1. Pendahuluan

1.1 Latar belakang

Sejalan dengan perkembangan teknologi informasi, setiap waktu dipastikan selalu terjadi pertambahan jumlah data tersimpan yang akhirnya dapat menimbulkan problem data *explosion*. Bertambahnya volume data tersebut, tidak diikuti dengan analisis dalam menggali sebuah pengetahuan baru. Hal tersebut yang melatarbelakangi muculnya sebuah teknik yang disebut dengan *data mining*. Dalam *data mining* terdapat empat *task* penting, antara lain (a) *dependency detection*, (b) *class identification*, (c) *class description*, dan (d) *exception/outlier detection* [9]. Tiga *task* pertama fokus kepada data yang memiliki kemunculan kejadian relatif banyak, sedangkan *task* terakhir fokus pada data yang mempunyai kemunculan kejadian relatif sedikit.

Outlier atau sering kali diistilahkan sebagai data yang memiliki karakteristik berbeda dengan kebanyakan data lainnya sering kali ditemukan dalam suatu *dataset* meskipun dalam jumlah yang relatif sedikit. Dari jumlah yang sedikit itu justru dapat memunculkan sebuah *knowledge* yang menarik dan tidak terduga. Deteksi *outlier* memiliki banyak kegunaan, antara lain berfungsi dalam melakukan *preprocessing data*, *data cleaning*, deteksi penyalahgunaan kartu kredit, deteksi adanya penyusupan pada jaringan komunikasi, aplikasi keuangan, prediksi cuaca, *stock market analysis*, analisis medis, dan beberapa aplikasi lain yang memiliki problem utama dalam mendeteksi *rare event, exceptions*, atau objek data yang menyimpang [1,6,8].

Dalam dataset yang besar, deteksi *outlier* dianggap merupakan permasalahan yang sangat signifikan. Ketika dataset cukup kecil, deteksi *outlier* dapat dilakukan secara manual karena masih dapat terlihat dan ditebak secara kasat mata. Namun, ketika pola dari nilai-nilai dalam dataset tidak diketahui secara jelas, hal tersebut akan menyulitkan pendeteksian *outlier*. Oleh sebab itu dibutuhkan suatu metode agar dapat memudahkan dalam melakukan pendeteksian *outlier*. Beberapa pendekatan metode yang dapat melakukan pendeteksian *outlier* antara lain, *distribution based methods*, *depth-based*, *deviation-based*, *distance based*, *density based*, *clustering-based*, *sub-space based*, *support vector based*, dan *neutral network based* [1,4,6]. Namun, seluruh metode tersebut dalam menemukan *rare event* atau *outlier* dalam *dataset* tidak memerhatikan secara khusus pada *class label* dalam kasusnya, tetapi mempertimbangkannya secara utuh pada keseluruhan data. Hal tersebut dapat mengakibatkan data yang seharusnya terdeteksi sebagai *outlier* dalam suatu *label class* tertentu dianggap sebagai objek data yang normal. Metode yang dapat mendeteksi suatu kasus dimana suatu objek tersebut memiliki kelakuan yang berbeda atau menyimpang dari *class*-nya disebut *class outlier detection* [8].

Beberapa peneliti telah melakukan penelitian mengenai *class outlier*, diantaranya Zengyou He, Xiaofei Xu, Joshua Zhexue Huang, dan Shengchun Deng (2004) serta Nabil M. Hewahi dan Motaz K. Saad (2007). Dalam melakukan pendeteksian *class outlier*, mereka tidak hanya memperhitungkan *outlier* yang menyimpang dari kelasnya sendiri, melainkan juga *outlier* yang

menyimpang dari kelas lain, sehingga hampir semua jenis *class outlier* dapat diidentifikasi. Dari hasil penelitiannya, mereka mengenalkan beberapa algoritma yang dapat mendeteksi *class outlier,* antara lain *Frequent Pattern based Class Outlier Detection*, *Cluster based Class Outlier Detection* [6], *Class Outlier Distance Based* (CODB) dan *Enhanced Class Outlier Distance Based* (*ECODB*) [7,8].

