Available online at :
http://ejournal.amikompurwokerto.ac.id/index.php/telematika/

Telematika

Accreditated SINTA "2" Kemenristek/BRIN, No. 85/M/KPT/2020

Optimizing Clustering of Indonesian Text Data Using Particle Swarm Optimization Algorithm: A Case Study of the Quran Translation

M. Didik R. Wahyudi[1], Agung Fatwanto[2]

[1,2]Department of Informatic, Faculty of Science and Technology
[1,2]Universitas Islam Negeri Sunan Kalijaga Yogyakarta, Yogyakarta, Indonesia
E-mail: m.didik@uin-suka.ac.id[1], agung.fatwanto@uin-suka.ac.id[2]

ARTICLE INFO

History of the article:
Received September 2, 2023
Revised February 13, 2024
Accepted February 20, 2024

Keywords:
Text_Mining
K-Means
Particle Swarm Optimization
Quran Mining
Quran Translation

Correspondece:
E-mail : m.didik@uin-suka.ac.id

ABSTRACT

The Quran considered the holy book for Muslims, contains scientific and historical facts affirming Islam's truth, beauty, and influence on human life. Consequently, the Quran text and its translations are valuable sources for text mining research, particularly for studying the interrelationship of its verses. One approach to grouping objects using certain algorithms is clustering, with K-Means Clustering being a prominent example. However, clustering results are often suboptimal due to the random selection of centroids. To address this, the study proposes using the Particle Swarm Optimization (PSO) algorithm, which selects centroids based on PSO results. The hybrid PSO algorithm initiates a single iteration of the K-means algorithm. It concludes either upon reaching the maximum iteration limit or when the average shift in the center of the mass vector falls below 0.0001. Evaluation of the clustering results from the three models indicates that the K-Means algorithm produced the lowest Sum of Squared Error (SSE) value of 1032.19. Additionally, the hybrid PSO algorithm generated the highest Silhouette value of 0.258 and the lowest quantization value of 0.00947. Further evaluation using a confusion matrix showed that K-Means clustering had an accuracy rate of 81.7%, K-Means with PSO had 82.5%, and the combination of K-Means with hybrid PSO yielded the highest accuracy rate of 91.1% among the three grouping models.

INTRODUCTION

The Quran, the holy book of Muslims, is a miracle and proof of the truth of Islam, including the beauty of its language, scientific content, historical stories, and influence on human life. The Quran has a unique structure, divided into 30 parts called "juz". This categorization makes it easier to memorize and understand the Quran, with each juz being further divided into several parts called "rukuk", based on the number of verses and the duration of each verse. The Quran can also be understood through its translation, so with a language that is understood, it is hoped that one can obtain the universal message of the Quran (Boulaouali, 2021). This makes the text of the Quran and its translations a valuable source of data in text mining research (Azmi, A.M., Al-Qabbany, 2019). One effort to understand the Quran can be made by using text mining by looking for similarities between translations of the Quran verses using the cosine similarity algorithm (R.Wahyudi, 2019), which is then continued with research to find the best way by utilizing various TF-IDF algorithm models (R Wahyudi, 2021). The translated Quranic verses were categorized using the Support Vector Machine (SVM), Naive Bayes, K-Nearest Neighbor (KNN), and J48

Decision Tree algorithms. This classification process was applied to both English and Indonesian translations of Surah Al-Baqarah. The findings revealed that the SVM algorithm achieved the highest accuracy, reaching 81.44%. (Hidayat, R., & Minati, 2019).

