

UNIVERSIDADE FEDERAL DE MINAS GERAIS
Escola de Engenharia
Programa de Pós-Graduação em Engenharia Elétrica

Felipe Augusto Rocha da Silva

**Integração de Séries Temporais Fuzzy, Representação por Imagens e
Aprendizado Federado na Classificação de Séries Temporais**

Belo Horizonte

2025

Felipe Augusto Rocha da Silva

**Integração de Séries Temporais Fuzzy, Representação por Imagens e
Aprendizado Federado na Classificação de Séries Temporais**

Tese apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Minas Gerais, como requisito parcial à obtenção do título de Doutor em Engenharia Elétrica.

Orientador: Prof. Dr. Frederico Gadelha Guimarães

Co-orientador: Prof. Dr. Ricardo Poley Martins Ferreira

Belo Horizonte

2025

S586i

Silva, Felipe Augusto Rocha da.

Integração de séries temporais fuzzy, representação por imagens e aprendizado federado na classificação de séries temporais [recurso eletrônico] / Felipe Augusto Rocha da Silva. - 2025.

1 recurso online (175 f. : il., color.) : pdf.

Orientador: Frederico Gadelha Guimarães.

Coorientador: Ricardo Poley Martins Ferreira.

Tese (doutorado) - Universidade Federal de Minas Gerais, Escola de Engenharia.

Inclui bibliografia.

1. Engenharia elétrica - Teses. 2. Análise de séries temporais - Teses. 3. Redes neurais convolucionais - Teses. I. Guimarães, Frederico Gadelha. II. Ferreira, Ricardo Poley Martins. III. Universidade Federal de Minas Gerais. Escola de Engenharia. IV. Título.

CDU: 621.3(043)



UNIVERSIDADE FEDERAL DE MINAS GERAIS

Escola de Engenharia

COLEGIADO DO CURSO DE GRADUAÇÃO / PÓS-GRADUAÇÃO EM Engenharia Elétrica

FOLHA DE APROVAÇÃO

**"Integração de Séries Temporais Fuzzy, Representação por Imagens e
Aprendizado Federado na Classificação de Séries Temporais"**

Felipe Augusto Rocha da Silva

Tese de Doutorado submetida à Banca Examinadora designada pelo Colegiado do Programa de Pós-Graduação em Engenharia Elétrica da Escola de Engenharia da Universidade Federal de Minas Gerais, como requisito para obtenção do grau de Doutor em Engenharia Elétrica.

Aprovada em 18 de agosto de 2025.

Por:

Prof. Dr. Frederico Gadelha Guimarães
DCC (UFMG) - Orientador

Prof. Dr. Ricardo Poley Martins Ferreira
DEM (UFMG) - Coorientador

Prof. Dr. Diego Furtado Silva
ICMC (USP)

Prof. Dr. Petrônio Cândido de Lima e Silva
(IFNMG)

Prof. Dr. Heitor Soares Ramos Filho
DCC (UFMG)

Prof. Dr. Cristiano Leite de Castro

DEE (UFMG)



Documento assinado eletronicamente por **Frederico Gadelha Guimaraes, Professor do Magistério Superior**, em 18/08/2025, às 14:31, conforme horário oficial de Brasília, com fundamento no art. 5º do [Decreto nº 10.543, de 13 de novembro de 2020](#).



Documento assinado eletronicamente por **Heitor Soares Ramos Filho, Professor do Magistério Superior**, em 19/08/2025, às 10:30, conforme horário oficial de Brasília, com fundamento no art. 5º do [Decreto nº 10.543, de 13 de novembro de 2020](#).



Documento assinado eletronicamente por **Ricardo Poley Martins Ferreira, Professor do Magistério Superior**, em 19/08/2025, às 13:03, conforme horário oficial de Brasília, com fundamento no art. 5º do [Decreto nº 10.543, de 13 de novembro de 2020](#).



Documento assinado eletronicamente por **Cristiano Leite de Castro, Professor do Magistério Superior**, em 19/08/2025, às 13:46, conforme horário oficial de Brasília, com fundamento no art. 5º do [Decreto nº 10.543, de 13 de novembro de 2020](#).



Documento assinado eletronicamente por **Diego Furtado Silva, Usuário Externo**, em 20/08/2025, às 09:33, conforme horário oficial de Brasília, com fundamento no art. 5º do [Decreto nº 10.543, de 13 de novembro de 2020](#).



Documento assinado eletronicamente por **Petrônio Cândido de Lima e Silva, Usuário Externo**, em 26/08/2025, às 19:33, conforme horário oficial de Brasília, com fundamento no art. 5º do [Decreto nº 10.543, de 13 de novembro de 2020](#).



A autenticidade deste documento pode ser conferida no site https://sei.ufmg.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0, informando o código verificador **4384793** e o código CRC **8F20600C**.

AGRADECIMENTOS

Agradeço, primeiramente, a Deus, pela oportunidade de concluir esta etapa tão importante da minha carreira acadêmica. Sou também grato por todas as conquistas que acompanharam este trabalho, motivando-me a explorar uma nova área que tem gerado frutos promissores, os quais acredito que permearão toda a minha trajetória profissional.

Agradeço à minha família — em especial à minha esposa, Érika; ao meu filho, Augusto; aos meus pais, Edir e Lúcia; ao meu irmão, Leonardo; e às minhas avós, Cecília e Zenaide — por todo o apoio ao longo desse caminho. Sou profundamente grato pelo tempo, paciência, esforço redobrado, orações e tantos auxílios, diretos e indiretos, que foram essenciais para que esta conquista se tornasse realidade.

Aos meus orientadores, Frederico Gadelha e Ricardo Poley, agradeço pelos inúmeros conselhos, mentorias, direcionamentos e pela paciência durante toda a jornada. Sua experiência e orientação foram fundamentais para concretizar este trabalho e me mostrar o caminho a seguir. Agradeço também aos professores que me proporcionaram o conhecimento necessário para desenvolver, analisar e interpretar os experimentos e resultados. Aos amigos que compreenderam minha ausência em alguns momentos e àqueles que caminharam comigo durante o desenvolvimento da pesquisa — especialmente Omid Orang, Fabrício Erazo, Petrônio Cândido, Pedro Barros e Marcelo Queiroz Oliveira — meu sincero obrigado pelo apoio, pelas ideias e sugestões que enriqueceram o trabalho de forma determinante.

Registro ainda meu agradecimento aos professores responsáveis pelos laboratórios nos quais tive a honra de atuar. Em primeiro lugar, ao professor Frederico Gadelha Guimarães e ao professor Cristiano Leite Castro, responsáveis pelo laboratório Machine Intelligence and Data Science (MINDS), onde iniciei minha pesquisa. Também agradeço aos professores Antônio Loureiro, Heitor Soares, Marcelo Costa e Martin Ravetti, que, juntamente com o professor Frederico Gadelha Guimarães, coordenam o FutureLab, vinculado ao Departamento de Ciência da Computação (DCC) da UFMG. Este laboratório ampliou meu conhecimento e me ofereceu oportunidades valiosas de pesquisa, incorporadas a este trabalho.

Por fim, agradeço a todos que, de maneira direta ou indireta, contribuíram para a realização e conclusão deste trabalho.

“O verdadeiro saber começa com o reconhecimento da própria ignorância.”

– Sócrates

RESUMO

Com o crescimento da quantidade de dados em diversas áreas de aplicação, inúmeros estudos e algoritmos têm sido desenvolvidos com o objetivo de oferecer soluções cada vez mais abrangentes, rápidas e precisas. Uma parcela significativa desses dados é obtida ao longo do tempo, o que caracteriza uma dependência em relação à ordem das medições. Esses dados, conhecidos como séries temporais, são de grande relevância em várias aplicações, tornando sua análise um tema de grande importância. Dentre as tarefas associadas às séries temporais, destaca-se a classificação, para a qual diversos métodos vêm sendo propostos, especialmente nas últimas décadas. Diante do grande volume de dados envolvido, muitos desses métodos buscam abordagens que visam maior eficiência e assertividade na análise. Com o avanço dos recursos computacionais, técnicas de inteligência artificial como redes neurais artificiais e lógica fuzzy têm se mostrado particularmente eficazes em tarefas de previsão, classificação e controle. Este trabalho investiga e propõe novas metodologias para a classificação de séries temporais, com ênfase na integração de abordagens baseadas em séries temporais fuzzy, representação em imagens e aprendizado federado. A primeira contribuição consiste na aplicação de medidas de similaridade diretamente sobre séries temporais fuzzificadas, visando à redução de ruídos e à valorização de padrões relevantes. A segunda abordagem propõe três estratégias distintas de transformação das séries temporais brutas e fuzzificadas em imagens, possibilitando a posterior classificação por meio de Redes Neurais Convolucionais (CNNs). A terceira contribuição, baseada na representação das séries em imagens, explora o uso de Aprendizado Federado (Federated Learning FL) para a classificação distribuída, com preservação da privacidade dos dados, demonstrando a adaptabilidade das metodologias propostas em ambientes descentralizados. A quarta contribuição, também baseada em FL e na representação das séries em imagens, aplica classificação multitarefa para a geração de um modelo global e classificação das séries temporais. Os resultados obtidos evidenciam a competitividade das abordagens frente a métodos de referência (*Benchmark*), ampliando as possibilidades de aplicação e estudo de séries temporais fuzzy, representações visuais e cenários com restrições de privacidade. Este trabalho contribui, assim, para o avanço das técnicas de classificação de séries temporais sob múltiplas perspectivas metodológicas e práticas.

Palavras-chave: Classificação de séries temporais; Séries Temporais Fuzzy; Redes Neurais Convolucionais; Representação em imagens; Aprendizado Federado.

ABSTRACT

With the growth of data across various application domains, numerous studies and algorithms have been developed to provide increasingly comprehensive, fast, and accurate solutions. A significant portion of this data is collected over time, exhibiting a dependency on the order of measurements. Such data, known as time series, play a crucial role in several applications, making their analysis a topic of substantial importance. Among the tasks associated with time series, classification stands out, and several methods have been proposed over the years, particularly in recent decades. Given the large volume of data typically involved, many of these methods seek approaches that enhance both the efficiency and accuracy of the analysis. With advances in computational resources, artificial intelligence techniques such as artificial neural networks and fuzzy logic have proven especially effective for forecasting, classification, and control tasks. This research investigates and proposes new methodologies for time series classification, emphasizing the integration of fuzzy time series approaches, image-based representations, and federated learning. The first contribution involves applying similarity measures directly to fuzzified time series, aiming to reduce noise and highlight relevant patterns. The second approach introduces three distinct strategies for transforming both raw and fuzzified time series into images, enabling subsequent classification via Convolutional Neural Networks (CNNs). The third contribution, also based on image representations, explores the use of Federated Learning (FL) for distributed classification while preserving data privacy, demonstrating the adaptability of the proposed methods in decentralized environments. The fourth contribution, also based on Federated Learning and on the image representation of time series, applies multitask classification for the generation of a global model and the classification of time series. The results obtained highlight the competitiveness of the proposed approaches in comparison with reference methods, expanding the potential for applying and studying fuzzy time series, visual representations, and privacy-constrained scenarios. This work, therefore, contributes to the advancement of time series classification techniques from multiple methodological and practical perspectives.

Keywords: Time series classification; Fuzzy Time Series; image representation; Convolutional Neural Networks; Federated Learning.

LISTA DE FIGURAS

Figura 1 – Exemplo de aplicação da Distância Euclidiana e distância DTW	27
Figura 2 – Exemplos de restrições para o algoritmo DTW	29
Figura 3 – Classificação geral dos algoritmos de medidas de similaridade discutidos.	34
Figura 4 – Ilustração esquemática do processo de fuzzificação da série temporal até sua classificação.	42
Figura 5 – Partição do Universo de Discurso com o método Grid (pyFTS)	48
Figura 6 – Exemplo de representação de uma série temporal como uma série temporal Fuzzy	49
Figura 7 – Exemplo de arquitetura de uma rede neural convolucional.	52
Figura 8 – Exemplo de aplicação de uma camada convolucional.	53
Figura 9 – Exemplo de uma camada totalmente conectada.	54
Figura 10 – Exemplo da transformação de uma série temporal em matriz e repre- sentação da matriz como uma imagem para ser classificada por uma CNN.	58
Figura 11 – Conversão da forma de uma imagem em série temporal	60
Figura 12 – Exemplo da representação de uma série temporal em coordenadas polares como imagem para ser classificada por uma CNN.	60
Figura 13 – Exemplo dos métodos de representação de imagem propostos para uma série temporal com duas classes	63
Figura 14 – Desenho esquemático do processo de Aprendizado Federado	65
Figura 15 – Representação simples da abordagem proposta	68
Figura 16 – Monitoramento da convergência do método de Aprendizado Federado proposto ao longo de 15 rodadas	71
Figura 17 – Exemplo genérico de organização para cada base de dados, contendo o tipo de transformação sofrido pela série temporal, a classe e as instâncias pertencentes a cada classe.	81
Figura 18 – <i>Boxplot</i> das acurácias obtidas para a base de dados <i>DodgerLoopGame</i> .	83
Figura 19 – Intervalo de confiança para a comparação par a par pelo teste de Games-Howell para a base de dados <i>DodgerLoopGame</i>	86
Figura 20 – Comparação entre o treinamento local dos clientes e o aprendizado federado	92

LISTA DE TABELAS

Tabela 1 – Bancos de dados utilizados do repositório UCR (DAU et al., 2018) . . .	77
Tabela 2 – Resultados da classificação baseada em medidas de similaridade aplicadas em FTS	78
Tabela 3 – Exemplo de arquitetura de rede neural aplicada em cada base de dados.	82
Tabela 4 – Resultado do teste de Games-Howell, na base de dados <i>DodgerLoop-Game</i> , para comparação par a par após o teste F	85
Tabela 5 – Resultados da classificação baseada em imagens a partir da CNN . . .	86
Tabela 6 – Resultados do método FL comparado com o método centralizado de classificação baseado em imagens	88
Tabela 7 – Comparação da acurácia do método FL-Server contra utilização de treinamento local dentro de cada cliente e teste com os dados de teste completos.	91
Tabela 8 – Resultados do método multitarefa comparado com a melhor acurácia centralizada, bem como com resultados obtidos em Xing et al. (2022b)	92
Tabela 9 – Comparação entre os melhores resultados das estratégias propostas com métodos de <i>benchmark</i>	95
Tabela 10 – Comparação dos métodos propostos com modelos de referência e métodos de Estado da Arte	99

LISTA DE ABREVIATURAS E SIGLAS

ABNT	Associação Brasileira de Normas Técnicas
APCA	<i>Adaptative Piecewise Constant Approximation</i>
BoW	<i>Bag-of-Words</i>
CNN	<i>Convolutional Neural Networks</i>
DFT	<i>Discrete Fourier Transform</i>
DTW	<i>Dynamic Time Warping</i>
DWT	<i>Discrete Wavelet Transform</i>
ED	<i>Edit Distance</i>
EDR	<i>Edit Distance on Real Sequencies</i>
ERP	<i>Edit Distance on Real Penalty</i>
FL	<i>Federated Learning</i>
FLR	<i>Fuzzy Logic Relationship</i>
FLRG	<i>Fuzzy Logic Relationship Group</i>
FTS	<i>Fuzzy Time Series</i>
LCS	<i>Longest Common Subsequence</i>
MLP	<i>Multilayer Perceptrons</i>
PAA	<i>Piecewise Agregate Approximation</i>
PLR	<i>Piecewise Linear Representation</i>
RNN	<i>Recurrent Neural Networks</i>
SAX	<i>Symbolic Agregate Approximation</i>
TSC	<i>Time Series Classification</i>
TWED	<i>Time Warp Edit Distance</i>
VSM	<i>Vector Space Model</i>

MINDS *Machine Intelligence and Data Science*

UFMG Universidade Federal de Minas Gerais

SVM *Support Vector Machine*

LISTA DE SÍMBOLOS

$\mathcal{L}_2$	Distância Euclideana
$\mathcal{W}$	Distância <i>Warping</i>
μ	Função de pertinência
U	Universo de discurso

SUMÁRIO

1	INTRODUÇÃO	16
1.1	Objetivo Geral	19
1.2	Principais contribuições da tese	20
2	CLASSIFICAÇÃO DE SÉRIES TEMPORAIS	22
2.1	Análise de Séries Temporais	22
2.2	Classificação baseada em medidas de similaridade	25
2.2.1	Métricas de distância	25
2.2.2	<i>Dynamic Time Warping</i>	26
2.2.3	Distância de Edição	29
2.3	Classificação baseada na extração de características	34
2.3.1	Representação de séries temporais	35
2.3.1.1	Métodos baseados na discretização por segmentação	35
2.3.1.2	Transformada de Fourier e Transformada Wavelet	36
2.3.2	Métodos baseados em dicionário de códigos	38
2.3.3	Shapelets	38
3	CLASSIFICAÇÃO BASEADA EM SÉRIES TEMPORAIS FUZZY E MEDIDAS DE SIMILARIDADE	41
3.1	Séries Temporais Fuzzy	42
3.1.1	FTS na classificação de séries temporais	46
3.1.2	Definição dos hiperparâmetros	46
3.2	Estratégias de classificação baseadas em medidas de similaridade propostas	48
4	MÉTODOS BASEADOS EM IMAGENS	51
4.1	Redes Neurais Convolucionais	51
4.1.1	Camada Convolutacional	52
4.1.2	Camada de <i>Pooling</i>	53
4.1.3	Camada Totalmente Conectada	53
4.2	Representação de séries temporais em imagens	54
4.2.1	<i>Recurrence Plots</i>	54
4.2.2	Gramian Angular Field	55
4.2.3	Markov Transition Field	56
4.3	Metodologias propostas para representação das séries temporais em imagens	56
4.3.1	Transformação da série temporal na forma matricial	57
4.3.2	Representação da série temporal em coordenadas polares	58
5	CLASSIFICAÇÃO ATRAVÉS DE APRENDIZADO FEDERADO	64
5.1	<i>Federated Learning</i> (FL)	64

5.2	Técnicas propostas baseadas em Aprendizado Federado	66
5.2.1	Abordagem baseada no particionamento das bases de dados	67
5.2.2	Classificação Multitarefa	70
6	RESULTADOS	74
6.1	Bases de dados UCR	74
6.2	Classificação baseada em medidas de similaridade aplicadas a Séries Tem- porais Fuzzy	75
6.3	Classificação baseada em imagens - Centralizada	80
6.3.1	Comparação entre os algoritmos	82
6.4	Classificação com Aprendizado Federado	87
6.4.1	Dados de treinamento particionados	87
6.4.2	Classificação Multitarefa com aprendizado federado	91
6.5	Comparação dos resultados obtidos	94
7	COMENTÁRIOS E CONCLUSÕES	103
7.1	Limitações e trabalhos futuros	105
7.2	Publicações	107
7.2.1	Publicação em revista	107
7.2.2	Publicações em congressos	107
	REFERÊNCIAS	109

1 INTRODUÇÃO

O aumento de equipamentos capazes de transmitir e armazenar dados é cada vez mais perceptível e, muitas vezes, necessário em diversos campos de análise, sendo utilizados, para isso, muitos sensores, os quais são empregados para gerar uma grande massa de dados (BAILLY, 2018; AZAR et al., 2020). Para lidar com esse cenário, há, também, o advento de novos computadores cada vez mais potentes, possibilitando a demanda, geração e tratamento de números ainda maiores de dados. Uma vez obtidos, o principal desafio, portanto, é a análise apropriada, a qual necessita se dar em tempo hábil, dependendo da aplicação, além de ser capaz de extrair o máximo de informações possíveis.

Um exemplo de classe de dados comum e caracterizada por seu grande volume de informação (FU, 2011) são as séries temporais. As séries temporais podem ser obtidas de diversas áreas, e possuem amostras, obrigatoriamente, obtidas de maneira ordenada, seja essa ordenação referente ao tempo ou não (BAGNALL et al., 2017). Uma tarefa importante, referente às séries temporais, é a mineração dos dados dessas séries. De maneira geral, a mineração de dados consiste em um processo analítico de extração de conhecimento, e, no caso das séries temporais, esse processo geralmente pode incluir: reconhecimento de padrões, clusterização, classificação, descoberta de regras, resumo e visualização (FU, 2011; KLEIST, 2015). A clusterização, classificação e previsão de dados das séries temporais são algumas das principais abordagens de análise, presentes em diversas pesquisas, referentes à mineração de dados.

No que se refere à classificação de séries temporais, seu intuito é de agrupar itens com características semelhantes (MAHARAJ; D'URSO; CAIADO, 2019), atribuindo-lhes uma entre duas ou mais classes predefinidas (KEOGH; KASETTY, 2003), sendo reconhecida, como uma abordagem supervisionada. Durante as duas últimas décadas, a classificação de séries temporais é considerada um dos problemas mais desafiadores da mineração de dados (FAWAZ et al., 2019) e com o intuito de realizar essa tarefa, centenas de métodos foram propostos (BAGNALL et al., 2017). A principal característica entre eles é o desenvolvimento de um classificador, treinado a partir de dados já rotulados, a fim de ser aplicado nos dados com classe desconhecida, para o processo de classificação propriamente dito.

Entre as pesquisas e métodos propostos para realizar a classificação de séries temporais, pode-se afirmar que os algoritmos que aplicam algum tipo de aprendizado de máquina ou inteligência artificial são de grande interesse. Uma das abordagens mais tradicionais e populares, referentes a essa tarefa, aplica o classificador *Nearest Neighbor* (NN) acoplado a alguma medida de distância, tais como distância Euclideana ou *Dynamic Time Warping*, o qual se mostrou, por longo tempo, um *benchmark* importante (FAWAZ

et al., 2019). Além dos classificadores baseados em medida de distância, existem também, estratégias de classificação baseadas na extração de características. Essas estratégias mapeiam a série temporal, transformando-a em um vetor de características, para então, realizar o processo de classificação.

Outras estratégias, denominadas de Classificadores *Ensemble*, os quais utilizam métodos combinados, como classificadores NN e diferentes medidas de distância, também produzem bons resultados, superando a associação NN-DTW. Classificadores baseados em *Ensembles* são também empregados em conjunto com métodos de extração de característica, os quais aplicam classificadores diversos, como *Support Vector Machines*, *Nearest Neighbors* em um ou mais espaços de características, tendo a representação da série temporal em novo vetor de características como uma propriedade em comum. Um dos principais exemplos dessa metodologia, intitulada Coletivo de Conjuntos de Transformação (*Collective Of Transformation-based Ensembles* — COTE), foi desenvolvida por Bagnall et al. (2015). Ao invés de apenas aplicar classificadores em um mesmo vetor de características, essa estratégia utiliza conjuntos de classificadores provenientes de diferentes representações de séries temporais. Em outro estudo, Lines, Taylor e Bagnall (2016) e Lines, Taylor e Bagnall (2018) propuseram uma extensão da estratégia COTE, nomeada de HIVE-COTE, a qual emprega um sistema de voto hierárquico. Essa metodologia apontou resultados significativamente melhores que seu antecessor, tornando-se um forte *benchmark* dentre os algoritmos de classificação de séries temporais (FAWAZ et al., 2019).

Transformar as séries temporais em um novo espaço no qual características determinantes são mais facilmente detectadas pode apresentar melhores resultados na tarefa de classificação (BAGNALL et al., 2012). Exemplos recentes de representação de séries temporais em um espaço de características (ZOU et al., 2019), proposta no final da última década, são trabalhos baseados em ciência de redes. Diferentes técnicas foram propostas seguindo essa abordagem (SILVA et al., 2021), com o intuito de melhorar a tarefa de classificação de séries temporais (LI; JIA; WAN, 2022). Muitos desses trabalhos propostos (LACASA; NICOSIA; LATORA, 2015; SUPRIYA et al., 2016; LI; JIA; WAN, 2022) seguiram o conceito conhecido como *Visibility Graphs*, propostos por Lacasa et al. (2008)

Mesmo com o grande número de algoritmos desenvolvidos, baseados em inteligência artificial e aprendizado de máquina, ainda existe campo para estudo de algumas abordagens, a princípio, bastante promissoras em determinadas áreas de aplicação, porém pouco exploradas na tarefa de classificação de séries temporais. Como exemplo dessas abordagens, pode-se citar as redes neurais profundas, as quais vêm apresentando resultados promissores em diversos campos relacionados à tarefa de classificação. Classificação de imagens, processamento natural de linguagem (Natural Language Processing - NLP) e reconhecimento de fala são algumas das áreas onde algoritmos de aprendizado profundo

apresentam bons resultados. No que tange ao reconhecimento de imagens, as Redes Neurais Convolucionais (*Convolutional Neural Networks* - CNN) são as principais abordagens aplicadas devido ao seu excelente desempenho em Visão Computacional ([KRIZHEVSKY; SUTSKEVER; HINTON, 2012](#)), capaz de apresentar resultados similares aos de um humano em tarefas de reconhecimento de imagem.

Apesar do sucesso obtido pelas abordagens anteriores, bem como da oportunidade de aplicação de técnicas de aprendizado profundo, muitas delas são consideradas como caixa-preta e limitadas por sua baixa interpretabilidade. Portanto, métodos de classificação de séries temporais utilizando Séries Temporais Fuzzy (*Fuzzy Time Series* - FTS) ([ZADEH, 1965](#)) são também reconhecidos como uma outra possível abordagem nesse contexto. As FTS são uma abordagem bastante conhecida e utilizada na Previsão de Séries Temporais, apresentando bons resultados nessa análise. Esses métodos são utilizados devido à sua facilidade de implementação, interpretabilidade, escalabilidade, robustez em modelar sistemas complexos, e alta acurácia, como evidenciado por [Lucas et al. \(2022\)](#).

Ao se observar os resultados promissores das Séries Temporais Fuzzy na análise de séries temporais, bem como as Redes Neurais Convolucionais no campo da visão computacional, percebe-se uma oportunidade de pesquisa com a junção das duas técnicas. Essa abordagem foi tratada anteriormente no trabalho de [Sadai et al. \(2019\)](#) ao propor um método baseado em FTS-CNN para previsão de séries temporais. FTS-CNN é um método híbrido para previsão de séries temporais multivariáveis que integra FTS e CNN para prever séries temporais de carga elétrica.

Apesar do sucesso dos métodos discutidos, alguns desafios estão atrelados a eles, como alto custo computacional e preocupações relacionadas à privacidade de dados, especialmente com o aumento de volumes de informações. Para lidar com esses problemas, o Aprendizado Federado (*Federated Learning* - FL), introduzido por [McMahan et al. \(2017\)](#), oferece uma solução ao permitir que modelos de aprendizado de máquina sejam treinados através de dados distribuídos sem a necessidade de sua centralização ([MOTHUKURI et al., 2021](#)). O treinamento distribuído preserva a privacidade dos dados ([TRUEX et al., 2019](#)), enquanto facilita o compartilhamento do aprendizado entre as fontes de dados, evitando, assim, os altos custos e riscos de dados centralizados ([ALEDHARI et al., 2020](#)).

Outro ponto interessante do aprendizado federado é a possibilidade de utilização de classificação multitarefa ([XING et al., 2022b](#)). Essa abordagem pode considerar diferentes situações para desenvolver um único modelo capaz de realizar a classificação de todas elas. O servidor agrega os parâmetros locais de cada cliente e depois utiliza um modelo global para realizar a classificação. Assim, tanto as vantagens do paradigma multitarefa, quanto as vantagens do aprendizado federado são disponibilizadas.

Devido a essas vantagens, o Aprendizado Federado é particularmente útil na classificação de séries temporais, principalmente onde privacidade e gerenciamento de

dados é de grande importância. Por agregar os parâmetros de um modelo, como por exemplo de uma rede neural artificial, identificou-se uma oportunidade de se utilizar essa técnica a fim de demonstrar a adaptabilidade da metodologia baseada em imagens, proposta em ambientes distribuídos, onde a privacidade de dados é importante. Portanto, em uma terceira vertente, esse estudo realiza a combinação de métodos de representação de imagens, CNN e Aprendizado Federado para a classificação de séries temporais.

Portanto, as principais abordagens propostas neste trabalho podem ser elencadas a seguir:

- discutir sobre os diversos algoritmos utilizados para classificação de séries temporais, baseados em medidas de similaridade e extração de características;
- discutir sobre as séries temporais fuzzy e propor métodos de classificação baseados em séries temporais fuzzy e medidas de similaridade;
- Propor métodos de representação da série temporal através de imagens, para sua posterior classificação por uma CNN;
- Utilizar a abordagem de Aprendizado Federado para a classificação de séries temporais em ambientes distribuídos, utilizando as metodologias propostas para a representação das séries temporais através imagens

1.1 Objetivo Geral

O objetivo geral do presente trabalho é propor novas abordagens de classificação de séries temporais, empregando métodos baseados no reconhecimento de imagens e medidas de similaridade, ambos aplicados a séries temporais fuzzy. Para alcançar o objetivo geral proposto, os seguintes objetivos específicos podem ser elencados:

- Revisar a bibliografia dos principais métodos de classificação de séries temporais
- Definir os métodos de fuzzificação da série temporal;
- Aplicar métodos de medida de similaridade na série temporal fuzzificada;
- Definir os métodos de transformação da série temporal em imagem;
- Definir os métodos que utilizam aprendizado federado;
- Apresentar e comparar cada método proposto entre si e com resultados de referência e Estado da Arte

1.2 Principais contribuições da tese

Pode-se, portanto, elencar como principais contribuições do trabalho a proposta de três novas metodologias de classificação de séries temporais. Um método que utiliza as séries temporais fuzzy para a obtenção de uma nova série temporal fuzzificada, a qual pode ser aplicada a métodos de medida de similaridade para a posterior classificação. Com essa metodologia, espera-se obter uma nova série temporal (discretizada através dos grupos fuzzy ou linguística, representada pelos grupos fuzzy), capaz de filtrar eventuais ruídos presentes nos dados originais. Com isso, é possível, em certos casos, obter uma classificação mais eficiente.

A segunda metodologia proposta é referente à representação das séries temporais através de imagens para sua posterior classificação. Nessa estratégia, foram propostas três metodologias de representação da série temporal em imagens: 1) Utilizando a série temporal numérica (NMotif); 2) utilizando a série temporal fuzzificada (FTSMotif); 3) Utilizando uma representação da série em coordenadas polares (Polar). Espera-se, com essa abordagem, usufruir da capacidade das CNN em encontrar padrões espaciais em imagens. A proposta se baseia na premissa de que, ao representar uma série temporal como uma imagem, os padrões e correlações temporais da série seriam transformados em padrões e correlações espaciais. A CNN, por fim, pode reconhecer esses padrões. Assim, estudar formas de representação de séries temporais para serem usadas como imagens para posterior classificação por meio de uma CNN representa uma oportunidade de pesquisa.

Um ponto interessante a ser notado sobre a metodologia baseada em imagens e CNN proposta é sua capacidade de ser aplicada, tanto em modo centralizado, quanto em modo distribuído, demonstrando a adaptabilidade da técnica. O modo distribuído é referente ao terceiro e quarto métodos propostos, e principais contribuições da tese. Eles possuem dois principais avanços:

- I) Primeira introdução, até onde o autor tem conhecimento, de um modelo distribuído de classificação de séries temporais (TSC) que combina representações em imagem das séries com CNNs dentro de um framework de Aprendizado Federado (FL).
- II) Proposta de duas abordagens baseadas em FL, uma que particiona os dados de treinamento em clientes, para simulação de ambiente federado e outra que utiliza uma abordagem multitarefa. Essas abordagens distribuem a tarefa de classificação entre múltiplos clientes, reduzindo a necessidade de coleta centralizada de dados e aumentando a privacidade e a segurança. Esse método oferece vantagens distintas em relação aos métodos baseados em similaridade, que ainda não foram amplamente explorados em cenários federados de TSC. Um desafio importante desses métodos é sua dependência de comparações diretas entre dados, o que é problemático no contexto

do FL devido às rigorosas restrições de privacidade. A natureza descentralizada do FL impõe dificuldades para calcular e agregar medidas de similaridade entre os clientes sem compartilhar os dados brutos, exigindo novas abordagens ou inovações para tornar esses métodos viáveis nesse contexto.

