Jurnal Teknologi Terpadu

https://journal.nurulfikri.ac.id/index.php/jtt
ISSN : 2477-0043 ISSN ONLINE : 2460-7908

OPTIMASI PARAMETER DBSCAN MENGGUNAKAN METODE *DIFFERENTIAL EVOLUTION* UNTUK DETEKSI ANOMALI PADA DATA TRANSAKSI BANK

Rifqi Karunia Ibadirachman[1], Yulison Herry Chrisnanto[2], Puspita Nurul Sabrina[3]

[1,2,3] Teknik Informatika, Universitas Jendral Achmad Yani
Cimahi, Jawa Barat, Indonesia 40525
rifqikarunia20@if.unjani.ac.id, yhc@if.unjani.ac.id, puspita.sabrina@lecture.unjani.ac.id

Abstract

Anomalies in bank transaction data often indicate fraudulent activity or errors. This research aims to detect anomalies in bank transaction data by optimizing DBSCAN parameters using the Differential Evolution (DE) method because there are shortcomings, namely the difficulty of determining the right parameters to create the right cluster in order to detect anomalies in bank transaction data properly. The data used is transaction data from Bank XYZ with more than 1011 data records. The research stages include data collection, data preprocessing (data cleaning, normalization, and transformation), system design, algorithm implementation, and analysis and testing using the Silhouette score and Z-score methods. The DE method is used to automatically determine the optimal parameters of MinPts and Epsilon. The results show that the use of DE can produce optimal parameters, with increased anomaly detection accuracy using DBSCAN. Evaluation with Silhouette score shows an average accuracy of 0.7916 and using DBI reaches 0.19791 at the lowest, while Z-score and MSE measurements show high cluster density with anomaly detection accuracy reaching 98.41% and 0.555537. The DE approach to parameter selection is effective in improving the performance of DBSCAN in detecting anomalies in bank transaction data. Suggestions for future research are to increase the number of data records and conduct experiments on a wider variety of data attributes.

Keywords: *Anomaly, Bank Transaction Data, DBSCAN, Differential Evolution, Optimize*

Abstrak

Anomali dalam data transaksi bank sering kali menandakan adanya aktivitas penipuan atau kesalahan. Penelitian ini bertujuan mendeteksi anomali pada data transaksi bank dengan optimasi parameter DBSCAN menggunakan metode *Differential Evolution* (DE) karena terdapat kekurangan yaitu sulitnya menentukan parameter yang tepat untuk membuat *cluster* yang tepat agar dapat mendeteksi anomali pada data transaksi bank dengan baik. Data yang digunakan adalah data transaksi dari Bank XYZ dengan lebih dari 1011 *record* data. Tahapan penelitian meliputi pengumpulan data, *preprocessing* data (pembersihan, normalisasi, dan transformasi data), perancangan sistem, implementasi algoritma, serta analisa dan pengujian menggunakan metode *Silhouette score* dan *Z-score* . Metode DE digunakan untuk menentukan parameter optimal MinPts dan Epsilon secara otomatis. Hasil penelitian menunjukkan bahwa penggunaan DE dapat menghasilkan parameter yang optimal, dengan peningkatan akurasi deteksi anomali menggunakan DBSCAN. Evaluasi dengan *Silhouette score* menunjukkan akurasi rata-rata 0.7916 dan menggunakan DBI mencapai 0.19791 paling rendah, sementara pengukuran *Z-score* dan MSE menunjukkan kepadatan *cluster* yang tinggi dengan akurasi deteksi anomali mencapai 98.41% dan 0.555537. Pendekatan DE untuk pemilihan parameter ini efektif dalam meningkatkan performa DBSCAN dalam mendeteksi anomali pada data transaksi bank. Saran untuk penelitian selanjutnya adalah meningkatkan jumlah *record* data dan lakukan percobaan pada variasi atribut data yang lebih beragam.

Kata kunci: Anomali, Data Transaksi Bank, DBSCAN, *Differential Evolution*, Optimasi

1. PENDAHULUAN

Anomali didefinisikan sebagai pola yang menyimpang dari perilaku yang diharapkan dari data. Anomali pada data bisa diketahui melalui beberapa teknik analisis dalam dunia teknologi. Memungkinkan pengidentifikasian kesalahan, anomali, ketidakefektifan, ketidaksesuaian, dan bias yang sering dikaitkan dengan penipuan, dengan penekanan pada jumlah dolar tertentu yang melebihi ambang batas kontrol internal. Salah satunya anomali pada transaksi bank merujuk pada aktivitas atau pola transaksi yang tidak biasa

Diterima 19 April 2024, Direvisi 24 Mei 2024, Diterima untuk publikasi 19 Juni 2024

atau mencurigakan dan berpotensi menunjukkan adanya penipuan, kesalahan, atau kegiatan ilegal. Anomali ini dapat berupa berbagai bentuk.

Deteksi anomali dapat diterapkan pada permasalahan seperti kecurangan pada transaksi bank, penipuan untuk kartu kredit, perawatan kesehatan, asuransi, dan deteksi kesalahan dalam sistem yang sangat penting sebelum terjadi kerusakan besar[1] .

Salah satu metode yang dapat digunakan dalam mendeteksi anomali data adalah metode *clustering*. Di mana metode *clustering* dapat mengidentifikasi dan Tujuannya adalah untuk membuat data dalam *cluster* yang sama lebih mirip satu sama lain dibandingkan dengan data di cluster lain. *Density-Based Spatial Clustering of Applications with Noise* (DBSCAN) adalah salah satu algoritma pengelompokan yang paling umum dan cukup banyak dikutip dalam literatur ilmiah. Algoritma DBSCAN umumnya menghubungkan kelompok sebagai area padat peristiwa dalam sampel data yang terisolasi oleh daerah dengan kepadatan rendah. DBSCAN secara alami mengidentifikasi dan mengabaikan titik-titik yang terletak di luar *cluster* yang padat, sehingga lebih *robust* terhadap *noise*. Ini membuatnya sangat efektif dalam mendeteksi anomali yang muncul sebagai *outliers* di luar area yang padat. DBSCAN juga sangat tepat digunakan pada pengelompokan data yang mempunyai atribut lokasi. Karena keunggulan algoritma DBSCAN, banyak peneliti mempertimbangkan penggunaannya untuk tujuan penelitian mereka. Sebuah algoritma unik berbasis DBSCAN untuk deteksi anomali disajikan dalam Penelitian terdahulu menggunakan metode DBSCAN untuk mendeteksi kecurangan dan anomali data pada Prediksi finansial[2] , *Finger Print*[3] , *Credit Card* [4], data kendaraan[5].

