

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO CEARÁ
PRÓ-REITORIA DE PESQUISA, PÓS-GRADUAÇÃO E INOVAÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Fagner José de Matos Macêdo

**Novas Técnicas Binárias de Seleção de Características Baseada na
Meta-heurística Otimizador de Fluxo de Água**

Fortaleza – CE

14 de abril de 2023

Fagner José de Matos Macêdo

Novas Técnicas Binárias de Seleção de Características Baseada na Meta-heurística Otimizador de Fluxo de Água

Dissertação submetida ao Programa de Pós-Graduação em Ciência da Computação do Instituto Federal de Educação, Ciência e Tecnologia do Ceará, como requisito parcial para obtenção do grau de Mestre em Ciência da Computação.

Linha de Pesquisa: Inteligência Artificial

Orientador: Prof. Dr. Ajalmar Rêgo da Rocha Neto

Instituto Federal de Educação, Ciência e Tecnologia do Ceará (IFCE)

Pró-Reitoria de Pesquisa, Pós-Graduação e Inovação (PRPI)

Programa de Pós-Graduação em Ciência da Computação (PPGCC)

Fortaleza – CE

14 de abril de 2023

Dados Internacionais de Catalogação na Publicação

Instituto Federal do Ceará - IFCE

Sistema de Bibliotecas - SIBI

Ficha catalográfica elaborada pelo SIBI/IFCE, com os dados fornecidos pelo(a) autor(a)

M113n Macêdo, Fagner José de Matos.

Novas Técnicas Binárias de Seleção de Características Baseada na Meta-heurística
Otimizador de Fluxo de Água / Fagner José de Matos Macêdo. - 2023.
105 f. : il. color.

Dissertação (Mestrado) - Instituto Federal do Ceará, Mestrado em Ciência da
Computação, Campus Fortaleza, 2023.

Orientação: Prof. Dr. Ajalmar Rêgo da Rocha Neto.

1. seleção de características. 2. computação evolucionária. 3. redução de
dimensionalidade. 4. inteligência artificial. I. Título.

CDD 004



Instituto Federal de Educação, Ciência e Tecnologia do Ceará (IFCE)
Pró-Reitoria de Pesquisa, Pós-Graduação e Inovação (PRPI)
Programa de Pós-Graduação em Ciência da Computação (PPGCC)

Fagner José de Matos Macêdo

Esta dissertação foi julgada adequada para a obtenção do título de Mestre em Ciência da Computação, sendo aprovada pela Coordenação do Programa de Pós-Graduação em Ciência da Computação do Instituto Federal de Educação, Ciência e Tecnologia do Ceará e pela banca examinadora:

Orientador: Prof. Dr. Ajalmar Rêgo da Rocha Neto

Instituto Federal de Educação, Ciência e Tecnologia do
Ceará (IFCE)

Prof. Dr. Saulo Anderson Freitas de Oliveira
Instituto Federal de Educação, Ciência e Tecnologia do
Ceará (IFCE)

Prof. Dr. Thiago Alves Rocha
Instituto Federal de Educação, Ciência e Tecnologia do
Ceará (IFCE)

Prof. Dr. João Paulo Pordeus Gomes
Universidade Federal do Ceará (UFC)

Fortaleza – CE
14 de abril de 2023

*Dedico este trabalho aos meus pais
Francisco e Rosineide,
aos meus irmãos Alyne e Fabrício
e a minha incrível esposa Ana Beatriz.*

Agradecimentos

Primeiramente agradeço a Deus por todas as bênçãos, permissões, forças e saúde que foi a mim permitido para que eu pudesse perseverar em toda a minha caminhada.

Aos meus pais Rosineide Rodrigues e Francisco José Rodrigues e ao meu padrinho Rodrigo Matos, por sempre me incentivarem e me darem todo apoio e suporte para eu estudar e realizar meus sonhos.

Aos meus irmãos Alyne Maria e Fabrício José e a minha esposa Ana Beatriz, que são fonte de inspiração em minha vida.

Aos meus amigos de curso e em especial a Tamara Arruda, pelos auxílios nos estudos, nos trabalhos ao decorrer do curso e pelos momentos de descontração.

Ao Professor Ajalmar Rocha pelas orientações fundamentais para a execução deste trabalho.

A todos os meus professores que tiveram papel importante na minha formação profissional e pessoal, e a todos os colegas e funcionários que juntos fazem o IFCE cada vez melhor.

*“Não existe triunfo sem perda,
não há vitória sem sofrimento,
não há liberdade sem sacrifício.”
(J.R.R. Tolkien)*

Resumo

A seleção de características é o processo de redução da dimensão de um conjunto de dados, realizado através da escolha dos atributos que contém maior relevância em suas informações. Nesse contexto, a seleção dos atributos mais representativos para as aplicações, visa a melhoria do conjunto de dados, bem como atenuar problemas relacionados a ruídos, e também, o da alta dimensionalidade que podem afetar o preditor.

Neste trabalho, são propostas três versões binárias para o algoritmo otimizador de fluxo de água (*water flow optimizer*, WFO), os métodos propostos são introduzidos abordando o problema de seleção de características visando manter ou até mesmo melhorar o desempenho do classificador floresta de caminhos ótimos (*optimum-path forest*, OPF), que por sua vez é um classificador de fácil desenvolvimento que trata problemas com múltiplas classes e é independente de parâmetros. A meta-heurística WFO, é um algoritmo evolutivo recente, inspirado na forma do fluxo de água na natureza, em que visa simular através de seus operadores, o fluxo laminar e o fluxo turbulento, que dão nome aos operadores. Entretanto, a meta-heurística em sua versão original é capaz de solucionar apenas problemas de otimização contínua.

Em vista disso, o primeiro método é proposto como uma primeira versão binária para a meta-heurística, em que seus operadores foram remodelados visando gerar apenas soluções binárias. A proposta inicial busca no espaço de solução encontrar os atributos de maior relevância, com base em sua aptidão. O segundo método proposto é semelhante ao primeiro, exceto que o seu diferencial é que a quantidade de atributos a serem selecionados é previamente definida, tendo assim, o tamanho fixo. A terceira proposta busca aumentar a capacidade de redução de atributos quando comparado a primeira proposta, assim os operadores foram aprimorados para que o método realize a tarefa com uma taxa maior de remoção de atributos irrelevantes do conjunto de dados. Foram realizadas simulações computacionais para fazer uma análise comparativa entre os métodos propostos e os métodos clássicos de redução de dimensionalidade, mais especificamente com a análise de componentes principais (*principal component analysis*, PCA) e análise discriminante linear (*linear discriminant analysis*, LDA), e com as meta-heurística busca cuco binário (*binary cuckoo search*, BCS), otimização de ondas de água binário (*binary water wave optimization*, BWWO) e o algoritmo do morcego binário (*binary bat algorithm*, BBA). Com base nos experimentos, é assegurado um bom desempenho dos novos métodos em termos de acurácia, obtidos pelo classificador OPF.

Palavras-chave: seleção de características, computação evolucionária, redução de dimensionalidade, inteligência artificial.

Abstract

Feature selection is the process of reducing the size of a data set, performed by choosing the attributes that are most relevant in their information. In this context, the selection of the most representative attributes for the applications, aims to improve the data set, as well as to mitigate problems related to noise, and also, the high dimensionality that can affect the predictor.

In this work, three binary versions of the water flow optimizer (WFO) algorithm are proposed, the proposed methods are introduced addressing the feature selection problem in order to maintain or even improve the performance of the optimum-path forest (OPF) classifier, which in turn is an easy-to-build classifier that handles problems with multiple classes, and is parameter-independent. The WFO meta-heuristic is a recent evolutionary algorithm, inspired by the form of water flow in nature, which aims to simulate through its operators, laminar flow and turbulent flow, which give their name to the operators. However, the metaheuristic in its original version is capable of solving only continuous optimization problems.

In view of this, the first method is proposed as a first binary version for the WFO, in which its operators were remodeled in order to generate only binary solutions. The initial proposal seeks in the solution space to find the most relevant attributes, based on their fitness. The second proposed method is similar to the first one, except that its differential is that the amount of characteristics to be selected is previously defined, thus having a fixed size. The third proposal seeks to increase the attribute reduction capacity when compared to the first proposal, so the operators were improved so that the method performs the task with a higher rate of removal of irrelevant attributes from the dataset. Computer simulations were carried out to make a comparative analysis between the proposed methods and the classical methods of dimensionality reduction, more specifically with the principal component analysis (PCA) and linear discriminant analysis (LDA), and with the metaheuristics binary cuckoo search (BCS), binary water wave optimization (BWVO) and binary bat algorithm (BBA). Based on the experiments, a good performance of the new methods in terms of accuracy, obtained by the OPF classifier, is ensured.

Keywords: feature selection, evolutionary computation, dimensionality reduction, artificial intelligence.

Lista de ilustrações

Figura 1 – Diagrama para o processo do método envoltório	34
Figura 2 – Diagrama para o processo do método <i>filter</i>	35
Figura 3 – Diagrama para o processo do método embutido	36
Figura 4 – Exemplo do processo de treinamento. (a) Grafo ponderado completo derivado de um determinado conjunto de treinamento. (b) MST de um grafo completo. (c) Protótipos selecionados como sendo os elementos adjacentes de diferentes classes no MST (nós circulados). (d) Floresta de caminho ótimo resultante da aplicação da função de custo do caminho f_{max} com os dois protótipos. A seta indica o nó predecessor no caminho ótimo. As entradas (a, b) ao lado dos nós representam, respectivamente, o custo e o rótulo dos padrões.	39
Figura 5 – Exemplo do processo de classificação. O primeiro passo é (a) conectar (linhas tracejadas) um padrão de teste (quadrado cinza) com os nós de treinamento. Depois disso, (b) o caminho ótimo do protótipo mais fortemente conectado, seu rótulo 1 e o custo de classificação 0,6 são atribuídos ao padrão de teste.	41
Figura 6 – Formas do fluxo da água. (a) Fluxo laminar. (b) Fluxo turbulento. . .	53
Figura 7 – Fluxograma do WFO.	58
Figura 8 – Processo do operador laminar binário	63
Figura 9 – Processo do operador laminar binário de tamanho fixo	67
Figura 10 – Metodologia utilizada para os métodos propostos	76
Figura 11 – Resultados do OPF, PCA, LDA, BWWO e BBA versus BWFO com base na Tabela 5 para os conjuntos de dados PID, VC, ACA, BLD, BCW e DM.	80
Figura 12 – Resultados do OPF, PCA, LDA, BWWO e BBA versus BWFO com base na Tabela 5 para os conjuntos de dados SNR, DIV, PDS, TKM, UBL e ARR.	81
Figura 13 – Relação de redução de atributos e acurácia com base nos resultados da Tabela 8 para todos os métodos e os conjuntos de dados PID e VC. . .	84
Figura 14 – Relação de redução de atributos e acurácia com base nos resultados da Tabela 8 para todos os métodos e os conjuntos de dados ACA, BLD, BCW, DM, SNR, DIV, PDS e TKM.	85
Figura 15 – Relação de redução de atributos e acurácia com base nos resultados da Tabela 8 para todos os métodos e os conjuntos de dados UBL e ARR. .	86
Figura 16 – Gráfico de convergência do WFO para os conjuntos de dados.	99
Figura 17 – Gráfico de convergência do FSBWFO para os conjuntos de dados. . .	101

Lista de tabelas

Tabela 1 – Vantagens e desvantagens dos métodos de seleção de características com base na abordagem	37
Tabela 2 – Descrição dos conjuntos de dados utilizados no trabalho.	75
Tabela 3 – Resumo dos parâmetros empregados no trabalho.	77
Tabela 4 – Resumo dos parâmetros de cada método para os testes do BWFO. . .	78
Tabela 5 – Resultados do OPF, PCA, LDA, BCS, BWFO e BBA versus BWFO.	79
Tabela 6 – Média de tempos de classificação no conjunto Z_3 em segundos para os métodos avaliados e o BWFO.	81
Tabela 7 – Resumo dos parâmetros de cada método para os testes do FSBWFO. .	82
Tabela 8 – Resultados do OPF, PCA, LDA, BCS, BWFO e BBA versus FSBWFO.	83
Tabela 9 – Média de tempos de classificação no conjunto Z_3 em segundos para os métodos avaliados e o FSBWFO.	86
Tabela 10 – Resumo dos parâmetros de cada método para os testes do FPWBFO. .	87
Tabela 11 – Resultados do OPF, PCA, LDA, BCS, BWFO e BBA versus FPWBFO.	88
Tabela 12 – Média da soluções ótimas retornadas pelos métodos avaliadas sobre o conjunto Z_2 , levando em consideração a acurácia (%).	89
Tabela 13 – Média de tempos de classificação no conjunto Z_3 em segundos para os métodos avaliados e o FPWBFO.	90

Lista de Algoritmos

1	Algoritmo OPF	40
2	Algoritmo BWWO	46
3	Algoritmo BCS	48
4	Algoritmo BBA	51
5	Algoritmo WFO	59
6	Algoritmo BWFO	65
7	Algoritmo FSBWFO	69
8	Algoritmo FPBWFO	72

Lista de abreviaturas e siglas

WFO	<i>Water Flow Optimizer</i>
BWFO	<i>Binary Water Flow Optimizer</i>
FSBWFO	<i>Fixed Size Binary Water Flow Optimizer</i>
FPBWFO	<i>Feature Pruning Binary Water Flow Optimizer</i>
BCS	<i>Binary Cuckoo Search</i>
WWO	<i>Water Wave Optimization</i>
BBA	<i>Binary Bat Algorithm</i>
OPF	<i>Optimum-Path Forest</i>
PCA	<i>Principal Component Analysis</i>
LDA	<i>Linear Discriminant Analysis</i>

Lista de símbolos

$\mathbf{x}$	Vetor solução
$\mathbf{z}$	Vetor de atributos
y	Rótulo da classe de um padrão
$\vec{d}$	Direção comum determinada por duas partículas
p_l	Parâmetro de probabilidade laminar
p_e	Parâmetro de probabilidade de turbulência
ψ	Transformação de turbulência
φ	Operação de movimentação geral sobre camada
$\mathbf{Z}$	Conjunto de dados
$\mathbf{Z}'$	Conjunto de dados reduzido
$\mathbf{Z}_1$	Conjunto de treinamento
$\mathbf{Z}_2$	Conjunto de validação
$\mathbf{Z}_3$	Conjunto de teste
n	Número de padrões de um conjunto de dados
m	Número de atributos de um conjunto de dados
q	Número de atributos de um conjunto de dados reduzido
$\mathbf{P}$	População de indivíduos
$d(\mathbf{z}, \mathbf{t})$	Distância entre dois padrões $\mathbf{z}$ e $\mathbf{t}$
G_{tr}	Grafo completo a partir dos dados de treinamento
$\mathbf{S}$	Conjunto de protótipos
$f_{max}(\cdot)$	Função de custo de caminho
L	Mapa de rótulos obtido pelo modelo
F	Mapa de nós predecessores no grafo

C	Mapa de custos
Q	Fila de prioridade
$\mathbf{W}$	Matriz de projeção/transformação
$\mathbf{S}_W$	Matriz de dispersão entre classes
$\mathbf{S}_B$	Matriz de dispersão dentro de classes
μ	Vetor de médias dos atributos do conjunto de dados
μ_j	Vetor de médias dos atributos da j -ésima classe
λ	Autovalores
$\mathbf{V}$	Autovetores

Sumário

1	INTRODUÇÃO	27
1.1	Objetivos geral e específicos	30
1.2	Produção científica	31
1.3	Estrutura da dissertação	31
2	FUNDAMENTAÇÃO TEÓRICA	33
2.1	Seleção de características	33
2.2	Floresta de caminhos ótimos	37
2.2.1	Treinamento	38
2.2.2	Classificação	40
2.3	Análise de componentes principais	41
2.4	Análise discriminante linear	42
2.5	Meta-heurísticas para seleção de características	44
2.5.1	Otimização de ondas de água binário	44
2.5.1.1	Operador propagação	45
2.5.1.2	Operador refração	45
2.5.1.3	Operador quebra	45
2.5.2	Busca cuco binário	46
2.5.3	Algoritmo do morcego binário	48
2.6	Conclusão	52
3	OTIMIZADOR DE FLUXO DE ÁGUA	53
3.1	Fluxo de água na hidráulica	53
3.2	Modelagem matemática do otimizador de fluxo de água	54
3.2.1	Operador laminar	55
3.2.2	Operador turbulento	55
3.3	Algoritmo	57
3.3.1	Regra evolucionária	57
3.3.2	Iniciação e operadores	57
3.4	Conclusão	59
4	MÉTODOS PROPOSTOS	61
4.1	Ambiente do problema	61
4.2	Representação do problema	61
4.3	Iniciação e nova abordagem para os operadores	62
4.4	BWFO (Proposta 1)	62

4.4.1	Operador de fluxo laminar binário	62
4.4.2	Operador de fluxo turbulento binário	63
4.4.3	Função de aptidão	64
4.4.4	Framework BWFO	64
4.5	FSBWFO (Proposta 2)	65
4.5.1	Operador de fluxo laminar binário de tamanho fixo	66
4.5.2	Operador de fluxo turbulento binário de tamanho fixo	67
4.6	Função de aptidão	68
4.7	Framework FSBWFO	68
4.8	FPBWFO (Proposta 3)	70
4.8.1	Operador de fluxo laminar binário	70
4.8.2	Operador de fluxo turbulento binário	70
4.9	Função de aptidão	71
4.10	Framework FPBWFO	71
4.11	Conclusão	73
5	SIMULAÇÕES COMPUTACIONAIS	75
5.1	Metodologia	75
5.2	Resultados	77
5.3	BWFO (Proposta 1)	78
5.4	FSBWFO (Proposta 2)	82
5.5	FPBWFO (Proposta 3)	87
5.6	Conclusão	90
6	CONCLUSÕES E TRABALHOS FUTUROS	91
6.1	Contribuições deste trabalho	92
6.2	Trabalhos futuros	92
	REFERÊNCIAS	93
	APÊNDICES	97
	APÊNDICE A – GRÁFICOS DE CONVERGÊNCIA	99

1 Introdução

A aprendizagem de máquina tem desempenhado grande evolução no campo da inteligência artificial, principalmente em problemas de classificação de padrões e suas aplicações, dentre elas pode-se citar: reconhecimento de voz (MYKHAILICHENKO et al., 2022), imagens (MEKHA; TEEYASUKSAET, 2021), robótica (GONNOCHENKO et al., 2021) e classificação de doenças (TEZCAN; KIRAS; BILGIN, 2022). Devido à grande variedade de aplicações desenvolvidas por técnicas de aprendizagem de máquina, avanços significativos têm sido alcançados na área da ciência da computação e em outros campos do conhecimento científico e tecnológico.

Apesar dos avanços, nos últimos anos grande parte das empresas e organizações vêm armazenando grandes volumes de dados (*big data*) sem fazer qualquer uso potencial. Com isso, grandes volumes de dados de alta dimensionalidade surgiram em diversas aplicações de aprendizado de máquina. Entretanto, a alta dimensionalidade compromete tanto em requisitos de memória como no alto custo computacional durante a fase de treinamento, além de degradar a capacidade de generalização dos algoritmos devido ao que se denomina maldição da dimensionalidade (VERÓNICA; NOELIA; AMPARO, 2015).

Em vista disso, a redução de dimensionalidade surge como o processo de conversão de um conjunto de dados de alta dimensão para um conjunto de menor dimensão, de forma que as informações sejam mantidas de forma clara e similares. Estas técnicas são tradicionalmente utilizadas em problemas de aprendizagem de máquinas a fim de obter melhores atributos para tarefas de classificação ou regressão (RAY; ANALYTICS, 2015).

Em algumas aplicações, os atributos do conjunto de dados que não têm correlação com as classes (rótulos) servem como ruído e podem introduzir erros sistemáticos no preditor e reduzir o desempenho da classificação, isso pode acontecer quando há falta de informação sobre o processo em estudo. Aplicando técnicas de seleção de características, pode-se obter algumas informações sobre o processo, bem como melhorar o requisito computacional e o desempenho da predição (CHANDRASHEKAR; SAHIN, 2014). Nesse contexto, estratégias de redução de dimensionalidade podem ser empregadas para atenuar problemas relacionados a ruídos e o da alta dimensionalidade dos conjuntos de dados que podem afetar o preditor. O estudo de redução de dimensionalidade é relevante, como exemplo, pode-se citar os requisitos das aplicações nesta era da internet das coisas ¹ (*internet of things*, IoT), nos quais modelos de aprendizado de máquina com bom desempenho devem ser embarcados em dispositivos como micro-controladores. Percebe-se então a

¹ A internet das coisas (IoT) é uma rede de dispositivos incorporados a sensores, software e outras tecnologias com o objetivo de conectar e trocar dados com outros dispositivos e sistemas pela internet (ORACLE, 2022).

necessidade da realização de uma análise do conjunto de dados em questão, para melhorar o desempenho e reduzir o tamanho de tais modelos.

A redução de dimensionalidade pode ser feita de duas maneiras diferentes: via extração de características (*feature extraction*), em que a redução de dimensionalidade é realizada através da combinação ou transformação dos atributos originais. Dessa forma, essa técnica é capaz de gerar um conjunto de novos atributos, que geralmente são mais compactos e de maior poder discriminatório; e via seleção de características ² (*feature selection*), que busca efetuar a redução de dimensionalidade por remover atributos redundantes e irrelevantes do conjunto de dados (VERÓNICA; NOELIA; AMPARO, 2015; RASCHKA, 2015).

Métodos clássicos de redução de dimensionalidade como o análise de componentes principais (*principal component analysis*, PCA) e o análise discriminante linear (*linear discriminant analysis*, LDA) são bem conhecidos e amplamente utilizados para a tarefa de redução de dimensionalidade (BISHOP, 2006; TANG et al., 2014; RASCHKA, 2014). O PCA é uma técnica de transformação linear não supervisionada que é amplamente utilizada em diferentes campos, principalmente para a redução de dimensionalidade (RASCHKA, 2015). O PCA identifica padrões nos dados com base na correlação entre os atributos. Em suma, o PCA tem como objetivo encontrar as direções de máxima variação em dados de alta dimensão e projeta-los em um novo subespaço com dimensões iguais ou menores que a original (ABDI; WILLIAMS, 2010). O conceito geral por trás do LDA é muito semelhante ao PCA, enquanto o PCA tenta encontrar os eixos ortogonais do componente de máxima variância em um conjunto de dados; O objetivo do LDA é encontrar o subespaço de recursos que otimiza a separabilidade das classes. Tanto o LDA como o PCA são técnicas de transformação linear que podem ser usadas para reduzir o número de dimensões em um conjunto de dados (RASCHKA, 2015).

Dentre as técnicas citadas para redução de dimensionalidade, também destacam-se os métodos de seleção de características que visam encontrar um subconjunto de atributos relevantes de um determinado conjunto de dados, onde possivelmente reduz a complexidade computacional do modelo aprendido, melhora a sua eficácia e reduz o sobreajuste (PAPA; ROSA; PAPA, 2017). Diante disso, diversos trabalhos abordaram a tarefa de seleção de características modelando o problema como uma tarefa de otimização, utilizando meta-heurísticas para obter o subconjunto dos atributos mais significativos. Meta-heurísticas são estruturas algorítmicas, muitas vezes inspiradas na natureza, projetadas para solucionar problemas complexos de otimização (BIANCHI et al., 2008).

Com relação aos trabalhos que utilizaram meta-heurísticas na tarefa de seleção de

² No trabalho será utilizado o termo “seleção de características” para referir-se a seleção de atributos de um conjunto de dados. Apesar de ser comum a sua utilização, o termo “seleção de atributos” é mais preciso.

características, os autores Macêdo, Barbosa e Rocha Neto (2019), propuseram uma versão binária para a meta-heurística otimização de ondas de água (*water wave optimization*, WWO), chamada otimização de ondas de água binário (*binary water wave optimization*, BWWO), através da remodelagem dos operadores de propagação, operador de quebra da onda e refração para uma versão binária, obtendo bons resultados para a tarefa de seleção de características. O BWWO utilizou a taxa de acerto do classificador floresta de caminhos ótimos (*optimum-path forest*, OPF) para avaliar cada solução durante a fase de pesquisa do algoritmo. O grande destaque para o BWWO é que ele desempenha bem mesmo com baixos valores de tamanho populacional. Em Rosa, Papa e Papa (2017), os autores propuseram a meta-heurística programação genética baseada em semântica geométrica (*geometric semantic genetic programming*, GSGP), também no contexto de seleção de características. Eles demonstram em seus experimentos que o GSGP poderia desempenhar satisfatoriamente obtendo bons resultados. Nakamura et al. (2012) apresentam o algoritmo do morcego binário (*binary bat algorithm*, BBA) tendo também inspiração na natureza com base no comportamento dos morcegos. O BBA combina a velocidade de exploração dos morcegos juntamente com o classificador OPF para a tarefa de seleção de características. A busca pela solução ótima é dirigida pelo BBA enquanto o conjunto de dados reduzido é avaliado usando o classificador OPF, cujo o objetivo é maximizar o desempenho sobre tal conjunto de dados.

Já Prudhvi et al. (2018) combinaram as meta-heurísticas otimização de enxame de partículas com o algoritmo de busca gravitacional (*particle swarm optimization - gravitational search algorithm*, PSOGSA) para criar uma meta-heurística binária utilizando a rede de função de base radial (*radial basis function network*, RBFN) para essa atividade. O PSOGSA binário ajuda a escolher os melhores atributos com base na aptidão obtida do RBFN. A função de aptidão por sua vez foi projetada de modo a garantir um número mínimo de atributos com valores de sensibilidade e especificidade balanceadas. Rodrigues et al. (2013) propuseram um novo método para seleção de características baseado no algoritmo busca cuco (*cuckoo search*, CS), denominado busca cuco binário (*binary cuckoo search*, BCS), cujo o funcionamento é baseado no comportamento dos pássaros cuco. Os experimentos foram direcionados no contexto de detecção de roubo em sistemas de distribuição de energia em dois conjuntos de dados, os quais foram obtidos de uma empresa brasileira de energia elétrica. A função a ser maximizada pelo BCS é dada pela taxa de acerto do classificador OPF. Assim, pode-se perceber que o uso de meta-heurísticas tem se destacado como uma forma eficiente na atividade de seleção de características.

Recentemente, foi proposto uma nova meta-heurística para otimização global conhecida como otimizador de fluxo de água (*water flow optimizer*, WFO), tendo como inspiração dois tipos de fluxos de água existentes na natureza, são eles: fluxo laminar e o fluxo turbulento (LUO, 2021). O fluxo laminar tem por característica baixa velocidade e fluidez contínua, de forma que o fluido, passa a se movimentar de forma retilínea. Por sua

vez, o fluxo turbulento, tem por característica uma fluidez mais rápida, em que o fluido passa a ter movimentos ondulados, fluindo de forma irregular.

Em vista da ideia de redução de dimensionalidade e da utilização de meta-heurísticas nessa tarefa, como nos trabalhos citados anteriormente, o WFO pode ser utilizado para reduzir a dimensão dos conjuntos de dados, bem como selecionar os melhores atributos destes conjuntos. Com base nas versões binárias como meio de solucionar a tarefa de seleções de características e pelo sucesso que esses métodos vem obtendo com bons resultados na literatura, este trabalho visa propor novos métodos binários com base na meta-heurística WFO aplicada ao problema de seleção de características, visto ser um algoritmo recente, de bom desempenho, de poucos parâmetros e de apenas dois operadores. Além disso, Luo (2021) prova em seu trabalho a convergência do algoritmo e apresenta nas simulações computacionais resultados estatísticos significativos obtidos pela meta-heurística. Durante o processo de otimização dos novos métodos propostos o classificador OPF será utilizado para avaliar as soluções com base na sua taxa de acerto. O OPF possui vantagens interessantes, pois é um classificador simples e de fácil desenvolvimento, trata problemas com múltiplas classes sem necessidade de modificações, é independente de parâmetros e não faz nenhuma suposição sobre a forma das classes (PAPA; FALCÃO; SUZUKI, 2009).

Assim, este trabalho visa que os novos algoritmos binários sejam uma alternativa eficaz e efetiva na redução de dimensionalidade para conjuntos de dados de alta dimensão, sendo assim, capazes de melhorar a predição e o processamento sobre quaisquer dados.

1.1 Objetivos geral e específicos

Este trabalho tem como objetivo geral a realização de um estudo comparativo entre métodos de redução de dimensionalidade, visando a obtenção de modelos reduzidos, bem como adequar a meta-heurística WFO para problemas desta natureza.

Nesse contexto, os seguintes objetivos específicos são:

- apresentar novas propostas de versões binárias para a meta-heurística WFO aplicado na tarefa de seleção de características;
- apresentar um novo método de seleção de características de tamanho fixo de atributos e
- avaliar o desempenho dos métodos propostos em relação a outros métodos de redução de dimensionalidade presentes na literatura.

1.2 Produção científica

Durante este projeto de mestrado, os seguintes trabalhos científicos foram aceitos para publicação em conferências nacionais e internacionais, a saber:

- **MACÊDO, F.J.M.; ROCHA NETO, A.R.. Uma Versão Binária da Meta-heurística Water Flow Optimizer Aplicada à Seleção de Características.** In: XXIV Congresso Brasileiro de Automática - CBA 2022. Fortaleza, Brazil.
- **MACÊDO, F.J.M.; ROCHA NETO, A.R.. A Binary Water Flow Optimizer Applied To Feature Selection.** In: 23rd International Conference on Intelligent Data Engineering and Automated Learning - IDEAL 2022. Manchester, United Kingdom. Disponível em: <https://doi.org/10.1007/978-3-031-21753-1_10>

1.3 Estrutura da dissertação

A continuação deste projeto é composta por mais 6 capítulos, além deste introdutório, em que são apresentadas técnicas de redução de dimensionalidade propostas na literatura; o objetivo geral e específicos, e as produções científicas realizadas durante este projeto de mestrado.