O restante do documento está organizado da seguinte maneira: Capítulo 2 apresenta uma revisão bibliográfica do assunto Classificação de Séries Temporais, bem como os principais métodos aplicados para essa tarefa. O Capítulo 3 discute sobre as Séries Temporais Fuzzy (*Fuzzy Time Series* - FTS), bem como apresenta os métodos de partição do universo de discurso abordados no processo de fuzzificação. O capítulo ainda discorre sobre os métodos de classificação utilizados, baseados nas séries temporais fuzzy e medidas de similaridade. O Capítulo 4 apresenta uma introdução sobre as Redes Neurais Convolucionais, além de introduzir as abordagens propostas para a representação da série temporal através de imagens para sua posterior classificação. O capítulo 5 apresenta a abordagem de Aprendizado Federado, bem como introduz os métodos propostos de classificação de séries temporais que utilizam essa técnica. Posteriormente, o Capítulo 6 apresenta os resultados obtidos com a aplicação dos métodos propostos. Esse capítulo apresenta uma comparação entre as metodologias propostas, a fim de entender suas principais diferenças no que se refere à acurácia de classificação. Posteriormente, o capítulo apresenta uma comparação dos resultados obtidos para as metodologias propostas com técnicas consideradas *benchmark*, tratados como resultados de *Baseline*. Em seguida é apresentada uma comparação entre os métodos propostos e as principais técnicas de Estado da Arte. Por fim, o Capítulo 7 descreve as principais conclusões sobre as metodologias propostas, bem como a tese como um todo, além de evidenciar as principais limitações dos métodos propostos e indicar possibilidades de pesquisas futuras.

2 CLASSIFICAÇÃO DE SÉRIES TEMPORAIS

Durante as duas últimas décadas, a classificação de séries temporais tem sido considerada como um dos problemas mais desafiadores na mineração de dados (YANG; WU, 2006; ESLING; AGON, 2012). Essa tarefa atrai bastante atenção em pesquisas atuais devido à sua vasta possibilidade de aplicação em diferentes campos, havendo, portanto, diversos algoritmos desenvolvidos para esse fim (ZHENG et al., 2014). Este capítulo apresenta os contextos e definições que circundam esse tema, bem como estudos relevantes que compõem o estado da arte sobre Classificação de Séries Temporais.

2.1 Análise de Séries Temporais

Antes de abordar o tema de Classificação de Séries temporais, propriamente dito, é necessário apresentar definições importantes para embasamento sobre o assunto. Segundo Fu (2011), as séries temporais podem ser definidas como uma coleção de observações realizadas ao longo do tempo, cuja natureza inclui grande massa de dados, alta dimensão e atualização contínua. Sua definição pode ser realizada como demonstrado adiante (BAILLY, 2018):

Definição 2.1.1 (Séries Temporais) *Uma série temporal $\mathbf{x}_i$ de tamanho n é uma sequência ordenada de dados, medidos em intervalos de tempo regulares, tal que*

$$\mathbf{x}_i = x_{i,1}, x_{i,2}, \dots, x_{i,n} \quad (2.1)$$

onde $x_{i,j}$ representa o j -ésimo elemento da série temporal $\mathbf{x}_i$ (BAILLY, 2018).

Uma série temporal pode ser univariada, contendo apenas observações de um único dado, ou multivariada, contendo observações de múltiplos dados obtidos de maneira ordenada. Os dados representados pelas séries temporais podem ser provenientes de diversos campos, sendo alguns dos exemplos encontrados no campo da saúde (eletrocardiogramas, eletro-encefalogramas, condições de saúde de um paciente); estudo de clima (análise de chuvas e eventos da natureza); economia; dentre outros (NIELSEN, 2019; MAHARAJ; D'URSO; CAIADO, 2019; CHATFIELD, 2016).

No que se refere ao estudo de séries temporais, uma tarefa importante e bastante pesquisada é sua análise. Segundo Nielsen (2019), a análise de uma série temporal é o esforço de extrair informações resumidas e estatisticamente significativas de pontos organizados em ordem cronológica. Algumas das principais tarefas da mineração de dados de séries temporais são: reconhecimento de padrões e regressão, reconhecimento de regras,

resumo e classificação (FU, 2011). Uma vez que o escopo principal do presente trabalho é a classificação de séries temporais, esse tema será explorado no decorrer da presente seção.

A tarefa de classificação de séries temporais é uma abordagem supervisionada para agrupar itens de interesse, de tal maneira que cada série estudada é considerada como sendo pertencente a uma determinada classe pré definida. Seja um conjunto de séries temporais, $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\}$, onde $\mathcal{X}$ pode ser considerado como um conjunto de dados (*dataset*) a associação de cada série temporal do conjunto com seu rótulo correspondente, pode ser representada como $(\mathbf{x}_i, y_i)$. Um conjunto de séries temporais rotuladas, por sua vez, pode ser representado pelo conjunto $\mathcal{D}$, definido a seguir (BAILLY, 2018):

Definição 2.1.2 (Conjunto de séries temporais rotuladas) *Um conjunto de séries temporais rotuladas, $\mathcal{D}$, corresponde a um grupo de séries temporais associadas e um grupo de rótulos, sendo um rótulo para cada série temporal. O rótulo y_i representa a classe da i -ésima série temporal $\mathbf{x}_i$.*

$$\mathcal{D} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_N, y_N)\} \quad (2.2)$$

O processo de classificação ocorre com o intuito de agrupar os dados em classes, categorias ou grupos, de acordo com suas características. Para o caso de uma série temporal não rotulada, o objetivo é relacioná-la a uma das classes pré-existentes, considerando-se suas características principais. Esse processo se divide em dois passos, treinamento e classificação, descritos a seguir.

i Treinamento

Durante o processo de treinamento, o algoritmo de classificação desenvolve um modelo, baseado nos dados já rotulados, introduzidos no modelo como conjunto de treinamento. O classificador desenvolvido no processo de treinamento deve ser capaz de generalizar bem para novos casos e situações.

ii Classificação

Após realizado o treinamento do classificador ele pode ser utilizado para classificar novas séries temporais, cujos rótulos não são conhecidos. Um classificador pode ser definido como uma função f que mapeia a série temporal do seu espaço original para o espaço de possíveis rótulos de classes, esse processo pode ser explicitado pela equação a seguir (BAILLY, 2018):

$$f : \mathcal{X} \longrightarrow \{1, \dots, m\}^N \text{ ou } f : \mathbf{x}_i \longmapsto \{1, \dots, m\} \quad (2.3)$$

Normalmente, os dados utilizados para avaliar a performance do classificador são divididos em conjunto de treino e conjunto de teste ($\mathcal{D}_{treino}$ e $\mathcal{D}_{teste}$), de tal maneira que o classificador é ajustado pelos dados de treinamento, aprendendo os parâmetros necessários para a classificação. Pode-se dizer, portanto, que o propósito de um classificador é encontrar a função f que maximiza a acurácia dado $\mathcal{D}_{treino}$. Após seu treinamento, o classificador é avaliado, sendo submetido à classificação dos dados de teste. Para o caso da equação (2.3), $f(\mathbf{x}_i)$ é a classe prevista para a série temporal $\mathbf{x}_i$, e caso $f(\mathbf{x}_i) = y_i$, o classificador realizou a previsão correta.

Uma característica desejada ao se desenvolver um classificador é que ele seja capaz de generalizar, de maneira satisfatória, quando aplicado em dados desconhecidos e ainda não classificados, mesmo sendo esses dados provenientes de diferentes áreas. Apesar de essa característica ser um desafio pesquisado por diversos autores, não é possível obter um método capaz de otimizar a função de custo e que seja igualmente aplicável a todos os tipos de problemas (WOLPERT; MACREADY, 1997). Devido à dificuldade de encontrar um classificador universal, que possa ser aplicado a diversos tipos de problemas, outras características devem ser levadas em conta ao se comparar a performance de métodos durante uma tarefa de classificação, sendo possível citar algumas delas:

- velocidade da tarefa de classificação;
- robustez a ruídos e *outliers*;
- habilidade de lidar com distorções temporais (como deslocamento no tempo e dilatação);
- custo computacional;
- número de parâmetros.

Com o intuito de se obter melhorias na tarefa de classificação de séries temporais, muitos métodos vêm sendo desenvolvidos (ZHENG et al., 2014). Apesar dos vários algoritmos existentes, diversos autores definem que, de uma maneira geral, os algoritmos de classificação de séries temporais podem ser divididos em três grandes categorias: classificação baseada no cálculo de distância, classificação baseada na extração de características e a classificação baseada em modelos, ou combinação de modelos (*ensembles*) (XING; PEI; KEOGH, 2010; JIANG, 2020). Dessa forma, a tarefa de classificar a série temporal pode ser realizada posteriormente à aplicação de uma medida de similaridade ou método de representação da série, sendo essas ferramentas, muitas vezes responsáveis por tratar os dados, para, então, serem classificados (KEOGH; KASETTY, 2003; FU, 2011).

Nas próximas seções, serão apresentados alguns dos principais métodos de classificação de séries temporais, levando-se em consideração duas categorias referentes a

essa tarefa, a classificação baseada em distância e a classificação baseada na extração de características.

2.2 Classificação baseada em medidas de similaridade

Os métodos baseados no cálculo de distância possuem, de maneira geral, o objetivo de estimar distâncias ponto a ponto entre as séries temporais analisadas (BAILLY, 2018), utilizando-as como métrica para similaridade entre as séries ou como classificadores propriamente ditos. Muitos dos métodos baseados na extração de características, os quais serão abordados adiante, utilizam um método de medida de similaridade após realizarem a representação da série temporal. O processo de medir a similaridade entre duas séries temporais pode ser entendido como a busca pela série que mais se aproxima da série temporal analisada. Essa tarefa pode ser dividida em duas abordagens principais: (1) encontrar similaridade entre duas séries temporais completas (ou correspondência da sequência inteira); (2) buscar similaridade em partes da série temporal (ou correspondência de subsequências) (FU, 2011). O objetivo da presente seção é abordar alguns dos principais métodos utilizados para a medida de similaridade entre séries temporais aplicados na tarefa de classificação, bem como citar diferentes estudos referentes ao assunto.

2.2.1 Métricas de distância

Um caso particular de medidas de similaridade são as chamadas métricas de distância. Para ser considerada uma métrica de distância d entre duas séries temporais x_i e x_l , a medida deve seguir as seguintes regras (BAILLY, 2018):

1. Não negatividade: $d(x_i, x_l) \geq 0$
2. Identidade: $d(x_i, x_l) = 0 \iff x_i = x_l$
3. Simetria: $d(x_i, x_l) = d(x_l, x_i)$
4. Desigualdade triangular: $\forall x_z : d(x_i, x_z) + d(x_z, x_l) \geq d(x_i, x_l)$

A métrica de distância d é uma função representada pela equação abaixo:

$$d : \mathcal{X} \times \mathcal{X} \rightarrow [0, +\infty] \quad (2.4)$$

onde $\mathcal{X}$ é um conjunto de séries temporais $\{x_1, x_2, \dots, x_N\}$.

Um exemplo muito comum de métricas de distância são as distâncias $\mathcal{L}_p$, também conhecidas como distância Minkowski, cujo valor de p é real e estritamente positivo. Elas podem ser representadas pela seguinte equação:

$$d_p(\mathbf{z}, \mathbf{v}) = \|\mathbf{z} - \mathbf{v}\|_p = \left(\sum_{i=1}^n |z_i - v_i|^p \right)^{1/p} \quad (2.5)$$

As distâncias Minkowski, equação (2.5), são consideradas como uma generalização das distâncias Manhattan, Euclidiana e Chebyshev, com $p = 1$, $p = 2$ e $p = \infty$, respectivamente.

Segundo Fu (2011), a abordagem mais popular para se realizar o cálculo da similaridade entre duas séries temporais é através da Distância Euclidiana aplicada em algum método de representação, como Transformada Discreta de Fourier (*Discrete Fourier Transform* - DFT) (DFT) ou Transformada Discreta Wavelet (*Discrete Wavelet Transform* - DWT), por exemplo. Apesar de bastante populares e de fácil implementação computacional, as distâncias $\mathcal{L}_p$ possuem a desvantagem de não serem apropriadas quando há distorções temporais entre as séries analisadas, ou mesmo quando as similaridades entre elas estão deslocadas. Para lidar com esse inconveniente, uma medida de similaridade também bastante aplicada, denominada *Time Warping*, pode ser utilizada como base para calcular a distância entre séries temporais através do método *Dynamic Time Warping* (DTW), descrito com mais detalhes a seguir.

2.2.2 *Dynamic Time Warping*

Introduzido por Berndt e Clifford (1994) e baseada na técnica de programação dinâmica apresentada por Sakoe e Chiba (1970), esse método possui propriedades elásticas, apropriado para lidar com distorções e deslocamentos no eixo do tempo. Apesar de ser um método de alinhamento de sequência capaz de comparar séries temporais de tamanho diferente, o algoritmo DTW é sensível a ruídos presentes nas séries temporais, além de serem piores que as distâncias $\mathcal{L}_p$ no que se refere a complexidade computacional: sendo o algoritmo DTW, $\mathcal{O}(n^2)$, diferentemente das distâncias $\mathcal{L}_p$, $\mathcal{O}(n)$. Ele também não representa uma métrica de distância, uma vez que não satisfaz a regra de desigualdade triangular. O algoritmo DTW pode ser sintetizado como a seguir (BAILLY, 2018):

Algoritmo DTW

Dadas duas séries temporais $\mathbf{x}_i$ e $\mathbf{x}_l$ (de tamanho n), forma-se uma rede de pontos em um plano, de tal maneira que cada ponto da rede (j, k) corresponde ao alinhamento entre $x_{i,j}$ e $x_{l,k}$. Para encontrar o custo total, deve-se, primeiramente, calcular o custo ponto a ponto. O caminho $\mathcal{W}$, denominado *warping path*, corresponde à sequência $w = w_1, w_2, \dots, w_L$, onde w_m se refere ao par (j_m, k_m) - indicando uma correspondência entre x_{i,j_m} e x_{l,k_m} . Um exemplo da aplicação do algoritmo DTW em duas séries temporais arbitrárias, bem como a comparação do caminho $\mathcal{W}$ e Distância Euclidiana, podem ser

encontrados na Figura 1. O espaço de possíveis *warping paths* - $\mathcal{W}$ - deve satisfazer as seguintes condições:

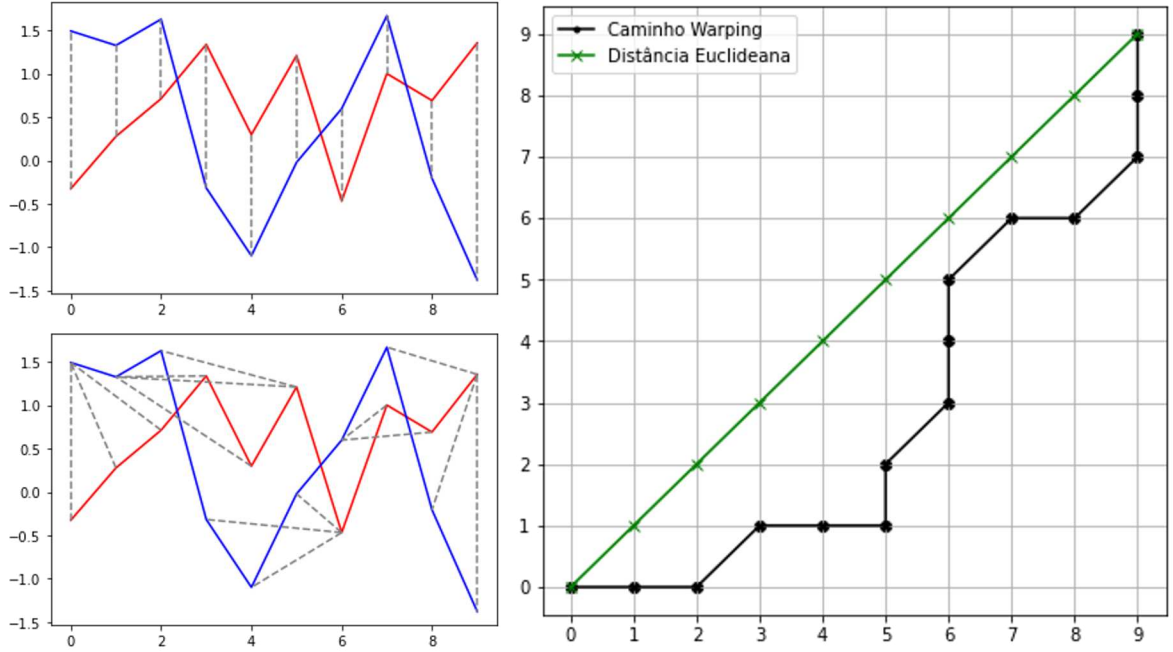


Figura 1 – Exemplo de aplicação da Distância Euclidiana (Superior esquerdo) e distância DTW (Inferior esquerdo) e os respectivos caminhos traçados pela Distância Euclidiana e *Warping Path* entre duas séries temporais.

1. Monotonicidade:

Os pontos do possível caminho $\mathcal{W}$ devem possuir ordem cronológica:

$$j_{m-1} \leq j_m \text{ e } k_{m-1} \leq k_m;$$

2. Continuidade:

Os passos na rede de pontos devem ser consecutivos:

$$j_m - j_{m-1} \leq 1 \text{ e } k_m - k_{m-1} \leq 1;$$

3. Condições de contorno:

O caminho $\mathcal{W}$ deve iniciar no início da série temporal e finalizar no último ponto da série:

$$w_1 = (1, 1) \text{ e } w_L = (n, n)$$

O custo final c de um caminho $\mathbf{w}$ entre $\mathbf{x}_i$ e $\mathbf{x}_l$ é definido pela seguinte equação:

$$c_{\mathbf{w}}(\mathbf{x}_i, \mathbf{x}_l) = \sum_{m=1}^L d(x_{i,j_m}, x_{l,k_m}) \quad (2.6)$$

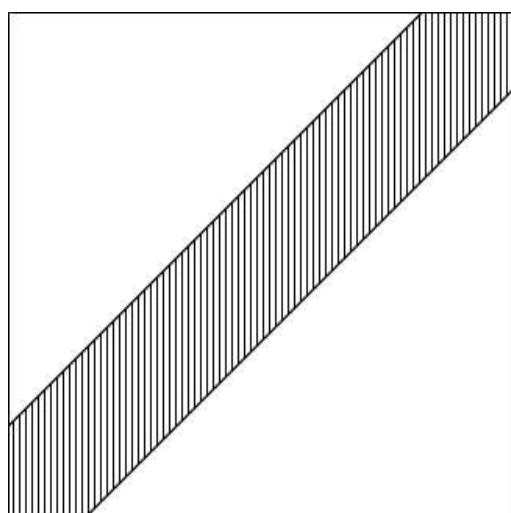
O caminho ótimo entre $\mathbf{x}_i$ e $\mathbf{x}_l$ é o caminho $\mathbf{w}$ de menor custo entre todos os possíveis caminhos em $\mathcal{W}$. A distância DTW corresponde ao custo do caminho $\mathbf{w}$ ótimo, representado pela equação a seguir:

$$DTW(\mathbf{x}_i, \mathbf{x}_l) = \min_{\mathbf{w} \in \mathcal{W}} c_{\mathbf{w}}(\mathbf{x}_i, \mathbf{x}_l) \quad (2.7)$$

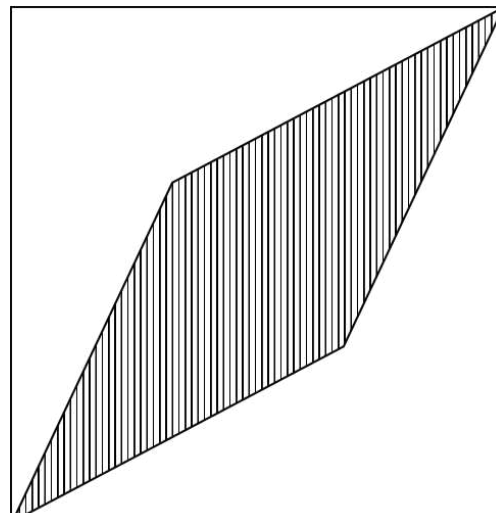
A distância DTW se mostrou uma ferramenta bastante importante na tarefa de classificação de séries temporais. Segundo Batista, Wang e Keogh (2011), a medida DTW, aliada com o algoritmo *Nearest Neighbors* é considerado como um *benchmarking* para outros modelos, difícil de ser superado e tendo sua notável eficiência reconhecida por pelo menos uma década (WANG; YAN; OATES, 2017). Mesmo com sua notável importância e bons resultados apresentados, esse método possui alguns pontos negativos, como por exemplo, o fato de serem sensíveis a ruídos presentes nas séries temporais, além de sua alta complexidade computacional. Esses inconvenientes foram responsáveis pelo desenvolvimento de diversos estudos e métodos, baseados no DTW, visando melhoria do algoritmo original.

Com o intuito de reduzir a complexidade computacional e, conseqüentemente, o tempo de execução do algoritmo DTW, algumas técnicas foram propostas, sendo algumas delas baseadas na restrição do espaço de possíveis caminhos $\mathcal{W}$. Dois dos estudos bastante populares, referentes a esse processo, foram propostos por Sakoe e Chiba (1978), Itakura (1975), conhecidas como bandas de Sakoe-Chiba e paralelogramo de Itakura. A banda de Sakoe-Chiba basicamente determina um valor de janela (r), positivo, ao longo da diagonal da grade de pontos. Essa janela possui dimensões fixas na horizontal e vertical, como demonstrado na Figura 2a. Essa janela estreita os possíveis caminhos $\mathcal{W}$, limitando os pontos à condição: $|j_m - k_m| \leq r$. O paralelogramo de Itakura, por sua vez, limita o espaço de caminhos possíveis, porém permitindo menores deslocamentos nas extremidades da série temporal, e maiores distorções ao longo do centro da série. A Figura 2b, apresenta um esboço da restrição proposta por Itakura.

Outra frente também bastante abordada no desenvolvimento de melhorias para o algoritmo DTW refere-se à sua sensibilidade a ruídos e *outliers*. Em 2001, por exemplo, Keogh e Pazzani (2001) propuseram uma extensão do algoritmo DTW original, denominada *Dynamic Time Warping Derivativo* (*Derivative Dynamic Time Warping - DDTW*). Essa técnica utiliza a distância entre as derivadas das séries temporais, com o intuito de reduzir as *singularidades* produzidas pelo algoritmo DTW para explicar diferenças no eixo y das séries temporais analisadas. DDTW é capaz de reduzir a sensibilidade a ruído e *outliers*, diferentemente do algoritmo original. Ainda com a característica de reduzir a sensibilidade a ruídos e *outliers*, outro exemplo que pode ser citado é o *Dynamic Time Warping* ponderado (WDTW). Introduzida por Jeong, Jeong e Omitaomu (2011), Essa técnica pode ser considerada como um DTW penalizado, cujo peso aplicado é proporcional



(a) Banda de Sakoe-Chiba



(b) Paralelogramo de Itakura

Figura 2 – Exemplos de restrições para o algoritmo DTW

Adaptado de [Bailly \(2018\)](#)

à diferença temporal entre os pontos. Uma vantagem importante desse algoritmo é que ele pode ser associado com outras variantes do DTW, como DDTW, ou pode-se aplicar restrições ou técnicas de aumento de velocidade.

A medida de similaridade *Dynamic Time Warping* é considerada um método muito importante na tarefa de classificação de séries temporais, sendo, ainda atualmente, apontado como um *benchmarking* para algoritmos. Existem, porém, outros métodos de busca por similaridade/dissimilaridade, os quais serão brevemente abordados adiante. Vale ressaltar que a importância da ferramenta DTW é tão notável, que alguns dos algoritmos apresentados a seguir se baseiam nessa técnica, com o intuito de obter melhores resultados.

2.2.3 Distância de Edição

Segundo [Jurafsky e Martin \(2020\)](#), a distância de edição (*Edit Distance* - ED) é um método que quantifica a similaridade em relação a duas sequências de caracteres, baseando-se no número de edições a ser realizada em determinada sequência para que ela se iguale à outra (inserção, deleção e substituição). Um dos primeiros algoritmos capazes de efetuar essas operações foi proposto por [Levenshtein et al. \(1966\)](#), por essa razão, ela é comumente denominada Distância Levenshtein ([SCHIMKE; VIELHAUER; DITTMANN, 2004](#)). A sequência de caracteres pode ser representada por valores discretos (como por exemplo letras) ou símbolos do alfabeto ([BAILLY, 2018](#)). Um conceito importante ao se utilizar o algoritmo de distância de edição é a mínima distância de edição, definida como o número mínimo de operações de edição necessárias para transformar uma sequência em

outra.

Um ponto considerável do algoritmo distância de edição para o presente trabalho, é que ele também pode ser utilizado para comparar séries temporais, realizando a transformação dos valores da série em valores discretos, ou sequência de caracteres. Ao ser aplicado em séries temporais, esse algoritmo possui a característica de levar em consideração *outliers*, diferentemente do algoritmo DTW. No que tange à classificação de séries temporais, diversos algoritmos baseados nessa ferramenta foram propostos, alguns dos quais serão abordados na presente seção.

O termo **mínima distância de edição** surgiu em 1974, no trabalho de [Wagner e Fischer \(1974\)](#), porém, o algoritmo foi descoberto independentemente por diversos outros autores ([JURAFSKY; MARTIN, 2020](#)). Muitos algoritmos desenvolvidos para encontrar distância entre sequência de caracteres baseiam-se em alguma versão da mínima distância de edição. Um exemplo abordado e aplicado no presente trabalho, é a distância baseada na Subsequência Comum Mais Longa (*Longest Common Subsequence* - LCS). Esse algoritmo, aplicado tanto em séries temporais univariadas ([DAS; GUNOPULOS; MANNILA, 1997](#)) quanto multivariadas ([VLACHOS; KOLLIOS; GUNOPULOS, 2002](#)), possui como base encontrar a subsequência mais longa, comum a duas sequências, permitindo não alinhamento entre elementos ou retirada de alguns deles. O cálculo da distância LCS pode ser descrito como a seguir:

Definição 2.2.1 (Distância LCS) *Dadas duas séries temporais $\mathbf{x}_i$ e $\mathbf{x}_l$, a distância LCS é definida como*

$$d_\delta(\mathbf{x}_i, \mathbf{x}_l) = \begin{cases} 0 & \text{se } \mathbf{x}_i \text{ ou } \mathbf{x}_l \\ 1 + d_\delta(h(\mathbf{x}_i), h(\mathbf{x}_l)) & \text{se } |x_{i,-1} - x_{l,-1}| < \delta \\ \max(d_\delta(h(\mathbf{x}_i), \mathbf{x}_l), d_\delta(\mathbf{x}_i, h(\mathbf{x}_l))) & \text{caso contrário} \end{cases} \quad (2.8)$$

onde δ é o limiar de tolerância e $h(\mathbf{x}_i)$ corresponde à série temporal $\mathbf{x}_i$ sem seu último ponto $x_{i,-1}$.

Para se utilizar o algoritmo LCS em séries temporais, foi necessário realizar uma relaxação no algoritmo LCS original, de tal forma que a igualdade entre dois caracteres foi substituída por uma desigualdade. Ou seja, apesar de ser improvável que dois valores contínuos dentre duas séries temporais sejam equivalentes, eles podem ser próximos um ao outro dentro de uma tolerância ($|x_{i,j} - x_{l,k}| < \delta$).

Com o intuito de levar em consideração a natureza contínua das variáveis, [Chen, Özsu e Oria \(2005\)](#) propôs o método conhecido como Distância de Edição em Sequências Reais (*Edit Distance on Real Sequencies* - EDR), cujo cálculo pode ser realizado como na definição abaixo:

Definição 2.2.2 (Distância de Edição em Sequências Reais) Dadas duas séries temporais $\mathbf{x}_i$ e $\mathbf{x}_l$, a EDR entre dois pontos é definida como:

$$d_{EDR*}(x_{i,j}, x_{l,k}) = \begin{cases} 0 & \text{se } |x_{i,j} - x_{l,k}| \leq \delta \\ 1 & \text{se } x_{i,j} \text{ ou } x_{l,k} \text{ é uma lacuna} \\ 1 & \text{caso contrário} \end{cases} \quad (2.9)$$

onde δ é o limiar de tolerância.

Para calcular a distância $d_{EDR}(\mathbf{x}_i, \mathbf{x}_l)$ entre duas séries temporais de tamanho n e m , respectivamente, pode-se realizar um algoritmo baseado em programação dinâmica, descrito de maneira geral como:

$$d_{EDR}(x_i, x_l) = \begin{cases} n & \text{se } m = 0 \\ m & \text{se } n = 0 \\ \min \left\{ \begin{array}{l} d_{EDR}(r(\mathbf{x}_i), r(\mathbf{x}_l)) + d_{EDR*}(x_{i,1}, x_{l,1}), \\ d_{EDR}(\mathbf{x}_i, r(\mathbf{x}_l)) + 1, \\ d_{EDR}(r(\mathbf{x}_i), \mathbf{x}_l) + 1 \end{array} \right\} & \text{caso contrário} \end{cases} \quad (2.10)$$

onde $r(\mathbf{x}_i)$ corresponde à série temporal $\mathbf{x}_i$ sem o seu primeiro ponto $x_{i,1}$.

Segundo [Chen, Özsu e Oria \(2005\)](#), EDR é capaz de tratar deslocamentos locais no tempo das séries temporais, além de ser mais robusta ao ruído que as distâncias DTW e LCS. Como pontos negativos desse método, pode-se citar sua complexidade quadrática e o fato de não ser uma métrica de distância. Essa última desvantagem foi mitigada em dois outros trabalhos, realizados por [Chen e Ng \(2004\)](#) e [Marteau \(2008\)](#), com os algoritmos *Edit Distance on Real Penalty* (ERP) e *Time Warp Edit Distance* (TWED), respectivamente.

Proposta por [Chen e Ng \(2004\)](#), a Distância de Edição em Penalidade Real (ERP) é uma métrica de distância associada com uma constante g , definida como uma penalização, aplicada quando lacunas são criadas, com o intuito de se obter um alinhamento ótimo. Capaz de lidar com *outliers* e deslocamentos na série temporal, esse algoritmo possui a vantagem de ser considerado como uma métrica de distância. A ERP entre duas amostras de dados pode ser descrita como:

Definição 2.2.3 (Distância de Edição em Penalidades Reais) Dadas duas séries temporais $\mathbf{x}_i$ e $\mathbf{x}_l$, a ERP entre duas amostras de dados é definida como:

$$d_{ERP*}(x_{i,j}, x_{l,k}) = \begin{cases} |x_{i,j} - x_{l,k}| & \text{se nem } x_{i,j} \text{ nem } x_{l,k} \text{ são lacunas} \\ |x_{i,j} - g| & \text{se } x_{l,k} \text{ é uma lacuna} \\ |x_{l,k} - g| & \text{se } x_{i,j} \text{ é uma lacuna} \end{cases} \quad (2.11)$$

onde g é uma constante.

Para calcular a distância $d_{ERP}(\mathbf{x}_i, \mathbf{x}_l)$ entre duas séries temporais, pode-se utilizar um algoritmo baseado em programação dinâmica, descrito de maneira geral como:

$$d_{ERP}(x_i, x_l) = \begin{cases} \sum_{k=1}^n |x_{l,k} - g| & \text{se } n = 0 \\ \sum_{j=1}^n |x_{i,j} - g| & \text{se } n = 0 \\ \min \left\{ \begin{array}{l} d_{ERP}(r(\mathbf{x}_i), r(\mathbf{x}_l)) + d_{ERP*}(x_{i,1}, x_{l,1}), \\ d_{ERP}(\mathbf{x}_i, r(\mathbf{x}_l)) + d_{ERP*}(x_{l,1}, g), \\ d_{ERP}(r(\mathbf{x}_i), \mathbf{x}_l) + d_{ERP*}(x_{i,1}, g), \end{array} \right\} & \text{caso contrário} \end{cases} \quad (2.12)$$

onde g é o termo de penalização, ou lacuna.

