William Adriano Alves

ESIREOS: Avaliação Interna Eficiente e Escalável de Métodos Não Supervisionados de Detecção de Anomalias

Sorocaba, SP

25 de Maio de 2023

William Adriano Alves

ESIREOS: Avaliação Interna Eficiente e Escalável de Métodos Não Supervisionados de Detecção de Anomalias

Dissertação de mestrado apresentada ao Programa de Pós-Graduação em Ciência da Computação (PPGCC-So) da Universidade Federal de São Carlos como parte dos requisitos exigidos para a obtenção do título de Mestre em Ciência da Computação. Linha de pesquisa: Sistemas Computacionais.

Universidade Federal de São Carlos – UFSCar

Centro de Ciências em Gestão e Tecnologia – CCGT

Programa de Pós-Graduação em Ciência da Computação – PPGCC-So

Orientador: Prof. Dr. Murilo C. Naldi

Sorocaba, SP

25 de Maio de 2023

Alves, William Adriano

ESIREOS: Avaliação internal, eficiente e escalável de métodos não supervisionados de detecção de anomalias / William Adriano Alves -- 2023.
56f.

Dissertação (Mestrado) - Universidade Federal de São Carlos, campus Sorocaba, Sorocaba
Orientador (a): Murilo Coelho Naldi
Banca Examinadora: Diego Furtado Silva, Márcio Porto Basgalupp
Bibliografia

1. Detecção de anomalias. 2. Mineração de dados. 3. Computação Paralela Massiva. I. Alves, William Adriano. II. Título.

Ficha catalográfica desenvolvida pela Secretaria Geral de Informática (SIn)

DADOS FORNECIDOS PELO AUTOR

Bibliotecário responsável: Maria Aparecida de Lourdes Mariano - CRB/8 6979

UNIVERSIDADE FEDERAL DE SÃO CARLOS

Centro de Ciências em Gestão e Tecnologia
Programa de Pós-Graduação em Ciência da Computação

Folha de Aprovação

Defesa de Dissertação de Mestrado do candidato William Adriano Alves, realizada em 25/05/2023.

Comissão Julgadora:

Prof. Dr. Murilo Coelho Naldi (UFSCar)

Prof. Dr. Márcio Porto Basgalupp (UNIFESP)

Prof. Dr. Diego Furtado Silva (UFSCar)

O Relatório de Defesa assinado pelos membros da Comissão Julgadora encontra-se arquivado junto ao Programa de Pós-Graduação em Ciência da Computação.

Resumo

A detecção de anomalias (*outliers*) é um dos principais problemas em mineração de dados, uma vez que anomalias podem representar informações importantes em numerosas áreas. Vários métodos foram desenvolvidos para identificá-las, especialmente métodos não supervisionados, foco deste trabalho. Endereçando a necessidade de estudos para avaliar e quantificar a qualidade do resultado desses métodos não supervisionados, o índice `IREOS` foi proposto como a primeira técnica de avaliação interna para métodos não supervisionados de detecção de anomalias. O `IREOS` permite selecionar o melhor algoritmo e parâmetros para um dado problema usando apenas informações intrínsecas dos dados. No entanto, o `IREOS` exige o treinamento de muitos classificadores com alta complexidade para cada objeto no conjunto de dados cujas soluções de detecção de anomalias estão sendo analisadas. Essa característica limita a aplicação do `IREOS` a conjuntos de dados pequenos, já que os classificadores necessários utilizam todos os objetos no conjunto de dados durante o treinamento. No presente trabalho, propomos o `ESIREOS`, a primeira versão do `IREOS` que aborda suas deficiências de desempenho e processamento usando técnicas de computação paralela massiva, eficientemente utilizadas para o escalonamento computacional horizontal de muitos problemas de aprendizado de máquina. O `ESIREOS` também usa grafos de vizinhos mais próximos aproximados para reduzir o volume de dados e o poder de processamento exigidos pelo `IREOS` sem perda significativa na qualidade dos resultados. Avaliamos o `ESIREOS` teoricamente, estimando sua complexidade assintótica, e via experimentos com conjuntos de dados reais e sintéticos, para atestar sua eficácia e eficiência em comparação a versão original, incluindo conjuntos de dados com grande volume. Os resultados apresentam que o `ESIREOS` obteve uma melhora significativa na complexidade computacional em comparação à versão original do `IREOS`, mantendo a qualidade dos resultados. O `ESIREOS` mostrou-se capaz de avaliar soluções para conjuntos de dados muito grandes, mesmo aqueles para os quais o `IREOS` não conseguiu avaliar em um tempo viável. Portanto, esta nova versão eficiente e escalável pode ser usada em muitos cenários, principalmente, mas não limitados a, aqueles com dados grandes ou distribuídos.

Palavras-chaves: Mineração de Dados. Detecção de Anomalias. MPC.

Abstract

Anomaly (*outlier*) detection is one of the main problems in data mining. Since anomalies can translate into important information in numerous fields, several methods were developed to identify them, especially unsupervised methods, which is the focus of this work. To soften the need for studies on assessing and quantifying the quality of the result of these unsupervised methods, the `IREOS` index was proposed as the first internal evaluation technique for unsupervised anomaly detection methods. `IREOS` allows one to select the best algorithm and parameters for a given problem using only intrinsic information from the data. However, `IREOS` demands the training of many highly complex classifiers for each object in the dataset whose outlier detection solutions are being analyzed. This feature limits the application of `IREOS` to small datasets since the classifiers use all points in the dataset during its training. In the present work, we propose `ESIREOS`, the first version of `IREOS` that addresses its performance and processing deficiencies using Massive Parallel Computing techniques that efficiently implement horizontal computational scaling for many machine learning problems. `ESIREOS` also makes use of approximated Nearest Neighbor Graphs to reduce the volume of data and processing power demanded by `IREOS` without any significant loss in the quality of the results. We evaluate `ESIREOS` theoretically, estimating its asymptotic complexity and with experiments over real and synthetic datasets to attest to its effectiveness and performance compared to the original version, including large datasets. The results showed that `ESIREOS` resulted in a significant improvement in computational complexity when compared to the original `IREOS` while maintaining quality. `ESIREOS` showed to be capable of evaluating solutions for very large datasets, even those which `IREOS` was not capable of evaluating in a feasible time. Therefore, this efficient and scalable new version can be used in many scenarios, mainly, but not limited to, those with large or distributed data.

Key-words: Data Mining. Outlier Detection. MPC.

Lista de ilustrações

Lista de Algoritmos

Lista de tabelas

Lista de abreviaturas e siglas

IREOS	*Internal, Relative Evaluation of Outlier Solutions*
KLR	*Kernel Logistic Regression*
MPC	*Massive Parallel Computing*
SVM	*Suport Vector Machine*
ESIREOS	*Efficient, Scalable, Internal, Relative Evaluation of Outlier Solutions*
AUC ROC	*Area Under the Curve ROC*
K-NNG	*K-Nearest Neighbour Graph*
NN-Descent	*Nearest Neighbor Descent*
ScaNN	*Scalable Nearest Neighbors*
Annoy	*Approximate Nearest Neighbors Oh Yeah*
MDCGenPy	*Multidimensional Dataset for Clustering Generator in Python*
OpenHPC	*Open High Performance Computing*
RDD	*Resilient Distributed Dataset*
OvA	*Outlier Validity index based on Aggregation*

Lista de símbolos

γ	Letra grega minúscula gama
Γ	Letra grega Gama
$\in$	Pertence
$\subset$	Contida
O	Limite superior
$\times$	Multiplicação

Sumário

1 Introdução

A detecção de anomalias é um importante problema de mineração de dados (AGGARWAL, 2017). Essa tarefa visa identificar objetos cuja distribuição difere significativamente dos outros pontos de dados em um conjunto (YAN; CAO; RUNDENSTEINER, 2017). Sua relevância se deve ao fato de que anomalias frequentemente se traduzem em informações importantes em diferentes domínios de aplicação (CHANDOLA; BANERJEE; KUMAR, 2009), como padrões de compras incomuns, dado o histórico de um usuário (PORWAL; MUKUND, 2019); doenças cardiovasculares indicadas por peculiaridades nos batimentos cardíacos (CHANDOLA; BANERJEE; KUMAR, 2009); ou até mesmo o surgimento de um novo tópico de discussão em redes sociais (MILJKOVIĆ, 2010).

Ao longo dos anos, diferentes técnicas de detecção de anomalias foram propostas. Elas podem ser categorizadas em três tipos, conforme a disponibilidade (ou não) de dados rotulados (HAN; KAMBER; PEI, 2012): (I) técnicas supervisionadas que pressupõem a existência de dados rotulados para o treinamento do modelo; (II) semi-supervisionadas, que utilizam apenas dados previamente rotulados como normais (*inliers*) para aprender uma definição de "normal" a partir deles e classificar como *outliers* aqueles que não se enquadram nessa definição; e (III) técnicas não supervisionadas, que não requerem observações rotuladas, aprendendo a partir das características dos dados. Além disso, estas técnicas podem ser classificadas como binárias, quando atribuem rótulos (*inlier* ou *outlier*) a cada objeto, ou como métodos de pontuação (*scoring*), quando atribuem uma pontuação a cada observação com base em sua "normalidade". Se o número de anomalias n for conhecido, pode-se rotular as n observações com maior pontuação como anomalias, reduzindo uma solução de pontuação para uma solução binária (o foco deste trabalho).

A seleção e avaliação de modelos tornou-se um passo importante para produzir a melhor solução possível de detecção de anomalias, tendo em vista a variedade de técnicas disponíveis. Para métodos supervisionados e semi-supervisionados, essa etapa pode ser realizada com a ajuda de medidas de validação externa. Essas medidas utilizam informações externas aos dados para avaliar um modelo, como rótulos indicando se uma observação é uma anomalia.

No entanto, é comum que informações externas não estejam disponíveis no mundo real, dificultando o uso de técnicas supervisionadas. Além disso, obter as informações necessárias, em muitos cenários de aplicação, pode ser muito custoso em tempo e recursos. Por exemplo, um especialista de domínio pode precisar rotular manualmente cada objeto no conjunto de dados. Portanto, são necessários métodos de validação interna, o foco deste trabalho, que não utilizam informações externas para avaliar o modelo. Em vez disso, estas

medidas avaliam o modelo com base apenas em informações intrínsecas do conjunto de dados.

Vários métodos de validação interna foram desenvolvidos em um campo relacionado, o de agrupamento (FULLÉR, 2018). No entanto, na detecção não supervisionada de anomalias, esse problema recebeu pouca atenção por um longo tempo, até o surgimento do `IREOS` (do inglês *Internal, Relative Evaluation of Outlier Solutions*) (MARQUES et al., 2015; MARQUES et al., 2020).

A partir da definição de que uma anomalia é uma observação que se desvia tanto das outras observações a ponto de levantar suspeitas de que ela foi gerada por um mecanismo diferente (HAWKINS, 1980), o `IREOS` avalia internamente diferentes soluções geradas por métodos de detecção de anomalias. Para uma coleção de observações multivariadas, o `IREOS` avalia as soluções com base na "separabilidade" das observações. Essa medida de separabilidade é calculada por meio de classificadores de margem máxima, sendo necessário um conjunto de classificadores para cada observação analisada. Soluções com maior "separabilidade" dos objetos rotulados como anomalias são melhor avaliadas.

O `IREOS` não está limitado a nenhum método de classificação específico, mas, na publicação original (MARQUES et al., 2015), os autores usam Regressão Logística de Kernel (*KLR*) como método de classificação. O *KLR* usa todos os objetos do conjunto para traçar a barreira de decisão, com uma complexidade assintótica de $O(n^3)$. No restante deste trabalho, consideraremos a complexidade do classificador usado na publicação original.