Another frequently employed approach for organizing data is clustering. Clustering is a method for categorizing data objects, but it has the drawback of arbitrarily determining centroids. Incorporating K-Means clustering with the Particle Swarm Optimization (PSO) algorithm addresses this issue by enabling the determination of centroids through a more optimal process, leading to improved data grouping (Van Der Merwe and Engelbrecht, 2003). A novel approach for selecting features is suggested to enhance the efficiency of text grouping and minimize computational time. This method employs the Particle Swarm Optimization (PSO) algorithm to optimize feature selection, aiming to maximize the creation of a new subset comprising informative text features (Abualigah, Khader, and Hanandeh, 2018). Executing the Fuzzy C-Means Clustering (FCM) algorithm with the integration of PSO introduces an enhanced PSO-designed FCM approach. This guarantees superior cluster performance, mainly when dealing with larger datasets (Niu and Huang, 2011). Attempts to enhance clustering quality include the utilization of the Particle Swarm Optimization (PSO) algorithm within the framework of K-Means (Gao, H., Li, Y., Kabalyants, P.S., Xu, H., & Martínez-Béjar, 2020). The implementation of Clustering and PSO combination coding was carried out by Augusto Luis Ballardini, referring to Merwe et al. (Ballardini, 2018). The success of PSO in producing more optimal clustering is confirmed by research by Abhinsha (Mano, 2020) and Rudi (Hariyanto and Zoqi Sarwani, 2021). The combination of K-Means Clustering with PSO in other research fields has also been successful in image-based identification of medicinal plants. In this research, the PSO method functions as an augmentation technique, improving the instrumentation's performance in identifying medicinal plants (Bisilisin, Herdiyeni, and Silalahi, 2017). The study titled "Enhancing K-Means Clustering for Identifying Endemic Areas of Infectious Diseases Using the Particle Swarm Optimization Algorithm in Semarang City" further affirms that the amalgamation of K-Means clustering with PSO yields enhanced performance in identifying infectious diseases (Rustam, Santoso and Supriyanto, 2018).

RESEARCH METHODS

The dataset used in this research is a translation of the Quran consisting of 114 letters with 6,236 verses published by the Department of Religion and documented in the digital Quran application version 2.1. The dataset has a CVS format, which consists of two attributes. The first attribute (*ID*) contains information about the letter and verse numbers, and the second attribute (*Terjemah*) includes the translation of the Quran in Indonesian. The stages of research conducted in this study are illustrated in Figure 1.

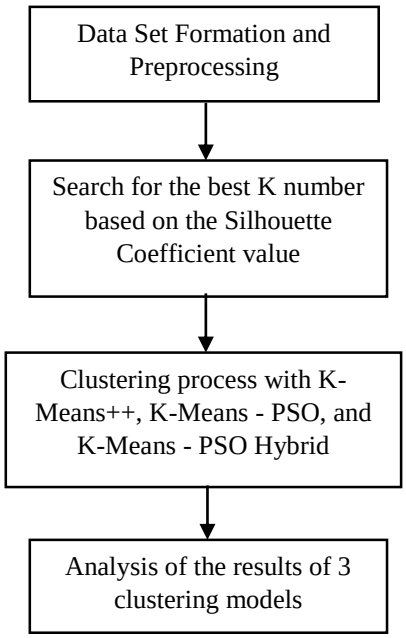

Figure 1. Research Steps

The research steps will be carried out: First, the preprocessing process for the Quran translation in the `Terjemah` column using the Python programming language will be stored in the `TjClean` column. This `TjClean` column will later be used in this research. The Quran translation dataset used in this research can be downloaded at: https://bit.ly/49Ah8DF. Figure 2 below shows the appearance of the dataset used in this research with the `TjClean` attribute.

Id	Terjemah	TjClean
[1.1]	Dengan menyebut nama Allah Yang Maha Pemurah lagi Maha Penyayang	menyebut nama allah maha pemurah maha penyayang
[1.2]	Segala puji bagi Allah Tuhan semesta alam	puji allah tuhan semesta alam
[1.3]	Maha Pemurah lagi Maha Penyayang	maha pemurah maha penyayang
[1.4]	Yang menguasai hari pembalasan	menguasai pembalasan
[1.5]	Hanya kepada Engkaulah kami menyembah dan hanya kepada Engkaulah kami mohon pertolongan	engkaulah menyembah engkaulah mohon pertolongan
[1.6]	Tunjukilah kami jalan yang lurus	tunjukilah jalan lurus
[1.7]	(yaitu) jalan orang-orang yang telah Engkau anugerahkan nikmat kepada mereka bukan (jalan) mereka yang dimurkai dan bukan (pula jalan) mereka yang sesat	jalan orangorang engkau anugerahkan nikmat jalan dimurkai jalan sesat
[2.1]	Alif Laam Miim	alif laam miim
[2.2]	Kitab (Al Qur'an) ini tidak ada keraguan padanya petunjuk bagi mereka yang bertakwa	kitab al quran keraguan petunjuk bertakwa
[2.3]	(yaitu) mereka yang beriman kepada yang gaib yang mendirikan shalat dan menafkahkan sebahagian rezki yang Kami anugerahkan kepada mereka	beriman gaib mendirikan shalat menafkahkan sebahagian rezki anugerahkan