A segunda métrica de distância citada, também baseado na Distância de Edição refere-se ao algoritmo *Time Warp Edit Distance* (TWED). Esse algoritmo, assim como a distância ERP, é considerado como uma medida elástica de similaridade, uma vez que é capaz de lidar com deslocamentos temporais na série, além de ser uma métrica de distância. A elasticidade da TWED é controlada pelo parâmetro γ (rigidez) utilizando diferenças entre intervalos de tempo como parte do custo de emparelhamento local. Quanto maior a rigidez, mais o algoritmo se aproxima da Distância de Edição, ou seja, tendendo ao infinito, a TWED se torna equivalente à Distância de Edição. Por outro lado, uma rigidez igual a zero, resulta em uma distância $\mathcal{W}$, equivalente ao algoritmo DTW sem janelamento. O outro parâmetro utilizado no algoritmo TWED, denominado λ , é introduzido como um parâmetro de penalidade para operações de inserção e deleção de caracteres, similar à distância ERP. A distância de edição $\mathcal{W}$ entre duas amostras de dados pode ser descrita como:

Definição 2.2.4 (*Time Warp Edit Distance*) Dadas duas séries temporais $\mathbf{x}_i$ e $\mathbf{x}_l$, a TWED pode ser definida como:

$$d_{\lambda,\lambda}(x_i, x_l) = \min \left\{ \begin{array}{ll} d_{\lambda,\lambda}(h(\mathbf{x}_i), \mathbf{x}_l) & +d(x_{i,-1}, x_{i,-2}) + \lambda + \gamma \cdot d(t_{i,-1}, t_{i,-2}) \\ d_{\lambda,\lambda}(\mathbf{x}_i, h(\mathbf{x}_l)) & +d(x_{l,-1}, x_{l,-2}) + \lambda + \gamma \cdot d(t_{l,-1}, t_{l,-2}) \\ d_{\lambda,\lambda}(h(\mathbf{x}_i), h(\mathbf{x}_l)) & +d(x_{i,-1}, x_{l,-1}) + d(x_{i,-2}, x_{l,-2}) \\ & +\gamma \cdot (d(t_{i,-1}, t_{l,-1}) + d(t_{i,-2}, t_{l,-2})) \end{array} \right. \quad (2.13)$$

onde λ é a constante de penalidade, adicionada em caso de uma deleção ($\lambda \geq 0$), γ é o parâmetro de rigidez ($\gamma \geq 0$), $h(\mathbf{x}_i)$ corresponde à série temporal $\mathbf{x}_i$ sem o seu último termo, $x_{i,-1}$, cujo instante de tempo é definido como $t_{i,-1}$ e d é uma métrica $\mathcal{L}_p$.

Assim como a distância ERP, a métrica TWED é capaz de lidar com distorções temporais e é robusta ao ruído. Além disso, o parâmetro de rigidez, γ , auxilia o algoritmo a controlar deslocamentos admissíveis na série temporal, o que é muito útil para dados onde padrões locais importam.

Mais tarde, em 2013, [Stefan, Athitsos e Das \(2012\)](#) introduz o método *Move Split Merge* (MSM), o qual é baseado em operações de movimentação, divisão e fusão, as quais são semelhantes aos métodos de distância de edição discutidos. Mover é equivalente a substituir, ou seja, uma mudança no valor do elemento; dividir duplica o elemento, ou seja, adiciona o mesmo elemento após ele mesmo; a fusão une dois elementos consecutivos em um só. Cada operação possui um custo associado, exceto a operação de movimentação, cujo custo corresponde ao valor absoluto da diferença entre o elemento original e o novo elemento. A distância entre duas séries temporais corresponde ao menor custo entre as sequências de operações que transformam a primeira série temporal na segunda. MSM é uma métrica de distância, robusta a deslocamentos temporais e é capaz de lidar com séries temporais de diferentes tamanhos e séries multivariadas.

Como discutido até então, existem diversos algoritmos propostos para a medida de similaridade entre séries temporais, sendo alguns deles capazes de apresentar vantagens em relação a outros em determinado quesito, porém, desvantagens em outros quesitos considerados. [Bailly \(2018\)](#) apresenta uma comparação entre as principais medidas de similaridade citadas, levando em consideração a performance de cada um deles, além de outros pontos importantes, como capacidade de considerar deslocamentos temporais, robustez ao ruído e *outliers*, ser ou não uma métrica de distância, custo computacional e número de hiper-parâmetros. A Figura 3 apresenta, de maneira geral, uma síntese dos algoritmos de medidas de similaridade discutidos na presente seção.

As medidas de similaridade podem ser utilizadas diretamente entre as séries temporais analisadas, para a realização da tarefa de classificação. Muitas vezes, porém, previamente à aplicação de uma medida de similaridade, a série temporal pode ser submetida a um processo de extração de características, processo no qual a série pode ser representada em outro espaço. Nesse caso, a série temporal é previamente transformada em outra série, aplicando-se posteriormente, por exemplo, uma medida de similaridade, para então realizar-se a tarefa de classificação. Na seção seguinte, são apresentados alguns dos principais algoritmos de extração de características e representação de séries temporais, os quais podem ser aplicados em conjunto com uma medida de similaridade, ou serem utilizados para o processo de classificação propriamente dito.

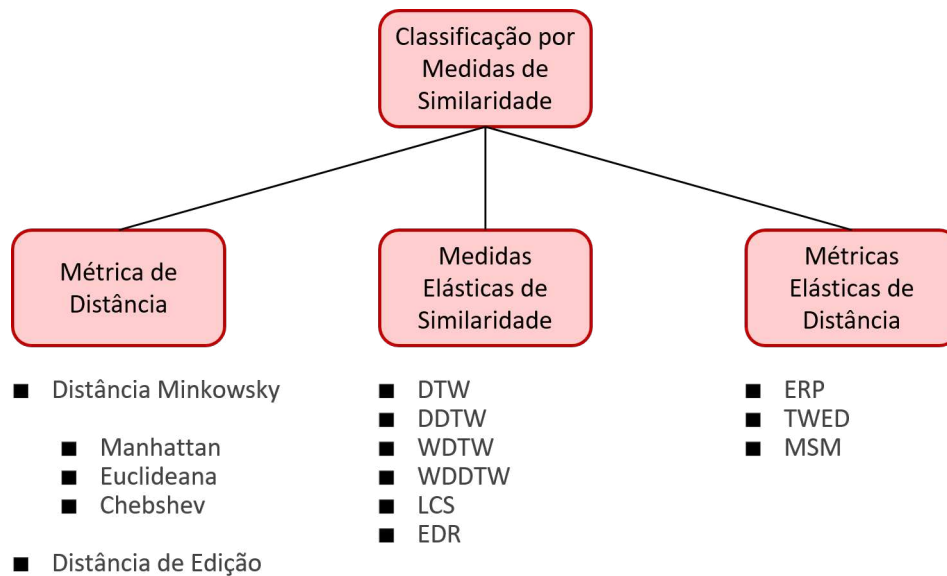


Figura 3 – Classificação geral dos algoritmos de medidas de similaridade discutidos.

2.3 Classificação baseada na extração de características

Os métodos baseados em extração de características têm o objetivo de representar a série temporal como um vetor de características para classificá-la. Existem diversos exemplos e finalidades desse tipo de metodologia, porém, um dos principais objetivos da representação de séries temporais é reduzir sua complexidade ao ser transformada para um espaço de características. A presente seção tem o objetivo de apresentar alguns dos principais exemplos de métodos baseados nessa abordagem, alguns dos quais serviram de base para as metodologias propostas no trabalho.

Ao se realizar uma tarefa de mineração de dados de séries temporais, pode-se considerar a própria série temporal, com seus dados brutos, como uma forma de representação, sendo essa a maneira mais natural de se representar a série (BAILLY, 2018). Um problema desse tipo de representação é que, muitas vezes, o número de informações contidas na série temporal é muito grande, de tal forma que as dimensões da série analisada afetam de maneira direta no tempo computacional necessário para a tarefa a ser realizada. Devido a esse problema, um dos principais objetivos dos algoritmos e técnicas baseadas na extração de características é a redução das dimensões dos dados originais (FU, 2011). No presente trabalho, serão abordados três tipos principais de metodologias baseadas na extração de características, para a posterior classificação: os métodos baseados em representação da série temporal como um vetor de características, representação baseada em dicionário de códigos e representação baseada em Shapelets.

2.3.1 Representação de séries temporais

Existem vários tipos de métodos para representação de séries temporais. Alguns deles realizam a extração de características da série em um determinado espaço de características como por exemplo os métodos baseados em transformadas (transformada de Fourier, transformada wavelet), outros realizam a discretização da série reduzindo suas dimensões. Alguns desses métodos, baseados nessas abordagens, serão discutidos adiante.

2.3.1.1 Métodos baseados na discretização por segmentação

Um dos métodos mais simples aplicados para representar uma série temporal foi discutido no trabalho de Åström (1969), denominado de re-amostragem (*Sampling*). Nesse método, divide-se a série temporal em pontos amostrais equidistantes uns dos outros, gerando uma série com menor complexidade em relação à série original, a qual, pode, por fim, ser utilizada no processo de classificação. Um dos problemas dessa estratégia de representação é que ela pode distorcer a forma da série, caso a taxa de amostragem seja muito baixa. Apesar de suas desvantagens, o método de re-amostragem serviu como base para diferentes outros estudos, realizados posteriormente. Em 2000, por exemplo, dois trabalhos realizados independentemente por Yi e Faloutsos (2000) e Keogh e Pazzani (2000) introduziram um método denominado *Piecewise Aggregate Approximation* (PAA). Assim como a re-amostragem, o método divide a série temporal em segmentos, com a diferença que ele computa a média entre os segmentos definidos para representar a série temporal, reduzindo, portanto, sua dimensão. Sua representação pode ser realizada como indicado na equação a seguir:

$$\hat{p}_k = \frac{1}{e_k - s_k + 1} \sum_{i=s_k}^{e_k} p_i \quad (2.14)$$

Já em 2001, Keogh et al. (2001) propõem uma extensão dessa abordagem, na qual a distância entre cada segmento é variável, adaptando-se à forma da série. Esse método ficou conhecido como *Adaptive Piecewise Constant Approximation* (APCA). Ainda baseada na estratégia de segmentação da série temporal, Keogh (1997), Keogh, Smyth et al. (1997), Smyth e Keogh (1997), Shatkay e Zdonik (1996) propuseram um método denominado *Piecewise Linear Representation* (PLR). Nesse método, após a segmentação da série, os pontos encontrados entre os segmentos são representados por uma regressão linear. As principais abordagens referente à representação linear são a interpolação linear e a regressão linear (FU, 2011). Sua premissa básica é a representação da série temporal como uma sequência de segmentos lineares, obtidos por uma regressão linear entre os pontos internos a cada segmento (BAILLY, 2018).

As abordagens citadas até então se referem à representação da série temporal como valores obtidos a partir de determinadas operações realizadas em cada segmento. Uma outra vertente de representação de séries temporais, ainda baseada na divisão da série em segmentos, realiza a representação de cada segmento obtido, em símbolos. Essa representação é obtida de tal maneira que, a série como um todo, é representada de maneira simbólica, ou seja, uma série temporal de símbolos. Um exemplo bastante aplicado desse tipo de representação é conhecido como *Symbolic Aggregate Approximation* (SAX), introduzido por [Lin et al. \(2003\)](#). Nesse método, primeiramente os dados da série temporal são normalizados para possuírem média zero e desvio padrão um e, após esse passo, reduz-se a dimensionalidade da série através do método PAA. Uma vez normalizada e com sua dimensionalidade reduzida, ocorre o processo de discretização da série em símbolos (normalmente representados como letras do alfabeto), de tal maneira que, cada símbolo, cuja quantidade é determinada previamente pelo usuário, corresponde a valores de média pertencentes a um determinado intervalo.

Uma das principais características dos métodos baseados em segmentação é a redução das dimensões da série temporal através da discretização da mesma. No caso do método SAX, a discretização e representação dos dados da série temporal como símbolos pode ser bastante aplicado com métodos de similaridade de cadeias de caracteres, como por exemplo os métodos baseados em distância de edição, discutidos na seção anterior. O princípio de analisar uma série temporal representada como uma discretização de seus dados, tanto numérica, como simbólica, serviu de base para a presente pesquisa, a qual tem o intuito de apresentar uma nova técnica de discretização, baseada em séries temporais fuzzy, antes de realizar o processo de classificação. A classificação de uma série representada como uma série temporal fuzzy será executada tanto como uma abordagem numérica, sendo a série temporal representada como números inteiros, quanto simbólica, sendo a série temporal representada como grupos fuzzy. Esse processo de representação será descrito com mais detalhes adiante, no capítulo 3.

2.3.1.2 Transformada de Fourier e Transformada Wavelet

As abordagens discutidas até então, realizam a representação das séries considerando análises desenvolvidas no domínio do tempo. Existem, porém, muitos trabalhos que extraem características de séries transformando-as e representando essas características em diferentes domínios. Uma das transformadas mais comuns para a representação de séries temporais é a transformada de Fourier, sendo a Transformada Discreta de Fourier a sua extensão mais utilizada, aplicada de maneira mais popular através da Transformada Rápida de Fourier, proposta por [Cooley e Tukey \(1965\)](#). Pode-se considerar, a grosso modo, que a Transformada Discreta de Fourier possui o objetivo de representar uma série como uma combinação linear de senos e cossenos, segundo a equação a seguir:

$$x_{ij} = \sum_{k=1}^n (a_k \cos(2\pi w_k j) + b_k \sin(2\pi w_k j)) \quad (2.15)$$

$$x_{ij} = \langle (a_0, b_0), (a_1, b_1), \dots, (a_n, b_n) \rangle \quad (2.16)$$

sendo a e b as amplitudes das senoides e w , a fase. A DFT de uma série temporal $\mathbf{x}_i$ pode ser representada pelos primeiros n coeficientes definidos.

Um trabalho pioneiro no assunto, sendo o primeiro a aplicar a Transformada de Fourier de uma série temporal, bem como sua classificação a partir de uma medida de similaridade, foi realizado por [Agrawal, Faloutsos e Swami \(1993\)](#). Em seu estudo, os autores utilizaram a transformada discreta de Fourier para representar a série temporal no domínio da frequência, e, por fim, utilizam a Distância Euclidiana como medida de similaridade entre as séries temporais. [Agrawal, Faloutsos e Swami \(1993\)](#) apontam as seguintes características favoráveis à utilização da transformada de Fourier: preservação a distância entre as séries analisadas, também obtida por qualquer série ortogonal; facilidade de ser calculada; capacidade de representar a energia do sinal em poucos coeficientes. Apesar das vantagens apresentadas, um dos problemas enfrentados por [Agrawal, Faloutsos e Swami \(1993\)](#) ao utilizar as DFT é assumir que os dados analisados possuem o mesmo tamanho. Novos estudos foram propostos posteriormente, para mitigar esse inconveniente e desenvolver novas ferramentas baseadas nessa abordagem, seja na melhoria do processo da transformada para representação da série ([FALOUTSOS; RANGANATHAN; MANOLOPOULOS, 1994](#); [RAFIEI; MENDELZON, 1997](#)), seja na medida de similaridade aplicada para a classificação, após a transformação da série ([LIN; SHIM, 1995](#); [YI; JAGADISH; FALOUTSOS, 1998](#)).

Métodos baseados na transformada de Fourier para a representação da série temporal antes do processo de classificação foram bastante utilizados, desde sua primeira aplicação. Mais tarde, porém, com o intuito de introduzir uma metodologia eficiente para substituir a Transformada Discreta de Fourier, [Chan e Fu \(1999\)](#) propõem a Transformada Discreta Wavelet (DWT). Essa abordagem de representação, geralmente apresenta, dentre outras vantagens, uma maior eficiência computacional que a DFT ([WU; AGRAWAL; ABBADI, 2000](#)). A base da transformada wavelet é a realização de translações e dilatações de uma determinada função, denominada *mother wavelet*, $\psi(x)$, onde as translações podem ser definidas como $\psi(z) \rightarrow \psi(z + 1)$ e as dilatações como $\psi(z) \rightarrow \psi(2 \cdot z)$. Em ([WU; AGRAWAL; ABBADI, 2000](#)), os autores demonstram que ambos os métodos apresentam resultados comparáveis, embora os métodos que aplicam DWT possuam menor complexidade que os métodos baseados em DFT.

2.3.2 Métodos baseados em dicionário de códigos

Na segunda década dos anos 2000, diversos algoritmos foram propostos, baseados na abordagem de representação por dicionários de códigos (*Codebooks*). Esse tipo de representação está fortemente ligado ao modelo *Bag-of-Words* (BoW), o qual consiste em representar uma instância através de um histograma de ocorrência de palavras, ignorando a ordem temporal dessas palavras, porém capturando informações estruturais de alto nível, utilizando tanto características locais quanto globais. Lin, Khade e Li (2012) propôs um método denominado *Bag-of-Patterns*, combinando a Aproximação Agregada Simbólica (SAX) com a técnica *Bag-of-Words*. Já Senin e Malinchik (2013) apresentam um algoritmo baseado em duas técnicas, SAX e VSM (*Vector Space Model*), sendo a técnica SAX responsável por representar a série temporal bruta como uma série simbólica e a técnica VSM, uma técnica similar à BoW, para criar o dicionário de códigos dos símbolos obtidos (BAILLY, 2018).

Ainda no início da segunda década de 2020, Baydogan, Runger e Tuv (2013) introduziram o algoritmo *Time Series Bag-of-Features* (TSBF), o qual extrai segmentos da série temporal em vários lugares e com diversos tamanhos, gerando um dicionário com esses padrões, utilizando-o para alimentar o classificador. O algoritmo TBSF pode ser considerado como um algoritmo que apresenta alta performance no processo de classificação de séries temporais. Outro tipo de abordagem utilizada com os algoritmos baseados em *Codebooks* é através da utilização de transformadas Wavelet, proposto por Wang et al. (2013) e a Aproximação Simbólica de Fourier, utilizada no algoritmo proposto por Schäfer (2015), *Bag-of-SFA-Symbols* (BOSS).

2.3.3 Shapelets

Inicialmente introduzidas por Ye e Keogh (2009), as *Shapelets* são subsequências de séries temporais, definidas, a grosso modo, como uma parte da série temporal que pode ser crucial para o processo de classificação. Em sua primeira aplicação, Ye e Keogh (2009) propõem um algoritmo de busca exaustiva para criar um classificador de árvore de decisão baseado em *shapelets*. Nesse algoritmo, as *shapelets* são escolhidas dentre todas as possíveis subsequências da série de um dado tamanho, consideradas como subsequências candidatas. As *shapelets* selecionadas são aquelas que possuem características de maior relevância referentes a cada classe. A medida de dissimilaridade entre uma determinada *shapelet* e a série temporal pode ser definida como a seguir.

Definição 2.3.1 (Dissimilaridade entre uma série temporal e uma *shapelet*) A medida de dissimilaridade entre uma série temporal $\mathbf{x}_i$ (de tamanho n) e uma *shapelet* $\mathbf{s}_k$ (de tamanho L - com $L < n$) é definida como:

$$d(\mathbf{x}_i, \mathbf{s}_k) = \min_{j=1, \dots, J} \frac{1}{L} \sum_{l=1}^L (x_{i,j+l-1} - s_{k,l})^2 \quad (2.17)$$

com $J = n - L + 1$.

Embora as *shapelets* apresentassem vantagens como interpretabilidade e rapidez no processo de classificação, o processo de descoberta das subsequências com o primeiro algoritmo proposto por Ye e Keogh (2009) demanda bastante tempo. Em 2011, porém, os mesmos autores apresentaram uma melhoria do algoritmo (YE; KEOGH, 2011), o qual aplica uma estratégia denominada *early abandon*, que, a grosso modo, cessa o cálculo da distância entre uma subsequência e uma determinada *shapelet* candidata assim que a menor distância encontrada, até então, for excedida, passando assim, para a próxima subsequência. Essa abordagem desenvolvida, bem como suas primeiras evoluções, foram porta de entrada para diversos outros trabalhos sobre o tema, e o consequente surgimento de novos temas baseados em *shapelets*. Dentre os mais relevantes, pode-se citar as Shapelets Rápidas (*Fast Shapelets*) (RAKTHANMANON; KEOGH, 2013), a Transformada *Shapelet* (*Shapelet Transform*) (LINES et al., 2012) e as *Shapelets* aprendidas (*Learning Shapelets*) (GRABOCKA et al., 2014).

As *Fast Shapelets*, propostas por Rakthanmanon e Keogh (2013), representam uma versão mais simples do algoritmo original das *shapelets*. A principal ideia no processo de descoberta de uma *fast shapelet* é, inicialmente, reduzir a dimensão da série temporal analisada, através de uma representação discretizada da série temporal, gerada pelo algoritmo de aproximação simbólica (SAX). Embora o algoritmo de *Fast Shapelets* seja mais rápido que o algoritmo original, os resultados obtidos não possuem expressiva diferença. Já em 2012, com a Transformada *Shapelet* (LINES et al., 2012), as subsequências são utilizadas para extração de características da série temporal e realização do processo de classificação. Esse algoritmo apresenta resultados melhores que o algoritmo original proposto por Ye e Keogh (2009) (BAILLY, 2018).

O algoritmo proposto por Grabocka et al. (2014), por sua vez, apresenta uma abordagem diferente dos discutidos até então. O principal propósito do método *Learning Shapelets* é de se aprender as subsequências, ao invés de procurar pelas melhores em um conjunto de subsequências candidatas (considerado, até então, como uma das principais desvantagens do algoritmo original, devido à busca exaustiva pelas *shapelets* candidatas). O método, baseado no gradiente descendente para o processo de aprendizado das *shapelets*, apresenta uma boa acurácia, comparado com o algoritmo original. As *Learning Shapelets*, assim como a *Shapelet Transform*, são os dois algoritmos baseados em *shapelets* mais desenvolvidos e com maior acurácia (BAILLY, 2018).

Os algoritmos discutidos até então são exemplos de alguns principais *benchmarks*

encontrados na literatura, referentes à tarefa de classificação de séries temporais, aplicando abordagens baseadas em medidas de similaridade e extração de características. Os próximos capítulos do trabalho apresentam diferentes abordagens para o processo de representação e/ou classificação. Inicialmente são discutidos os métodos de classificação propostos, baseados em representação da série temporal como uma imagem e classificados com uma rede neural convolucional, seguidos dos métodos de classificação baseados em medidas de similaridade, aplicados em séries temporais fuzzy. Esses métodos propostos serão, posteriormente, comparados a métodos de classificação já conhecidos como *benchmarks*, tais como a Distância Euclidiana e o método DTW.

3 CLASSIFICAÇÃO BASEADA EM SÉRIES TEMPORAIS FUZZY E MEDIDAS DE SIMILARIDADE

As séries temporais fuzzy são uma abordagem de séries temporais, bastante utilizadas para tarefas de previsão. O conceito de séries temporais fuzzy (*Fuzzy Time Series* - FTS) pode ser utilizado para se referir a uma série temporal composta por termos linguísticos, obtidos através de um processo baseado na teoria de conjuntos nebulosos, proposta por Zadeh (1965) (SILVA, 2019). As FTS foram inicialmente abordadas no trabalho de Song e Chissom (1991), cujos valores da série são representados por conjuntos Fuzzy, e desde então, têm sido amplamente utilizadas em muitas aplicações de previsão (SADAEI et al., 2019). Essa abordagem combina a teoria de conjuntos Fuzzy, desenvolvida por Zadeh (1965) e a análise estatística de séries temporais proposta por Song e Chissom (SONG; CHISSOM, 1991; SONG; CHISSOM, 1994), emergindo como uma alternativa prática e eficiente entre as ferramentas de previsão conhecidas (JR; JR, 2017). Eles têm sido comumente empregados na previsão de inscrição em universidades (SONG; CHISSOM, 1993a; SONG; CHISSOM, 1994; ISMAIL; EFENDI, 2011), mercado de ações (SADAEI et al., 2016; LEE; SADAEI; SUHARTONO, 2013; CHEN, 2014; SUN et al., 2015; TALAR-POSHTI et al., 2016; EFENDI; ISMAIL; DERIS, 2013), turismo (LEE; JAVEDANI et al., 2011), carga elétrica (ISMAIL; EFENDI; DERIS, 2015; SADAEI et al., 2017), dentre outros (SILVA, 2019).

Dado o reconhecimento dessa técnica em tarefas de previsão, um dos métodos propostos no presente trabalho é baseado na utilização de séries temporais fuzzy na tarefa de classificação de séries temporais. Entre os métodos discutidos na seção 2 muitos deles são baseados na extração de características da série temporal, na medida de similaridades entre séries temporais, ou mesmo em ambos. Nesse último caso, um determinado método de extração de características pode ser utilizado para gerar uma nova série temporal, para, posteriormente, essa nova série ser classificada por uma determinada medida de similaridade. Nesse contexto, o primeiro método proposto no presente trabalho, a ser discutido nesse capítulo refere-se à aplicação de séries temporais fuzzy para a extração de características da série. Uma vez realizada a extração de características, uma nova série é obtida, porém com seus valores fuzzificados. Essa nova série é, portanto, comparada com as demais séries temporais através de medidas de similaridade para realizar o processo de classificação. Esse procedimento está ilustrado na figura 4

Um ponto interessante ao se utilizar essa técnica é que, o processo de fuzzificação da série temporal, gera uma nova série capaz de filtrar eventuais ruídos presentes na série original. Com isso, a classificação através de séries temporais fuzzy pode apresentar melhor acurácia, por não levar em consideração esses ruídos. Para se entender o processo utilizada

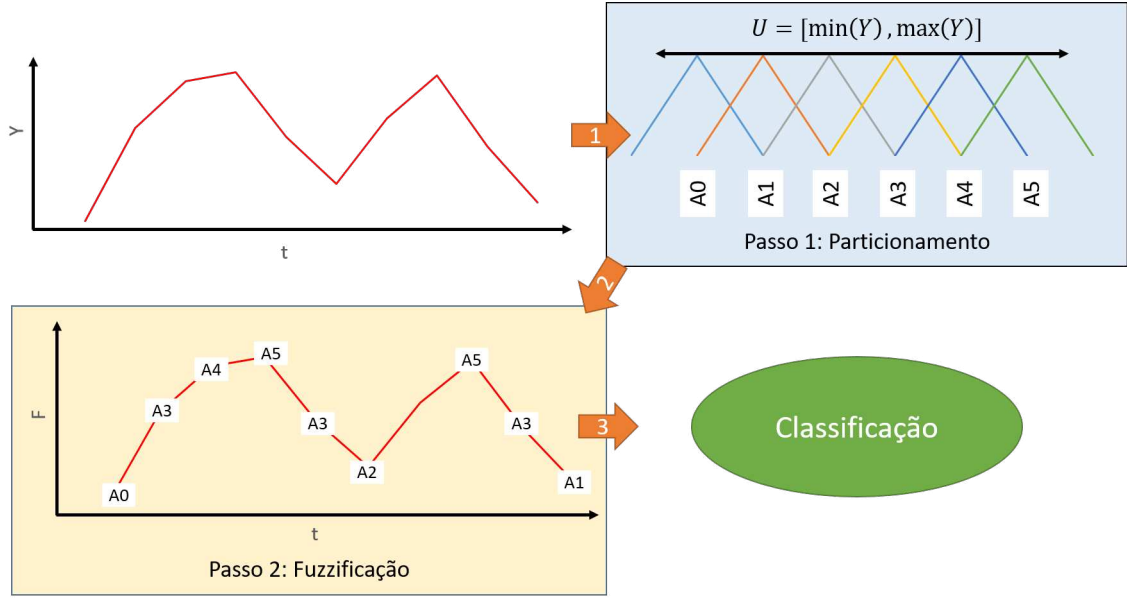


Figura 4 – Ilustração esquemática do processo de fuzzificação da série temporal até sua classificação.

Adaptado de [Silva \(2019\)](#)

no método proposto, o conceito de séries temporais fuzzy será abordado a seguir.

3.1 Séries Temporais Fuzzy

Dada a definição de um conjunto Fuzzy:

$$A_i = \{\mu_{A_i}(\chi)/\chi | \chi \in U\} = \int_U \mu_{A_i}(\chi)/\chi \quad (3.1)$$

onde, $\mu_{A_i} : U \rightarrow [0, 1]$ é a função de pertinência do conjunto Fuzzy A_i , $\chi \in U$ é um elemento genérico do conjunto Fuzzy A_i , $\mu_{A_i}(\chi)$ é o grau de pertinência de χ em A_i . A integral não deve ser interpretada algebricamente, pois essa notação apenas indica que A_i é uma coleção de pares ordenados ([SADAEI et al., 2019](#)).

[Song e Chissom \(1991\)](#) definem uma série temporal Fuzzy como:

Seja uma série temporal $y(t) : t = \dots, 0, 1, 2, \dots$ e seja o conjunto de valores $y(t)$, um sub-conjunto de números reais, formando o Universo de Discurso (domínio) pelo qual conjuntos Fuzzy $\mu_{A_i}, i = 1, \dots, N$, são definidos. Se $F(t)$ é uma coleção de conjuntos Fuzzy no tempo, então, $F(t)$ é chamado de Serie Temporal Fuzzy (FTS) definido em $y(t)$.

De forma similar, [Chen et al. \(1996\)](#), propuseram um modelo com operadores aritméticos simplificados, substituindo a abordagem mais complexa, introduzida por [Song e Chissom \(1991\)](#). Diversos modelos baseados na abordagem de séries temporais fuzzy foram propostos e utilizados ao longo dos anos, principalmente para a tarefa de previsão de valores futuros das séries temporais modeladas ([SINGH, 2017](#); [PANIGRAHI; BEHERA, 2020](#)). Em geral, os modelos para previsão de valores de séries temporais, baseados na abordagem FTS, seguem o procedimento apresentado no trabalho de [Chen et al. \(1996\)](#). Essa metodologia é dividida em 5 passos básicos:

Passo 1: Partição do Universo de Discurso em intervalos

Definição 3.1.1 (Universo de Discurso) *Seja L_{bd} e U_{bd} , respectivamente o limite inferior e superior dos dados de uma série temporal $\S$, pode-se definir o Universo de Discurso, U , como:*

$$U = [L_{bd}, U_{bd}] \quad (3.2)$$

Após a definição do comprimento das subdivisões, o Universo de Discurso pode ser dividido em diversos outros intervalos, sendo eles de mesmo tamanho ou não. Esse passo de divisão do Universo de Discurso é de grande importância, afetando significativamente a acurácia do modelo ([PANIGRAHI; BEHERA, 2020](#)). Essa divisão pode ser realizada de basicamente duas formas, através de intervalos constantes ou variados. [Panigrahi e Behera \(2020\)](#) apresenta em seu estudo uma tabela contendo métodos de previsão de séries temporais, dividindo-os de acordo com a característica do tamanho dos intervalos.

Passo 2: Definição de termos linguísticos para cada intervalo

Uma vez particionado o Universo de Discurso, deve-se definir um termo linguístico para cada um dos intervalos, considerando-se que os dados da série temporal estão distribuídos dentro desses intervalos. Seja n o número de partições, define-se n termos linguísticos $\tilde{A}_1, \tilde{A}_2, \dots, \tilde{A}_n$, os quais podem ser representados por conjuntos Fuzzy, seguindo a definição apresentada na equação (3.1). Após a definição dos termos linguísticos, calcula-se o grau de pertinência de cada um dos valores da série temporal em pertencer a cada uma das partições, de acordo com a função de pertinência definida para cada grupo. Uma maneira simples e muito utilizada de se definir a função de pertinência para cada intervalo, divididos como $a_1, a_2, \dots, a_n$, é através de funções de pertinência triangulares, como mostrado na equação (3.3).

$$\begin{aligned}
\tilde{A}_1 &= \frac{1}{a_1} + \frac{0,5}{a_2} + \frac{0}{a_3} + \dots + \frac{0}{a_{n-1}} + \frac{0}{a_n} \\
\tilde{A}_2 &= \frac{0,5}{a_1} + \frac{1}{a_2} + \frac{0,5}{a_3} + \dots + \frac{0}{a_{n-1}} + \frac{0}{a_n} \\
\tilde{A}_3 &= \frac{0}{a_1} + \frac{0,5}{a_2} + \frac{1}{a_3} + \dots + \frac{0}{a_{n-1}} + \frac{0}{a_n} \\
&\vdots \\
\tilde{A}_n &= \frac{0}{a_1} + \frac{0}{a_2} + \frac{0}{a_3} + \dots + \frac{0,5}{a_{n-1}} + \frac{1}{a_n}
\end{aligned} \tag{3.3}$$

Passo 3: Fuzzificação dos dados da série temporal

Após definidas as funções de pertinência para cada partição, e calculados os graus de pertinência de cada valor da série temporal em relação a cada uma das partições, pode-se realizar a fuzzificação dos valores da série temporal, construindo, então, a série temporal fuzzificada. Na série temporal fuzzificada, o valor linguístico referente a um determinado valor da série original será aquele relacionado ao maior grau de pertinência que esse valor obteve em relação a um determinado intervalo. Ou seja, caso o maior valor de pertinência de um dado valor da série temporal original ocorra para o intervalo a_k , com $k \in [1, n]$, a variável linguística atrelada a esse valor da série temporal será $\tilde{A}_k$.