Experimentos realizados por Marques et al. (2015) demonstraram com sucesso a eficácia do `IREOS`. No entanto, o método não é computacionalmente viável para conjuntos de dados grandes por dois motivos: (I) O `IREOS` exige o treinamento de muitos modelos de classificação com alta complexidade para cada objeto rotulado como anomalia no conjunto de dados cujas soluções de detecção de anomalias estão sendo analisadas, e o número de anomalias em um conjunto de dados tende a ser proporcional ao tamanho do conjunto de dados (YU et al., 2017), exigindo o treinamento de um número proporcional de classificadores; (II) os classificadores usam todo o conjunto de dados durante o treinamento, embora apenas os vizinhos mais próximos do objeto em análise sejam necessários para obter uma barreira de decisão. Portanto, quanto maior o conjunto de dados, mais dados são usados para treinar os modelos de classificação com complexidade de $O(n^3)$, aumentando o consumo de memória e processamento. Como computadores têm uma limitação física em seus recursos, há uma limitação no tamanho dos conjuntos de dados que podem ser analisados com métodos executados em apenas uma única máquina, como o `IREOS`.

Outras áreas de aprendizado de máquina que sofreram limitações semelhantes às do `IREOS`, como otimização de parâmetros e conjuntos de modelos (*ensembles*), buscaram utilizar técnicas de computação paralela massiva (NAVARRO; HITSCHFELD-KAHLER; MATEU, 2014a) para, por exemplo, dividir o treinamento dos diferentes modelos de

classificação necessários em várias máquinas, em uma abordagem conhecida conceitualmente como escalonamento horizontal. Essas técnicas endereçam questões relacionadas aos limites computacionais dos recursos de uma única máquina. No entanto, o escalonamento horizontal de um método envolve desafios em relação à forma como os dados devem ser armazenados e distribuídos entre as várias máquinas que realizarão o treinamento dos modelos, como: acesso aos pontos de dados necessários para treinamento; realização de computação paralela e concorrente; garantia de resiliência a múltiplos erros, possivelmente distribuídos; alocação e liberação de vários recursos; entre outros problemas intrínsecos à paralelização e distribuição de recursos.

As limitações restantes do `IREOS` original advêm da necessidade de usar todos os objetos no conjunto de dados ao treinar os modelos. Mesmo quando o treinamento é distribuído em várias máquinas, as limitações de memória ainda podem ser um problema devido ao tamanho dos conjuntos. Por exemplo, ao usar Máquinas de Vetores de Suporte (SVM) como classificadores, apenas o objeto sendo analisado e seus vizinhos mais próximos são necessários para traçar a barreira de decisão e calcular a separabilidade necessária. No entanto, encontrar a vizinhança de cada ponto de dados também pode ser desafiador e computacionalmente complexo, especialmente para dados distribuídos.

Tarefas como a detecção dos vizinhos mais próximos de um objeto, em geral, têm complexidade $O(n^2)$ quando se é necessário comparar, par a par, todos os objetos para se obter a ordem exata dos vizinhos. Pesquisas recentes buscando reduzir a complexidade dessas tarefas de análise de vizinhança, como a realizada por Dong, Moses e Li (2011), propõem o uso de heurísticas aproximadas para calcular a similaridade entre pares de objetos. Ao arriscar reduzir uma pequena quantia da precisão dos resultados, é possível realizar a análise de vizinhança com complexidade computacional próxima a linear (DONG; MOSES; LI, 2011).

Portanto, as limitações do `IREOS` relacionadas ao volume de dados e disponibilidade de recursos podem ser superadas distribuindo os dados em diferentes máquinas, utilizando um método computacionalmente eficiente para encontrar o subconjunto de vizinhos mais próximos de cada objeto de dados e treinando os vários modelos de classificação com esses subconjuntos em paralelo. No entanto, não deve haver perda significativa na qualidade do índice final gerado pelo `IREOS`, ou seja, a classificação das soluções de detecção de anomalias candidatas não deve mudar significativamente.

Neste trabalho, abordamos a limitação do `IREOS` em relação a conjuntos de dados grandes recorrendo a técnicas de computação paralela massiva que permitem o escalonamento horizontal do `IREOS`, permitindo treinar os modelos de classificação paralelamente e de maneira distribuída, ao mesmo tempo em que lidamos com desafios relacionados à distribuição. Além disso, visamos reduzir o consumo de memória e processamento relacionado ao treinamento dos classificadores necessários, usando SVM como classificadores e

reduzindo o número de objetos de dados utilizados, alcançado por meio da detecção de vizinhos mais próximos de maneira aproximada, mantendo a complexidade da análise de vizinhança próxima à linear sem perda significativa de qualidade dos resultados.

1.1 Objetivos

- Propomos um método para escalonar a indução de múltiplos classificadores usando heurísticas aproximadas e técnicas de computação paralela massiva.
- Avaliamos o impacto sobre a qualidade dos modelos de classificação gerados por amostragens criadas por diferentes métodos de análise de vizinhança aproximados.
- Apresentamos o `ESIREOS`, um índice interno escalável que avalia soluções de métodos de detecção de anomalias que podem lidar com aplicações em vários cenários, especialmente aqueles com grandes volumes de dados.

2 Trabalhos Relacionados

Neste capítulo, fornecemos uma visão geral simplificada do campo de avaliação de soluções de detecção de anomalias, apresentamos o método de avaliação interna de soluções de detecção de anomalias chamado `IREOS` e discutimos as limitações do `IREOS` relacionadas à avaliação de grandes conjuntos de dados. Em seguida, discutimos possíveis soluções para as limitações do `IREOS`, como técnicas de computação paralela massiva e análise de vizinhança de maneira aproximada.

2.1 Avaliação de Soluções de Detecção de Anomalias

A avaliação dos resultados de métodos de detecção de anomalias não supervisionados tem sido quase restrita a experimentos realizados com conjuntos de dados famosos do campo de classificação (MARQUES et al., 2015), como os presentes no trabalho realizado por Campos et al. (2016). Em um procedimento conhecido como avaliação externa, as soluções produzidas por diferentes métodos são comparadas com alguma informação externa, como rótulos. Campos et al. (2016) aplicam a avaliação externa selecionando alguns objetos de uma classe específica de um conjunto de dados para serem rotuladas como *outliers*, enquanto as observações das demais classes no conjunto de dados são rotuladas como *inliers* (ZIMEK; CAMPELLO; SANDER, 2014). Em seguida, soluções produzidas por diferentes métodos de detecção de anomalias são comparadas com os rótulos verdadeiros para avaliar o desempenho dessas soluções. No entanto, a necessidade de dados rotulados exigidos pelas medidas de avaliação externa não é consistente com a ideia de aprendizado não supervisionado (MARQUES et al., 2015).

Medidas de avaliação interna, que não exigem dados rotulados, foram introduzidas no campo de detecção de anomalias por Yousri (2010). O autor propõe um índice chamado *OvA* (*Outlier Validity index based on Aggregation*), construído a partir da combinação de quatro medidas estatísticas que avaliam a separabilidade dos grupos formados pelos objetos de um conjunto de dados e pela medida de similaridade entre cada objeto e seu vizinho mais próximo.

A primeira medida estatística é a distância média intra-grupo, que mede a dissimilaridade média entre cada objeto e seu centroide. A segunda medida é a distância média inter-grupo, que mede a dissimilaridade média entre os centroides de cada par de grupos. A terceira medida é a variação da distância inter-grupo, que mede a variância das distâncias entre os centroides de cada par de grupos. E, finalmente, a quarta medida é a proporção entre a distância média intra-grupo e a distância média inter-grupo.

Essas quatro medidas estatísticas são utilizadas para calcular a pontuação *OvA* de cada ponto de dados no conjunto de dados. A pontuação é baseada na ideia de que um ponto é considerado um *outlier* se ele estiver longe dos grupos e mais próximo dos seus vizinhos mais próximos.

Assim, a pontuação *OvA* é definida como a soma ponderada dessas quatro medidas estatísticas, onde os pesos são ajustados para dar mais importância à medida que melhor descreve o conjunto de dados. A pontuação final *OvA* é normalizada em uma escala de 0 a 1, onde valores mais altos indicam maior probabilidade de um ponto ser uma anomalia.

Com base no trabalho de Yousri (2010), Marques et al. (2015) apresentaram o `IREOS`, um método interno e relativo de avaliação de soluções de detecção de anomalias, permitindo a seleção do melhor modelo e parametrização para um dado problema.

2.2 IREOS

Seja $X = x_1, ..., x_n$ um conjunto de dados contendo n objetos d-dimensionais x_i. Suponha que um ou mais métodos de detecção de anomalias não supervisionados produzirão, para esse conjunto de dados, c soluções binárias *top-m*, ou seja, cada método retornará um subconjunto $S \subset X$ contendo os m objetos com maior probabilidade de serem anomalias, segundo a definição de anomalia adotada pelo método, resultando em um conjunto $C = S_1, ..., S_c$ de c soluções de detecção de anomalias candidatas S_j para X.

O `IREOS` mede a separabilidade de cada objeto $x_i \in S_j$ por meio do tamanho da margem entre o objeto e uma barreira de decisão que o separa de todas as outras observações em X. Essa barreira é obtida por meio de um classificador de margem máxima (que pode atingir uma complexidade de $O(n^3)$ ao usar classificadores *KLR*, como feito em Marques et al. (2015)).

A barreira de decisão precisa ser não linear para separar objetos específicos, de acordo com Marques et al. (2015). Portanto, o `IREOS` requer um classificador com uma função *kernel* para transformar o problema original (possivelmente não linearmente separável) em um problema linearmente separável.

Para produzir um efeito semelhante ao da ordem de um *kernel* polinomial, o `IREOS` utiliza um valor γ para controlar a capacidade de discriminação de um classificador baseado em *kernel*, começando linearmente com a primeira ordem e se tornando cada vez mais não linear conforme a ordem de γ aumenta.

Uma característica do `IREOS` é que o usuário não precisa escolher um valor particular de γ (MARQUES et al., 2015), uma vez que o `IREOS` usa classificadores apenas para medir o nível de dificuldade de se discriminar um objeto dos demais no conjunto de dados, e não está interessado no classificador em si ou em seu desempenho em dados novos.

Em vez de usar um único valor de γ, o IREOS calcula um valor γ_{max}, a ordem polinomial na qual todos os objetos rotulados como anomalias pelas soluções podem ser individualmente discriminados dos outros objetos no conjunto de dados. O IREOS então discretiza o intervalo $[0, \gamma_{max}]$ em y valores inteiros de 0 a γ_{max}, e em seguida mede a separabilidade média p de um objeto x_i computando a *área sob a curva ROC* (*AUC ROC*) em todo esse intervalo de valores de γ.

Portanto, para cada objeto $x_i \in S_j$, y classificadores são treinados para calcular a separabilidade media p deste objeto, de modo que essa medida seja independente da ordem polinomial da função *kernel* usada pelos classificadores.

No cenário em que temos uma coleção C de soluções S_j, com $|C| = c$ e $|S| = m$, $c \times y \times m$ classificadores são treinados para obter a separabilidade p de todos os x_i que estão sendo avaliados. A separabilidade média de cada solução S_j é então calculada a partir dessas medições.

Em uma solução composta principalmente de anomalias corretamente detectadas, espera-se que a separabilidade média seja alta. Por outro lado, para uma solução que contém muitos falsos positivos, a separabilidade média deve ser baixa. Portanto, o resultado do IREOS é um índice que classifica as soluções a partir daquela com a maior separabilidade média para a que tem a menor.

O IREOS também fornece um parâmetro de controle opcional, m_{cl}, que permite aos usuários definir um tamanho máximo de aglomerado. Aglomerados são subconjuntos de objetos que estão numa mesma região do espaço de dados, relativamente mais próximos uns dos outros do que de outros objetos, mas num número muito pequeno a ponto de serem considerados um grupo. Eles podem existir por diferentes motivos, como resultado de anomalias cujas instâncias são relativamente raras, mas tendem a ser um pouco semelhantes entre si. Para investigações adicionais, um analista, por exemplo, pode querer identificar esses objetos como possíveis anomalias. Com o parâmetro m_{cl}, os usuários podem expressar o que consideram como "muito pequeno" para ser interpretado como um grupo. É opcional porque, ao definir $m_{cl} = 1$, o método é reduzido ao caso em que aglomerados não são modelados. O pseudocódigo do IREOS é apresentado no Algoritmo 1.

No entanto, a necessidade de treinar $c \times y \times m$ classificadores de margem máxima com complexidade $O(n^3)$ ao analisar c soluções S_j, com $|S| = m$, torna o IREOS computacionalmente inviável para conjuntos de dados grandes. À medida que o volume de dados aumenta, também aumenta o número de anomalias presentes nos dados (YU et al., 2017), resultando em um aumento no número de classificadores necessários para calcular o índice final. Uma vez que esses classificadores usam todos os n objetos do conjunto de dados durante a etapa de treinamento, quanto maior o conjunto de dados utilizado, mais dados são usados para treinar os modelos de classificação com complexidade de $O(n^3)$, aumentando o consumo de memória e processamento.