Figure 2: Example of the dataset

The second step is identifying the optimal K value by assessing the Silhouette Coefficient using Python programming. Subsequently, execute the clustering procedure with K-Means++, K-Means-PSO, and the hybrid K-Means-PSO using the Python programming language. Thirdly, analyze the outcomes of three clustering models: K-Means clustering, K-Means with PSO, and the hybrid of K-Means and PSO. The evaluation involved calculating the sum of squared error (SSE), silhouette coefficient, and quantization score to compare the clustering models comprehensively.

RESULTS AND DISCUSSION

One way to understand the Quran is to group each verse based on text similarities or certain meanings. To carry out this grouping, previous research that has been carried out includes grouping based on the similarity between translations of the Quran verses with Cosine Similarity (R.Wahyudi, 2019) as well as conducting clustering with K-Means combined with various TF-IDF models (Wahyudi, 2020). This research showed distance-based clustering of the Quran translations using K-Mean clustering. The K-Mean clustering algorithm categorizes data by considering the proximity to a central point or centroid, resulting in different data clusters (Kaur, 2017). The centroid is determined randomly. This model of randomly determining centroids is a weakness of k-means clustering. To overcome this weakness, this research will use particle swarm optimization with the fitness function silhouette coefficient to determine the initial center of mass. PSO is an algorithm for population search that mimics the social behavior observed in bees or schools of fish. Initially designed to replicate flocking birds' graceful and unpredictable movements, PSO is a direct optimization technique capable of adjusting various parameters. (Basari *et al.*, 2013).

Preprocessing in this study includes lemmatization and stop word removal. The purpose of lemmatization is to return the notation to the source word. In this study, the prohibited words were not removed to change the true meaning of the verses translated in the Quran in the opposite sense. Next, the weight of each document's word will be determined using the TF-IDF algorithm. TF-IDF is a statistical measurement to measure a term's importance in a document. In the calculations carried out, the weight will be calculated based on the frequency of occurrence without considering the position of the word in the text, its semantics, and its event with other terms in the document.

Finding the best K number in the group

K-means divides the dataset into clusters, where each cluster shares common characteristics but varies in components compared to other clusters. The outcomes of the K-Means method are contingent on the initial value of the group center. Diverse initial values can result in different clusters. The clustering process with the K-Means algorithm was repeatedly simulated ten times to determine the optimal number of K. Model testing involved calculating silhouette coefficients for each clustering process. The assessment of silhouette coefficients gauges the proximity of objects within a cluster and the separation between different clusters. It combines two techniques: the cohesion method, which measures the proximity between objects in a cluster, and the separation method, which gauges the distance between distinct clusters. The calculation of the silhouette coefficient is performed using the Python programming language, as demonstrated in Figure 3.

```
range_n_clusters = [3, 4, 5, 6, 7, 8, 9]
ulang = [1,2,3,4,5,6,7,8,9,10]
ssc=[]
for u in ulang :
    for n_clusters in range_n_clusters:
        clusterer = KMeans(init='k-means++', n_clusters=n_clusters)
        cluster_labels = clusterer.fit_predict(x)
        silhouette_avg = silhouette_score(x, cluster_labels)
        ssc.append(silhouette_avg)
    print(u,";",f"{ssc[0]:.3f}",";",f"{ssc[1]:.3f}",";",f"{ssc[2]:.3f}",";",
          f"{ssc[3]:.3f}",";",f"{ssc[4]:.3f}",";",f"{ssc[5]:.3f}",";",f"{ssc[6]:.3f}")
    ssc.clear()
```

Figure 3. Program code to find the best number of K in clustering

From the program code in Figure 3, the best number of clusters between 3 and 9 clusters was searched for by carrying out 10 simulations of the silhouette coefficient calculation. From 10 simulations of the silhouette coefficient calculation, the data presented in Table 1 were obtained. From Table 1, it can be concluded that 10 simulations of the search for the best number of clusters in the translation of the Quran resulted in 90% of the number of clusters. The best is 4. However, only one simulation indicates that the optimal cluster count is 9. As a result, this study will utilize 4 clusters.