O Capítulo 2 introduz os fundamentos teóricos necessários para o restante do trabalho, abordando a técnica de seleção de características e suas abordagens; os fundamentos teóricos do classificador OPF, apresentando os procedimentos de aprendizagem e classificação do OPF utilizando grafo completo e descrevendo os métodos de redução de dimensionalidade PCA, LDA, BWFO, BCS e BBA, que serão utilizados para como base de comparação com os métodos propostos. No Capítulo 3 é apresentada a meta-heurística WFO, seus operadores, bem como o seu funcionamento, visando o entendimento para uma melhor compreensão das contribuições do trabalho.

No Capítulo 4 são apresentadas as contribuições desta dissertação, contendo os métodos propostos: otimizador de fluxo de água binário (*binary water flow optimizer*, BWFO), otimizador de fluxo de água binário de tamanho fixo (*fixed size binary water flow optimizer*, FSBWFO) e o otimizador de fluxo de água binário com poda de características (*feature pruning binary water flow optimizer*, FPBWFO). Os métodos propostos são exemplificados tendo como base a tarefa de seleção de características.

O Capítulo 5 se concentra em validar a aplicação dos métodos propostos, ilustrando a metodologia, expondo os experimentos realizados e as análises dos resultados obtidos. Por fim, o capítulo 6 descreve as conclusões e contribuições do trabalho.

2 Fundamentação teórica

Neste capítulo são apresentados alguns fundamentos teóricos para a devida compreensão do restante do trabalho. Inicialmente, é exposto o problema de seleção de características, sua definição e categorias; na Seção 2.2 o classificador OPF será apresentado, bem como a sua construção e as fases de treinamento e classificação do classificador; em seguida, os métodos de redução de dimensionalidade PCA e LDA, respectivamente, serão descritos e, finalmente, as meta-heurísticas utilizadas para comparação com os métodos propostos serão apresentados.

2.1 Seleção de características

Segundo Verónica, Noelia e Amparo (2015), a seleção de características pode ser definida como o processo de seleção dos atributos mais relevantes e de descarte dos atributos redundantes e irrelevantes, tendo como objetivo obter um subconjunto de atributos que descreve adequadamente o problema a ser trabalhado, a fim de atenuar a degradação do desempenho.

Nesse contexto, a seleção de características tem como objetivo selecionar o subconjunto dos atributos mais representativos para as aplicações visando o aumento do desempenho do conjunto de dados. As técnicas de seleção de características são utilizadas pelas seguintes razões, segundo Banu e Ganesh (2015):

- a) remoção de dados irrelevantes, redundantes ou ruidosos;
- b) melhoria da qualidade dos dados;
- c) aumento da velocidade do processamento dos algoritmos de aprendizado de máquina;
- d) economia de recursos por meio da redução de atributos e
- e) aumento do desempenho do classificador.

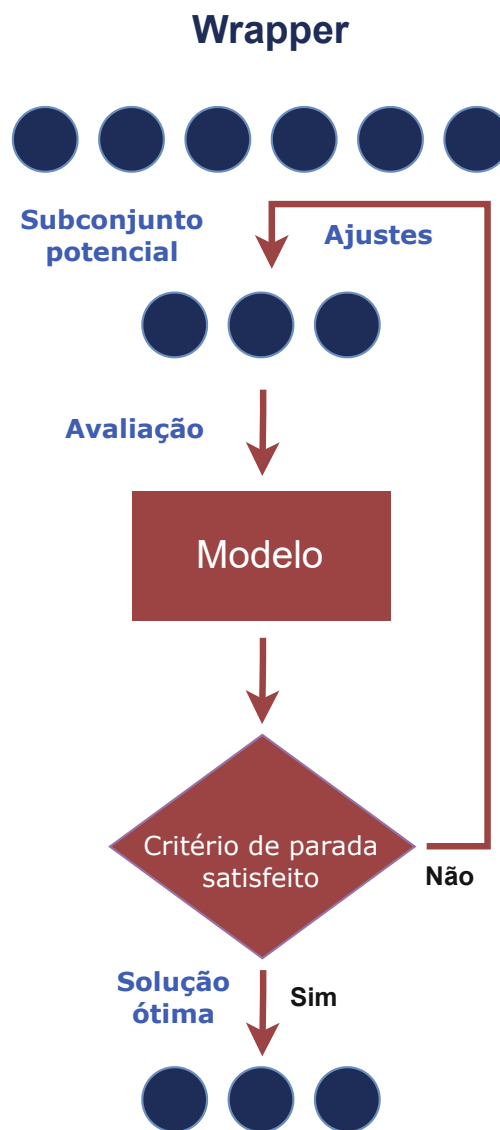
Os métodos de seleção de atributos tradicionais para aprendizado de máquina geralmente se enquadram em três categorias: modelo de filtro (*filter*), modelo envoltório (*wrapper*) e o modelo embutido (*embedded*) (BANU; GANESH, 2015).

A abordagem envoltório consiste de uma pesquisa heurística em um subespaço de todas as combinações possíveis de atributos. A qualidade dos atributos são obtidas por um algoritmo de aprendizado de máquina, sendo assim, a função de adequação para o problema (RODRIGUES et al., 2014). A abordagem é bastante complexa, no entanto, tem

por vantagem encontrar subconjuntos que por meio de classificadores alcançam melhor desempenho.

A Figura 1 apresenta a ideia geral da abordagem envoltório. O modelo envoltório deve fornecer uma combinação dos melhores atributos (solução), descobertos durante a fase de otimização de uma meta-heurística. Na abordagem, um conjunto de soluções deve ser fornecido na iniciação da meta-heurística para que o mesmo realize otimizações em cada solução, de forma que a cada iteração do algoritmo novas combinações de atributos sejam geradas a partir de cada solução candidata. Após os ajustes, as soluções candidatas são avaliadas, e caso algum critério de parada seja satisfeito a solução ótima é fornecida pela meta-heurística.

Figura 1 – Diagrama para o processo do método envoltório



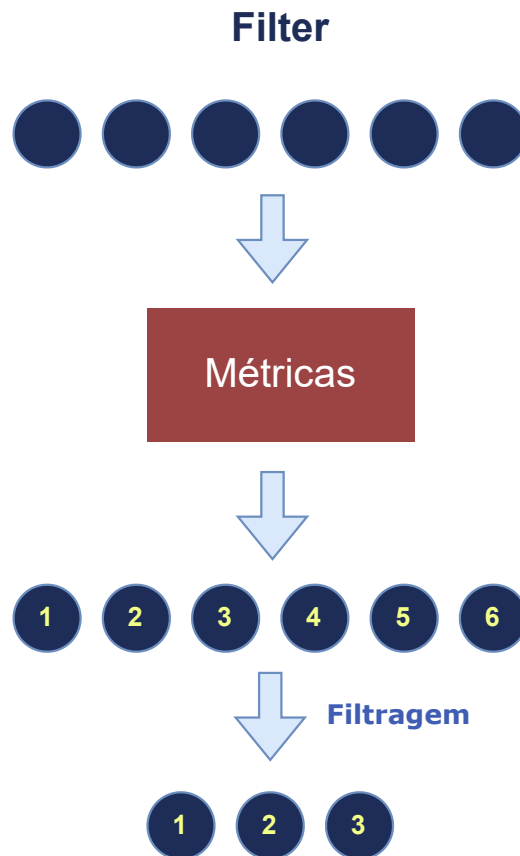
Fonte: próprio autor.

Os métodos de filtragem são baseados em métricas de avaliação de desempenho calculadas diretamente sobre os dados, a fim de avaliar os atributos por sua relevância,

sem utilizar qualquer tipo de métrica fornecida por classificadores, que por sua vez, são utilizados somente na avaliação do conjunto de dados com o número de atributos já reduzido (GUYON et al., 2006). Tal abordagem é geralmente menos cara computacionalmente do que os métodos envoltório e embutido. Alguns dos métodos de filtragem mais populares podem ser citados: qui-quadrado (LIU; SETIONO, 1995), ganho de informação (QUINLAN, 1986), seleção de características com base na correlação (HALL, 2000) e o filtro com base na consistência (DASH; LIU, 2003).

A Figura 2 exemplifica o processo dos métodos de filtragem. Neste método, a partir do conjunto de dados, são realizados cálculos de métricas, e, posteriormente, um ranqueamento e filtragem dos melhores atributos com base em sua relevância com valores obtidos a partir das métricas calculadas sobre os dados.

Figura 2 – Diagrama para o processo do método *filter*



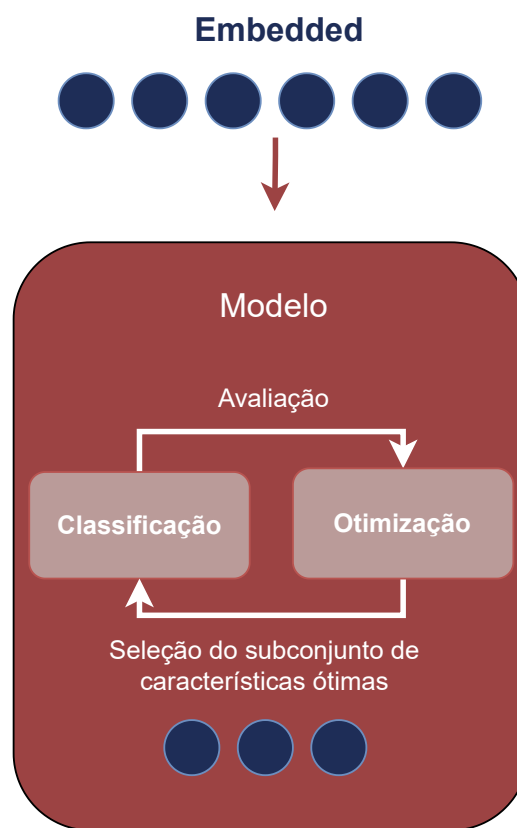
Fonte: próprio autor.

Já na abordagem embutido, a seleção de características é realizada no processo de treinamento e são geralmente específicos para determinadas máquinas de aprendizagem. Portanto, a busca por um subconjunto ótimo de atributos é realizada na construção do classificador e pode ser vista como uma pesquisa no espaço combinado de subconjuntos de características e hipóteses (VERÓNICA; NOELIA; AMPARO, 2015). Essa abordagem é capaz de fazer seleção de características com um custo computacional menor quando

comparados com os métodos de envoltório.

Os métodos embutido combinam as qualidades dos métodos de filtro e envoltório. Alguns dos exemplos mais populares desses métodos são a regressão *LASSO* e *RIDGE*, que possuem funções de penalização embutidas para reduzir o *overfitting* do preditor. A regressão *Lasso* realiza a regularização L1 que adiciona penalidade equivalente ao valor absoluto da magnitude dos coeficientes. A regressão *Ridge* realiza a regularização L2 que adiciona penalidade equivalente ao quadrado da magnitude dos coeficientes (KAUSHIK, 2016). A Figura 3 apresenta um ideia geral sobre o método embutido.

Figura 3 – Diagrama para o processo do método embutido



Fonte: próprio autor.

A Tabela 1 apresenta as vantagens e desvantagens dos métodos de filtro, envoltório e embutido para a tarefa de seleção de características, em que leva em conta parâmetros, bem como a dependência do método do classificador, custo computacional, capacidade de generalização, interação com o classificador e risco de sobreajuste.

Tabela 1 – Vantagens e desvantagens dos métodos de seleção de características com base na abordagem

Método	Vantagens	Desvantagens
<i>Filtro</i>	Independência do classificador.	Sem interação com o classificador.
	Menor custo computacional do que métodos envoltório.	
	Velocidade.	
	Boa capacidade de generalização.	
<i>Envoltório</i>	Interação com o classificador	Computacionalmente caro.
	Captura dependências de atributos.	Risco de overfitting.
		Seleção dependente do classificador.
<i>Embutido</i>	Interação com o classificador.	Seleção dependente do classificador.
	Menor custo computacional do que envoltório.	
	Captura dependências de recursos.	

Fonte: próprio autor.

2.2 Floresta de caminhos ótimos

O classificador floresta de caminhos ótimos (*optimum-path forest*, OPF) modela o problema de classificação de padrões como uma tarefa de partição de grafo, na qual o vetor de atributos extraído de cada padrão é considerado como um nó no grafo. Os arcos são definidos por alguma relação de adjacência predefinida e ponderada por uma função de distância aplicada aos seus vetores de atributos correspondentes.

Os elementos mais representativos de cada classe do conjunto de treinamento (protótipos), são os elementos pertencentes às regiões de fronteira entre as classes. A ideia principal é comandar um processo de competição entre os protótipos que oferecem aos outros nós um menor custo, e, conseqüentemente, seus rótulos. Esse processo particiona o grafo em uma coleção de árvores de caminhos ótimos, cada uma delas contendo os padrões mais fortemente conectados. A coleção dessas árvores geram uma floresta de caminhos ótimos, que assim dá o nome ao classificador. O processo de competição é realizado por meio de funções de custo de caminho oferecidas a cada padrão pelos protótipos. Vale notar que, diferentes funções de custo de caminho levam a diferentes partições no grafo.

Papa, Falcão e Suzuki (2009) e Papa e Falcão (2008) apresentaram duas diferentes versões supervisionadas do classificador OPF. A primeira (PAPA; FALCÃO; SUZUKI, 2009) usa como relação de adjacência um grafo completo que identifica os protótipos nos

limites das classes. Na segunda (PAPA; FERNANDES; FALCÃO, 2017; PAPA; FALCÃO, 2008), a relação de adjacência utilizada é um grafo com o método k -vizinhos mais próximos (k -nearest neighbors, KNN) e os protótipos são identificados como sendo os máximos de uma função de densidade de probabilidade, calculada utilizando os pesos dos arcos ².

2.2.1 Treinamento

Seja $\mathbf{Z}$ um conjunto de dados $\mathbf{y}$ -rotulado de tal modo que $\mathbf{Z} = \mathbf{Z}_1 \cup \mathbf{Z}_2 \cup \mathbf{Z}_3$, no qual $\mathbf{Z}_1$, $\mathbf{Z}_2$ e $\mathbf{Z}_3$ são conjuntos de treinamento, validação e teste, respectivamente. Além disso, considere $\mathbf{z} \in \mathbf{Z}$ um padrão m -dimensional mapeado para um nó no grafo. Seja $G_{tr} = (\mathbf{Z}_1, A)$ um grafo apresentado na Figura 4(a), derivado do conjunto de treinamento, com A sendo uma relação de adjacência que conecta todos os pares de nós (grafo completo). Considere um caminho π_z em G_{tr} terminando no nó $\mathbf{z}$ e uma função $f(\pi_z)$ que associa um valor de custo a esse caminho. O classificador OPF (PAPA; FALCÃO; SUZUKI, 2009) tenta minimizar a função $f(\pi_z)$ para cada padrão $\mathbf{z}$ usando uma função de custo de caminho suave que pode agrupar padrões com propriedades semelhantes (FALCÃO; STOLFI; LOTUFO, 2004), de tal maneira que:

$$f_{max}(\langle \mathbf{z} \rangle) = \begin{cases} 0 & \text{se } \mathbf{z} \in \mathbf{S}, \\ +\infty & \text{caso contrário} \end{cases} \quad (2.1)$$

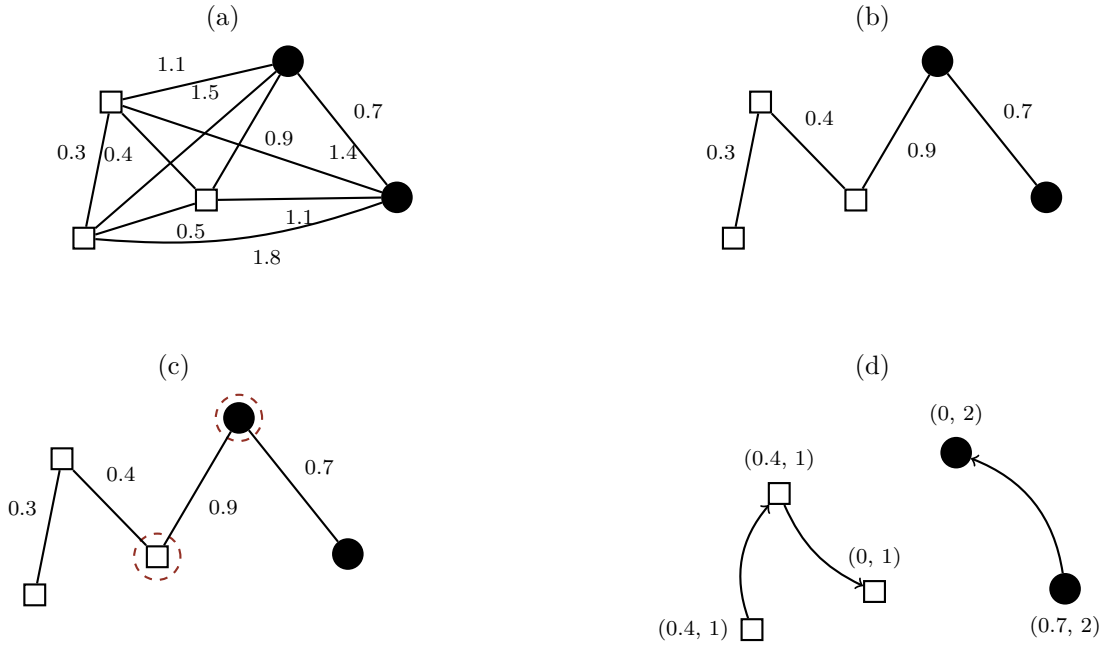
$$f_{max}(\pi \cdot \langle \mathbf{z}, \mathbf{t} \rangle) = \max\{f_{max}(\pi), d(\mathbf{z}, \mathbf{t})\},$$

em que $\pi \cdot \langle \mathbf{z}, \mathbf{t} \rangle$ denota a concatenação de um caminho π e um arco $(\mathbf{z}, \mathbf{t})$ e $d(\mathbf{z}, \mathbf{t})$ mede a distância entre os nós adjacentes. O conjunto $S \subseteq \mathbf{Z}_1$ refere-se aos protótipos.

Considere que $\mathbf{S}^* \subset \mathbf{S}$ é o conjunto de protótipos que levam ao processo de competição. Papa, Falcão e Suzuki (2009) propuseram selecionar os protótipos como sendo os nós de diferentes classes com a menor distância d entre qualquer outro, como vista na Figura 4(c). Isto pode ser obtido com base em uma árvore geradora mínima (*minimum-spanning tree*, MST) em G_{tr} , como apresentado na Figura 4(b). Observe que cada classe pode ser representada por vários protótipos e deve haver pelo menos um protótipo por classe. Os caminhos ótimos são calculados a partir dos protótipos para cada padrão de treinamento, de forma que cada protótipo se torne raiz de uma árvore de caminho ótimo composta por seus padrões mais fortemente conectadas (Figura 4(d)).

² Neste trabalho a abordagem empregada foi a que utiliza o grafo completo. Portanto, todas as referências subsequentes ao classificador OPF referem-se apenas a esta versão.

Figura 4 – Exemplo do processo de treinamento. (a) Grafo ponderado completo derivado de um determinado conjunto de treinamento. (b) MST de um grafo completo. (c) Protótipos selecionados como sendo os elementos adjacentes de diferentes classes no MST (nós circulados). (d) Floresta de caminho ótimo resultante da aplicação da função de custo do caminho f_{max} com os dois protótipos. A seta indica o nó predecessor no caminho ótimo. As entradas (a, b) ao lado dos nós representam, respectivamente, o custo e o rótulo dos padrões.



Fonte: Costa e Rocha Neto (2020)

O Algoritmo 1 apresenta os passos para a realização do treinamento do classificador OPF. Nas linhas 1-6 é realizada a iniciação dos mapas e a inserção dos protótipos na fila de prioridade Q . No laço principal, contido nas linhas 7-17, é obtido um caminho ótimo de cada protótipo $\mathbf{S}$ para cada padrão $\mathbf{z}$ em uma ordem não decrescente de custo mínimo.

Observe que em cada iteração o nó de menor custo da fila é removido, o que significa que tal nó já alcançou o menor custo que o mesmo pode oferecer para os demais nós. Na iteração contida nas linhas 9-17 é realizada a tentativa de $\mathbf{z}$ para conquistar os demais padrões. Assim na linha 10 é computado o valor de f_{max} , como descrito na Equação (2.1) para o padrão $\mathbf{z}$ e o padrão adjacente $\mathbf{t}$ na iteração atual, assim, se o novo custo computado for menor que o custo atual de $\mathbf{t}$ (veja a linha 11), $\mathbf{t}$ é removido da fila se seu custo for diferente de infinito e a seguir as posições de $\mathbf{t}$ são atualizadas em $F(t)$, $L(t)$, $C(t)$ e na fila $Q(t)$. Todos os nós adjacentes a $\mathbf{z}$ são avaliados com o intuito de serem conquistados pelo mesmo.

Algoritmo 1 Algoritmo OPF

Entrada: Conjunto de treinamento $\mathbf{Z}_1$ rotulado, protótipos $\mathbf{S} \subseteq \mathbf{Z}_1$, $d(\mathbf{z}, \mathbf{t})$ para cálculos de vetores de características e distâncias.

Saída: Floresta de caminhos ótimos F , mapa de custo C e mapa de rótulos L .

Auxiliar: Fila de prioridade Q e variável de custo cst .

```

1: para cada  $\mathbf{z} \in \mathbf{Z}_i \setminus \mathbf{S}$  faça
2:    $C(\mathbf{z}) \leftarrow +\infty$ 
3: para cada  $\mathbf{z} \in \mathbf{S}$  faça
4:    $C(\mathbf{z}) \leftarrow 0$ 
5:    $F(\mathbf{z}) \leftarrow \text{nulo}$ 
6:    $L(\mathbf{z}) \leftarrow Y(\mathbf{z})$ 
7:   Adicione  $\mathbf{z}$  em  $Q$ 
8: enquanto  $Q$  não está vazio faça
9:   Remova de  $Q$  um padrão  $\mathbf{z}$  tal que  $C(\mathbf{z})$  é mínimo.
10:  para cada  $\mathbf{t} \in \mathbf{Z}_1$  tal que  $\mathbf{t} \neq \mathbf{z}$  e  $C(\mathbf{t}) > C(\mathbf{z})$  faça
11:    Compute  $cst \leftarrow \max\{C(\mathbf{z}), d(\mathbf{z}, \mathbf{t})\}$ .
12:    se  $cst < C(\mathbf{t})$  então
13:      se  $C(\mathbf{t}) \neq +\infty$  então
14:        Remova  $\mathbf{t}$  de  $Q$ .
15:         $P(\mathbf{t}) \leftarrow \mathbf{z}$ 
16:         $L(\mathbf{t}) \leftarrow L(\mathbf{z})$ 
17:         $C(\mathbf{t}) \leftarrow cst$ 
18:        Adicione  $\mathbf{t}$  em  $Q$ .
```

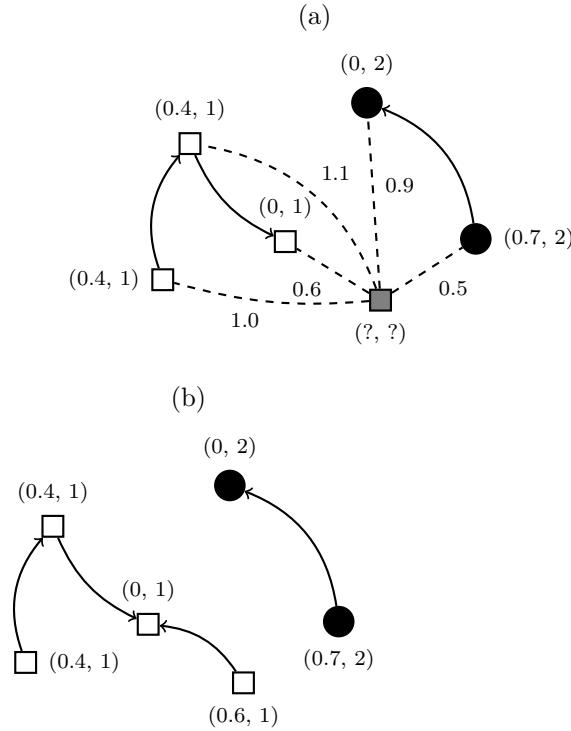
2.2.2 Classificação

Durante a fase de classificação, cada padrão $\mathbf{t} \in \mathbf{Z}_3$ é conectada a todas os padrões de $\mathbf{Z}_1$, como se $\mathbf{t}$ fizesse parte do grafo de treinamento, como descrito na Figura 5(a). O nó $\mathbf{t}$ recebe então um custo calculado como segue:

$$C(\mathbf{t}) = \operatorname{argmin}\{\max\{C(\mathbf{v}_1), d(\mathbf{v}_1, \mathbf{t})\}, \dots, \max\{C(\mathbf{v}_n), d(\mathbf{v}_n, \mathbf{t})\}\}, \mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n. \quad (2.2)$$

O processo de classificação atribui a $\mathbf{t}$ o rótulo do padrão $\mathbf{v}^*$ que satisfaz a Equação (2.2), como mostra a Figura 5(b). A fase de classificação, portanto, consiste em encontrar o nó $\mathbf{v}$ do conjunto de treinamento que minimiza $C(\mathbf{t})$.

Figura 5 – Exemplo do processo de classificação. O primeiro passo é (a) conectar (linhas tracejadas) um padrão de teste (quadrado cinza) com os nós de treinamento. Depois disso, (b) o caminho ótimo do protótipo mais fortemente conectado, seu rótulo 1 e o custo de classificação 0,6 são atribuídos ao padrão de teste.



Fonte: Costa e Rocha Neto (2020)

2.3 Análise de componentes principais

A análise de componentes principais (*principal component analysis*, PCA) é uma técnica de transformação linear não supervisionada amplamente utilizada em compressão de dados, extração de recursos, visualização de dados e principalmente na redução de dimensionalidade de conjuntos de dados (RASCHKA, 2015; JOLLIFFE, 2002). Com o PCA é possível extrair as informações mais importantes de um conjunto de dados e expressar essas novas informações como um conjunto de novas variáveis ortogonais chamadas de componentes principais (ABDI; WILLIAMS, 2010).

Desta forma, é praticável utilizar o PCA na tarefa redução de dimensionalidade, visto que, após encontrar os novos padrões é possível compactar os dados, reduzindo o número de dimensões sem muita perda de informação (SMITH, 2002). Os passos do algoritmo PCA para redução de dimensionalidade poder ser vista a seguir:

- normalize o conjunto de dados
- compute a matriz de covariância;
- compute os autovetores e autovalores a partir da matriz de covariância;
- selecione q autovetores (componentes principais) que correspondem aos q maiores

autovalores;

- construa a matriz de projeção $\mathbf{W}$ com os q componentes selecionados e
- obtenha o novo conjunto de dados transformando a partir do conjunto de dados de entrada $\mathbf{Z}$ usando a matriz de projeção $\mathbf{W}$.

Considerando um conjunto de dados $\mathbf{Z} \in \mathbb{R}^{n \times m}$ normalizado, o passo a seguir é calcular a matriz de covariância $\mathbf{A} \in \mathbb{R}^{m \times m}$. A covariância entre dois atributos x e y no nível da população pode ser calculada por meio da seguinte equação (RASCHKA, 2015):

$$\text{cov}(x, y) = \frac{1}{n} \sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y}) \quad (2.3)$$

em que, $\bar{x}$ e $\bar{y}$ são as médias amostrais dos atributos x e y , respectivamente. Por exemplo, uma matriz de covariância de três atributos x , y e z pode então ser escrita como:

$$\mathbf{A} = \begin{bmatrix} \text{cov}(x, x) & \text{cov}(x, y) & \text{cov}(x, z) \\ \text{cov}(y, x) & \text{cov}(y, y) & \text{cov}(y, z) \\ \text{cov}(z, x) & \text{cov}(z, y) & \text{cov}(z, z) \end{bmatrix} \quad (2.4)$$

Assim, considere $\mathbf{A} \in \mathbb{R}^{m \times m}$ uma matriz de covariância, $\mathbf{v}$ é um vetor e λ um escalar que satisfaz $\mathbf{A}\mathbf{v} = \lambda\mathbf{v}$, λ é chamado autovalor associado ao autovetor $\mathbf{v}$ de $\mathbf{A}$. Computando os autovalores e autovetores da matriz de covariância $\mathbf{A}$, os autovetores $\mathbf{V}$ da matriz de covariância representam as componentes principais (as direções de máxima variância), enquanto os autovalores $\boldsymbol{\lambda}$ correspondentes definirão sua magnitude. Uma vez que os autovalores definem a magnitude dos autovetores, ordenando os autovetores de forma decrescente com base em seus autovalores, seleciona-se os q primeiros autovetores para a obtenção da matriz de projeção $\mathbf{W} \in \mathbb{R}^{m \times q}$ (RASCHKA, 2015).

A fim de obter a transformação de um determinado conjunto de dados, basta que seja realizado uma multiplicação de matrizes, entre as informações do conjuntos de dados e a matriz de projeção $\mathbf{W}$, conforme a seguir:

$$\mathbf{Z}' = \mathbf{Z}\mathbf{W} \quad (2.5)$$

em que $\mathbf{Z}' \in \mathbb{R}^{n \times q}$ é o novo conjunto de dados reduzido e $\mathbf{Z}$ o conjunto de dados original.