Algoritmo 1 IREOS

```
1: procedure IREOS(X, C, m_cl)
2:     γ_max = valor de γ necessário para separar todo
3:     objeto rotulado como anomalia por uma S ∈ C dos demais objetos
4:     setOfGammas = [0, γ_max] discretizado em y valores
5:     for (S ∈ C) do
6:         for (γ ∈ setOfGammas) do
7:             for (x_j ∈ S) do
8:                 prob[x_j] = Classificador(X, x_j, S, m_cl, γ)
9:             end for
10:            MediaProb[γ] = Media(prob)
11:        end for
12:        IREOS[S] = AUC(avgProb, setOfGammas)
13:    end for
14: end procedure
```

Problemas de otimização de parâmetros e métodos de conjunto de modelos (*ensembles*), assim como o `IREOS`, também exigem o treinamento de muitos modelos. Trabalhos buscando a melhoria da eficiência de métodos nessas áreas aplicaram técnicas de computação paralela em massiva para, por exemplo, dividir o treinamento dos diferentes modelos de classificação necessários em várias máquinas em paralelo, à medida que a demanda por poder computacional aumenta. Esse conceito é conhecido como escalabilidade horizontal (ALI, 2019).

2.3 Computação Paralela Massiva

A Computação Paralela Massiva (do inglês *Massive Parallel Computing — MPC*) se tornou um assunto essencial na ciência da computação e tem se mostrado crítica quando se pesquisa soluções de alto desempenho. Em alguns campos, há uma necessidade constante de resolver problemas desafiadores em um tempo razoável, como o estudo de formação de galáxias, dinâmica molecular e mudanças climáticas (NAVARRO; HITSCHFELD-KAHLER; MATEU, 2014b). Para tornar a resolução desses problemas mais rápida ou até mesmo computacionalmente viável, arcabouços de *MPC* buscam resolver três grandes desafios: (1) eficiência, (2) escalabilidade e (3) confiabilidade (CHEN et al., 2017).

Escalabilidade refere-se à capacidade de um sistema tolerar um aumento na demanda de processamento de dados (ALI, 2019). Diferentes técnicas de escalonamento para processar grandes conjuntos de dados podem ser agrupadas em duas categorias: (I) escalonamento vertical, que envolve a instalação de processadores adicionais e *hardware* e memória mais rápidos em uma única máquina; e (II) escalonamento horizontal, que envolve o uso de várias máquinas independentes em um *cluster* para aumentar seu poder de processamento de dados, com a carga de trabalho distribuída pelas máquinas paralela-

mente. Devido ao preço do *hardware* e limitações físicas, o escalonamento vertical só pode acontecer até certo ponto. Por outro lado, "virtualmente" não há limites para o número de máquinas que podem ser adicionadas a um *cluster*. Portanto, o escalonamento horizontal é mais adequado para problemas de computação com conjuntos de dados com grande volume. No entanto, nem todos os problemas podem ser computados paralelamente.

O paralelismo é uma propriedade em tempo de execução onde duas ou mais tarefas são executadas simultaneamente. Essa propriedade depende do problema que está sendo computado. Alguns problemas não têm uma solução paralela, como problemas onde uma iteração depende do valor da anterior. Por outro lado, alguns problemas podem ser naturalmente divididos em muitos subproblemas independentes, por exemplo, a multiplicação de matrizes pode ser dividida em várias operações independentes de multiplicação e adição (NAVARRO; HITSCHFELD-KAHLER; MATEU, 2014b). Utilizando escalonamento horizontal em problemas formulados para que subproblemas sejam executados em paralelo, é possível distribuir o cálculo dos diferentes subproblemas por várias máquinas e executá-los simultaneamente.

Na otimização de parâmetros, os principais trabalhos com foco em escalabilidade horizontal são as técnicas paralelas e distribuídas de busca em grade (VERBRAEKEN et al., 2020a). Essas técnicas utilizam um conjunto pré-definido de valores para cada parâmetro e buscam determinar qual combinação resulta no melhor modelo (EITRICH; LANG, 2005). Para esse fim, muitos modelos são treinados numa variedade de combinações possíveis dos valores dos parâmetros. Segundo Celis e Musicant (2002), ao utilizar arcabouços *MPC*, como o Map-Reduce (DEAN; GHEMAWAT, 2004), os métodos de otimização de parâmetros tornaram-se viáveis para conjuntos com grandes volumes de dados.

Métodos de *ensemble* combinam vários modelos na busca de melhores resultados (MARQUES et al., 2015; HAN; KAMBER; PEI, 2012; VERBRAEKEN et al., 2020a). Técnicas de *MPC* aplicadas a essa área também utilizam escalabilidade horizontal, buscando o treinamento simultâneo desses modelos em ambientes distribuídos (OPITZ; MACLIN, 1999). Exemplos são vistos em Graf et al. (2004) e Collobert, Bengio e Bengio (2002).

A escalabilidade proporcionada pelas técnicas de *MPC* também pode ser observada em diversas outras áreas de aprendizado de máquina (ALHAM et al., 2013; MEYER; BISCHL; WEIHS, 2014; CARUANA; LI; LIU, 2013; XU et al., 2014; LIU et al., 2016; DO; POULET, 2015) que abordam o problema de limitação dos recursos de uma máquina, semelhantemente ao que ocorre com o `IREOS`.

No entanto, escalar horizontalmente um algoritmo envolve desafios em relação a como os dados devem ser armazenados e distribuídos pelas várias máquinas que irão realizar o treinamento dos modelos. Em primeiro lugar, a distribuição de um banco de dados requer a determinação da fragmentação e alocação dos dados (CERI; PERNICI; WIEDERHOLD, 1987). A fragmentação é a subdivisão de um conjunto de dados em vários

pedaços, chamados de fragmentos. A alocação é o mapeamento de cada fragmento para uma ou mais máquinas do *cluster* (NASHAT; AMER, 2018). Também existem problemas e limitações em relação à execução de consultas por meio da rede e à sincronização de acesso aos dados distribuídos, que devem ser realizados de maneira mais eficiente e garantir a integridade dos dados. Devido à importância da distribuição de banco de dados para o desenvolvimento de algoritmos de aprendizado de máquina distribuídos (VERBRAEKEN et al., 2020b), uma infinidade de técnicas de *design* de banco de dados distribuídos para abordar essas questões foi desenvolvida (CERI; PERNICI; WIEDERHOLD, 1987; NASHAT; AMER, 2018; HONG; SUN; CHEN, 2020; TARUN; BATTH; KAUR, 2019; FUAAD et al., 2018), e implementada em arcabouços de *MPC* comuns, como o Apache Hadoop (SHVACHKO et al., 2010) e Apache Spark (ZAHARIA et al., 2010), por meio de Conjuntos de Dados Distribuídos Resilientes (do inglês *Resilient Distributed Dataset — RDD*) (ZAHARIA et al., 2012).

A escalabilidade horizontal também requer que o problema siga abstrações estruturais específicas para permitir a paralelização e o processamento dos subproblemas de maneira distribuída. Outra questão essencial e não trivial é como combinar os resultados dos métodos distribuídos e paralelos, o que, para métodos como o `IREOS`, é essencial para obter o resultado.

2.4 Análise de Vizinhança Aproximada

Outro ponto a ser considerado para a escalabilidade do `IREOS` está relacionado ao volume de dados utilizados pelos classificadores durante a etapa de treinamento. A princípio, todo o conjunto de dados é utilizado para treinar cada classificador necessário para avaliar a separabilidade p de um objeto x_i. No entanto, a barreira de decisão pode ser aproximada com base nos vizinhos mais próximos do objeto, uma vez que classificadores de margem máxima buscam a barreira de decisão cuja margem possui a maior distância entre o objeto sendo analisado e os demais do conjunto de dados, a qual é também a maior margem entre o objeto analisado e seus vizinhos mais próximos (FRANC; HLAVÁČ, 2003). No `IREOS`, os objetos que não são os vizinhos mais próximos têm pouco ou nenhum impacto na avaliação final. Assim, selecionar um subconjunto apenas com os pontos de dados mais relevantes para o treinamento de um classificador diminuiria o consumo de memória e processamento.

Uma maneira de encontrar os subconjuntos que possuem cada objeto analisado e seus vizinhos mais próximos consiste em calcular um grafo com os K-vizinhos mais próximos (do inglês *K-Nearest Neighbours — K-NNG*), usado em muitos algoritmos de mineração de dados e aprendizado de máquina que requerem análise de vizinhança. No entanto, a construção por força bruta de um *K-NNG* tem uma complexidade assintótica

de $O(n^2)$, o que pode ser um problema para conjuntos de dados grandes (BRATIĆ et al., 2018).

Muitos algoritmos aproximados desenvolvidos para encontrar um *K-NNG* alcançam uma melhor eficiência computacional em troca de uma pequena parte da precisão do resultado. Por exemplo, o algoritmo NN-Descent proposto por Dong, Moses e Li (2011) demonstrou complexidade assintótica de $O(n^{1.14})$ através de diversos testes, o que é uma melhoria significativa em relação a métodos de força bruta (BRATIĆ et al., 2018). Além disso, testes realizados por Dong, Moses e Li (2011) demonstram que o NN-Descent alcança cerca de 90% da precisão do resultado exato. Dois outros exemplos de métodos de construção de *K-NNG* aproximados são o ScaNN (GUO et al., 2020) e o Annoy (BERNHARDSSON, 2015), este ultimo criado pelo serviço de streaming de música *Spotify*.

O NN-descent (*Nearest Neighbours Descent*) se baseia na construção de um grafo esparso para representar a estrutura de vizinhança dos dados. A técnica começa selecionando aleatoriamente um conjunto de objetos para construir um subgrafo inicial, sendo então expandido iterativamente adicionando novos objetos com base em sua similaridade com os objetos existentes. A construção do subgrafo é feita por meio de um processo chamado de "busca descendente" (*descendent search*), em que novos objetos são adicionados ao grafo se estiverem próximos o suficiente de um objeto já presente no grafo. Esse processo é repetido várias vezes, e o resultado é um grafo esparso que aproxima a estrutura de vizinhança dos dados. A técnica pode ser usada para indexação de dados em alta dimensão e recuperação de informação baseada em similaridade.

O ScaNN (*Scalable Nearest Neighbors*) utiliza uma combinação de aprendizado profundo e estruturas de dados otimizadas para gerar representações compactas dos dados de entrada, permitindo que as pesquisas sejam realizadas em espaços reduzidos. Ele também emprega uma estratégia de particionamento para dividir o conjunto de dados em partes menores, reduzindo a complexidade da busca de vizinhos mais próximos. O ScaNN é utilizado com sucesso em aplicações como recomendação de produtos, processamento de linguagem natural e análise de imagens (GUO et al., 2020).

O Annoy (*Approximate Nearest Neighbors Oh Yeah*) utiliza uma estrutura de árvore de busca binária para dividir recursivamente o conjunto de dados em partições menores, de modo que as consultas possam ser realizadas apenas em uma fração do conjunto de dados completo, reduzindo assim o custo computacional. Também é utilizada uma abordagem de projeção aleatória para mapear os pontos de alta dimensão em um espaço de dimensão menor, onde a distância euclidiana entre dois pontos é aproximada. Este mapeamento é realizado por meio da escolha de vetores aleatórios que dividem o espaço de alta dimensão em partições, resultando em uma representação de baixa dimensão que preserva as relações de distância dos pontos no espaço de alta dimensão.