Namun, meskipun DBSCAN telah menunjukkan kinerja yang baik dalam banyak aplikasi, terdapat beberapa tantangan dan keterbatasan yang dihadapi oleh peneliti. Pada penelitian [2] DBSCAN digunakan untuk memprediksi *error* pada *financial stements*. Dengan hasil *Accuracy* 51.5%, *Precision* 16.4%, *Recall* 53.1%. Dengan melakukan beberapa kali *experiman* untuk menentukan parameter agar dapat membentuk *cluster* yang sesuai. Penelitian terdahulu [6], Kesulitan dalam menentukan parameter yang sesuai ini sering kali memerlukan beberapa iterasi dan penyesuaian berdasarkan uji coba dan *error* atau analisis yang lebih mendalam mengenai distribusi data yang dianalisis. Karena DBSCAN bekerja dengan baik pada data yang memiliki densitas yang seragam, namun menjadi kurang efektif ketika data memiliki variasi densitas yang tinggi. Hal ini membuat penentuan parameter yang tepat menjadi lebih kompleks.

Penelitian terdahulu [7] hasil eksperimen menunjukkan bahwa pendekatan *Modified* data dengan menambahkan data musiman per bulan dan pemilihan parameter yang baik pada DBSCAN mampu mendeteksi lebih banyak titik anomali (4.79%) dibandingkan dengan DBSCAN standar (2.16%), menunjukkan peningkatan dalam deteksi anomali lokal dalam data musiman. Penelitian ini [8] menujukan kemampuan DBSCAN dengan memilih parameter yang baik, dalam mendeteksi kesalahan data atau anomali pada data yang terbatas atau kurangnya data historis yang cukup untuk menganalisis dan mengidentifikasi pola anomali.

Sering kali peneliti terdahulu mengalami kesulitan saat menentukan parameter yang tepat untuk DBSCAN, sehingga banyak titik data yang seharusnya menjadi bagian dari *cluster* dianggap sebagai *noise* atau sebaliknya. Pemilihan parameter *Minimum Points* (MinPts) dan *Epsilon* (Eps) yang baik sangat penting untuk identifikasi anomali yang lebih akurat. Namun, metode konvensional dalam menentukan parameter dengan melakukan eksperimen pemilihan parameter ini masih kurang efektif dan efisien. Oleh karena itu, pemilihan parameter yang optimal menjadi salah satu tantangan utama dalam penerapan DBSCAN. Karena itu diperlukan metode untuk menentukan atau melakukan optimasi pemilihan parameter DBSCAN agar memudahkan penggunaan algoritma DBSCAN tersebut. Salah satu pendekatan yang digunakan untuk menangani sulitnya menentukan parameter DBSCAN adalah metode *Differential Evolution* [9].

Defferential Evolution (DE) adalah salah satu algoritma optimasi yang digunakan untuk menemukan solusi optimal pada masalah optimasi global. DE memungkinkan penentuan parameter yang optimal secara otomatis, meningkatkan kualitas hasil *clustering* yang dihasilkan oleh DBSCAN [10]. Penelitian lain [11] DE digunakan bersama dengan *algoritma clustering* DBSCAN untuk membentuk algoritma NCjDE-2LS yang dapat menemukan dan mengoptimalkan *multiple global optima.* Dengan menghasilkan rata-rata akurasi pada tingkat puncak 100%. Algoritma DE juga dapat digunakan untuk mengoptimalkan parameter *epsilon* dalam DBSCAN, penelitian ini [12] menggunakan DE untuk mengoptimasikan parameter Eps karena DE memiliki mekanisme yang baik untuk mengeksplorasi ruang solusi dan mengeksploitasi solusi terbaik. Ini juga penting untuk pengaturan parameter DBSCAN, yang harus menemukan keseimbangan antara kepadatan dan radius pencarian. Selain itu, DE mampu menangani optimasi pada ruang multidimensi, yang membuatnya cocok untuk menemukan kombinasi parameter terbaik.

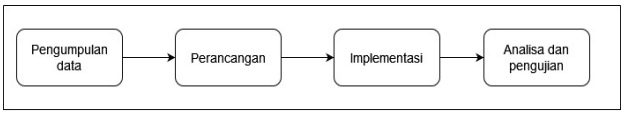
Pada penelitian terdahulu masih melakukan *experimen* menentukan parameter *MinPts* dan *Epsilon* untuk membentuk *cluster* pada DBSCAN [1], [2]. Yang menghasilkan banyak titik data yang seharusnya termasuk dalam kluster dianggap sebagai *noise* (anomali) atau sebaliknya. Dan juga DE digunakan untuk mengoptimasikan parameter Eps [6], [9], [12] . belum banyak peneliti yang menggunakan metode DE untuk membantu DBSCAN menemukan parameter terbaik untuk mendeteksi anomali pada data transaksi bank. DE juga dapat

digunakan untuk mengoptimalkan parameter MinPts dan Epsilon untuk membentuk *cluster* yang baik agar DBSCAN dapat mendeteksi adanya anomali pada data transaksi. Setelah iterasi pengetesan pemilihan parameter untuk proses DBSCAN. Hasil *cluster* akan divalidasi menggunakan metode *Silhouette score* dan *Davies-Bouldin Indeks* (DBI). untuk mengukur keakuratan anomali yang di identifikasi menggunakan *Z-score* sebagai pengukuran proporsional data dan menggunakan MSE pengukuran anomali hasil cluster.

Berdasarkan uraian tersebut pengenalan algoritma DBSCAN dan metode *Differential evolution*. Diharapkan dapat memberikan hasil yang baik untuk menentukan kedua parameter MinPts dan *Epsilon* agar dapat menghasilkan *cluster* yang baik untuk mendeteksi anomali pada data transaksi bank.

2. METODE PENELITIAN

Pada penelitian ini dilakukan beberapa tahapan penelitian yang bisa dilihat pada gambar 1.



Gambar 1. Tahapan Pengujian

Dapat dilihat dari gambar 1, ada beberapa tahapan yang harus dilewati dimulai dengan Pengumpulan data hingga pada proses Analisa dan pengujian. Berikut adalah uraian dari tahapan pada gambar 1.

2.1 Pengumpulan data

Pada tahap ini melakukan pencarian dan pengumpulan data. Yang di mana data yang diusulkan berupa data transaksi pada bank xyz. Set data yang sesuai untuk penggunaan DBSCAN dengan karakteristik berupa Numerik, Skala yang serupa dan juga relevan untuk *cluster*. Maka dari itu, diperlukan *preprocessing* untuk menyesuaikan data agar bisa diolah pada algoritma DBSCAN.

Jumlah *record* data transaksi lebih dari atau kurang dari 1011 digunakan. Objek deteksi anomali akan digunakan untuk mengidentifikasi transaksi pada data. Dengan DBSCAN, dapat menemukan pengelompokan data dan menemukan adanya data yang tidak biasa pada data transaksi.

Pada tahap *preprocessing* awal akan dilakukan tahap *data clean*, normalisasi data dan transformasi data. Di mana *preprocessing data clean* untuk menghapus *missing value* dan transformasi data untuk mengubah data yang bertipe objek menjadi numerik agar dapat diolah menggunakan algoritma DBSCAN. Berikut adalah data yang diusulkan dapat dilihat pada tabel 1.