Passo 4: Estabelecer as Regras de Inferência Fuzzy entre os valores fuzzificados da série temporal e criar os Grupos de Regras de Inferência Fuzzy

Definição 3.1.2 (Regras de Inferência Fuzzy) (*CHEN et al., 1996*) *Assuma que $F(t-1) = \tilde{A}_i$ e $F(t) = \tilde{A}_j$. A relação entre $F(t)$ e $F(t-1)$ é denominada como uma regra de inferência fuzzy (Fuzzy Logic Relationship - FLR), que pode ser representada como:*

$$A_i \rightarrow A_j, \tag{3.4}$$

onde $\tilde{A}_i$ e $\tilde{A}_j$ se referem ao lado esquerdo (antecedente) e lado direito (consequente) da regra de inferência, respectivamente

A definição 3.1.2, apresentada anteriormente, se refere a uma regra referente a apenas uma variável, denominada regra de inferência fuzzy de primeira ordem. É possível também que a FLR seja de ordem n , relacionada a mais de um antecessor, como descrito na definição 3.1.3

Definição 3.1.3 (FLR de maior ordem) *Assume-se que $F(t)$ é causada por $F(t-1)$, $F(t-2)$, ..., e $F(t-n)$, com $n > 0$. A regra de inferência fuzzy de ordem n pode ser representada por:*

$$F(t - n), \dots, F(t - 2), F(t - 1) \rightarrow F(t). \quad (3.5)$$

Definição 3.1.4 (Grupos de Regras de Inferência Fuzzy) *Considerando-se as seguintes regras de inferência:*

$$\begin{aligned} A_i &\rightarrow A_{k1}, \\ A_i &\rightarrow A_{k2}, \\ &\vdots \\ A_i &\rightarrow A_{km}, \end{aligned}$$

Segundo [Chen et al. \(1996\)](#), as regras de inferência que possuem o mesmo antecedente podem ser agrupadas em um único grupo de regra de inferência (Fuzzy Logic Relationship Group - FLRG). Portanto, pode-se concluir que para as regras de inferência estabelecidas acima:

$$A_i \rightarrow A_{k1}, A_{k2}, \dots, A_{km}.$$

Passo 5: Defuzificação e previsão de valores futuros

[Song e Chissom \(1993b\)](#) apresentaram uma abordagem para a defuzificação de uma série temporal Fuzzy a partir de um operador de composição max-min, através da equação a seguir:

$$Y(t) = Y(t - 1) \circ R \quad (3.6)$$

onde $Y(t - 1)$ é o valor fuzzificado no tempo $t - 1$, $Y(t)$ é o valor atual a ser previsto, " $\circ$ " é o operador de composição max-min e " R " é a união de regras fuzzy. Como esse método se mostrou complicado, diversas outras abordagens foram propostos por outros autores ([SINGH, 2017](#)). Um deles, o qual utiliza operadores matemáticos mais simples, foi proposto por [Chen et al. \(1996\)](#) e utiliza o seguinte procedimento:

Caso no tempo $t - 1$, o valor fuzzificado de $\tilde{A}_i$ possui apenas um valor de consequente dentre todas as regras de inferência associadas a ele, ou seja, $\tilde{A}_i \rightarrow \tilde{A}_j$, então o valor defuzificado para $\tilde{A}_j$ será o ponto médio da função de pertinência atrelada ao intervalo a_j , C_j

Para o caso do valor fuzzificado de $\tilde{A}_i$ estar ligado a um grupo de regras de inferência, ou seja, $\tilde{A}_i \rightarrow \tilde{A}_{j1}, \tilde{A}_{j2}, \dots, \tilde{A}_{jp}$, encontra-se o ponto médio dos intervalos atrelados

a cada um dos consequentes do FLRG $(a_{j1}, a_{j2}, \dots, a_{jp})$, definidos como $C_{j1}, C_{j2}, \dots, C_{jp}$. O valor previsto para o intervalo t pode ser estimado como:

$$Y(t) = \left[\frac{C_{j1} + C_{j2} + \dots + C_{jp}}{p} \right] \quad (3.7)$$

onde p representa o número de conjuntos fuzzy no grupo de regras de inferência.

3.1.1 FTS na classificação de séries temporais

No presente estudo, o procedimento apresentado por [Chen et al. \(1996\)](#) será utilizado para a fuzzificação dos valores da série temporal, formando-se a série temporal fuzzy, a partir da qual, o processo de classificação é realizado, podendo ser ilustrado de maneira genérica na Figura 4. Os passos subsequentes, referentes à criação das regras de inferência e defuzzificação não serão aplicados para a tarefa de classificação das séries temporais. Devido à sua importância para o presente trabalho, o particionamento do Universo de Discurso, a fuzzificação da série, bem como as estratégias utilizadas para classificação, a partir de uma série temporal fuzzy, serão descritas mais detalhadamente adiante.

3.1.2 Definição dos hiperparâmetros

O processo de particionamento do Universo de Discurso pode ser considerado um dos passos mais importantes da obtenção da série temporal fuzzy ([SILVA, 2019](#)), cujo principal objetivo é a divisão do Universo de Discurso U em funções de pertinência definidas. As funções de pertinência serão, portanto, utilizadas na consequente obtenção dos termos linguísticos da série temporal fuzzy.

A definição mais intuitiva do Universo de Discurso U foi anteriormente discutida na equação (3.2), encontrada na definição 3.1.1. Segundo [Silva \(2019\)](#), porém, é usual que os limites inferiores e superiores do Universo de Discurso sejam excedidos por uma margem de confiança, l_d e u_d , respectivamente, de tal maneira que o intervalo utilizado é definido como $U = [\underline{L}_{bd}, \bar{U}_{bd}]$. Assim, os valores de $\underline{L}_{bd}$ e $\bar{U}_{bd}$ podem ser dados como $\underline{L}_{bd} = \min(Y) - l_d$ e $\bar{U}_{bd} = \max(Y) + u_d$, onde $l_d = \min(Y) \cdot 0,2$ e $u_d = \max(Y) \cdot 0,2$. Esse processo de definir as margens excedentes (l_d e u_d) auxilia no processo de fuzzificação, sendo capaz de acomodar flutuações nas extremidades do Universo de Discurso([SILVA, 2019](#)).

Uma vez determinado o intervalo do Universo de Discurso, o próximo passo é definir as funções de pertinência (μ) utilizadas na fuzzificação da série temporal. Diversos autores apresentam vários tipos de funções de pertinência ([JANG; SUN, 1997](#); [KLIR](#); [YUAN, 1995](#); [PEDRYCZ; GOMIDE, 2007](#)), como por exemplo, as funções *singleton*, triangulares,

trapezoidais, gaussianas, sino generalizadas. No presente trabalho foi utilizada a função triangular para o cálculo das pertinências dos valores da série temporal a cada uma das partições do Universo de Discurso. Seu cálculo pode ser dado segundo a equação abaixo:

$$\mu(x, a, b, c) = \begin{cases} 0 & \text{se } x \leq a \\ \frac{x-a}{b-a} & \text{se } a \leq x \leq b \\ \frac{c-x}{c-b} & \text{se } b \leq x \leq c \\ 0 & \text{se } c \geq x \end{cases} \quad (3.8)$$

sendo a, b e c os vértices do triângulo (a o limite inferior, c o limite superior e b o ponto intermediário entre a e c , cujo valor de pertinência é igual a 1).

O próximo hiper-parâmetro a ser definido é o número de partições do Universo de Discurso k . Esse é um parâmetro que também tem importância no processo de partição, fuzzificação e criação das regras fuzzy (DURU; YOSHIDA, 2012; SILVA, 2019), sendo que um pequeno valor dessa variável gera poucos grupos fuzzy para representar os valores da série temporal, resultando em um sub-ajuste do modelo. Por outro lado, um número alto de partições pode resultar na consideração de ruídos e pequenas flutuações da série temporal original pela série temporal fuzzy, e consequente sobre-ajuste do modelo (SILVA, 2019). Nesse sentido, o número de partições k deve ser definido de tal forma a evitar esses tipos de inconvenientes, sendo aconselhável que ele seja otimizado para cada tipo de problema, a fim de se obter um compromisso entre acurácia e parcimônia do modelo (menor quantidade possível de variáveis, necessárias para uma boa interpretação do mesmo). Nos primeiros resultados obtidos na presente pesquisa, o número de intervalos definidos para particionar o Universo de Discurso se manteve fixo, como 10 intervalos. Definiu-se esse valor ao longo das simulações, o qual apresentou bons resultados em relação a outros testados, além de ser um número não muito grande de intervalos, a ponto de reduzir demasiadamente a parcimônia do modelo. Será proposto, para a continuidade da pesquisa, a otimização do valor desse hiperparâmetro para cada base de dados estudada, a fim de investigar seu efeito na acurácia obtida. Por fim, o último parâmetro a ser definido antes do processo de fuzzificação da série temporal em si, é o método de particionamento do Universo de Discurso (II). Silva (2019) discute sobre diferentes metodologias para se realizar essa tarefa: Partição por *Grid* (*Grid Partitioning*), originalmente proposta por Song e Chissom (1993a), a qual utiliza intervalos k de tamanhos iguais; partição por Entropia (*Entropy Partitioning*), apresentada por Cheng, Chang e Yeh (2006), onde os autores utilizam um valor fixo para k , cujos pontos centrais são definidos pela entropia dos dados; técnicas baseadas em *clusterização*, como *Fuzzy-C-Means*, por exemplo (LI; CHENG; LIN, 2008; ASKARI; MONTAZERIN, 2015; BAS et al., 2015; SUN et al., 2015; YOLCU; LAM, 2017).

Seguindo o que foi exposto por Silva (2019), a fim de se utilizar uma abordagem inicial mais simples e com baixo custo computacional, na presente pesquisa, para a obtenção

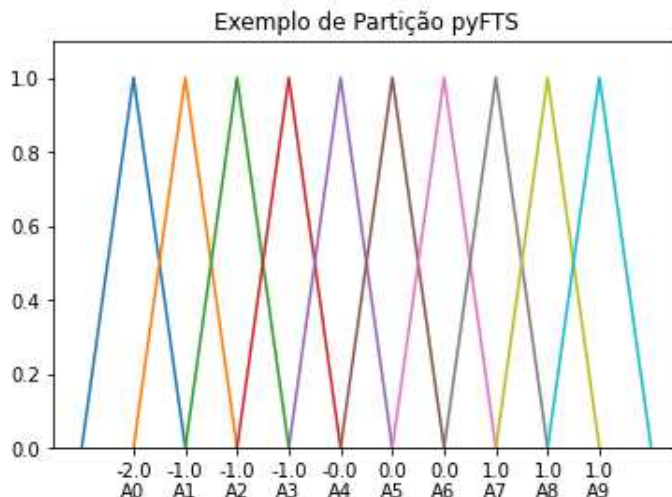


Figura 5 – Partição do Universo de Discurso com o método Grid (pyFITS)

dos resultados iniciais, foram aplicadas apenas abordagens baseadas em partição por Grid. Para tal, foi utilizada a partição por *Grid* triangular, obtida diretamente da aplicação para Python PyFITS, apresentada por [Silva \(2019\)](#). Nesse método, utilizam-se os conceitos introduzidos na presente seção, através da metodologia apresentada por [Song e Chissom \(1993a\)](#), com k intervalos igualmente espaçados, divididos dentro do Universo de Discurso U . A Figura 5 apresenta um exemplo de particionamento do Universo de Discurso com a biblioteca PyFITS para uma série temporal exemplo, considerando $k = 10$.

Uma vez definidos os passos necessários para a fuzzificação da série temporal, pode-se tratar do processo de classificação propriamente dito. A primeira proposta de classificação de séries temporais utilizando FTS, a qual será abordada no restante desse capítulo, é baseada na aplicação de séries temporais fuzzy em medidas de similaridade. Para isso serão aplicadas três das principais medidas abordadas no capítulo 2, a distância Euclideana, o DTW e o método LCSS. A aplicação de cada um deles será explicada com mais detalhes adiante, nesse capítulo.

3.2 Estratégias de classificação baseadas em medidas de similaridade propostas

O foco do presente capítulo é apresentar as bases da fuzzificação das séries temporais, abordagem fundamental para a primeira metodologia proposta. O primeiro passo realizado para a classificação da série temporal é o processo de fuzzificação. Com a fuzzificação da série temporal, uma nova série é formada, porém com valores linguísticos, os quais representam cada grupo fuzzy ao qual seus valores originais pertencem. Uma vez realizada a fuzzificação, e obtida a série temporal fuzzy, o processo de classificação pode ser realizado. Esse processo foi ilustrado na Figura 4.

A representação da série temporal como uma série temporal fuzzy, depois da

fuzzificação de seus valores, gera uma nova série temporal com valores simbólicos, representando grupos fuzzy aos quais cada valor da série original pertencem. Esses grupos podem, portanto, serem descritos por um determinado símbolo, como por exemplo $A_1, A_2, \dots, A_n$, dando origem à série temporal fuzzy. A série temporal fuzzificada é, então submetida a medidas de similaridade para que possa se realizar a tarefa de classificação. Duas abordagens distintas, que utilizam conceitos apresentados no capítulo 2, foram propostas:

Representação da FTS como valores referentes aos grupos fuzzy

Uma vez obtida a série temporal fuzzy, caracterizada pelos grupos fuzzy definidos no processo de partição do Universo de Discurso, a representação simbólica da série pode ser convertida em valores inteiros, os quais se referem a cada grupo fuzzy:

$$\{A_1, A_2, \dots, A_n\} \rightarrow \{1, 2, \dots, n\}$$

Esse processo de representação da série temporal pode ser considerado semelhante aos métodos de aproximação por partes, apresentados no capítulo 2, uma vez que seu objetivo é discretizar a série original, gerando uma nova série temporal menos complexa, cujas características foram aproximadas a grupos pré-definidos. A nova série temporal obtida pode ser, portanto, classificada por uma medida de similaridade. No presente trabalho, os resultados foram obtidos aplicando-se a estratégia *Nearest Neighbors* (K-NN) com k igual a 1, associada tanto com a distância Euclideana quanto com a distância $\mathcal{W}$ (DTW). A Figura 6 apresenta um exemplo da representação dos dados brutos de uma série temporal como uma série temporal fuzzy, fuzzificada pelo método Grid da API PyFTS, com número de partições igual a 10. O método de fuzzificação foi aplicado à primeira instância de treinamento da base *DodgerLoopGame* do banco de dados UCR (DAU et al., 2018).

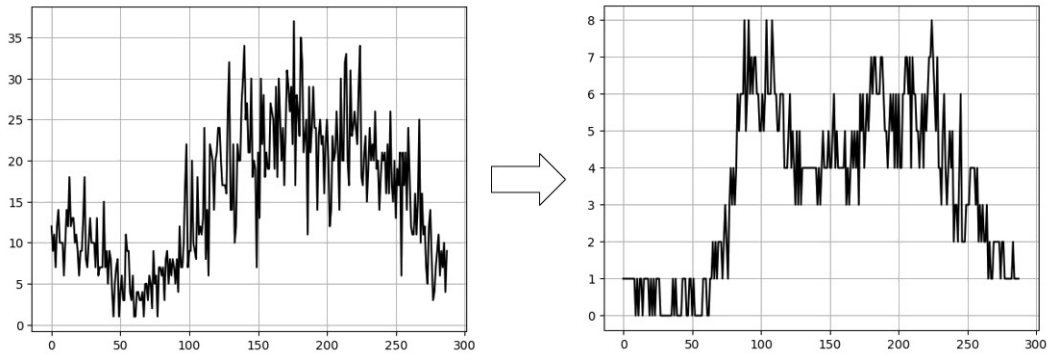


Figura 6 – Exemplo de representação de uma série temporal como uma série temporal Fuzzy, aplicados aos dados *DodgerLoopGame* (DAU et al., 2018). A série temporal à esquerda se refere aos dados brutos da primeira instância de treinamento da base utilizada; a figura à direita se refere à aos dados brutos fuzzificados e representados como o índice de seus grupos fuzzy.

Representação da FTS de forma simbólica

Além da representação da série temporal através de números inteiros, referentes aos grupos fuzzy definidos no processo de fuzzificação, propõe-se também sua representação através dos valores simbólicos obtidos durante esse processo: $\{A_1, A_2, \dots, A_n\}$. Para a tarefa de classificação, porém, foi utilizado um método de medida de similaridade baseado em Distância de Edição, também discutido no capítulo 2. Esse método foi apresentado como LCSS (*Longest Common Subsequence*).

Os resultados obtidos através da aplicação de medidas de similaridade em uma série temporal fuzzy serão mais detalhados posteriormente no capítulo 6, onde serão também comparados com os demais métodos propostos, bem como com alguns dos métodos considerados como *benchmarking*.

4 MÉTODOS BASEADOS EM IMAGENS

Os métodos apresentados na seção anterior utilizam uma medida de similaridade como classificador para o processo de classificação de séries temporais, seja essa medida aplicada à série bruta ou sobre uma representação da série fuzzificada. A presente seção trata da utilização de métodos que realizam a transformação da série temporal em uma imagem, e, a partir da imagem gerada, realizar a classificação. Para esse processo, porém, é necessário utilizar um classificador capaz de lidar com esse tipo de representação, e, para isso, vários métodos já foram propostos com o intuito de realizar a análise de imagens, bem como se realizar esse processo com maior acurácia (LU; WENG, 2007; NATH et al., 2014).

Dentre os diversos métodos apresentados para a tarefa de classificação de imagens, alguns dos mais comuns são os métodos baseados em redes neurais profundas, as quais, nas últimas décadas, vêm sendo muito estudadas e consideradas como um tipo emergente de rede neural, principalmente devido ao crescente interesse gerado pelo assunto de redes neurais e sua capacidade de aprender mapeamentos complexos e não lineares (AL-MAHASNEH; ANAVATTI; GARRATT, 2017; LECUN; BENGIO et al., 1995). Com o crescente número de pesquisas no assunto, diferentes métodos de aprendizado profundo podem ser encontrados na literatura, os quais têm se destacado por permitirem o processo e aprendizado de informação pelas máquinas, a partir de dados em diversos formatos (REZENDE, 2021). Alguns dos exemplos mais comuns desse tipo de arquitetura podem ser citadas, como as Redes Neurais Recorrentes (*Recurrent Neural Networks* - RNN) e as Redes Neurais Convolucionais (*Convolutional Neural Networks* - CNN). No que compete à visão computacional e classificação de imagens, essa última atingiu bastante sucesso prático, sendo, portanto, amplamente adotada pela comunidade acadêmica (LECUN; BENGIO; HINTON, 2015). Tendo em vista os bons resultados apresentados pelas Redes Neurais Convolucionais no campo da visão computacional, esse tipo de ferramenta foi bastante abordada e aplicada nos classificadores baseados em imagens estudados no presente trabalho. Uma breve contextualização sobre o assunto será realizada adiante, na presente seção.

4.1 Redes Neurais Convolucionais

As CNN são um tipo de rede neural de aprendizagem profunda com rede direta (*Feedforward*), capaz de aplicar uma variedade de perceptrons multicamada utilizando um processamento mínimo (SADAEI et al., 2019). Esse tipo de rede neural consiste em uma variação das redes de Perceptrons de Múltiplas Camadas (MLP), e são inspiradas no

processo biológico de processamentos de dados visuais (VARGAS; PAES; VASCONCELOS, 2016). As redes neurais convolucionais geralmente possuem três tipos de camadas: as camadas convolucionais, as camadas de *pooling* e as camadas totalmente conectadas, tal que, cada uma delas possui um conjunto de filtros (*kernel*), responsáveis por extrair características das imagens de entrada, sendo possível a criação de diversos mapas de camadas de convolução e *pooling*. Cada mapa de característica possui um conjunto de pesos compartilhados, diminuindo, assim, a complexidade da CNN (PENHA et al., 2018). A Figura 7 apresenta um exemplo de arquitetura de uma rede neural convolucional com aplicação das camadas convolucional, *pooling* e totalmente conectada.

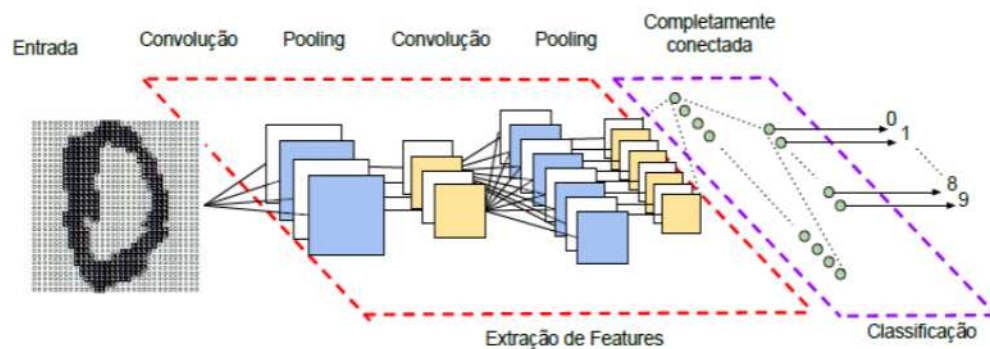


Figura 7 – Exemplo de arquitetura de uma rede neural convolucional.

Fonte: Vargas, Paes e Vasconcelos (2016)

4.1.1 Camada Convolucional

Após receber a imagem a ser analisada, a primeira camada, responsável pela extração de informações através de filtros, chamados de *kernels*, é a camada convolucional. Esses filtros, de dimensões $n \times n$, são deslocados ao longo da imagem, realizando a operação de convolução, a qual consiste em multiplicações matriciais entre a imagem ou a saída de uma camada intermediária e o *kernel*, gerando um mapa de característica, denominado *Feature Map* (REZENDE, 2019). Esse processo pode ser ilustrado a seguir na Figura 8.

O filtro é deslocado ao longo da matriz de entrada, realizando a operação de convolução matricial (*Conv*), representada por um asterisco (*), até que todos os elementos da matriz de saída resultante sejam calculados. Para essa operação, existe a definição de dois parâmetros: o passo, responsável por descrever quantos *pixels* cada deslocamento percorre; e o *padding*, relativo ao preenchimento das bordas da imagem, geralmente com o valor zero ou repetindo-se o valor das bordas. Esses parâmetros estão diretamente ligados à dimensão da matriz de saída e às informações consideradas durante a convolução (REZENDE, 2019; REZENDE, 2021).

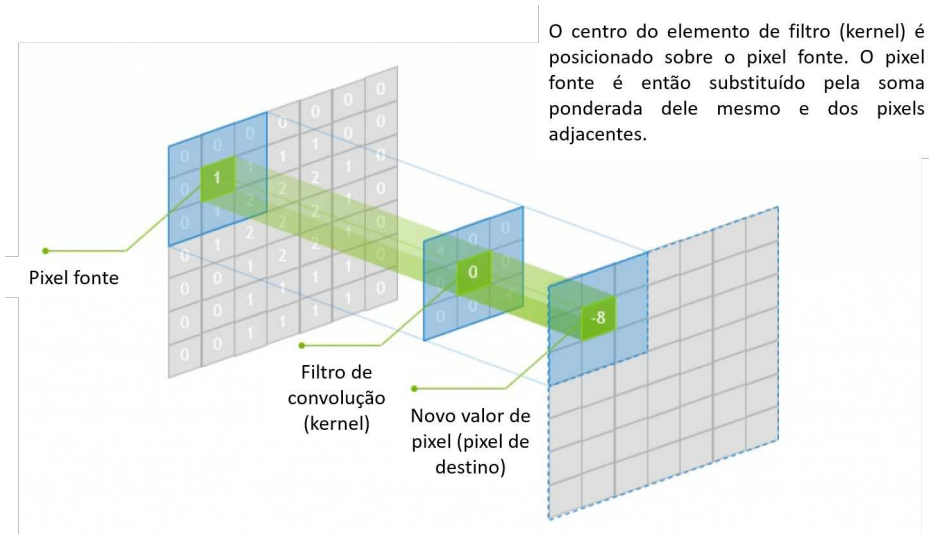


Figura 8 – Exemplo de aplicação de uma camada convolucional.

Adaptado de [Martin \(2018\)](#)

4.1.2 Camada de *Pooling*

Após finalizado o processo de convolução pela camada convolucional ao longo de toda a matriz de entrada, pode-se dar início um novo processo por um outro tipo de camada, sendo nesse caso, geralmente utilizada uma camada do tipo *pooling*. A camada de *pooling* (redução) é geralmente aplicada após uma camada convolucional, ou entre duas camadas convolucionais, recebendo diversos mapas de características e aplicando uma operação de *pooling* em cada um deles. Seu principal intuito é de reduzir o tamanho da imagem, simplificando a mesma, enquanto preserva suas características importantes. Esse processo torna a rede computacionalmente menos exigente, aumentando sua velocidade e reduzindo risco de *overfitting* ([GUO et al., 2016](#)). Existem diferentes tipos de operações de redução, como *max pooling*, *average pooling*, *min pooling*, etc. as quais, respectivamente, mantêm a máxima, média e mínima característica da sub-região considerada do mapa de características ([BAILLY, 2018](#)). Segundo [Scherer, Müller e Behnke \(2010\)](#), a técnica de *max pooling* pode auxiliar em uma convergência mais rápida da rede, uma vez que ela seleciona as melhores características invariáveis e tem capacidade de generalização superior ao *average pooling* ([REZENDE, 2019](#)).

4.1.3 Camada Totalmente Conectada

As camadas totalmente conectadas são geralmente as últimas camadas de uma CNN, e são utilizadas para realizar a classificação propriamente dita, baseada nas características extraídas pelas camadas anteriores, existindo, portanto, pelo menos uma camada totalmente conectada no final de uma rede neural convolucional. Assim como em uma

rede neural convencional, as camadas totalmente conectadas ligam cada elemento de sua camada a cada um dos elementos da camada anterior, como demonstrado na Figura 9.

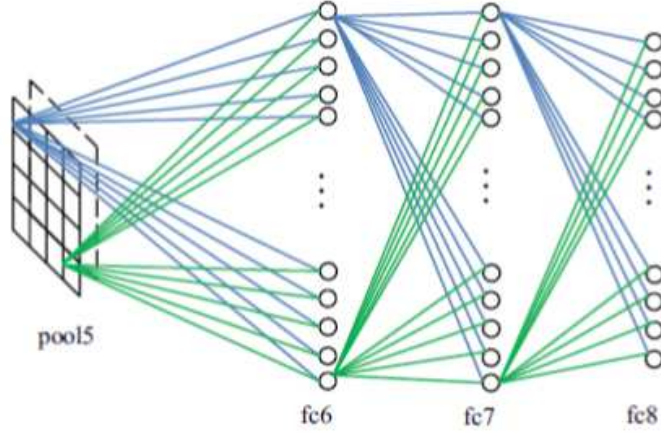


Figura 9 – Exemplo de uma camada totalmente conectada.

Fonte: [Guo et al. \(2016\)](#)

4.2 Representação de séries temporais em imagens

Para realizar a classificação através de uma Rede Neural Convolutiva é necessário transformar o vetor de entrada composto pela série temporal em uma matriz (imagem). Existem diversos algoritmos propostos para este processo, como pode ser visto em ([FAOUZI, 2022](#)). Esta seção discute alguns dos principais algoritmos existentes na literatura.

4.2.1 Recurrence Plots

Proposto por [Eckmann et al. \(1995\)](#), *recurrence plots* é uma ferramenta utilizada para fornecer informações sobre séries temporais de sistemas dinâmicos através de trajetórias, definidas como uma subsequência de valores igualmente espaçados, dados por ([FAOUZI, 2022](#)):

$$\forall i \in \{1, \dots, n - (m - 1)\tau\}, \vec{x}_i = (x_i, x_{i+\tau}, \dots, x_{i+(m-1)\tau}) \quad (4.1)$$

onde m é o comprimento da trajetória e τ é o intervalo de tempo entre pontos de tempo consecutivos na trajetória (atraso de tempo).

O gráfico de recorrência R é uma matriz binária de dimensões $\mathcal{N} \times \mathcal{N}$, cujos valores 0 ou 1 dependem da comparação entre cada trajetória de uma série temporal com um valor limite. Sempre que a distância entre um par (x_i, x_j) de caminhos for menor que

este limite, o valor da função R é 1 e caso contrário, seu valor é 0. Essa relação é dada pela equação:

$$\forall i, j \in \{1, \dots, n - (m - 1)\tau\}, R_{i,j} = \infty(\|\vec{x}_1 - \vec{x}_2\|_2 < \varepsilon) \quad (4.2)$$

onde ε é o limite usado para criar os valores binários da matriz e ∞ é a função indicadora com valor um se a distância for menor que o limite e 0 caso contrário.

4.2.2 Gramian Angular Field

Proposto por Wang e Oates (2015), o *Gramian Angular Field* (GAF) é um método baseado na representação da série temporal em coordenadas polares, diferentemente das trajetórias em espaço de fases utilizadas nos *recurrence plots*. Considerando uma série temporal $X = (x_1, \dots, x_n)$, para a construção do GAF, seus valores são inicialmente normalizados dentro do intervalo $[a, b]$, onde $-1 \leq a < b \leq 1$. Esse procedimento pode ser obtido da seguinte forma:

$$\forall i \in \{1, \dots, n\}, \tilde{x}_i = a + (b - a) \times \frac{x_i - \min(X)}{\max(X) - \min(X)} \quad (4.3)$$

Os valores de a e b estão relacionados à escala da série temporal. Se a escala for relevante, esses valores devem ser definidos de forma adequada; caso contrário, normalmente adota-se $[a, b] = [-1, 1]$ ou $[a, b] = [0, 1]$ (FAOUZI, 2022).

A série temporal normalizada, $\tilde{X}$, é então representada em coordenadas polares, com o raio relacionado à quantidade de pontos da série e os ângulos relacionados aos valores normalizados:

$$\forall i \in \{1, \dots, n\}, r_i = \frac{i}{n} \quad (4.4)$$

$$\forall i \in \{1, \dots, n\}, \phi_i = \arccos(\tilde{x}_i) \quad (4.5)$$

Como o GAF é uma matriz $n \times n$, sua representação pode ser desafiadora quando n é grande. Wang e Oates (2015) propuseram dois métodos para reduzir a série temporal de n para m pontos, sendo m o tamanho desejado do GAF (FAOUZI, 2022). As técnicas propostas são chamadas de *Gramian Angular Summation Fields* (GASF) e *Gramian Angular Difference Fields* (GADF), definidas como:

$$\forall i, j \in \{1, \dots, n\}, GASF_{i,j} = \cos(\phi_i + \phi_j) \quad (4.6)$$

$$\forall i, j \in \{1, \dots, n\}, \text{GADF}_{i,j} = \sin(\phi_i - \phi_j) \quad (4.7)$$

Os *Gramian Angular Fields* foram utilizados para gerar imagens para classificação de séries temporais em (WANG; OATES, 2015), onde foram usados como entrada de uma *Tiled Convolutional Neural Network*, e também por Karimi-Bidhendi, Munshi e Munshi (2018), acoplados a um modelo *Inception v3* pré-treinado seguido de um perceptron multicamadas (FAOUZI, 2022).

4.2.3 Markov Transition Field

O *Markov Transition Field* (MTF), também proposto por Wang e Oates (2015), é um método que assimila a série temporal como uma cadeia de Markov após a discretização. Dada uma série temporal $X = (x_1, \dots, x_n)$, os Q quantis são identificados, e cada valor x_i da série é atribuído ao respectivo intervalo q_i , onde $j \in [1, Q]$, sendo Q o número de intervalos. Esse processo resulta em uma série temporal discretizada de tamanho n . Após a discretização, uma matriz $Q \times Q$ é construída, relacionando as transições entre os quantis como uma cadeia de Markov de primeira ordem. A matriz é normalizada tal que $\sum_j w_{ij} = 1$, onde w_{ij} representa a frequência com que um ponto do quantil q_i é seguido por um ponto do quantil q_j , resultando na matriz de transição de Markov W .

Considerando apenas as frequências de transição, a matriz de transição de Markov não considera a distribuição temporal da série X , o que leva à perda de informações. Para mitigar esse problema, Wang e Oates (2015) definem o *Markov Transition Field* (MTF), obtido pela projeção da matriz de transição de Markov para uma matriz $n \times n$. Cada valor do MTF representa uma probabilidade de transição associada ao par (x_i, x_j) , ou seja, a probabilidade da transição $q(x_i) \rightarrow q(x_j)$:

$$\forall i, j \in \{1, \dots, n\}, \text{MTF}_{i,j} = P(q(x_j) \mid q(x_i)) \quad (4.8)$$

Os *Markov Transition Fields* foram utilizados no trabalho de Wang e Oates (2015) para representar séries temporais como imagens, com posterior classificação por uma *Tiled Convolutional Neural Network* (FAOUZI, 2022).

