Avaliação das Técnicas Gulosa e Probabilística no Desempenho do Algoritmo de Otimização de Colônia de Formigas

Ana Carolina Medeiros Gonçalves, Maria Eduarda Oliveira Brito, Henrique Cota de Freitas, Cristiane Neri Nobre

Grupo de Arquitetura de Computadores e Processamento Paralelo (CArT)
Laboratório de Inteligência Computacional Aplicada (LICAP)
Instituto de Ciências Exatas e Informática (ICEI)
Pontifícia Universidade Católica de Minas Gerais (PUC Minas)
Belo Horizonte, MG – Brasil

ana.medeiros@sga.pucminas.br, oeduarda239@gmail.com,

{cota,nobre}@pucminas.br

Resumo. *O Big Data Analytics e os algoritmos de Aprendizado de Máquina enfrentam desafios significativos ao lidar com grandes volumes de dados, tornando as técnicas de pré-processamento essenciais nesse contexto. Uma dessas técnicas é a Seleção de Instâncias, que identifica as instâncias mais relevantes em uma base de dados. Este estudo compara duas abordagens do algoritmo de Otimização por Colônia de Formigas (ACO) para a seleção de instâncias: a heurística gulosa e a abordagem probabilística. Em 16 bases de dados, a abordagem gulosa reduziu o tamanho das bases em média 50% e apresentou um tempo de execução quase pela metade em relação à abordagem probabilística.*

1. Introdução

O *Big Data Analytics* (BDA) tem se tornado uma prática amplamente adotada pelas organizações, resultando no crescimento exponencial dos volumes de dados (Akinyelu, 2020). Esses dados são submetidos a algoritmos de Aprendizado de Máquina, que utilizam modelos matemáticos para abstrair problemas do mundo real e realizar previsões com base nos dados rotulados (Tovias-Alanis et al., 2021).

A complexidade computacional dos algoritmos depende diretamente do número de instâncias de treinamento. Em bases de dados com muitas instâncias, o tempo computacional torna-se elevado, fazendo das predições um grande desafio (Tovias-Alanis et al., 2021). Para aprimorar os modelos de predição, é essencial realizar procedimentos de pré-processamento, como a seleção de instâncias (Tsai et al., 2021). A fase de pré-processamento, portanto, torna-se uma aliada crucial na busca por maior precisão e desempenho nas bases de dados.

Este trabalho recebeu suporte da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) - PROAP 88887.842889/2023-00–PUC/MG, PDPG 88887.708960/2022-00–PUC/MG, Código de Financiamento 001; do Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) - 311573/2022-3, 311697/2022-4; da Fundação de Amparo à Pesquisa do Estado de Minas Gerais (FAPEMIG) - PCE-00437-24, APQ-03076-18, APQ-05058-23; e da Pontifícia Universidade Católica de Minas Gerais (PUC Minas) - FIP 2024/30947, PIBIC/PIBIT-2023/29487.

A Seleção de Instâncias é uma fase de pré-processamento de dados amplamente utilizado em problemas de classificação. O objetivo é obter um subconjunto de instâncias de uma base de dados e este subconjunto manter o mesmo desempenho ou melhor em um tempo mais curto de execução, do que na base de dados original (Cheng et al., 2021). Um dos algoritmos mais utilizados na seleção de instâncias é o otimização de colônia de formigas, do inglês *Ant Colony Optimization* (ACO), utilizado neste trabalho. Em resumo, o ACO é um algoritmo meta-heurístico que usa uma população de formigas e seus feromônios para encontrar as melhores instâncias (Dorigo e Di Caro, 1999).

O algoritmo de colônia de formigas (ACO), conforme evidenciado por Gonçalves (2022), provou ser altamente eficiente na seleção das melhores instâncias. No entanto, apesar de ser um algoritmo de tempo polinomial, o tempo de execução muitas vezes se torna impraticável em grandes conjuntos de dados devido à alta complexidade resultante da busca probabilística pelo melhor resultado possível.

Normalmente, algoritmos que entregam ótimos resultados globais são computacionalmente caros. Alternativamente, métodos heurísticos foram propostos para fornecer ótimos aproximados (Barus et al., 2011). Algoritmos de busca heurística subótimos, como a busca gulosa, permitem encontrar soluções quando as restrições de tempo, memória ou ambos impedem a aplicação de algoritmos ótimos (Wilt e Ruml, 2015).

Diante dessa realidade, este estudo propõe uma nova abordagem de busca, baseada na estratégia gulosa. O objetivo é comparar o desempenho computacional e as estimativas de predição obtidas tanto pelo ACO original, conforme conduzido por Hott. et al. (2022), quanto pelo ACO aprimorado com a estratégia de busca gulosa. A proposta visa identificar se a introdução dessa estratégia pode melhorar a eficiência do algoritmo, tanto em termos de tempo de execução quanto na qualidade das soluções encontradas. Essa comparação é crucial para avaliar o potencial da estratégia gulosa como uma ferramenta para aprimorar algoritmos de otimização inspirados em colônias de formigas, oferecendo, assim, uma alternativa promissora para problemas que demandam alta capacidade computacional e precisão nas estimativas de predição. Portanto, a principal contribuição deste trabalho é a proposta de uma nova abordagem de busca baseada na estratégia gulosa para aprimorar o desempenho do algoritmo ACO.