Table 1: Silhouette coefficients resulting from the search for the best K

Simulation to	*Number of Clusters*						
	3	*4*	*5*	*6*	*7*	*8*	*9*
1	0.255	0.269	0.245	0.234	0.196	0.261	0.251
2	0.255	0.269	0.265	0.235	0.242	0.212	0.251
3	0.255	0.269	0.265	0.234	0.242	0.261	0.231
4	0.255	0.263	0.264	0.235	0.242	0.261	0.268
5	0.255	0.270	0.265	0.234	0.242	0.212	0.262
6	0.255	0.269	0.265	0.235	0.242	0.212	0.223
7	0.255	0.269	0.265	0.235	0.243	0.261	0.268
8	0.254	0.269	0.265	0.235	0.243	0.212	0.262
9	0.255	0.269	0.264	0.234	0.253	0.219	0.268
10	0.245	0.269	0.264	0.235	0.242	0.219	0.252

Clustering with K-Means, K-Means - PSO, and K-Means - PSO Hybrid

In this study, the clustering procedure will employ three distinct models, specifically K-Means++, K-Means optimized through the integration of the Particle Swarm Optimization (PSO) algorithm, and K-Means optimized using a hybrid Particle Swarm Optimization (PSO) algorithm.

Clustering with K-Means

Following the number of clusters produced in the previous process, namely four clusters, the clustering process will be carried out with K-Means++ with a total of 4 clusters, as shown in the program code in Figure 3. The number of clusters is defined through the variable jk with a value of 4. Next, A dictionary named kmeanspp will be used to store evaluation metrics for each iteration. Stored evaluation metrics include 'silhouette_score', 'sse' (Sum of Squared Errors), and 'quantization_error' (quantization error). The iteration process is performed 20 times using the for _ in range(20) loop. In each iteration, the K-Means model will be initialized with the number of parameter clusters jk and using k-means++ initialization (init_pp=True). This model is then trained using the `TjClean (x)` dataset. After training the model, the next step is to calculate the evaluation metrics silhouette_score), SSE, and quantization_error. The results of the evaluation metrics for each iteration are then stored in the kmeanspp dictionary.

Clustering with PSO and K-Means

In this segment, we will implement the Particle Swarm Optimization (PSO) algorithm in conjunction with K-Means Clustering, utilizing a specified number of clusters (n_cluster) set to 4 based on the preceding process. The program code is illustrated in Figure 4. The PSO algorithm is iterated 20 times to attain optimal results. In each iteration, a ParticleSwarmOptimizedClustering object (pso_rep) is initialized with parameters such as the number of clusters (n_cluster), the number of particles (n_particles), the dataset (x),

hybrid mode (hybrid), the maximum number of iterations (max_iter), and debug settings (print_debug). The execution of the PSO algorithm is initiated by invoking the run() method on the pso_rep object. After the PSO algorithm has finished running, the results are used to initialize a K-Means object (pso_kmeans) with the same number of clusters. The K-Means centroids are set to match the best centroids (gbest_centroids) found by PSO. Cluster prediction uses the predict() method from K-Means, and several evaluation metrics such as silhouette score, the sum of squared errors (SSE), and quantization values are calculated. The results of the evaluation metrics are then entered into the pso_plain dictionary at each iteration, using the keys ('silhouette', 'sse', 'quantization').

```
jk = 4
kmeanspp = {
    'silhouette': [],
    'sse' : [],
    'quantization' : [],
}
for _ in range(20):
    kmean_rep = KMeans(n_cluster=jk, init_pp=True)
    kmean_rep.fit(x)
    predicted_kmean_rep = kmean_rep.predict(x)
    silhouette = silhouette_score(x, predicted_kmean_rep)
    sse = kmean_rep.SSE
    quantization = quantization_error(centroids=kmean_rep.centroid, data=x, labels=predicted_kmean_rep)
    kmeanspp['silhouette'].append(silhouette)
    kmeanspp['sse'].append(sse)
    kmeanspp['quantization'].append(quantization)
```

Figure 4. Clustering program code with K-Means

Clustering with Hybrid PSO and K-Means

In this section, we will investigate a hybrid approach for the assembly process, integrating both the K-means and PSO algorithms. Although the K-means algorithm is faster and requires fewer feature evaluations than PSO, it might produce less precise clustering outcomes. To overcome this limitation, the hybrid PSO model enhances the performance of the PSO clustering algorithm by initializing clusters with results obtained from the K-means algorithm.