4.3 Metodologias propostas para representação das séries temporais em imagens

Uma das propostas do presente trabalho é apresentar métodos para a classificação de séries temporais que possam auxiliar na abertura para novas pesquisas científicas referentes a esse assunto. Apesar dos bons resultados apresentados pelas redes neurais

profundas, especialmente a revolução no campo de visão computacional, gerada pelas redes neurais convolucionais, poucos estudos realizados consideraram essas abordagens na tarefa de classificação de séries temporais (FAWAZ et al., 2019). A presente seção tem o intuito de identificar os métodos estudados e propostos para a classificação de séries temporais baseados na aplicação de uma rede neural convolucional, a partir da representação da série como uma imagem.

4.3.1 Transformação da série temporal na forma matricial

A primeira metodologia abordada para representação da série temporal como uma imagem foi discutida por Sadaei et al. (2019). Em seu trabalho, os autores realizaram a representação da série temporal a partir de uma matriz de zeros e uns, obtida por meio da concatenação de vetores, nos quais o valor unitário representa a ocorrência de determinado valor da série temporal com seus valores ordenados. Essa metodologia utilizada, detalhada mais adiante, foi aplicada por Sadaei et al. (2019) para problemas de previsão de carga de curto tempo (*short-term load forecasting*), apresentando resultados promissores e sendo, portanto, aplicada no presente trabalho, porém para a tarefa de classificação de séries temporais.

O processo de obtenção da matriz responsável pela geração da imagem pode ser realizado pelo seguinte procedimento, descrito por Sadaei et al. (2019):

1. A partir de uma dada série temporal, ou de uma subsequência da série, calcula-se a lista ordenada da sequência analisada. Considera-se, por exemplo, um fragmento de uma dada série temporal:

$$x_i = \begin{bmatrix} -0,3769 & 1,2249 & 0,3439 & 0,3284 \end{bmatrix}$$

Sua correspondente lista ordenada pode ser obtida:

$$sx_i = \begin{bmatrix} -0,3769 & 0,3284 & 0,3439 & 1,2249 \end{bmatrix}$$

2. Encontrar o k-ésimo elemento da lista (x_i) , onde $1 \leq k \leq n$, ou seja, l_k e procurar pela posição do termo l_k na lista ordenada (sx_i) , denominado p ;
3. Definir o $vetor_k$, de tamanho n , de forma que o p-ésimo valor do vetor receberá valor 1 e o restante dos elementos, valor 0. Com o intuito de visualizar melhor a imagem final formada e sua relação com a série temporal original, os vetores formados têm sua inicialização de baixo para cima:

$$vetor_1 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \quad vetor_2 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad vetor_3 = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} \quad vetor_4 = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

4. Realizar a concatenação vertical dos vetores a fim de se obter a $matriz_{i,j}$:

$$matriz_i = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

5. Uma vez definida a matriz da sequência a ser representada, constrói-se a imagem a partir da matriz encontrada. A imagem obtida será uma representação em preto e branco da matriz. O exemplo definido acima gerará a imagem apresentada na Figura 10.

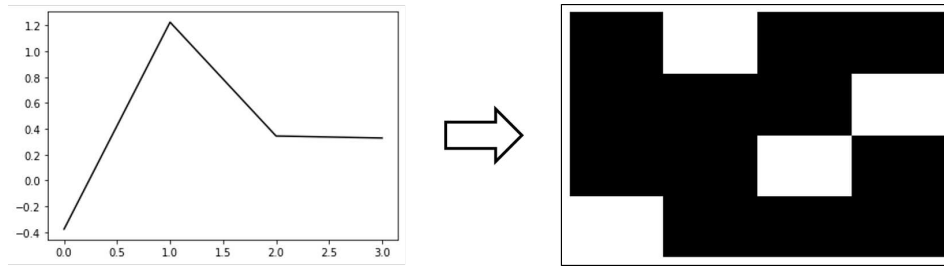


Figura 10 – Exemplo da transformação de uma série temporal em matriz e representação da matriz como uma imagem para ser classificada por uma CNN.

Esse processo é, portanto, realizado em todas as instâncias das séries temporais de cada base de dados. Para maior detalhamento, o Algoritmo 1 apresenta o procedimento para geração da imagem através do método matricial, utilizando os dados numéricos da série temporal, nomeado de NMotif.

As simulações realizadas contemplam a imagem gerada para a classificação através da série completa. Os resultados obtidos serão apresentados mais adiante no capítulo 6, e comparados com os outros métodos propostos, bem como *benchmarkings* encontrados na literatura.

4.3.2 Representação da série temporal em coordenadas polares

O segundo método proposto utiliza como base uma abordagem discutida anteriormente por Keogh et al. (2006), na qual os autores, com o intuito de classificar uma

Algoritmo 1 Geração de imagens NMotif

Entrada: Caminho do conjunto de dados com séries temporais numéricas

Saída: Imagens NMotif para cada classe

```

1: para cada serieTemporalNumerica em caminhoDoConjuntoDeDados faça
2:   Criar uma variável rotuloSerieTemporal
3:   Definir rotuloSerieTemporal como o valor do rótulo da serieTemporalNumerica
4:   Criar uma variável matrizOrdenada
5:   Copiar serieTemporalNumerica para matrizOrdenada
6:   Ordenar matrizOrdenada em ordem crescente
7:   Inicializar um array 2D colunaIndice de tamanho serieTemporalNumerica.size() preenchido
   com 255
8:   para cada elemento i em matrizOrdenada faça
9:     para cada j em serieTemporalNumerica faça
10:      se matrizOrdenada[i] = serieTemporalNumerica[j] então
11:        colunaIndice[i][j] ← 0
12:      fim se
13:    fim para
14:  fim para
15:  Criar uma imagem em escala de cinza imgCinza a partir de colunaIndice
16:  Construir o caminho de saída de acordo com a classe da série temporal
17:  Salvar a imagem no caminho construído
18: fim para

```

determinada imagem, realizam sua transformação em uma série temporal. Esse procedimento foi também aplicado por [Ye e Keogh \(2011\)](#), onde os autores transformam uma imagem em uma série temporal para, posteriormente encontrar as subsequências que caracterizam a série analisada, denominadas *shapelets*. A Figura 11 ilustra o procedimento apresentado por [Keogh et al. \(2006\)](#), no qual, para a geração da série temporal, a partir de uma imagem, as bordas dessa imagem são evidenciadas e a distância de cada ponto até o centro da imagem é tratado como um valor da série temporal, enquanto o eixo do tempo parte de um ponto pré determinado.

Com os bons resultados apresentados pelas redes neurais convolucionais no campo da visão computacional, existe também a alternativa de realizar a classificação da imagem diretamente. Tendo em vista a possibilidade de aplicação de uma CNN para a tarefa de classificação, bem como o procedimento proposto por [Keogh et al. \(2006\)](#), pode-se, portanto, a partir de uma série temporal pré-definida, realizar o processo inverso, ao considerar cada valor da série como uma distância até um determinado centro. Para realizar esse procedimento, o objetivo seria, basicamente, representar a série temporal em coordenadas polares, sendo o raio definido por cada valor da série e o ângulo determinado por cada intervalo de tempo.

Portanto, o método proposto, baseado na abordagem de representação de uma série temporal como uma imagem em coordenadas polares possui a seguinte metodologia

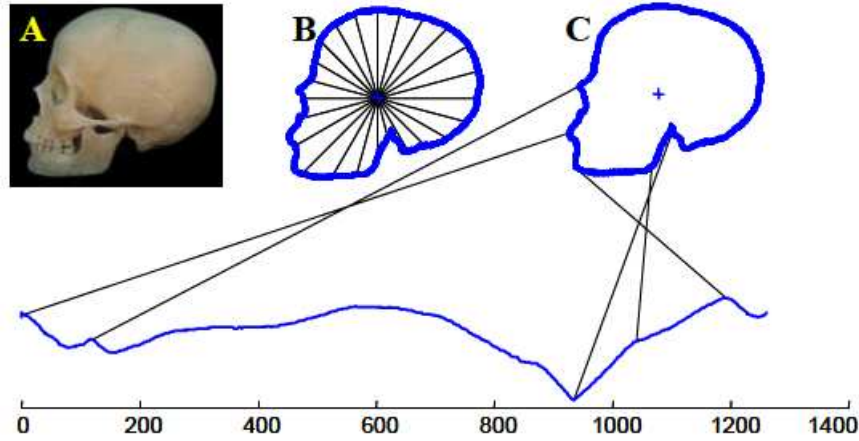


Figura 11 – Conversão da forma de uma imagem em série temporal. A) Uma imagem de um crânio humano. B) A distância de cada ponto da forma até o centro é medida e tratada como eixo Y de uma série temporal de tamanho n (C).

Fonte: [Keogh et al. \(2006\)](#)

para construção da imagem: seja uma série temporal $X = (x_1, \dots, x_n)$, o raio r_i de cada ponto e o ângulo θ_i podem ser definidos como:

$$\forall i \in 1, \dots, n, r_i = x_i \quad (4.9)$$

$$\forall i \in 1, \dots, n, \theta_i = i \cdot \frac{2 \cdot \pi}{n} \quad (4.10)$$

Tomando-se como exemplo a sequência utilizada na seção anterior:

$$x_i = \begin{bmatrix} -0,3769 & 1,2249 & 0,3439 & 0,3284 \end{bmatrix}$$

a imagem formada pela sequência, além de sua representação em coordenadas polares, pode ser ilustrada na Figura 12.

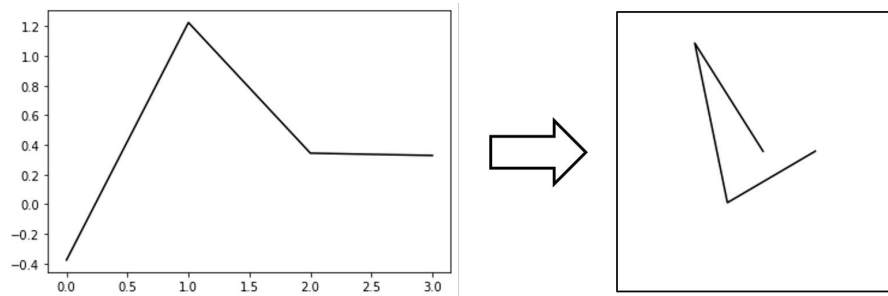


Figura 12 – Exemplo da representação de uma série temporal em coordenadas polares como imagem para ser classificada por uma CNN.

Para um maior detalhamento, o Algoritmo 2 apresenta o pseudocódigo para a geração das imagens através da representação em coordenadas polares.

Algoritmo 2 Geração de imagens polares

Entrada: Caminho do conjunto de dados contendo séries temporais numéricas

Saída: Imagens polares geradas para cada classe

```

1: para cada serieTemporalNumerica em caminhoDoConjuntoDeDados faça
2:   para cada i em serieTemporalNumerica.size() faça
3:     criar variável r representando o raio do gráfico em coordenadas polares
4:      $r \leftarrow \text{serieTemporalNumerica}[i]$ 
5:   fim para
6:   Calcular os ângulos theta uniformemente espaçados entre 0 e  $2\pi$ , com tamanho igual a
     serieTemporalNumerica.size()
7:   Gerar o ambiente de imagem em coordenadas polares
8:   Traçar o gráfico em coordenadas polares com r como raio e theta como ângulos
9:   Construir o caminho de saída de acordo com a classe da série temporal
10:  Salvar a imagem no caminho construído
11: fim para

```

Uma vez definidas as metodologias de pré-processamento da série temporal para a geração de imagens e posterior processo de classificação pela CNN, deve-se, então, decidir as bases de dados as quais serão expostas ao processo de classificação. Para tal, foi utilizado o banco de dados de séries temporais da UCR (*UCR Time Series Classification Archive*) (DAU et al., 2018), um repositório de diversos tipos de séries temporais, criado para auxiliar nas pesquisas sobre o assunto. Os resultados obtidos através dos métodos propostos no presente capítulo estão mais detalhados no capítulo 6.

Um ponto interessante sobre a classificação de séries temporais baseada em imagens, através de uma CNN, é que essa técnica pode ser aplicada tanto na série temporal com seus dados brutos, quanto na série temporal representada em um novo espaço de características. Para isso, foram consideradas, também, para a geração das imagens as séries temporais fuzzificadas, mais especificamente, a partir dos valores inteiros de cada grupo fuzzy obtido, como discutido na subseção anterior, sendo, portanto, discretizada. Essa nova série temporal, representada por grupos fuzzy, pode, então, ser transformada em uma imagem seguindo-se as abordagens apresentadas na seção 4.3.1. O Algoritmo 3 apresenta o procedimento de geração e armazenamento de imagens das séries temporais fuzzificadas, nomeadas, no trabalho, de FTSMotif. Essas imagens serão, também, utilizadas como entrada para uma rede neural convolucional para a realização da tarefa de classificação.

Por fim, a Figura 13 apresenta um exemplo de todas as abordagens de representação da série temporal como imagens aplicada a uma dada base de dados. Os resultados obtidos com essa abordagem serão comparados com as demais metodologias e apresentados no capítulo 6.

Algoritmo 3 Geração de imagens FTSMotif

Entrada: Caminho do conjunto de dados com séries temporais numéricas

Saída: Imagens FTSMotif para cada classe

```

1: Criar uma variável ngrupos
2: ngrupos  $\leftarrow$  10
3: para cada serieTemporalNumerica em caminhoDoConjuntoDeDados faça
4:   Criar uma variável rotuloSerieTemporal
5:   Definir rotuloSerieTemporal como o valor do rótulo da serieTemporalNumerica
6:   Particionar o Universo do Discurso usando o pyFTS
7:   particoes  $\leftarrow$  pyFTS.GridPartitioner(data = serieTemporalNumerica, npart = ngrupos)

8:   Criar um vetor FTS de tamanho serieTemporalNumerica.size()
9:   para cada elemento i em serieTemporalNumerica faça
10:    Calcular o grau de pertinência do elemento em cada conjunto fuzzy (A0, ..., A9) em
        particoes
11:    Definir o grupo fuzzy do elemento como aquele com o maior grau de pertinência
12:    Definir uma variável valorFuzzy
13:    Atribuir o valor inteiro do grupo fuzzy (0, ..., 9) do elemento à variável valorFuzzy
14:    FTS[i]  $\leftarrow$  valorFuzzy
15:  fim para
16:  Criar um vetor FTSOrdenado
17:  Copiar FTS para FTSOrdenado
18:  Ordenar FTSOrdenado em ordem crescente
19:  Inicializar um array 2D colunaIndice de tamanho FTS.size() preenchido com 255
20:  para cada elemento i em FTSOrdenado faça
21:    para cada elemento j em FTS faça
22:      se FTSOrdenado[i] = FTS[j] então
23:        colunaIndice[i][j]  $\leftarrow$  0
24:      fim se
25:    fim para
26:  fim para
27:  Criar uma imagem em escala de cinza imgCinza a partir de colunaIndice
28:  Construir o caminho de saída de acordo com a classe da série temporal
29:  Salvar a imagem no caminho construído
30: fim para

```

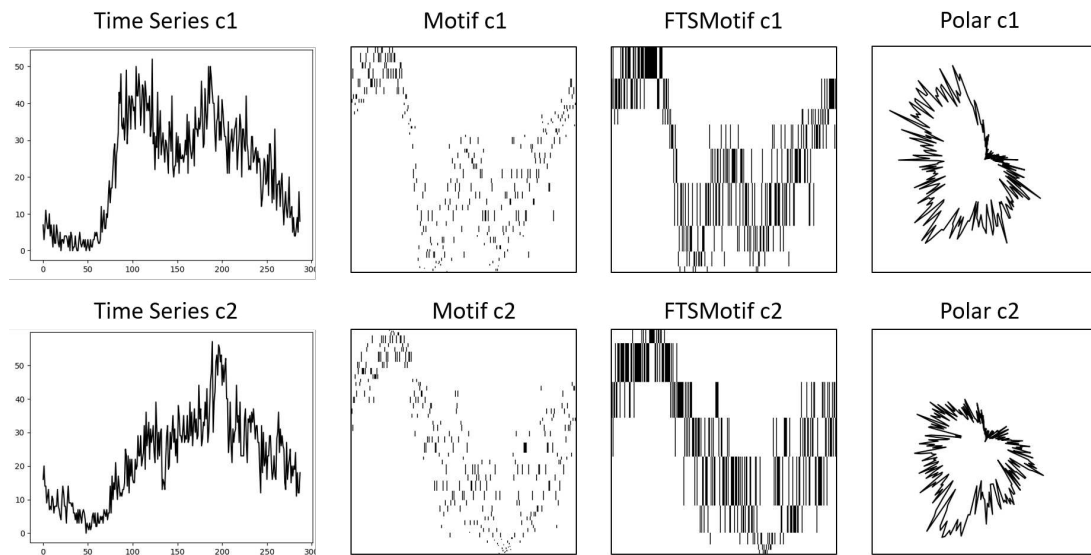


Figura 13 – Exemplo dos métodos de representação de imagem propostos para uma série temporal com duas classes. A primeira coluna exibe a série temporal numérica para cada classe, a segunda coluna apresenta os Motifs Numéricos (NMotif), a terceira coluna mostra os Motifs Fuzzy (FTSMotif) e a última coluna ilustra a representação em coordenadas polares.

5 CLASSIFICAÇÃO ATRAVÉS DE APRENDIZADO FEDERADO

Até então, os métodos utilizados consideram a possibilidade de acesso a todos os dados para o processo de treinamento e teste. Isso ocorre, pois, em métodos de aprendizado de máquina tradicionais os dados existentes são utilizados para treinar um modelo global, o qual é utilizado para realizar a tarefa à qual ele foi projetado, seja previsão, classificação, ou outro objetivo necessário. No âmbito cliente-servidor, os dados de um ou vários clientes são enviados a um servidor central, sendo esse servidor central responsável por treinar o modelo com esses dados. Após o treinamento, o modelo é enviado novamente ao cliente para realizar o processo de inferência (BANABILAH et al., 2022). Esses métodos caracterizam-se por dois principais problemas: (1) preocupações com privacidade de dados devido à transferência de dados para o servidor central; (2) a incapacidade de capturar padrões personalizados de clientes individuais.

Para lidar com essas limitações, uma abordagem descentralizada de aprendizado de máquina, conhecida como Aprendizado Federado (*Federated Learning* - FL) foi introduzida em 2016 (KONECNY et al., 2016; MCMAHAN et al., 2016). O Aprendizado Federado apresenta uma abordagem de aprendizado híbrido utilizando as vantagens de um modelo global centralizado e um conjunto de modelos locais descentralizados (BANABILAH et al., 2022). Para se entender melhor sobre essa abordagem, uma breve contextualização pode ser encontrada adiante no presente capítulo.

5.1 *Federated Learning* (FL)

O Aprendizado Federado é uma técnica de aprendizado de máquina descentralizada implementada através de múltiplos usuários (ou clientes) (GUENDOUI et al., 2023; QI et al., 2023). Essa abordagem envolve N usuários, representados como $\{U_1, U_2, \dots, U_N\}$. Cada usuário possui um banco de dados local $\{D_1, D_2, \dots, D_N\}$ que permanece com seu usuário, sem o compartilhamento entre os outros clientes ou meios externos. Portanto, FL é capaz de resolver problemas relacionados à privacidade e segurança de dados, principalmente quando dados heterogêneos são obtidos de dispositivos distribuídos, o que contribui para o processo de aprendizado (GUENDOUI et al., 2023; BONAOWITZ et al., 2021).

O processo de treinamento, realizado em rodadas, pode ser dividido em três etapas, e ocorre da seguinte maneira: 1) O servidor distribui o modelo inicial para cada cliente; 2) Cada cliente (U_i) pode treinar seus próprios modelos (W_i) localmente, utilizando seus dados (D_i), sem os compartilhar; 3) O servidor agrega os modelos locais $\{W_1, \dots, W_n\}$ no modelo global W^g , o qual, em seguida, é atualizado em cada cliente local (ZHANG et al., 2021; BANABILAH et al., 2022). A figura 14 apresenta a estrutura genérica do

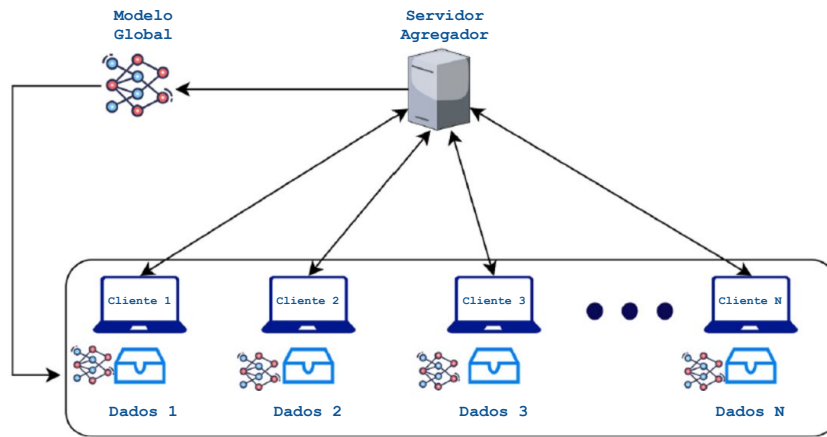


Figura 14 – Desenho esquemático do processo de Aprendizado Federado

Aprendizado Federado.

Diversos trabalhos de revisão vêm sendo apresentados na literatura com o intuito de disponibilizar mais informação sobre a técnica de Aprendizado Federado, bem como suas categorias, graus de liberdade, tipos de topologias, desafios, algoritmos de agregação, aplicações e *frameworks* (GUENDOUZI et al., 2023; QI et al., 2023; CHELLAPANDI et al., 2023; RANI et al., 2023; ZHANG et al., 2021; NGUYEN et al., 2021). O foco da presente pesquisa, porém, é a aplicação do Aprendizado Federado na classificação de séries temporais. Sendo assim, a seguir, alguns dos estudos encontrados referentes a esse tema serão introduzidos.

Para exemplificar, alguns desses estudos, pode-se citar o trabalho de Paragliola (2022), no qual os autores combinaram uma técnica de Aprendizado Profundo na área da saúde, com o intuito de classificar séries temporais de Eletro Cardiogramas (ECG), obtendo 91% de acurácia. FedTSC, outra abordagem baseada no Aprendizado Federado, introduzida por Liang e Wang (2022), efetivamente combina segurança, interpretabilidade, acurácia e eficiência. Em (PARAGLIOLA; RIBINO, 2024), um modelo de classificação através de uma rede neural federada foi aplicada para classificar bases de dados de séries temporais de ECG, com acurácia de 91%. Esse modelo opera assumindo dados distribuídos de maneira não estacionária e não uniformes, denominada distribuição heterogênea. Em 2023, Younis et al. (2023) introduzem uma técnica federada com boa interpretabilidade, denominada FLAMES2Graph para realizar classificação de séries temporais multivariadas. Nesse trabalho, quatro clientes treinam redes CNN para extrair períodos altamente ativados (Highly Activated Periods - MHAPs) dos dados, a fim de criar um gráfico de evolução desses MHAPS para cada round n . Esse grafos são, então, agregados pelo servidor em um grafo global, e, posteriormente, distribuídos de volta aos clientes para o próximo round de treinamento, $n+1$. Zhiyu e Wang (2023), em seu trabalho, propõe um novo método de classificação de séries temporais, baseado no FL, utilizando uma técnica de transformação Shapelet centralizada para aumentar acurácia, interpretabilidade e flexibilidade. Xing et

al. (2022a), por sua vez, introduziu um sistema federado de aprendizado por destilação (Federated Distillation Learning System - EFDLS) para classificação multitarefa de séries temporais. EFDLS inclui um servidor central e múltiplos usuários móveis, de tal maneira que cada usuário realiza diferentes tarefas de classificação de séries temporais. Esse sistema contém dois novos componentes: um sistema aluno professor baseado em características (feature-based student-teacher - FBST) e um método de correspondência de pesos com base em distância (distance-based weights matching - DBWM). EFDLS foi comparado com 40 bases de dados escolhidas da base da UCR2018 (DAU et al., 2018), obtendo a maior acurácia média (70.14%) nas bases de dados avaliadas. Um algoritmo de Federated Learning (FL) baseado em otimização (FedAdam) foi proposto por Kumar et al. (2023) para classificar falhas em séries temporais industriais. A base de dados utilizado nesta pesquisa é conhecido como PRONTO, a qual abrange ambos os casos: distribuídos de forma independente e idêntica (IID) e não independente e não idêntica (non-IID). Um método de FL horizontal hierárquico heterogêneo (Hierarchical Heterogeneous Horizontal FL - HHHFL), baseado em redes neurais convolucionais (CNN), foi proposto por Gao et al. (2019) para a classificação de um conjunto de dados reais heterogêneo de eletroencefalografia (EEG).

Percebe-se, após as pesquisas realizadas no presente trabalho, que poucos estudos exploraram a classificação de séries temporais (TSC) utilizando Federated Learning (FL). Além disso, nenhum deles investigou a fusão de representações de séries temporais baseadas em imagens com redes neurais convolucionais (CNNs) em um cenário de FL.

5.2 Técnicas propostas baseadas em Aprendizado Federado

A última, e principal contribuição desse trabalho é o desenvolvimento de um método federado (distribuído) para classificar de séries temporais. Como no método de aprendizado federado os dados não são compartilhados, mas sim os parâmetros de modelos, esse método foi aplicado apenas às técnicas baseadas em imagens, propostas anteriormente. Isso se dá, principalmente devido ao fato de os métodos baseados em distância necessitarem de acesso a todos os dados para sua aplicação e o cálculo das distâncias ou custos. Portanto, o método proposto utiliza as três técnicas de representação baseadas em imagens discutidas, integradas com uma rede neural convolucional, e emprega o Federated Learning (FL) para viabilizar uma abordagem descentralizada. De uma forma mais clara, o cerne do método proposto, discutido no presente capítulo, é baseado na metodologia centralizada proposta, discutida no capítulo 4.

Duas abordagens baseadas em aprendizado federado foram propostas. A primeira é uma extensão da técnica centralizada, de tal forma que, os dados de treinamento de cada base de dados foram divididos entre três clientes. O treinamento era realizado de

maneira local em cada um desses clientes, tendo seus parâmetros compartilhados com o servidor. O modelo global agrega esses parâmetros e realiza a classificação centralizada com os dados de teste. Nessa abordagem, cada base de dados é classificada separadamente, sendo cada experimento federado utilizado para a classificação de uma base de dados.

A segunda abordagem, conhecida como multitarefa, é uma abordagem de classificação que utiliza um único modelo para classificar diferentes tipos de dados [Harutyunyan et al. \(2019\)](#). Nela, cada base de dados representa um cliente, e o modelo global é gerado a partir dos parâmetros locais de cada cliente. Esse modelo é, então, agregado pelo servidor, o qual realiza a classificação dos dados de teste. Nessa abordagem, o modelo global realiza, de uma única vez, a classificação de todas as bases de dados. Ou seja, a classificação das bases de dados é realizada através de um único experimento federado. As seções a seguir detalham cada uma das abordagens discutidas acima.

5.2.1 Abordagem baseada no particionamento das bases de dados

Essa abordagem é uma extensão do método centralizado, sendo cada base de dados particionada entre clientes. Como o modelo centralizado foi discutido e apresentado anteriormente, a presente seção destaca a aplicação do aprendizado federado na classificação de séries temporais através de uma CNN, utilizando as representações baseadas em imagens discutidas no capítulo anterior. As figuras 15-A e 15-B apresentam um resumo das duas abordagens desenvolvidas, centralizada e descentralizada, respectivamente. Apesar da abordagem federada utilizar métodos centralizados, ao combinar técnicas de representação de séries temporais em imagens e CNNs, elas se diferem no processo de treinamento. Os modelos federados transferem o processo computacional para os dados, ao invés de transferir os dados para o processo computacional. Ou seja, cada nó local treina seu próprio modelo e envia as atualizações para serem agregadas no servidor. Os passos do método proposto estão detalhados a seguir:

Passo 1: Particionamento dos dados para treinamento local

O banco de dados UCR, detalhada no capítulo de resultados, é utilizada nesse estudo e disponibiliza dados de treinamento e teste previamente separados para cada base. Antes do processo de treinamento, os dados separados para esse fim são divididos em clientes para permitir a aplicação do aprendizado federado. De maneira específica, os dados de treinamento são divididos entre três clientes para treinamento local. Esse particionamento foi realizado, intencionalmente, de maneira não balanceada, sendo a divisão dos dados de cada cliente seguindo a distribuição de Dirichlet ([NG; TIAN; TANG, 2011](#)), considerando $\alpha = 0.5$.

Como ilustrado na figura 15-B, supondo que existam um total de N séries temporais utilizadas em uma tarefa de classificação binária, divididas entre duas classes, sendo que

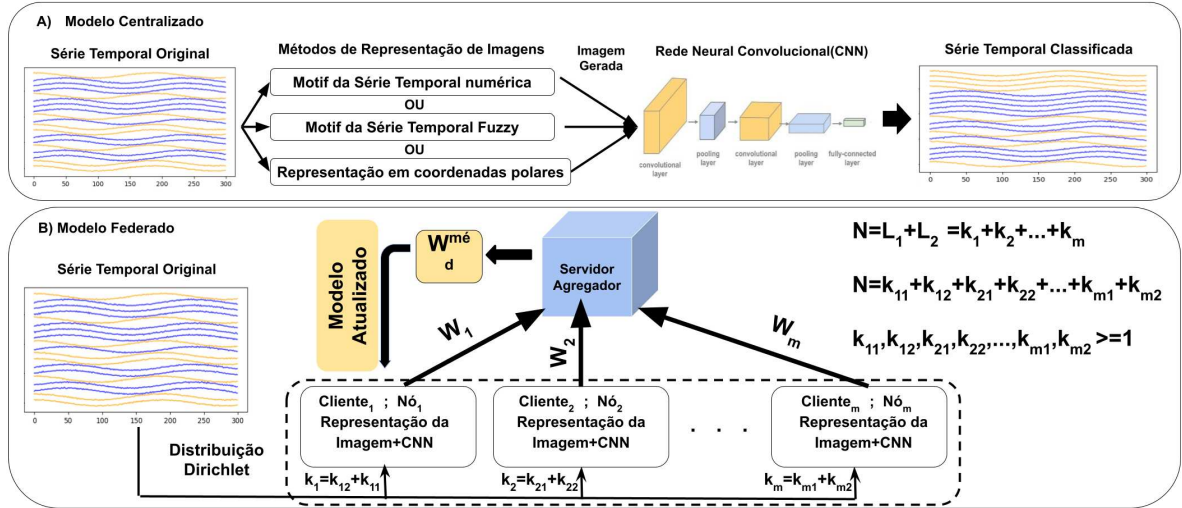


Figura 15 – Representação simples da abordagem proposta. (A) Modelo centralizado: A série temporal original é transformada em uma imagem usando um dos métodos de representação de imagem propostos e, em seguida, alimentada em uma CNN para classificação de séries temporais. (B) Modelo TSC federado: Esta abordagem estende o modelo centralizado. A distribuição de Dirichlet particiona a série temporal entre um número predefinido de clientes com base em uma condição especificada. Cada cliente treina uma CNN com seus dados atribuídos, após esse passo, os pesos do modelo são compartilhados com o servidor para agregação. Os pesos atualizados são, então, compartilhados de volta aos clientes, e esse processo é repetido por um número definido de rodadas.

L_1 representa o número de séries temporais da Classe 1 e L_2 o número de séries da Classe 2, então, $N = L_1 + L_2$. Para um modelo federado com m clientes, as séries temporais são distribuídas entre os clientes, de tal forma que cada cliente i tem k_i séries temporais, satisfazendo $N = k_1 + k_2 + \dots + k_m$. Para ter certeza que cada cliente tenha dados de ambas as classes, especifica-se no modelo que cada k_i é composta de k_{i1} series da Classe 1 e k_{i2} da Classe 2, resultando na expressão $N = k_{11} + k_{12} + k_{21} + k_{22} + \dots + k_{m1} + k_{m2}$. A condição $k_{11}, k_{12}, k_{21}, k_{22}, \dots, k_{m1}, k_{m2} \geq 1$ garante que cada cliente possua pelo menos uma série temporal de cada classe, promovendo uma representação balanceada das classes entre os clientes e favorecendo um aprendizado federado eficaz. O Algoritmo 4, ilustra o processo de particionamento dos dados de treinamento com mais detalhes.