Este texto segue a seguinte estrutura: na Seção 2 é apresentado o referencial teórico, a explicação de como a Colônia de Formigas funciona e a análise de sua complexidade. Na Seção 3 são apresentados alguns trabalhos relacionados sobre o ACO. Na Seção 4 é descrita a Metodologia que contém a descrição das bases de dados utilizadas neste artigo, os parâmetros utilizados no ACO, o algoritmo de Aprendizado de Máquina utilizado, a métrica de avaliação de qualidade de classificação *F-Measure* e a análise de complexidade e de desempenho do ACO agora utilizando a estratégia gulosa. Os resultados obtidos e suas análises serão mostrados na Seção 6. Por fim, na Seção 7 são expostas as conclusões obtidas no trabalho e os trabalhos futuros.

2. Referencial Teórico

Nesta seção são apresentados e aprofundados os conceitos fundamentais abordados neste trabalho, como a Seleção de Instâncias, o funcionamento do algoritmo ACO, e a análise de complexidade do algoritmo original baseado na heurística probabilística e uma breve introdução sobre como funciona a heurística gulosa.

2.1. Seleção de Instâncias

A Seleção de Instâncias é uma etapa de pré-processamento de Aprendizado de Máquina, que consiste em produzir subconjuntos de uma base de dados, com o objetivo de melhorar a eficiência das bases utilizadas, e gerar melhores resultados nas métricas utilizadas como referência (Carbonera e Abel, 2020). Segundo Tsai et al. (2021), a utilização da seleção de instâncias como uma técnica de pré-processamento normalmente obtém resultados melhores nas bases de dados reduzidas, comparando-as com as bases antes do pré-processamento.

O funcionamento da Seleção de Instâncias consiste em um conjunto de treinamento T, que seleciona as instâncias mais importantes em um subconjunto S de T, sendo que S não contém dados redundantes. Assim, os modelos treinados do subconjunto S devem ter um desempenho semelhante ou superior ao conjunto T (Tsai et al., 2021).

2.2. Colônia de formigas

O algoritmo de otimização de colônia de formigas (ACO) é uma meta-heurística inspirada no comportamento de colônias de formigas reais em busca de alimento. A ideia central é utilizar um conjunto de formigas artificiais cooperando entre si para encontrar o menor caminho entre dois pontos em um grafo. O caminho mais curto encontrado representa a melhor solução para o problema (Anwar et al., 2015).

As formigas comunicam-se indiretamente por meio do depósito de feromônios. A solução considerada a melhor naquele momento recebe uma quantidade de feromônio "artificial" das formigas (Salama e Freitas, 2013). A quantidade de feromônio atribuída a cada solução reflete sua qualidade relativa, sendo que soluções melhores têm maior probabilidade de receber mais feromônios (Salama e Freitas, 2013). Com o tempo, as formigas tendem a se concentrar nos caminhos que representam soluções quase ótimas dentro do espaço de busca (Anwar et al., 2015).

2.2.1. Análise de complexidade da Colônia

Apesar da alta complexidade, o ACO tem se mostrado bem-sucedido na resolução de problemas de classificação (Salama et al., 2016). O ACO consegue encontrar soluções de boa qualidade em tempo polinomial devido à sua abordagem distribuída e estocástica. Nesse método, as formigas constroem soluções parciais e compartilham informações sobre a qualidade delas. A utilização inteligente de "feromônios" direciona a busca para soluções promissoras, permitindo ao ACO concentrar seus esforços onde soluções ótimas são mais prováveis. Isso mantém a eficiência do ACO mesmo em problemas desafiadores.

A função de complexidade determinada para o ACO[1], com base no número de operações, é expressa pela função de tempo T(n), que descreve a complexidade computacional do algoritmo em função das principais variáveis envolvidas: o número de instâncias (*num_instances*), o número de atributos (*num_attributes*) e o número de formigas (*num_ants*), no contexto do ACO.

[1]O código implementado com a abordagem probabilística, juntamente com os comentários sobre as complexidades identificadas, podem ser acessados em: https://github.com/AnaCarolina99/TypesOfACO/blob/main/ProbabilisticComplexity.py.

$T(n) = O(num\_instances \times num\_attributes + num\_ants \times num\_instances^2 \times log(num\_instances) + num\_ants \times num\_instances \times num\_attributes)$

Ao analisar a complexidade assintótica, são considerados os termos que apresentam o crescimento mais rápido à medida que os valores de *num_instances*, *num_attributes* e *num_ants* aumentam. Nesse contexto, o termo dominante é $num\_ants \times num\_instances^2 \times log(num\_instances)$, pois o fator $num\_instances^2$ e o logaritmo crescem mais rapidamente em comparação aos demais termos.