The hybrid PSO algorithm initially runs the K-means algorithm once, terminating when either (1) the maximum number of iterations is reached or (2) the mean change in the center vector drops below 0.0001, a parameter defined by the user. The output of the K-means algorithm is utilized as one of the particles, while the remaining particles are randomly initialized.

```
jk = 4
pso_plain = {
    'silhouette': [],
    'sse' : [],
    'quantization' : [],
}
for _ in range(20):
    pso_rep = ParticleSwarmOptimizedClustering(
        n_cluster=jk, n_particles=10, data=x, hybrid=False, max_iter=2000, print_debug=2000)
    pso_rep.run()
    pso_kmeans = KMeans(n_cluster=jk, init_pp=False)
    pso_kmeans.centroid = pso_rep.gbest_centroids.copy()
    predicted_pso_rep = pso_kmeans.predict(x)

    silhouette = silhouette_score(x, predicted_pso_rep)
    sse = calc_sse(centroids=pso_rep.gbest_centroids, data=x, labels=predicted_pso_rep)
    quantization = pso_rep.gbest_score
    pso_plain['silhouette'].append(silhouette)
    pso_plain['sse'].append(sse)
    pso_plain['quantization'].append(quantization)
```

Figure 5. Clustering program code with PSO and K-Means

After executing the program code in Figure 5, which utilizes K-Means Clustering and PSO, the resulting clustering was reprocessed using the program code presented in Figure 6. This experiment was conducted with 20 iterations of the PSO Clustering algorithm, and the results obtained from each iteration were used as the initial centroid of the K-Means algorithm. The K-Means algorithm was then used to predict the input dataset `TjClean(x)`, and the prediction results were evaluated using the `Silhouette Score, Sum of Squared Errors (SSE)`, and PSO `quantization` values. The evaluation results of each iteration were saved in the pso_hybrid dictionary for further analysis.

```
jk = 4
pso_hybrid = {
    'silhouette': [],
    'sse' : [],
    'quantization' : [],
}
for _ in range(20):
    pso_rep = ParticleSwarmOptimizedClustering(
        n_cluster=jk, n_particles=10, data=x, hybrid=True, max_iter=2000, print_debug=2000)
    pso_rep.run()
    pso_kmeans = KMeans(n_cluster=jk, init_pp=False)
    pso_kmeans.centroid = pso_rep.gbest_centroids.copy()
    predicted_pso_rep = pso_kmeans.predict(x)

    silhouette = silhouette_score(x, predicted_pso_rep)
    sse = calc_sse(centroids=pso_rep.gbest_centroids, data=x, labels=predicted_pso_rep)
    quantization = pso_rep.gbest_score
    pso_hybrid['silhouette'].append(silhouette)
    pso_hybrid['sse'].append(sse)
    pso_hybrid['quantization'].append(quantization)
```

Figure 6. PSO hybrid and K-Means clustering program code

Figure 7 displays the outcomes of clustering the translated verses of the Quran using K-Means, PSO, and K-Means algorithms individually, along with the results obtained through the hybrid PSO and K-Means approach.