Passo 2: Treinamento local no cliente

Após o particionamento dos dados de treinamento, cada cliente possui seus próprios dados locais, os quais serão utilizados no processo de treinamento. Esses dados foram organizados em diretórios, contendo as imagens geradas segundo o que foi explicitado no capítulo 4. A fim de tornar o processo menos oneroso, para a classificação utilizando o aprendizado federado, foram utilizadas as imagens que geraram a maior acurácia na classificação centralizada. Os diretórios contendo as imagens particionadas para cada cliente foram alocadas segundo o número do cliente ao qual elas pertencem. Para cada cliente, as imagens estão organizadas em diretórios baseados nos rótulos das séries temporais, sendo

Algoritmo 4 Particionamento de Imagens para FL (Aprendizado Federado)

Entrada: Caminho de uma das representações de imagem geradas. As pastas de imagens são consideradas divididas de acordo com a classe da imagem.

Saída: Imagens particionadas entre três clientes, seguindo a Distribuição de Dirichlet, e considerando ao menos uma amostra de cada classe para cada cliente.

```

1: Definir a pasta raiz com a representação de imagem a ser analisada
2: Criar uma pasta AprendizadoFederado
3: Criar três pastas para cada cliente dentro de AprendizadoFederado
4: Definir uma variável criterioClasses a ser satisfeita se todos os clientes tiverem pelo menos
   uma imagem de cada classe
5: criterioClasses ← false
6: enquanto criterioClasses = false faça
7:   Particionar as imagens dentro de cada pasta de classe em pastaRaiz entre os 3 clientes,
   seguindo a Distribuição de Dirichlet
8:   se cada cliente possuir pelo menos uma amostra de cada classe então
9:     para cada cliente em particionamentoClientes faça
10:      para cada classe em cliente faça
11:        para cada imagem em classe faça
12:          Salvar imagem na pasta correspondente dentro da pasta AprendizadoFederado
13:        fim para
14:      fim para
15:    fim para
16:    criterioClasses ← true
17:   fim se
18: fim enquanto

```

cada uma delas armazenadas nos diretórios correspondentes às suas respectivas classes. Essa organização é realizada para garantir consistência na importação dos dados.

Passo 3: Atualização do Modelo Global e Classificação dos dados de Teste

Após finalizar o treinamento local, realizado por um determinado número de rodadas (*rounds*), cada cliente compartilha as atualizações de seus modelos com o servidor, o qual agrega esses parâmetros para atualizar o modelo global. O servidor, então, distribui os parâmetros globais atualizados com cada cliente para a próxima rodada de treinamento. Uma vez que os parâmetros foram agregados, o servidor avalia o desempenho do modelo global atualizado, através dos dados de teste, os quais não são particionados entre clientes como os dados de treinamento. O intuito de não particionar os dados de teste é que seja possível comparar os resultados obtidos com a abordagem federada e os resultados anteriores. Nesse estudo, o modelo global é obtido pela média ponderada dos parâmetros dos clientes, através do algoritmo FedAvg (QI et al., 2024).

Passo 4: Iterações por Rodadas e Monitoramento da Convergência

O processo discutido até o terceiro passo é repetido por, no mínimo, quinze rodadas (*rounds*) para cada base de dados utilizadas no trabalho. O número de rodadas pode aumentar, baseado na convergência do modelo. A acurácia dos últimos três rounds

é monitorada com uma tolerância de 1% para garantir estabilidade. Se essa condição não é satisfeita, rodadas adicionais são realizadas até que a convergência seja atingida. A Figura 16 apresenta a convergência do método federado proposto em 20 bases de dados. No entanto, observa-se que o método não converge completamente para o conjunto de dados Ham, indicando a necessidade de rodadas adicionais de treinamento para atingir estabilidade e desempenho ideal nesse caso. Assim, os experimentos foram conduzidos com 30 rodadas nessa base, a fim de satisfazer o critério de convergência com acurácia igual a 75%.

Para uma melhor compreensão do procedimento do modelo, o Algoritmo 5 apresenta, de forma geral, as etapas para a classificação da metodologia proposta, utilizando aprendizado federado.

Algoritmo 5 Procedimento de classificação com FL (Aprendizado Federado)

Entrada: Caminho para as imagens de treino particionadas entre 3 clientes e imagens de teste para classificação centralizada.

Saída: Acurácia no conjunto de teste centralizado após o treinamento descentralizado usando os 3 clientes definidos para a representação de imagem escolhida.

- 1: Definir a pasta de treino contendo 3 clientes para treinamento descentralizado com FL
 - 2: Definir a pasta com o conjunto de teste para classificação centralizada
 - 3: Carregar as imagens de treino para cada cliente
 - 4: Carregar as imagens de teste para classificação centralizada no servidor
 - 5: Iniciar o Aprendizado Federado
 - 6: Criar a variável *rodadas*
 - 7: Definir o número de rodadas do FL e atribuí-lo a *rodadas*
 - 8: Iniciar cada cliente e atribuir suas imagens correspondentes
 - 9: **para cada** *rodada* em *rodadas* **faça**
 - 10: **para cada** *cliente* em *clientes* **faça**
 - 11: Carregar imagens do cliente
 - 12: Iniciar o procedimento de treinamento com CNN
 - 13: Obter os parâmetros do modelo após o processo de treinamento
 - 14: Enviar modelo ao servidor
 - 15: **fim para**
 - 16: Servidor coleta os modelos de cada cliente
 - 17: Servidor agrega os parâmetros dos modelos
 - 18: Carregar imagens do conjunto de teste
 - 19: Classificação centralizada no servidor com o modelo agregado
 - 20: Obter a acurácia da classificação e salvar para o relatório final
 - 21: **fim para**
 - 22: Gerar o relatório final com todas as acurácias de classificação
 - 23: Salvar o relatório de acurácia
-

5.2.2 Classificação Multitarefa

O aprendizado multitarefa, cujas raízes estão na área da saúde e previsão clínica (CARUANA; BALUJA; MITCHELL, 1995), vem sendo bastante estudado em muitos trabalhos recentes (HUANG; LV; ZHANG, 2025; ZHANG; YANG, 2021). No contexto

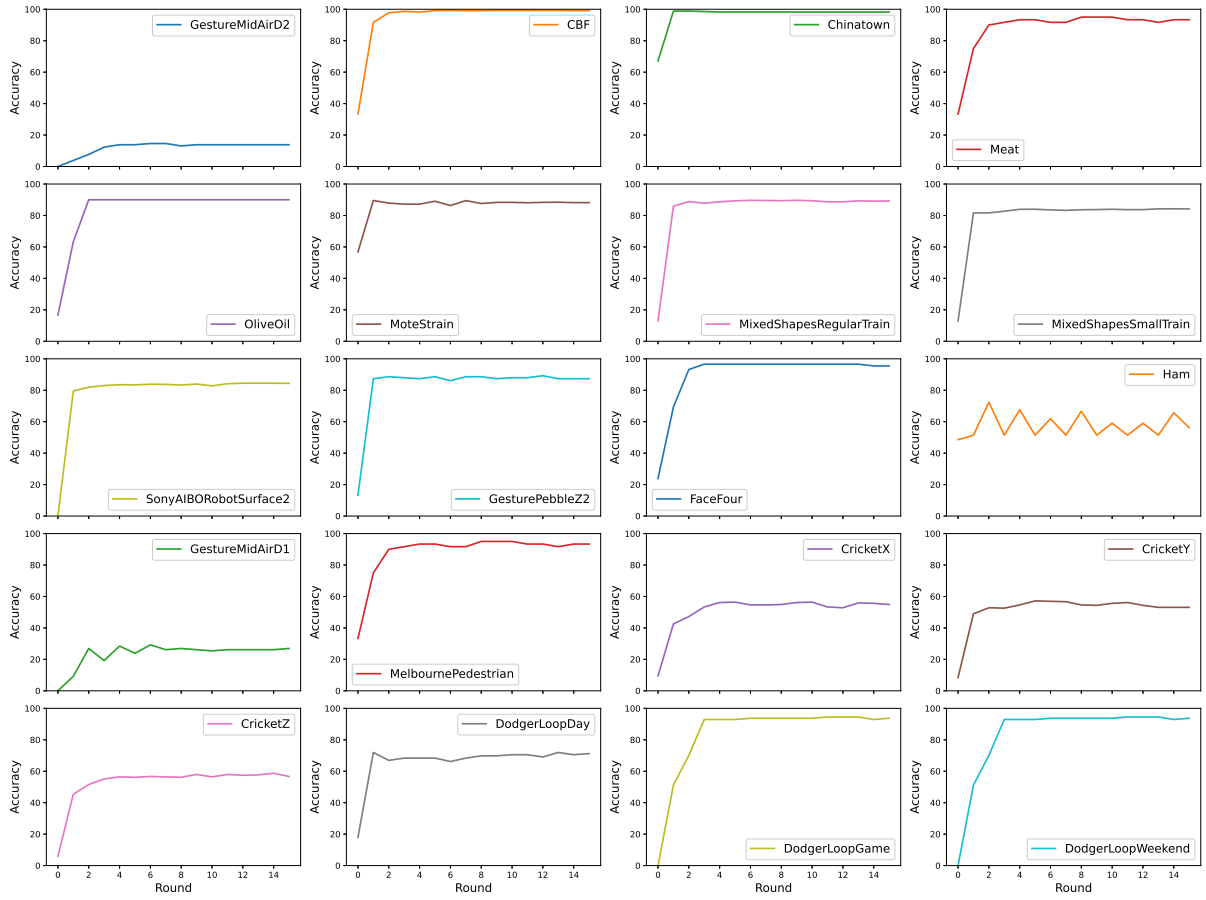


Figura 16 – Monitoramento da convergência do método de Aprendizado Federado proposto ao longo de várias bases de dados após 15 rodadas. Com exceção da base Ham que foi posteriormente simulada por 30 rodadas, todas as demais bases convergiram nas 15 rodadas.

de aprendizado federado, muitos trabalhos também vêm utilizando esse paradigma para realizar uma classificação generalizada, mantendo privacidade de dados (XING et al., 2022b; WU et al., 2024; SHAHEEN et al., 2022)

Diferentemente do método anterior, na classificação multitarefa não há o particionamento dos dados de treinamento para se simular cada cliente. No presente caso, cada base de dados é considerada como um cliente diferente, sendo o modelo treinado localmente com cada um deles, obtendo seus parâmetros locais. Esses parâmetros, após o treinamento local, são compartilhados com o servidor, responsável pela agregação dos parâmetros para a obtenção do modelo global. Por fim, o modelo global obtido é utilizado para classificar todas as bases de dados de uma só vez, sendo cada uma delas um cliente. Esse tipo de classificação foi possível, principalmente, pela possibilidade de se padronizar o tamanho da entrada de dados, proporcionado pela representação da série temporal como imagem.

O processo geral de classificação com essa abordagem é semelhante ao realizado anteriormente, a diferença é a não necessidade da fase de particionamento das bases de

dados. Os passos seguidos para a classificação com essa abordagem estão dispostos a seguir:

Passo 1: Organização das imagens de cada base de dados

O primeiro passo para a classificação multitarefa é a organização das imagens obtidas para cada base de dados em clientes, a fim de realizar o treinamento local em cada uma delas. Assim como na abordagem anterior, foram utilizadas para a classificação multitarefa as imagens cuja representação apresentou melhor resultado de acurácia na classificação centralizada. Elas foram dispostas dentro de seu respectivo diretório, com as imagens de treinamento e teste em diretórios separados dentro do diretório da base de dados. Cada base possui seu próprio diretório, totalizando 44 diretórios clientes.

Passo 2: Treinamento local nos dados de treinamento

De maneira semelhante à abordagem anterior, durante o processo de aprendizado federado, os dados de treinamento das bases são utilizados para treinamento local e os dados de teste para classificação final. As imagens correspondentes às instâncias de treinamento se localizam dentro do diretório da respectiva base de dados.

Passo 3: Treinamento local e atualização do modelo global

Assim como na abordagem anterior, o modelo global é atualizado a cada rodada do aprendizado federado, agregando os parâmetros de cada modelo local. Para que esse modelo fosse possível, algumas considerações foram necessárias:

1. O tamanho das imagens de todas as séries temporais foi padronizado a fim de se padronizar o tamanho da entrada dos dados na CNN, bem como os parâmetros de entrada para o modelo global;
2. Como as diversas bases de dados possuem diferentes números de classes, a arquitetura da CNN foi desacoplada das camadas densas. Assim, o modelo é treinado localmente em cada base de dados, mas os parâmetros a serem agregados são considerados apenas para as camadas de convolução e pooling da CNN;
3. Devido ao grande número de clientes e imagens para cada cliente, é indicado definir uma fração mínima de clientes a serem utilizados para o treinamento. Essa fração foi definida como 10%.

Com essas considerações, foi possível realizar o treinamento local para cada cliente, agregar os parâmetros intermediários da CNN, gerando o modelo global. As camadas densas, responsáveis pela classificação, são agregadas apenas localmente.

Passo 4: Classificação e avaliação nos dados de teste

Uma vez agregados os parâmetros locais de cada cliente no modelo global, e percorrido o número definido de rodadas de aprendizado federado, o processo de avaliação

dos dados de teste pode ser realizado. Nessa etapa, para cada uma das bases, os parâmetros intermediários da CNN são congelados e os pesos das camadas densas e camada de saída são treinados uma única vez pelos dados locais de treinamento do respectivo cliente, em processo de avaliação, a fim de se obter os pesos dessas camadas. Após a obtenção desses pesos e, conseqüentemente, do modelo local por completo, pode-se realizar a classificação dos dados de teste, específicos para aquele cliente. O resultado da acurácia, obtido após a última rodada do aprendizado federado, é utilizado como o resultado de classificação para a respectiva base de dados. O algoritmo desse processo é semelhante ao observado no Algoritmo 5, porém com modificações, principalmente após as rodadas de aprendizado federado. Esse algoritmo pode ser observado no Algoritmo 6.

Algoritmo 6 Procedimento de classificação Multitarefa (Aprendizado Federado)

Entrada: Caminho para as 44 bases de dados, contendo as imagens de treinamento e teste.

Saída: Acurácia no conjunto de teste centralizado após o treinamento descentralizado usando os 44 clientes definidos para a representação de imagem escolhida.

- 1: Definir a pasta contendo 44 os clientes para treinamento descentralizado com FL
 - 2: Para cada cliente, definir a pasta com o conjunto de treinamento e a pasta de teste para classificação centralizada
 - 3: Iniciar o Aprendizado Federado
 - 4: Criar a variável *rodadas*
 - 5: Definir o número de rodadas do FL e atribuí-lo a *rodadas*
 - 6: Iniciar os clientes selecionados e atribuir suas imagens correspondentes
 - 7: **para cada** *rodada* em *rodadas* **faça**
 - 8: **para cada** *cliente* em *clientes* **faça**
 - 9: Carregar imagens do cliente
 - 10: Iniciar o procedimento de treinamento com CNN
 - 11: Obter os parâmetros do modelo após o processo de treinamento
 - 12: Selecionar apenas as camadas convolucionais e pooling para enviar ao modelo global
 - 13: Enviar os parâmetros selecionados ao servidor
 - 14: **fim para**
 - 15: Servidor coleta os modelos de cada cliente
 - 16: Servidor agrega os parâmetros dos modelos
 - 17: Congela os parâmetros do modelo global e cria as camadas totalmente conectadas e a camada de saída para o número de classes do cliente
 - 18: Carregar imagens do conjunto de treinamento e teste para aquele cliente
 - 19: Treinamento único dos parâmetros de saída com os dados de treinamento
 - 20: Classificação centralizada no servidor com o modelo agregado, os parâmetros obtidos para a camada de saída e os dados de teste
 - 21: Obter a acurácia da classificação e salvar para o relatório final
 - 22: **fim para**
 - 23: Gerar o relatório final com todas as acurácias de classificação
 - 24: Salvar o relatório de acurácia
-

6 RESULTADOS

Neste capítulo serão apresentados os resultados obtidos com as estratégias de classificação de séries temporais propostas. Os resultados para cada uma delas serão comparados entre si e, posteriormente, comparados com resultados obtidos com estratégias consideradas como *benchmark*. Os resultados serão apresentados através de três seções, divididas entre algoritmos baseados em medidas de similaridade aplicados a séries temporais fuzzy e algoritmos de classificação baseados em imagens, centralizados e descentralizados. O primeiro passo do capítulo, portanto, será descrever as bases de dados utilizadas para classificação, seguidas pelos resultados obtidos por cada estratégia.

6.1 Bases de dados UCR

Para a realização do processo de classificação, bem como teste das estratégias propostas, foram selecionadas diversas bases de dados do repositório UCR (DAU et al., 2018). O repositório foi introduzido em 2002, e tem sido muito utilizado como um *benchmark* para diversos trabalhos. Atualmente ele possui 128 séries temporais, de tal maneira que cada uma delas está dividida entre dados de treinamento e dados de teste. O presente trabalho tem o intuito de apresentar a análise de 40 bases de dados do repositório, selecionadas de acordo com os trabalhos de Estado da Arte utilizados para comparação. As séries temporais utilizadas na pesquisa são apresentadas na Tabela 1, que define, para cada uma delas, o número de instâncias (treinamento e teste), o número de classes e o número de amostras para cada instância, o qual é o mesmo para as bases de treinamento e teste.

Vale salientar que os bancos de dados utilizados foram selecionados devido à possibilidade de comparação entre todas as abordagens propostas. Para que essa comparação fosse possível, principalmente com as propostas baseadas em FL, foram consideradas bases de dados aplicadas em metodologias de referência. A principal delas pode ser encontrada no trabalho de Xing et al. (2022b). Para cada um dos bancos de dados apresentados na Tabela 1, os algoritmos propostos discutidos nos capítulos 3, 4 e 5 foram aplicados, a fim de serem comparados com algoritmos, considerados *benchmark*, encontrados nas tabelas de resultados apresentados em (DAU et al., 2018), bem como algoritmos considerados como Estado da Arte.

Os algoritmos baseados em imagens, discutidos no capítulo 4, são caracterizados por possuírem princípios estocásticos, uma vez que realizam o treinamento de uma CNN. Sendo assim, tanto o treinamento da rede neural, como os testes realizados com o modelo treinado, foram repetidos trinta vezes, a fim de se conseguir uma base estatística

considerável. Já no caso dos algoritmos baseados em medidas de similaridade, abordados no capítulo 3, como seu resultado é obtido através do método 1 *Nearest Neighbor*, esse pode ser considerado como determinístico, sem necessidade de repetição. Por fim, o algoritmo que utiliza Aprendizado Federado, foi repetido ao longo das rodadas, portanto, as repetições e convergência são intrínsecas ao método utilizado.

Mais detalhes dos resultados obtidos por cada um dos métodos discutidos, serão apresentados adiante, sendo cada um dos métodos propostos abordado em seções separadas. Após a análise e definição dos melhores resultados obtidos com cada um dos métodos propostos, os algoritmos referentes a esses resultados serão, portanto, comparados entre si, entre métodos de *benchmark* e entre métodos considerados Estado da Arte. Para possibilitar a comparação com métodos de referência e de estado da arte aplicados à base UCR, adotou-se a acurácia de classificação como métrica de avaliação. Embora métricas como AUROC e F1-score apresentem maior adequação em cenários com desbalanceamento de classes, sua aplicação em problemas multiclasse é mais complexa e, em muitos casos, pouco padronizada na literatura. Dessa forma, ainda que a acurácia não seja a métrica mais indicada para lidar com classes desbalanceadas, ela permanece como uma das métricas mais empregadas em trabalhos anteriores, o que justifica sua escolha para fins de comparação neste estudo.

6.2 Classificação baseada em medidas de similaridade aplicadas a Séries Temporais Fuzzy

Os resultados discutidos na presente seção são referentes à classificação das séries temporais fuzzy aplicando-se métodos de medidas de similaridade apresentados no capítulo 2. Os algoritmos baseados em medidas de similaridade utilizados para o processo de classificação foram discutidos no capítulo 3, sendo utilizado o algoritmo 1 *Nearest Neighbor* para a obtenção da menor distância e posterior classificação. Os métodos utilizados estão elencados abaixo:

1. Dados da série temporal fuzzyficados e classificados pela Distância Euclideana (FTS- $\mathcal{L}_2$)
2. Dados da série temporal fuzzyficados e classificados pela Distância $\mathcal{W}$ (FTS-DTW)
3. Dados da série temporal fuzzyficados e classificados pela *Longest Common Subsequence* (FTS-LCS)

Como os algoritmos apresentados acima disponibilizam um resultado determinístico, o qual não depende de variáveis estocásticas, sua execução será realizada apenas uma vez, e o valor de acurácia obtido apresentado na Tabela 2. O algoritmo

1NN com cada método de classificação foi executado para as bases citadas na Tabela 1 e uma posterior comparação com os outros métodos propostos, bem como com métodos de referência e Estado da Arte será realizada posteriormente.

Tabela 1 – Bancos de dados utilizados do repositório UCR (DAU et al., 2018)

Série temporal	Instâncias		Classes	Amostras
	Treino	Teste		
ACSF1	100	100	10	1460
AllGestureWiimoteX	300	700	10	Variável
AllGestureWiimoteY	300	700	10	Variável
AllGestureWiimoteZ	300	700	10	Variável
Beef	30	30	5	470
CBF	30	900	3	128
Chinatown	20	343	2	24
Computers	250	250	2	720
CricketX	390	390	12	300
CricketY	390	390	12	300
CricketZ	390	390	12	300
DistalPhalanxOutlineAgeGroup	400	139	3	80
DistalPhalanxTW	400	139	6	80
DodgerLoopDay	78	80	7	288
DodgerLoopGame	20	138	2	288
DodgerLoopWeekend	20	138	2	288
ECG200	100	100	2	96
FaceFour	24	88	4	350
Fish	175	175	7	463
GestureMidAirD1	208	130	26	Variável
GestureMidAirD2	208	130	26	Variável
GestureMidAirD3	208	130	26	Variável
GesturePebbleZ1	132	172	6	Variável
GesturePebbleZ2	142	158	6	Variável
Ham	109	105	2	431
Lightning2	60	61	2	637
Meat	60	60	3	448
MelbournePedestrian	1194	2439	10	24
MixedShapesRegularTrain	500	2425	5	1024
MixedShapesSmallTrain	100	2425	5	1024
MoteStrain	20	1252	2	84
OliveOil	30	30	4	570
PickupGestureWiimoteZ	50	50	10	Variável
PLAID	537	537	11	Variável
SemgHandGenderCh2	300	600	2	1500
ShakeGestureWiimoteZ	50	50	10	Variável
SonyAIBORobotSurface1	20	601	2	70
SonyAIBORobotSurface2	27	953	2	65
StarLightCurves	1000	8236	3	1024
TwoLeadECG	23	1139	2	82

Tabela 2 – Resultados da classificação baseada em medidas de similaridade aplicadas em FTS

Base	FTS- $\mathcal{L}_2$	FTS-DTW(w)	FTS-LCS
ASCF1	0,600	0,590(4)	0,640
AllGestureWiimoteX	0,147	0,529(100)	0,243
AllGestureWiimoteY	0,257	0,601(100)	0,347
AllGestureWiimoteZ	0,214	0,479(100)	0,276
Beef	0,667	0,633(0)	0,600
CBF	0,950	0,998(11)	0,998
Chinatown	0,918	0,892(100)	0,825
Computers	0,596	0,676(100)	0,624
CricketX	0,585	0,605(10)	0,515
CricketY	0,546	0,603(12)	0,538
CricketZ	0,595	0,659(5)	0,600
DistalPhalanxOutlineAgeGroup	0,698	0,712(100)	0,647
DistalPhalanxTW	0,583	0,604(100)	0,568
DodgerLoopDay	0,376	0,442(1)	0,403
DodgerLoopGame	0,795	0,756(1)	0,630
DodgerLoopWeekend	0,913	0,944(1)	0,944
ECG200	0,890	0,900(0)	0,830
FaceFour	0,716	0,784(2)	0,614
Fish	0,806	0,783(4)	0,880
GestureMidAirD1	0,369	0,692(5)	0,377
GestureMidAirD2	0,361	0,631(6)	0,392
GestureMidAirD3	0,208	0,515(100)	0,392
GesturePebbleZ1	0,343	0,645(100)	0,378

Continua na próxima página

Tabela 2 – *Continua na página anterior*

GesturePebbleZ2	0,329	0,658(100)	0,342
Ham	0,342	0,590(0)	0,581
Lightning2	0,770	0,836(6)	0,787
Meat	0,733	0,733(0)	0,733
MelbournePedestrian	0,714	0,698(1)	0,617
MixedShapesRegularTrain	0,906	0,903(4)	0,883
MixedShapesSmallTrain	0,856	0,858(7)	0,826
MoteStrain	0,803	0,822(1)	0,812
0,767 OliveOil	0,567	0,567(0)	0,600
PickupGestureWiimoteZ	0,300	0,500(17)	0,300
PLAID	0,551	0,754(12)	0,538
SemgHandGenderCh2	0,770	0,803(1)	0,790
ShakeGestureWiimoteZ	0,200	0,720(100)	0,420
SonyAIBORobotSurface1	0,729	0,699(0)	0,646
SonyAIBORobotSurface2	0,849	0,839(0)	0,749
StarLightCurves	0,889	0,894(16)	0,906
TwoLeadECG	0,767	0,770(4)	0,804

6.3 Classificação baseada em imagens - Centralizada

Os resultados discutidos na presente seção são referentes à aplicação de uma rede neural convolucional para a classificação de séries temporais, sejam elas com seus dados brutos ou a partir das séries fuzzificadas, transformadas em imagens. Como discutido no capítulo 4, foram propostas duas abordagens para a obtenção da imagem para o processo de treinamento e classificação da CNN: uma metodologia baseada em uma matriz, cujos valores são referentes à ocorrência dos valores da série temporal, e nomeados motifs; uma metodologia baseada na representação da série em coordenadas polares. Ambas as metodologias discutidas geram imagens de treinamento e teste, alocadas em diretórios específicos, de acordo com tipo de dado da instância (treinamento ou teste), sua classe e a transformação sofrida. Para a realização dos experimentos, utilizou-se a série temporal completa como base para a obtenção das imagens a serem classificadas, ou seja, cada imagem obtida representa uma instância da série. A Figura 17 ilustra genericamente a organização de diretórios para que seja possível a realização do treinamento e teste pela CNN durante a classificação centralizada.

A maneira definida para organização dos diretórios está diretamente ligada à aplicação utilizada para realizar a classificação. O algoritmo de classificação foi desenvolvido em linguagem Python, a qual possui diversas bibliotecas capazes de auxiliar, tanto no pré-processamento dos dados, quanto no processo de geração da CNN e classificação em si. Para tal propósito, foi utilizada a aplicação Keras, executada em conjunto com a biblioteca Tensorflow. As imagens utilizadas para a classificação com a transformação das séries temporais foram obtidas de duas principais maneiras:

- as imagens criadas pelo método matricial foram desenvolvidas em linguagem C++, através do ambiente Google Colab, utilizando, principalmente, a biblioteca Opencv
- as imagens criadas pelo método de coordenadas polares foram desenvolvidas em linguagem Python, através do ambiente Google Colab, utilizando, principalmente, a biblioteca Matplotlib

O objetivo determinante para a escolha de cada uma das linguagens de programação foi o compromisso entre complexidade de implementação e tempo de execução.

Uma vez obtidas as imagens, o próximo passo é definir a arquitetura de rede neural convolucional aplicada no processo de classificação. Para os resultados alcançados, discutidos no trabalho, foi selecionada a rede neural convolucional baseada na arquitetura LeNet, umas das arquiteturas mais conhecidas dentre as diversas arquiteturas CNN, proposta por Yann LeCun em 1988 (LECUN et al., 1998) e bastante utilizada para reconhecimento de textos escritos à mão (GÉRON, 2019). Para a construção da rede a ser utilizada, porém, foram realizados diversos testes com configurações diversas, sendo

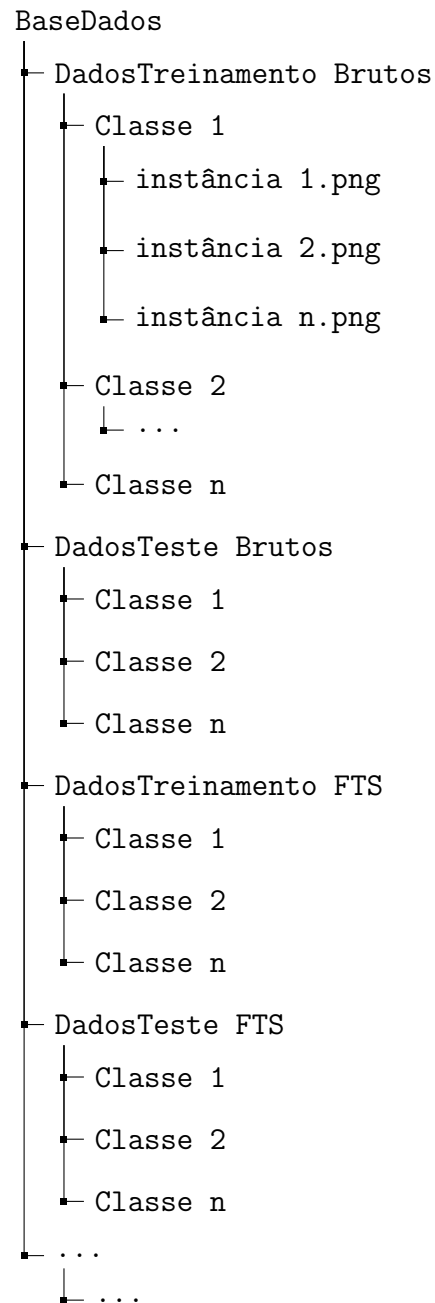


Figura 17 – Exemplo genérico de organização para cada base de dados, contendo o tipo de transformação sofrido pela série temporal, a classe e as instâncias pertencentes a cada classe.

aplicada, por fim, a que apresentou maior média de acurácia para a maioria das bases. A tabela a seguir apresenta um resumo da arquitetura da rede neural utilizada para o processo de classificação.

Tabela 3 – Exemplo de arquitetura de rede neural aplicada em cada base de dados.

Tipo de camada	Tamanho da camada	Número de filtros
Convolutacional	3 x 3	16
MaxPooling	2 x 2	16
Convolutacional	3 x 3	32
MaxPooling	2 x 2	32
Convolutacional	3 x 3	64
MaxPooling	2 x 2	64
Totalmente Conectada	128	—
Dropout	0,2	—
Camada de saída	n	—

Como o treinamento da rede neural convolutacional é um processo que apresenta valores diferentes de pesos treinados e acurácia a cada rodada em que ele é executado, esse processo foi realizado várias vezes para se obter uma base estatística. Para cada base e para cada entrada proposta para a geração das imagens (dados brutos, FTS, Polar) foram executadas 30 repetições do algoritmo para cada base de dados. Cada uma das trinta execuções realizadas retornou um valor de acurácia ao ser aplicado nos dados de teste disponibilizados. Esses resultados foram agrupados para cada um dos tipos de entradas e, portanto, analisados e comparados entre si, a fim de se entender se a média da acurácia obtida é estatisticamente diferente entre os algoritmos propostos. As simulações para treinamento e teste pela CNN foram realizadas no ambiente Google Colab e os resultados finais obtidos para cada base de dados, após as devidas análises estatísticas, estão apresentados mais adiante, na Tabela 5.

6.3.1 Comparação entre os algoritmos

Devido à natureza estocástica do processo de treinamento da rede neural convolutacional, necessário para o posterior processo de classificação baseado em imagens, foram coletadas 30 amostras de acurácia para cada uma das metodologias de transformação da série temporal e geração das imagens, discutidas anteriormente e sintetizadas a seguir:

1. Dados brutos da série temporal transformados em uma imagem a partir da representação matricial (Motifs)
2. Série temporal fuzzificada pela biblioteca PyFTS a partir do método *Grid Partitioning*, transformada em uma imagem a partir da representação matricial (FTSMotifs)

3. Dados brutos da série temporal transformados em uma imagem em coordenadas polares (Polar)

Após realizadas as trinta rodadas de treinamento e teste com cada uma das técnicas de geração das imagem, descritas acima, pode-se, por fim, comparar os resultados obtidos, a fim de se entender se há diferença estatística entre elas. Para tal, foi realizado um teste de análise de variâncias, capaz de indicar se há diferença relevante entre as médias de acurácia obtidas. Caso essa diferença exista, realiza-se um teste par a par posterior (todos contra todos), a fim de quantificá-la com certo intervalo de confiança. O processo de comparação entre as médias de acurácias foi realizado na linguagem R, através do Software RStudio. Para ilustrar o procedimento realizado, bem como análise dos resultados obtidos, esse processo será ilustrado para a base de dados *Adiac*, podendo ser replicado de maneira semelhante para as demais bases de dados avaliadas.