Assim, é possível chegar à seguinte ordem de complexidade: $O(num\_ants \times num\_instances^2 \times log(num\_instances))$.

2.2.2. Estratégia Gulosa

A heurística gulosa constrói a solução de maneira incremental, adicionando a cada etapa o elemento candidato que, segundo um critério de escolha, é considerado o "melhor" dentro do contexto atual (da Silva Fonseca, 2022). Este método de busca segue um caminho até o objetivo, podendo eventualmente encontrar um "beco sem saída" (Maggio, 2004). A heurística gulosa faz escolhas locais ótimas em cada etapa, o que pode levar a decisões irreversíveis que não asseguram uma solução globalmente ótima (Silva e Pereira, 2020). Entre suas principais aplicações estão problemas de otimização como roteamento, cobertura de conjuntos, coloração de grafos e o problema da mochila (*Knapsack Problem*).

3. Trabalhos Relacionados

Esta seção apresenta uma revisão dos trabalhos relacionados ao tema em questão, focando especialmente nos estudos que abordam o algoritmo ACO. Diversos estudos têm investigado o uso de algoritmos baseados em otimização por colônia de formigas (ACO) para resolver problemas complexos. Entre as referências mais relevantes, destaca-se o artigo dos autores que desenvolveram a heurística do ACO (Dorigo e Di Caro, 1999), bem como o artigo dos autores que a utilizaram para criar um algoritmo de seleção de instâncias baseado nessa técnica (Anwar et al., 2015). Além disso, são discutidos artigos nos quais a implementação gulosa foi utilizada em combinação com o algoritmo de Colônia de Formigas. Todos esses trabalhos indicam resultados satisfatórios em suas respectivas abordagens algorítmicas, evidenciando a eficiência e a capacidade de resolver problemas complexos.

O artigo de Dorigo e Di Caro (1999) apresenta a Otimização por Colônia de Formigas (ACO), uma heurística inspirada no comportamento de forrageamento das formigas, que tem sido aplicada à solução de problemas complexos de otimização discreta. O artigo fornece exemplos paradigmáticos de aplicações dessa nova heurística e oferece uma visão geral das aplicações já existentes. Essas aplicações evidenciam a eficácia e versatilidade da ACO em resolver problemas difíceis, contribuindo significativamente para o campo da otimização.

Anwar et al. (2015) foram responsáveis por aplicar a lógica do ACO com o propósito de desenvolver um algoritmo para a seleção de instâncias. O objetivo era criar um algoritmo que produzisse para os modelos de classificação uma predição de acurácia melhor. Os autores utilizaram 37 bases de dados e chegaram à conclusão de que as

reduções foram eficazes, além de terem obtidos resultados melhores para os valores de acurácia.

Salama et al. (2016) e Anwar et al. (2015) compararam a eficiência do ACO com o *Coevolution of Instance Selection and Weighting Schemes for Nearest Neighbour*(CIW-NN), utilizando 43 bases de dados e 5 algoritmos de classificação conhecidos. Como conclusão, o ACO obteve ótimos resultados na redução de bases e resultados melhores nos algoritmos de classificação. O CIW-NN obteve resultados positivos tanto na redução quanto nos algoritmos de classificação. No entanto, os autores não conseguiram concluir qual dos dois seria melhor.

Hott. et al. (2022) utilizaram o algoritmo de seleção de instâncias colônia de formigas (ACO) aplicada a uma base de dados de crianças e adolescentes com TDAH, quanto ao desempenho escolar. Foram utilizados também os algoritmos de classificação, *Random Forest*, *Neural Network*, *K-Nearest Neighbors*(KNN) e *Classification and Regression Trees*(CART). Como resultado, a seleção de instâncias com o algoritmo ACO gerou predições mais efetivas.

O artigo de Vendramin et al. (2011) apresenta um novo protocolo de roteamento baseado em previsão para Redes Tolerantes a Atrasos (DTNs), denominado *Greedy Ant* (GrAnt). As simulações mostram que o GrAnt alcança uma taxa de entrega de mensagens significativamente superior e uma menor sobrecarga de bytes em comparação com outros protocolos. Por exemplo, com um tamanho de *buffer* de 5 MB, o GrAnt entrega quase 79% das mensagens, enquanto os outros protocolos entregam apenas 39% e 45%, respectivamente.

Jia et al. (2022) propõem um algoritmo de cobertura de caminho para Veículos Aéreos Não Tripulados (UAVs), que combina a estratégia gulosa com a Otimização por Colônia de Formigas (ACO). As simulações realizadas evidenciam que o algoritmo proposto reduz o tempo de conclusão da tarefa em 2,8% e melhora a taxa de cobertura em 4,4% em comparação com métodos anteriores.