Tabel 1. *Dataset* Transaksi Bank XYZ

Coloum	***Tipe***	***Missing***	***min***	***max***
TID	*Object*	0	*null*	*null*
CID	*Object*	0	*null*	*null*
CDOB	*Object*	0	*null*	*null*
CG	*Object*	0	*null*	*null*
CL	*Object*	0	*null*	*null*
CAB	*float64*	0	0	3,053,688
TD	*date*	0	*null*	*null*
TT	*int64*	0	*null*	*null*
TA (INR)	*float64*	0	188,846	230,002
Latitude	*numeric*	0	*null*	*null*
Longitude	*numeric*	0	*null*	*null*

Tahap pengumpulan data dilakukan dan diambil dari *dataset* yang telah digunakan oleh para peneliti sebelumnya. Data dan *Type* data yang dapat dilihat pada tabel 1. Menunjukkan beberapa kolom dengan akronim sebagai berikut pada tabel 2.

Tabel 2. *Dataset* nama akronim

Coloum	***Name***
TID	*TransactionID*
CID	*CustomerID*
CDOB	*CustomerDOB*
CG	*CustGender*
CL	*CustLocation*
CAB	*CustAccountBalance*
TD	*TransactionDate*
TT	*TransactionTime*
TA (INR)	*TransactionAmount (INR)*
Latitude	*Latitude*
Longitude	*Longitude*

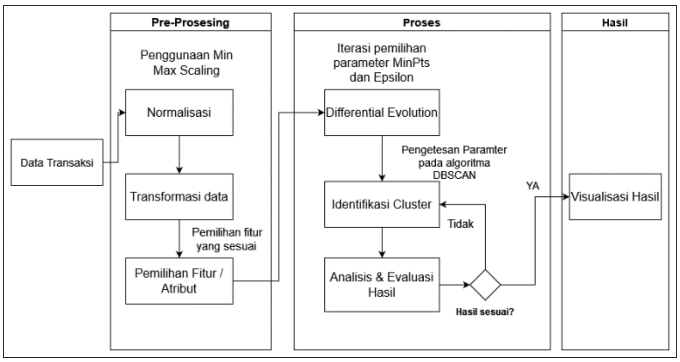
Dari data yang digunakan terdapat ±1011 *record* data yang terdapat pada *dataset* transaksi bank. Data yang didapatkan akan melalui *preprocessing* agar dapat disesuaikan untuk penggunaan algoritma DBSCAN.

2.2 Perancangan

Pada perancangan ini melakukan tahapan-tahapan mengenai hal yang akan dilakukan pada proses *clustering* data transaksi bank. Tahapan awal yang akan dilakukan adalah *preprocessing* data yaitu normalisasi untuk

menyesuaikan data agar dapat digunakan pada algoritma DBSCAN, Selanjutnya pemilihan fitur yang di mana penyesuaian fitur yang relevan untuk *cluster*, Selanjutnya adalah pemilihan parameter menggunakan metode *Differential Evolution* yang di mana dilakukan beberapa iterasi untuk menemukan parameter yang paling baik agar menghasilkan *cluster* yang baik.

Pada tahap proses dilakukan pengetesan parameter yang telah terdefinisi oleh metode *differential evolution*, untuk menghasilkan *cluster*. Tahap ini juga menguji dan menganalisa hasil apakah akurasi dari *cluster* yang dibentuk untuk mendeteksi anomali akurat. Pada tahap terakhir menampilkan visualisasi hasil dari *cluster anomaly* data. Dapat dilihat rincian dari rancangan pada gambar 2 berikut.



Gambar 2. Tahap Perancangan

2.2.1 *Differential Evolution*

Metode *Differential Evolution* (DE) digunakan dalam bentuk *Binary Differential Evolution* (BDE), di mana solusi-solusi direpresentasikan dalam bentuk biner[11]. DE membantu dalam menentukan parameter-parameter DBSCAN dengan cara mencari kombinasi parameter yang optimal untuk mencapai hasil *clustering* yang baik. DE membantu dalam menyesuaikan nilai Eps dan MinPts secara otomatis berdasarkan evaluasi kualitas *clustering* yang dilakukan selama proses optimasi. Adapun beberapa tahapan penggunaan DE pada proses tahapan optimasi di antaranya sebagai berikut ;

a) Inisialisasi populasi Langkah ini menciptakan populasi awal secara acak dalam ruang n dimensi dengan mempertimbangkan batasan atas dan bawah untuk setiap variabel. bisa dilihat pada persamaan 2.1

$$X_i G = L + r * (U - L) \quad (2.1)$$

di mana:

- L dan U adalah batas bawah dan atas dari ruang pencarian.
- r adalah vektor acak yang nilainya antara 0 dan 1.
- i adalah indeks individu.
- G adalah indeks generasi.

b) Mutasi DE secara acak memilih dua vektor populasi yang berbeda dan menggunakan perbedaan antara individu-individu tersebut untuk memutasi individu target. Bisa dilihat pada persamaan 2.2

$$V_i G = X_{r1} G + F * (X_{r2} G - X_{r3} G) \quad (2.2)$$

di mana:

- r1,r2,r3 adalah indeks acak yang berbeda dari i.
- F adalah faktor skala (biasanya antara 0 dan 1).

c) *Crossover* individu baru dihasilkan dengan menggabungkan individu target dan *mutant*, memastikan bahwa vektor uji tidak menduplikasi vektor individu target. Bisa dilihat pada persamaan 2.3

$$U_{j,i} G + 1 = \{ V_{j,i} G \; if \; Rand \; j \leq CR \; or \; J = J_{rand} \quad (2.3)$$

di mana:

- CR adalah rasio *crossover*.
- randjrandj adalah nilai acak antara 0 dan 1.
- J *rand* adalah indeks acak yang memastikan minimal satu elemen dari vi,G+1 dipilih.

d) Seleksi Algoritma menggunakan seleksi *greedy* untuk memilih individu yang lebih baik antara vektor uji dan vektor individu target untuk generasi berikutnya berdasarkan fungsi kebugaran (*fitness*). Bisa dilihat pada persamaan 2.4

$$X_{,i} G + 1 = \{ U_{j,i} G \; jika \; f\left(U_{j,i} G\right) \leq f \; X_{,i} G \quad (2.4)$$

di mana f(·) adalah fungsi *fitness* yang dievaluasi.

Dengan pendekatan ini, secara efektif menggunakan DE untuk mengeksplorasi ruang parameter dari DBSCAN dan secara otomatis menemukan kombinasi nilai *Epsilon* dan MinPts yang menghasilkan pengklusteran terbaik menurut metrik yang kita pilih. Ini membantu mengatasi tantangan pemilihan parameter manual yang sering kali membutuhkan eksperimen berulang dan pengetahuan mendalam tentang data.

2.2.2 DBSCAN

Algoritma DBSCAN (*Density-Based Spatial Clustering of Applications with Noise*), sebuah algoritma pengelompokan berbasis kepadatan yang populer. Algoritma ini dapat menemukan kelompok data dengan bentuk dan ukuran yang berbeda dari sejumlah besar data, yang mengandung *noise* dan *outliers*. DBSCAN mampu mendeteksi anomali atau *outlier* lebih baik berdasarkan adanya atribut lokasi pada data. Kemampuan DBSCAN memungkinkan untuk mengekstrak pola-pola yang tidak biasa atau anomali pada kerapatan data[13].

