various text mining techniques and converted the lyrics into vector representations

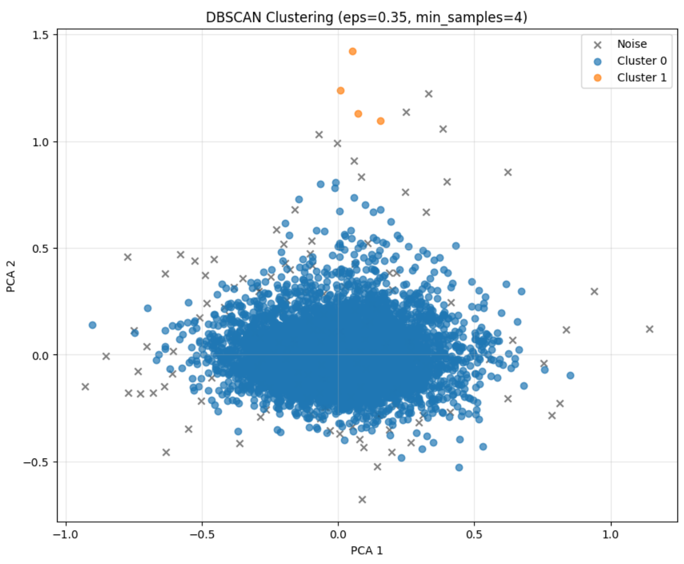


Fig. 4. DBSCAN Clusters Visualization

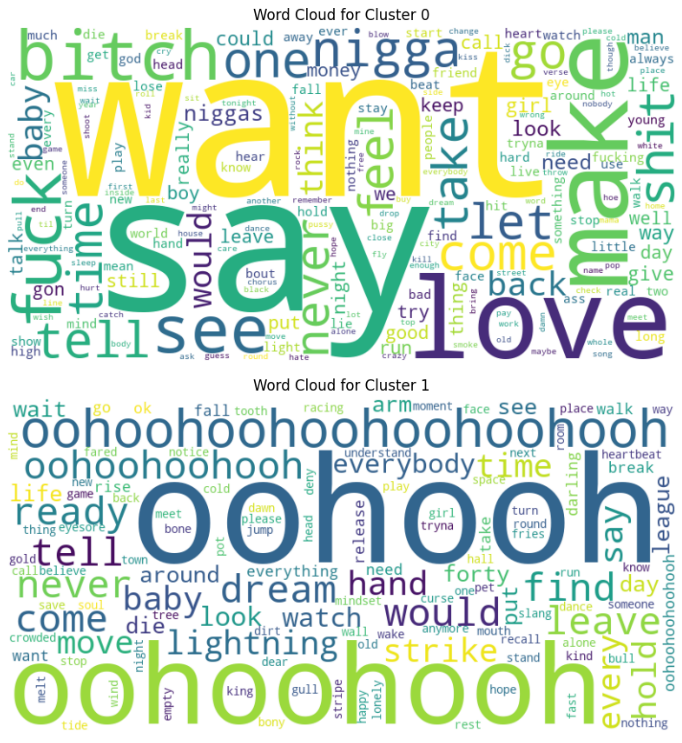


Fig. 5. Word cloud of two DBSCAN clusters

using Word2Vec. We then applied three clustering algorithms: K-Means, Agglomerative, and DBSCAN.

Our evaluation, using the Silhouette Score, suggested that two is the optimal number of clusters for the dataset. Visualization using PCA and word clouds revealed that K-Means produced the best separation between lyrics into two thematically distinct groups, which we interpreted as Rap/Hip-Hop and Romantic Pop-themed songs, followed by Agglomerative clustering with Ward linkage. The Rap/Hip-Hop cluster was characterized by more explicit and aggressive language, while the Romantic Pop cluster featured words related to love, emotions, and relationships.

In contrast, DBSCAN, despite achieving a higher Silhouette Score with optimized parameters, produced highly imbalanced clusters and identified a significant portion of the data as noise, indicating its unsuitability for this particular task and dataset.

Overall, this comparative analysis demonstrates the potential of K-Means and Agglomerative Clustering for uncovering thematic structures in song lyrics. The study provides practical insights into the process of clustering textual musical data and highlights the importance of evaluating clustering results not only quantitatively but also through qualitative interpretation.

VI. Future Works

Building upon the findings of this study, several avenues for future research can be explored

- **Incorporating Metadata:** Integrate available metadata such as genre, artist, and release year into the clustering process or use it as external validation. This could provide richer insights and potentially lead to more meaningful clusters.
- **Incorporating Contextual Embeddings:** Incorporating contextual embeddings such as BERT or SBERT may capture deeper semantic relationships compared to Word2Vec
- **Addressing Class Imbalance in DBSCAN:** Further investigate parameter tuning or modifications to the DBSCAN algorithm to handle the high dimensionality and potential density variations in the lyrics embeddings, which might improve its performance.

References

[1] Killin, A. (2018). The origins of music: Evidence, theory, and prospects. https://doi.org/10.1177/2059204317751971

[2] Ahmed, M. H., Tiun, S., Omar, N., and Sani, N. S. (2023). Short Text Clustering Algorithms, Application and Challenges: A Survey. Applied Sciences, 13(1), 342. https://doi.org/10.3390/app13010342

[3] D. B. Rarasati (2020), A Grouping of Song-lyrics Themes Using KMeans Clustering, JISA (Jurnal Informatika dan Sains), vol. 3, no. 2

[4] Gupta, V., S, J., and Kumar, S. (2021), Songs recommendation using Context-Based semantic similarity between lyrics. India Council International Subsections Conference (INDISCON), 43, 1-6. https://doi.org/10.1109/indiscon53343.2021.9582158

[5] Bataineh, Bilal. (2022). Fast Component Density Clustering in Spatial Databases: A Novel Algorithm. Information. 13. 477. 10.3390/info13100477.

[6] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, ”A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise,” Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining (KDD-96), Portland, OR, USA, 1996, 4-5

[7] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Efficient Estimation of Word Representations in Vector Space,” arXiv.org, Sep. 06, 2013. http://arxiv.org/abs/1301.3781

[8] Indeed Editorial Team (2025, March 26). What Are Filler Words? (Examples and Tips To Avoid Them). Indeed. https://www.indeed.com/career-advice/career-development/filler-word

[9] Bird, Steven, Edward Loper and Ewan Klein (2009), Natural Language Processing with Python. O’Reilly Media Inc.