2.4 Análise discriminante linear

A análise discriminante linear (*linear discriminant analysis*, LDA) é uma técnica comumente utilizada para problemas de redução de dimensionalidade como uma etapa de pré-processamento para aplicações de aprendizado de máquina e classificação de padrões (THARWAT et al., 2017). Portanto, o LDA pode ser utilizado para aumentar a eficiência

computacional e reduzir o grau de sobreajuste ² (*overfitting*) em modelos não regularizados devido à maldição da dimensionalidade (RASCHKA, 2015).

O conceito do LDA é muito semelhante ao do PCA, enquanto o PCA tenta encontrar os eixos das componentes ortogonais de máxima variância em um conjunto de dados, o LDA visa encontrar o subespaço de atributos que otimiza a separabilidade de classes (RASCHKA, 2015).

A redução de dimensionalidade de um conjunto de dados com o LDA pode ser realizada em 6 etapas:

- normalize o conjunto de dados;
- calcule os vetores médios para as diferentes classes do conjunto de dados;
- calcule as matrizes de dispersão (matrizes de dispersão entre classes e dentro de classes);
- calcule a matriz que maximiza a fórmula de Fisher e os autovetores e autovalores correspondentes;
- ordene de forma decrescente os autovetores com base em seus autovalores correspondentes e escolha q autovetores com os maiores autovalores e
- use esta matriz de autovetores para transformar as amostras no novo subespaço.

Considerando um conjunto de dados $\mathbf{Z} \in \mathbb{R}^{n \times m}$ normalizado, o passo a seguir é calcular as médias dos atributos de cada classe e a média de todos os dados. A média dos atributos por classes é descrita conforme a equação a seguir:

$$\boldsymbol{\mu}_j = \frac{1}{n_j} \sum_{\mathbf{z}_i \in \mathbf{y}_j}^{n_j} \mathbf{z}_i \quad (2.6)$$

em que $\boldsymbol{\mu}_j$ é o vetor de médias dos atributos da j -ésima classe; $\mathbf{y}_j$ a j -ésima classe e $\mathbf{z}_i$ e n_j são respectivamente o i -ésimo vetor de atributos e o número de padrões da classe. A média de todos os atributos do conjunto de dados segue conforme a equação abaixo:

$$\boldsymbol{\mu} = \frac{1}{n} \sum_{i=1}^n \mathbf{z}_i, \quad (2.7)$$

em que $\boldsymbol{\mu}$ é o vetor de média dos atributos; $\mathbf{z}_i$ é o i -ésimo atributo e n o número de atributos total do conjunto de dados.

De posse das médias, calcula-se as matrizes de dispersão entre classes, descrita pela seguinte equação:

² *overfitting* ou sobre-ajuste é um problema comum em aprendizado de máquina, em que o modelo captura bem os padrões nos dados de treinamento, mas falha em tentar generalizar para dados ainda não vistos (dados de teste) (RASCHKA, 2015).

$$\mathbf{S}_W = \sum_{j=1}^{|\mathbf{y}|} \sum_{i=1}^{n_j} (\mathbf{z}_i - \mu_j)(\mathbf{z}_i - \mu_j)^T \quad (2.8)$$

em que $\mathbf{S}_W \in \mathbb{R}^{m \times m}$ é a matriz de dispersão entre classes, $\mathbf{z}_i$ é o i -ésimo padrão da j -ésima classe e $|\mathbf{y}|$ o número de classes. Uma vez calculada a matriz $\mathbf{S}_W$, calcula-se a matriz de dispersão dentro de classes, como apresentado abaixo:

$$\mathbf{S}_B = \sum_{j=1}^{|\mathbf{y}|} n_j (\mu_j - \mu)(\mu_j - \mu)^T \quad (2.9)$$

em que $\mathbf{S}_B \in \mathbb{R}^{m \times m}$ é a matriz de dispersão dentro de classes e n_j é o número de padrões da j -ésima classe. A matriz $\mathbf{A}$ que maximiza a fórmula de Fisher é definida a seguir pela seguinte equação (THARWAT et al., 2017):

$$\mathbf{A} = \mathbf{S}_W^{-1} \mathbf{S}_B \quad (2.10)$$

Os autovalores λ e autovetores $\mathbf{V}$ de $\mathbf{A}$ são então calculados. De posse dos autovetores e autovalores, ordenando os autovetores em ordem decrescente de acordo com o valor de seus autovalores correspondentes, a matriz de projeção $\mathbf{W} \in \mathbb{R}^{m \times q}$ é construída com os q primeiros autovetores, sendo então utilizada para a construção de um novo conjunto de dados com um espaço dimensional inferior (THARWAT et al., 2017), descrito pela equação abaixo:

$$\mathbf{Z}' = \mathbf{Z}\mathbf{W} \quad (2.11)$$

em que $\mathbf{Z}' \in \mathbb{R}^{n \times q}$ é o novo conjunto de dados reduzido e $\mathbf{Z}$ o conjunto de dados original.

2.5 Meta-heurísticas para seleção de características

Meta-heurísticas são procedimentos de alto nível capazes de coordenar simples heurísticas e regras com o intuito de encontrar boas aproximações, ou até mesmo soluções ótimas, para difíceis problemas de otimização combinatória. Algumas meta-heurísticas de destaque são: recozimento simulado, busca tabu, algoritmos genéticos, colônia de formigas, entre outras. Elas são baseadas em paradigmas distintos e oferecem diferentes mecanismos para evitar ficar preso em mínimos locais do espaço de busca (BIANCHI et al., 2008; BLUM; ROLI, 2001).

2.5.1 Otimização de ondas de água binário

O algoritmo otimização de ondas de água binário (*binary water wave optimization*, BWWO) (MACÊDO; BARBOSA; ROCHA NETO, 2019) é um método evolutivo para

seleção de características inspirado no algoritmo de otimização de ondas de água (*water wave optimization*, WWO) (ZHENG, 2015). O BWWO utiliza as operações de propagação, refração e quebra da onda para evoluir a população de ondas (soluções) e encontrar a melhor combinação de atributos para o problema.

2.5.1.1 Operador propagação

O operador propagação realiza a seleção ou remoção dos atributos com base no comprimento de onda λ da solução. Assim, λ é um fator para a seleção de atributos sendo descrito conforme a equação a seguir:

$$\lambda = 1 - \alpha^{1-f(\mathbf{x})/f(\mathbf{x}^*)} \quad (2.12)$$

em que α é o coeficiente de comprimento de onda; e $f(\mathbf{x})$ e $f(\mathbf{x}^*)$ é aptidão da solução atual e a aptidão da melhor solução, respectivamente. O operador de propagação é empregado em $k = \{k \in \mathbb{Z}^+ | 1 \leq k \leq k_{max}\}$ componentes escolhidas aleatoriamente em que $k_{max} = \min(12, m/2)$, sendo m o número de componentes. Portanto, a modificação da componente é realizada com base em uma verificação considerando um número aleatório $r \in [0, 1]$, em que uma vez satisfeita a condição, o valor da k -ésima componente da i -ésima solução $\mathbf{x}_{i,k}$ tem seu valor invertido.

2.5.1.2 Operador refração

No operador de refração são escolhidas k componentes de forma aleatória em que cada componente da solução atual assume o valor da respectiva componente da melhor solução. Além disso, a altura da solução h_i é atualizada assumindo o valor parâmetro h_{max} , definido na iniciação do algoritmo.

2.5.1.3 Operador quebra

O operador de quebra, realiza uma espécie de mutação quando a solução atual supera a melhor solução na iteração, realizando inversões dos valores das k componentes selecionadas aleatoriamente. A cada mutação a solução é avaliada para verificar se a nova solução melhora sua aptidão com base em sua aptidão anterior.

O Algoritmo 2 descreve a meta-heurística BWWO. Na linha 1 são definidas as variáveis necessárias para a execução do algoritmo. Nas linhas 4-8, é realizada a operação de propagação, assim, novas soluções são geradas modificando as k componentes selecionadas de forma aleatória. O comprimento de onda λ é utilizado com um decisor para a seleção ou a remoção das componentes. Nas linhas 12-19 a operação de quebra realiza inversões nos valores das componentes selecionadas aleatoriamente, avaliando cada solução após as modificações.

Nas linhas 23-26 a operação de refração é responsável por alterar as soluções de baixa aptidão, assim, quando a altura h de uma solução após sucessivos decrementos chegar a zero, k componentes selecionadas aleatoriamente têm seus valores substituídos pelas respectivas componentes da melhor solução $\mathbf{x}^*$ na iteração. Na linha 27 o valor de λ é atualizado com base na Equação (2.12). Finalmente, quando o critério de parada do BWWO é satisfeito a melhor solução encontrada é fornecida.

A avaliação de cada solução é realizada pelo classificador OPF, assim, gera-se para cada solução um novo conjunto de dados a partir das componentes selecionadas pela solução, a taxa de acerto do classificador é a métrica utilizada como valor de aptidão.

Algoritmo 2 Algoritmo BWWO

Entrada: População P de n ondas (soluções); k_{max} e a altura máxima da onda h_{max} .

Saída: $\mathbf{x}^*$

```

1: Inicie um vetor de alturas  $h$ ; vetor de comprimento de ondas  $\lambda$  e um vetor de aptidões  $f$ .
2: enquanto o critério de parada não é satisfeito faça
3:   para cada onda  $\mathbf{x}_i$  ( $\forall_i = 1, \dots, n$ ) faça
4:      $\mathbf{x}'_i \leftarrow \mathbf{x}_i$ 
5:     Selecione  $k$  componentes aleatórias de  $\mathbf{x}_i$ .
6:     para cada componente  $k$  faça
7:       Gerar  $r$  distribuído uniformemente entre  $[0, 1]$ .
8:        $\mathbf{x}'_{i,k} \leftarrow$  se  $r > \lambda_i$  então 1 senão 0.
9:     se  $f(\mathbf{x}'_i) > f(\mathbf{x}_i)$  então
10:       $\mathbf{x}_i \leftarrow \mathbf{x}'_i$ ;
11:      se  $f(\mathbf{x}'_i) > f(\mathbf{x}_i^*)$  então
12:         $\mathbf{x}_i^* \leftarrow \mathbf{x}'_i$ 
13:        Selecione  $k$  componentes aleatórias.
14:        para cada componente  $k$  faça
15:           $\mathbf{x}'_{i,k} \leftarrow$  se  $\mathbf{x}'_{i,k} == 1$  então 0 senão 1
16:          se  $f(\mathbf{x}'_i) > f(\mathbf{x}_i)$  então
17:             $\mathbf{x}_i \leftarrow \mathbf{x}'_i$ 
18:            se  $f(\mathbf{x}'_i) > f(\mathbf{x}_i^*)$  então
19:               $\mathbf{x}_i^* \leftarrow \mathbf{x}'_i$ 
20:        senão
21:           $h_i \leftarrow h_i - 1$ 
22:          se  $h_i == 0$  então
23:            Selecione  $k$  componentes aleatórias de  $\mathbf{x}_i$ .
24:            para cada componente  $k$  faça
25:               $\mathbf{x}_{i,k} \leftarrow \mathbf{x}_{i,k}^*$ 
26:            Redefine  $h_i$  para  $h_{max}$ .
27:   Atualizar os comprimentos de onda conforme a Equação (2.12).
28: retorna  $\mathbf{x}^*$ 

```

2.5.2 Busca cuco binário

O algoritmo busca cuco binário (*binary cuckoo search*, BCS) (RODRIGUES et al., 2013) é uma versão binária da meta-heurística busca cuco (*cuckoo search*, BC) (YANG;

DEB, 2010) aplicado a tarefa de seleção de características. A ideia principal é associar a cada ninho (solução) um vetor binário que denota se um atributo pertence ou não a solução, sendo que a função a ser maximizada é dada pela precisão de um classificador supervisionado (RODRIGUES et al., 2013).

A versão original do CS pode ser resumida em três regras, como a seguir (RODRIGUES et al., 2013):

- cada cuco escolhe um ninho (solução) aleatoriamente para botar os ovos;
- o número de ninhos hospedeiros disponíveis é fixo, e ninhos com alta qualidade de ovos (componente do problema) serão transferidos para as próximas gerações e
- em casos em que uma ave hospedeira descobre o ovo de cuco, ela pode jogar o ovo fora ou abandonar o ninho e construir um ninho completamente novo.

Assim como o CS, o BCS utiliza o passeio aleatório voos de Lévy para atualizar as soluções, o passeio aleatório voos de Lévy é descrito nas equações a seguir:

$$\mathbf{x}_{i,j}(t) = \mathbf{x}_{i,j}(t-1) + \alpha \otimes Levy(\lambda) \quad (2.13)$$

$$Levy(\lambda) = s^{-\lambda}, (1 < \lambda \leq 3) \quad (2.14)$$

em que $\mathbf{x}_{i,j}(t)$, é a j -ésima componente (ovo) do i -ésimo ninho na t -ésima iteração; s é o tamanho do passo; $\alpha > 0$ é o fator/parâmetro de escala do tamanho do passo e $\otimes$ é um produto elemento a elemento. O BCS utiliza o parâmetro p_α com uma probabilidade para remover as soluções de menor qualidade, substituindo-os por novas soluções.

Visto o BCS ser um algoritmo binário, é provido as seguinte equações para que ocorra somente a geração de soluções binárias:

$$S(\mathbf{x}_{i,j}(t)) = \frac{1}{1 + e^{-\mathbf{x}_{i,j}(t)}} \quad (2.15)$$

$$\mathbf{x}_{i,j}(t) = \begin{cases} 1, & \text{se } S(\mathbf{x}_{i,j}(t)) > \sigma \\ 0, & \text{caso contrário} \end{cases} \quad (2.16)$$

em que $\sigma \sim U(0,1)$ e $S(\mathbf{x}_{i,j}(t))$ é a função sigmoide aplicado a j -ésima componente do ninho x_i na t -ésima iteração.

O Algoritmo 3 descreve o algoritmo BCS proposto para seleção de características. O algoritmo define na linha 1 as variáveis necessárias para a sua execução. Dentro do laço principal das linhas 2-20, mais precisamente nas linhas 3-6 é realizada as avaliações das soluções e sua atualização no vetor de aptidões. Nas linhas 8-10, o melhor valor de aptidão

e a solução ótima também são armazenadas em variáveis auxiliares. As linhas 11-13 são responsáveis por gerar com probabilidade p_α novas soluções para as piores soluções da iteração. As linhas 14-20 realizam as modificações nas soluções com base nas Equações 2.13 e 2.15. Finalmente, quando o número de iterações é alcançado ou algum outro critério de parada é satisfeito, o BCS fornece a melhor solução encontrada durante a fase de pesquisa do algoritmo.

O algoritmo BCS realiza as avaliações das soluções gerando um novo conjunto de treino e teste com base no valor da solução, a aptidão de cada solução é a taxa de acerto fornecida pelo classificador OPF após a avaliação do novo conjunto de dados gerado.

Algoritmo 3 Algoritmo BCS

Entrada: População P de n ninhos (soluções); a probabilidade remoção p_α e α .

Saída: $\mathbf{x}^*$

```

1: Inicie um vetor de taxa de acertos  $f$  com menos infinito; a solução ótima  $\mathbf{x}^*$  e o melhor
   fitness global  $f_{global}$ .
2: para cada  $t = 1$  até  $T$  faça
3:   para cada ninho  $\mathbf{x}_i$  ( $\forall i = 1, \dots, n$ ) faça
4:     Avalie o ninho  $\mathbf{x}_i$  e armazene a sua aptidão  $f'$  em  $f_i$ .
5:     se  $f' > f_i$  então
6:        $f_i \leftarrow f'$ 
7:        $\hat{\mathbf{x}}_{i,j} = \mathbf{x}_{i,j}$ 
8:   Selecione a maior aptidão  $f_{max}$  e o seu índice  $b$ 
9:   se  $f_{max} > f_{global}$  então
10:     $f_{global} \leftarrow f_{max}$ 
11:     $\mathbf{x}^* \leftarrow \mathbf{x}_b$ 
12:   para cada ninho  $\mathbf{x}_i$  ( $\forall i = 1, \dots, n$ ) faça
13:     para cada componente  $j$  ( $\forall j = 1, \dots, m$ ) faça
14:       Selecione os piores ninhos de acordo com  $p_\alpha$  e gere novas soluções.
15:   para cada ninho  $\mathbf{x}_i$  ( $\forall i = 1, \dots, n$ ) faça
16:     para cada componente  $j$  ( $\forall j = 1, \dots, m$ ) faça
17:        $\mathbf{x}_{i,j}(t) \leftarrow \mathbf{x}_{i,j}(t-1) + \alpha \otimes Levy(\lambda)$ 
18:       se  $\sigma < \frac{1}{1+e^{-\mathbf{x}_{i,j}(t)}}$  então
19:          $\mathbf{x}_{i,j}(t) \leftarrow 1$ 
20:       senão
21:          $\mathbf{x}_{i,j}(t) \leftarrow 1$ 

```

2.5.3 Algoritmo do morcego binário

O algoritmo do morcego binário (*binary bat algorithm*, BBA) (NAKAMURA et al., 2012) é uma técnica de seleção de características inspirada no comportamento dos morcegos. O algoritmo é uma versão binária da meta-heurística algoritmo do morcego (*bat algorithm*, BA) (YANG, 2012) que visa simular a ecolocalização dos morcegos e seu comportamento. A ecolocalização funciona como um sonar, em que quando um pulso alto e curto de som é emitido, caso o pulso atinja um objeto, em uma fração de tempo o

eco retorna aos ouvidos dos morcegos, servindo para localização de obstáculos e presas (NAKAMURA et al., 2012).

Com base no comportamento dos morcegos, o BA em sua versão original foi desenvolvido como uma técnica de otimização, visando simular o comportamento de um bando de morcegos rastreando presas ou alimentos com sua capacidade de ecolocalização (YANG, 2012). A modelagem do algoritmo segue algumas regras como descrita a seguir (YANG, 2012):

- todos os morcegos usam a ecolocalização para sentir a distância e distinguir presa e caçadores;
- um morcego b voa aleatoriamente com velocidade $\mathbf{v}_i$ na posição $\mathbf{x}_i$ e na frequência fixa f_{min} ; variando o comprimento de onda λ e a volume do som A_o para caça e podendo ajustar automaticamente o comprimento de onda e a taxa do pulso de emissão $r \in [0, 1]$ e
- o volume varia de um número grande positivo para uma constante mínima A_{min} .

O BBA segue o mesmo conceito do algoritmo BA em sua versão original, com pequenas alterações nas funções de movimentação, o movimento dos morcegos são atualizados com base na sua velocidade e na sua posição atual, seguindo as seguintes equações abaixo:

$$f_i = f_{min} + (f_{min} - f_{max})\beta, \quad (2.17)$$

$$\mathbf{v}_{i,j} = \mathbf{v}_{i,j} + (\mathbf{x}_j^* - \mathbf{x}_{i,j})f_i, \quad (2.18)$$

$$\mathbf{x}_{i,j} = \mathbf{x}_{i,j} + \mathbf{v}_{i,j}, \quad (2.19)$$

em que $\mathbf{v}_{i,j}$ é a velocidade do i -ésimo morcego na j -ésima componente; $\mathbf{x}_{i,j}$ é a posição do i -ésimo morcego na j -ésima componente; $\mathbf{x}_j^*$ é a j -ésima componente da melhor solução e $\beta \in [0, 1]$ um número gerado aleatoriamente.

Visando melhorar a variabilidade das soluções, Yang (2012) propôs passeios aleatórios aplicados com base na taxa de emissão de pulso r_i , gerando novas soluções a partir de uma solução selecionada aleatoriamente entre as melhores soluções da população, como descrito na equação a seguir:

$$\mathbf{x}' = \mathbf{x} + \epsilon \bar{A} \quad (2.20)$$

em que $\mathbf{x}'$ é o novo valor da solução; $\bar{A}$ é a média de todos os volumes da população de morcegos e $\epsilon \in [-1, 1]$ indica a direção e força do passeio aleatório. Em cada iteração o volume A_i e a taxa de emissão de pulso r_i são atualizados como descritos nas seguintes equações:

$$A_i = \alpha A_i \quad (2.21)$$

$$r + i = r_i(1 - e^{-\gamma t}) \quad (2.22)$$

em que α e γ são parâmetros que devem ser informados na iniciação do BBA.

Com o intuito de gerar soluções binárias, Nakamura et al. (2012) restringiu as posições dos morcegos utilizando a função sigmoide, como descrito nas equações abaixo:

$$S(v_{i,j}) = \frac{1}{1 + e^{-v_{i,j}}} \quad (2.23)$$

$$\mathbf{x}_{i,j} = \begin{cases} 1, & \text{se } S(v_{i,j}) > \sigma \\ 0 & \text{caso contrário} \end{cases} \quad (2.24)$$

em que $\sigma \sim U(0, 1)$ e $S(v_{i,j})$ é a função sigmoide aplicado a j -ésima componente da velocidade v_i . Assim, conforme a equação, as componentes da nova posição de cada solução assumem somente os valores 0 ou 1, com base na Equação (2.24).

O Algoritmo 4, descreve os passos da meta-heurística BBA. Na linha 1 é iniciada as variáveis necessárias para a execução do algoritmo. No primeiro laço das linhas 2-7 é iniciada as velocidades de cada morcego; as variáveis de taxa de emissão de pulso r_i ; volume os A_i de cada morcego e o vetor de aptidão f . Dentro do laço principal 9-41 é realizado os procedimentos para os passos de atualização de cada solução. Nas linhas 10-15, cada solução é avaliada e atualizado os seus valores de aptidão; volume e taxa de emissão de pulso, com base na condição da linha 12. Nas linhas 16-19 é realizado a verificação e o armazenamento do valor da melhor solução na iteração em $\mathbf{x}^*$ e a sua aptidão em f_{global} .

Dentro do laço 20-41, mas precisamente nas linhas 24-30, cada morcego tem sua posição atualizada, caso a verificação da linha 23 seja satisfeita, com base na taxa de emissão de pulso do atual morcego da iteração, utilizando a Equação (2.24). Ainda assim, caso a condição da linha 23 não seja satisfeita, é realizada outra verificação, mais precisamente na linha 32 com base no volume A_i do morcego e sua aptidão f_i , caso seja satisfeita, cada componente da solução será atualizada com base nas linhas 34-41, mas precisamente com base também na Equação (2.24), linhas 38-41.

O algoritmo BBA, assim como o BCS, realiza as avaliações de cada solução gerando um novo conjunto de treino e teste com base no valor da solução, e a aptidão da solução é a taxa de acerto fornecida pelo classificador OPF após a avaliação utilizando o novo conjunto de dados.

Algoritmo 4 Algoritmo BBA

Entrada: população P de n morcegos (soluções); sonoridade A ; taxa de emissão de pulso r ; e os parâmetros ϵ , α e γ .

Saída: $\mathbf{x}^*$

```

1: Inicie um vetor de taxa de acertos  $f$ ; um vetor de frequência inicial  $r^\circ$ ; a solução ótima  $\mathbf{x}^*$ ; o
   melhor fitness global  $f_{global}$ .
2: para cada morcego  $\mathbf{x}_i$  ( $\forall i = 1, \dots, n$ ) faça
3:   para cada componente  $j$  ( $\forall j = 1, \dots, m$ ) faça
4:      $v_{i,j} = 0$ 
5:      $A_i \leftarrow rand[1, 2]$ 
6:      $r_i \leftarrow rand[0, 1]$ 
7:      $f_i \leftarrow -\infty$ 
8:    $f_{global} \leftarrow -\infty$ 
9:   para cada  $t = 1$  até  $T$  faça
10:    para cada morcego  $\mathbf{x}_i$  ( $\forall i = 1, \dots, n$ ) faça
11:      Avalie o morcego  $\mathbf{x}_i$  e armazene sua aptidão em  $acc$ .
12:      se  $rand < A_i$  e  $acc > f_i$  então
13:         $f_i \leftarrow acc$ 
14:         $A_i \leftarrow \alpha A_i$ 
15:         $r_i \leftarrow r_i^\circ [1 - e^{-\gamma t}]$ 
16:      Selecione a maior aptidão  $f_{max}$  e o seu índice  $b$ 
17:      se  $f_{max} > f_{global}$  então
18:         $f_{global} \leftarrow f_{max}$ 
19:         $\mathbf{x}^* \leftarrow \mathbf{x}_b$ 
20:    para cada morcego  $\mathbf{x}_i$  ( $\forall i = 1, \dots, n$ ) faça
21:       $\beta \leftarrow rand[0, 1]$ 
22:       $rnd \leftarrow rand[0, 1]$ 
23:      se  $rnd > r_i$  então
24:        para cada componente  $j$  ( $\forall j = 1, \dots, m$ ) faça
25:           $\mathbf{x}_{i,j} = \mathbf{x}_{i,j} + \epsilon A$ 
26:           $\sigma \leftarrow rand\{0, 1\}$ 
27:          se  $\sigma < \frac{1}{1 + e^{-\mathbf{x}_{i,j}}}$  então
28:             $\mathbf{x}_{i,j} \leftarrow 1$ 
29:          senão
30:             $\mathbf{x}_{i,j} \leftarrow 0$ 
31:         $rnd \leftarrow rand[0, 1]$ 
32:        se  $rnd < A_i$  e  $f_i < f_{global}$  então
33:          para cada componente  $j$  ( $\forall j = 1, \dots, m$ ) faça
34:             $f_i \leftarrow f_{min} + (f_{max} - f_{min})\beta$ 
35:             $v_{i,j} \leftarrow v_{i,j} + (\mathbf{x}_j^* - \mathbf{x}_{i,j})$ 
36:             $\mathbf{x}_{i,j} = \mathbf{x}_{i,j} + v_{i,j}$ 
37:             $\sigma \leftarrow rand\{0, 1\}$ 
38:            se  $\sigma < \frac{1}{1 + e^{-\mathbf{x}_{i,j}}}$  então
39:               $\mathbf{x}_{i,j} \leftarrow 1$ 
40:            senão
41:               $\mathbf{x}_{i,j} \leftarrow 0$ 

```

2.6 Conclusão

Neste capítulo foram apresentados os fundamentos teóricos necessários para o restante do trabalho. Com base na seção de seleção de características, o trabalho seguirá a abordagem envoltório, visto que este trabalho propõe métodos de seleção de características com base na meta-heurística WFO, que deve guiar o conjunto de soluções candidatas a sua evolução em busca da solução ótima. O classificador OPF também apresentado neste capítulo é o responsável por realizar as avaliações de cada solução candidata a cada iteração no WFO, além de ser utilizado para os demais métodos de redução de dimensionalidade também apresentados neste capítulo, que serão avaliados juntamente com os novos métodos.

Os métodos PCA e LDA, além das meta-heurísticas BCS, BWWO e BBA, foram os métodos escolhidos para realizar a comparação com os métodos propostos, visto a sua importância na literatura. O WFO será apresentado com mais detalhes no capítulo a seguir, visto a sua importância para o trabalho.

3 Otimizador de fluxo de água

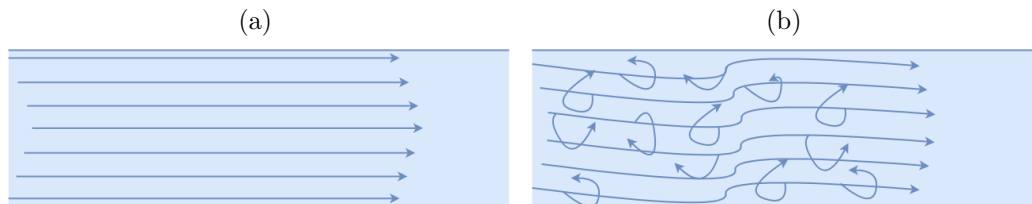
Este capítulo apresenta a meta-heurística otimizador de fluxo de água (*water flow optimizer*, WFO), tendo por objetivo descrever o seu funcionamento, sua estrutura e seus operadores, visando um melhor entendimento dos métodos propostos neste trabalho.

3.1 Fluxo de água na hidráulica

O cientista britânico Osborne Reynolds, em 1883 conduziu experimentos verificando o comportamento do fluxo de um filete de líquido colorido através de um tubo. No experimento, o filete colorido com a mesma densidade da água foi injetada em um tubo de vidro transparente com uma válvula na extremidade. Reynolds observou que quando a válvula foi aberta lentamente, a velocidade do fluxo de água no tubo foi pequena e o líquido colorido fluíu em linha reta por todo o comprimento do tubo. No entanto, a medida que a válvula foi aberta, houve um aumento gradual da velocidade e o fluxo do líquido tomou formas irregulares, tornado-se primeiramente ondulado e depois surgindo vórtices. Este fenômeno regular, ocorrido pelo escoamento de menor velocidade é definido como fluxo laminar na hidráulica. Por sua vez, o fenômeno irregular é definido como fluxo turbulento (ABBOTT; MINNS, 2017).

Em suma, o fluxo laminar ocorre quando a velocidade do fluxo de água é pequeno, de forma que as partículas de água distribuídas em diferentes camadas se movem regularmente em linhas retas paralelas, como exemplificado na Figura 6(a). Já o fluxo turbulento, ocorre quando a velocidade de água é grande, fazendo com que as partículas se choquem de maneira estocástica e se movam sobre as demais camadas, como pode ser visto na Figura 6(b).

Figura 6 – Formas do fluxo da água. (a) Fluxo laminar. (b) Fluxo turbulento.



Fonte: próprio autor.