Beberapa istilah dalam metode DBSCAN antara lain:

a) ***Core*** Pusat titik dalam sebuah *cluster* ditentukan oleh densitas, di mana sejumlah titik harus berada dalam jangkauan Eps (*radius* atau nilai ambang batas). Pengguna menentukan nilai MinPts (jumlah *minimum* titik dalam sebuah *cluster*).

b) ***Border*** Titik yang menjadi batasan dalam kawasan titik pusat (*core*).

c) ***Noise*** Titik yang tidak dapat dijangkau oleh *core* dan bukan merupakan border seperti persamaan 2.5

$$NOISE : \{x \in X \mid \forall i : x \in C\, i\} \qquad (2.5)$$

Dimana X merupakan gugus data, dan Ci merupakan *cluster* ke-1.

d) Densitas terjangkau secara langsung: Sebuah titik dianggap sebagai titik terjangkau secara langsung jika titik tersebut terhubung secara langsung dengan titik pusat (*core*) seperti persamaan 2.6

$$x \in N\,Eps\,(y) \wedge |\,N\,Eps\,(y)\,| \geq MinPts \qquad (2.6)$$

NEps (y) = titik sekitar y dalam radius Eps
MinPts = minimal titik dalam *cluster*

e) Densitas terjangkau: Sebuah titik dianggap sebagai titik terjangkau jika titik tersebut terhubung secara tidak langsung dengan titik pusat (*core*).

f) Densitas terhubung: Sebuah titik dianggap terhubung dengan titik lainnya. Dalam DBSCAN, diperlukan dua parameter *input*, yaitu *Epsilon* (Eps) dan titik minimum (MinPts). Eps-titik sekitar didefinisikan seperti persamaan 2.7:

$$N\,Eps\,(x) = \{y \in D \mid dist\,(x,y) \leq Eps\} \qquad (2.7)$$

N : Eps (x) menunjukkan titik-titik dalam jarak Eps dari titik x.
D : adalah kumpulan data.
Dist : (x, y) merupakan jarak *Euclidean* antara objek x dan y.
Eps : adalah nilai radius atau ambang batas.

Untuk melakukan pengelompokan data, langkah awalnya adalah menentukan nilai MinPts dan Eps. Secara umum, DBSCAN dimulai dengan memilih titik awal secara acak. Berikut adalah langkah-langkah dari penggunaan DBSCAN dengan menggunakan persamaan 2.8, yaitu;

a) Tentukan titik awal atau p secara acak.

b) Hitung semua jarak titik menggunakan persamaan 2.8

$$E\,(x,y) = \sqrt{\sum_{i=0}^{n} = 0\,(xi - yi\,)^2} \qquad (2.8)$$

c) Tentukan nilai epsilon dan jumlah titik minimum. Penentuan nilai optimal untuk epsilon dan MinPts melibatkan serangkaian percobaan yang harus diulangi beberapa kali. Dalam proses tersebut, nilai *epsilon* dan MinPts yang terpilih adalah yang menghasilkan indeks siluet tinggi dan jumlah *cluster* yang signifikan.

2.3 Implementasi

Pada tahap implementasi dilakukan percobaan pembuatan sistem dengan bahasa pemrograman *Python*. Sistem yang dibuat sesuai dengan alur perancangan yang di mana *dataset* yang dilakukan melalui beberapa tahap *preprocessing*. Setelah itu data akan langsung dilakukan proses iterasi pencarian parameter terbaik menggunakan DE sehingga parameter yang di usulkan oleh metode DE bisa digunakan untuk pengolahan data *cluster* DBSCAN untuk mendeteksi anomali pada data.

2.4 Analisa dan Pengujian

Pada tahap pengujian dilakukan *testing* saat praproses melakukan iterasi menggunakan metode *Differential Evolution*. Dan juga ada beberapa pengujian yaitu ;

a. Pengukuran akurasi *cluster* menggunakan metode *Silhouette score* [14] dan *Davies-Bouldin Index* (DBI) [15] pada optimasi parameter MinPts dan *Epsilon*.

b. Pengukuran akurasi anomali pada *cluster* dengan menggunakan metode *Z-score* [16] dan Reconstruction Error Mean Squared Error (RE-MSE) [17].

Evaluasi yang digunakan untuk mengukur performa dari DBSCAN. *Silhouette score* dapat digunakan untuk mengukur performa dari *cluster* yang dihasilkan oleh DBSCAN.

3. HASIL DAN PEMBAHASAN

Penggunaan *Differential Evolution* (DE) dilakukan pada awal percobaan dengan menerapkan persamaan yang telah diuraikan sebelumnya. Tahap pertama penggunaan DE yaitu membangkitkan populasi individu. Individu di sini adalah nilai dari MinPts dan Eps. menggunakan persamaan (2.1) setelah itu dilakukan pengukuran fungsi *fitness* menggunakan metode DBSCAN untuk menghitung *Euclidean distance* pada data transaksi menggunakan persamaan (2.5 – 2.8). Untuk menghitung *score* awal pada percobaan setiap individu menggunakan metode *Silhouette score*. Dapat dilihat pada gambar 3 sebagai berikut.

```
MaxIt = 10
nPop = 10
nVar = 2
bounds = [(2, 20), (0.1, 2)]

# Inisialisasi populasi dengan nilai awal yang telah ditentukan
Population = np.zeros((nPop, nVar))
for i in range(nPop):
    Population[i, :] = [np.random.randint(bounds[0][0], bounds[0][1]),  # MinPts
                        np.random.uniform(bounds[1][0], bounds[1][1])]  # Eps

def fitness_function(individual, scaled_data):
    dbscan = DBSCAN(eps=individual[1], min_samples=int(individual[0])).fit(scaled_data)
    labels = dbscan.labels_
    n_clusters = len(set(labels)) - (1 if -1 in labels else 0)
    if n_clusters > 1:
        silhouette_avg = silhouette_score(scaled_data, labels)
    else:
        silhouette_avg = -1  # Skor siluet tidak valid jika hanya ada 1 cluster
    return silhouette_avg, {'MinPts': individual[0], 'Eps': individual[1], 'nClusters': n_clusters}
```

Gambar 3. *Code* membangkitkan Individu dan *Fitness*

Pada gambar 3 menampilkan kode untuk menginisialisasi populasi berukuran 10 dengan dua parameter DBSCAN (MinPts dan Eps) dalam batas yang telah ditentukan, lalu

mendefinisikan fungsi *fitness* yang mengevaluasi setiap individu dalam populasi berdasarkan skor siluet yang dihasilkan dari *clustering* DBSCAN pada data yang telah diskalakan, mengembalikan nilai siluet dan informasi parameter kluster.