O NN-descent, o ScaNN e o Annoy são ferramentas eficientes para a realização

de pesquisas de vizinhos mais próximos em grandes conjuntos de dados, permitindo que essas pesquisas sejam realizadas em tempo razoável mesmo em conjuntos de dados com milhões de objetos. Ainda, essas técnicas são especialmente úteis para conjuntos de dados distribuídos, nos quais a execução da análise de vizinhança em um único sistema pode ser limitada pelo tamanho da memória disponível ou pelo tempo de processamento. Com o suporte de um arcabouço de *MPC* para distribuir e gerenciar conjuntos de dados por múltiplas máquinas, como o Hadoop (WHITE, 2015) ou o Spark (KARAU et al., 2015) (mas não limitado a eles), qualquer um desses métodos pode obter o *K-NNG* aproximado do conjunto de dados de maneira distribuída, paralela e com segurança contra erros. Para cenários práticos, tais métodos atingem complexidade assintótica linear em relação ao tamanho do conjunto de dados, possibilitando a criação de múltiplos subconjuntos contendo o objeto a ser analisado e seus vizinhos mais próximos.

3 O Método Proposto: ESIREOS

Neste trabalho, propomos o ESIREOS (sigla, em inglês, de Avaliação Interna, Relativa, Eficiente e Escalável de Soluções de Anomalias), a versão aproximada e escalável do IREOS, a fim de superar as limitações computacionais que tornam o IREOS inviável para grandes conjuntos de dados, a saber: (I) a necessidade de treinar em tempo viável um conjunto de classificadores proporcional ao número de possíveis anomalias no conjunto de dados e ao número de soluções de detecção de anomalias candidatas que estão sendo analisadas, resultando em um custo computacional assintótico de $O(n^3)$; e (II) o uso de todos os n objetos do conjunto de dados durante a etapa de treinamento dos classificadores, embora, em geral, os vizinhos mais próximos da observação analisada sejam suficientes para avaliar sua separabilidade p e criar, necessária para a criação do índice IREOS.

Para abordar (I), propomos o uso de técnicas de *MPC* para escalar horizontalmente o IREOS e distribuir e paralelizar o armazenamento de dados e o esforço computacional para treinar os classificadores necessários. O ESIREOS pode trabalhar com qualquer sistema de gerenciamento de banco de dados distribuído, uma vez que não está vinculado a nenhuma tecnologia específica. Por exemplo, neste trabalho, implementamos o ESIREOS com bibliotecas Python e *RDDs* do Apache Spark (KARAU et al., 2015) para gerenciar e distribuir os dados, uma vez que são amplamente utilizados em aprendizado de máquina.

O IREOS é altamente paralelizável e pode ser implementado facilmente em ambientes distribuídos utilizando arcabouços de computação paralela, semelhantes ao MapReduce (MARQUES et al., 2015) ou outros arcabouços de *MPC*. O MapReduce usa funções de mapeamento para processar pares chave/valor, gerando novos pares chave/valor intermediários que serão combinados por funções de redução associadas a cada chave (DEAN; GHEMAWAT, 2004). No presente trabalho, utilizamos o MapReduce para processar cada objeto de cada solução de detecção de anomalias candidata simultaneamente. Primeiro, cada solução do conjunto é mapeada para recuperar as observações rotuladas como *outliers*. Em seguida, para cada observação, remapeamos com diferentes valores de *gamma* para treinar os classificadores. Em seguida, combinamos a medida de separabilidade para cada observação e *gamma*, reduzindo-os na medida geral de separabilidade da observação, que, por sua vez, é reduzida na medida de separabilidade média para a solução. Finalmente, ordenamos e reduzimos a medida de separabilidade média de todas as soluções no índice final ESIREOS. A solução apresentada aborda o problema de combinar os resultados dos diferentes modelos de classificação distribuídos e paralelos treinados, adicionando escalabilidade horizontal ao IREOS.

Neste trabalho, para abordar o problema (II), avaliamos métodos aproximados

de construção de *K-NNG* para gerar subconjuntos do conjunto de dados contendo as observações analisadas e seus k vizinhos mais próximos aproximados. Ao usar os subconjuntos gerados pelo *K-NNG* para treinar classificadores e avaliar as separabilidades, a complexidade de treinar cada um dos $c \times y \times m$ classificadores é reduzida de $O(n^3)$ para $O(k^3)$, com $k << n$ para grandes conjuntos de dados. Além disso, neste trabalho, os classificadores *SVM* substituirão os classificadores *KLR* usados no trabalho original, reduzindo ainda mais a complexidade do treinamento para $O(k^2)$ para cada classificador, resultando em uma complexidade total de $O(n + k^2)$.

Tradicionalmente, encontrar a vizinhança exata de observações tem uma complexidade computacional de $O(n^2)$, sendo inviável para grandes conjuntos de dados. Por outro lado, os métodos aproximados apresentam complexidade assintótica linear em troca de uma pequena redução na precisão em comparação com um *K-NNG* construído com um método de construção de *K-NNG* exato. No entanto, espera-se que essa pequena perda de precisão não afete significativamente a qualidade dos resultados do `IREOS`, permitindo que ele lide com grandes conjuntos de dados.

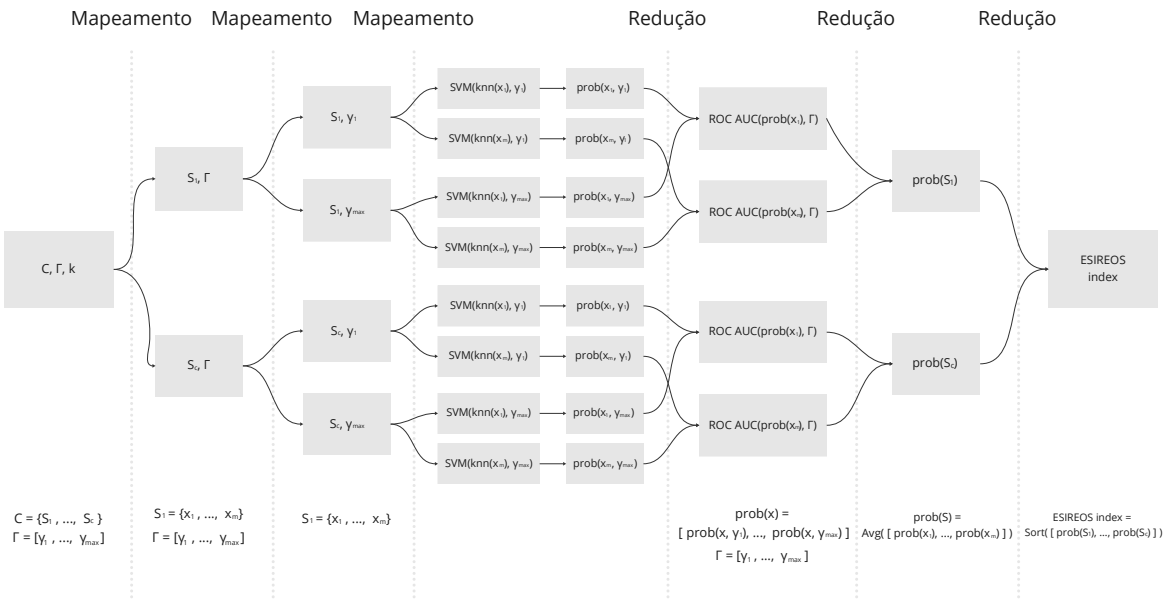
Para garantir a qualidade do índice `ESIREOS` para as soluções candidatas e reduzir qualquer possível viés devido à escolha do algoritmo *K-NNG* aproximado, foram analisados três métodos neste trabalho: NN-Descent, Annoy e ScanNN. Os três métodos foram escolhidos por sua adequação para um ambiente de computação distribuída e pela pequena necessidade de replicação de dados entre as máquinas (DONG; MOSES; LI, 2011; GUO et al., 2020). No entanto, o `ESIREOS` não está ligado a esses métodos específicos, já que qualquer *K-NNG* aproximado de qualidade razoável deve funcionar.

Vale ressaltar que, ao avaliar a separabilidade de um objeto usando a aproximação de vizinhança, o resultado sempre será igual ou maior do que a verdadeira separabilidade p do objeto. O resultado reflete que, ao perder alguns vizinhos mais próximos, a separabilidade será aumentada por uma constante $\epsilon \geq 0$. O valor de ϵ é esperado ser relativamente pequeno em áreas densas (ou seja, *inliers*), já que há muitos vizinhos com distâncias similares. No entanto, para um objeto em uma área esparsa (ou seja, anomalias), é esperado que ϵ seja consideravelmente maior do que em regiões densas. Para a avaliação de anomalias, tal característica é desejável, por aumentar relativamente mais a medida de separabilidade de anomalias verdadeiras do que de objetos normais, para os quais, em contraste, é esperado apenas um ligeiro aumento.

3.1 Formalização e Algoritmo

Um diagrama de alto nível é apresentado na Figura 1 para ilustrar a execução do `ESIREOS` e todas as operações de mapeamento e redução. O Algoritmo 2 detalha o `ESIREOS` em pseudocódigo.

Figura 1 – Diagrama representando as funções de mapeamento e redução do ESIREOS



Fonte: Produzido pelos autores

Para um conjunto de dados multidimensional $X = x_1, ..., x_n$ distribuído por um *cluster* de máquinas e um conjunto C de soluções S, onde $|C| = c$ e $|S| = m$, um grafo aproximado com os k vizinhos mais próximos de cada objeto em X é criado por meio de um método *K-NNG* aproximado adequado para um ambiente de computação distribuída, conforme o passo 1 no Algoritmo 2.

O `ESIREOS` recebe como entrada X, C e k, além de um parâmetro opcional m_{cl}, que o usuário pode usar para definir um tamanho máximo de aglomerado (se o usuário não definir nenhum valor para m_{cl}, o `ESIREOS` assume que $m_{cl} = 1$, onde nenhum aglomerado é permitido).

Um valor de γ é usado pelo `ESIREOS` para controlar a capacidade de discriminação dos classificadores baseados em *kernel*, e γ_{max} é a ordem polinomial em que todos os objetos rotulados como anomalias por uma solução candidata podem ser individualmente discriminados de todas as outras observações no conjunto de dados. O valor de γ_{max} é estimado da mesma forma que no `IREOS` original: conforme o passo 2 do Algoritmo 2, o `ESIREOS` adiciona os objetos rotulados como anomalias em uma lista L e inicia γ_{max} com base na dimensionalidade de X. Em seguida, para cada objeto em L, ele verifica se o objeto é separável usando o γ máximo atual. Se sim, a anomalia é removida de L, caso contrário, γ_{max} é aumentado.

No passo 3, o algoritmo calcula um conjunto Γ de valores de γ discretizando γ_{max} em y valores diferentes em uma escala logarítmica, a partir de zero. O valor de y também

Algoritmo 2 ESIREOS

Entrada: $\{X, C, m_{cl}, k, y\}$
Saída: {Índice ESIREOS}

1. Dado um conjunto de dados distribuído X, encontre os k vizinhos mais próximos de cada objeto em X por meio de um método distribuído e eficiente de *K-NNG*.
2. Compute γ_{max}, a ordem polinomial de uma função de *kernel* que pode segregar cada objeto rotulado como anomalia dos outros objetos em X.
   a) Adicione os objetos rotulados como anomalias em uma lista L
   b) Inicie γ_{max} com base na dimensionalidade d de X: $\gamma_{max} = 1 \div (d \times 1000)$
   c) Para cada objeto em L, verifique se ele é separável usando o γ_{max} atual; se sim, o objeto rotulado anomalia é removido de L; caso contrário, γ_{max} é aumentado.
3. Compute um conjunto $\Gamma = [\gamma_1 = 0, ..., \gamma_y = \gamma_{max}]$ discretizando γ_{max} em y valores diferentes em uma escala logarítmica, a partir do zero.
4. Mapeie cada solução $S_j \in C$
   a) Para cada solução S_j mapeada, mapeie cada objeto $x_i \in S_j$
      i. Para cada objeto $x_i \in S_j$ mapeado, mapeie cada $\gamma_z \in \Gamma$
         A. Construa um subconjunto Set_{x_i} com x_i e seus k vizinhos mais próximos
         B. Treine um classificador de margem máxima com função de *kernel* para obter a separabilidade p_{x_i} de x_i, usando γ_z e m_{cl} como parâmetros
      ii. Reduza todos os valores de separabilidade p_{x_i} obtidos para x_i com o intervalo de valores γ em Γ para calcular a *Area Sob a Curva ROC* (AUC ROC) para x_i, obtendo a separabilidade média de x_i
   b) Reduza a separabilidade média de todos os objetos $x_i \in S_j$ para calcular a separabilidade média de S_j.
5. Reduza a separabilidade média de todas as $S_j \in C$ ordenado as soluções por sua separabilidade, da maior para a menor, obtendo o índice ESIREOS.