Visualização das médias obtidas

O primeiro passo realizado foi a visualização das acurácias obtidas para cada metodologia através de um *Boxplot*, apresentado na Figura 18, o qual se refere aos resultados obtidos para a base de dados *DodgerLoopGame*.

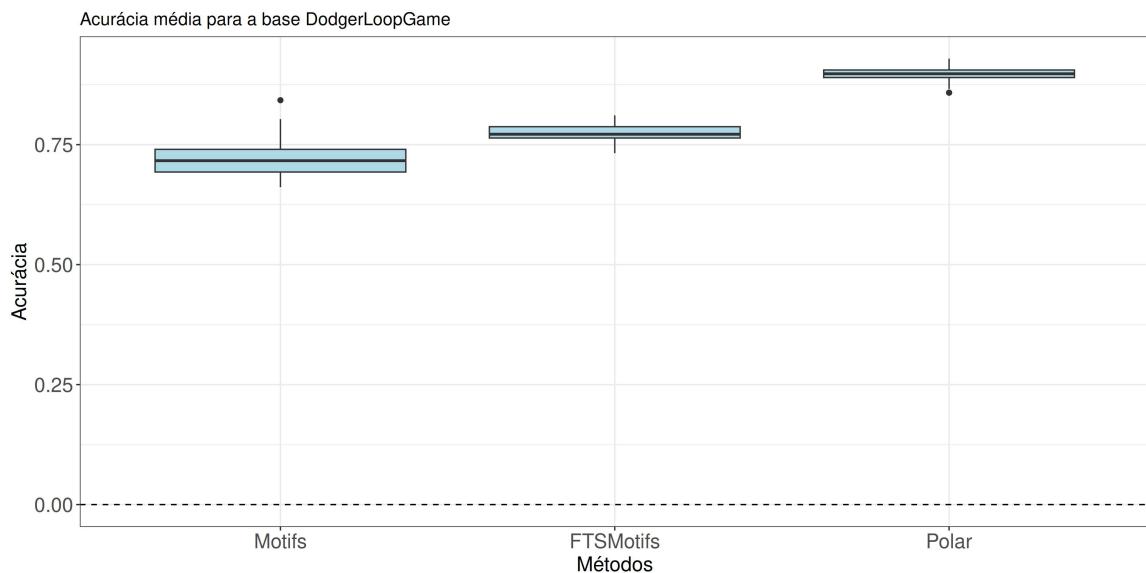


Figura 18 – *Boxplot* das acurácias obtidas para a base de dados *DodgerLoopGame*

Para o caso da base de dados utilizada como exemplo, pode-se inferir, visualmente pelo *Boxplot* que existe relativa diferença entre as acurácias obtidas por cada metodologia de representação da série temporal em da imagem, além de uma maior variância obtida para o método Motifs. Essas características podem influenciar na comparação entre os métodos, e serão, portanto, consideradas ao longo da análise.

Análise de variâncias

O passo seguinte, após análise visual dos dados pelo *boxplot*, é o teste de análise de variâncias propriamente dito. Sabendo-se que, para cada método, existe um valor τ_i , referente ao efeito da acurácia do método i na média geral μ , caso não haja evidências de que a média das acurácias seja diferente, o valor do efeito τ_i para todos os métodos deve ser zero, com certo intervalo de confiança, definido como 0,05. Sendo assim, o seguinte teste de hipóteses pode ser definido:

$$\begin{cases} H_0 : \tau_i = 0 \\ H_1 : \tau_i \neq 0 \end{cases} \quad \forall i \in [1, 2, 3, 4] \quad (6.1)$$

onde τ é o efeito da acurácia de cada método na média geral μ , representando o desvio da média da acurácia de cada método. Caso o p-valor obtido pelo teste seja menor que 0,05, rejeita-se a hipótese nula de que as médias são iguais; caso contrário, não há evidências suficientes para se rejeitar a hipótese nula, considerando-se as mesmas médias de acurácia.

O teste mais utilizado para esse fim é denominado ANOVA (*Analysis of Variance*), o qual retorna um p-valor, indicando se há evidências suficientes para aceitar ou rejeitar a hipótese nula de diferença entre as médias. Esse teste, porém, requer o cumprimento de algumas condições (MONTGOMERY, 2017; MONTGOMERY; RUNGER, 2010):

1. As observações devem seguir uma distribuição normal
2. As observações devem ser independentemente distribuídas
3. Igualdade de variâncias entre cada grupo (homocedasticidade)

Como cada simulação para a obtenção dos pesos da rede neural foi realizada separadamente, independentemente uma da outra, a partir do mesmo procedimento, pode-se considerá-las como independentes e identicamente distribuídas. Para as demais considerações, normalidade e homocedasticidade, foram realizados testes no software R, a fim de aceitar ou rejeitar cada uma dessas hipóteses. Para a hipótese de normalidade foi realizado o teste de Shapiro-Wilk e para a hipótese de homocedasticidade, o teste de Barlett, caso as observações sejam normais e o teste de Fligner, caso não sejam. Segundo Montgomery e Runger (2010), em situações de amostras balanceadas, ou seja, número de amostras iguais para cada método, mesmo que a hipótese de homogeneidade das variâncias seja rejeitada, o resultado do teste F é levemente afetado. De qualquer maneira, caso essa situação ocorra, a fim de se certificar o resultado, será também realizado um teste para indicar se há a diferença entre as médias, capaz de ser utilizado para dados

com variâncias não homogêneas, conhecido como teste de Welch. O próximo passo, é testar a homogeneidade das variâncias, obtido através do teste de Barlett.

A seguir, é possível realizar a análise de variâncias propriamente dita, a fim de indicar se há diferença relevante entre as médias de cada método de transformação e geração das imagens. O teste Welch, para a base de dados *DodgerLoopGame*, apresentou um p-valor muito menor que 0,05 (2.54×10^{-33} exibido pelo software R), rejeitando, portanto, a hipótese de igualdade entre as médias. Com essa situação, o próximo passo é realizar a comparação par a par, a fim de se avaliar o intervalo de confiança para cada par de métodos. Para tal, foram considerados dois testes, o teste de Turkey e o teste de Games-Howell, ambos capazes de calcular o intervalo de confiança, porém, dependem da homogeneidade das variâncias das amostras. Caso as observações sejam homocedasticas, será aplicado o teste de Turkey, caso contrário, será aplicado o teste de Games-Howell. A Tabela 4 apresenta o resultado do teste de Games-Howell para a base *DodgerLoopGame*, uma vez que o teste de Barlett rejeitou a hipótese de homocedasticidade.

Tabela 4 – Resultado do teste de Games-Howell, na base de dados *DodgerLoopGame*, para comparação par a par após o teste F .

Método 1	Método 2	p-valor	H_0
Motifs	FTSMotifs	$2.99e - 06$	Rejeitada
Motifs	Polar	$3.48e - 21$	Rejeitada
FTSMotifs	Polar	$8.00e - 32$	Rejeitada

Por fim, segue a Figura 19, a qual apresenta, visualmente, a comparação entre cada método através dos intervalos de confiança obtidos pelo teste de Games-Howell. Através da imagem é, também, possível perceber se determinado método apresentou acurácia inferior ou superior em relação a outro.

A partir da Tabela 4 e da Figura 19, percebe-se que a hipótese nula de igualdade entre as médias foi rejeitada para todas as metodologias de representação das séries temporais. Sendo assim, considera-se todas as médias diferentes uma das outras para essa base.

Os resultados serão apresentados em tabelas, de tal maneira que uma tabela apresenta a média e desvio padrão para todas as metodologias baseadas em imagens. Posteriormente, para a comparação entre métodos baseados em imagens e métodos baseados em medidas de similaridade aplicadas a Séries Temporais Fuzzy, será considerado o melhor resultado obtido, ou seja, a maior média estatisticamente relevante, seguida de seu desvio padrão. Avaliando-se o resultado obtido para a base *DodgerLoopGame*, considera-se que a maior média obtida foi referente à representação da série através de coordenadas polares.

O procedimento apresentado até então foi repetido para todas as bases de dados analisadas, sendo o resultado dos métodos baseados em imagens dispostos na Tabela 5. Os

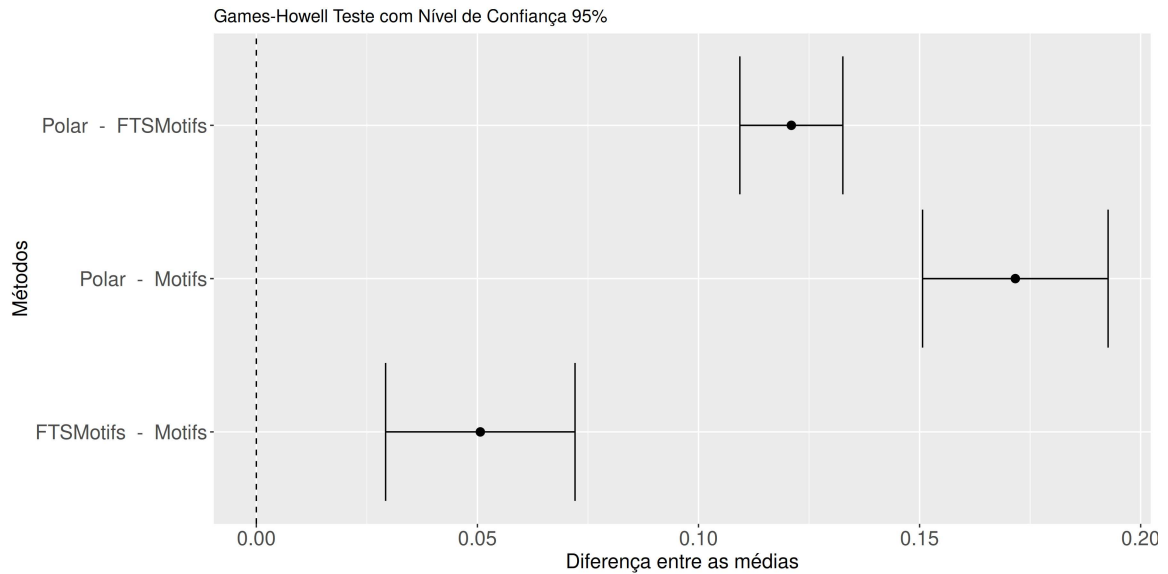


Figura 19 – Intervalo de confiança para a comparação par a par pelo teste de Games-Howell para a base de dados *DodgerLoopGame*

melhores resultados obtidos serão indicados em negrito na tabela. Adiante, os melhores resultados obtidos por essa abordagem serão comparados com as demais abordagem de classificação, apresentados posteriormente, na Tabela 9.

Tabela 5 – Resultados da classificação baseada em imagens a partir da CNN

Base	Motifs	FTSMotifs	Polar
ACSF1	0,482	0,520	0,384
AllGestureWiimoteX	0,439	0,407	0,338
AllGestureWiimoteY	0,459	0,466	0,416
AllGestureWiimoteZ	0,371	0,346	0,366
Beef	0,657	0,551	0,642
CBF	0,942	0,925	0,988
Chinatown	0,981	0,983	0,857
Computers	0,696	0,550	0,691
CricketX	0,501	0,480	0,536
CricketY	0,518	0,509	0,542
CricketZ	0,488	0,528	0,552
DistalPhalanxOutlineAgeGroup	0,695	0,703	0,708
DistalPhalanxTW	0,690	0,619	0,628
DodgerLoopDay	0,601	0,576	0,543
DodgerLoopGame	0,723	0,773	0,894
DodgerLoopWeekend	0,976	0,978	0,962
ECG200	0,838	0,846	0,819
FaceFour	0,924	0,902	0,817

Continua na próxima página

Tabela 5 – *Continua na página anterior*

Fish	0,813	0,759	0,806
GestureMidAirD1	0,361	0,393	0,422
GestureMidAirD2	0,325	0,378	0,346
GestureMidAirD3	0,228	0,209	0,196
GesturePebbleZ1	0,891	0,894	0,752
GesturePebbleZ2	0,859	0,891	0,653
Ham	0,597	0,683	0,628
Lightning2	0,631	0,698	0,760
Meat	0,852	0,803	0,912
MelbournePedestrian	0,826	0,821	0,866
MixedShapesRegularTrain	0,844	0,866	0,891
MixedShapesSmallTrain	0,779	0,801	0,842
MoteStrain	0,884	0,876	0,782
OliveOil	0,881	0,591	0,492
PickupGestureWiimoteZ	0,606	0,656	0,592
PLAID	0,657	0,683	0,820
SemgHandGenderCh2	0,620	0,768	0,914
ShakeGestureWiimoteZ	0,707	0,719	0,605
SonyAIBORobotSurface1	0,716	0,750	0,776
SonyAIBORobotSurface2	0,792	0,784	0,836
StarLightCurves	0,961	0,940	0,958
TwoLeadECG	0,825	0,781	0,765

6.4 Classificação com Aprendizado Federado

Os últimos resultados a serem discutidos são referentes à classificação através de Aprendizado Federado. Esses resultados foram divididos em duas partes: a primeira é referente ao método de aprendizado federado aplicado aos dados de treinamento particionados e distribuídos entre três clientes, a fim de simular uma situação de dados distribuídos; a segunda refere-se à classificação multitarefa sendo cada base de dados considerada como um cliente separadamente.

6.4.1 Dados de treinamento particionados

Na presente subseção, são apresentados os resultados obtidos com a proposta de aprendizado federado realizada através do particionamento dos dados de treinamento de cada base de dados. Além disso, os resultados obtidos com a proposta essa proposta

será comparado à proposta centralizada, considerando-se cada os dados de treinamento particionados, a fim de se entender o efeito da metologia nos resultados da classificação. Posteriormente, os melhores resultados das propostas anteriores, bem como os resultados utilizando FL para os dados particionados serão comparados com métodos *benchmark* e Estado da Arte.

A Tabela 6 apresenta uma comparação detalhada entre o modelo federado (FL) proposto e o método centralizado de classificação baseado em imagens. O modelo FL foi especificamente projetado e otimizado para emular a estrutura e o desempenho do modelo centralizado com melhor performance em cada conjunto de dados. Por exemplo, no caso do conjunto de dados ACSF1, o modelo centralizado FTSMotif obteve a maior acurácia, portanto o modelo federado foi ajustado para alinhar-se a essa abordagem. Essa estratégia garante uma comparação justa, destacando a eficácia do modelo FL em ambientes distribuídos. Características-chave, como o número de amostras de treinamento e teste, o número de classes e o comprimento das amostras, revelam as complexidades e os desafios impostos por cada conjunto de dados, oferecendo uma compreensão mais aprofundada do desempenho do FL em relação às abordagens centralizadas.

Tabela 6 – Resultados do método FL comparado com o método centralizado de classificação baseado em imagens

Base	Motifs	FTSMotifs	Polar	FL
ACSF1	0,482	0,520	0,384	0,550
AllGestureWiimoteX	0,439	0,407	0,338	0,352
AllGestureWiimoteY	0,459	0,466	0,416	0,458
AllGestureWiimoteZ	0,371	0,346	0,366	0,193
Beef	0,657	0,551	0,642	0,667
CBF	0,942	0,925	0,988	0,991
Chinatown	0,981	0,983	0,857	0,982
Computers	0,696	0,550	0,691	0,620
CricketX	0,501	0,480	0,536	0,548
CricketY	0,518	0,509	0,542	0,530
CricketZ	0,488	0,528	0,552	0,566
DistalPhalanxOutlineAgeGroup	0,695	0,703	0,708	0,741
DistalPhalanxTW	0,690	0,619	0,628	0,712
DodgerLoopDay	0,601	0,576	0,543	0,426
DodgerLoopGame	0,723	0,773	0,894	0,682
DodgerLoopWeekend	0,976	0,978	0,962	0,874
ECG200	0,838	0,846	0,819	0,81
FaceFour	0,924	0,902	0,817	0,954
Fish	0,813	0,759	0,806	0,811

Continua na próxima página

Tabela 6 – *Continua na página anterior*

GestureMidAirD1	0,361	0,393	0,422	0,269
GestureMidAirD2	0,325	0,378	0,346	0,138
GestureMidAirD3	0,228	0,209	0,196	0,192
GesturePebbleZ1	0,891	0,894	0,752	0,796
GesturePebbleZ2	0,859	0,891	0,653	0,873
Ham	0,597	0,683	0,628	0,651
Lightning2	0,631	0,698	0,760	0,704
Meat	0,852	0,803	0,912	0,933
MelbournePedestrian	0,826	0,821	0,866	0,850
MixedShapesRegularTrain	0,844	0,866	0,891	0,892
MixedShapesSmallTrain	0,779	0,801	0,842	0,842
MoteStrain	0,884	0,876	0,782	0,881
OliveOil	0,881	0,591	0,492	0,900
PickupGestureWiimoteZ	0,606	0,656	0,592	0,620
PLAID	0,657	0,683	0,820	0,802
SemgHandGenderCh2	0,620	0,768	0,914	0,905
ShakeGestureWiimoteZ	0,707	0,719	0,605	0,660
SonyAIBORobotSurface1	0,716	0,750	0,776	0,742
SonyAIBORobotSurface2	0,792	0,784	0,836	0,844
StarLightCurves	0,961	0,940	0,958	0,958
TwoLeadECG	0,825	0,781	0,765	0,816

Alguns pontos interessantes podem ser citados ao se comparar os resultados obtidos com as estratégias centralizada e FL. Bases de dados com instâncias de treinamento limitadas, como ACSF1, com 100 instâncias, e Beef, com 30 instâncias, demonstram a resiliência do modelo FL, em cenários descentralizados, nos quais a escassez de dados representa um desafio significativo. Apesar do tamanho reduzido desses conjuntos, o modelo FL atinge um desempenho próximo ou até superior ao dos melhores métodos centralizados, evidenciando sua capacidade de generalização eficaz em ambientes distribuídos. Para os dados com maior volume de instâncias de treinamento, o modelo federado continua a apresentar desempenho competitivo, como StarLightCurves, que possui 1000 instâncias de treinamento e MelbournePedestrian, com 1194 instâncias. Esse resultado ilustra sua escalabilidade e capacidade de lidar com volumes maiores de dados, mantendo elevada acurácia. Para o caso das bases com alta complexidade de classificação, como CricketX, CricketY, and CricketZ, que possuem 12 classes, ou GestureMidAirD1, GestureMidAirD2, and GestureMidAirD3 com 26 classes, também destacam a eficácia do modelo FL. Esses conjuntos representam desafios significativos devido à sua natureza com várias classes, porém o modelo FL demonstra desempenho consistente, reforçando sua adequação para tarefas complexas de classificação.

Outra característica capaz de ser notada no modelo FL é sua adaptabilidade, particularmente evidente em bancos de dados com tamanhos variáveis de amostras por instâncias. Bases como AllGestureWiimoteX, AllGestureWiimoteY, AllGestureWiimoteZ, and PLAID apresentam essa característica, podendo introduzir variabilidade adicional nas resoluções temporais. Mesmo com esse desafio, o modelo FL obtém resultados competitivos, demonstrando sua flexibilidade frente a distribuições de dados diversas.

O modelo FL se assemelha às abordagens centralizadas nas bases com alto volume de instâncias de teste, como CBF, que possui 900 instâncias, e StarLightCurves, com 8236 instâncias. Essa capacidade de generalização diante de grandes conjuntos de teste, potencialmente desbalanceados, evidencia a robustez do FL em aplicações reais. Outro caso em que o modelo FL apresenta resultados satisfatórios ocorre em bases pequenas e especializadas, como OliveOil e Lightning2, que possuem quantidades reduzidas de amostras de treinamento e teste. Esse comportamento reforça sua versatilidade e sua capacidade de apresentar bons resultados em uma ampla variedade de conjuntos, independentemente do tamanho ou domínio.

Percebe-se, portanto, que os resultados reforçam a capacidade do modelo FL proposto em lidar com diversos desafios presentes em ambientes descentralizados. A escalabilidade, robustez e adaptabilidade do modelo FL permitem alcançar um desempenho comparável (em alguns casos, superior) ao dos modelos centralizados, tornando-o uma escolha promissora para aplicações reais de aprendizado distribuído. Para ilustrar ainda mais os ganhos do modelo FL devido ao compartilhamento dos parâmetros e treinamento descentralizado, a Tabela 7 apresenta uma comparação da acurácia obtida com o modelo FL, classificado pelo servidor, com uma classificação realizada pelos clientes individuais, treinados de maneira isolada. Nesse cenário, cada cliente é treinado localmente com dados oriundos de uma distribuição de Dirichlet e testado com o conjunto total de teste. Os resultados mostram que a classificação com o servidor FL supera consistentemente os clientes individuais, evidenciando o poder do aprendizado colaborativo. Por exemplo, no conjunto AllGestureWiimoteX, o servidor FL atinge uma acurácia de 0,352, significativamente superior à dos clientes individuais. De forma similar, em conjuntos como Beef e CricketX, o servidor FL supera os modelos isolados, comprovando sua capacidade de agregar conhecimento de forma eficaz.

A justificativa teórica para o desempenho superior do modelo em relação ao treinamento isolado é respaldada pela literatura existente, que demonstra que o FL supera consistentemente o treinamento local isolado ao agregar dados diversos dos clientes, melhorando assim a generalização do modelo e reduzindo o viés (BARROS et al., 2024; ZHU; MA; BLASCHKO, 2023). Além disso, garantias de convergência estabelecidas em pesquisas anteriores confirmam que os métodos de otimização em FL são eficazes no tratamento de dados não-IID, assegurando um treinamento robusto e desempenho estável em distribuições heterogêneas entre os clientes (LI et al., 2019).

Para ilustrar os resultados obtidos na Tabela 7, segue a Figura 20, com uma representação visual da comparação entre o treinamento individualizado em cada cliente, sem colaboração, e o treinamento com aprendizado federado. Os pontos pretos representam, no eixo y, a acurácia dos clientes individuais e, no eixo x, a acurácia federada. A linha preta diagonal denota a igualdade de

Tabela 7 – Comparação da acurácia do método FL-Server contra utilização de treinamento local dentro de cada cliente e teste com os dados de teste completos.

Dataset	FL Server	Client0	Client1	Client2
ACSF1	0,550	0,484	0,383	0,386
AllGestureWiimoteX	0,352	0,348	0,314	0,275
AllGestureWiimoteY	0,458	0,332	0,362	0,339
Beef	0,667	0,441	0,470	0,520
CBF	0,991	0,946	0,853	0,929
Computers	0,620	0,576	0,506	0,539
CricketX	0,548	0,405	0,401	0,384
CricketY	0,530	0,417	0,425	0,428
CricketZ	0,566	0,417	0,440	0,396
DistalPhalanxOutlineAgeGroup	0,741	0,702	0,711	0,680
DistalPhalanxTW	0,712	0,644	0,663	0,642
DodgerLoopDay	0,426	0,122	0,140	0,103
DodgerLoopGame	0,937	0,748	0,770	0,862
DodgerLoopWeekend	0,874	0,702	0,711	0,680
ECG200	0,810	0,802	0,686	0,647
StarLightCurves	0,958	0,941	0,937	0,928
TwoLeadECG	0,816	0,602	0,682	0,587

desempenho entre os clientes e o modelo federado, dividindo a figura em duas regiões. A primeira região (sombreada em cinza), acima da linha preta, indica os casos em que os clientes obtiveram maior acurácia do que o modelo federado. Em contraste, a segunda região (sem sombreadamento), abaixo da linha preta, mostra os casos em que o aprendizado federado superou os modelos dos clientes individuais. As acurácias dos três clientes para cada conjunto de dados (Tabela 7) são comparadas com as acurácias obtidas de forma colaborativa (federada).

Adicionalmente, a linha vermelha conecta os valores médios de acurácia dos três clientes, fornecendo uma comparação direta com o desempenho do modelo federado. Na Figura 20, um ponto preto (Client0 do conjunto de dados DistalPhalanxOutlineCorrect) encontra-se na região dos clientes, enquanto os demais pontos pretos e a linha vermelha situam-se na região do FL. Como esperado, esse resultado demonstra que o modelo federado supera o desempenho dos clientes individuais. Embora o método FL proposto tenha utilizado apenas três clientes, o resultado já evidencia uma vantagem clara do FL.

Por fim, pode-se perceber que a técnica FL proposta oferece duas vantagens significativas: preserva a privacidade dos dados e melhora a acurácia preditiva por meio de sua estrutura de aprendizado colaborativo.

6.4.2 Classificação Multitarefa com aprendizado federado

Na classificação multitarefa, cada cliente é representado por uma das bases de dados analisadas, existindo, portanto, 44 clientes. Os resultados obtidos nessa classificação consideraram

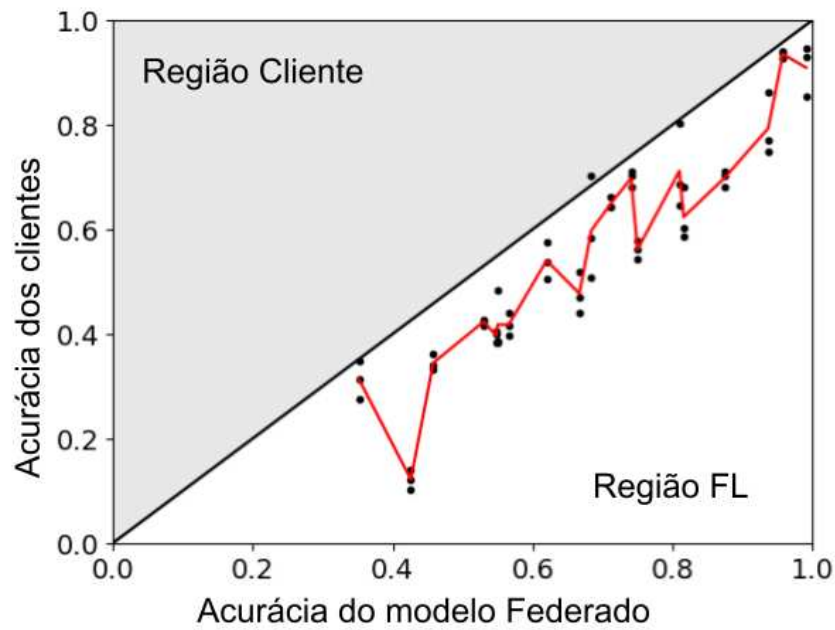


Figura 20 – Comparação entre o treinamento local dos clientes e o aprendizado federado. Os pontos pretos representam a acurácia dos clientes, com a acurácia individual no eixo y e a acurácia federada no eixo x. A linha vermelha indica a acurácia média dos três clientes. Uma linha diagonal separa as regiões sombreada e não sombreada, onde os pontos na região não sombreada indicam que o aprendizado federado obteve acurácia superior ao treinamento individual.

um treinamento realizado utilizando no mínimo 10% desses. A avaliação final, por sua vez, realizada com os dados de teste para todos os clientes. Isso ocorre, pois a acurácia de avaliação foi aplicada para se medir a acurácia final do método em cada base de dados, com o intuito de ser comparada com os demais métodos estudados e com métodos de Estado da Arte.

A Tabela 8 apresenta os resultados obtidos com a aplicação da classificação multitarefa nos dados de teste. Para se ter uma referência de comparação, resultados de *baseline* utilizando o agregador FedAVG, apresentados no trabalho de Xing et al. (2022b), bem como o método proposto no trabalho foram incluídos na tabela, uma vez que tratam de objetivo semelhante e serviram de base para a proposta de classificação multitarefa realizada.

Tabela 8 – Resultados do método multitarefa comparado com a melhor acurácia centralizada, bem como com resultados obtidos em Xing et al. (2022b)

Base	Centralizada	FedAvg	EFDLS	FL-Multitask
ACSF1	0,520	0,100	0,880	0,550
AllGestureWiimoteX	0,439	0,100	0,291	0,321
AllGestureWiimoteY	0,466	0,100	0,283	0,360
AllGestureWiimoteZ	0,371	0,100	0,379	0,290
Beef	0,657	0,200	0,767	0,367
CBF	0,988	0,333	0,996	0,332

Continua na próxima página

Tabela 8 – *Continua na página anterior*

Chinatown	0,983	0,275	0,948	0,982
Computers	0,696	0,500	0,804	0,544
CricketX	0,536	0,069	0,672	0,079
CricketY	0,542	0,095	0,597	0,074
CricketZ	0,552	0,085	0,726	0,061
DistalPhalanxOutlineAgeGroup	0,708	0,108	0,647	0,467
DistalPhalanxTW	0,690	0,129	0,640	0,727
DodgerLoopDay	0,601	0,150	0,537	0,597
DodgerLoopGame	0,894	0,522	0,746	0,512
DodgerLoopWeekend	0,978	0,739	0,920	0,944
ECG200	0,846	0,360	0,850	0,800
FaceFour	0,924	0,159	0,682	0,409
Fish	0,813	0,137	0,623	0,743
GestureMidAirD1	0,422	0,038	0,577	0,038
GestureMidAirD2	0,378	0,038	0,531	0,362
GestureMidAirD3	0,228	0,038	0,277	0,223
GesturePebbleZ1	0,894	0,163	0,488	0,738
GesturePebbleZ2	0,891	0,152	0,589	0,791
Ham	0,683	0,486	0,695	0,638
Lightning2	0,760	0,459	0,787	0,541
Meat	0,912	0,333	0,917	0,333
MelbournePedestrian	0,866	0,100	0,945	0,105
MixedShapesRegularTrain	0,891	0,189	0,891	0,270
MixedShapesSmallTrain	0,842	0,189	0,839	0,172
MoteStrain	0,884	0,539	0,820	0,819
OliveOil	0,881	0,167	0,833	0,767
PickupGestureWiimoteZ	0,656	0,100	0,740	0,540
PLAID	0,820	0,061	0,259	0,173
SemgHandGenderCh2	0,914	0,650	0,720	0,728
ShakeGestureWiimoteZ	0,719	0,100	0,960	0,600
SonyAIBORobotSurface1	0,776	0,571	0,882	0,429
SonyAIBORobotSurface2	0,836	0,383	0,896	0,617
StarLightCurves	0,961	0,143	0,958	0,947
TwoLeadECG	0,825	0,500	0,766	0,708

Analisando os resultados da Tabela 8, percebe-se que a abordagem multitarefa apresenta resultados competitivos, comparados com as referências utilizadas. Em bases como DistalPhalanxOutlineCorrect, DistalPhalanxTW percebem-se resultados superiores para essa metodologia, comparadas a todas as referências. Ao se analisar os resultados para essas mesmas bases de dados

na abordagem anterior, com particionamento dos dados de treinamento, observam-se resultados semelhantes, os quais foram, também, superiores aos métodos centralizados. Uma provável razão desse resultado é o fato de essas bases possuírem um grande número de dados de treinamento, comparados aos dados de teste, bem como em relação às demais bases de dados. Assim, o modelo pode ter se adequado bem aos dados provenientes dessas bases, apresentando bons resultados.

Alguns resultados, por sua vez, apresentaram desempenho semelhante aos melhores resultados das técnicas comparadas, porém com acurácia ligeiramente inferior. Exemplos dessas bases são *DodgerLoopDay*, *DodgerLoopWeekend*, *Ham*, *MoteStrain*, dentre outros. Essas bases ou apresentam um número de instâncias de treinamento próximo ao número de instâncias de teste, ou apresentam um grande número de amostras. Esses fatos podem contribuir para um valor de acurácia moderado para a maioria delas.

Para muitas das bases, porém, o resultado obtido pela estratégia Multitarefa foi baixo, comparado tanto com o método centralizado quanto o algoritmo EFDLS (XING et al., 2022b). Porém, ao ser comparado com o método baseado no extrator de características proposto em Xing et al. (2022b), mas que utiliza o agregador FedAvg, a maioria dos resultados se mostrou com melhor desempenho. Das 44 bases, 35 apresentaram acurácia igual ou superior ao agregador FedAvg, estratégia de agregação semelhante à utilizada no modelo multitarefa proposto. Esse fato destaca a possibilidade de melhoria a ser implementada no modelo multitarefa. Estudos posteriores podem ser direcionados para a utilização de diferentes estratégias de agregação do modelo pelo servidor, a fim de se obter uma melhor generalização do modelo e melhores valores de acurácia.