Setelah membangkitkan individu awal. Selanjutnya DE akan memproses dengan beberapa tahapan yaitu mutasi, perhitungan *crossover* dan seleksi individu baru sesuai dengan ketentuan persamaan yang ada. Pada percobaan ini menggunakan persamaan (2.2 – 2.4). dapat dilihat pada gambar 4 berikut.

```
def mutate(individuals, target_idx, F=0.8):
    idxs = [idx for idx in range(len(individuals)) if idx != target_idx]
    a, b, c = np.random.choice(idxs, 3, replace=False)
    mutant = individuals[a] + F * (individuals[b] - individuals[c])
    return np.clip(mutant, [bounds[0][0], bounds[1][0]], [bounds[0][1], bounds[1][1]])

def crossover(target, mutant, CR=0.7):
    cross_points = np.random.rand(nVar) < CR
    trial = np.where(cross_points, mutant, target)
    return trial

def select(target, trial, scaled_data):
    fitness_ind, _ = fitness_function(target, scaled_data)
    fitness_trial, _ = fitness_function(trial, scaled_data)
    return trial if fitness_trial > fitness_ind else target
```

Gambar 4. *Code* implementasi *Mutate*, *Crossover* dan Seleksi

Kode pada gambar 4 mengimplementasikan tiga komponen utama dari algoritma DE yaitu mutasi, *crossover*, dan seleksi. Fungsi *mutate* menghasilkan vektor mutan dengan memilih tiga individu acak dari populasi dan menggabungkannya dengan faktor skala F yang tetapkan adalah 0.8, lalu memastikan hasilnya tetap dalam batas yang ditentukan. Fungsi *Crossover* membuat vektor percobaan dengan menggabungkan elemen-elemen dari individu target dan vektor mutan berdasarkan probabilitas *crossover* CR yang tetapkan adalah 0.7. Fungsi *select* menentukan apakah individu target atau vektor percobaan yang akan masuk ke generasi berikutnya dengan membandingkan nilai *fitness* mereka dan memilih yang memiliki *fitness* lebih tinggi. Setelah proses *select* akan ditetapkan pengulangan percobaan iterasi hingga proses mencapai batas maksimal atau sudah tidak ada perubahan pada pengujian *Silhouette score*. Data yang akan ditampilkan berupa iterasi percobaan setiap individu terbaik pada setiap iterasi.

Differential Evolution (DE) melakukan satu kali percobaan pada *dataset* dengan ±1011 *record* data transaksi. Menghasilkan lima iterasi pemilihan parameter dengan menampilkan hasil *cluster* dari penggunaan MinPts dan *Epsilon* pada DBSCAN. Iterasi dapat dilihat pada gambar 5 sebagai berikut.

```
Iterasi Ke -  1  Parameter :  {'MinPts': 12.600000000000001, 'Eps': 1.4287135653962773, 'nClusters': 3}
Iterasi Ke -  2  Parameter :  {'MinPts': 16.0, 'Eps': 0.15232448270225807, 'nClusters': 8}
Iterasi Ke -  3  Parameter :  {'MinPts': 7.0, 'Eps': 1.674463695487391, 'nClusters': 2}
Iterasi Ke -  4  Parameter :  {'MinPts': 11.000000000000002, 'Eps': 2.0, 'nClusters': 2}
Iterasi Ke -  5  Parameter :  {'MinPts': 7.960000000000001, 'Eps': 2.0, 'nClusters': 2}
```

Gambar 5. Hasil iterasi penggunaan DE

Dapat dilihat pada gambar 5 hasil iterasi yang diberikan pada setiap iterasi memiliki *nCluster* yang berbeda pada iterasi ke-1 dengan *Minpts* 12,6 dan *Epsilon* 1.428..3 mendapatkan *nCluster* sebanyak 3. Iterasi ke-2 dengan *MinPts* 16 dan *Epsilon* 0,152..7 mendapatkan *nCluster* 8. Dan iterasi tiga terakhir dengan menghasilkan *nCluster* 2 cenderung memberikan optimasi dari *Epsilon* ≤ 2. Pada percobaan penggunaan DE ke-2 mendapatkan hasil parameter yang lebih stabil bisa dilihat pada gambar 4.

```
Iterasi Ke -  1  Parameter :  {'MinPts': 4.0, 'Eps': 1.8296508691942042, 'nClusters': 3}
Iterasi Ke -  2  Parameter :  {'MinPts': 10.0, 'Eps': 1.6429907465044589, 'nClusters': 2}
Iterasi Ke -  3  Parameter :  {'MinPts': 8.0, 'Eps': 1.8725944223109998, 'nClusters': 2}
Iterasi Ke -  4  Parameter :  {'MinPts': 8.0, 'Eps': 2.0, 'nClusters': 2}
```

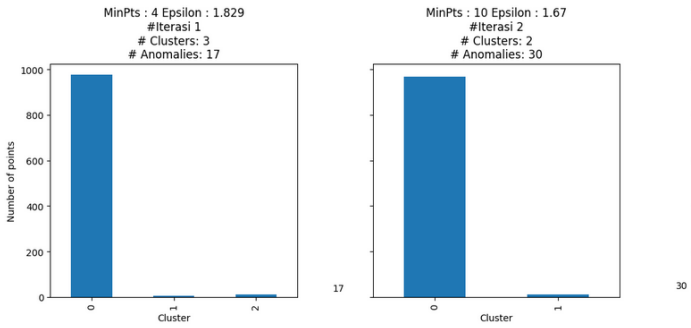
Gambar 6. Hasil Iterasi penggunaan DE ke-2

Pada gambar 6 memberikan informasi percobaan ke-2 penggunaan DE untuk optimasi parameter mendapatkan 4 iterasi yang di mana parameter dan *nCluster* yang diberikan cenderung sama atau lebih stabil. Selanjutnya dilakukan evaluasi *nCluster* menggunakan metode *Silhouette score* untuk mengukur seberapa akurat parameter yang diberikan pada percobaan DE yang ke-2 dapat dilihat pada tabel 3.

Tabel 3. Akurasi dari pengukuran *cluster* yang diuji

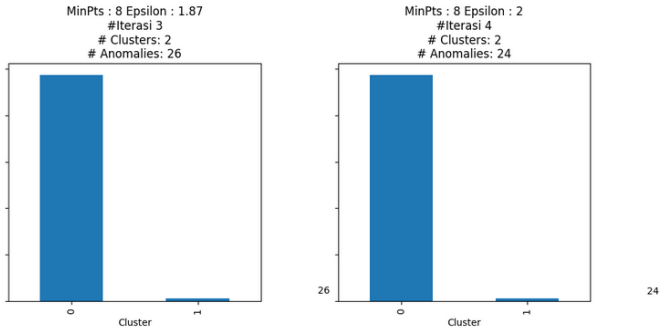
Iterasi	*MinPts*	*Epsilon*	*nCluster*	*Silhouette score*	*DBI*
1	4	1.829	3	0.7836	0.19955
2	10	1.67	2	0.7926	0.19791
3	8	1.87	2	0.79463	0.19995
4	8	2	2	0.79563	0.20149

Tabel 3 memberikan informasi mengenai seberapa akurat *nCluster* dari percobaan ke-2 penggunaan DE untuk optimasi parameter *MinPts* dan *Epsilon* pada DBSCAN. Dapat dilihat pada iterasi pertama akurasi yang diperoleh mencapai 0.7836 menggunakan *Silhouette score* dan akurasi DBI 0.19955, iterasi kedua diperoleh akurasi mencapai 0.7926 dan 0.19791, iterasi ketiga memperoleh akurasi mencapai 0.79463 dan 0.19995, iterasi terakhir memperoleh akurasi hingga mencapai 0.79563 dan 0.20149. Dapat dilihat peningkatan akurasi yang tidak terlalu signifikan, tetapi pada setiap iterasi yang dilakukan selalu ada peningkatan untuk akurasi *nCluster* yang diberikan dari optimasi yang dilakukan metode DE tersebut. Kemudian parameter akan diuji untuk mendeteksi anomali oleh DBSCAN dapat dilihat pada gambar 5 untuk iterasi 1 dan 2 dan gambar 6 untuk iterasi 3 dan 4 sebagai berikut.