Osborne Reynold descobriu que a forma do fluxo depende principalmente da relação entre as forças inerciais e as forças viscosas do fluido (ABBOTT; MINNS, 2017). Buscando identificar os tipos de escoamento, Reynolds relacionou propriedades do fluido, a saber: massa específica, viscosidade, como também a geometria do tubo e a velocidade do fluxo,

criando assim, o número de Reynolds dada pela seguinte relação para tubos circulares, a saber:

$$Re = \frac{\rho V D}{\mu} = \frac{V D}{\nu} \quad (3.1)$$

em que V é a velocidade média do fluido, D o diâmetro para o fluxo no tubo, μ a viscosidade dinâmica do fluido, ρ a massa específica do fluido e ν a viscosidade cinemática do fluido.

Assim, o número de Reynolds indica que a velocidade não é o único fator que determina se o fluxo é laminar ou turbulento. Por exemplo, durante as movimentações das camadas adjacentes, a força de atrito entre a camada mais lenta tenta reduzir a velocidade da camada mais rápida. Tal resistência, chamada de viscosidade indica a capacidade do fluido em resistir a cisalhamentos internos das partículas em movimento entre camadas.

Em geral, pode-se determinar se o fluxo é laminar ou turbulento pelo número de Reynolds, em que quando as forças inerciais são grandes em relação as forças viscosas (Re alto), as forças viscosas não conseguem evitar o comportamento aleatório e rápido dos fluídos (fluxo turbulento). Quando as forças viscosas são suficientemente grandes em relação as forças inerciais (Re baixo), há uma resistência maior ao comportamento aleatório e rápido (fluxo laminar).

3.2 Modelagem matemática do otimizador de fluxo de água

O algoritmo Otimizador de Fluxo de Água (*Water Flow Optimizer*, WFO) (LUO, 2021) é um método evolutivo baseada no fluxo da água para solucionar problemas de otimização. No algoritmo, assim como nos demais algoritmos evolutivos, a solução ótima pode ser encontrada por um processo iterativo buscando a cada iteração a evolução de suas soluções, sendo tais soluções avaliadas por uma função objetivo. Sem perda de generalidade, suponha que o problema tenha uma função objetivo a ser minimizada, a solução ótima, na analogia com o fluxo de água, é uma solução que no processo pode iniciar com um alto valor de função objetivo para um baixo valor, uma vez que o fluxo de água parte de uma fonte de maior altitude para uma de menor altitude na natureza.

Nesse algoritmo, cada partícula de água é uma solução e o valor dela é dado por sua posição, enquanto a energia potencial da partícula representa o valor de aptidão. Assim, a função objetivo é utilizada para avaliar a energia potencial de cada partícula no espaço de soluções. O fluxo laminar e o fluxo turbulento presente na mecânica dos fluídos servem de inspiração para os operadores do WFO, os quais foram projetados para realizar buscas estocásticas regulares e irregulares no espaço de soluções, no processo de otimização (LUO, 2021). Durante o processo de otimização, as operações de fluxo laminar e de fluxo turbulento são aplicadas para evoluir a população.

3.2.1 Operador laminar

No fluxo laminar, todas as partículas se movem em linhas retas paralelas. Devido à viscosidade da água, a velocidade das partículas podem ser diferentes, por exemplo, as partículas em camadas próximas de paredes ou obstáculos perdem velocidade em relação as partículas que estão mais afastadas das camadas com esses tipos de percalços. O movimento regular denotado pelo fluxo laminar é modelado no otimizador de fluxo de água por:

$$\mathbf{x}'_i(t) = \mathbf{x}_i(t) + s_i \vec{d}, \quad \forall i \in \{1, 2, \dots, n\} \quad (3.2)$$

$$\vec{d} = \mathbf{x}_b(t) - \mathbf{x}_k(t), \quad (b \neq k, \quad f(\mathbf{x}_b(t)) \leq f(\mathbf{x}_k(t))), \quad (3.3)$$

em que t é o número da iteração, i é o índice da partícula na população de soluções, n é o tamanho da população de partículas de água empregadas, $\mathbf{x}_i(t)$ é a posição da i -ésima partícula na t -ésima iteração, $\mathbf{x}'_i(t)$ denota sua possível posição de movimento, o número aleatório $s_i \sim U(0, 1)$ representa o coeficiente de deslocamento de uma partícula de água e o vetor $\vec{d}$ indica uma direção de movimento comum para todas as partículas.

A direção comum é determinada por duas partículas distintas selecionadas, em que $\mathbf{x}_b(t)$ é a melhor solução encontrada na t -ésima iteração, e $\mathbf{x}_k(t)$, é uma partícula selecionada aleatoriamente na t -ésima iteração, tal que, considerando um problema de minimização, a aptidão da melhor solução $\mathbf{x}_b(t)$ deve ser menor que a aptidão da partícula escolhida aleatoriamente $\mathbf{x}_k(t)$, sendo b e k , os índices dessas partículas, os quais devem ser obrigatoriamente diferentes. A melhor solução encontrada até o momento é selecionada como partícula guia na implementação do operador laminar.

O valor de $\vec{d}$ na Equação (3.2), permanece inalterado para todas as partículas da iteração, já o valor de s_i varia de partícula para partícula. O operador laminar no algoritmo WFO adota um padrão de pesquisa unidirecional paralela regular, em que $\vec{d}$, visa a busca unidirecional paralela, e a aleatoriedade de s_i garante que as diferentes partículas de água tenham deslocamentos distintos.

3.2.2 Operador turbulento

No fluxo turbulento, há um movimento irregular das partículas de água, o que faz com que o fluxo retilíneo seja interrompido. Devido ao rápido fluxo de água, as partículas se chocam de maneira aleatória, se movem sobre as camadas adjacentes, visto que a amplitude da oscilação é grande o suficiente a força de cisalhamento, faz com que as partículas água girem, formando redemoinhos (LUO, 2021). No operador de fluxo turbulento, uma componente de solução do problema é alterada para simular uma mudança de camada no fluxo de água. Assim, o operador simula o fenômeno turbulento por realizar

uma transformação entre as componentes, a fim de produzir um movimento irregular nas partículas de água. A movimentação da partícula é produzida por oscilar uma componente como segue:

$$x'_i(t) = (... , \mathbf{x}_{i,j_1-1}(t), u, \mathbf{x}_{i,j_1+1}(t), ...) \quad (3.4)$$

em que $j_1 \in \{1, 2, ..., m\}$ é uma componente do vetor solução selecionada aleatoriamente de m componentes, enquanto o valor de mutação u apresentando na Equação (3.4) é determinado pela seguinte equação:

$$u = \begin{cases} \psi(\mathbf{x}_{i,j_1}(t), \mathbf{x}_{k,j_1}(t)), & \text{se } r < p_e \\ \varphi(\mathbf{x}_{i,j_1}(t), \mathbf{x}_{k,j_2}(t)), & \text{caso contrário} \end{cases} \quad (3.5)$$

em que $k \in \{1, 2, ..., n\}$ é o índice de uma partícula distinta selecionada aleatoriamente, tal que $k \neq i$; j_2 é uma componente distinta selecionada aleatoriamente, tal que $j_2 \neq j_1$, r é um número aleatório no intervalo $[0, 1]$ e $p_e \sim U(0, 1)$ é um parâmetro de controle denominado probabilidade de turbulência.

O operador de fluxo turbulento utiliza uma transformação de turbulência ψ com probabilidade p_e e uma operação de movimentação geral sobre camada φ com probabilidade $1 - p_e$. A transformação de turbulência ψ é construída com base em uma espiral arquimediana, pela semelhança com um redemoinho, dada por:

$$\psi(\mathbf{x}_{t,j_1}(t), \mathbf{x}_{k,j_1}(t)) = \mathbf{x}_{i,j_1}(t) + \rho * \theta * \cos(\theta) \quad (3.6)$$

em que θ é um número aleatório dentro do intervalo $[-\pi, \pi]$ e ρ é determinado dinamicamente por uma regra auto-adaptativa.

$$\rho = |\mathbf{x}_{i,j_1}(t) - \mathbf{x}_{k,j_1}(t)| \quad (3.7)$$

e portanto, ρ pode ser visto como uma força de cisalhamento da k -ésima para a i -ésima partícula. Além disto, também é interessante usar a função seno como descrito na Equação (3.6) uma vez que $\sin(\theta) = \cos(\theta - [\pi/2])$.

Vale ressaltar ainda que a função φ , por sua vez, aplicada com probabilidade $1 - p_e$, visa simular o comportamento geral das partículas sobre as camadas com base na seguinte função de transformação:

$$\varphi(\mathbf{x}_{i,j_1}(t), \mathbf{x}_{k,j_2}(t)) = (ub^{j_1} - lb^{j_1}) * \frac{\mathbf{x}_{k,j_2}(t) - lb^{j_2}}{ub^{j_2} - lb^{j_2}} + lb^{j_1} \quad (3.8)$$

em que lb e ub , respectivamente, representam os limites inferior e superior do problema de otimização.

3.3 Algoritmo

3.3.1 Regra evolucionária

A evolução das partículas de água no WFO, é alcançado através dos operadores de fluxo laminar e fluxo turbulento. Os fluxos ocorrem com base em um limiar, denominado número de Reynolds, de tal maneira que, se o fluxo de água for menor que tal limiar, o fluxo é laminar; caso contrário, o fluxo passa a ser turbulento. Esses operadores são simulados de forma estocástica, o operador laminar é executado com probabilidade p_l , já o operador turbulento, por sua vez é executado com probabilidade de $1 - p_l$. O parâmetro p_l , no algoritmo se comporta como um limiar, em que com base no seu valor é que se determina a operação a ser executada (LUO, 2021).

No final de cada iteração, após a aplicação de um dos operadores do WFO, cada partícula de água deve ser avaliada, assim, partículas que realizaram boas movimentações dentro do espaço de pesquisa conseguem melhorar sua energia potencial (ou seja, melhorar sua aptidão). No entanto, partículas que não realizaram boas movimentações seguem para a próxima iteração com sua posição anterior dentro do espaço de busca. A regra para atualização das partículas para a próxima iteração é apresentada a seguir:

$$\mathbf{x}_i(t+1) = \begin{cases} \mathbf{x}'_i(t), & \text{se } f(\mathbf{x}'_i(t)) < f(\mathbf{x}_i(t)) \\ \mathbf{x}_i(t), & \text{caso contrário.} \end{cases} \quad (3.9)$$

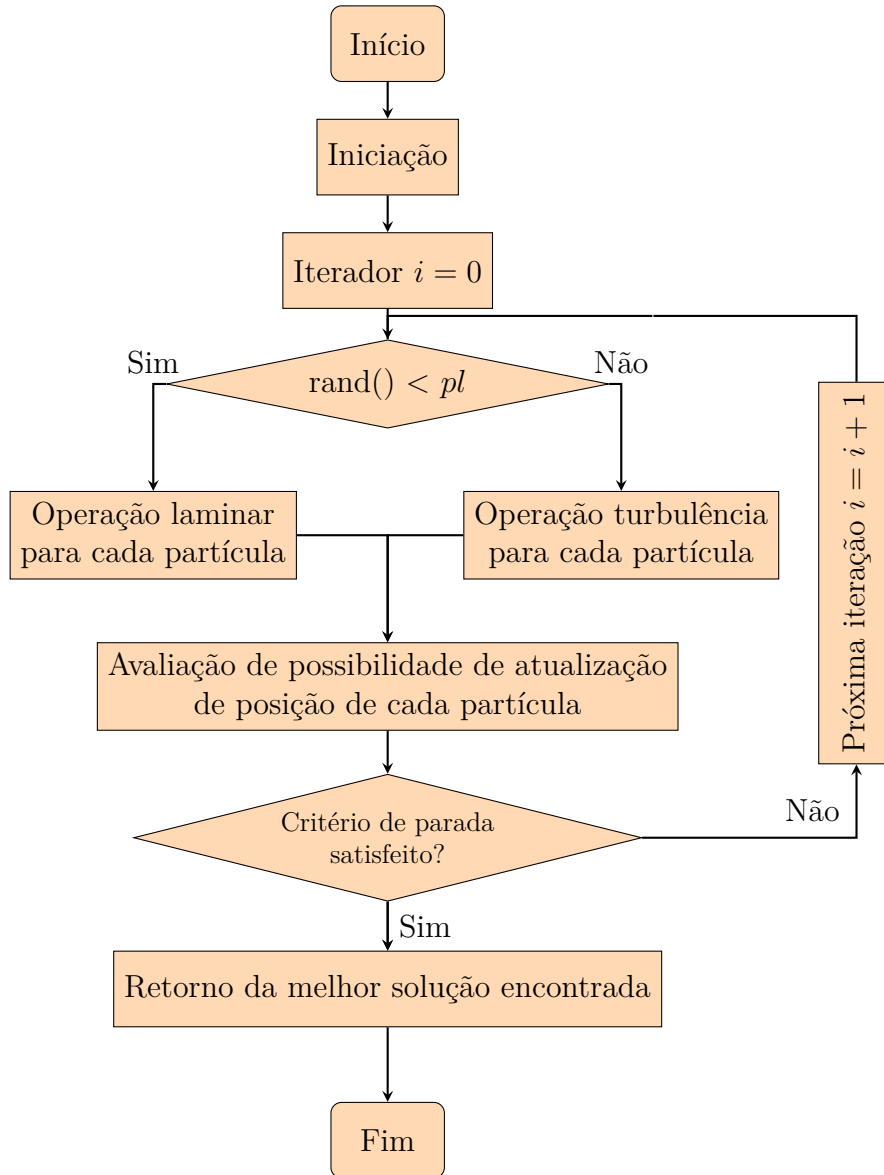
O fluxograma da Figura 7 descreve resumidamente as etapas do algoritmo.

3.3.2 Iniciação e operadores

O algoritmo WFO tem como parâmetros os valores para as probabilidade laminar p_l e probabilidade de turbulência p_e e, portanto devem ser definidos antes do processo de otimização. Além das definições de tais parâmetros, é necessário que as posições e a energia de cada partícula seja computada. Luo (2021), em seus experimentos contra os demais algoritmos evolutivos, fixou em seus testes valores como: $p_l = 0,3$, $p_e = 0,7$ e o tamanho da população $n = 50$. A posição da partícula por sua vez é iniciada aleatoriamente dentro do espaço de busca e, após isso, a energia potencial de cada partícula é calculada diretamente utilizando a função objetivo.

Em resumo, o operador de fluxo laminar faz com que as partículas de água movam-se de forma serena em direção a solução ótima, o operador de fluxo turbulento intensifica a busca próximo a região em que a partícula se encontra dentro do espaço de busca. A probabilidade de turbulência p_e é relativamente propenso a impactar no desempenho do algoritmo, visto que para valores muito baixos, a chance de operação de movimentação sobre camadas aumentam e, conseqüentemente, pode ocorrer um aumento do risco de

Figura 7 – Fluxograma do WFO.



Fonte: próprio autor.

convergência prematura. Vale salientar que no operador de fluxo laminar o valor de s_i , como pode ser observado no Algoritmo 5, na linha 9, é distinto para cada partícula, mas idêntico para cada componente de posição (componente) da mesma partícula. Já o valor de $\vec{d}$, linha 7, pode ser diferente em diferentes iterações, mas idêntico para cada partícula de água durante a mesma iteração. É bom observar que no computar da direção $\vec{d}$, a melhor partícula na iteração, a partícula h , funciona com um guia, para acelerar a taxa de convergência.

Além disso, Luo (2021) utilizou um esquema de retenção para lidar com a violação de restrição de limite, para que o algoritmo não compute movimentos de partículas fora do espaço de busca. Mais especificamente, $\mathbf{x}'_{i,j}(t) = \mathbf{x}_{i,j}(t)$, se $\mathbf{x}'_{i,j}(t) > ub^j$ ou $\mathbf{x}'_{i,j}(t) < lb^j$. Deve-se notar que o valor da solução fornecido pela Equação (3.8) não excede nenhum

limite. Entretanto não há violação de restrição em soluções geradas pela Equação (3.8).

Algoritmo 5 Algoritmo WFO

```

1: para cada partícula  $i \in \{1, 2, \dots, n\}$  faça
2:   para cada componente  $j \in \{1, 2, \dots, m\}$  faça
3:     Gere aleatoriamente uma posição  $\mathbf{x}_{i,j} \leftarrow lb^j + rand() * (ub^j - lb^j)$ 
4:     Calcule a energia potencial,  $f_i \leftarrow f(\mathbf{x}_i)$ 
5: enquanto o critério de parado não é satisfeito faça
6:   se  $rand() < pl$  então
7:     Determine uma direção laminar,  $\vec{d} \leftarrow \mathbf{x}_b - \mathbf{x}_k$ 
8:     para cada partícula  $i \in \{1, 2, \dots, n\}$  faça
9:        $s_i \leftarrow rand()$ 
10:       $\mathbf{x}'_i \leftarrow \mathbf{x}_i + s_i * \vec{d}$ 
11:   senão
12:     para cada partícula  $i \in \{1, 2, \dots, n\}$  faça
13:       Inicie uma posição temporária,  $\mathbf{x}'_i \leftarrow \mathbf{x}_i$ .
14:       Selecione uma posição aleatória,  $k \in \{1, 2, \dots, n\}$  e  $k \neq i$ .
15:       Selecione aleatoriamente um componente,  $j_1 \in \{1, 2, \dots, m\}$ .
16:       se  $rand() < p_e$  então
17:          $\rho \leftarrow |\mathbf{x}_{i,j_1} - \mathbf{x}_{k,j_1}|$ 
18:          $\theta \leftarrow 2 * \pi * rand() - \pi$ 
19:          $\mathbf{x}'_{i,j_1} \leftarrow \mathbf{x}_{i,j_1} + \rho * \theta * \cos(\theta)$ 
20:       senão
21:         Selecione aleatoriamente um componente,  $j_2 \in \{1, 2, \dots, m\}$  e  $j_2 \neq j_1$ .
22:          $\mathbf{x}'_{i,j_1} \leftarrow (ub^{j_1} - lb^{j_1}) * \frac{\mathbf{x}_{k,j_2} - lb^{j_2}}{ub^{j_2} - lb^{j_2}} + lb^{j_1}$ 
23:     para cada partícula  $i \in 1, 2, \dots, n$  faça
24:       se  $f(\mathbf{x}'_i) < f_i$  então
25:          $f_i \leftarrow f(\mathbf{x}'_i)$ 
26:          $\mathbf{x}_i \leftarrow \mathbf{x}'_i$ 
27:       se  $f_i < f_b$  então
28:          $f_b \leftarrow f_i$ 
29:          $\mathbf{x}_b \leftarrow \mathbf{x}_i$ 
30:          $b \leftarrow i$ 

```

3.4 Conclusão

Neste capítulo foram apresentados os fundamentos teóricos do WFO, bem como sua estrutura e os operadores de fluxo laminar e fluxo turbulento. O algoritmo se destaca por ser um algoritmo de otimização global, e tem essa característica visto o operador de fluxo laminar ter um padrão de pesquisa paralelo e unidirecional, em que todas as componentes do problema são modificadas na iteração com base na mesma direção $\vec{d}$, enquanto que no operador de fluxo turbulento a pesquisa passa a ser intensificada na região da posição da partícula.

Em suma, o WFO é um algoritmo simples de ser implementado e de poucos parâmetros, sendo eles o parâmetro de probabilidade laminar p_l e de probabilidade de turbulência p_e , em que o parâmetro de probabilidade laminar é um limiar responsável por indicar qual operador será executado na iteração e o de probabilidade de turbulência é responsável por indicar a transformação a ser aplicada na execução do operador de fluxo turbulento.

4 Métodos propostos

Neste trabalho, o algoritmo WFO é adaptado para ser capaz de resolver problemas binários, visto que em sua versão original o algoritmo é desenvolvido para solucionar apenas problemas de otimização contínuo. Mais especificamente, neste trabalho é avaliada a capacidade das propostas aqui apresentadas de solucionar problemas binários quando aplicados na tarefa de seleção de características em problemas de classificação. Para isso, é necessário remodelar os operadores do WFO no contexto de problemas binários, a saber: o operador de fluxo laminar e operador de fluxo turbulento. Além disso, é preciso estabelecer uma função de aptidão para guiar o processo de otimização em busca da melhor solução.

O objetivo deste capítulo é portanto, descrever tais modificações no WFO e a metodologia utilizada no desenvolvimento das novas propostas.

4.1 Ambiente do problema

Neste trabalho busca-se gerar, a partir de um conjunto de dados $\mathbf{Z} \in \mathbb{R}^{n \times m}$ de mais alta dimensão, um novo conjunto de dados $\mathbf{Z}' \in \mathbb{R}^{n \times q}$, tal que a quantidade de atributos de $\mathbf{Z}'$ seja menor de que o do conjunto $\mathbf{Z}$, ou seja, $q < m$.

Em problemas de aprendizagem supervisionada, os conjuntos de dados são formados por diversos padrões. Por convenção, cada padrão $\mathbf{z}$ é representado como um vetor de entradas $\mathbf{z}^T = [z_1, z_2, \dots, z_m]^T$, e a classe $\mathbf{y}$ a qual o padrão pertence. Nesse tipo de problema, o conjunto de dados é representado por uma matriz $\mathbf{Z} \in \mathbb{R}^{n \times m}$, em que cada linha da matriz representa um padrão e as classes destas é definida por um vetor coluna $\mathbf{y} \in \mathbb{R}^{n \times 1}$, de modo que a primeira linha do vetor $\mathbf{y}$ descreve a classe do primeiro padrão $\mathbf{z}_0$, e, assim, respectivamente para as demais classes e padrões (RASCHKA, 2015).

4.2 Representação do problema

A solução para o problema é uma partícula de água, que por sua vez é formada por um conjunto de componentes (camadas), de tal maneira que cada componente é associada a um atributo. Mais especificamente, a partícula é representada por um vetor binário $\mathbf{x}^T = [d_1 \ d_2 \ \dots \ d_i \ \dots \ d_{m-1} \ d_m]^T$, em que $d_i \in \{0, 1\}$ é a d -ésima componente da partícula, e m é o número de atributos do conjunto de dados em questão. O valor 1 indica se determinado atributo permanecerá para a construção do novo conjunto de dados e o valor 0 indica se o atributo será removido. Por exemplo, considere uma partícula $\mathbf{x}^T = [1 \ 1 \ 0 \ 1 \ 0]^T$ pertencente a um conjunto de dados contendo cinco atributos, nesse caso

apenas o primeiro, o segundo e o quarto atributo serão selecionadas para compor o novo conjunto de dados.

A representação escolhida gera simplicidade e facilita a construção de novas partículas de água juntamente com a avaliação de seus valores de aptidão, que é definida adiante no trabalho.

4.3 Iniciação e nova abordagem para os operadores

O WFO mantém uma população de soluções, no qual cada solução é análoga a uma partícula de água. A iniciação é dada por um conjunto de partículas geradas aleatoriamente, de tal maneira que cada partícula recebe como valores para suas componentes apenas os valores 0 ou 1. As partículas geradas devem seguir a representação do problema como descrito na seção anterior.

A seguir são apresentados os métodos propostos com descrição dos novos operadores binários e da função de aptidão.

4.4 BWFO (Proposta 1)

Nessa abordagem denominada otimizador de fluxo de água binário (*binary water flow optimizer*, BWFO), novos operadores binários são apresentados com base no que foi descrito nas seções anteriores. Também é apresentado uma função de aptidão para avaliar as soluções geradas no processo de otimização.

4.4.1 Operador de fluxo laminar binário

O novo operador de fluxo laminar binário é empregado em cada partícula de água da iteração atual, em que cada partícula pode sofrer ou não alterações em suas camadas, decidida por um número aleatório gerado para cada camada da partícula na iteração, dada por:

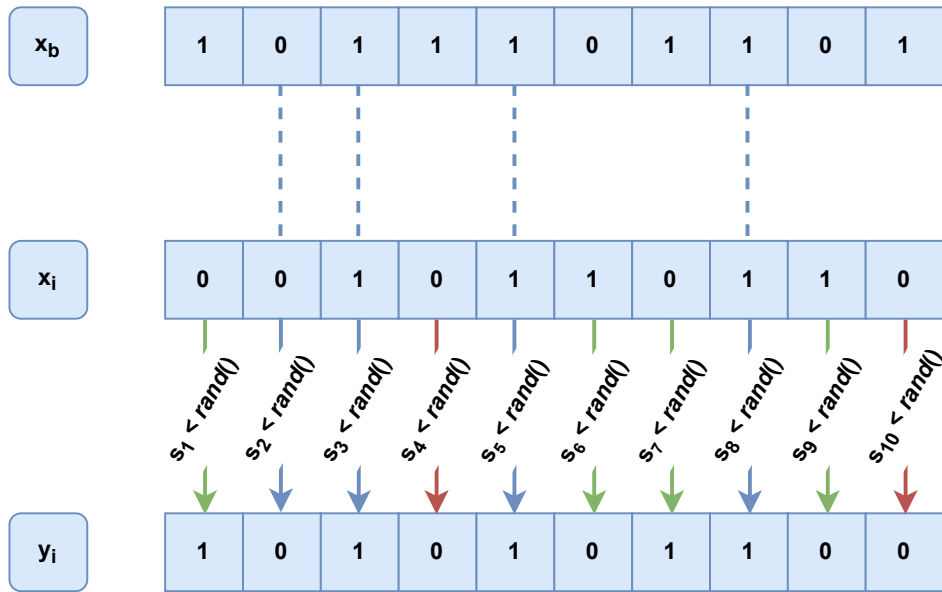
$$\mathbf{x}_{i,j}(t+1) = \begin{cases} \mathbf{x}_{b,j}(t), & \text{se } s_j < rand \\ \mathbf{x}_{i,j}(t), & \text{caso contrário.} \end{cases} \quad (4.1)$$

em que t é a t -ésima iteração, $\mathbf{x}_{i,j}$ é a j -ésima camada da i -ésima partícula, e $\mathbf{x}_{b,j}$ é a j -ésima camada da melhor partícula na iteração e b o seu índice, $s_j \in [0, 1]$ é um valor aleatório gerado para a j -ésima camada da partícula, e que deve ser menor que um valor aleatório $rand \in [0, 1]$. Assim, quando $s_j < rand$, a j -ésima camada da partícula $\mathbf{x}_i$ tem seu valor substituído pela respectiva camada da melhor partícula vigente na iteração. O operador de fluxo laminar binário, faz com que partículas de água consigam obter

camadas da partícula de melhor energia potencial na geração atual, realizando uma nova movimentação em direção da melhor partícula.

A Figura 8, exemplifica a execução do operador laminar binário. As flechas direcionais verdes indicam que a primeira condição da Equação (4.1) foi satisfeita, e assim a camada j da partícula $\mathbf{x}_i$ recebe o valor da respectiva camada da melhor partícula atual $\mathbf{x}_b$. Já as flechas azuis indicam que não houveram alterações nos valores das camadas independentemente da condição ter sido satisfeita ou não, pois os valores das respectivas camadas de $\mathbf{x}_b$ têm os mesmos valores da partícula $\mathbf{x}_i$. As flechas vermelhas indicam que a primeira condição não foi satisfeita, assim a camada de $\mathbf{x}_i$ continua com o mesmo valor anterior.

Figura 8 – Processo do operador laminar binário



Fonte: próprio autor.

4.4.2 Operador de fluxo turbulento binário

O novo operador de fluxo turbulento, por sua vez utiliza do parâmetro de probabilidade de turbulência p_e para escolher quais camadas irão sofrer alterações. Quando o operador de turbulência é aplicado, é selecionado uma partícula de água $\mathbf{x}_k$ aleatoriamente na população, a camada da partícula a ser modificada recebe a camada da partícula $\mathbf{x}_k$ se ambas forem diferentes, caso contrário a camada da partícula atual recebe aleatoriamente um valor $v \in \{0, 1\}$, assumindo assim de forma aleatória o valor 0 ou 1, como segue:

$$\mathbf{x}_{i,j}(t+1) = \begin{cases} \mathbf{x}_{k,j}(t), & \text{se } \mathbf{x}_{k,j}(t) \neq \mathbf{x}_{i,j}(t) \\ v \in \{0, 1\}, & \text{caso contrário.} \end{cases} \quad (4.2)$$

O operador de fluxo turbulento binário visa que as partículas possam receber atributos das demais partículas ou sofrer uma espécie de mutação quando a camada selecionada da partícula $\mathbf{x}_k$ tem o mesmo valor da partícula $\mathbf{x}_i$, a fim de trazer mais diversidade para a pesquisa dentro do espaço de busca, realizando uma busca mais exploratória.

4.4.3 Função de aptidão

A função de aptidão (*fitness*) visa avaliar a qualidade das soluções (partículas de água) encontradas durante o processo de otimização de um determinado algoritmo evolutivo, neste caso o BWFO. Como o problema a ser resolvido é o de seleção de características aplicada ao problema de classificação, a função de aptidão implementada neste trabalho utiliza a taxa de acerto de um classificador quando se considera a redução de dimensionalidade de um conjunto de dados e, assim, a tarefa em questão é modelada como um problema de maximização. O classificador escolhido é o floresta de caminhos ótimos e a função de aptidão é dada por:

$$fitness(\mathbf{x}) = \frac{total\ de\ acertos}{total\ de\ padrões} \quad (4.3)$$

em que o total de acertos é o somatório de todas as predições corretas do classificador, e o total de padrões é a quantidade de padrões no conjunto de dados avaliado.

4.4.4 Framework BWFO

O Algoritmo 6 mostra a ideia geral para o algoritmo BWFO utilizado para o problema de seleção de características. Como dito, o BWFO segue a estrutura do WFO com alterações do funcionamento de seus operadores de fluxo laminar e turbulento na nova versão proposta.