No artigo de Morin et al. (2023), foi abordado o problema da otimização de caminhos de busca. Variantes do ACO foram avaliadas e desenvolvidas, incluindo a implementação de uma heurística gulosa. Os resultados mais promissores, sobretudo em instâncias de maior escala, foram obtidos com o uso dessa heurística, mostrando sua eficácia em cenários mais complexos.

Embora vários estudos tenham explorado a aplicação do ACO para resolver problemas complexos de otimização e seleção de instâncias, o presente trabalho se distingue ao introduzir uma abordagem que combina a estratégia gulosa com o ACO. Dorigo e Di Caro (1999), Anwar et al. (2015), Hott. et al. (2022) destacam a eficácia do ACO em otimização e seleção de instâncias, e mostram a utilidade do ACO em contextos específicos. Além disso, o trabalho de Vendramin et al. (2011) se concentra em protocolos de roteamento para redes tolerantes a atrasos e o trabalho de Jia et al. (2022) combina ACO com uma estratégia gulosa para UAVs, mas não aborda a comparação entre diferentes abordagens do ACO. Assim, a contribuição para este trabalho está na análise detalhada do impacto da estratégia gulosa na eficiência e precisão do ACO, oferecendo uma perspectiva nova e prática para a melhoria de algoritmos de otimização em cenários de alta complexidade computacional com avaliação de desempenho.

4. Metodologia

Esta seção apresenta as bases de dados utilizadas neste estudo, detalha o ambiente de hardware empregado, descreve o algoritmo de aprendizado de máquina selecionado e define as métricas utilizadas para avaliar os resultados.

4.1. Descrição das bases de dados

Para avaliar o desempenho da Colônia de Formiga na seleção das instâncias, foram utilizadas 16 (dezesseis) bases públicas, com grandes variações de tamanho de instâncias e atributos, além de quatro bases privadas.

A descrição detalhada da quantidade de instâncias, os tipos de atributos de entrada, a quantidade total de atributos presentes em cada base e o endereço das bases utilizadas neste trabalho estão apresentados na Tabela 1. Vale ressaltar que as classificações dos atributos de entrada foram baseadas nas descrições divulgadas pelos autores de cada uma das bases de dados.

Tabela 1. Descrição das bases de dados quanto ao número de instâncias e tipos de atributos

		Atributos de Entrada						
Base de Dados	**Instâncias**	Contínuo	Discreto	Cat. Ord.	Cat.Não Ord.	Cat. Bin.	**Quant. Total**	**Link**
Optdigits	5620	-	62	-	-	3	65	Link
Brain Stroke	4981	2	1	-	2	6	11	Link
Abalone	4177	7	-	-	2	-	9	Link
Vaccines	3353	8	1	14	2	17	42	Link
Splice	3186	-	-	-	1	180	181	Link
Marketing Analysis	2205	-	19	-	3	17	39	Link
Fetal Health	2126	7	13	-	2	-	22	Link
Yeast	1484	7	-	-	1	1	9	Link
Contraceptive	1473	-	2	5	-	3	10	Link
Titanic	1309	1	20	2	3	2	28	Link
Cyber Security	1247	-	2	5	4	-	11	Link
Tic-Tac-Toe	959	-	-	-	9	1	10	Link
Diabetes	768	2	5	1	-	1	9	Link
Data Science	607	-	3	4	5	-	12	Link
Cirrhosis	418	5	6	2	6	1	20	Link
Ionosphere	351	32	-	-	-	3	35	Link
Lung Cancer	309	-	1	-	-	15	16	Link
Haberman	306	-	2	1	-	1	4	Link

*Esta tabela está ordenada, de forma decrescente, pela quantidade de instâncias.

Cat. Ord = Atributos Categóricos Ordinais. *Cat. Não Ord* = Atributos Categóricos Não Ordinais. *Cat. Bin* = Atributos Categóricos Binários

4.2. Ambiente de Desenvolvimento e Teste

Para a execução dos algoritmos, foi utilizado o *Python* 3.9 em um computador com processador Intel i7-4770 3.40GHz, 4 núcleos e 8 threads, e 32 GB de memória RAM.

4.3. Algoritmo de Aprendizado de Máquina

Para avaliar a eficiência do ACO, foi escolhido o algoritmo *Random Forest*, introduzido por Breiman (2001). O *Random Forest* é um método *Ensemble* que constrói um conjunto de árvores de decisão aleatórias, treinadas em subconjuntos de atributos selecionados aleatoriamente, utilizando o método de reamostragem *Bootstrap*[2]. O algoritmo prossegue até atingir a quantidade pré-definida de árvores, e o resultado final para problemas de classificação é determinado pelo voto majoritário.

[2]Conjuntos aleatórios de dados com possibilidade de repetição.