é um parâmetro opcional. Se o usuário não determinar nenhum valor para y, o ESIREOS assume $y = 10$.

Usando um arcabouço MapReduce, o ESIREOS mapeia cada solução $S_j \in C$ (passo 4). Para cada solução S_j mapeada, são mapeados cada objeto $x_i \in S_j$ (passo 4.a) e, para cada objeto $x_i \in S_j$ mapeado, são mapeados cada $\gamma_z \in \Gamma$ (4.a.i). Para cada γ_z, o algoritmo constrói um subconjunto Set_{x_i} com x_i e seus k vizinhos mais próximos (4.a.i.A) e treina um classificador de margem máxima com uma função de *kernel* para obter a separabilidade p_{x_i} de x_i, usando γ_z e m_{cl} como parâmetros (4.a.i.B). O ESIREOS reduz todos os valores de separabilidade p_{x_i} obtidos para x_i com o intervalo de valores de γ em Γ para calcular o valor da *Área Sob a Curva ROC* (*AUC ROC*) para x_i, resultando na separabilidade média de x_i (4.a.ii). É então reduzida a separabilidade de todos os objetos $x_i \in S_j$ para calcular

a separabilidade média de S_j (4.b).

Finalmente, no passo 5, o algoritmo reduz a separabilidade média de todos os $S_j \in C$ ordenando as soluções por sua separabilidade, da mais alta para a mais baixa, obtendo o índice `ESIREOS`.

4 Avaliação

Nesta seção, apresentamos uma avaliação teórica e prática (experimental) do IREOS e do ESIREOS. A avaliação teórica é realizada por meio de análises computacionais assintóticas dos algoritmos relacionados.

Os experimentos foram divididos em dois tipos: o primeiro tipo, chamado de teste de eficácia, consiste em comparar os resultados obtidos pelo IREOS e ESIREOS em um conjunto de conjuntos de dados reais amplamente utilizados para a avaliação de detecção de anomalias, a fim de avaliar a qualidade do ESIREOS com o uso de *K-NNG* aproximados para a criação de subconjuntos com cópias de cada objeto e seus k vizinhos mais próximos, e a utilização desses subconjuntos para treinar modelos *SVM*. O segundo tipo, chamado teste de eficiência, consiste em comparar a eficiência computacional do IREOS e do ESIREOS em um conjunto de conjuntos de dados sintéticos com grande volume de dados, a fim de verificar se o ESIREOS pode dimensionar e processar conjuntos de dados muito grandes para o IREOS.

4.1 Complexidade Computacional

No caso mais geral, a complexidade computacional assintótica do IREOS é limitada por $O(c \times y \times m \times n^3)$, porque é necessário treinar $c \times y \times m$ classificadores *KLR* com complexidade $O(n^3)$ para calcular o índice final do IREOS.

Para o ESIREOS, é possível construir o *K-NNG* aproximado com custo de complexidade $O(n)$. Em seguida, seriam treinados $c \times y \times m$ classificadores *SVM* usando apenas $x_i \in S_j$ e seus k vizinhos mais próximos. Na maioria dos cenários, assume-se que $k << n$, especialmente para conjuntos de dados grandes, resultando em uma complexidade de $O(k^2)$ por classificador. Portanto, ao executar em um único núcleo, a complexidade final do ESIREOS é $O(n + c \times y \times m \times k^2)$. Como o ESIREOS permite treinamento distribuído e paralelo por várias máquinas, no caso em que temos $c \times y \times m$ nós de trabalho para processamento paralelo, a complexidade do ESIREOS pode ser expressa como $O(n + k^2)$.

O ESIREOS é altamente escalável e permite o processamento de conjuntos de dados muito grandes, tendo uma complexidade quase linear em relação a n, se $k << n$, quando executado em um *cluster* com a disponibilidade necessária de nós de trabalho.

4.2 Teste de Eficácia

4.2.1 Conjuntos de Dados

Para avaliar a precisão do índice `ESIREOS` em comparação com o valor original do `IREOS`, utilizamos dez conjuntos de dados disponíveis no material suplementar de Campos et al. (2016) e no repositório ODDS [1]. No entanto, devido aos recursos computacionais necessários para o `IREOS`, nossos testes de eficácia usaram apenas uma amostra dos conjuntos de dados originais. O número original de objetos e anomalias de cada conjunto de dados são apresentados na Tabela 1, juntamente com o tamanho das amostras destes conjuntos.

Tabela 1 – Conjuntos de dados utilizados para o teste de eficácia

Conjunto	# Objetos	# Anomalias	# Objetos Amostrados	# Anomalias Amostradas
PenDigits	9869	20	990	10
WaveForm	3443	100	2990	10
WBC	454	10	454	10
Pima	768	268	490	10
Vowels	1456	50	990	10
Cardio	1831	176	990	10
Thyroid	3772	93	990	10
BreastW	683	239	490	10
Letter	1600	100	990	10
Musk	3062	97	990	10

Fonte: Produzido pelos autores.

4.2.2 Metodologia

Para cada um dos conjuntos de dados selecionados, primeiro produzimos um conjunto C de soluções de detecção de anomalias candidatas S_j para criar os índices `IREOS` e `ESIREOS` durante os experimentos e avaliar os resultados. Este conjunto de soluções de detecção de anomalias foi produzido conforme a metodologia adotada por Marques et al. (2015) e descrita aqui: começa-se criando uma solução candidata sobre os dados da amostra onde todas as anomalias conhecidas, conforme os rótulos fornecidos, estão presentes. Em seguida, iterativamente, novas soluções são produzidas substituindo uma das anomalias por uma observação normal aleatória, até que uma solução sem anomalias conhecidas seja gerada. Como as amostras de todos os conjuntos de dados têm dez anomalias, onze soluções candidatas são geradas para cada conjunto de dados. Cada solução é nomeada

[1] Repositório ODDS. Disponível em: <http://odds.cs.stonybrook.edu>. Acesso em 06/06/2022.

conforme o número de anomalias conhecidas presentes: a 'solução 10' tem todas as dez anomalias conhecidas da amostra, enquanto 'solução 0' não tem nenhuma, por terem sido substituídas por objetos normais.

Para cada conjunto de dados, usando o conjunto C de soluções candidatas criadas, calculamos os índices `ESIREOS` com k, o número de vizinhos mais próximos, de 5%, 10% e 20% do tamanho total da amostra. Em seguida, comparamos esses índices com um índice gerado pelo `IREOS` original, que usou a amostra inteira do conjunto de dados para calcular a separabilidade dos objetos. Para todos os diferentes valores de k, calculamos o erro no valor médio de separabilidade de cada solução candidata ao comparar o índice `ESIREOS` e o índice `IREOS` e analisamos como isso afetou (ou não) a posição das soluções no índice `ESIREOS`. Dessa forma, pudemos avaliar a qualidade das soluções geradas pelo `ESIREOS` com diferentes tamanhos de subconjuntos contendo o objeto analisado e seus vizinhos aproximados.

Esse processo foi repetido usando os três métodos *K-NNG* aproximados descritos na Seção 2.4: NN-Descent, ScaNN e Annoy. Como os métodos aproximados têm sementes probabilísticas, executamos o `ESIREOS` usando cada um dos métodos *K-NNG* e valor de k trinta vezes e depois usamos o valor médio da medida de separabilidade para avaliar cada solução candidata.

4.2.3 Resultados

As Figuras 2 a 11 mostram a diferença e o desvio padrão da média da separabilidade encontrada pelo `ESIREOS` para cada solução quando comparada com a separabilidade das soluções de acordo com um índice gerado pelo `IREOS`. Cada figura apresenta os valores por método *K-NNG* aproximado e valor de vizinhança k usado pelo `ESIREOS` para um dos conjuntos de dados mostrados na Tabela 1.

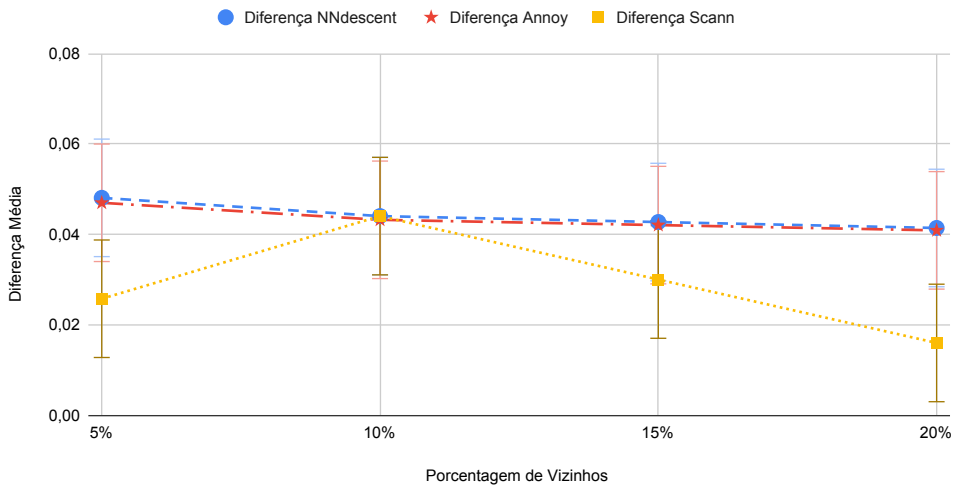
A média da diferença varia de 0,001 a 0,053 para os diferentes conjuntos de dados e métodos *K-NNG*, sendo que o menor erro é observado nos conjuntos de dados *Cardio* e *WBC* para k igual a 20% e NNdescent como o método *K-NNG* aproximado. O desvio padrão da diferença é geralmente baixo, variando de 0,0002 a 0,013, indicando que os resultados são consistentes em várias execuções. Por exemplo, o conjunto de dados *PenDigits* teve uma média de diferença variando de 0,015 a 0,048 com um desvio padrão em torno de 0,013 para todos os métodos. Da mesma forma, o conjunto de dados *Thyroid* teve uma diferença variando de 0,001 a 0,008 com um desvio padrão em torno de 0,0003 com o NNDescent, 0,0006 com o Scann e 0,001 com o Annoy.

Para avaliar melhor os resultados, é essencial notar que o `IREOS` (e o `ESIREOS`) possui um limite superior de valor 1, que reflete uma solução de correspondência perfeita e é ajustado ao acaso para assumir valores próximos de zero para soluções aleatórias.

Para todos os conjuntos de dados e métodos *K-NNG*, o `ESIREOS` produziu valores de índice muito próximos ao `IREOS` original, o que é refletido em taxas de diferença abaixo de 0,06 para todos os conjuntos de dados, com pequenos valores de desvio padrão. Uma pequena taxa de erro foi encontrada para todos os resultados com diferentes tamanhos de vizinhança (k), embora os resultados tenham mostrado que quanto maior a vizinhança, menor a diferença na maioria dos cenários. Tal comportamento é esperado, já que vizinhanças maiores tendem a ter mais informações sobre a separabilidade dos objetos. Além disso, a ordenação das soluções candidatas nos índices obtidos pelo `ESIREOS` corresponde precisamente à posição das soluções candidatas no índice `IREOS`, isto é, o uso de métodos *K-NNG* e vizinhanças aproximadas não impactou na posição das soluções no ranqueamento durante estes experimentos.