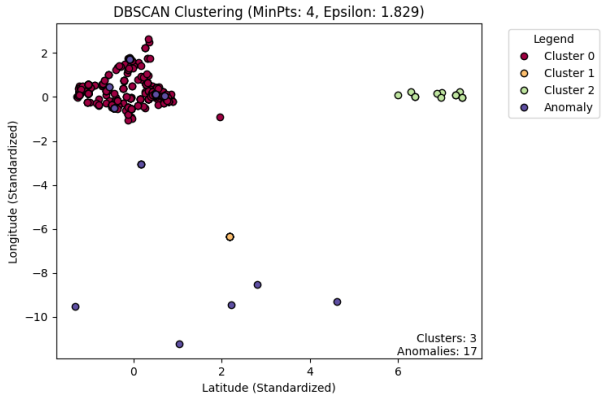
Gambar 7. Hasil deteksi anomali iterasi (a) dan (b)

Bisa dilihat pada gambar 7 menampilkan hasil dari deteksi *anomaly* menggunakan DBSCAN pada iterasi 1 dan 2, di mana iterasi pertama mampu mendeteksi 17 anomali pada 3 *cluster*. Selanjutnya iterasi kedua mampu mendeteksi 30 *anomaly* data dari 2 *cluster*.



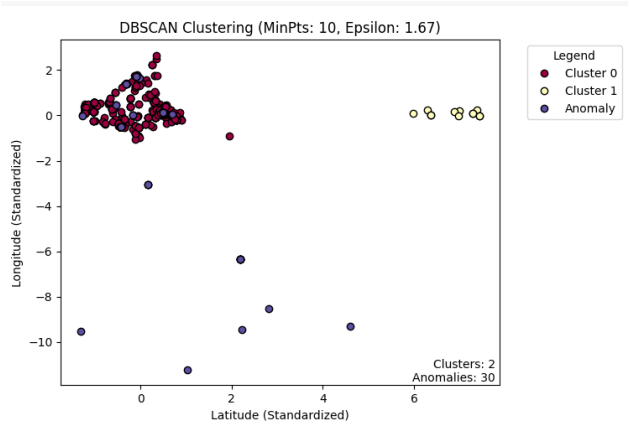
Gambar 8. Hasil deteksi anomali iterasi (c) dan (d)

Selanjutnya gambar 8 menunjukkan iterasi ke-3 dan ke-4 pada penggunaan parameter iterasi ke-3 dengan minPts 8 dan *Epsilon* 1.87 dapat mendeteksi *anomaly* sebanyak 26. Iterasi ke-4 dengan minPts 8 dan *Epsilon* 2 dapat mendeteksi sebanyak 24 *anomaly* data. Nilai MinPts yang lebih tinggi seperti pada iterasi ke-2 menghasilkan jumlah *cluster* yang lebih sedikit dan jumlah *anomaly* yang lebih banyak. Ini menunjukkan bahwa dengan meningkatkan nilai MinPts, algoritma menjadi lebih ketat dalam menentukan kepadatan yang diperlukan untuk membentuk kluster, sehingga lebih banyak *poin* yang dianggap sebagai *noise* atau *anomaly*. Nilai *Epsilon* yang lebih tinggi seperti pada iterasi ke-4 tampaknya tidak terlalu mempengaruhi jumlah kluster, yang tetap berada di angka 2. Namun, *Epsilon* yang lebih besar biasanya memungkinkan poin untuk menjadi bagian dari kluster meskipun jaraknya lebih jauh. Ini mungkin alasan mengapa ada sedikit penurunan dalam jumlah *anomaly* dari iterasi ke-2 ke iterasi ke-4, meski perbedaannya tidak signifikan. Selanjutnya pengujian MinPts dan *Epsilon* menggunakan *scatter plot* dapat dilihat pada gambar berikut.



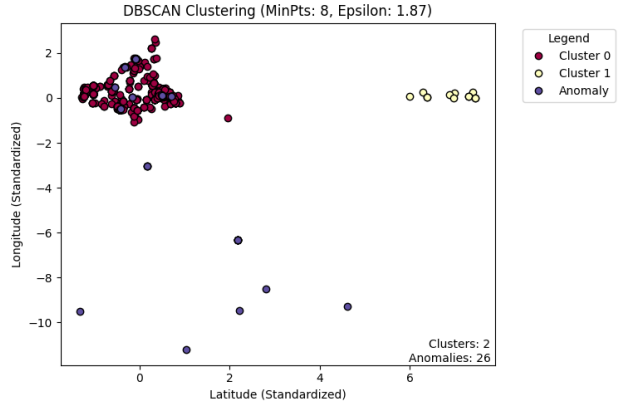
Gambar 9. Hasil *cluster* iterasi ke-1

Dapat dilihat pada gambar 9 parameter DBSCAN pada iterasi ke-1 dengan MinPts = 4 dan *Epsilon* = 1,829. Algoritma DBSCAN ini telah mengidentifikasi sebuah *cluster* besar (merah), sebuah *cluster* yang lebih kecil (ungu), dan beberapa pencilan (hijau) yang dianggap sebagai *anomaly*, pada data yang ada jumlah total *anomaly* adalah 17.



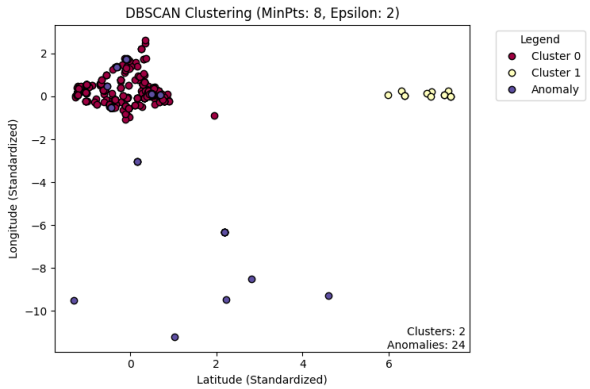
Gambar 10. Hasil *cluster* iterasi ke-2

dalam iterasi ke-2 pada gambar 10 dengan memasukkan parameter MinPts telah meningkat menjadi 10 dan *Epsilon* adalah 1,67. Tampaknya peningkatan MinPts menghasilkan klaster yang lebih tepat dan lebih banyak titik data yang diklasifikasikan sebagai anomali menjadi 30 data.