Para implementar e comparar os algoritmos, foi utilizada a biblioteca *Scikit-Learn*[3] da linguagem *Python*, com 70% dos dados para treinamento e 30% para teste.

4.4. Métricas de avaliação

Para avaliar a eficiência das bases de dados reduzidas pelo ACO, será utilizado a métrica de avaliação *F-Measure*.

A *F-Measure*, Equação 1, representa a média harmônica entre precisão e sensibilidade.

$$F - Measure = \frac{2 \times Recall \times Precision}{Recall + Precision} \tag{1}$$

O Ganho, Equação 2, é uma métrica que quantifica o aumento de velocidade ao comparar o tempo de execução de dois algoritmos. Neste contexto, ele é utilizado para avaliar o ganho de desempenho da versão gulosa em relação à versão probabilística do ACO.

$$Ganho = \frac{ACOProb.T}{ACOGulosaT} \tag{2}$$

Representado pela Equação 3, o desvio padrão é utilizado para avaliar a variabilidade, confiabilidade e identificação de *outliers* no tempo de execução. Ou seja, quanto menor o desvio padrão, maior a confiabilidade do tempo de execução e menor a variabilidade dos resultados.

$$DesvioPadrão = \sqrt{\frac{\sum_{i=1}^{n}(x_i - \bar{x})^2}{n}} \tag{3}$$

5. Desenvolvimento da Abordagem Gulosa

Considerando que a heurística de busca do ACO é probabilística e possui a complexidade *O(num_ants* $\times$ *num_instances*^{2} $\times$ *log(num_instances))*, surgiu a necessidade de explorar alternativas para otimizar o tempo de execução em bases de dados extensas, permitindo a viabilidade do algoritmo em cenários polinomiais sem comprometer a busca.

Com esse objetivo, foi proposta a abordagem gulosa[4]. Na heurística gulosa, é necessário modificar a forma como as formigas escolhem a próxima instância a ser visitada durante a construção de seus trajetos. Na versão probabilística, as formigas selecionam a próxima instância com base em probabilidades calculadas a partir das trilhas de feromônio e taxas de visibilidade. Já na versão gulosa, as formigas selecionam a próxima instância com base apenas no valor mais alto da trilha de feromônio, sem levar em consideração a visibilidade.

Com essas alterações, as formigas passam a seguir a heurística gulosa, selecionando a próxima instância com base no maior valor de trilha de feromônio, o que resultará em uma estratégia de exploração diferente em comparação com a abordagem probabilística.

[3]https://scikit-learn.org/stable/

[4]O código implementado com a abordagem gulosa, juntamente com os comentários sobre as complexidades identificadas, podem ser acessados em: https://github.com/AnaCarolina99/TypesOfACO/blob/main/GulosoComplexity.py.

A função de complexidade obtida para o ACO, utilizando a heurística gulosa e considerando o número de operações, é representada pela expressão a seguir:

$T(n) = O(num\_ants \times num\_instances^2) + O(num\_ants \times num\_instances \times num\_attributes) + O(num\_instances^2 \times num\_attributes)$

Deste modo, é possível chegar à seguinte ordem de complexidade: $O(num\_ants \times num\_instances^2)$.

6. Resultados

Nesta seção, são realizadas avaliações e comparações do tempo de execução, da porcentagem de redução e do ganho proporcionado pela heurística gulosa em relação à heurística probabilística. É importante destacar que ambas as heurísticas foram executadas cinco vezes em cada base de dados, e os resultados, incluindo o desvio padrão e o tempo médio de execução para cada base, estão representados na Tabela 2.

A Tabela 2 apresenta as seguintes informações: 1) Bases de dados selecionadas; 2) Quantidade de instâncias na base de dados original; 3) Quantidade de instâncias reduzidas após a aplicação do ACO com a heurística probabilística; 4) Porcentagem de redução para cada base de dados utilizando a heurística probabilística; 5) Tempo de processamento para cada base de dados com a heurística probabilística; 6) Desvio Padrão de cada base de dados utilizando a heurística probabilística; 7) Quantidade de instâncias reduzidas após a aplicação do ACO com a heurística gulosa; 8) Porcentagem de redução para cada base de dados utilizando a heurística gulosa; 9) Tempo de processamento para cada base de dados com a heurística gulosa; 10) Desvio Padrão de cada base de dados utilizando a heurística gulosa; 11) Ganho obtido pela abordagem gulosa.