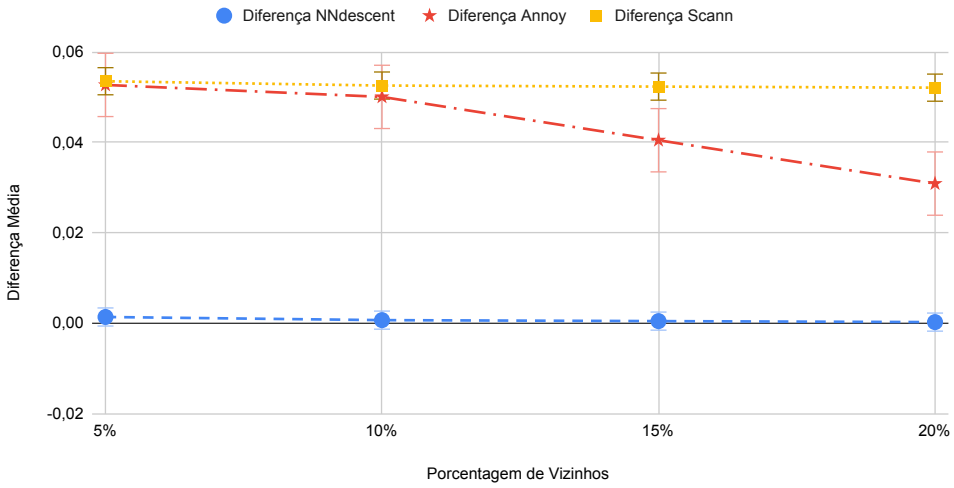
Com exceção do conjunto de dados PenDigits, os experimentos mostraram que o NN-Descent apresentou o melhor desempenho como método *K-NNG* aproximado, embora o `ESIREOS` com Annoy ou ScaNN também tenha resultado em erros médios de separabilidade baixos. Por esse motivo, escolhemos implementar o `ESIREOS` usando NN-Descent para o teste de desempenho sobre o Apache Spark apresentado na Seção 4.3.

Figura 2 – Conjunto de dados PenDigits: média e o desvio padrão da diferença de 30 execuções do ESIREOS comparado ao IREOS



Fonte: Produzido pelos autores

Figura 3 – Conjuntos de dados WaveForm: média e o desvio padrão da diferença de 30 execuções do ESIREOS comparado ao IREOS



Fonte: Produzido pelos autores

4.3 Teste de Eficiência

4.3.1 Conjuntos de Dados

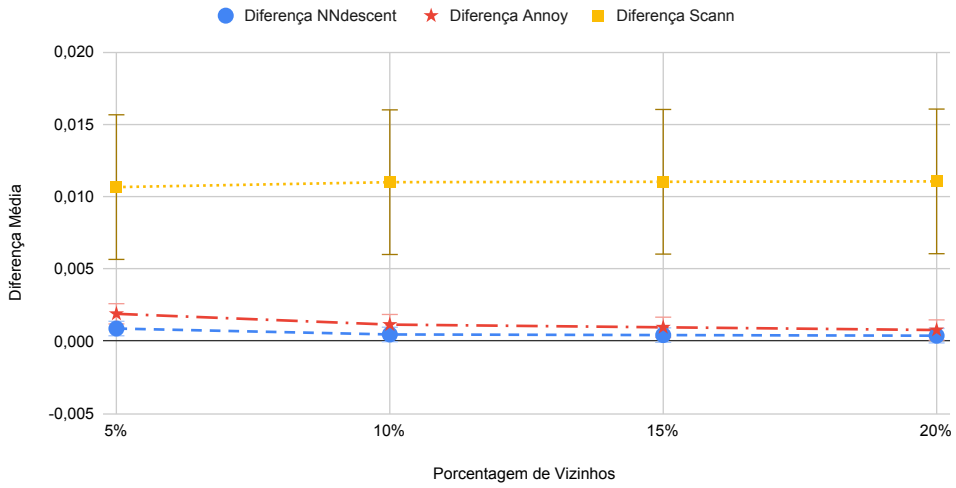
A fim de avaliar o desempenho e escalabilidade do `ESIREOS` no processamento de grandes conjuntos de dados, quatro conjuntos de dados sintéticos com um grande volume de dados foram gerados com o MDCGenPy (VáZQUEZ et al., 2019). O número de objetos, dimensões e *clusters* de cada conjunto de dados é apresentado na Tabela 2. Os tamanhos dos *clusters* foram escolhidos uniformemente a partir do intervalo [1000, 25000] para os conjuntos de dados 1 e 2, e [5000, 50000] para os conjuntos de dados 3 e 4. Para cada *cluster*, os pontos foram gerados pelo MDCGenPy seguindo um modelo Gaussiano com os atributos, como média e desvio padrão, mantendo o valor padrão do MDCGenPy e semente 0.

Tabela 2 – Conjuntos de dados sintéticos gerados para os testes de eficiência

Conjunto	# Objetos	# Dimensões	# Grupos	# Anomalias
1	100.000	10	20	2.500
2	250.000	20	40	6.250
3	500.000	30	60	12.500
4	1.000.000	40	80	25.000

Fonte: Produzido pelos autores.

Figura 4 – Conjuntos de dados WBC: média e o desvio padrão da diferença de 30 execuções do ESIREOS comparado ao IREOS



Fonte: Produzido pelos autores

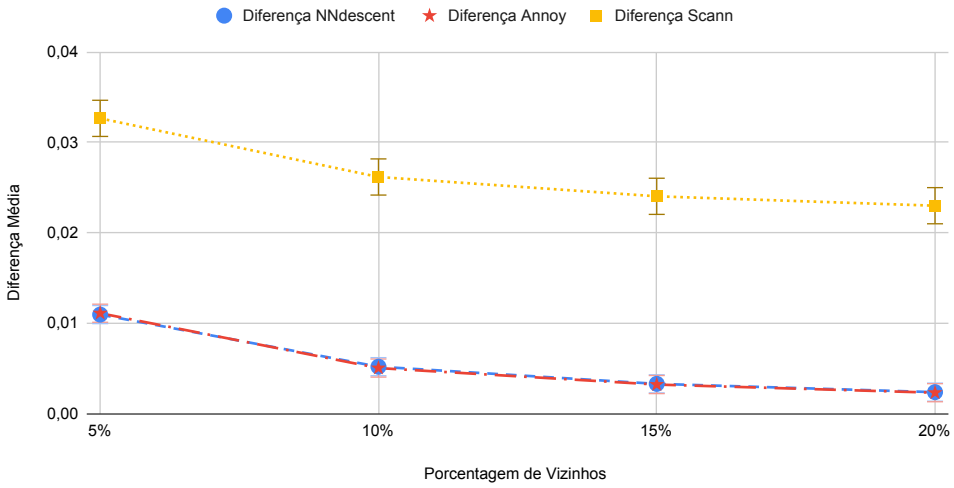
Os conjuntos de dados artificiais foram construídos seguindo o processo apresentado em Zimek et al. (2013) e descrito aqui: seguindo uma distribuição x^2 com d graus de liberdade, a distância de Mahalanobis entre a média de um *cluster* e cada ponto desse *cluster* é calculada; os pontos com uma distância para o centro do *cluster* maior que o quantil de 0,975 foram rotulados como anomalias, gerando cerca de 2,5% de anomalias por conjunto de dados.

4.3.2 Metodologia

Seguindo a mesma metodologia adotada para o teste de eficácia (descrito na Seção 4.2.2), criamos um conjunto C de soluções candidatas, iterativamente, a partir de uma solução onde todos as anomalias verdadeiras são rotuladas como anomalias até a solução sem nenhuma anomalia verdadeira. Para cada um dos quatro conjuntos de dados sintéticos, executamos o `ESIREOS` com valores de tamanho de vizinhança k iguais a 5%, 10% e 20% do tamanho total do conjunto de dados.

Os experimentos foram executados no *cluster* da Universidade Federal de São Carlos (UFSCar), um serviço baseado em `OpenHPC` para a comunidade universitária. O *cluster* tem 18 máquinas com 40 *threads* de *CPU*, 320 *GB* de *RAM* (*DDR4*) e duas interfaces de rede 10G *SFP+*. Os usuários podem enviar trabalhos para serem executados em nós regulares por até sete dias. Trabalhos que não terminam no tempo estabelecido são interrompidos. Tentamos executar o `IREOS` original para os conjuntos de dados sintéticos criados para o teste de eficiência. No entanto, exceto para o conjunto de dados com 100

Figura 5 – Conjuntos de dados Pima: média e o desvio padrão da diferença de 30 execuções do ESIREOS comparado ao IREOS



Fonte: Produzido pelos autores

mil objetos, o `IREOS` foi interrompido após sete dias de execução, o que nos obrigou a truncar o tempo total desse algoritmo nos resultados apresentados aqui.

Justificado pelos bons resultados no teste de efetividade, o NN-Descent foi adotado para o `ESIREOS` nos testes de desempenho. Para permitir o escalonamento horizontal do `ESIREOS`, usamos o PySpark[2], uma interface para o Apache Spark em Python, com um nó mestre e quatro nós de trabalho com a configuração padrão do PySpark.

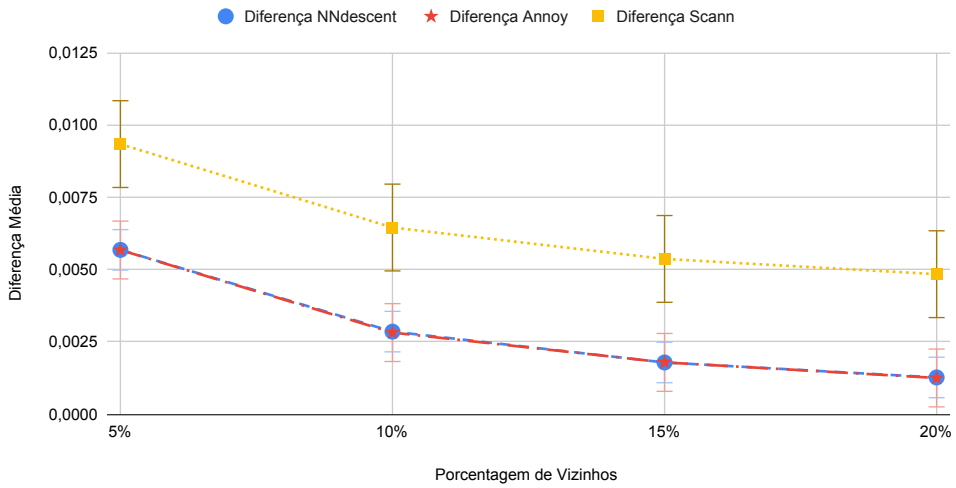
4.3.3 Resultados

A Figura 12 apresenta o tempo de processamento em relação ao número de objetos para o `ESIREOS` com NN-Descent e tamanhos de vizinhança k iguais a 5%, 10% e 20% do conjunto de dados e para o `IREOS`.

O `IREOS` gerou um índice apenas para o conjunto de dados 1, com 100 mil objetos. Para todos os outros conjuntos de dados, o `IREOS` não concluiu o processamento no período de sete dias, tamanho dos trabalhos oferecidos pelo *cluster* da UFSCar. Por outro lado, o `ESIREOS` foi consideravelmente mais rápido: para todos os valores de k (5%, 10% e 20%), a inclinação da curva de tempo de processamento (em minutos) em relação ao número de objetos (em milhares) foi abaixo da linear, com todos os experimentos concluindo em menos de vinte e duas horas. Como esperado, quanto menor o valor de k, mais rápido é para o `ESIREOS` produzir seu índice.

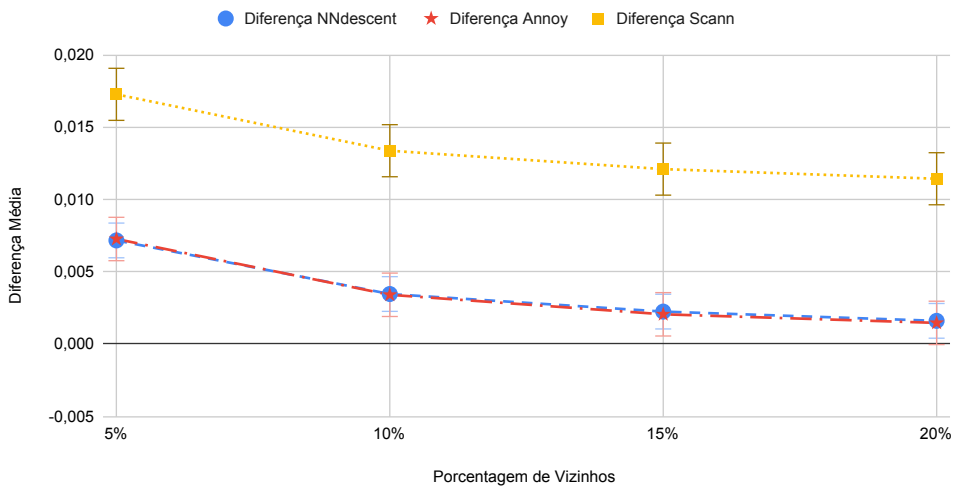
[2] Documentação do PySpark. Disponível em: <https://spark.apache.org/docs/latest/api/python/>. Acessado em 01/06/2022.