Gambar 11. Hasil *cluster* iterasi ke-3

Gambar 11 menujukan iterasi ke-3 MinPts berkurang menjadi 8, dan *Epsilon* meningkat menjadi 1,87. Jumlah *anomaly* telah berkurang menjadi 26 dibandingkan dengan iterasi 2, yang menunjukkan bahwa sedikit peningkatan *Epsilon* memungkinkan lebih banyak titik untuk dimasukkan ke dalam *cluster*.



Gambar 12. Hasil *cluster* iterasi ke-4

Terakhir dapat dilihat pada gambar 12 menujukan iterasi ke-4 dengan MinPts = 8 dan *Epsilon* = 2. Jumlah *anomaly* tetap 24, sama seperti iterasi sebelumnya, namun klaster-klaster tersebut tampaknya bergabung karena peningkatan *Epsilon*, yang memungkinkan adanya lingkungan yang lebih besar di sekitar setiap titik.

Dari iterasi yang dilakukan, dapat dilihat bagaimana pilihan parameter mempengaruhi hasil pengelompokan. Meningkatkan *Epsilon* cenderung menggabungkan *cluster* dan mengurangi jumlah *anomaly*, sementara meningkatkan MinPts umumnya menghasilkan lebih banyak *anomaly* dan *cluster* yang lebih kecil dan lebih berbeda. Ini adalah keseimbangan untuk menemukan parameter yang tepat yang paling sesuai dengan pengelompokan alami dalam data. Selanjutnya pengujian pengukuran *anomaly* menggunakan metode *Z-score* yang akan ditunjukkan pada tabel 4 sebagai berikut:

Tabel 4. Hasil pengukuran *Z-score* pada *anomaly*

Its	*MPts*	*Eps*	*Upper*	*Lower*	*Acurracy*
1	4	1.829	0.02474076	0.00888931	98.41%
2	10	1.67	0.04013322	0.01921396	97.91%
3	8	1.87	0.03547433	0.01595989	98.05%
4	8	2	0.03312282	0.01435493	98.12%

Rata-rata *Z-score* menunjukkan jarak rata-rata yang dihitung dari pusat *cluster* ke titik data, di mana semakin kecil nilai ini, semakin padat *cluster* tersebut. Dari data, skor tertinggi adalah 2.00 dan terendah adalah 1.67. Skor yang lebih rendah pada MPts yang lebih tinggi (10) menunjukkan bahwa *cluster* dengan lebih banyak titik cenderung lebih padat atau homogen. Akurasi mencerminkan seberapa baik model DBSCAN mengidentifikasi titik data sebagai bagian dari *cluster* atau sebagai *anomaly*. Nilai akurasi berkisar dari 97.91% hingga 98.41%. Meskipun perbedaan tidak besar, pengaturan dengan MPts 4 mencapai akurasi tertinggi (98.41%).

Tabel 5. Hasil pengukuran *anomaly* menggunakan MSE

Its	*MPts*	*Eps*	*RE-MSE*	*MSE-A*	*MSE-non A*
1	4	1.829	0.671778	21.37948	0.33879
2	10	1.67	0.935866	19.06420	0.38148
3	8	1.87	0.555537	13.64477	0.21003
4	8	2	0.695493	14.84811	0.35135

Dapat dilihat pada tabel 5 yang disajikan, hasil eksperimen parameter algoritma DBSCAN untuk mengidentifikasi anomali dalam data transaksi bank. dengan fokus pada *Reconstruction Mean Squared Error* (RE-MSE),data yang terdeteksi anomali (MSE-A) serta data yang terdeteksi non-anomali (MSE-non A). Dari keempat percobaan yang dicatat, Iterasi 3 dengan 8 MPts dan Eps 1.87 menunjukkan kinerja terbaik dengan RE-MSE terendah sebesar 0.555537, serta nilai MSE-A yang di indikasikan sangat jauh dari data non anomali mencapai 13.64477 menandakan kemampuan yang efisien dalam mendeteksi anomali dengan tingkat kesalahan yang lebih rendah. Sementara itu, nilai MSE untuk non-anomali juga paling rendah pada iterasi ini, yaitu 0.21003, mengindikasikan bahwa model ini secara akurat merekonstruksi mayoritas data non-anomali.

Dalam penelitian ini, mengusulkan metode kombinasi antara algoritma DBSCAN dan *Differential Evolution* (DE) untuk mendeteksi anomali pada data transaksi bank. Metode ini bertujuan untuk mengatasi tantangan utama dalam penggunaan DBSCAN, yaitu penentuan parameter MinPts dan Epsilon yang optimal. Berdasarkan hasil eksperimen, metode yang diusulkan menunjukkan peningkatan yang signifikan dalam akurasi deteksi anomali dibandingkan dengan pendekatan DBSCAN konvensional. Tahap pertama dalam penelitian ini adalah pengumpulan data transaksi dari Bank XYZ yang melibatkan lebih dari 1011 rekaman. Data tersebut kemudian melalui tahapan praproses seperti pembersihan data, normalisasi, dan transformasi. Praproses ini dilakukan untuk menghapus *missing values* dan memastikan bahwa data dalam format yang sesuai untuk pengelompokan dengan DBSCAN.

Selanjutnya, DE digunakan untuk mengoptimalkan parameter DBSCAN. DE melakukan inisialisasi populasi, mutasi, *crossover*, dan seleksi untuk menemukan kombinasi parameter yang menghasilkan *clustering* terbaik. Proses ini melibatkan beberapa iterasi untuk mengeksplorasi ruang parameter dan mengeksploitasi solusi terbaik yang ditemukan. Hasil iterasi menunjukkan bahwa penggunaan DE mampu meningkatkan akurasi deteksi anomali dengan nilai *Silhouette Score*, *DBI*, *MSE* dan *Z-score* yang lebih baik. Hasil pengelompokan menunjukkan bahwa peningkatan nilai MinPts menghasilkan jumlah *cluster* yang lebih sedikit namun lebih padat, sementara peningkatan nilai Epsilon cenderung menggabungkan *cluster* dan

mengurangi jumlah anomali. Contohnya, pada iterasi ke-2 dengan parameter MinPts=10 dan Epsilon=1.67, jumlah anomali yang terdeteksi meningkat menjadi 30 data, yang menunjukkan bahwa nilai MinPts yang lebih tinggi membuat algoritma lebih ketat dalam menentukan kepadatan *cluster*.

Penggunaan DE juga terbukti efektif dalam menangani data dengan variasi densitas yang tinggi, yang merupakan salah satu keterbatasan utama dari DBSCAN konvensional. DE membantu menemukan kombinasi parameter yang lebih optimal dengan cara yang lebih efisien daripada metode *trial and error* manual. Secara keseluruhan, penelitian ini menunjukkan bahwa kombinasi DBSCAN dan DE adalah pendekatan yang efektif untuk deteksi anomali pada data transaksi bank. Metode ini tidak hanya meningkatkan akurasi deteksi tetapi juga efisiensi dalam penentuan parameter DBSCAN. Penelitian lebih lanjut dapat mengaplikasikan metode ini pada jenis data lain dan menguji ketahanannya terhadap berbagai skenario anomali.