Na iniciação do Algoritmo 6 é definida uma população com n soluções de forma aleatória (linha 1), seguindo a representação da Seção 4.2. A seguir, na linha 2 é calculada a energia potencial (aptidão) de cada partícula. No laço principal, linhas 3-28, é empregado os operadores de fluxo laminar e turbulento. Uma vez que a condição da linha 4 seja satisfeita, com base no parâmetro de probabilidade laminar p_l , o operador de fluxo laminar é executado (linhas 5-10). Assim com base na melhor partícula da iteração as demais terão suas camadas alteradas conforme a linha 10, recebendo o valor da camada da melhor solução vigente, sendo efetuados quando um valor aleatório $s_j \in [0, 1]$ for menor que um valor aleatório dentro do mesmo intervalo de s_j .

Nas ocorrências em que a condição da linha 4 não é satisfeita, o operador de fluxo turbulento é aplicado (linhas 12-20). Assim, a partir do parâmetro de probabilidade de turbulência p_e as partículas tem suas camadas modificadas nas linhas 16-20. No operador

turbulento uma partícula de índice k é selecionada aleatoriamente para guiar a alteração para a camada vigente na iteração. Quando o valor da camada da partícula k difere da respectiva camada da partícula atual, o valor de $\mathbf{x}'_{i,j}$ obtém o valor de $\mathbf{x}_{k,j}$. Por sua vez, quando o valor das camadas não diferem, $\mathbf{x}'_{i,j}$ recebe o valor 1 ou 0 aleatoriamente (linhas 19-20). As linha que se sucedem avaliam as soluções.

Algoritmo 6 Algoritmo BWFO

Entrada: Conjunto de treinamento $\mathbf{Z}_1$ e validação $\mathbf{Z}_2$ rotulados e um número T de iterações.

Saída: Partícula de melhor acurácia.

```

1: Inicie aleatoriamente uma população de  $n$  soluções (partículas)
2: Calcule a energia potencial de cada partícula (acurácia)
3: enquanto o critério de parado não é satisfeito faça
4:   se  $rand < pl$  então
5:     Selecione o índice da melhor partícula vigente  $b$ 
6:     para cada partícula  $i \in \{1, 2, \dots, n\}$  faça
7:       para cada componente  $j \in \{1, 2, \dots, m\}$  faça
8:          $s_j \leftarrow rand$ 
9:         se  $s_j < rand$  então
10:           $\mathbf{x}'_{i,j} \leftarrow \mathbf{x}_{b,j}$ 
11:   senão
12:     para cada partícula  $i \in \{1, 2, \dots, n\}$  faça
13:       para cada componente  $j \in \{1, 2, \dots, m\}$  faça
14:         Selecione aleatoriamente o índice de uma partícula  $k$ 
15:         se  $rand < p_e$  então
16:           se  $\mathbf{x}'_{i,j} \neq \mathbf{x}_{k,j}$  então
17:              $\mathbf{x}'_{i,j} \leftarrow \mathbf{x}_{k,j}$ 
18:         senão
19:            $v \leftarrow randint(0, 1)$ 
20:            $\mathbf{x}'_{i,j} \leftarrow v$ 
21:   para cada partícula  $i \in 1, 2, \dots, n$  faça
22:     se  $f(\mathbf{x}'_i) > f_i$  então
23:        $f_i \leftarrow f(\mathbf{x}'_i)$ 
24:        $\mathbf{x}_i \leftarrow \mathbf{x}'_i$ 
25:     se  $f_i > f_b$  então
26:        $f_b \leftarrow f_i$ 
27:        $\mathbf{x}_b \leftarrow \mathbf{x}_i$ 
28:        $b \leftarrow i$ 

```

4.5 FSBWFO (Proposta 2)

Nessa abordagem denominada otimizador de fluxo de água binário de tamanho fixo (*fixed size binary water flow optimizer*, FSBWFO) diferente da proposta anterior, os operadores existentes são remodelados de forma a gerar soluções com uma taxa fixa de seleção de atributos, guiada pelo parâmetro de poda θ , empregado na definição da

população de soluções. Assim, vale ressaltar que no FSBWFO os novos operadores devem sempre produzir soluções com o mesmo número de atributos. Nas subseções a seguir os novos operadores e o novo método proposto são apresentados.

4.5.1 Operador de fluxo laminar binário de tamanho fixo

O operador de fluxo laminar de tamanho fixo é executado conforme a proposta anterior, seguindo o parâmetro de probabilidade laminar p_l e é empregada em cada partícula de água e em cada camada da solução candidata, sendo escolhidas para sofrer ou não alteração em sua camada com base nas equações a seguir:

$$f' = \frac{f(\mathbf{x}_i) - f_{min} + 1}{f_{max} - f_{min} + 1}, \quad (4.4)$$

$$p_j = 1 - \mathcal{N}(s_j \cdot f', 1), \quad (4.5)$$

em que f' é um número no intervalo entre 0 e 1, visando ser uma espécie de peso com base no valor de aptidão da solução; f_{max} e f_{min} são o melhor e o pior valor de aptidão na iteração; $s_j \in [0, 1]$ é um número aleatório gerado para a j -ésima camada; $\mathcal{N}(\mu, \sigma)$ é uma distribuição normal, com desvio padrão σ e média μ , enquanto p_j é a probabilidade da j -ésima camada sofrer alteração. Como pode-se perceber com base na Equação (4.5) partículas mais distantes da melhor solução na iteração recebem mais informações da mesma, pois os valores de p_j são frequentemente mais altos para as partículas de menor *fitness*. As partículas mais próximas da melhor solução entretanto, tem valores frequentemente mais baixos e aderem com menor frequência os atributos da melhor solução. O processo de modificação das camadas é empregado com base nas seguintes equações:

$$\mathbf{x}_{i,j}(t+1) = \begin{cases} \mathbf{x}_{b,j}(t), & \text{se } rand < p_j \text{ tal que } \mathbf{x}_{i,j}(t) \neq \mathbf{x}_{b,j}(t) \\ \mathbf{x}_{i,j}(t), & \text{caso contrário.} \end{cases} \quad (4.6)$$

$$\mathbf{x}_{i,h}(t+1) = \begin{cases} 1 - \mathbf{x}_{i,h}(t), & \text{se } rand < p_j \text{ tal que } \mathbf{x}_{i,j}(t) \neq \mathbf{x}_{b,j}(t) \text{ e } \mathbf{x}_{i,h}(t) \neq \mathbf{x}_{i,j}(t) \\ \mathbf{x}_{i,h}(t), & \text{caso contrário.} \end{cases} \quad (4.7)$$

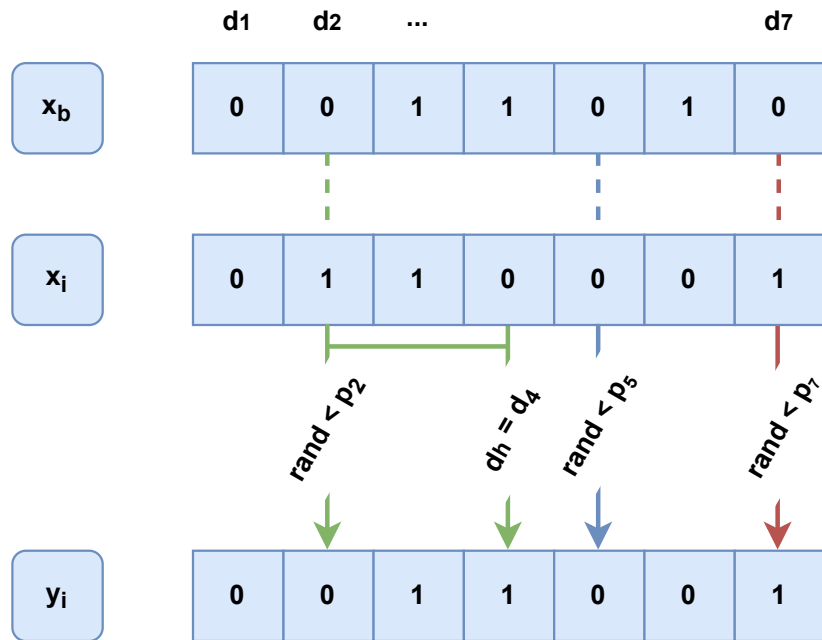
em que *rand*, como já apresentado, é um número aleatório gerado entre $[0, 1]$ utilizado nas Equações (4.6) e (4.7). Com base na Equação (4.6), os valores das camadas são modificados quando o valor de p_j é maior que o valor de *rand* e $\mathbf{x}_{i,j}(t) \neq \mathbf{x}_{b,j}(t)$, em que b é o índice da melhor partícula na iteração, então $\mathbf{x}'_{i,j}(t+1)$ assume o valor da camada de mesmo índice da melhor solução $\mathbf{x}_{b,j}(t)$. Nos casos em que a alteração venha a acarretar na quebra da restrição da quantidade de atributos selecionados, com base na taxa de poda θ , é escolhido

aleatoriamente uma camada h para que o seu valor seja invertido a fim de manter a restrição do problema, conforme apresentado na Equação (4.7), em que $\mathbf{x}_{i,h}(t) \neq \mathbf{x}_{i,j}(t)$.

A Figura 9, exemplifica a execução do operador laminar binário de tamanho fixo. As flechas direcionais verdes indicam que a primeira condição da Equação (4.6) foi satisfeita e, assim, a j -ésima camada da partícula $\mathbf{x}'_i$ recebe o valor da respectiva camada da melhor partícula atual $\mathbf{x}_b$. Além disso, vale ressaltar que uma camada h é escolhida pra sofrer uma inversão de seu valor para balancear a quantidade de atributos selecionados. Já as flechas azuis indicam que não houveram alterações nos valores das camadas, independente da condição ter sido satisfeita ou não, pois o valor da respectiva camada de $\mathbf{x}_b$ tem o mesmo valor da camada da partícula $\mathbf{x}_i$. As flechas vermelhas indicam que a primeira condição não foi satisfeita, assim a camada de $\mathbf{x}_i$ permanece com o mesmo valor.

Em razão da aplicação da Equação (4.5) percebe-se que as camadas das partículas de baixo potencial energético sofrem modificações com mais frequência, assumindo alguns dos valores da partícula de melhor potencial. Entretanto, para as soluções com potencial mais próximo a melhor solução, em razão da Equação (4.5) percebe-se uma diminuição considerável na frequência das modificações de suas camadas.

Figura 9 – Processo do operador laminar binário de tamanho fixo



Fonte: próprio autor.

4.5.2 Operador de fluxo turbulento binário de tamanho fixo

O operador de fluxo turbulento binário de tamanho fixo funciona de forma similar ao operador laminar da subseção anterior. A primeira diferença reside no fato de que no operador de fluxo turbulento o parâmetro de probabilidade de turbulência p_e é o limiar

para aplicação do operador em cada camada. A outra diferença se dá na questão de que ao invés de utilizar como referência a melhor partícula na iteração, desta vez é selecionado aleatoriamente uma partícula k dentro do conjunto de soluções candidatas, tal que $\mathbf{x}_k \neq \mathbf{x}_i$. Nos casos em que a restrição de taxa de poda de atributos é quebrada, uma camada h é selecionada aleatoriamente tal que $\mathbf{x}_{i,h} \neq \mathbf{x}_{i,j}$ tenha o seu valor invertido. As Equações (4.8) e (4.9) descrevem a operação empregada no operador fluxo turbulento binário de tamanho fixo.

$$\mathbf{x}_{i,j}(t+1) = \begin{cases} \mathbf{x}_{k,j}(t), & \text{se } rand < p_e \text{ tal que } \mathbf{x}_{i,j}(t) \neq \mathbf{x}_{k,j}(t) \\ \mathbf{x}_{i,j}(t), & \text{caso contrário.} \end{cases} \quad (4.8)$$

$$\mathbf{x}_{i,h}(t+1) = \begin{cases} 1 - \mathbf{x}_{i,h}(t), & \text{se } rand < p_e \text{ tal que } \mathbf{x}_{i,j}(t) \neq \mathbf{x}_{k,j}(t) \text{ e } \mathbf{x}_{i,h}(t) \neq \mathbf{x}_{i,j}(t) \\ \mathbf{x}_{i,h}(t), & \text{caso contrário.} \end{cases} \quad (4.9)$$

em que $rand$ é um número aleatório gerado entre $[0, 1]$.

4.6 Função de aptidão

A função de aptidão (*fitness*) para o FSBWFO segue semelhante ao da abordagem anterior, conforme descrito na subseção 4.4.3.

4.7 Framework FSBWFO

No Algoritmo 7 é mostrado uma visão geral para o otimizador FSBWFO, que apresenta-se similar em termos de estrutura ao otimizador BWFO, sendo realizado somente alterações no funcionamento de seus operadores na nova versão proposta, em que a grande mudança está na taxa de remoção de atributos, que é fixa e na forma com que os operadores trabalham para sempre manter essa restrição uma vez que a camada da partícula é modificada. Tanto no operador de fluxo laminar como no operador de fluxo turbulento, há uma verificação da remoção adequada de atributos e, para os casos nos quais ocorre a quebra da restrição uma camada é selecionada de forma aleatória e alterada a fim de manter a restrição.

Na linha 1 é definida uma população de soluções com uma quantidade fixa de atributos com base na taxa de poda θ . Já na linha 2 é calculada a energia potencial (aptidão) de cada partícula. No laço principal, linhas 3-29, é empregado os operadores de fluxo laminar e turbulento binário de tamanho fixo. Se a condição da linha 4 for satisfeita com base no parâmetro de probabilidade laminar p_l , o operador de fluxo laminar é aplicado (linhas 5-13) e desta forma, com base na melhor partícula da iteração, as demais partículas

tem suas camadas alteradas como descrita nas linhas 11-13, caso a condição apresentada da linha 10 seja verdadeira.

Quando a condição da linha 4 é falso, o operador de fluxo turbulento é aplicado (linhas 15-21) e, assim, com base no parâmetro de probabilidade de turbulência p_e as partículas tem suas camadas modificadas como descrito no código apresentado nas linhas 19-21, em que uma partícula de índice k selecionada aleatoriamente guia a alteração por fornecer o valor de sua camada. Nas linhas 22-29 é realizada a avaliação de cada partícula e, caso a energia potencial da partícula tenha aumentado, o valor da partícula é então atualizado. A atualização da melhor solução é realizada quando uma das partículas avaliadas tem superado a energia potencial da melhor partícula na iteração.

Algoritmo 7 Algoritmo FSBWFO

Entrada: Conjunto de treinamento $\mathbf{Z}_1$ e validação $\mathbf{Z}_2$ λ -rotulados e um número T de iterações, taxa de poda de características θ .

Saída: Partícula de melhor acurácia.

```

1: Inicie aleatoriamente uma população de  $n$  soluções (partículas) obedecendo a taxa de
   poda  $\theta$ .
2: Calcule a energia potencial de cada partícula (acurácia).
3: enquanto o critério de parada não é satisfeito faça
4:   se  $rand < pl$  então
5:     Selecione o índice da melhor partícula vigente  $b$ 
6:     para cada partícula  $i \in \{1, 2, \dots, n\}$  faça
7:       para cada componente  $j \in \{1, 2, \dots, m\}$  faça
8:          $s_j \leftarrow rand$ 
9:          $p_j \leftarrow 1 - N\left(s_j \left(\frac{f(\mathbf{x}_i) - f_{min} + 1}{f_{max} - f_{min} + 1}\right), 1\right)$ 
10:        se  $rand < p_j$  e  $\mathbf{x}_{i,j} \neq \mathbf{x}_{b,j}$  então
11:           $\mathbf{x}'_{i,j} \leftarrow \mathbf{x}_{b,j}$ 
12:          Selecione aleatoriamente uma camada de índice  $h$  tal que  $\mathbf{x}_{i,h} \neq \mathbf{x}_{i,j}$ .
13:           $\mathbf{x}'_{i,h} \leftarrow 1 - \mathbf{x}_{i,h}$ 
14:        senão
15:          para cada partícula  $i \in \{1, 2, \dots, n\}$  faça
16:            para cada componente  $j \in \{1, 2, \dots, m\}$  faça
17:              Selecione aleatoriamente o índice de uma partícula  $k$ 
18:              se  $rand < p_e$  e  $\mathbf{x}_{i,j} \neq \mathbf{x}_{k,j}$  então
19:                 $\mathbf{x}'_{i,j} \leftarrow \mathbf{x}_{k,j}$ 
20:                Selecione aleatoriamente uma camada de índice  $h$  tal que  $\mathbf{x}_{i,h} \neq \mathbf{x}_{i,j}$ .
21:                 $\mathbf{x}'_{i,h} \leftarrow 1 - \mathbf{x}_{i,h}$ 
22:            para cada partícula  $i \in 1, 2, \dots, n$  faça
23:              se  $f(\mathbf{x}'_i) > f_i$  então
24:                 $f_i \leftarrow f(\mathbf{x}'_i)$ 
25:                 $\mathbf{x}_i \leftarrow \mathbf{x}'_i$ 
26:              se  $f_i > f_b$  então
27:                 $f_b \leftarrow f_i$ 
28:                 $\mathbf{x}_b \leftarrow \mathbf{x}_i$ 
29:                 $b \leftarrow i$ 

```

4.8 FPBWFO (Proposta 3)

A abordagem dessa seção, denominada otimizador de fluxo de água binário com poda de características (*feature pruning binary water flow optimizer*, FPBWFO), traz melhorias quando comparado ao BWFO (proposta 1). O BWFO realiza suas operações focando em explorar o espaço de busca e em encontrar a melhor solução com base na sua aptidão, fornecida por um classificador, sem levar em consideração a redução efetiva do número de atributos selecionados nas soluções geradas. Por sua vez, o FPBWFO busca encontrar a melhor solução ao mesmo passo em que tal solução tenha selecionado o menor número de atributos possível, isso é realizado através do novo operador de fluxo turbulento, uma vez que o novo operador gera novas soluções visando a redução de atributos. Nas subseções a seguir os novos operadores e o novo método proposto são apresentados.

4.8.1 Operador de fluxo laminar binário

O operador de fluxo laminar binário do FPBWFO assim como nas propostas anteriores, é aplicado a partir do parâmetro de probabilidade laminar p_l , que atua como um limiar entre o operador de fluxo laminar e o de fluxo turbulento. O novo operador laminar utiliza a Equação (4.5) como a probabilidade de cada camada da partícula ser alterada, visando melhorar o equilíbrio na aplicação do operador nas soluções de baixa aptidão como nas de maior aptidão. Assim, a modificação da solução é descrita conforme a seguir:

$$\mathbf{x}_{i,j}(t+1) = \begin{cases} \mathbf{x}_{b,j}(t), & \text{se } rand < p_j \\ \mathbf{x}_{i,j}(t), & \text{caso contrário.} \end{cases} \quad (4.10)$$

em que p_j é a probabilidade da j -ésima camada sofrer alteração (veja Equação (4.5)); $rand \in [0, 1]$ um número aleatório e b o índice da melhor solução. Assim, o novo operador faz com que partículas de mais baixa aptidão recebam com uma maior frequência os atributos da melhor solução na iteração, enquanto que as soluções próximo da melhor solução venham a ter uma movimentação mais serena em sua direção de posição.

4.8.2 Operador de fluxo turbulento binário

O novo operador de fluxo turbulento, binário assim como no BWFO, utiliza o parâmetro de probabilidade de turbulência p_e para escolher quais camadas devem sofrer alteração. Por sua vez, quando satisfeita a condição com base no parâmetro p_e , o novo operador modifica as camadas da partícula na iteração como descrito nas equações a seguir.

$$p' = s_j \cdot e^{-f'} \cdot \frac{1}{n} \sum_{i=1}^n \mathbf{x}_{i,j} \quad (4.11)$$

$$\mathbf{x}_{i,j}(t+1) = \begin{cases} 1, & \text{se } rand < p' \\ 0, & \text{caso contrário.} \end{cases} \quad (4.12)$$

em que f' é um peso para a solução com base no seu valor de aptidão, definida pela Equação (4.4), $s_j \in [0, 1]$ é um número aleatório gerado para cada camada e o somatório se refere a média de seleção da camada j na população, sendo n o tamanho da população.

O novo operador de fluxo turbulento para o FPBWFO visa explorar soluções no espaço de busca com quantidades menores de atributos, visto que ao se utilizar a Equação (4.11) a remoção de atributos ocorre com mais frequência.

4.9 Função de aptidão

A função de aptidão para o FPBWFO é obtida de forma semelhante ao da abordagem anterior (veja a subseção 4.4.3).

4.10 Framework FPBWFO

No Algoritmo 8 é mostrado o otimizador FPBWFO, que tem a estrutura similar aos métodos anteriores. Como pode ser visto, o otimizador é de fato uma melhoria em relação ao BWFO, pois visa tanto a encontrar soluções com a melhor aptidão como também à redução da quantidade de atributos selecionados. Assim, o algoritmo busca gerar soluções ótimas com menos atributos que o BWFO.

O algoritmo traz consigo algumas melhorias para os operadores. No operador de fluxo laminar, a Equação (4.5) faz com que solução de baixa aptidão sofram modificações em suas camadas com uma maior frequência, enquanto soluções de aptidão próximas a melhor solução realize modificações com uma frequência menor.

No operador de fluxo turbulento do FPBWFO, o parâmetro de probabilidade de turbulência p_e passa a ser um decisor para que ocorra a modificação na camada da partícula. Assim, caso um valor aleatório $rand \in [0, 1]$ seja menor que p_e , a camada atual deverá selecionar atributos com base na Equação (4.11), que por sua vez gera valores consideravelmente de menor probabilidade para a verificação, fazendo com que o operador realize mais remoções de atributos do que seleção.

Nas linhas 1-2 são realizadas a definição da população e sua avaliação, adiante no laço principal, linhas 3-19 são empregados os novos operadores binários de fluxo laminar e

turbulento. Se a condição da linha 4 for satisfeita, levando em consideração p_l , o operador de fluxo laminar é aplicado (linhas 5-11) e, assim, com base na melhor partícula da iteração, as demais partículas tem suas camadas alteradas conforme a linha 11, sendo executadas quando a condição da linha 10 é satisfeita, dessa forma $\mathbf{x}_{i,j}$ obtém o valor da camada $\mathbf{x}_{b,j}$ da melhor partícula na iteração, realizando movimentações em direção de $\mathbf{x}_b$. O operador de fluxo turbulento, conforme a linha 16, faz com que ocorra a remoção de atributos com maior frequência, visto a equação gerar valores um pouco mais próximos de zero. Consequentemente, nas linhas 20-27 são realizadas as avaliações das soluções para as próximas iterações.

Algoritmo 8 Algoritmo FPBWFO

Entrada: Conjunto de treinamento $\mathbf{Z}_1$ e validação $\mathbf{Z}_2$ λ -rotulados e um número T de iterações.

Saída: Partícula de melhor acurácia.

```

1: Inicie aleatoriamente uma população de  $n$  soluções (partículas).
2: Calcule a energia potencial de cada partícula (acurácia).
3: enquanto o critério de parado não é satisfeito faça
4:   se  $rand < p_l$  então
5:     Selecione o índice da melhor partícula vigente  $b$ 
6:     para cada partícula  $i \in \{1, 2, \dots, n\}$  faça
7:       para cada componente  $j \in \{1, 2, \dots, m\}$  faça
8:          $s_j \leftarrow rand$ 
9:          $p_j \leftarrow 1 - N\left(s_j \left(\frac{f(\mathbf{x}_i) - f_{min} + 1}{f_{max} - f_{min} + 1}\right), 1\right)$ 
10:        se  $rand < p_j$  então
11:           $\mathbf{x}'_{i,j} \leftarrow \mathbf{x}_{b,j}$ 
12:        senão
13:          para cada partícula  $i \in \{1, 2, \dots, n\}$  faça
14:            para cada componente  $j \in \{1, 2, \dots, m\}$  faça
15:              se  $rand < p_e$  então
16:                se  $rand < s_j \cdot e^{-f'} \cdot \frac{1}{n} \sum_{i=1}^n \mathbf{x}_{i,j}$  então
17:                   $\mathbf{x}'_{i,j} \leftarrow 1$ 
18:                senão
19:                   $\mathbf{x}'_{i,j} \leftarrow 0$ 
20:            para cada partícula  $i \in 1, 2, \dots, n$  faça
21:              se  $f(\mathbf{x}'_i) > f_i$  então
22:                 $f_i \leftarrow f(\mathbf{x}'_i)$ 
23:                 $\mathbf{x}_i \leftarrow \mathbf{x}'_i$ 
24:              se  $f_i > f_b$  então
25:                 $f_b \leftarrow f_i$ 
26:                 $\mathbf{x}_b \leftarrow \mathbf{x}_i$ 
27:                 $b \leftarrow i$ 

```

4.11 Conclusão

Neste capítulo foram propostos três técnicas de seleção de características baseados na meta-heurística WFO. Os métodos buscam selecionar um conjunto de atributos que mantenham ou até mesmo melhora o desempenho de um classificador e o seu custo computacional.

O algoritmo BWFO é similar ao WFO em questão de estrutura, operadores, parâmetros e é simples de ser implementado. O parâmetro de probabilidade laminar p_l , continua como um limiar de decisão para a aplicação do fluxo laminar ou turbulento. Já no fluxo turbulento, o parâmetro de probabilidade de turbulência p_e é utilizado como um decisor para a modificação da cada camada de cada partícula.

O operador de fluxo laminar visa que a partícula a sofrer modificação venha a receber os valores da camada da partícula de maior potencial energético na iteração, de forma que a partícula atual venha a movimentar-se em direção da melhor solução. Já o operador de fluxo turbulento aumenta a diversidade das soluções, pois cada camada da partícula pode receber o valor da camada de uma outra solução selecionada aleatoriamente na população, ou receber aleatoriamente o valor 0 ou 1, quando a solução selecionada aleatoriamente tem o mesmo valor que a respectiva camada da partícula a ser modificada.

O algoritmo FSBWFO segue com algumas similaridades em relação ao BWFO, mas sua principal diferença é que em sua iniciação a quantidade de atributos a serem selecionados deve ser informado previamente. Assim, o algoritmo FSBWFO gera e trabalha somente com soluções com uma quantidade fixa de atributos. O operador de fluxo laminar binário de tamanho fixo, assim como no BWFO, aplica em cada camada das soluções os valores da camada da melhor partícula na iteração, uma vez que partículas com baixo valor de aptidão tendem a receber as camadas da melhor solução com uma maior frequência. Além disso, o operador tem um mecanismo para manter a quantidade correta de atributos selecionados, a fim de evitar que uma operação cause um desbalanceamento na taxa de seleção escolhida.

Já operador de fluxo turbulento binário de tamanho fixo segue a ideia principal do fluxo laminar no que diz a respeito a forma com que se modifica as camadas de cada solução, com exceção de que as modificações para a solução vigente na iteração é realizada com a probabilidade de turbulência p_e e a alteração é efetuado com base em uma solução k escolhida aleatoriamente. Entretanto, o operador turbulento de tamanho fixo segue o mesmo mecanismo do operador laminar para manter a proporção de seleção de atributos do problema.

O algoritmo FPBWFO é um método que busca encontrar a solução que maximize a taxa de acerto de um classificador, assim como no algoritmo BWFO, entretanto realizando uma busca com uma taxa menor de seleção de atributos. O operador de fluxo laminar

binário do FPBWFO também faz com que as partículas recebam os valores da melhor solução na iteração, assim como nos métodos propostos BWFO e FSBWFO, buscando uma rápida evolução para as soluções com baixa aptidão. O operador de fluxo turbulento por sua vez, busca diversificar as soluções realizando uma busca exploratória ao mesmo tempo que as modificações realizadas pelo operador geram soluções com uma quantidade menor de seleção de atributos.

5 Simulações computacionais

Neste Capítulo são apresentados os resultados obtidos para avaliar os métodos propostos. São apresentados ainda os resultados para outras técnicas existentes na literatura para redução de dimensionalidade, confrontando-os em termos de acurácia e redução de atributos.

5.1 Metodologia

Para avaliar o desempenho dos métodos propostos neste trabalho, foram utilizadas doze bases de dados obtidas do repositório UCI Machine Learning ¹, a saber: *Australian Credit Approval*, *Breast Cancer Wisconsin*, *Dermatology*, *Vertebral Column*, *Sonar*, *Liver Disorders*, *Pima Indians Diabetes*, *Divorce Predictors*, *Parkinson's Disease*, *Turkish Music Emotion*, *Urban Land Cover* e *Arrhythmia* ². A Tabela 2 apresenta as principais informações desses conjuntos de dados, incluindo as siglas, número de padrões, número de atributos e número de classes.

Tabela 2 – Descrição dos conjuntos de dados utilizados no trabalho.

<i>Dataset</i>	Abreviação	# Padrões	# Atributos	# Classes
Australian Credit Approval	ACA	690	14	2
Breast Cancer Wisconsin	BCW	699	10	2
Dermatology	DM	366	33	6
Vertebral Column	VC	310	6	3
Sonar (Mine vs Rocks)	SNR	208	60	2
Liver Disorders (Bupa dataset)	LD	345	6	2
Pima Indians Diabetes	PID	768	8	2
Divorce Predictors	DIV	170	54	2
Parkinson's Disease	PDS	756	753	2
Turkish Music Emotion	TKM	400	50	4
Urban Land Cover	UBL	540	147	9
Arrhythmia	ARR	452	279	16

Fonte: próprio autor.