Dari beberapa algoritma yang dapat mendeteksi *class outlier,* algoritma *Frequent Pattern based Class Outlier Detection* dan *Cluster based Class Outlier Detection* belum dapat menangani dataset yang bertipe *numeric* atau *mixed* dataset, serta belum dapat menangani dataset yang memiliki lebih dari dua label *class* [6,8], sedangkan algoritma CODB dan ECODB sudah dapat menangani kekurangan tersebut [7,8]. Dalam melakukan pendeteksian *class outlier*, baik algoritma CODB maupun algoritma ECODB melakukan beberapa perhitungan, antara lain perhitungan *K-Distance, Probability of the class label* (PCL), *Deviation,* serta *Class Outlier Factor (COF).* Algoritma ECODB sendiri merupakan pengembangan dari algoritma CODB dengan menghilangkan faktor *heuristic* yang ada dalam algoritma CODB dengan cara melakukan normalisasi *K-Dist* dan *Deviation*.

Melalui Tugas Akhir ini, akan dilakukan analisis tingkat akurasi dan skalabilitas dari hasil implementasi deteksi *class outlier* menggunakan algoritma ECODB pada beberapa dataset bertipe numerik dan kategorikal.

1.2 Perumusan masalah

Pada Tugas Akhir ini, penelitian difokuskan pada analisis dan implementasi deteksi *class outlier* dengan menggunakan algoritma ECODB. Berdasarkan latar belakang masalah, maka beberapa permasalahan utama yang akan dirumuskan antara lain:

1. Bagaimana mengimplementasikan algoritma ECODB dalam mendeteksi *class outlier*?
2. Bagaimana pengaruh inputan nilai K dan nilai N terhadap akurasi dari algoritma ECODB dalam mendeteksi *class outlier*?
3. Bagaimana mengevaluasi skalabilitas dari algoritma ECODB dalam mendeteksi *class outlier* setelah diimplementasikan ke dalam sebuah sistem?

1.3 Tujuan

Tujuan yang ingin dicapai dalam penelitian ini adalah:

1. Mengimplementasikan algoritma ECODB dalam mendeteksi *class outlier*.
2. Menganalisis pengaruh nilai K dan nilai N terhadap akurasi dari algoritma ECODB dalam mendeteksi *class outlier* setelah diimplementasikan ke dalam sebuah sistem.
3. Menguji skalabilitas dari algoritma ECODB dalam mendeteksi *class outlier*.

Hipotesa awal: Dengan memperbesar nilai N dan memilih nilai K yang tepat dapat meningkatkan nilai akurasi dari ECODB hingga mencapai nilai akurasi 0.75. Dengan memperbesar jumlah data dapat menurunkan skalabilitas dari ECODB.

1.4 Batasan masalah

Dalam implementasi tugas akhir ini dibatasi oleh beberapa hal, antara lain:

1. *Preprocessing* data berupa normalisasi dataset yang bertipe numerik dilakukan di luar sistem dengan menggunakan *tools* WEKA.
2. Evaluasi sistem akan dilakukan dengan cara memanfaatkan klasterisasi dengan algoritma K-Means. Proses klasterisasi dilakukan di luar sistem dengan menggunakan *tools* WEKA.

1.5 Metodologi penyelesaian masalah

Metodologi penyelesaian masalah yang akan digunakan untuk permasalahan di atas adalah dengan menggunakan langkah-langkah sebagai berikut :

1. Studi literatur

   Pada tahap ini dilakukan pencarian sumber-sumber bacaan yang terkait dengan konsep deteksi *outlier,* deteksi *class outlier*, algoritma ECODB, pendekatan dan algoritma yang seringkali dipakai dalam mendeteksi *class outlier*, serta informasi lainnya yang dapat menunjang pembuatan tugas akhir ini.
2. Analisis dan Desain

   Pada tahapan ini dilakukan analisis pemecahan masalah dari dataset yang akan dideteksi *class outlier*nya dengan menggunakan dasar teori yang telah dipelajari pada tahapan sebelumnya.
3. Implementasi

   Hasil yang telah dilakukan pada tahapan Analisis dan Desain diimplementasikan ke dalam bahasa pemrograman Borland Delphi.
4. Pengujian dan Analisa Hasil

   Pada tahap ini dilakukan pengujian dan analisis terhadap sistem hasil implementasi dari algoritma ECODB dalam mendeteksi *class outlier.*
5. Penyusunan Laporan

   Pada tahap ini dilakukan pembuatan dokumentasi dari keseluruhan tahapan penyelesaian masalah berupa laporan terhadap penelitian yang dilakukan serta laporan hasil dari penelitian tersebut hingga mendapat sebuah kesimpulan.