4. KESIMPULAN

Berdasarkan hasil penelitian ini berhasil mengoptimalkan parameter DBSCAN, yaitu MinPts dan Epsilon, menggunakan metode *Differential Evolution* (DE). Hasilnya menunjukkan bahwa optimasi parameter ini meningkatkan akurasi deteksi anomali dalam data transaksi bank. Iterasi yang dilakukan menunjukkan peningkatan akurasi yang stabil menggunakan dengan DBI mencapai 0.19955, dengan hasil terbaik mencapai akurasi Z-Score 98.41% dan MSE mencapai 0.671778 pada penggunaan MinPts 4 dan Eps 1,829 dapat mendeteksi 17 anomali. Penggunaan DE juga membantu mengurangi kesulitan dalam menentukan parameter secara manual. pendekatan yang diusulkan dapat digunakan untuk meningkatkan deteksi anomali dalam berbagai aplikasi yang membutuhkan analisis data transaksi. Dengan menggunakan optimasi parameter yang lebih efisien. Penelitian ini memiliki beberapa batasan, di antaranya adalah penggunaan *dataset* yang terbatas pada transaksi bank tertentu. Selain itu, meskipun metode DE membantu dalam optimasi parameter, hasilnya masih bergantung pada kualitas dan karakteristik data yang digunakan. Variasi data yang berbeda mungkin memerlukan penyesuaian lebih lanjut. Untuk penelitian selanjutnya, direkomendasikan untuk mengaplikasikan metode optimasi ini pada *dataset* transaksi yang lebih besar dan beragam. Penelitian mendatang juga dapat mengeksplorasi kombinasi algoritma *clustering* lainnya dengan metode optimasi berbeda untuk membandingkan hasilnya. Selain itu, pengembangan sistem deteksi anomali yang *real-time* menggunakan pendekatan ini dapat menjadi kontribusi signifikan bagi masyarakat umum.

DAFTAR PUSTAKA

[1] P. Jain, M. S. Bajpai, and R. Pamula, “A Modified DBSCAN Algorithm for Anomaly Detection in Time-series Data with Seasonality,” *International Arab Journal of Information Technology*, vol. 19, no. 1, pp. 23–28, Jan. 2022, doi: 10.34028/iajit/19/1/3.

[2] M. Tatusch, G. Klassen, M. Bravidor, and S. Conrad, “Predicting Erroneous Financial Statements Using a Density-Based Clustering Approach,” in *ACM International Conference Proceeding Series*, Association for Computing Machinery, Aug. 2020, pp. 89–94. doi: 10.1145/3418653.3418673.

[3] I. Kamil and B. Pharmasetiawan, “Fingerprint Presence Fraud Detection Using Tight Clustering on Employee\’s Presence and Activity Data,” 2019. doi: DOI: 10.1109/ICITISEE48480.2019.9003914.

[4] Mohamad Zamini and Gholamali Montazer, *Credit Card Fraud Detection using autoencoder based clustering*. 2018 9th International Symposium on Telecommunications (IST’2018), 2018. doi: DOI: 10.1109/ISTEL.2018.8661129.

[5] M. Yang and D. Ergu, “Anomaly Detection of Vehicle Data Based on LOF Algorithm,” *Frontiers in Signal Processing*, vol. 4, no. 1, Jan. 2020, doi: 10.22606/fsp.2020.41007.

[6] D. Deng, “Research on Anomaly Detection Method Based on DBSCAN Clustering Algorithm,” in *Proceedings - 2020 5th International Conference on Information Science, Computer Technology and Transportation, ISCTT 2020*, Institute of Electrical and Electronics Engineers Inc., Nov. 2020, pp. 439–442. doi: 10.1109/ISCTT51595.2020.00083.

[7] P. Jain, M. S. Bajpai, and R. Pamula, “A Modified DBSCAN Algorithm for Anomaly Detection in Time-series Data with Seasonality,” *International Arab Journal of Information Technology*, vol. 19, no. 1, pp. 23–28, Jan. 2022, doi: 10.34028/iajit/19/1/3.

[8] Z. Zhang, L. Chen, Q. Liu, and P. Wang, “A Fraud Detection Method for Low-Frequency Transaction,” *IEEE Access*, vol. 8, pp. 25210–25220, 2020, doi: 10.1109/ACCESS.2020.2970614.

[9] M. Z. Hossain, M. J. Islam, M. W. R. Miah, J. H. Rony, and M. Begum, “Develop a dynamic DBSCAN algorithm for solving initial parameter selection problem of the DBSCAN algorithm,” *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 23, no. 3, pp. 1602–1610, Sep. 2021, doi: 10.11591/ijeecs.v23.i3.pp1602-1610.

[10] K. I. Ghathwan and A. J. Mohammed, “Intelligent Bat Algorithm for Finding Eps Parameter of DbScan Clustering Algorithm,” *Iraqi Journal of Science*, vol. 63, no. 12, pp. 5572–5580, 2022, doi: 10.24996/ijs.2022.63.12.41.

[11] G. Dominico and R. S. Parpinelli, "Multiple global optima location using differential evolution, clustering, and local search," *Appl Soft Comput*, vol. 108, Sep. 2021, doi: 10.1016/j.asoc.2021.107448.

[12] M. T. Guerreiro *et al.*, "Anomaly detection in automotive industry using clustering methods—a case study," *Applied Sciences (Switzerland)*, vol. 11, no. 21, Nov. 2021, doi: 10.3390/app11219868.

[13] Manoj Kumar Reddy Mallidi and Yeshwanth Zagabathuni, "Analysis of Credit Card Fraud detection using Machine Learning models on balanced and imbalanced datasets," *International Journal of Emerging Trends in Engineering Research*, vol. 9, no. 7, pp. 846–852, Jul. 2021, doi: 10.30534/ijeter/2021/02972021.

[14] H. Sayed Ramadan, H. Amin Maghawry, M. El-Eleamy, and K. El-Bahnasy, "A Heuristic Novel Approch For Determination Of Otimal Epsilon For DBSCAN Clustering Algorithm," *J Theor Appl Inf Technol*, vol. 15, no. 7, 2022, [Online]. Available: www.jatit.org

[15] N. Gholizadeh, H. Saadatfar, and N. Hanafi, "K-DBSCAN: An improved DBSCAN algorithm for big data," *Journal of Supercomputing*, vol. 77, no. 6, pp. 6214–6235, Jun. 2021, doi: 10.1007/s11227-020-03524-3.

[16] H. Liu, Y. Wang, and W. G. Chen, "Anomaly detection for condition monitoring data using auxiliary feature vector and density-based clustering," *IET Generation, Transmission and Distribution*, vol. 14, no. 1, pp. 108–118, Jan. 2020, doi: 10.1049/iet-gtd.2019.0682.

[17] X. Olive and L. Basora, "Identifying Anomalies in past en-route Trajectories with Clustering and Anomaly Detection Methods," 2019. [Online]. Available: https://hal.science/hal-02345597