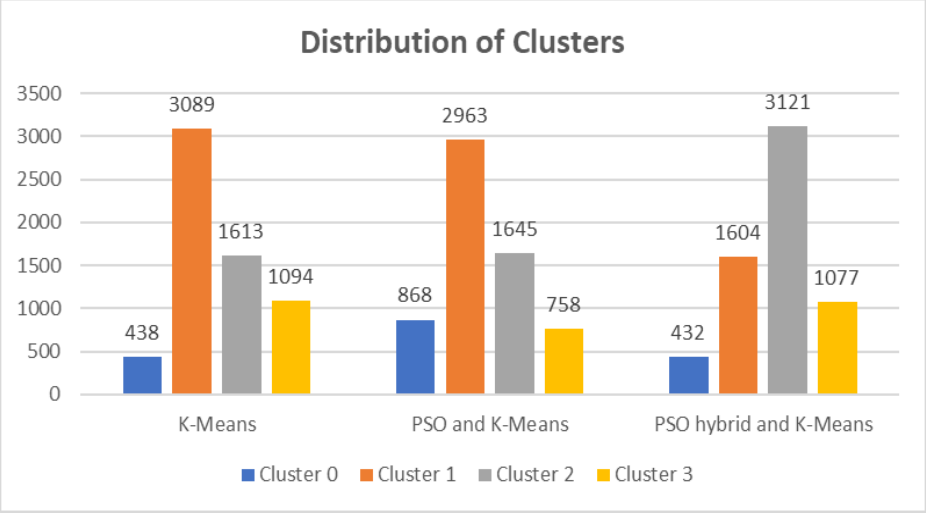

Figure 7. Distribution of data from clustering

Figure 7 displays the results of organizing verses into clusters using three separate clustering models. Among these three algorithms, it is clear that the data pattern in the K-Means algorithm is consistent with the combined PSO and K-Means clustering. In contrast, the hybrid PSO algorithm and K-Means clustering produce a model with a unique data pattern distinct from K-Means and PSO-K-Means combinations. The clustering results will be employed to compute the Sum of Squared Error (SSE), silhouette coefficient, and

quantization score to facilitate a comparison of the clustering effects. SSE represents the summation of squared distances between data points and cluster centers. A comparison of calculation results is shown in Figure 8.

	method	sse_mean	sse_stdev	silhouette_mean	silhouette_stdev	quantization_mean	quantization_stdev
0	K-Means++	1032.197322	24.648917	0.244784	0.033314	0.372876	0.000948
1	PSO	1362.894855	167.060285	0.161804	0.047703	0.403137	0.022370
2	PSO Hybrid	1072.786079	149.905357	0.258894	0.027964	0.368320	0.009474

Figure 8: Comparison of results of three clustering models

Figure 8 shows that the K-Means algorithm generated the lowest SSE value, and the silhouette score and quantitative error score were generated by the PSO hybrid algorithm and K-Means clustering.

Discussion

The clustering results obtained from the three models were tested using a confusion matrix to test the consistency of the models included in each cluster. The program code for calculating the confusion matrix with the Python program code is shown in Figure 9.

```
def plot_confusion_matrix(
    y_true, y_pred, classes, normalize=False, title=None,
    if not title:
        if normalize:
            title = 'Normalized confusion matrix'
        else:
            title = 'Confusion matrix, without normalization'
    # Compute confusion matrix
    cm = confusion_matrix(y_true, y_pred)
    classes = classes[unique_labels(y_true, y_pred)]
    if normalize:
        cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
        print("Normalized confusion matrix")
    else:
        print('Confusion matrix, without normalization')
    print(cm)
    fig, ax = plt.subplots()
    im = ax.imshow(cm, interpolation='nearest', cmap=cmap)
    ax.figure.colorbar(im, ax=ax)
    ax.set(xticks=np.arange(cm.shape[1]), yticks=np.arange(cm.shape[0]),
           xticklabels=classes, yticklabels=classes,
           title=title, ylabel='True label', xlabel='Predicted label')
    plt.setp(ax.get_xticklabels(), rotation=45, ha="right",
             rotation_mode="anchor")
    fmt = '.2f' if normalize else 'd'
    thresh = cm.max() / 2.
    for i in range(cm.shape[0]):
        for j in range(cm.shape[1]):
            ax.text(j, i, format(cm[i, j], fmt), ha="center", va="center",
                    color="white" if cm[i, j] > thresh else "black")
    fig.tight_layout()
    return ax
```

Figure 9. Program code to calculate the confusion matrix

The program code shown in Figure 9 is written in Python and defines a function named '`plot_confusion_matrix`'. This function helps create and display a confusion matrix, which gives an idea of how effectively the model can classify samples into their relevant categories. To use this function, you need to input the actual label (`y_true`) and predicted label (`y_pred`), and a list of classes likely to be present in the classification results.

Moreover, there is an option to normalize the confusion matrix as well. By setting the '`normalize`' parameter to '`True`', the matrix will display the proportion of correct predictions for each class. This will

provide a more detailed analysis of the model's performance. Executing the Python program code presented in Figure 9 in calculating the confusion matrix produces consistency values for each cluster formed for the three models, as shown in Figure 10.