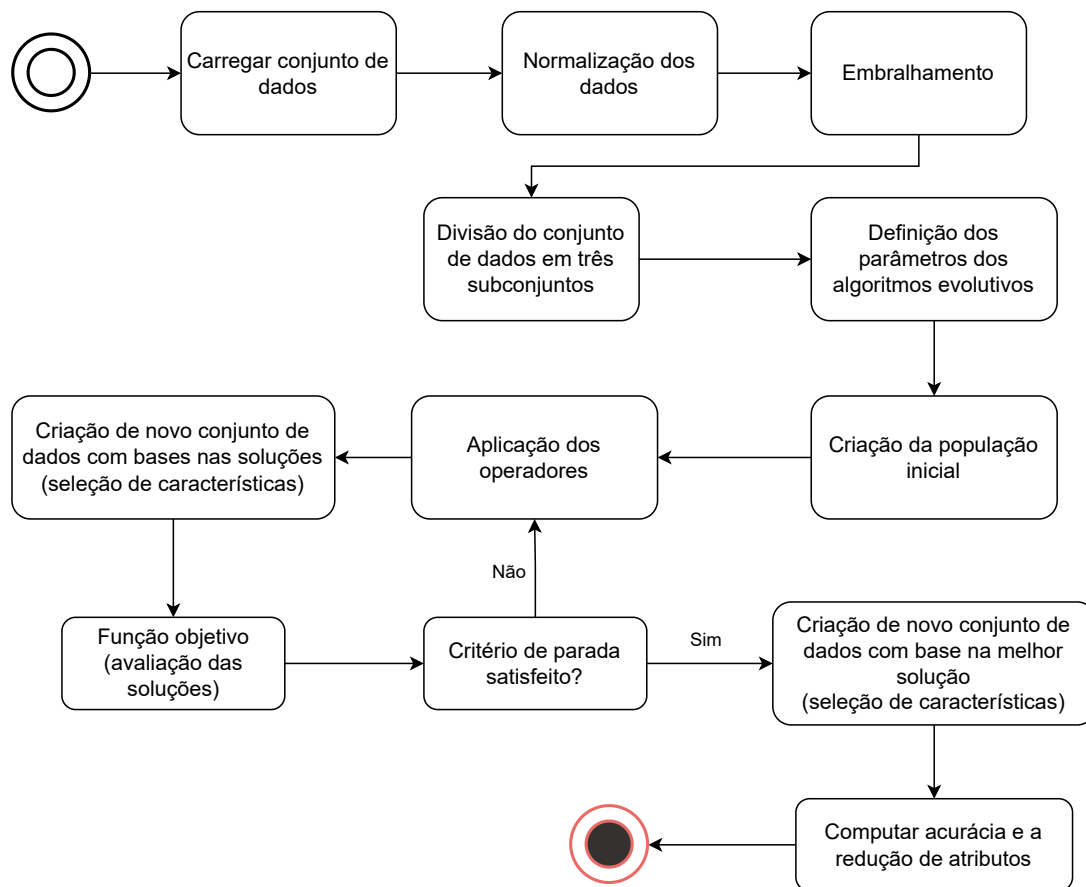
Inicialmente, os conjuntos de dados foram normalizados, embaralhados e particionados em três subconjuntos: conjunto de treinamento (Z_1), validação (Z_2) e teste (Z_3), em que 30% dos dados foram destinados ao conjunto de treinamento, 30% para o conjunto de validação e os outros 40% restantes para o conjunto de teste. Durante a execução dos métodos evolutivos, novos subconjuntos Z'_1 , Z'_2 e Z'_3 são gerados de acordo com os valores binários presentes em cada solução gerada aleatoriamente, em que o valor 1 e 0 representam,

¹ <https://archive.ics.uci.edu/ml/>

² O conjunto de dados *Arrhythmia* foi modificado para considerar apenas duas classes no problema, considerando a normalidade e a existência de arritmia nos batimentos cardíacos.

respectivamente, a presença e ausência de um atributo nos novos subconjuntos. Então, o algoritmo OPF é treinado em Z'_1 e realiza a predição utilizando Z'_2 para avaliar a aptidão de cada solução candidata. Esse processo é repetido durante um determinado número de iterações (gerações) ou quando o melhor valor de aptidão de uma onda tenha convergido para um valor específico. Neste trabalho, o número de gerações foi definido como 30, enquanto o valor de aptidão de uma onda é visto como tendo convergido caso o valor de melhor solução até o momento não progrida durante cinco gerações. Após os critérios de parada serem satisfeitos, os métodos evolutivos retornam como resposta a melhor solução encontrada, ou seja, um vetor binário mapeando os atributos da base de dados que permitiram que o classificador obtivesse uma melhor taxa de acerto em Z'_2 . Após isso, o classificador treinado em Z'_1 é utilizado para computar um novo valor de taxa de acerto em Z'_3 . A Figura 10 apresenta em um fluxograma os passos utilizados na metodologia deste trabalho para as meta-heurísticas. Todos os algoritmos foram implementados utilizando o software MATLAB como ambiente de desenvolvimento em um computador com um processador Intel® Core i7 3.40GHz $\times$ 8, com 16GB de memória RAM e sistema operacional Linux Ubuntu 18.04.2 LTS.

Figura 10 – Metodologia utilizada para os métodos propostos



Fonte: próprio autor.

A fim de realizar uma comparação justa entre os métodos estudados neste tra-

balho, os experimentos foram conduzidos utilizando os mesmos padrões de dados para cada abordagem. Portanto, cada método recebe como parâmetro exatamente os mesmos subconjuntos Z_1 , Z_2 e Z_3 . Além disso, foram escolhidos como métodos de comparação aos métodos propostos, a busca cuco binária (*binary cuckoo search*, BCS) (RODRIGUES et al., 2013), otimização binária de ondas de água (*binary water wave optimization*, BWWO) (MACÊDO; BARBOSA; ROCHA NETO, 2019) e o algoritmo do morcego binário (*binary bat algorithm*, BBA) (NAKAMURA et al., 2012), além do PCA e o LDA que possuem como quantidade de componentes e discriminantes o número de atributos retornado pelos métodos propostos. Visto que tanto o PCA quanto o LDA quando comparados aos métodos evolutivos não necessitam de um conjunto de validação, então apenas os subconjuntos Z_1 e Z_3 foram utilizados quando aplicados nos mesmos. A seguir, para cada método, o classificador OPF é treinado em Z'_1 e computado a taxa de acerto utilizando os dados de teste em Z'_3 . Então, o valor da taxa de acerto alcançada é comparado com os resultados obtidos pelos demais métodos. Um procedimento similar acontece para obter o valor da taxa de acerto para o classificador OPF tradicional. No entanto, utilizam-se os conjuntos Z_1 e Z_3 sem redução de dimensionalidade. Todo esse processo é repetido 20 vezes para computar o valor de acurácia e desvio padrão, bem como o número médio de atributos retornados por cada método de redução de dimensionalidade. Um resumo das configurações da metodologia empregada no trabalho pode ser verificado na Tabela 3.

Tabela 3 – Resumo dos parâmetros empregados no trabalho.

Divisão da base	30% para treino, 30% para validação e 40% para teste
Número de realizações	20
Número de gerações	No máximo 30 ou até atingir o critério de parada
Tamanho da população	100

Fonte: próprio autor.

5.2 Resultados

As subseções a seguir apresentam os resultados obtidos pelos métodos empregados neste trabalho, referentes a taxa de acerto e o número de atributos medidos durante 20 realizações para cada conjunto de dados presentes na Tabela 2. Note que nas figuras e tabelas podem ser visualizados somente os valores do número de atributos dos métodos propostos e das meta-heurísticas (BCS, BWWO e BBA). Isso ocorre porque o número de componentes e discriminantes utilizados no PCA e LDA, respectivamente, são iguais ao número de atributos selecionados pelo BWFO na Seção 5.3, o mesmo ocorre para o PCA e LDA que possuem o mesmo número de atributos iguais ao FSBWFO na Seção 5.4 e pelo FPBWFO na Seção 5.5.

5.3 BWFO (Proposta 1)

Os testes realizados para o primeiro método proposto, o BWFO leva em consideração a configuração da Tabela 4, utilizando os conjuntos de dados descritos na Tabela 2. Em busca dos melhores parâmetros para as meta-heurísticas BCS, BWFO, BBA e o próprio BBWFO, foram realizadas pesquisas em grade (*grid search*) para cada método, assim os parâmetros descritos na tabela foram os que tiveram o melhor resultado com base na taxa de acerto. Pode-se observar na Tabela 5 os resultados obtidos para o BWFO e os demais métodos descritos na metodologia, juntamente com o OPF sem seleção de características. Vale ratificar que o número de componentes e discriminantes, respectivamente para o PCA e o LDA, é fornecido pelo número de atributos selecionadas pelo BWFO. No Apêndice A, Figura 16, é apresentado a evolução das soluções do método ao longo de 30 iterações. A Tabela 5 apresenta a taxa de acerto (ACC) de cada modelo, bem como o taxa de redução de atributos (RA) das meta-heurísticas.

Tabela 4 – Resumo dos parâmetros de cada método para os testes do BWFO.

Métodos	Parâmetros
BWFO-OPF	$p_l = 0,3$ e $p_e = 0,7$
BCS-OPF	$\alpha = 0,1$ e $P_\alpha = 0,9$
BWFO-OPF	$kMax = \min(12, \text{dimensão}/2)$, $hMax = 3$ e $\alpha = 0,01$
BBA-OPF	$\alpha = 0,9$ e $\gamma = 0,9$
PCA-OPF	Nº de componentes = Nº de atributos selecionadas pelo BWFO
LDA-OPF	Nº de discriminantes = Nº de atributos selecionadas pelo BWFO

Fonte: próprio autor.

Observando a Tabela 5, é possível perceber os bons resultados em termos de acurácia para o BWFO, levando em consideração as doze bases de dados utilizadas. O BWFO obteve a maior média de acurácia nos conjuntos PDS, TKM e UBL em relação a todos os outros modelos, e considerando também o OPF sem redução de atributos. É possível observar como os métodos de seleção de características são superiores aos métodos clássicos de redução de dimensionalidade, como o PCA e LDA³, uma vez que o BWFO obteve maior valor de acurácia em dez dos doze conjuntos de dados utilizados (PID, VC, ACA, BLD, SNR, DIV, PDS, TKM, UBL e ARR).

Considerando as meta-heurísticas pode-se observar que o BWFO tem melhor desempenho em conjuntos de dados com um maior número de atributos, que é o caso dos conjuntos de dados PDS, TKM, UBL e ARR. Nos demais conjuntos o BWFO empata com o BWFO e o BBA para o conjunto de dados VC, e é superior também aos demais no conjunto de dados DM. O BWFO tem desempenho superior quando comparado ao BCS em oito dos doze conjuntos de dados, já contra o BBA e o BWFO, o BWFO se sobressai em sete conjuntos de dados.

³ Espera-se que isso ocorra, pois tais métodos não são guiados pelo desempenho de classificação.

Tabela 5 – Resultados do OPF, PCA, LDA, BCS, BWFO e BBA versus BWFO.

<i>Dataset</i>	<i>Métrica</i>	BWFO	BCS	BWFO	BBA	OPF	PCA	LDA
PID	Acc(%)	63,66	63,80	63,74	63,48	64,07	62,81	63,01
	Std(Acc)	2,55	2,38	2,54	2,73	2,84	3,41	2,70
	RA(%)	68,57	56,25	43,75	43,75			
VC	Acc(%)	78,63	78,22	78,63	78,63	77,82	74,56	74,44
	Std(Acc)	3,04	3,31	3,04	3,04	2,64	5,16	6,03
	RA(%)	43,33	44,17	43,33	43,33			
ACA	Acc(%)	81,57	83,22	81,61	81,99	78,57	75,84	78,57
	Std(Acc)	2,51	3,43	3,21	2,39	2,09	2,01	2,54
	RA(%)	55,36	74,64	50,00	60,71			
BLD	Acc(%)	56,81	56,39	56,58	56,85	57,33	53,71	54,30
	Std(Acc)	4,54	5,27	4,48	4,53	3,36	5,04	5,07
	RA(%)	44,17	49,17	43,33	45,83			
BCW	Acc(%)	93,77	94,76	94,38	94,36	94,89	94,80	94,25
	Std(Acc)	2,53	1,66	1,99	1,97	1,70	1,39	1,47
	RA(%)	46,50	51,50	43,00	47,00			
DM	Acc(%)	95,35	95,14	95,33	95,15	95,27	95,43	91,96
	Std(Acc)	1,64	2,18	1,93	1,70	1,18	1,39	2,84
	RA(%)	50,00	59,56	46,76	49,71			
SNR	Acc(%)	76,70	75,44	77,17	77,09	77,24	76,69	60,65
	Std(Acc)	4,86	3,79	5,18	5,28	4,73	5,76	7,36
	RA(%)	58,67	66,00	52,83	47,83			
DIV	Acc(%)	97,36	97,21	97,14	97,53	97,58	97,50	84,21
	Std(Acc)	1,31	1,82	1,67	1,68	1,46	1,61	3,63
	RA(%)	59,81	72,22	50,93	52,04			
PDS	Acc(%)	76,51	75,39	75,31	76,09	75,65	75,13	59,91
	Std(Acc)	2,83	2,65	2,94	3,71	3,71	3,77	3,67
	RA(%)	57,18	71,40	50,39	66,21			
TKM	Acc(%)	75,27	72,44	74,40	74,39	74,59	73,72	67,12
	Std(Acc)	1,88	1,90	2,62	3,11	3,06	2,50	3,55
	RA(%)	54,10	65,40	45,80	49,50			
UBL	Acc(%)	83,27	83,07	83,26	82,94	82,31	82,15	72,74
	Std(Acc)	1,95	2,13	1,89	1,87	2,00	2,09	5,92
	RA(%)	56,29	72,76	51,12	62,99			
ARR	Acc(%)	61,23	61,78	60,86	60,03	58,99	59,12	58,46
	Std(Acc)	4,92	3,45	3,93	3,37	4,58	4,41	4,41
	RA(%)	57,01	71,85	51,02	68,30			

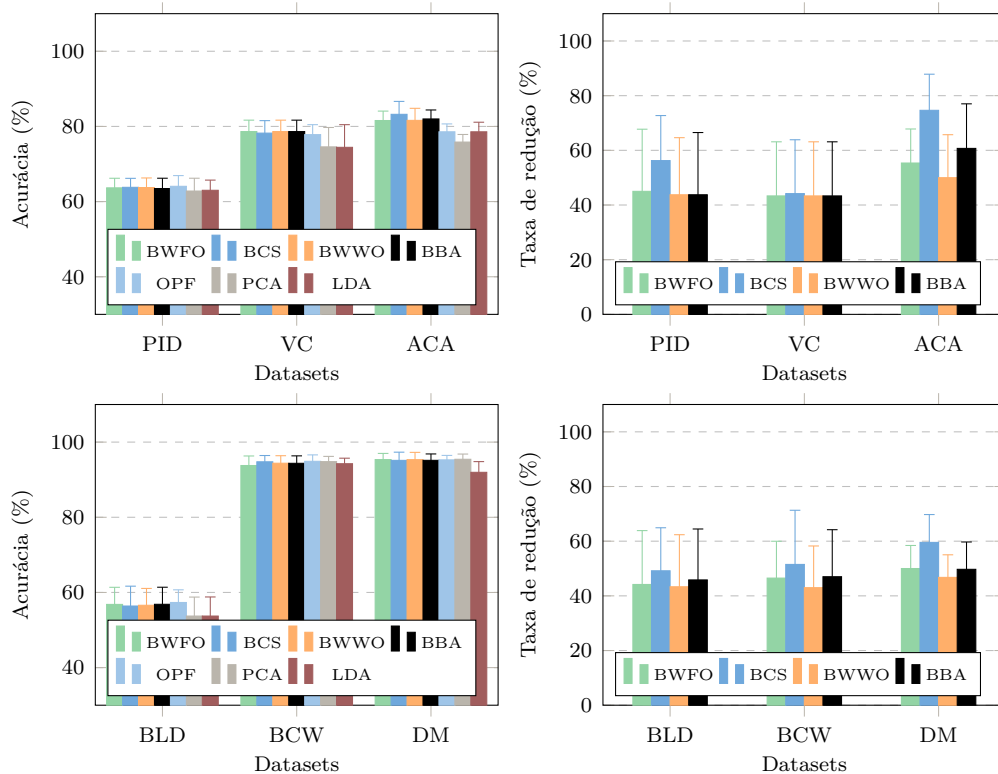
Fonte: próprio autor

No quesito redução de dimensionalidade o BWFO teve uma taxa redução menor em relação aos concorrentes, a pesquisa realizada pelo BWFO é exploratória e busca diversidade nas soluções com o operador de turbulência, assim, suas soluções candidatas têm mais variedade em seus valores. O BWFO reduz mais atributos em relação ao BWFO, visto que o mesmo tem características de busca semelhante ao BWFO. As meta-heurísticas BCS e BBA tem mecanismos que fazem com que os algoritmos realizem buscas removendo

mais atributos em suas soluções. No BBA, os parâmetros de taxa de pulso de emissão e volumes são ajustados no decorrer de sua execução e, assim, fazem com que soluções com menos atributos possam ser gerados a cada iteração. Já para o BCS, nos testes realizados para a atualização das soluções, pequenos valores são gerados para a aplicação do operador e faz com que menos atributos sejam selecionados. Assim, o BCS, realiza uma busca exploratória no espaço de busca, mas constantemente reduzindo o número de atributos selecionados.

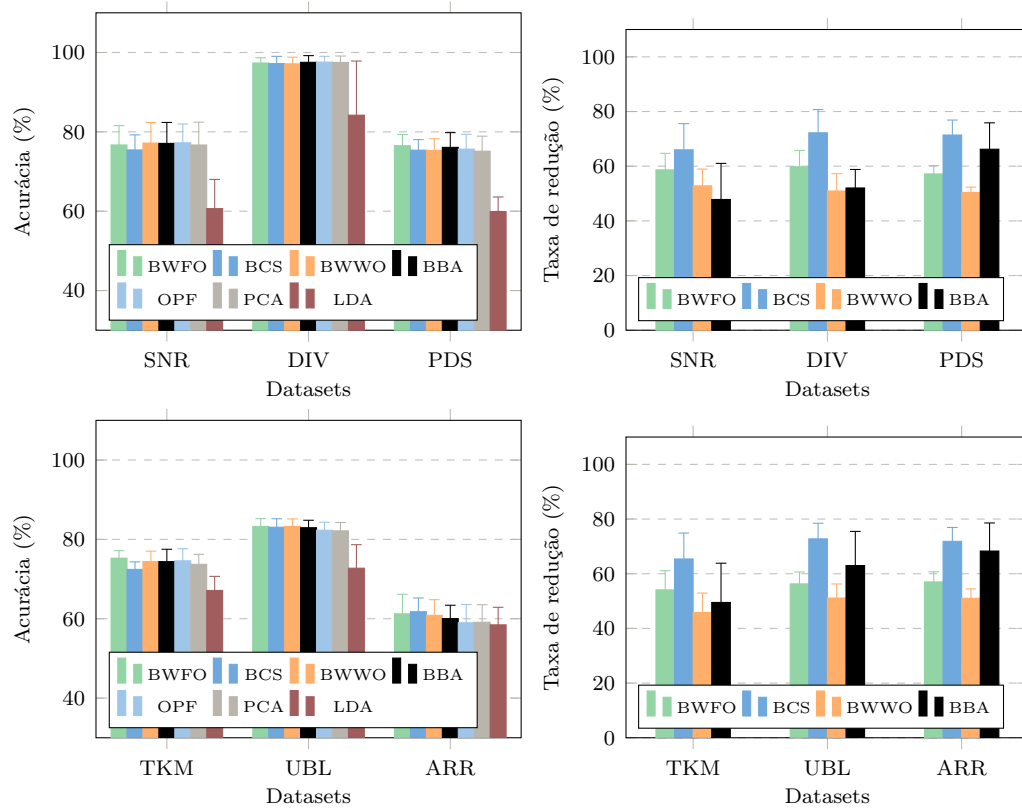
As Figura 11 e 12 apresentam os resultados da Tabela 5 em forma de gráfico de barras. Nos gráficos é possível visualizar com mais clareza os resultados para os métodos utilizados nas simulações computacionais. Com base nos resultados é possível inferir que o BWFO tem um bom desempenho, obtendo bons resultados em relação as demais meta-heurísticas. Em contrapartida, apesar de obter o foco principalmente na taxa de assertividade, conforme sua função de aptidão, a pesquisa realizada pelo BWFO atinge bons resultados sem levar em consideração de forma efetiva a redução de atributos. Portanto, podemos considerar o BWFO uma boa alternativa para seleção de atributos em razão de seu desempenho.

Figura 11 – Resultados do OPF, PCA, LDA, BWFO e BBA versus BWFO com base na Tabela 5 para os conjuntos de dados PID, VC, ACA, BLD, BCW e DM.



Fonte: próprio autor.

Figura 12 – Resultados do OPF, PCA, LDA, BWFO e BBA versus BWFO com base na Tabela 5 para os conjuntos de dados SNR, DIV, PDS, TKM, UBL e ARR.



Fonte: próprio autor.

A Tabela 6 apresenta a média de tempos de classificação sobre o conjunto Z_3 em segundos, utilizando os conjuntos já reduzidos após a fase de busca, os resultados levam em consideração o tempo de classificação do OPF.

Tabela 6 – Média de tempos de classificação no conjunto Z_3 em segundos para os métodos avaliados e o BWFO.

<i>Dataset</i>	BWFO	BCS	BWFO	BBA	OPF	PCA	LDA
PID	0,1070	0,1073	0,1128	0,1075	0,1143	0,1088	0,1079
VC	0,0193	0,0189	0,0189	0,0188	0,0191	0,0189	0,0189
ACA	0,0915	0,0878	0,0922	0,0905	0,0928	0,0907	0,0907
BLD	0,0221	0,0219	0,0220	0,0221	0,0232	0,0219	0,0219
BCW	0,0896	0,0897	0,0899	0,0900	0,0908	0,0890	0,0900
DM	0,0280	0,0284	0,0283	0,0280	0,0289	0,0287	0,0282
SNR	0,0090	0,0089	0,0091	0,0092	0,0096	0,0090	0,0092
DIV	0,0068	0,0066	0,0068	0,0067	0,0070	0,0065	0,0071
PDS	0,1917	0,1704	0,2006	0,1775	0,2701	0,1729	0,1937
TKM	0,0362	0,0357	0,0364	0,0365	0,0379	0,0366	0,0361
UBL	0,1102	0,1063	0,1107	0,1086	0,1275	0,1093	0,1111
ARR	0,0519	0,0485	0,0538	0,0488	0,0627	0,0512	0,0515

Fonte: próprio autor.

A Tabela apresenta também o tempo de classificação do OPF sem seleção de atributos. Pode-se observar que o BCS leva vantagem em tempos de execução de classificação, visto a redução de dimensionalidade realizada pelo a meta-heurística ser maior.

5.4 FSBWFO (Proposta 2)

Esta subseção visa avaliar o segundo método proposto, denominado otimizador de fluxo de água binário de tamanho fixo (FSBWFO), o FSBWFO é um método que buscar realizar as tarefa de seleção de características fixando um tamanho pré definido de atributos a serem selecionados. A execução dos testes foi realizada com os demais métodos de redução de dimensionalidade, a saber: BCS, BWFO, BBA, PCA e LDA. São considerados ainda os resultados para o classificador OPF sem redução de dimensionalidade.

Em vista disso, nos testes realizados para o método FSBWFO foram definidos valores para a taxa de poda θ , com valores de 50%, 60% e 70% para remoção de atributos. A Tabela 7 apresenta as configurações para os parâmetros de cada método empregado na simulação. Em relação ao número de componentes e discriminantes, tanto para o PCA como para o LDA, respectivamente, foi definido como o número de atributos obtidos para o FSBWFO com $\theta = 60\%$, ou seja, tanto o PCA como o LDA, selecionam 40% dos componentes e discriminantes com maior informação sobre o conjunto de dados.

Tabela 7 – Resumo dos parâmetros de cada método para os testes do FSBWFO.

Métodos	Parâmetros
FPBWFO-OPF ₁	$p_l = 0, 2$, $p_e = 0, 8$ e $\theta = 50\%$
FPBWFO-OPF ₂	$p_l = 0, 2$, $p_e = 0, 8$ e $\theta = 60\%$
FPBWFO-OPF ₃	$p_l = 0, 2$, $p_e = 0, 8$ e $\theta = 70\%$
BCS-OPF	$\alpha = 0, 1$ e $P_\alpha = 0, 9$
BWFO-OPF	$kMax = \min(12, \text{dimensão}/2)$, $hMax = 3$ e $\alpha = 0, 01$
BBA-OPF	$\alpha = 0, 9$ e $\gamma = 0, 9$
PCA-OPF	Nº de componentes = Nº de atributos selecionadas pelo FSBWFO ₂
LDA-OPF	Nº de discriminantes = Nº de atributos selecionadas pelo FSBWFO ₂

Fonte: próprio autor.

A Tabela 8 apresenta os resultados para os métodos. O método proposto obteve resultados interessantes com $\theta = 50\%$, apresentando resultados superiores aos outros métodos nos conjuntos de dados SNR, DIV, PDS, e UBL. O FSBWFO também apresentou bom desempenho em termos de acurácia nos conjuntos de dados PID e TKM com $\theta = 60\%$, e esteve a frente dos demais métodos no conjunto de dados ARR, dessa vez, utilizando $\theta = 70\%$. É possível observar e ter uma ideia sobre a quantidade de informações relevantes para cada conjunto de dados, levando em consideração os resultados de redução de dimensionalidade das meta-heurísticas. Observando o conjunto PID, o FSBWFO obteve um bom resultado, tendo removido 60% dos atributos do conjuntos de dados, o BCS obtém um resultado um pouco similar, pois removeu 55% do atributos.

Tabela 8 – Resultados do OPF, PCA, LDA, BCS, BWFO e BBA versus FSBWFO.

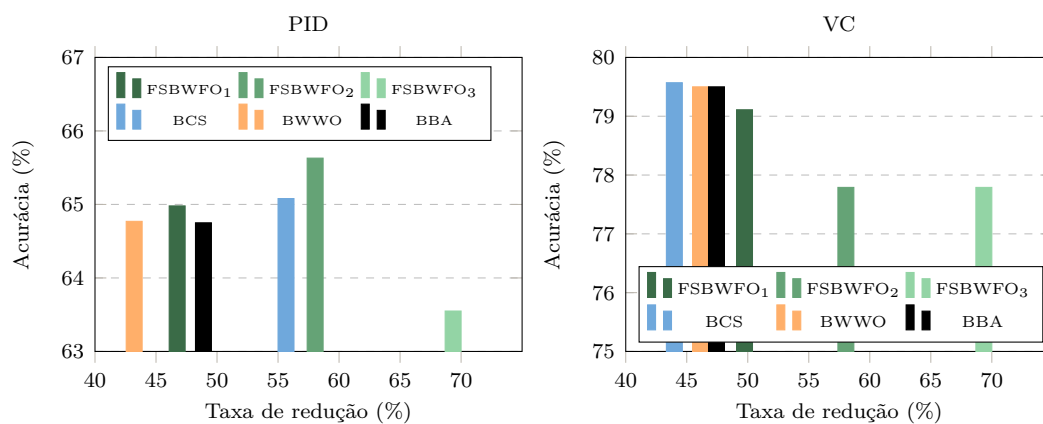
Dataset	Métrica	FSBWFO			BCS	BWFO	BBA	OPF	PCA	LDA
		$\theta = 70\%$	$\theta = 60\%$	$\theta = 50\%$						
PID	Acc(%)	63,55	65,63	64,98	65,08	64,77	64,75	64,51	62,00	63,55
	Std(Acc)	2,89	2,83	3,21	3,54	2,64	2,83	1,97	3,05	3,07
	RA(%)	70,00	60,00	50,00	55,00	41,25	45,62			
VC	Acc(%)	77,79	77,79	79,11	79,57	79,50	79,50	78,49	66,72	74,50
	Std(Acc)	3,41	3,41	3,13	2,95	2,91	2,91	2,56	2,98	7,44
	RA(%)	70,00	60,00	50,00	43,33	44,17	44,17			
ACA	Acc(%)	83,25	81,21	80,87	84,33	82,28	82,14	78,12	75,63	77,75
	Std(Acc)	2,42	1,61	1,94	2,66	2,09	3,06	2,75	1,96	2,60
	RA(%)	70,00	60,00	50,00	77,50	54,64	59,64			
BLD	Acc(%)	54,63	54,63	57,79	57,91	57,49	57,49	57,60	52,10	55,09
	Std(Acc)	4,33	4,33	3,80	3,98	3,85	3,85	2,86	3,53	4,92
	RA(%)	70,00	60,00	50,00	53,33	50,83	50,83			
BCW	Acc(%)	93,54	93,89	93,81	94,19	94,23	94,21	93,96	94,36	92,99
	Std(Acc)	2,16	1,23	1,68	1,61	1,36	1,34	1,52	1,22	1,84
	RA(%)	70,00	60,00	50,00	55,50	47,00	44,50			
DM	Acc(%)	95,91	95,51	95,42	94,51	96,28	95,19	95,50	94,87	89,62
	Std(Acc)	2,97	1,87	1,49	1,92	1,57	2,53	1,09	1,09	2,37
	RA(%)	70,00	60,00	50,00	59,85	46,91	37,50			
SNR	Acc(%)	75,48	76,75	78,02	75,54	77,27	77,36	75,86	75,41	65,95
	Std(Acc)	5,85	3,81	5,06	4,24	4,02	4,40	4,24	4,14	6,30
	RA(%)	70,00	60,00	50,00	66,50	51,42	56,58			
DIV	Acc(%)	97,27	97,75	97,92	97,58	97,82	97,75	97,96	98,03	86,22
	Std(Acc)	1,92	1,54	1,29	1,44	1,51	1,28	1,36	1,33	9,33
	RA(%)	70,00	60,00	50,00	75,00	47,78	52,87			
PDS	Acc(%)	75,88	76,31	76,43	75,51	75,24	74,41	74,68	74,10	59,16
	Std(Acc)	3,25	3,55	2,63	3,02	3,76	3,21	4,39	4,65	2,80
	RA(%)	70,00	60,00	50,00	72,44	50,05	66,61			
TKM	Acc(%)	74,67	75,51	74,52	73,09	75,37	74,53	74,60	72,52	64,94
	Std(Acc)	2,98	2,43	2,66	2,25	2,06	3,42	1,92	3,34	3,87
	RA(%)	70,00	60,00	50,00	69,00	47,60	50,70			
UBL	Acc(%)	83,75	83,93	84,22	83,72	83,38	83,85	82,97	82,76	72,27
	Std(Acc)	1,99	2,07	2,26	1,93	2,90	1,58	1,96	1,87	4,81
	RA(%)	70,00	60,00	50,00	69,63	51,26	57,93			
ARR	Acc(%)	62,96	62,13	61,63	62,32	62,17	61,97	59,93	60,10	58,65
	Std(Acc)	4,20	3,28	2,42	4,24	3,26	3,57	3,40	3,28	3,53
	RA(%)	70,00	60,00	50,00	73,41	51,49	66,72			

Fonte: próprio autor.