Tabela 2. Comparação de Redução de Bases de Dados e Tempos de Processamento: ACO Probabilístico X ACO Guloso em Função da Quantidade de Instâncias

Base de Dados	Colônia Probabilística		% de Redução	Tempo	Desvio Padrão	Colônia Guloso	% de Redução	Tempo	Desvio Padrão	Ganho
	Orig.	Redução				Redução				
Optdigits	5620	2858	49	34h05	2.4min	2845	49	17h23	0.66min	1.96
Banana	5300	2664	50	29h07	2.56min	2694	49	14h51	1.17min	1.96
Brain Stroke	4981	2486	50	21h04	1.45min	2547	49	11h22	0.92min	1.85
Employee Future	4653	2375	50	17h00	1.41min	2333	49	9h48	0.4min	1.73
Abalone	4177	2204	47	13h01	1.96min	2181	48	6h44	1.55min	1.93
Vaccines	3152	1573	50	5h42	1.02min	1591	50	2h59	1.62min	1.91
Marketing Analysis	2205	1154	48	1h58	0.63min	1126	49	59min	1.25min	2
Fetal Health Classification	2126	1089	49	1h46	0.75min	1103	48	56min	0.51min	1.86
Yeast	1484	770	48	36min	0.42min	787	48	18min	1.27min	2
Titanic	1309	644	51	24min	0.8min	674	50	12min	0.76min	2
Cyber Security Salaries	1247	667	47	20min	2.1min	658	47	10min	0.75min	2
Tic-Tac-Toe	959	495	48	10min	0.66min	507	47	5min	0.49min	2
Diabetes	768	406	47	5min	0.63min	338	56	2min	0.44min	2.5
Data Science Salaries	607	333	45	2min	0.49min	338	44	1min	0.4min	2
Cirrhosis	418	220	47	65s	0.75s	228	46	29s	0.89s	2.24
Haberman	306	158	48	20s	0.75s	166	46	11s	1,02s	1.82

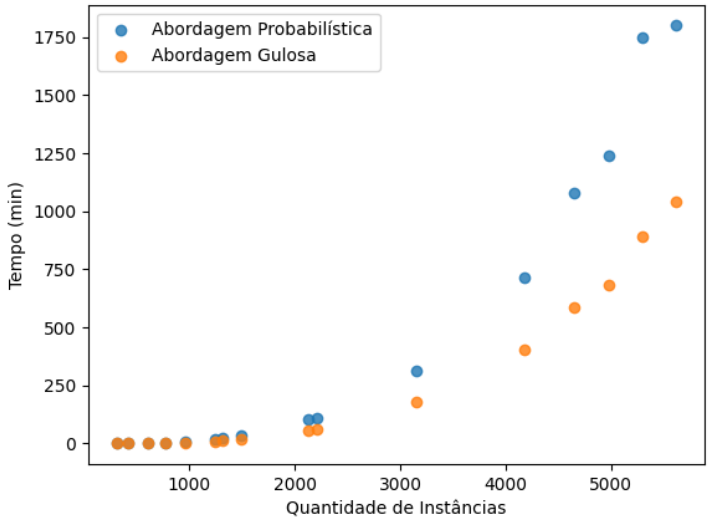
*Esta tabela está ordenada, de forma decrescente, pela quantidade de instâncias.

Nesta tabela, observa-se que a quantidade média de redução das bases de dados com a abordagem gulosa foi de aproximadamente 50%, semelhante à abordagem proba-

bilística. No entanto, o tempo de execução da abordagem gulosa é quase metade do tempo necessário para o algoritmo original (abordagem probabilística), como evidenciado pelo ganho médio de 1,99. Outro ponto a ser destacado é que, em ambas as abordagens, o desvio padrão foi baixo, indicando uma pequena variabilidade nos tempos de execução e uma alta confiabilidade nos resultados obtidos. Na versão probabilística, o desvio padrão médio foi de 1,17 minutos, enquanto na implementação gulosa foi de 0,88 minutos.

A Figura 1 ilustra a relação entre o tempo de execução e a quantidade de instâncias para cada uma das estratégias, conforme detalhado na Tabela 2. Como mostrado, o tempo de execução aumenta com a quantidade de instâncias. A abordagem gulosa, embora exiba um tempo de execução cerca de metade do tempo do algoritmo original (abordagem probabilística), ainda apresenta um crescimento quadrático.