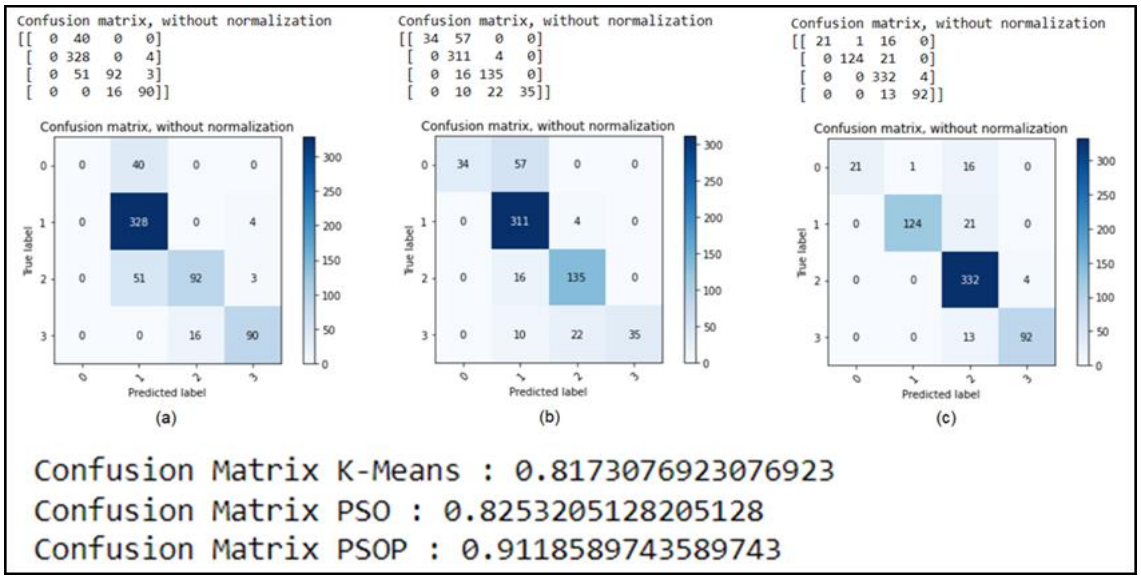

Figure 10. The results of the confusion matrix

The computation of the confusion matrix displayed in Figure 10 reveals that K-Means clustering yields the lowest accuracy result at 0.817 (81.7%). Conversely, the hybrid PSO and K-sMeans clustering achieved the highest accuracy outcomes, reaching 0.911 (91.1%).

Based on the results of the confusion matrix analysis, it can be concluded that grouping translations of the Quran verses with K-Means Clustering optimized with PSO gives better results compared to previous research, namely grouping with cosine similarity and Clustering with K-Means Clustering as proven by the percentage of confusion values, matrix in this research.

CONCLUSIONS AND RECOMMENDATIONS

This study explored the use of clustering algorithms, particularly K-Means clustering, for analyzing the interrelationship of Quranic verses. However, the random selection of centroids in traditional K-Means clustering often leads to suboptimal results. A hybrid approach combining K-Means with Particle Swarm Optimization (PSO) was proposed to address this limitation. The hybrid PSO algorithm demonstrated improved clustering performance compared to traditional K-Means clustering. It achieved a lower Sum of Squared Error (SSE), indicating better data point clustering around centroids. Furthermore, the hybrid PSO algorithm produced a higher Silhouette value and a lower quantization value, suggesting better clustering quality and reduced distortion. Evaluation using a confusion matrix revealed that the combination of K-Means with hybrid PSO yielded the highest accuracy rate among the three models tested, indicating the effectiveness of the proposed approach in grouping Quranic verses based on their interrelationship. Future research could explore additional optimization techniques and algorithms to enhance the clustering performance further and gain deeper insights into the Quran's textual structure and content.

ACKNOWLEDGEMENT

This study received financial support from Lembaga Penelitian dan Pengabdian kepada Masyarakat (LPPM) UIN Sunan Kalijaga, in accordance with Chancellor's Decree No. 127.3 of 2021 at UIN Sunan Kalijaga, pertained to the Penetapan Judul dan Personalia Penelitian Terapan Kajian Strategis UIN Sunan Kalijaga Yogyakarta in 2021.

REFERENCES

Abualigah, L. M., Khader, A. T. and Hanandeh, E. S. (2018) 'A new feature selection method to improve the document clustering using particle swarm optimization algorithm', *Journal of Computational Science*, 25(September), pp. 456–466. doi: 10.1016/j.jocs.2017.07.018.