Ao se analisar os resultados, é possível perceber que quando a redução é de 70%, a acurácia é afetada, e quando está aproximadamente entre 40% e 50%, os resultados são superiores ao OPF sem redução de dimensionalidade. Ao se analisar os resultados para os demais conjuntos de dados, é possível perceber comportamentos semelhantes sobre o estudo de quantidade atributos relevantes para cada conjuntos de dados. Por exemplo, para o conjunto VC, os métodos obtêm melhores resultados quando a porcentagem de atributos removidos fica aproximadamente entre 40% e 50%. Para o conjunto ACA, os melhores resultados são obtidos com redução aproximada de 70%. Já para o conjunto BLD, são obtidos bons resultados quando a quantidade atributos é reduzida pela metade. No Apêndice A, Figura 17, é apresentado a evolução das soluções do método ao longo de 30 iterações.

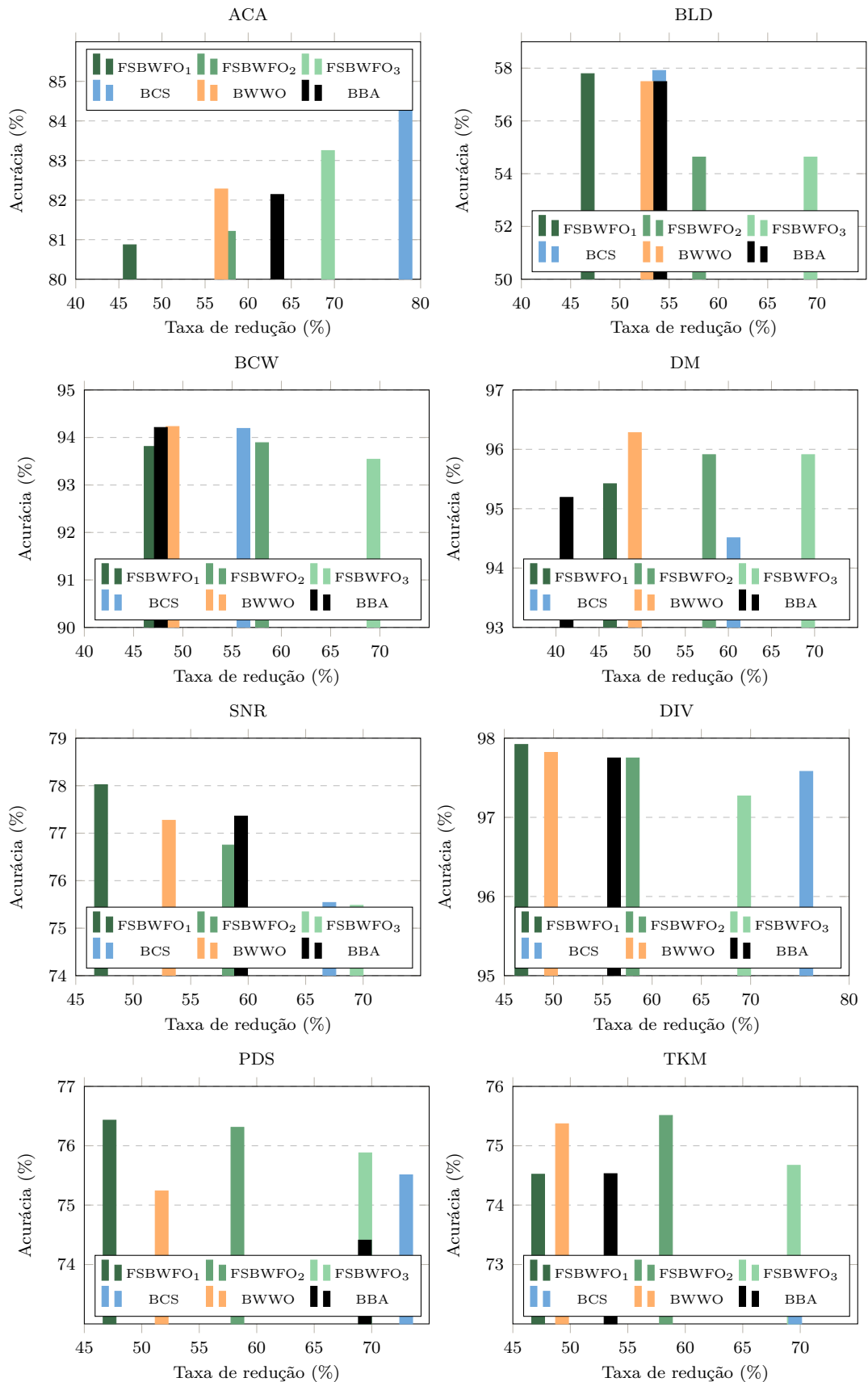
As Figuras 13, 14 e 15 apresentam a relação entre a redução de dimensionalidade e a acurácia obtida para as meta-heurísticas avaliadas com base nos resultados da Tabela 8. É interessante de ser observada a taxa de remoção de atributos para cada conjunto de dados, visto que a medida que a taxa de remoção de atributos aumenta para alguns conjuntos de dados, estes perdem informações importantes, afetando o desempenho do classificador. Por outro lado, outros conjuntos de dados quando realizado uma maior remoção do número de atributos, ocasionou em um aumento de desempenho do classificador, implicando em uma maior remoção de atributos irrelevantes e ruidosos. O FSBWFO é um método relevante para aplicações na área de internet das coisas (*internet of things*, IoT), uma vez que modelos de aprendizado de máquina necessitam ser embarcados em dispositivos com baixo recurso computacional. Assim, com o FSBWFO é possível realizar remoções de atributos levando também em consideração os requisitos computacionais das aplicações.

Figura 13 – Relação de redução de atributos e acurácia com base nos resultados da Tabela 8 para todos os métodos e os conjuntos de dados PID e VC.



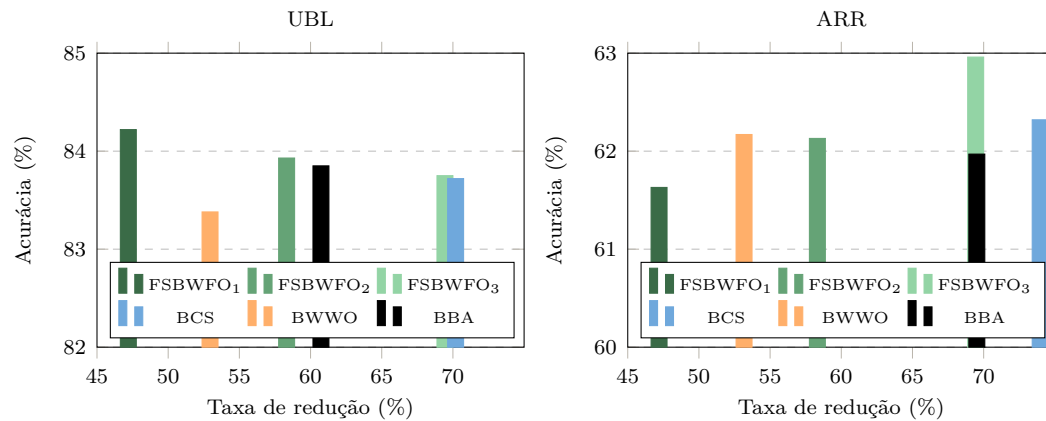
Fonte: próprio autor

Figura 14 – Relação de redução de atributos e acurácia com base nos resultados da Tabela 8 para todos os métodos e os conjuntos de dados ACA, BLD, BCW, DM, SNR, DIV, PDS e TKM.



Fonte: próprio autor.

Figura 15 – Relação de redução de atributos e acurácia com base nos resultados da Tabela 8 para todos os métodos e os conjuntos de dados UBL e ARR.



Fonte: próprio autor.

A Tabela 9 apresenta a média de tempos de classificação sobre o conjunto Z_3 em segundos. Pode-se observar que conforme a Tabela 8, o BCS obteve maiores taxa de redução de dimensionalidade nos conjuntos de dados ACA, DIV, PDS e ARR. Estes resultados explicam o fato de que as soluções do BCS obtiveram os melhores tempos de classificação nestes conjuntos de dados, como descrito na Tabela 9.

Tabela 9 – Média de tempos de classificação no conjunto Z_3 em segundos para os métodos avaliados e o FSBWFO.

<i>Dataset</i>	FSBWFO			BCS	BWWO	BBA	OPF	PCA	LDA
	$\theta = 70\%$	$\theta = 60\%$	$\theta = 50\%$						
PID	0,1120	0,1121	0,1123	0,1143	0,1144	0,1133	0,1169	0,1143	0,1136
VC	0,0194	0,0193	0,0193	0,0194	0,0194	0,0193	0,0195	0,0193	0,0192
ACA	0,0912	0,0922	0,0917	0,0803	0,0933	0,0923	0,0948	0,0912	0,0919
BLD	0,0234	0,0234	0,0235	0,0221	0,0217	0,0219	0,0239	0,0234	0,0234
BCW	0,0891	0,0890	0,0903	0,0907	0,0911	0,0904	0,0918	0,0896	0,0896
DM	0,0284	0,0285	0,0288	0,0290	0,0290	0,0290	0,0295	0,0279	0,0279
SNR	0,0091	0,0092	0,0093	0,0090	0,0092	0,0096	0,0097	0,0091	0,0093
DIV	0,0063	0,0062	0,0062	0,0061	0,0063	0,0062	0,0066	0,0061	0,0062
PDS	0,1577	0,1725	0,1867	0,1561	0,1853	0,1653	0,2479	0,1594	0,1609
TKM	0,0336	0,0342	0,0338	0,0338	0,0341	0,0341	0,0353	0,0341	0,0337
UBL	0,0983	0,1009	0,1018	0,0986	0,1028	0,1005	0,1156	0,1008	0,1003
ARR	0,0472	0,0478	0,0522	0,0457	0,0517	0,0469	0,0600	0,0466	0,0463

Fonte: próprio autor.

5.5 FPBWFO (Proposta 3)

As simulações computacionais realizadas para o terceiro método proposto, o otimizador de fluxo de água com poda de características (FPBWFO), leva em consideração as configurações apresentadas na Tabela 10, tendo sido aplicados aos conjuntos de dados descritos na Tabela 2. Em busca dos melhores parâmetros para a meta-heurística, foram realizadas pesquisas em grade (*grid search*) para o método, assim os parâmetros descritos na tabela foram os que obtiveram o melhor resultado com base na acurácia fornecido pelo classificador OPF.

Tabela 10 – Resumo dos parâmetros de cada método para os testes do FPWBFO.

Métodos	Parâmetros
FPBWFO-OPF	$p_l = 0,4$ e $p_e = 0,6$
BCS-OPF	$\alpha = 0,1$ e $P_a = 0,9$
BWWO-OPF	$kMax = \min(12, \text{dimensão}/2)$, $hMax = 3$ e $\alpha = 0,01$
BBA-OPF	$\alpha = 0,9$ e $\gamma = 0,9$
PCA-OPF	Nº de componentes = Nº de atributos selecionadas pelo FPBWFO
LDA-OPF	Nº de discriminantes = Nº de atributos selecionadas pelo FPBWFO

Fonte: próprio autor.

A Tabela 11 apresenta os resultados obtidos para o FPBWFO e os demais métodos descritos na metodologia juntamente com o OPF sem redução de dimensionalidade. Em relação aos métodos clássicos, o número de componentes e discriminantes, respectivamente para o PCA e o LDA, são fornecidos pelo número de atributos selecionadas pelo FPBWFO para cada base de dados. Conforme os resultados pode-se observar que a redução de atributos realizada pelo FPBWFO foram maiores quando comprados as meta-heurísticas BWWO e BBA. Observa-se que para alguns conjuntos de dados os valores ficam próximo até mesmo do BCS. Esses resultados significativos são motivados pela melhoria dos operadores implementados no FPBWFO comparativamente ao BWFO. Destaca-se aqui principalmente a alteração realizada no operador de fluxo turbulento binário, que dessa vez, reduz com uma maior frequência a quantidade de atributos das soluções.

O desempenho em termos de acurácia também são melhores, o FPBWFO tem melhores acurácias em relação aos demais métodos em quatro conjuntos de dados, sendo eles nos conjuntos PID, DM, UBL e ARR, empatando com o BWWO e BBA no conjunto de dados VC. A superioridade das meta-heurísticas sobre os métodos clássicos de redução de dimensionalidade continua relevante, o próprio FPBWFO obteve melhores acurácias do que o PCA e LDA em dez dos doze conjuntos de dados.

Tabela 11 – Resultados do OPF, PCA, LDA, BCS, BWFO e BBA versus FPBWFO.

<i>Dataset</i>	<i>Métrica</i>	FPBWFO	BCS	BWFO	BBA	OPF	PCA	LDA
PID	Acc(%)	63,75	63,72	63,44	64,27	64,64	62,28	62,88
	Std(Acc)	2,16	2,73	2,41	2,64	2,16	3,51	2,99
	RA(%)	53,12	53,75	51,88	48,75			
VC	Acc(%)	79,49	78,53	79,49	79,49	78,57	74,45	75,07
	Std(Acc)	2,90	3,64	2,90	2,90	2,92	5,41	4,29
	RA(%)	42,50	46,67	42,50	44,50			
ACA	Acc(%)	82,51	83,91	81,27	82,78	77,69	75,90	78,51
	Std(Acc)	3,40	3,61	3,54	3,33	2,46	2,27	2,52
	RA(%)	65,36	78,21	55,00	61,79			
BLD	Acc(%)	58,91	58,83	58,94	58,94	58,47	53,59	58,48
	Std(Acc)	2,53	2,44	2,47	2,47	3,22	2,81	4,29
	RA(%)	40,83	42,50	41,67	41,67			
BCW	Acc(%)	94,31	94,30	94,15	94,28	94,43	94,76	93,82
	Std(Acc)	1,55	1,56	1,58	1,90	1,81	1,62	2,22
	RA(%)	38,50	55,00	41,50	37,50			
DM	Acc(%)	95,67	94,17	95,50	95,65	95,34	95,16	93,02
	Std(Acc)	1,50	2,25	1,36	1,58	1,47	1,48	2,26
	RA(%)	43,24	60,59	42,79	43,38			
SNR	Acc(%)	75,76	76,29	77,73	76,94	76,23	76,60	63,44
	Std(Acc)	3,69	5,99	3,18	4,33	5,19	4,71	6,01
	RA(%)	60,50	71,50	50,67	57,17			
DIV	Acc(%)	97,98	97,47	97,70	97,81	98,03	97,95	86,89
	Std(Acc)	1,41	2,42	1,52	1,66	1,27	1,30	8,88
	RA(%)	51,94	71,02	51,48	48,43			
PDS	Acc(%)	76,20	76,53	75,85	76,03	76,12	75,57	60,30
	Std(Acc)	3,67	3,45	3,95	3,27	3,66	3,61	2,74
	RA(%)	62,28	69,15	49,81	61,53			
TKM	Acc(%)	74,18	73,15	73,99	75,03	74,53	72,75	66,64
	Std(Acc)	2,69	2,32	3,59	2,66	2,26	2,82	3,46
	RA(%)	59,00	64,50	49,50	52,30			
UBL	Acc(%)	83,27	82,94	82,94	82,89	82,02	82,06	73,37
	Std(Acc)	1,83	1,73	1,83	1,79	1,98	2,06	4,59
	RA(%)	63,54	72,28	49,90	62,89			
ARR	Acc(%)	62,83	62,78	61,75	62,21	60,17	60,19	59,18
	Std(Acc)	2,82	4,65	2,25	4,21	3,77	3,64	3,56
	RA(%)	72,51	73,41	50,45	68,30			

Fonte: próprio autor.

Visando entender o comportamento do método proposto em sua fase de busca, a Tabela 12 apresenta os resultados com base no valor de acurácia das melhores soluções retornadas para cada meta-heurística. Os resultados são referentes aos valores de aptidão das soluções ótimas, obtidos no fim da execução do método, e avaliadas sobre o conjunto reduzido Z_2 da solução. É possível perceber que o FPBWFO se destaca em sua fase de busca,

estando a frente na maior parte dos conjuntos de dados em relação aos demais métodos. Tais resultados são interessantes de serem observados, pois é possível ver o desempenho das soluções ótimas obtidas para cada método no final do processo de otimização. Além disso, é interessante analisar os resultados do FPBWFO sobre os conjuntos de dados ACA, BCW, PDS e TKM, no final da fase de otimização sobre o conjunto reduzido Z_2 , como descrito na Tabela 12. Apesar do FPBWFO ter obtido os melhores valores de acurácia para estes conjuntos em Z_2 , o FPWBFO não manteve o mesmo desempenho para os mesmos conjuntos de dados em Z_3 , durante a fase de validação, conforme os resultados da Tabela 11. Este comportamento pode ser causado por possíveis desbalanceamentos dos conjuntos de dados na fase de validação. Entretanto, o algoritmo obtém bons resultados, principalmente em conjuntos de dados de dimensionalidades maiores.

Tabela 12 – Média da soluções ótimas retornadas pelos métodos avaliadas sobre o conjunto Z_2 , levando em consideração a acurácia (%).

<i>Dataset</i>	FPBWFO	BCS	BWFO	BBA
PID	69,54	69,28	69,59	69,28
VC	83,95	83,79	83,95	83,95
ACA	86,84	86,57	86,84	86,28
BLD	65,62	65,69	65,92	65,92
BCW	97,64	96,97	97,60	97,44
DM	99,00	97,21	99,11	98,92
SNR	90,92	88,59	90,95	90,44
DIV	99,01	99,14	98,68	98,38
PDS	86,32	82,61	84,90	84,34
TKM	82,73	78,33	82,04	80,64
UBL	87,79	86,11	86,73	87,04
ARR	75,67	71,97	72,04	73,12

Fonte: próprio autor.

A Tabela 13 apresenta a média de tempos de classificação sobre o conjunto Z_3 em segundos. O FPBWFO obteve bons resultados em questão de tempo de classificação com base nas soluções obtidas pelo método, em que conseguiu melhores tempos de classificação nos conjuntos de dados PID, VC e ARR, empatando com o PCA na conjunto PID, e com o LDA no conjunto VC. O BCS, o método que se caracteriza por remover bastante atributos, esteve a frente nos resultados em tempos de classificação nos conjuntos de dados ACA, BLD, DM, SNR, DIV, TKM e UBL, em que esses resultados obtidos pelo BCS são motivados pela grande taxa de remoção de atributos realizadas pelo método.

Com base nos resultados apresentados, pode-se perceber que o FPBWFO tem melhor desempenho em termos de redução de atributos quando comparado ao primeiro método proposto (BWFO). O FPBWFO esteve a frente do BWFO e do BBA em oito das doze oportunidades no quesito redução de atributos. É possível visualizar bons resultados até mesmo levando consideração a acurácia obtida por suas soluções. Portanto, é válido

afirmar que o FPBWFO é também uma alternativa válida e eficaz na tarefa de seleção de características.

Tabela 13 – Média de tempos de classificação no conjunto Z_3 em segundos para os métodos avaliados e o FPBWFO.

<i>Dataset</i>	FPBWFO	BCS	BWFO	BBA	OPF	PCA	LDA
PID	0,1153	0,1154	0,1176	0,1169	0,1164	0,1153	0,1160
VC	0,0197	0,0199	0,0200	0,0198	0,0203	0,0198	0,0197
ACA	0,0930	0,0843	0,0935	0,0928	0,0965	0,0927	0,0934
BLD	0,0233	0,0225	0,0228	0,0226	0,0240	0,0234	0,0238
BCW	0,0901	0,0903	0,0901	0,0897	0,0913	0,0909	0,0903
DM	0,0289	0,0283	0,0287	0,0286	0,0298	0,0295	0,0287
SNR	0,0090	0,0089	0,0092	0,0090	0,0097	0,0092	0,0093
DIV	0,0061	0,0060	0,0061	0,0061	0,0065	0,0062	0,0067
PDS	0,1670	0,1570	0,1816	0,1666	0,2443	0,1559	0,1671
TKM	0,0328	0,0325	0,0329	0,0332	0,0340	0,0327	0,0329
UBL	0,1007	0,0991	0,1041	0,1020	0,1179	0,1019	0,1015
ARR	0,0458	0,0459	0,0505	0,0465	0,0589	0,0460	0,0459

Fonte: próprio autor.

5.6 Conclusão

Neste capítulo foram apresentados os resultados obtidos para as simulações computacionais realizadas para fins de avaliação dos métodos propostos: BWFO, FSBWFO e FPBWFO. Além disso, foi realizado também uma análise comparativa das técnicas propostas com as técnicas clássicas de redução de dimensionalidade, PCA e LDA, bem como com as meta-heurísticas BWFO, BCS, e BBA e com o classificador OPF com o conjunto de dados sem redução de dimensionalidade. Os estudos foram feitos com base nos seguintes conjuntos de dados: *Australian Credit Approval*, *Breast Cancer Wisconsin*, *Dermatology*, *Vertebral Column*, *Sonar*, *Liver Disorders*, *Pima Indians Diabetes*, *Divorce Predictors*, *Parkinson's Disease*, *Turkish Music Emotion*, *Urban Land Cover* e *Arrhythmia*.

Durante as simulações foram apresentados as métricas de desempenho obtidas de cada método, sendo expostos as análises e comparações com base nos resultados.

6 Conclusões e trabalhos futuros

Neste trabalho, foram propostos três novos métodos de seleção de características baseados na meta-heurística otimizador de fluxo de água (*water flow optimizer*, BWFO), a saber: otimizador de fluxo de água binário (*binary water flow optimizer*, BWFO); otimizador de fluxo de água binário de tamanho fixo (*fixed size binary water flow optimizer*, FSBWFO) e otimizador de fluxo de água binário com poda de características (*feature pruning binary water flow optimizer*, FPBWFO).

A primeira proposta (BWFO) tem por objetivo obter a combinação de atributos de maior relevância em um determinado conjunto de dados. Os operadores do BWFO são uma remodelagem binária dos operadores originais do WFO, são eles: operador de fluxo laminar e fluxo turbulento, visando assim, construir novos operadores para a obtenção de uma versão binária do algoritmo. Além das alterações dos operadores, cada solução (partícula de água) é representada por um vetor binário, mapeando cada atributo de um conjunto de dados em uma determinada posição do vetor, no qual os valores 1 e 0 em uma componente representa respectivamente, a presença e ausência de um atributo no conjunto de dados final.

A segunda proposta (FSBWFO) apresenta um novo parâmetro para determinar o percentual de atributos que serão removidos na solução, denominado taxa de poda de atributos θ . Deste modo, os operadores de fluxo laminar e fluxo turbulento foram remodelados para gerar soluções com a quantidade fixa de seleção de atributos. Por fim, a terceira proposta (FPBWFO), tem como diferencial a realização de otimizações que geram soluções com um número menor de atributos. Tal comportamento se deve ao novo operador de fluxo turbulento binário, que passa a explorar os espaços de busca reduzindo o número de atributos das soluções com uma maior frequência, levando em consideração também soluções com boas aptidões.

Além disso, foram apresentados alguns métodos de redução de dimensionalidade presentes na literatura, desde métodos clássicos como o PCA e o LDA, além das meta-heurísticas BCS, BBA e BWFO que foram utilizados para compor as simulações computacionais junto aos métodos propostos. Os experimentos foram conduzidos por avaliar os métodos aplicando-os em doze conjuntos de dados obtidas do repositório *UCI Machine Learning*. Os resultados expõem a eficácia dos métodos propostos e os bons resultados obtidos em termos de acurácia e redução de atributos. Nas simulações computacionais, a taxa de acerto do classificador OPF foi utilizado como função de aptidão para cada solução gerada pelos métodos avaliados.

6.1 Contribuições deste trabalho

As contribuições apresentadas nesta dissertação estão relacionadas aos temas: redução de dimensionalidade, seleção de características, algoritmos evolutivos e aprendizagem de máquina.

O trabalho contribui com três novos métodos que com base nos experimentos são válidos e eficazes para a tarefa de seleção de características. Além disso, com o método FSBWFO é possível fixar o número de atributos a serem removidos de um conjunto de dados, sendo de uma grande valia para projetos de IoT, em que há a necessidade de algoritmos rápidos e que façam um baixo uso de recurso computacional, sendo possível reduzir o número de atributos de um conjunto de dados conforme a necessidade da aplicação.

Os métodos BWFO e FPBWFO são métodos que buscam encontrar soluções que obtenham a melhor taxa de acerto com base no classificador OPF, sendo o FPBWFO uma melhoria do BWFO na questão de redução de atributos. O operador de fluxo turbulento do BWFO contribui na geração de soluções com novas combinações de atributos dentro espaço de busca. Já o operador turbulento do FPBWFO contribui na geração de soluções com um baixo número de seleção de atributos. Ambos os operadores têm comportamentos interessantes e que podem ser incorporados em outras meta-heurísticas binárias. Os métodos propostos são meta-heurísticas de fácil implementação e que encontram ótimas soluções.

6.2 Trabalhos futuros

Como trabalhos futuros pretende-se a realização de mais experimentos com outras meta-heurísticas presentes na literatura que abordam o tema. Almeja-se também a realização de mais experimentos com outros conjuntos de dados de alta dimensionalidade, além da utilização de uma função-objetivo que avalie os métodos levando em consideração tanto a taxa de acerto do classificador como também o número de seleção de atributos das soluções.

Outro trabalho a ser sugerido é a criação de uma versão binária do WFO para realizar a seleção de padrões de um conjunto de dados, visto que tal técnica é uma boa alternativa para a redução de custo computacional de classificadores. Também pode ser sugerido uma nova versão do WFO que vise tanto a seleção de padrões como também a de atributos de um conjunto de dados. Além disso, pode-se sugerir meta-heurísticas que possam gerar boas soluções iniciais dentro do espaço de busca, visto que a iniciação do método com uma boa população pode acelerar o processo de otimização.

Referências

ABBOTT, M. B.; MINNS, A. W. *Computational Hydraulics (2nd ed.)*. London: Routledge, 2017. ISBN 9781351949859. Citado na página 53.

ABDI, H.; WILLIAMS, L. J. Principal component analysis. *Wiley interdisciplinary reviews: computational statistics*, Wiley Online Library, v. 2, n. 4, p. 433–459, 2010. Citado 2 vezes nas páginas 28 e 41.

BANU, A. S.; GANESH, S. H. A study of feature selection approaches for classification. p. 1–4, 03 2015. Citado na página 33.

BIANCHI, L. et al. A survey on metaheuristics for stochastic combinatorial optimization. *Natural Computing*, v. 8, p. 239–287, 2008. Citado 2 vezes nas páginas 28 e 44.

BISHOP, C. M. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Berlin, Heidelberg: Springer-Verlag, 2006. ISBN 0387310738. Citado na página 28.

BLUM, C.; ROLI, A. Metaheuristics in combinatorial optimization: Overview and conceptual comparison. *ACM Comput. Surv.*, v. 35, p. 268–308, 01 2001. Citado na página 44.

CHANDRASHEKAR, G.; SAHIN, F. A survey on feature selection methods. *Computers & Electrical Engineering*, v. 40, n. 1, p. 16 – 28, 2014. ISSN 0045-7906. 40th-year commemorative issue. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S0045790613003066>>. Citado na página 27.

COSTA, L. da S.; ROCHA NETO, A. R. da. *Novas técnicas de poda no classificador Floresta de Caminhos Ótimos via Algoritmos Genéticos*. Dissertação (Mestrado), 2020. Citado 2 vezes nas páginas 39 e 41.

DASH, M.; LIU, H. Consistency-based search in feature selection. *Artificial Intelligence*, v. 151, n. 1, p. 155–176, 2003. ISSN 0004-3702. Disponível em: <<https://www.sciencedirect.com/science/article/pii/S0004370203000791>>. Citado na página 35.

FALCÃO, A. X.; STOLFI, J.; LOTUFO, R. de A. The image foresting transform: Theory, algorithms, and applications. *IEEE transactions on pattern analysis and machine intelligence*, IEEE, v. 26, n. 1, p. 19–29, 2004. Citado na página 38.

GONNOCHENKO, A. et al. Coinbot: Intelligent robotic coin bag manipulation using artificial brain. p. 67–74, 2021. Citado na página 27.

GUYON, I. et al. *Feature Extraction: Foundations and Applications (Studies in Fuzziness and Soft Computing)*. Berlin, Heidelberg: Springer-Verlag, 2006. 90 p. ISBN 3540354875. Citado na página 35.