Figura 1. Quantidade de Instâncias x Tempo em Cada Abordagem



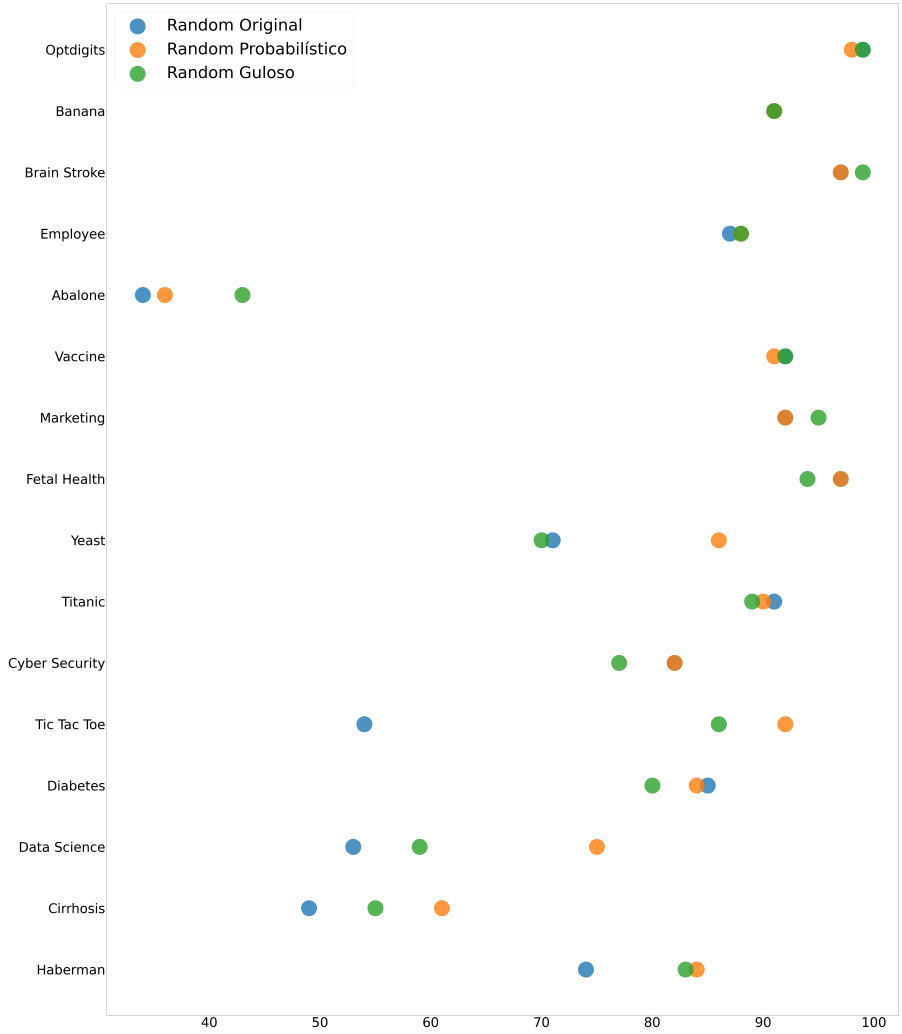
Com o intuito de realizar uma comparação mais precisa dos valores obtidos pelo *F-Measure* em cada uma das abordagens, utilizou-se uma visualização por meio de Gráficos de Dispersão na Figura 2. Esses gráficos permitem observar os valores obtidos pela base original antes da redução e os valores obtidos em cada tipo de abordagem.

Ao analisar as figuras, observa-se que tanto a abordagem probabilística quanto a gulosa do ACO resultam em ganhos ou resultados similares no F-Measure para o Random Forest, em comparação com as bases originais. A abordagem gulosa apresenta melhores valores no F-Measure a partir da base “Marketing” até a “Optdigits”. Para bases abaixo de “Marketing”, os resultados são geralmente semelhantes aos da abordagem probabilística ou ligeiramente inferiores, mas quase sempre melhores do que os valores das bases antes da aplicação do ACO.

7. Conclusão

O objetivo deste trabalho foi analisar e comparar o desempenho do algoritmo ACO utilizando uma heurística alternativa à original (probabilística). Foi possível executar o algoritmo em todas as bases de dados e reduzir o número de instâncias. A heurística gulosa

Figura 2. F-Measure no Random Forest com dados originais e após a redução com todas as abordagens



apresentou um tempo de execução aproximadamente metade do tempo da heurística probabilística, enquanto ambas as abordagens obtiveram resultados de redução semelhantes.

Focando na avaliação dos resultados, observa-se que as bases de dados reduzidas pela heurística gulosa apresentaram um aumento percentual no F-Measure ou mantiveram valores semelhantes aos das bases redduzidas pelo ACO original. Vale ressaltar que a abordagem gulosa obteve valores melhores para bases a partir de 2205 instâncias, em comparação tanto com as bases originais quanto com as bases reduzidas pela heurística probabilística.

Apesar de sua natureza de resolver problemas com soluções subótimas e decisões irreversíveis, o ACO guloso apresentou eficácia na seleção das melhores instâncias com tempos de execução melhores em comparação com o ACO original.

Para trabalhos futuros, podem ser implementadas novas abordagens, incluindo heurísticas aleatórias, programação dinâmica e "divisão e conquista" no ACO. O objetivo é comparar os resultados em termos de métricas de aprendizado de máquina e tempo de execução, em relação às heurísticas já implementadas (gulosa e probabilística) e às bases de dados antes da aplicação do ACO. Esses esforços podem representar um avanço significativo no campo, oferecendo soluções mais eficientes para a redução de bases de dados e aumento do desempenho.

Referências

A. A. Akinyelu. Bio-inspired technique for improving machine learning speed and big data processing. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, 2020. doi: 10.1109/IJCNN48605.2020.9206762.