Azmi, A.M., Al-Qabbany, A. O. & H. (2019) 'A Computational and natural language processing based studies of hadith literature: a survey'. doi: 10.1007/s10462-019-09692-w.

Ballardini, A. L. (2018) 'A tutorial on Particle Swarm Optimization Clustering'. doi: https://doi.org/10.48550/arXiv.1809.01942.

Basari, A. S. H. *et al.* (2013) 'Opinion mining of movie review using hybrid method of support vector machine and particle swarm optimization', *Procedia Engineering*, 53, pp. 453–462. doi: 10.1016/j.proeng.2013.02.059.

Bisilisin, F. Y., Herdiyeni, Y. and Silalahi, B. P. (2017) 'Optimasi K-Means Clustering Menggunakan Particle Swarm Optimization pada Sistem Identifikasi Tumbuhan Obat Berbasis Citra', *Jurnal Ilmu Komputer dan Agri-Informatika*, 3(1), p. 37. doi: 10.29244/jika.3.1.37-46.

Boulaouali, T. (2021) 'Quran Translation: A Historical-Theological Exploration', *International Journal of Islamic Thought*, 19. doi: 10.24035/ijit.19.2021.202.

Gao, H., Li, Y., Kabalyants, P.S., Xu, H., & Martínez-Béjar, R. (2020) 'A Novel Hybrid PSO-K-Means Clustering Algorithm Using Gaussian Estimation of Distribution Method and Lévy Flight', *IEEE Access*, pp. 122848–122863.

Hariyanto, R. and Zoqi Sarwani, M. (2021) 'Optimizing K-Means Algorithm by Using Particle Swarm Optimization in Clustering for Students Learning Process', *Jurnal Ilmiah Bidang Teknologi Informasi dan Komunikasi*, 6(1). doi: 10.25139/inform.v6i1.3459.

Hidayat, R., & Minati, S. (2019) 'Comparative Analysis of Text Mining Classification Algorithms for English and Indonesian Qur'an Translation', *IJID (International Journal on Informatics for Development)*, 8(1), pp. 47–51. doi: 10.14421/ijid.2019.08108.

Kaur, P. (2017) 'Outlier Detection Using Kmeans and Fuzzy Min Max Neural Network in Network Data', *Proceedings - 2016 8th International Conference on Computational Intelligence and Communication Networks, CICN 2016*, pp. 693–696. doi: 10.1109/CICN.2016.142.

Mano, A. (2020) 'A Novel Approach based on PSO Optimized K-Means in MRI Brain Image Segmentation'. doi: 10.36227/techrxiv.12310100.v2.

Van Der Merwe, D. W. and Engelbrecht, A. P. (2003) 'Data clustering using particle swarm optimization', *2003 Congress on Evolutionary Computation, CEC 2003 - Proceedings*, 1, pp. 215–220. doi: 10.1109/CEC.2003.1299577.

Niu, Q. and Huang, X. (2011) 'An improved fuzzy C-means clustering algorithm based on PSO', *Journal of Software*, 6(5), pp. 873–879. doi: 10.4304/jsw.6.5.873-879.

R.Wahyudi, M. D. (2019) 'Penerapan Algoritma Cosine Similarity pada Text Mining Terjemah Al- Qur' an Berdasarkan Keterkaitan Topik', *Semesta Teknika*, 22(1), pp. 41–50. doi: 10.18196/st.221235.

R Wahyudi, M. D. (2021) 'Evaluation of TF-IDF Algorithm Weighting Scheme in The Qur'an Translation Clustering with K-Means Algorithm', *Journal of Information Technology and Computer Science*, 6(2), pp. 117–129. doi: 10.25126/jitecs.202162295.

Rustam, S., Santoso, H. A. and Supriyanto, C. (2018) 'Optimasi K-Means Clustering Untuk Identifikasi Daerah Endemik Penyakit Menular Dengan Algoritma Particle Swarm Optimization Di Kota Semarang', *ILKOM Jurnal Ilmiah*, 10(3), pp. 251–259. doi: 10.33096/ilkom.v10i3.342.251-259.

Wahyudi, M. D. R. (2020) 'Evaluation of TF-IDF Algorithm Weighting Scheme in The Qur'an Translation Clustering with K-Means Algorithm'. doi:10.25126/jitecs.202162295.