Figura 6 – Conjuntos de dados Vowels: média e o desvio padrão da diferença de 30 execuções do ESIREOS comparado ao IREOS



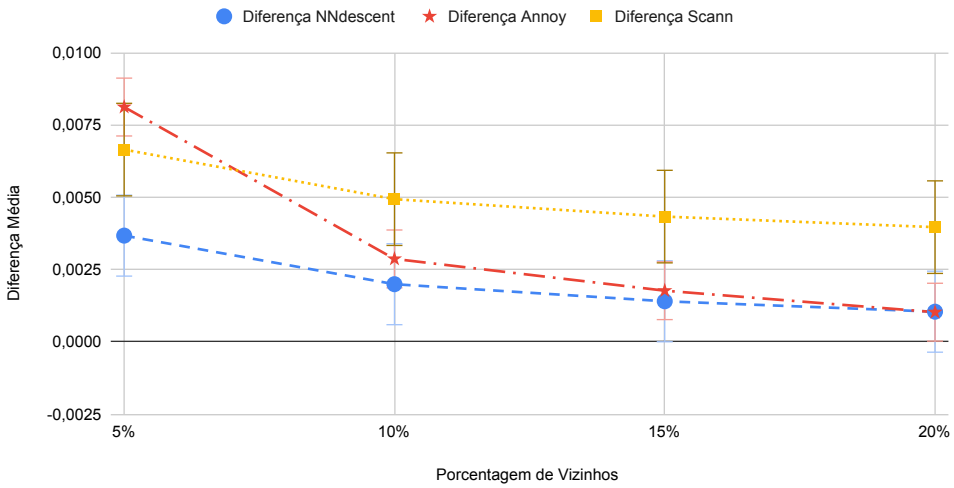
Fonte: Produzido pelos autores

Figura 7 – Conjuntos de dados Cardio: média e o desvio padrão da diferença de 30 execuções do ESIREOS comparado ao IREOS



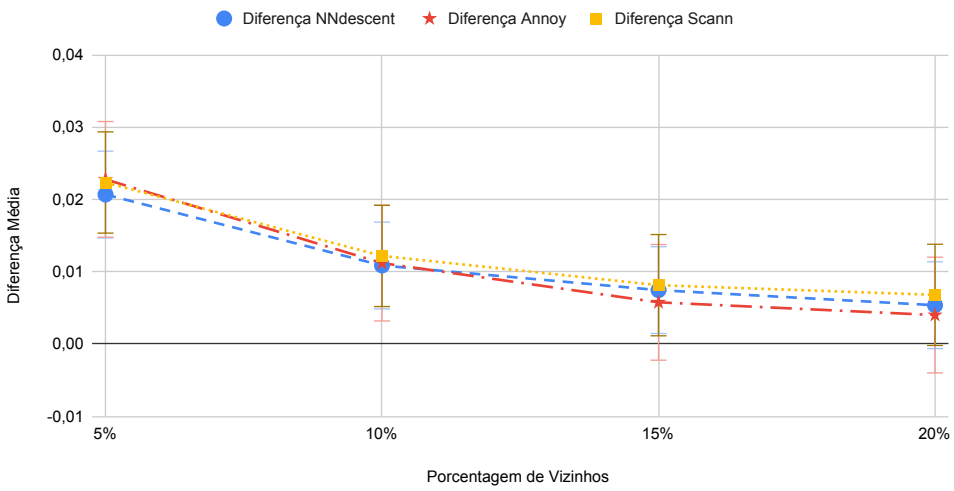
Fonte: Produzido pelos autores

Figura 8 – Conjuntos de dados Thyroid: média e o desvio padrão da diferença de 30 execuções do ESIREOS comparado ao IREOS



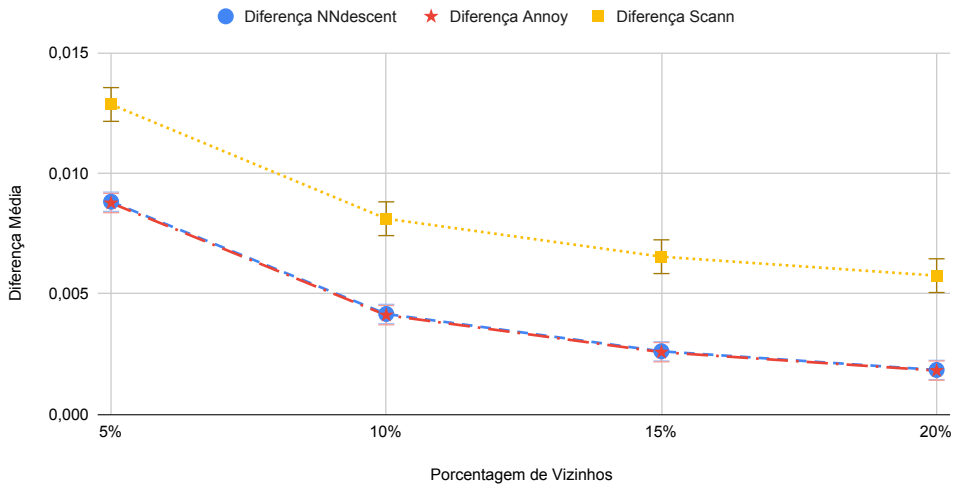
Fonte: Produzido pelos autores

Figura 9 – Conjuntos de dados Breastw: média e o desvio padrão da diferença de 30 execuções do ESIREOS comparado ao IREOS



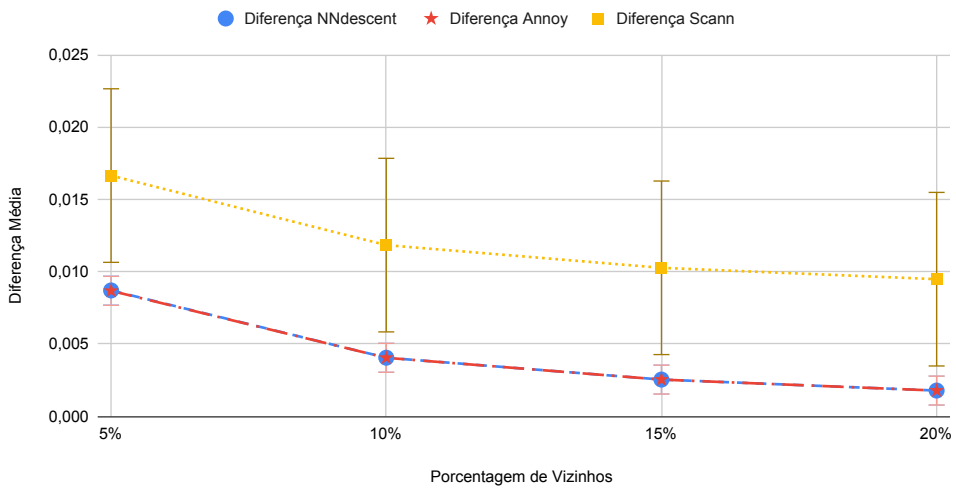
Fonte: Produzido pelos autores

Figura 10 – Conjuntos de dados Letter: média e o desvio padrão da diferença de 30 execuções do ESIREOS comparado ao IREOS



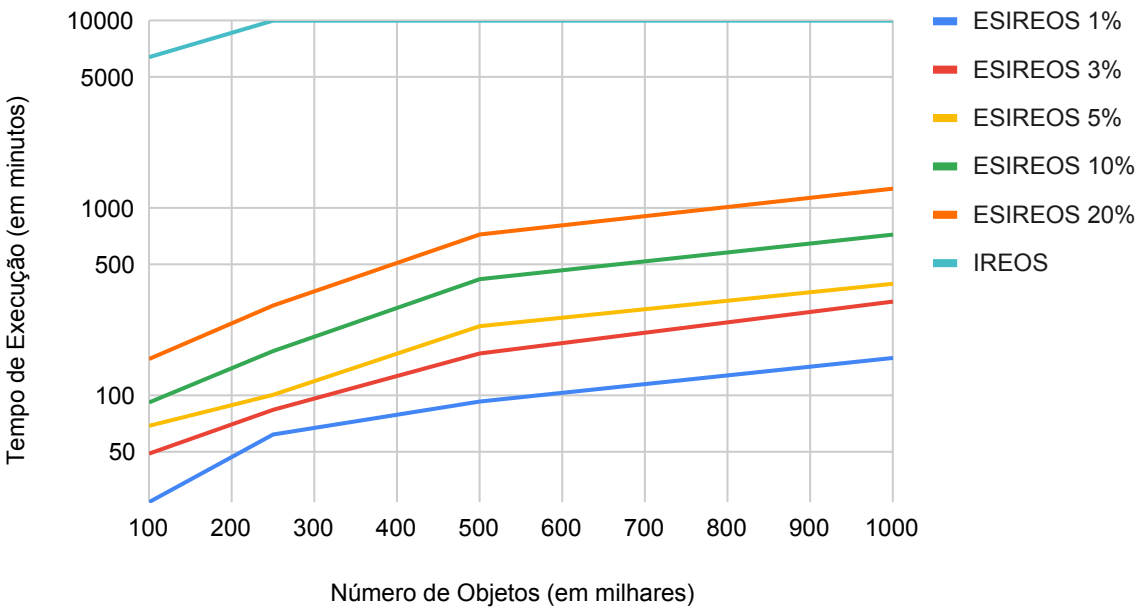
Fonte: Produzido pelos autores

Figura 11 – Conjuntos de dados Musk: média e o desvio padrão da diferença de 30 execuções do ESIREOS comparado ao IREOS



Fonte: Produzido pelos autores

Figura 12 – Tempo de execução do IREOS e do ESIREOS em relação ao tamanho do conjunto de dados



Fonte: Produzido pelos autores

Conclusão

Neste trabalho, nosso objetivo foi endereçar as limitações de desempenho e processamento do IREOS sem perda significativa de qualidade no resultado, permitindo seu uso em um número crescente de aplicações, especialmente aquelas com grande volume de dados. Os experimentos mostraram que o ESIREOS obteve sucesso em (I) adotar métodos aproximados de *K-NNG* para criar eficientemente subconjuntos do conjunto de dados de entrada, reduzindo o número de objetos para treinar os classificadores, e (II) escalonar horizontalmente o IREOS, por meio da paralelização e distribuição da indução de todos os classificadores necessários para avaliar a separação dos objetos e soluções e o do índice IREOS.

Os resultados apresentados nas Figuras 2 a 11 do teste de eficácia mostraram uma diferença inferior a 0,06 (em uma escala de zero a um) para todos os dez conjuntos de dados reais e públicos utilizados, com todas as três variações de métodos aproximados de *K-NNG* utilizados. Além disso, o erro não foi suficiente para causar quaisquer mudanças na classificação das soluções candidatas no índice ESIREOS em comparação com um índice gerado pelo IREOS, mantendo um espelho perfeito das soluções originais do índice usando um número significativamente menor de objetos para induzir os classificadores.

O fato dos índices ESIREOS serem um espelho de um índice IREOS, dado um conjunto de dados e conjunto de soluções, demonstram que o ESIREOS mantêm a qualidade do resultando enquanto aponta a melhor solução de detecção de anomalias em um tempo consideravelmente menor e de maneira escalável. Em cenários de aplicações onde a análise de *outliers* é importante, como em conjuntos de dados médicos ou financeiros, por exemplos, o ESIREOS pode ser utilizado para encontrar o melhor método e parametrização para se obter o melhor resultado. Ainda, pode ser utilizado para auxiliar na limpeza de conjuntos de dados para o treinamento de modelos de aprendizado de máquina, auxiliando na obtenção de melhores resultados, mesmo com conjuntos de dados muito grandes.

O ESIREOS se destacou ainda mais nos testes de desempenho, apresentando uma inclinação subquadrática / próxima à linear na curva de tempo de execução vs. tamanho do conjunto de dados, tornando-o escalável para grandes conjuntos de dados. Em contraste, a implementação original do IREOS não conseguiu concluir a execução em menos de sete dias para a maioria dos conjuntos de dados. Esses resultados refletem os limites superiores computacionais assintóticos do ESIREOS, que é $O(n + c \times y \times m \times k^2)$, e do IREOS, que é $O(c \times y \times m \times n^3)$.