HALL, M. Correlation-based feature selection for machine learning. *Department of Computer Science*, v. 19, 06 2000. Citado na página 35.

JOLLIFFE, I. *Principal Component Analysis*. Springer, 2002. (Springer Series in Statistics). ISBN 9780387954424. Disponível em: <https://books.google.com.br/books?id=_olByCrhjwIC>. Citado na página 41.

KAUSHIK, S. *Introduction to Feature Selection methods with an example (or how to select the right variables?)*. 2016. Disponível em: <<https://www.analyticsvidhya.com/blog/2016/12/introduction-to-feature-selection-methods-with-an-example-or-how-to-select-the-right-variables/#:~:text=Embedded%20methods%20combine%20the%20qualities,penalization%20functions%20to%20reduce%20overfitting.>> Citado na página 36.

LIU, H.; SETIONO, R. Chi2: feature selection and discretization of numeric attributes. In: *Proceedings of 7th IEEE International Conference on Tools with Artificial Intelligence*. Herndon, VA, USA: IEEE, 1995. p. 388–391. Citado na página 35.

LUO, K. Water flow optimizer: A nature-inspired evolutionary algorithm for global optimization. *IEEE Transactions on Cybernetics*, p. 1–12, 2021. Citado 6 vezes nas páginas 29, 30, 54, 55, 57 e 58.

MACÊDO, F.; BARBOSA, G.; ROCHA NETO, A. A binary water wave optimization algorithm applied to feature selection. In: *Anais do XVI Encontro Nacional de Inteligência Artificial e Computacional*. Porto Alegre, RS, Brasil: SBC, 2019. p. 448–459. ISSN 0000-0000. Disponível em: <<https://sol.sbc.org.br/index.php/eniac/article/view/9305>>. Citado 3 vezes nas páginas 29, 44 e 77.

MEKHA, P.; TEEYASUKSAET, N. Image classification of rice leaf diseases using random forest algorithm. In: *2021 Joint International Conference on Digital Arts, Media and Technology with ECTI Northern Section Conference on Electrical, Electronics, Computer and Telecommunication Engineering*. Cha-am, Thailand: IEEE, 2021. p. 165–169. Citado na página 27.

MYKHAILICHENKO, I. et al. Application of deep neural network for real-time voice command recognition. In: *2022 IEEE 3rd KhPI Week on Advanced Technology (KhPIWeek)*. Kharkiv, Ukraine: IEEE, 2022. p. 1–4. Citado na página 27.

NAKAMURA, R. Y. M. et al. Bba: A binary bat algorithm for feature selection. In: *2012 25th SIBGRAPI Conference on Graphics, Patterns and Images*. Ouro Preto, Brazil: IEEE, 2012. p. 291–297. Citado 5 vezes nas páginas 29, 48, 49, 50 e 77.

ORACLE. *O que é IoT?* 2022. Acesso em: 24 Out. 2022. Disponível em: <<https://www.oracle.com/br/internet-of-things/what-is-iot/>>. Citado na página 27.

PAPA, J. P.; FALCÃO, A. X. A new variant of the optimum-path forest classifier. In: SPRINGER. *International Symposium on Visual Computing*. Berlin, Heidelberg: Springer, 2008. p. 935–944. Citado 2 vezes nas páginas 37 e 38.

PAPA, J. P.; FALCÃO, A. X.; SUZUKI, C. T. Supervised pattern classification based on optimum-path forest. *International Journal of Imaging Systems and Technology*, Wiley Online Library, v. 19, n. 2, p. 120–131, 2009. Citado 3 vezes nas páginas 30, 37 e 38.

PAPA, J. P.; FERNANDES, S. E. N.; FALCÃO, A. X. Optimum-path forest based on k-connectivity: Theory and applications. *Pattern Recognition Letters*, Elsevier, v. 87, p. 117–126, 2017. Citado na página 38.

- PAPA, J. P.; ROSA, G. H.; PAPA, L. P. A binary-constrained geometric semantic genetic programming for feature selection purposes. *Pattern Recognition Letters*, v. 100, p. 59 – 66, 2017. ISSN 0167-8655. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S0167865517303641>>. Citado na página 28.
- PRUDHVI, R. B. et al. Feature selection using binary psogsa and radial basis network with a novel fitness function. In: *2018 9th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*. Bengaluru, India: IEEE, 2018. p. 1–6. Citado na página 29.
- QUINLAN, J. R. Induction of decision trees. *Mach. Learn.*, Kluwer Academic Publishers, USA, v. 1, n. 1, p. 81–106, mar 1986. ISSN 0885-6125. Disponível em: <<https://doi.org/10.1023/A:1022643204877>>. Citado na página 35.
- RASCHKA, S. *Linear discriminant analysis*. 2014. Acesso em: 04 dez. 2022. Disponível em: <https://sebastianraschka.com/Articles/2014_python_lda.html>. Citado na página 28.
- RASCHKA, S. *Python Machine Learning*. Birmingham, UK: Packt Publishing, 2015. ISBN 1783555130, 9781783555130. Citado 5 vezes nas páginas 28, 41, 42, 43 e 61.
- RAY, S.; ANALYTICS, B. *Beginners Guide To Learn Dimension Reduction Techniques*. 2015. Acesso em: 30 mar. 2019. Disponível em: <<https://www.analyticsvidhya.com/blog/2015/07/dimension-reduction-methods/>>. Citado na página 27.
- RODRIGUES, D. et al. A wrapper approach for feature selection based on bat algorithm and optimum-path forest. *Expert Systems with Applications*, v. 41, n. 5, p. 2250 – 2258, 2014. ISSN 0957-4174. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S0957417413007574>>. Citado na página 33.
- RODRIGUES, D. et al. Bcs: A binary cuckoo search algorithm for feature selection. In: *2013 IEEE International Symposium on Circuits and Systems (ISCAS2013)*. Beijing, China: IEEE, 2013. p. 465–468. ISSN 0271-4302. Citado 4 vezes nas páginas 29, 46, 47 e 77.
- ROSA, G. H.; PAPA, J. P.; PAPA, L. P. Feature selection using geometric semantic genetic programming. In: *Proceedings of the Genetic and Evolutionary Computation Conference Companion*. New York, NY, USA: ACM, 2017. (GECCO '17), p. 253–254. ISBN 978-1-4503-4939-0. Disponível em: <<http://doi.acm.org/10.1145/3067695.3076020>>. Citado na página 29.
- SMITH, L. I. *A tutorial on principal components analysis*. University of Otago, New Zealand, 2002. Disponível em: <http://www.cs.otago.ac.nz/cosc453/student_tutorials/principal_components.pdf>. Citado na página 41.
- TANG, L. et al. A new method combining lda and pls for dimension reduction. *PloS one*, v. 9, p. e96944, 05 2014. Citado na página 28.
- TEZCAN, C. E.; KIRAS, B.; BILGIN, G. Classification of breast cancer histopathological images with deep transfer learning methods. In: *2022 30th Signal Processing and Communications Applications Conference (SIU)*. Safranbolu, Turkey: IEEE, 2022. p. 1–4. Citado na página 27.

THARWAT, A. et al. Linear discriminant analysis: A detailed tutorial. *Ai Communications*, v. 30, p. 169–190,, 05 2017. Citado 2 vezes nas páginas 42 e 44.

VERÓNICA, B.-C.; NOELIA, S.-M.; AMPARO, A.-B. *Feature Selection for High-Dimensional Data*. Switzerland: Springer, 2015. 13,14,16 p. Citado 4 vezes nas páginas 27, 28, 33 e 35.

YANG, X.-S. Bat algorithm for multi-objective optimisation. *International Journal of Bio-Inspired Computation*, v. 3, 03 2012. Citado 2 vezes nas páginas 48 e 49.

YANG, X.-S.; DEB, S. Cuckoo search via levy flights. 03 2010. Citado na página 47.

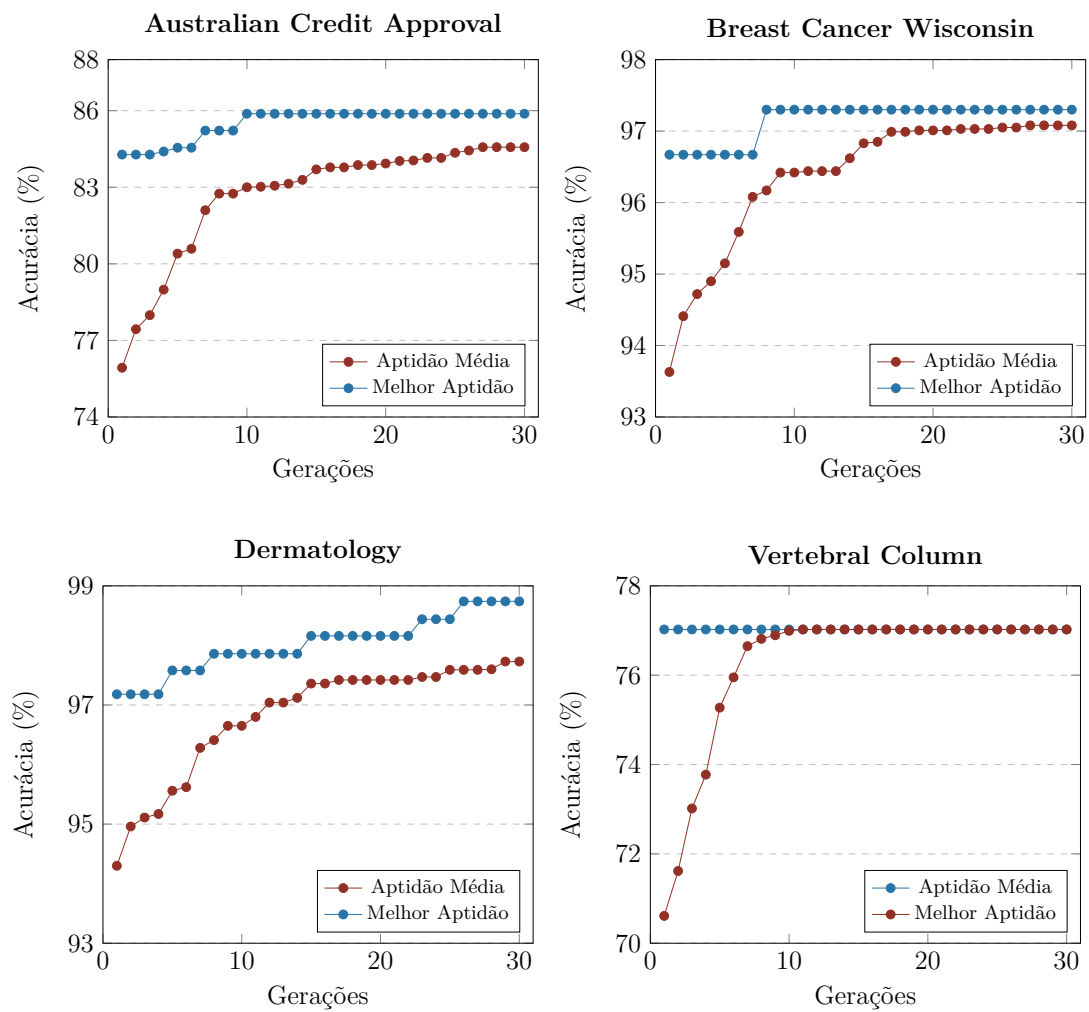
ZHENG, Y.-J. Water wave optimization: A new nature-inspired metaheuristic. *Computers & Operations Research*, v. 55, p. 1 – 11, 2015. ISSN 0305-0548. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S0305054814002652>>. Citado na página 45.

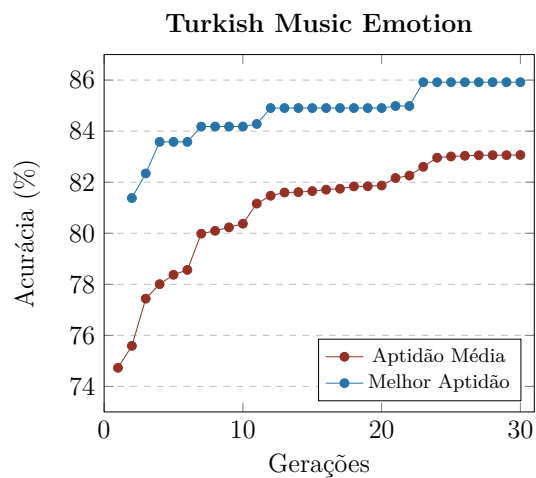
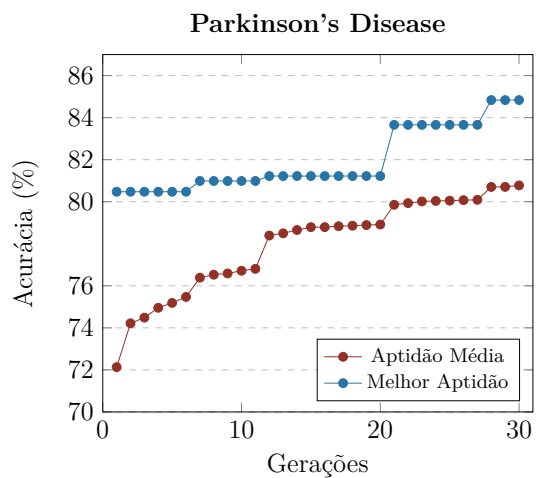
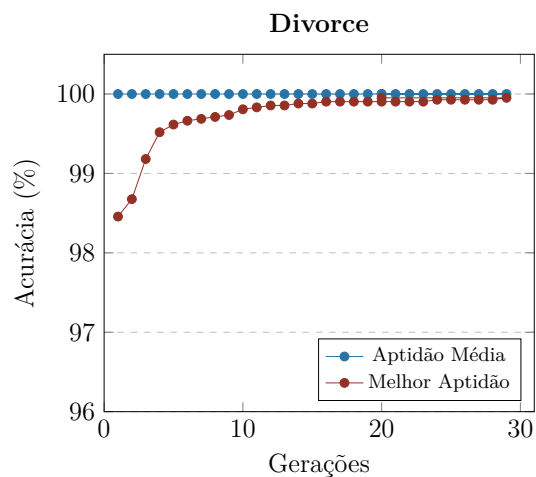
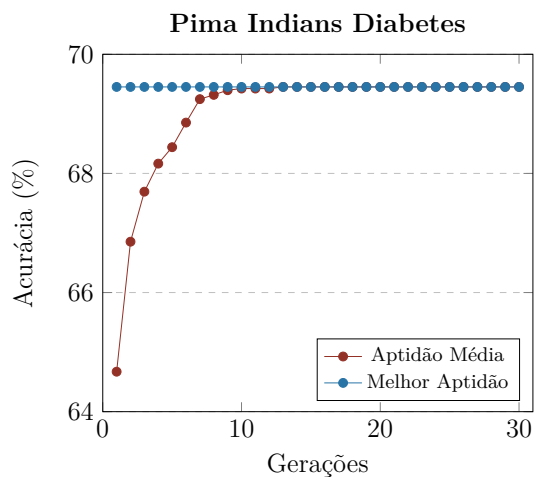
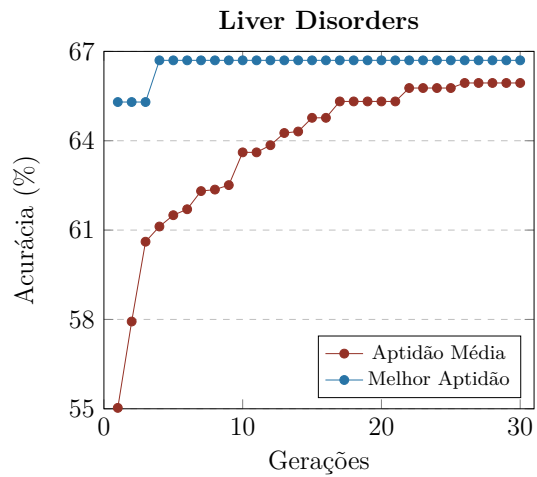
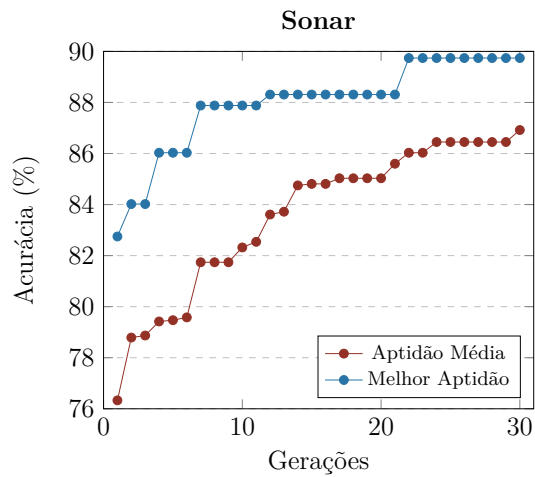
Apêndices

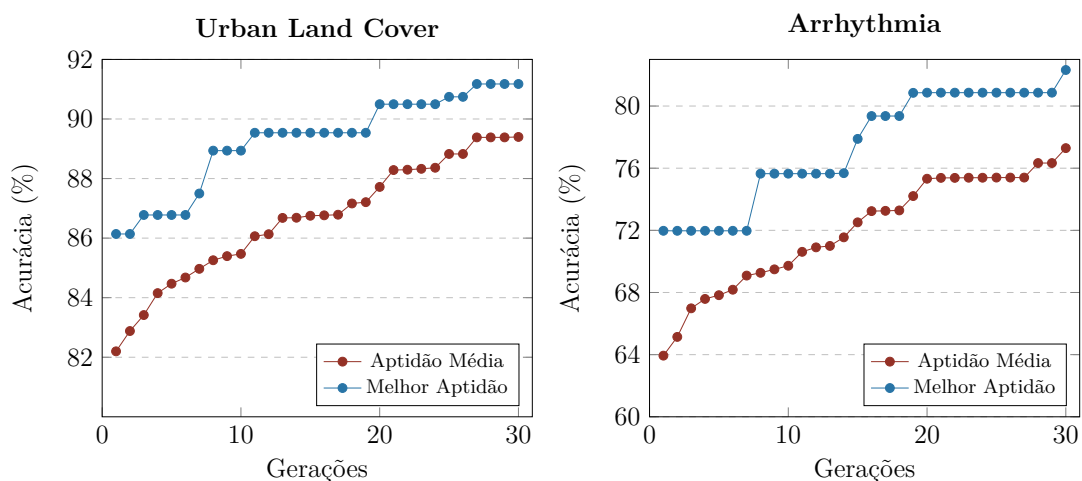
APÊNDICE A – Gráficos de Convergência

Este apêndice visa demonstrar a convergência dos algoritmos BWFO e FSBWFO, ao longo das gerações com base nas aptidões obtidas pelo algoritmo. As Figuras 16 e 17 apresentam os valores de melhor aptidão, como também o valor médio de aptidões para cada geração a partir de uma realização. O valores presente no gráfico foram obtidos a partir dos conjuntos de dados da Tabela 2, utilizando o classificador OPF.

Figura 16 – Gráfico de convergência do WFO para os conjuntos de dados.

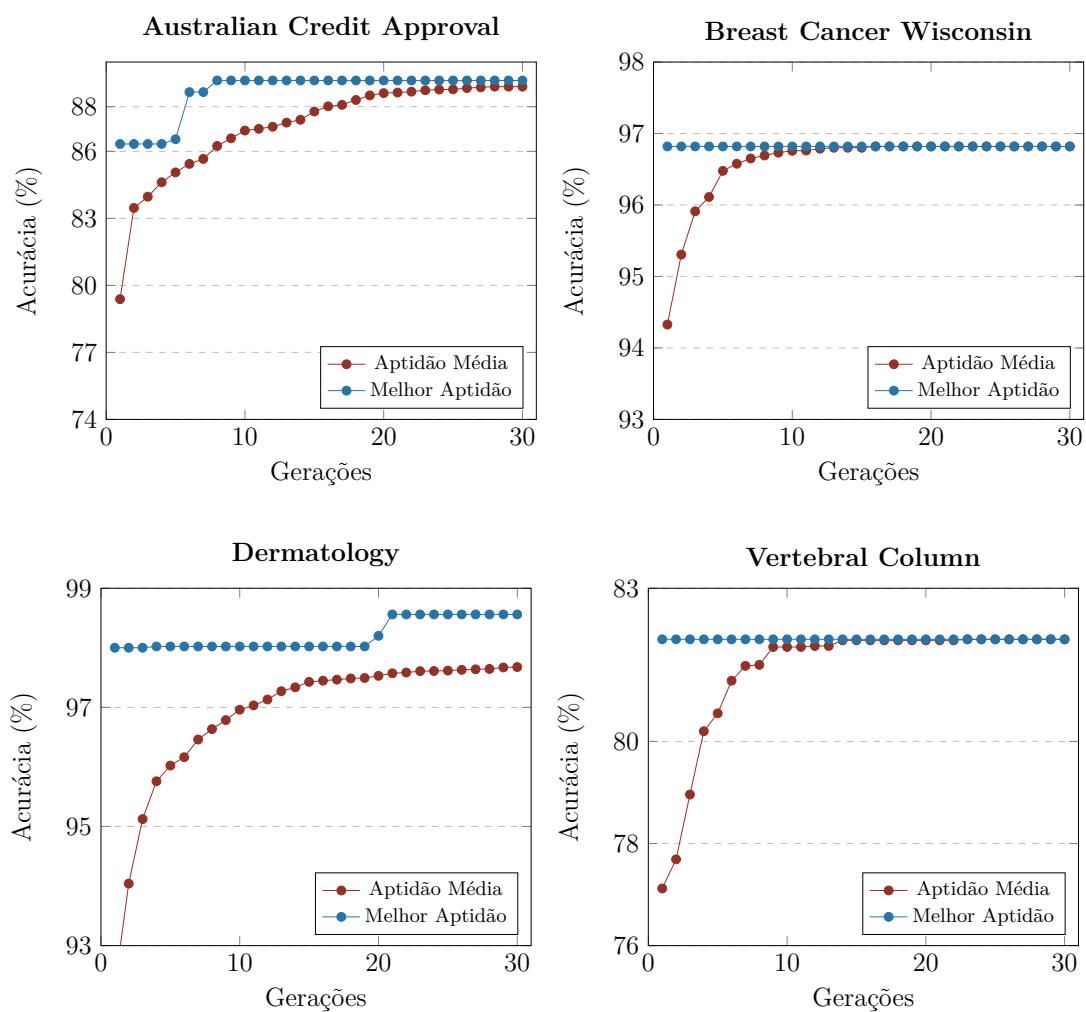


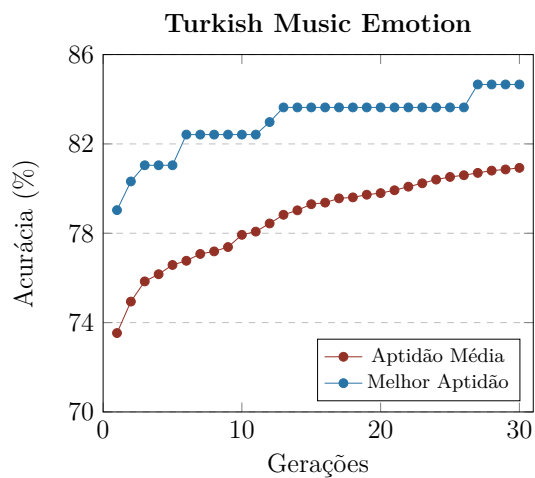
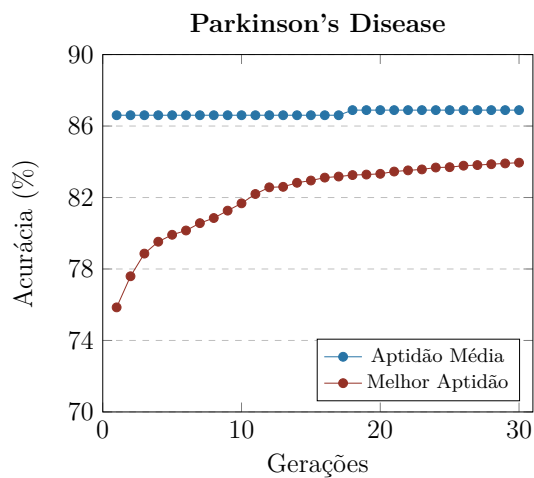
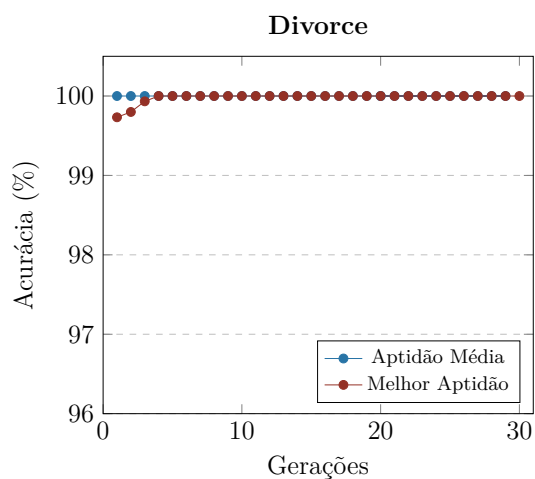
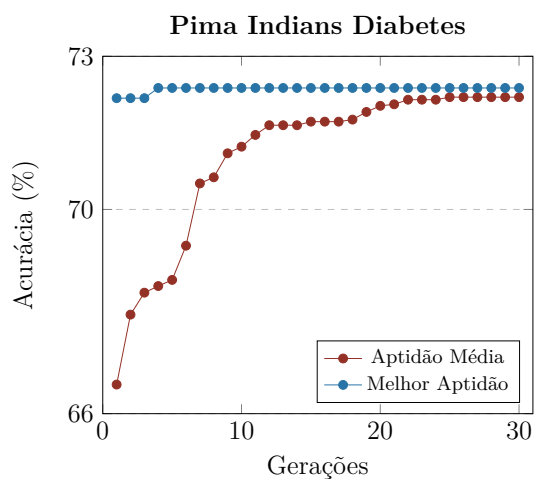
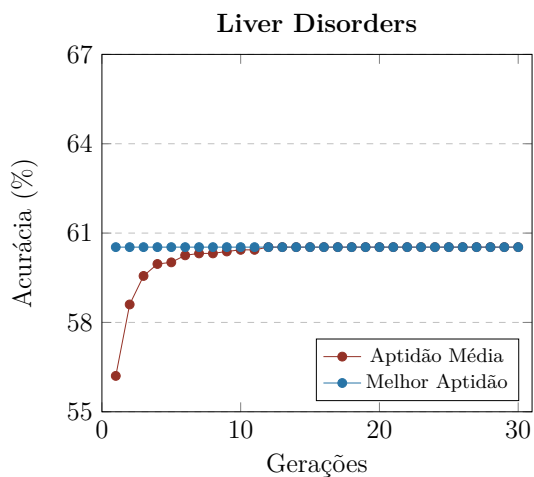
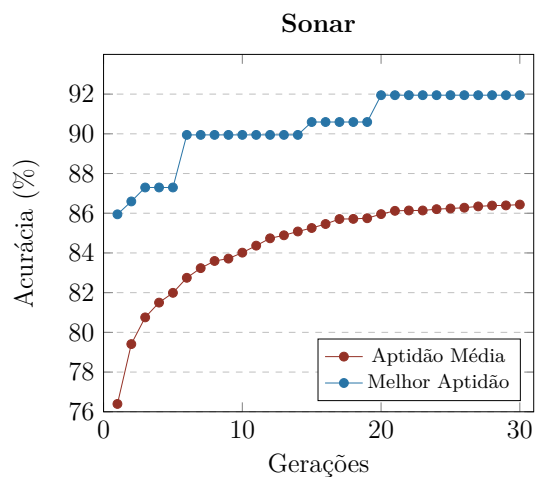


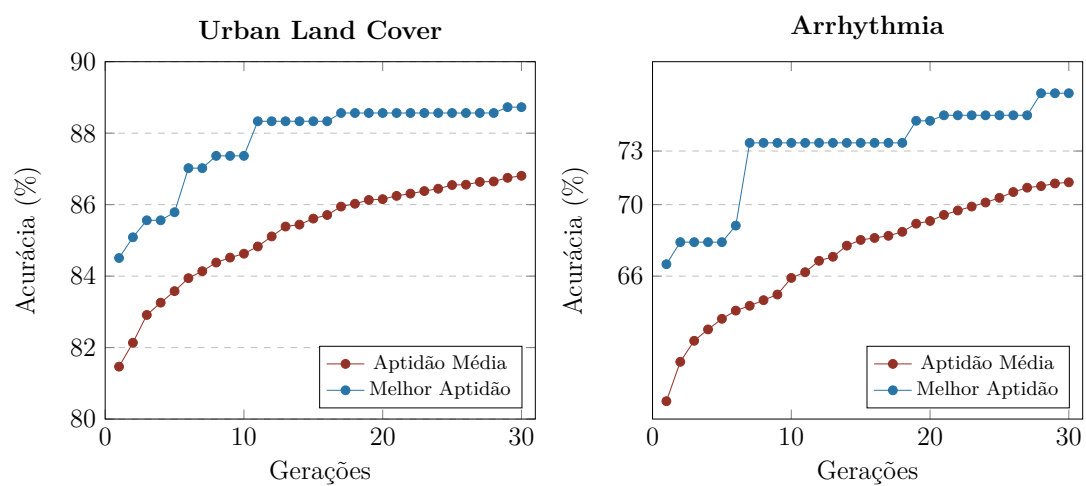


Fonte: próprio autor.

Figura 17 – Gráfico de convergência do FSBWFO para os conjuntos de dados.







Fonte: próprio autor.