I. M. Anwar, K. M. Salama, e A. M. Abdelbar. Instance selection with ant colony optimization. *Procedia Computer Science*, 53:248–256, 2015. ISSN 1877-0509. doi: https://doi.org/10.1016/j.procs.2015.07.301. INNS Conference on Big Data 2015 Program San Francisco, CA, USA 8-10 August 2015.

A. C. Barus, T. Y. Chen, D. Grant, F. C. Kuo, e M. F. Lau. Testing of heuristic methods: A case study of greedy algorithm. In Z. Huzar, R. Koci, B. Meyer, B. Walter, e J. Zendulka, editors, *Software Engineering Techniques*, pages 246–260, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg. doi: 10.1007/978-3-642-22386-0_19.

L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001. doi: 10.1023/A:1010950718922.

J. L. Carbonera e M. Abel. An attraction-based approach for instance selection. In *2020 IEEE 32nd International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 1053–1058, 2020. doi: 10.1109/ICTAI50040.2020.00161.

F. Cheng, F. Chu, e L. Zhang. A multi-objective evolutionary algorithm based on length reduction for large-scale instance selection. *Information Sciences*, 576, 06 2021. doi: 10.1016/j.ins.2021.06.052.

G. da Silva Fonseca. Heurística gulosa para o problema dinâmico de coleta e entrega de pacientes. *Galoá Proceedings*, 54:152846, 2022.

M. Dorigo e G. Di Caro. Ant colony optimization: a new meta-heuristic. In *Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406)*, volume 2, pages 1470–1477 Vol. 2, 1999. doi: 10.1109/CEC.1999.782657.

A. C. M. Gonçalves. Análise de desempenho do algoritmo colônia de formigas para seleção de instâncias. Trabalho de conclusão de graduação, Pontifícia Universidade Católica de Minas Gerais, Belo Horizonte, 2022.

H. Hott., C. Jandre., P. Xavier., A. Miloud-Aouidate., D. Miranda., M. Song., L. Zárate., e C. Nobre. Selection of representative instances using ant colony: A case study in a database of children and adolescents with attention-deficit/hyperactivity disorder. In *Proceedings of the 15th International Joint Conference on Biomedical Engineering Systems and Technologies - HEALTHINF,*, pages 103–110. INSTICC, SciTePress, 2022. ISBN 978-989-758-552-4. doi: 10.5220/0010843000003123.

Y. Jia, S. Zhou, Q. Zeng, C. Li, D. Chen, K. Zhang, L. Liu, e Z. Chen. The uav path coverage algorithm based on the greedy strategy and ant colony optimization. *Electronics*, 11(17), 2022. ISSN 2079-9292. doi: 10.3390/electronics11172667.

E. Maggio. *Uma Heurística para a Programação da Produção de FMS usando Modelagem em Redes de Petri*. Editora XYZ, 2004. Busca heurística - Estratégias de Busca para Solução de Problemas.

M. Morin, I. Abi-Zeid, e C.-G. Quimper. Ant colony optimization for path planning in search and rescue operations. *European Journal of Operational Research*, 305(1): 53–63, 2023. ISSN 0377-2217. doi: 10.1016/j.ejor.2022.06.019.

K. Salama, A. Abdelbar, e I. Anwar. Data reduction for classification with ant colony algorithms. *Intelligent Data Analysis*, 20:1021–1059, 09 2016. doi: 10.3233/IDA-160855.

K. M. Salama e A. A. Freitas. Learning bayesian network classifiers using ant colony optimization. *Swarm Intelligence*, 7(2):229–254, 2013. doi: 10.1007/s11721-013-0087-6.

J. Silva e M. Pereira. Heurísticas e algoritmos de busca em problemas de otimização. *Revista de Computação e Matemática*, 15(2):123–134, 2020. Discussão sobre a irreversibilidade das decisões na heurística gulosa.

S. O. Tovias-Alanis, W. Gómez-Flores, e G. Toscano-Pulido. Instance selection based on linkage trees. In *2021 18th International Conference on Electrical Engineering, Computing Science and Automatic Control (CCE)*, pages 1–6, 2021. doi: 10.1109/CCE53527.2021.9633116.

C.-F. Tsai, K.-L. Sue, Y.-H. Hu, e A. Chiu. Combining feature selection, instance selection, and ensemble classification techniques for improved financial distress prediction. *Journal of Business Research*, 130:200–209, 06 2021. doi: 10.1016/j.jbusres.2021.03.018.

A. C. B. K. Vendramin, A. Munaretto, M. R. Delgado, e A. C. Viana. A greedy ant colony optimization for routing in delay tolerant networks. In *2011 IEEE GLOBECOM Workshops (GC Wkshps)*, pages 1127–1132, 2011. doi: 10.1109/GLOCOMW.2011.6162354.

C. M. Wilt e W. Ruml. Building a heuristic for greedy search. In *International Symposium on Combinatorial Search*, pages 131–139, 2015. doi: 10.1609/socs.v6i1.18352.