Submissões

- The 10th IEEE International Conference On Data Science And Advanced Analytics (DSAA 2023)

Referências

AGGARWAL, C. C. *Outlier Analysis*. Springer International Publishing, 2017. Disponível em: <https://doi.org/10.1007/978-3-319-47578-3>. Citado na página 21.

ALHAM, N. K. et al. A MapReduce-based distributed SVM ensemble for scalable image classification and annotation. *Computers & Mathematics with Applications*, Elsevier BV, v. 66, n. 10, p. 1920–1934, dez. 2013. Disponível em: <https://doi.org/10.1016/j.camwa.2013.07.015>. Citado na página 29.

ALI, A. H. A survey on vertical and horizontal scaling platforms for big data analytics. *International Journal of Integrated Engineering*, v. 11, n. 6, p. 138–150, 2019. Citado na página 28.

BERNHARDSSON, E. *Spotify/Annoy at github: https://github.com/spotify/annoy*. 2015. Citado na página 31.

BRATIĆ, B. et al. Nn-descent on high-dimensional data. In: *Proceedings of the 8th International Conference on Web Intelligence, Mining and Semantics*. [S.l.: s.n.], 2018. p. 1–8. Citado na página 31.

CAMPOS, G. et al. On the evaluation of unsupervised outlier detection: measures, datasets, and an empirical study. *Data Mining and Knowledge Discovery*, v. 30, 07 2016. Citado 2 vezes nas páginas 25 e 40.

CARUANA, G.; LI, M.; LIU, Y. An ontology enhanced parallel svm for scalable spam filter training. *Neurocomputing*, p. –, 05 2013. Citado na página 29.

CELIS, S.; MUSICANT, D. R. *Weka-Parallel: Machine Learning in Parallel*. [S.l.], 2002. Citado na página 29.

CERI, S.; PERNICI, B.; WIEDERHOLD, G. Distributed database design methodologies. *Proceedings of the IEEE*, IEEE, v. 75, n. 5, p. 533–546, 1987. Citado 2 vezes nas páginas 29 e 30.

CHANDOLA, V.; BANERJEE, A.; KUMAR, V. Anomaly detection. *ACM Computing Surveys*, Association for Computing Machinery (ACM), v. 41, n. 3, p. 1–58, jul. 2009. Disponível em: <https://doi.org/10.1145/1541880.1541882>. Citado na página 21.

CHEN, D. et al. Brain big data processing with massively parallel computing technology: challenges and opportunities. *Software: Practice and Experience*, Wiley Online Library, v. 47, n. 3, p. 405–420, 2017. Citado na página 28.

COLLOBERT, R.; BENGIO, S.; BENGIO, Y. A parallel mixture of SVMs for very large scale problems. *Neural Computation*, MIT Press - Journals, v. 14, n. 5, p. 1105–1114, maio 2002. Disponível em: <https://doi.org/10.1162/089976602753633402>. Citado na página 29.

DEAN, J.; GHEMAWAT, S. Mapreduce: Simplified data processing on large clusters. In: *OSDI'04: Sixth Symposium on Operating System Design and Implementation*. San Francisco, CA: [s.n.], 2004. p. 137–150. Citado 2 vezes nas páginas 29 e 33.

DO, T.-N.; POULET, F. Random local svms for classifying large datasets. In: . [S.l.: s.n.], 2015. p. 3–15. ISBN 978-3-319-26134-8. Citado na página 29.

DONG, W.; MOSES, C.; LI, K. Efficient k-nearest neighbor graph construction for generic similarity measures. In: *Proceedings of the 20th international conference on World wide web - WWW '11*. ACM Press, 2011. Disponível em: <https://doi.org/10.1145/1963405.1963487>. Citado 3 vezes nas páginas 23, 31 e 34.

EITRICH, T.; LANG, B. Parallel tuning of support vector machine learning parameters for large and unbalanced data sets. In: *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2005. p. 253–264. Disponível em: <https://doi.org/10.1007/11560500_23>. Citado na página 29.

FRANC, V.; HLAVÁČ, V. An iterative algorithm learning the maximal margin classifier. *Pattern recognition*, Elsevier, v. 36, n. 9, p. 1985–1996, 2003. Citado na página 30.

FUAAD, H. et al. A survey on distributed database fragmentation allocation and replication algorithms. *British Journal of Applied Science and Technology*, v. 27, n. 2, p. 1–12, 2018. Citado na página 30.

FULLÉR, R. Unsupervised clustering for deep learning: A tutorial survey. *Acta Polytechnica Hungarica*, v. 15, p. 29–53, 12 2018. Citado na página 22.

GRAF, H. et al. Parallel support vector machines: The cascade svm. In: . [S.l.: s.n.], 2004. Citado na página 29.

GUO, R. et al. Accelerating large-scale inference with anisotropic vector quantization. In: *International Conference on Machine Learning*. [s.n.], 2020. Disponível em: <https://arxiv.org/abs/1908.10396>. Citado 2 vezes nas páginas 31 e 34.

HAN, J.; KAMBER, M.; PEI, J. *Data mining concepts and techniques, third edition*. Waltham, Mass.: Morgan Kaufmann Publishers, 2012. Disponível em: <http://www.amazon.de/Data-Mining-Concepts-Techniques-Management/dp/0123814790/ref=tmm_hrd_title_0?ie=UTF8&qid=1366039033&sr=1-1>. Citado 2 vezes nas páginas 21 e 29.

HAWKINS, D. M. *Identification of Outliers*. Springer Netherlands, 1980. Disponível em: <https://doi.org/10.1007/978-94-015-3994-4>. Citado na página 22.

HONG, A.; SUN, C.; CHEN, M. A survey of distributed database systems based on blockchain. In: IEEE. *2020 3rd International Conference on Smart BlockChain (SmartBlock)*. [S.l.], 2020. p. 191–196. Citado na página 30.

KARAU, H. et al. *Learning Spark*. O'Reilly, 2015. ISBN 9781449358624 1449358624. Disponível em: <https://www.amazon.de/Learning-Spark-Lightning-Fast-Data-Analysis-ebook/dp/B00SW0TY8O>. Citado 2 vezes nas páginas 32 e 33.

LIU, C. et al. Multiple submodels parallel support vector machine on spark. In: *2016 IEEE International Conference on Big Data (Big Data)*. [S.l.: s.n.], 2016. p. 945–950. Citado na página 29.

MARQUES, H. O. et al. Internal evaluation of unsupervised outlier detection. *ACM Transactions on Knowledge Discovery from Data*, Association for Computing Machinery (ACM), v. 14, n. 4, p. 1–42, jul. 2020. Disponível em: <https://doi.org/10.1145/3394053>. Citado na página 22.

MARQUES, H. O. et al. On the internal evaluation of unsupervised outlier detection. In: *Proceedings of the 27th International Conference on Scientific and Statistical Database Management*. New York, NY, USA: Association for Computing Machinery, 2015. (SSDBM '15). ISBN 9781450337090. Disponível em: <https://doi.org/10.1145/2791347.2791352>. Citado 6 vezes nas páginas 22, 25, 26, 29, 33 e 40.

MEYER, O.; BISCHL, B.; WEIHS, C. Support vector machines on large data sets: Simple parallel approaches. In: _____. [S.l.: s.n.], 2014. p. 87–95. Citado na página 29.

MILJKOVIĆ, D. Review of novelty detection methods. In: . [S.l.: s.n.], 2010. p. 593–598. ISBN 978-1-4244-7763-0. Citado na página 21.

NASHAT, D.; AMER, A. A. A comprehensive taxonomy of fragmentation and allocation techniques in distributed database design. *ACM Computing Surveys (CSUR)*, ACM New York, NY, USA, v. 51, n. 1, p. 1–25, 2018. Citado na página 30.

NAVARRO, C. A.; HITSCHFELD-KAHLER, N.; MATEU, L. A survey on parallel computing and its applications in data-parallel problems using gpu architectures. *Communications in Computational Physics*, Cambridge University Press, v. 15, n. 2, p. 285–329, 2014. Citado na página 22.

NAVARRO, C. A.; HITSCHFELD-KAHLER, N.; MATEU, L. A survey on parallel computing and its applications in data-parallel problems using gpu architectures. *Communications in Computational Physics*, Cambridge University Press, v. 15, n. 2, p. 285–329, 2014. Citado 2 vezes nas páginas 28 e 29.

OPITZ, D.; MACLIN, R. Popular ensemble methods: An empirical study. *Journal of Artificial Intelligence Research*, AI Access Foundation, v. 11, p. 169–198, ago. 1999. Disponível em: <https://doi.org/10.1613/jair.614>. Citado na página 29.

PORWAL, U.; MUKUND, S. Credit card fraud detection in e-commerce. In: *2019 18th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/13th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*. IEEE, 2019. Disponível em: <https://doi.org/10.1109/trustcom/bigdatase.2019.00045>. Citado na página 21.

SHVACHKO, K. et al. The hadoop distributed file system. In: *2010 IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST)*. [S.l.: s.n.], 2010. p. 1–10. Citado na página 30.

TARUN, S.; BATTH, R. S.; KAUR, S. A review on fragmentation, allocation and replication in distributed database systems. In: *2019 International Conference on Computational Intelligence and Knowledge Economy (ICCIKE)*. [S.l.: s.n.], 2019. p. 538–544. Citado na página 30.

VERBRAEKEN, J. et al. A survey on distributed machine learning. *ACM COMPUTING SURVEYS*, v. 53, n. 2, p. 33, 2020. ISSN 0360-0300. Disponível em: <http://dx.doi.org/10.1145/3377454>. Citado na página 29.

VERBRAEKEN, J. et al. A survey on distributed machine learning. *Acm computing surveys (csur)*, ACM New York, NY, USA, v. 53, n. 2, p. 1–33, 2020. Citado na página 30.

VáZQUEZ, F. I. et al. Mdcgen: Multidimensional dataset generator for clustering. *Journal of Classification*, v. 36, 04 2019. Citado na página 43.

WHITE, T. *Hadoop: The Definitive Guide*. 4. ed. Beijing: O'Reilly, 2015. ISBN 978-1-4919-0163-2. Disponível em: <https://www.safaribooksonline.com/library/view/hadoop-the-definitive/9781491901687/>. Citado na página 32.

XU, C. et al. A mapreduce based parallel svm for email classification. *Journal of Networks*, v. 9, 06 2014. Citado na página 29.

YAN, Y.; CAO, L.; RUNDENSTEINER, E. A. Distributed top-n local outlier detection in big data. In: *2017 IEEE International Conference on Big Data (Big Data)*. IEEE, 2017. Disponível em: <https://doi.org/10.1109/bigdata.2017.8257998>. Citado na página 21.

YOUSRI, N. A. A validity index for outlier detection. *2010 10th International Conference on Intelligent Systems Design and Applications*, p. 325–329, 2010. Citado 2 vezes nas páginas 25 e 26.

YU, Y. et al. Outlier detection over massive-scale trajectory streams. *ACM Transactions on Database Systems*, Association for Computing Machinery (ACM), v. 42, n. 2, p. 1–33, jun. 2017. Disponível em: <https://doi.org/10.1145/3013527>. Citado 2 vezes nas páginas 22 e 27.

ZAHARIA, M. et al. Resilient distributed datasets: A {Fault-Tolerant} abstraction for {In-Memory} cluster computing. In: *9th USENIX Symposium on Networked Systems Design and Implementation (NSDI 12)*. [S.l.: s.n.], 2012. p. 15–28. Citado na página 30.

ZAHARIA, M. et al. Spark: Cluster computing with working sets. In: *2nd USENIX Workshop on Hot Topics in Cloud Computing (HotCloud 10)*. [S.l.: s.n.], 2010. Citado na página 30.

ZIMEK, A.; CAMPELLO, R. J.; SANDER, J. Ensembles for unsupervised outlier detection. *ACM SIGKDD Explorations Newsletter*, Association for Computing Machinery (ACM), v. 15, n. 1, p. 11–22, mar. 2014. Disponível em: <https://doi.org/10.1145/2594473.2594476>. Citado na página 25.

ZIMEK, A. et al. Subsampling for efficient and effective unsupervised outlier detection ensembles. In: *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*. [S.l.: s.n.], 2013. p. 428–436. Citado na página 44.