6.5 Comparação dos resultados obtidos

Após a obtenção dos resultados com cada uma das estratégias estudadas, os melhores resultados obtidos foram comparados entre si e entre resultados de *benchmark* encontrados na página da UCR (DAU et al., 2018). Os autores apresentam uma comparação entre três principais metodologias: a Distância Euclideana (ED), chamada, no presente trabalho, de $\mathcal{L}_2$; a distância DTW com janelamento (o valor da janela utilizada está indicada entre parêntesis). A Tabela 9 apresenta a comparação entre a acurácia das estratégias de *benchmark* com os melhores resultados obtidos, tanto para a classificação baseada em medidas de similaridade aplicadas a séries temporais fuzzy (MSFTS) quanto para a classificação baseada em imagens (CNN).

A tabela está disposta da seguinte forma:

- a primeira coluna indica a base de dados estudada;
- as colunas dois a quatro representam os métodos de *benchmark*;
- as demais colunas representam os métodos propostos, com sua respectiva técnica responsável pelos melhores resultados, quando for o caso

Tabela 9 – Comparação entre os melhores resultados das estratégias propostas com métodos de *benchmark*

Base	$\mathcal{L}_2$	DTW(100)	DTW(w)	MSFTS		CNN		FL-Server	FL-Multitask
ACSF1	0,540	0,640	0,620(4)	0,640	FTS-LCS	0,520	FTSMotifs	0,550	0,550
AllGestureWiimoteX	0,516	0,716	0,717(14)	0,529(100)	FTS-DTW	0,439	Motifs	0,352	0,321
AllGestureWiimoteY	0,569	0,729	0,730(9)	0,601(100)	FTS-DTW	0,466	FTSMotifs	0,458	0,360
AllGestureWiimoteZ	0,454	0,643	0,651(11)	0,479(100)	FTS-DTW	0,371	Motifs	0,193	0,290
Beef	0,667	0,633	0,667(0)	0,667	FTS- $\mathcal{L}_2$	0,657	Motifs	0,667	0,367
CBF	0,852	0,997	0,996	0,998(11)	FTS-DTW	0,988	Polar	0,991	0,332
Chinatown	0,953	0,956	0,953(0)	0,918	FTS- $\mathcal{L}_2$	0,983	FTSMotifs	0,982	0,982
Computers	0,576	0,700	0,620(12)	0,676(100)	FTS-DTW	0,696	Motifs	0,620	0,544
CricketX	0,577	0,754	0,772(10)	0,605(10)	FTS-DTW	0,536	Polar	0,548	0,079
CricketY	0,567	0,744	0,759(17)	0,603(12)	FTS-DTW	0,542	Polar	0,530	0,074
CricketZ	0,587	0,754	0,746(5)	0,659(5)	FTS-DTW	0,552	Polar	0,566	0,061
DistalPhalanxOutlineAgeGroup	0,626	0,770	0,626(0)	0,712(100)	FTS-DTW	0,708	Polar	0,741	0,467
DistalPhalanxTW	0,633	0,590	0,633(0)	0,604(100)	FTS-DWT	0,690	Motifs	0,712	0,727
DodgerLoopDay	0,550	0,500	0,587(1)	0,442	FTS-DTW	0,601	Motifs	0,426	0,597
DodgerLoopGame	0,884	0,877	0,927(1)	0,795	FTS- $\mathcal{L}_2$	0,894	Polar	0,682	0,512
DodgerLoopWeekend	0,985	0,949	0,978(1)	0,944	FTS-DTW	0,978	FTSMotifs	0,874	0,944
ECG200	0,880	0,770	0,880(0)	0,900(0)	FTS-DTW	0,846	FTSMotifs	0,810	0,800
FaceFour	0,784	0,829	0,886(2)	0,784(2)	FTS-DTW	0,924	Motifs	0,954	0,409
Fish	0,783	0,823	0,846(4)	0,880	FTS-LCS	0,813	Motifs	0,811	0,743
GestureMidAirD1	0,577	0,569	0,638(5)	0,692(5)	FTS-DTW	0,422	Polar	0,269	0,038
GestureMidAirD2	0,492	0,608	0,600(6)	0,631(6)	FTS-DTW	0,378	FTSMotifs	0,138	0,362
GestureMidAirD3	0,346	0,323	0,377(1)	0,5154(100)	FTS-DTW	0,228	Motifs	0,192	0,223
GesturePebbleZ1	0,733	0,791	0,826(2)	0,645(100)	FTS-DTW	0,894	FTSMotifs	0,796	0,738

Continua na próxima página

Tabela 9 – *Continua na página anterior*

GesturePebbleZ2	0,671	0,671	0,778(6)	0,658(100)	FTS-DTW	0,891	FTSMotifs	0,873	0,791
Ham	0,600	0,467	0,600(0)	0,590(0)	FTS-DTW	0,683	FTSMotifs	0,651	0,638
Lightning2	0,754	0,869	0,869(6)	0,836(6)	FTS-DTW	0,760	Polar	0,704	0,541
Meat	0,933	0,933	0,933(0)	0,733	Todas	0,912	Polar	0,933	0,333
MelbournePedestrian	0,847	0,815(1)	0,791	0,714	FTS- $\mathcal{L}_2$	0,866	Polar	0,850	0,105
MixedShapesRegularTrain	0,897	0,842	0,909(4)	0,906	FTS- $\mathcal{L}_2$	0,891	Polar	0,892	0,270
MixedShapesSmallTrain	0,835	0,780	0,833(7)	0,858(7)	FTS-DTW	0,842	Polar	0,842	0,172
MoteStrain	0,879	0,835	0,866	0,822(1)	FTS-DTW	0,884	Motifs	0,881	0,819
OliveOil	0,867	0,833	0,867(0)	0,600	FTS-LCS	0,881	Motifs	0,900	0,767
PickupGestureWiimoteZ	0,560	0,660	0,660(17)	0,500(17)	FTS-DTW	0,656	FTSMotifs	0,620	0,540
PLAID	0,523	0,836	0,834(12)	0,754(12)	FTS-DTW	0,820	Polar	0,802	0,173
SemgHandGenderCh2	0,762	0,802	0,845(1)	0,803(1)	FTS-DTW	0,914	Polar	0,905	0,728
ShakeGestureWiimoteZ	0,600	0,860	0,840(6)	0,720(100)	FTS-DTW	0,719	FTSMotifs	0,660	0,600
SonyAIBORobotSurface1	0,695	0,725	0,695	0,729	FTS- $\mathcal{L}_2$	0,776	Polar	0,742	0,429
SonyAIBORobotSurface2	0,859	0,831	0,859(0)	0,849	FTS- $\mathcal{L}_2$	0,836	Polar	0,844	0,617
StarLightCurves	0,849	0,907	0,905(16)	0,906	FTS-LCS	0,961	Motifs	0,958	0,947
TwoLeadECG	0,747	0,904	0,868(4)	0,804	FTS-LCS	0,825	Motifs	0,816	0,708

Ao se comparar as estratégias centralizadas de classificação propostas com as estratégias de *benchmark* disponíveis em (DAU et al., 2018), percebe-se que, na maioria das bases analisadas, houve resultados semelhantes, tanto pelas metodologias baseadas em imagens e CNN quanto pelas metodologias baseadas em medidas de similaridade aplicadas a séries temporais fuzzy. Em alguns deles, como CBF, Chinatown, ECG200, dentre outras, as estratégias propostas apresentaram maior acurácia que as estratégias *benchmark*. Vale salientar, porém, que, como explicitado em seus respectivos capítulos, os métodos centralizados propostos não passaram por otimização de seus hiperparâmetros, e ainda sim, apresentando bons resultados, comparáveis com estratégias de *benchmark*. Isso indica que esses métodos apresentam resultados promissores, podendo ser melhorados com novas abordagens e otimizações, a fim de se obter maiores acurácias.

No que tange às abordagens de aprendizado federado, principais contribuições do presente trabalho, além de trazer consigo as características da classificação baseada em imagens, elas são capazes de promover a privacidade dos dados dos clientes, além do compartilhamento do aprendizado em cada nó. Assim, considerando-se, por exemplo, a abordagem que aplica o particionamento dos dados de treinamento em três clientes, e cuja classificação é realizada pelo modelo global, mesmo que treinado de maneira isolada, agrega esse aprendizado compartilhado, auxiliando na classificação dos dados de teste. Esse fato pode ser percebido na Figura 20 que apresenta esse ganho de maneira explícita, porém também na Tabela 9, na qual em bases de dados como FaceFour, Meat, Oliveoil, dentre outras, o resultado obtido pelo método FL superou a classificação pelo método centralizado. Assim, a vantagem do método que utiliza FL em relação aos métodos *benchmark*, os quais necessitam dos dados para a realização do aprendizado e classificação, fica ainda mais evidente.

A abordagem multitarefa, por sua vez, possui muitas das características da metodologia anterior, como, por exemplo, a aplicação da representação das séries temporais em imagens, e a possibilidade de promover a privacidade dos dados dos clientes. Um ponto positivo na técnica é que foi possível obter o valor de acurácia para todas as bases através de um único processo federado. É importante notar, porém, que apesar da maioria das bases possuírem resultado de acurácia semelhante à abordagem com particionamento dos dados, em algumas delas esse resultado ficou consideravelmente abaixo, indicando a necessidade de melhorias na técnica para que seja possível obter melhores resultados de acurácia.

Por fim, os resultados dos métodos propostos serão, também, comparados com métodos de Estado da Arte, relacionados com os métodos propostos. Os métodos comparados foram: Rocket (DEMPSTER; PETITJEAN; WEBB, 2020), ResNet (WANG; YAN; OATES, 2017), e CCN F (Complex Network Features) (LI; JIA; WAN, 2022). As bases a serem comparadas foram aquelas capazes de serem encontradas em todos os trabalhos de

Estado da Arte, utilizados para comparação. A Tabela 10 apresenta a comparação entre os métodos de Estado da Arte selecionados. Para se ter uma perspectiva do desempenho dos métodos de Estado da Arte e dos métodos propostos frente aos métodos de *benchmark*, esses também foram inclusos na tabela de comparação.

Ao analisar a Tabela 10, percebe-se que os métodos baseados em medidas de similaridade, aplicados a séries temporais fuzzy apresentaram resultado médio similar aos resultados de *benchmark*. Sua acurácia é superior à distância Euclideana e ligeiramente superior ao DTW com janelamento igual a 100 - 0,758, 0,776 para distância Euclideana e 0,776 para o DTW(100) - contra 0,785 para os métodos MSFTS. Porém, sua acurácia é ligeiramente inferior ao método DTW com janelamento treinado, o qual atingiu 0,794. Isso indica que esses métodos ainda precisam ser otimizados, tendo, por exemplo, seus números de grupos fuzzy ajustados para cada base de dados, uma vez que esse parâmetro é mantido fixo igual a 10. Pode-se notar, também, que em determinadas bases de dados, a acurácia dos métodos MSFTS foram ligeiramente superiores aos demais métodos de *benchmark*, indicando que, o processo de fuzzificação da série temporal pode ser capaz de filtrar alguns ruídos inerentes à série temporal original, aumentando a acurácia de classificação.

A comparação dos métodos MSFTS com os métodos de Estado da Arte já não é tão satisfatória quanto a comparação com métodos de *benchmark*. Percebe-se uma diferença considerável entre os melhores resultados em relação aos métodos de medidas de similaridade aplicados a FTS. Essa diferença, porém, se mantém ao se comparar os métodos de Estado da Arte com os métodos de *benchmark*, o que indica que a metodologia possui suas limitações, apesar de apresentarem acurácias superiores a diversos outros métodos por muito tempo, sendo, por isso consideradas como estratégias de *Baseline*.

Essas análises, portanto, indicam que, melhorias necessitam ser aplicadas aos métodos de medida de similaridade aplicadas a séries temporais fuzzy, porém, as limitações do método podem limitar a acurácia final. É importante pontuar, porém, que o desenvolvimento dessa técnica serviu de base para a proposta de outros trabalhos futuros os quais podem, tanto apresentar maiores acurácias, quanto auxiliar na resolução de problemas relacionados à proteção de dados. Como exemplos desses trabalhos, pode-se citar:

- a utilização do conceito de séries temporais fuzzy para a transformação da série temporal em uma série linguística e o posterior treinamento e classificação da série temporal linguística resultante através de métodos de processamento de linguagens, *Transformers* ou *Large Language Models - LLM*;
- a utilização do conceito de FTS discutido para discretizar a série temporal através de seus grupos fuzzy e aplicar *Locality Sensitive Hashing - LSH* para a geração de uma nova série temporal criptografada, para a posterior classificação.

Tabela 10 – Comparação dos métodos propostos com modelos de referência e métodos de Estado da Arte

Base	$\mathcal{L}_2$	DTW(w)	Rocket	Resnet	CCNF	MSFTS	CNN	FL Server	FL Multitask
Beef	0,667	0,667(0)	0,833	0,753	0,567	0,667	0,657	0,667	0,367
CBF	0,852	0,997(100)	1.00	0,995	0,894	0,998	0,988	0,991	0,332
DistalPhalanxOutlineAgeGroup	0,626	0,770(100)	0,755	0,716	0,833	0,712	0,708	0,741	0,467
ECG200	0,880	0,880(0)	0,906	0,874	0,846	0,900	0,846	0,810	0,800
FaceFour	0,784	0,886(2)	0,975	0,954	0,887	0,784	0,724	0,954	0,409
Ham	0,600	0,600(0)	0,726	0,757	0,714	0,590	0,683	0,651	0,638
MoteStrain	0,879	0,866(1)	0,914	0,928	0,858	0,822	0,884	0,881	0,819
SonyAIBORobotSurface1	0,695	0,725(100)	0,924	0,958	0,880	0,729	0,776	0,742	0,429
SonyAIBORobotSurface2	0,859	0,859(0)	0,916	0,979	0,869	0,849	0,836	0,844	0,617
TwoLeadECG	0,747	0,904(100)	0,999	1.00	0,948	0,804	0,825	0,816	0,708
Average Accuracy	0,758	0,815	0,894	0,891	0,829	0,785	0,813	0,809	0,559

Ao se analisar os resultados obtidos com a estratégia de representação da série temporal em imagens e classificados por CNN, percebe-se melhores resultados. A acurácia média obtida por esse método 0,813, supera, mesmo que ligeiramente, todos os métodos de *benchmark*, além de apresentar resultados competitivos com as demais estratégias de Estado da Arte. Esses resultados promissores indicam possibilidades de melhoria para o método de representação por imagens proposto. É possível notar, em algumas bases, que a técnica foi capaz de capturar, tanto padrões temporais quanto espaciais, evidenciado pela alta acurácia obtida em bases como CBF e MoteStrain, se igualando ou mesmo ultrapassando alguns métodos de Estado da Arte. Esses padrões não foram observados, porém, em bases cujo valor de acurácia foi relativamente baixo, como SonyAIBORobotSurface1 ou TwoLeadECG, indicando que novas estruturas de CNN, ou novas abordagens de representação podem auxiliar nessa tarefa.

Outra hipótese que afeta a acurácia do método CNN é referente ao número de imagens geradas, utilizadas para o treinamento e classificação. Caso esse número não seja suficiente para treinamento eficiente da CNN utilizada, cuja estrutura é relativamente simples (baseada na rede LeNet), seu desempenho pode ser degradado, principalmente ao ser comparado com os métodos de Estado da Arte. Portanto, a metodologia de classificação por CNN atrelado à representação da série temporal como imagem pode apresentar valores mais baixos em bases como Beef, onde o número de instâncias de treinamento é limitado.

Para melhorar os resultados obtidos com a estratégia de classificação centralizada, baseada em imagens, algumas abordagens, como por exemplo, a utilização de estruturas mais complexas de CNN. Outra abordagem seria basear-se em princípios dos métodos de Estado da Arte, como *Contrastive Learning*, ou ainda a aplicação de *Data Augmentation*. Outro ponto importante a ser citado sobre essa metodologia é sua flexibilidade de utilização. Como exemplo dessa flexibilidade, ela capacidade de ser facilmente adaptada para classificação de séries temporais multivariáveis, bastando, apenas, adicionar as novas séries como canais de imagens. Outra aplicação é referente a sua fácil adaptabilidade para utilização em ambientes descentralizados, a qual foi, é, inclusive, escopo do presente trabalho, e cujos resultados serão abordados a seguir.

Comparando-se o método FL Server, com particionamento dos dados de treinamento, com os demais métodos de *benchmark* e Estado da Arte, esse método atingiu uma acurácia média de 0,809, pouco acima do método DTW com janelamento otimizado, DTW(w) - 0,794 - e superando os demais métodos de *benchmark*, $\mathcal{L}_2$ e DTW (100), com 0,753 e 0,767, respectivamente. Comparado com os métodos de Estado da Arte, como Rocket e ResNet, o modelo apresenta desempenho inferior. Esses métodos obtiveram acurácias médias significativamente maiores, de 0,894 e 0,891, respectivamente, seguidos pelo CCNF com 0,830. Essa diferença de desempenho evidencia os desafios inerentes aos ambientes de aprendizado descentralizado, nos quais os modelos devem agregar conhecimento a partir

de dados distribuídos — frequentemente heterogêneos — sem acesso direto ao conjunto completo de dados. Enquanto os métodos centralizados de Estado da Arte se beneficiam do acesso integral aos dados, os modelos FL precisam lidar com particionamento dos dados, restrições de privacidade e distribuições não-IID, o que pode limitar seu desempenho.

Apesar desses desafios, o modelo FL Server apresenta forte desempenho em determinadas bases de dados. Por exemplo, atinge 0,991 de acurácia na base CBF e 0,954 na base FaceFour, se aproximando bastante, ou mesmo ultrapassando, o desempenho de alguns métodos de Estado da Arte. Porém, em bases como Ham (0,651) e SonyAIBORobotSurface1 (0,742), o desempenho do modelo foi inferior ao de Rocket, ResNet e CCNF, evidenciando variações de desempenho entre as diferentes bases. Esses resultados refletem as complexidades dos ambientes federados, nos quais a diversidade e a distribuição dos dados impactam diretamente a efetividade do modelo.

A configuração atual do FL Server utiliza apenas três clientes, de tal maneira que, para manter a consistência durante a agregação do modelo, todos empregam a mesma arquitetura de CNN. Essa uniformidade facilita atualizações globais eficazes, mas limita a capacidade de aproveitar a diversidade dos dados federados. Em cenários reais, os clientes podem possuir conjuntos de dados com distribuições únicas, e personalizar a arquitetura dos modelos para se adequar aos dados locais pode melhorar tanto o aprendizado local quanto o desempenho global. Além disso, a estratégia de agregação adotada neste estudo baseia-se em na estratégia FedAVG, que, embora eficaz para a convergência, pode não ser suficiente para lidar com a natureza não-IID dos dados dos clientes. Essa limitação pode comprometer a capacidade do modelo global de generalizar de forma eficaz entre diferentes distribuições de dados, abrindo espaço para otimizações no tratamento da heterogeneidade e no aproveitamento da colaboração inerente ao FL.

Apesar de o modelo FL Server ainda não ultrapassar a acurácia dos métodos de Estado da Arte, ele oferece vantagens distintas além das métricas de desempenho por si só. Ao preservar a privacidade dos dados e permitir o treinamento em conjuntos de dados descentralizados, o modelo aborda desafios críticos em cenários nos quais a centralização dos dados é inviável devido a restrições regulatórias e éticas. Essas características de preservação da privacidade, aliadas à capacidade de treinamento colaborativo entre múltiplos clientes, tornam o modelo FL uma solução promissora para aplicações reais em que abordagens centralizadas não são viáveis.

O método FL Multitask, por sua vez, apresentou resultados não muito competitivos ao serem comparados com *baselines* e com métodos de Estado da Arte. Esse fato indica que o modelo gerado pela abordagem não foi capaz de generalizar as características de todas as bases para que fosse possível se obter valores de acurácia competitivos. Porém, ao ser comparado com metodologias que utilizam abordagens semelhantes, como o agregador FedAvg na Tabela 8, esses resultados são melhores, indicando que a abordagem

é promissora.

Algumas melhorias podem ser consideradas para aumentar o desempenho dos modelos FL propostos e reduzir a diferença em relação aos métodos de Estado da Arte centralizados. A personalização das arquiteturas dos modelos com base nas distribuições específicas dos dados locais pode melhorar significativamente o aprendizado local e, por consequência, a acurácia do modelo global. Além disso, a incorporação de técnicas de agregação avançadas, como FedProx ou SCAFFOLD, pode mitigar os desafios causados pela heterogeneidade dos dados e pelo desvio entre clientes, promovendo uma melhor convergência e um equilíbrio mais sólido entre o aprendizado global e local.

Embora os modelos FL propostos ainda não ultrapassem as técnicas de Estado da Arte centralizadas, sua escalabilidade, capacidade de preservar a privacidade e adaptabilidade a ambientes distribuídos ressaltam seu potencial como uma alternativa robusta para aplicações do mundo real em que soluções centralizadas são impraticáveis. Com ajustes adicionais, os modelos têm potencial para alcançar desempenho competitivo aliado à viabilidade prática em cenários federados.

Outra vertente de grande importância ao se comparar metodologias é a análise de complexidade de tempo dos algoritmos, a fim de se entender o impacto das propostas nesse parâmetro. O contexto do trabalho foi focado na proposta de novas metodologias de classificação de séries temporais, baseadas em séries temporais fuzzy e aprendizado profundo. Porém, o principal foco foi agregar a possibilidade de manter privacidade de dados, agregando metodologias baseadas em aprendizado federado. Nos trabalhos subsequentes, espera-se poder, também, agregar novas metodologias para melhorar acurácia dos métodos propostos, bem como realizar uma análise de complexidade de tempo.

7 COMENTÁRIOS E CONCLUSÕES

Nessa seção são apresentados os principais comentários e conclusões acerca da tese como um todo, bem como dos resultados obtidos com as estratégias propostas nos capítulos 3, 4 e 5.

O principal objetivo da pesquisa realizada foi estudo e proposta de técnicas de classificação de séries temporais utilizando conceitos de séries temporais fuzzy e aprendizado profundo. Em suma, quatro metodologias foram propostas para a tarefa de classificação de séries temporais. Em todas as quatro metodologias propostas foi possível aplicar abordagem de séries temporais fuzzy com o intuito de obter uma melhor acurácia de classificação. Em três delas, além da utilização das séries temporais fuzzy, aplicou-se o conceito de *Deep Learning* e representação baseada em imagens na classificação. Além disso, as últimas abordagens propostas apresentam conceitos de privacidade de dados, conceitos esses cada vez mais necessários e aplicados em diversos contextos, quando se refere à classificação de séries temporais.

No que concerne ao processo de classificação baseado em medidas de similaridade aplicadas a séries temporais fuzzy, essa técnica apresentou resultados comparáveis aos métodos de *benchmark* ao longo da pesquisa. Em muitos casos para as bases de dados analisadas, a acurácia obtida para essa abordagem sobressaiu em relação a esses métodos. É interessante notar que o trabalho apresentou uma forma de classificação não abordada anteriormente por pesquisas referentes ao tema, sendo, portanto, uma porta de entrada para a classificação de séries temporais utilizando séries temporais fuzzy, normalmente aplicadas em tarefas de previsão. Um ponto positivo encontrado nessa abordagem é que, para alguns métodos o tempo de execução do algoritmo até a obtenção do resultado é longo, principalmente para as técnicas baseadas em distância de edição (LCS). Como a análise de complexidade de execução foge ao escopo do presente trabalho, os resultados de acurácia obtidos não apresentam os dados de tempo de execução. Essa tarefa, porém, pode ser sugerida como trabalhos futuros, com o intuito de se realizar a estimativa e comparação da complexidade de tempo atrelada à aplicação de séries temporais fuzzy na tarefa de classificação de séries temporais.

No que tange aos métodos propostos baseados em imagens, o principal objetivo é o desenvolvimento de novas técnicas de representação para classificação de séries temporais, integrando *deep learning* para a classificação da representação da série numérica e fuzzificada. Em resumo, dois métodos distintos de representação de imagens foram introduzidos para TSC, aproveitando o aprendizado profundo e a classificação baseada em imagens, a fim de preencher lacunas existentes na literatura. O desempenho das abordagens propostas foi avaliado em comparação com metodologias de referência e com o

estado da arte. Os resultados apresentados destacaram a eficácia e validade das técnicas propostas em relação aos métodos de *benchmark*. Um ponto notado ao longo da pesquisa é a influência da arquitetura da rede nos resultados obtidos. Para se conseguir os melhores resultados, várias arquiteturas de rede foram testadas, sendo algumas delas semelhantes, com o mesmo número de camadas, modificando-se o tamanho dos filtros, e outras com arquiteturas diferentes, modificando-se as camadas e os filtros. Esse fato indica que existe uma possibilidade de se encontrar uma arquitetura capaz de gerar resultados ótimos, a qual pode ser obtida para cada base de dados, ou para o processo de classificação como um todo, dependendo do tempo disponível para a otimização, que pode ser considerável.

Por fim, foram propostas duas abordagens descentralizadas, baseadas em aprendizado federado (FL), para a classificação de séries temporais univariadas. Os métodos integram FL com o modelo centralizado de redes neurais convolucionais (CNNs), abordado anteriormente. Na primeira delas, o processo começa com a distribuição de Dirichlet, que particiona as séries temporais de entrada entre os clientes. Cada cliente, então, treina um modelo local, no qual uma das três representações baseadas em imagens é selecionada para converter as séries temporais em imagens que são usadas como entrada da CNN. Após o treinamento local, os clientes compartilham suas atualizações de modelo (pesos) com o servidor central. O servidor agrega essas atualizações, distribui o modelo global atualizado de volta aos clientes e avalia seu desempenho em um conjunto de teste. Esse processo iterativo continua por um número predefinido de rodadas até que a convergência seja alcançada. Os resultados demonstram que o método proposto é competitivo e promissor em comparação com métodos de referência e de estado da arte. No entanto, há espaço para melhorias, especialmente no aprimoramento da acurácia do modelo.

A segunda técnica FL proposta é baseada na classificação multitarefa, na qual um único modelo é utilizado para classificar todas as bases de dados, provenientes de diferentes áreas. Uma característica desses dados é sua heterogeneidade, uma vez que, para se realizar a classificação multitarefa, a origem dos dados é diversa. De posse dos dados, os quais já caracterizam os clientes por si só, é realizado o treinamento local em cada cliente a fim de se obter os parâmetros do modelo. Uma vez realizado o treinamento local, os parâmetros são enviados ao servidor, o qual agrega essas atualizações dos clientes. Apenas os pesos das camadas intermediárias são compartilhados, sendo as camadas totalmente conectadas e camada de saída específicas para cada cliente. Os resultados apresentam limitações tanto na agregação quando na capacidade de generalização do modelo, cujos resultados foram relativamente baixos ao serem comparados com métodos de *baseline* e de estado da arte. Porém, ao serem comparados com métodos que utilizam abordagens semelhantes, esses resultados se tornam promissores, evidenciando a necessidade de melhorias e a possibilidade de obtenção de maiores acurácias.

Tendo em vista as limitações e possibilidades de melhoria ou trabalhos futuros

dos modelos propostos, esse assunto será discutido a seguir.

7.1 Limitações e trabalhos futuros

Embora as abordagens propostas tenham apresentado resultados semelhantes aos métodos de *benchmark*, sendo alguns deles com desempenho promissor em comparação com os métodos de estado da arte, melhorias adicionais ainda são necessárias. Assim, direções futuras pretendem aprimorar o desempenho dos métodos propostos através de inclusão de técnicas de otimização, além de aplicar metodologias que se inspiram nas técnicas de estado da arte. A presente seção apresenta algumas das principais limitações referentes a esses métodos, bem como propostas de trabalhos futuros que podem ser focados, tanto na melhoria dos métodos, quanto no desenvolvimento de novas metodologias, inspiradas nesses métodos.

Em relação aos métodos baseados em medidas de similaridade aplicadas a séries temporais fuzzy, utilizou-se um número fixo de grupos de pertinência para a fuzzificação da série temporal, o que pode acarretar em desempenhos inferiores para algumas bases. Propõe-se abordagens para otimização do número de grupos fuzzy a serem utilizados, a fim de encontrar melhores valores de acurácia de classificação. Além disso, pode-se aplicar outras técnicas de particionamento do universo de discurso para o processo de fuzzificação da série temporal.

Apesar das limitações do método, os quais necessitam ser trabalhadas para se obter melhor acurácia, a fuzzificação da série temporal para a geração de uma série fuzzificada gera novas oportunidades de pesquisa. Uma vez obtida a série linguística, ela pode ser utilizada como entrada em modelos de processamento de texto, como os *Transformers* e os *Large Language Models* (LLM), os quais estão sendo objeto de pesquisa em diversos trabalhos. Outra oportunidade está relacionada à aplicação da abordagem de fuzzificação da série temporal na segurança de dados. Isso pode ser obtido com utilização dessa da série discretizada por grupos fuzzy em algoritmos como *Locality Sensitive Hashing*.

No que tange aos métodos de representação de imagens, um primeiro ponto de melhoria é referente ao número de grupos fuzzy utilizados para a fuzzificação da série temporal, semelhante ao que foi observado para os métodos baseados em medida de similaridade. Além disso, no que se refere à rede CNN como um todo, pode-se focar na melhoria de sua arquitetura, buscando-se por melhores resultados através da aplicação de redes mais complexas, capazes de realizar uma melhor extração de características. Outra vertente é o desenvolvimento de um modelo que possa determinar automaticamente a estrutura ideal da CNN com base nos conjuntos de dados analisados, a fim de abordar a influência da arquitetura da CNN no desempenho do modelo. Além disso, o método proposto tem excelente capacidade de ser facilmente adaptado para aplicações multivariadas

de classificação de séries temporais. Sendo assim, novos trabalhos podem ser direcionados a esse tipo de modelo CNN para classificar séries temporais multivariadas.

No que tange aos métodos baseados em Aprendizado Federado, por serem versões distribuídas do método baseado em imagens, aquelas citadas para o método podem ser, também, consideradas para os modelos FL. Além disso, os seguintes pontos podem ser elencados como limitações das propostas:

1. **Número limitado de clientes:** A configuração do método que particiona os dados de treinamento utiliza apenas três clientes, o que restringe a capacidade do modelo de aprender a partir de uma gama mais ampla de distribuições de dados. Um conjunto mais diversificado e maior de clientes poderia enriquecer o ambiente federado, ajudando o modelo a generalizar melhor entre padrões variados e reduzindo o sobreajuste a características locais dos dados.
2. **Arquiteturas uniformes entre os conjuntos de dados:** O uso da mesma arquitetura de CNN para todos os conjuntos de dados para ambas as abordagens limita a flexibilidade do modelo em se adaptar às características e distribuições específicas de cada conjunto de dados.
3. **Estratégia básica de agregação:** O método de agregação baseia-se em uma média simples, o que pode não ser eficaz para lidar com a heterogeneidade dos dados entre os clientes.
4. **Melhorias no aprendizado multitarefa:** As abordagens propostas exploram de maneira limitada o potencial de frameworks de aprendizado multitarefa. Incorporar novas estratégias de agregação nessa metodologia permitiria melhor desempenho do modelo em aprender representações compartilhadas entre os clientes, ao mesmo tempo em que atende tarefas específicas de cada cliente. Essa abordagem é de grande importância, pois leva a uma melhor generalização e redução das lacunas de desempenho em relação aos métodos centralizados, e estudar novas tecnologias capazes de serem incorporadas a ela podem agregar bastante na classificação com aprendizado federado.

Com o intuito de melhorar resultados de acurácia, ou mesmo propor pesquisas em novas abordagens, direções futuras podem contemplar as seguintes ações:

1. Ampliar o número de clientes participantes dos experimentos.
2. Otimizar as arquiteturas de CNN para se adequar de forma mais eficaz às características específicas de cada conjunto de dados.

3. Explorar a substituição da CNN por técnicas de Estado da Arte, como mecanismos de atenção, transformers e suas variantes.
4. Realizar os experimentos utilizando técnicas de agregação como FedProx ou SCAFFOLD.
5. Transitar de técnicas centralizadas de FL para abordagens descentralizadas, visando maior escalabilidade e resiliência.
6. Incorporar técnicas de preservação da privacidade, como privacidade diferencial, para reforçar a segurança e a confidencialidade dos dados.

Por fim, um ponto ainda não tratado no presente trabalho, porém também importante, é a análise de complexidade de tempo das metodologias propostas, comparadas com métodos de *baseline* e estado da arte. Propõe-se como continuidade do presente trabalho uma análise da complexidade de tempo dos métodos propostos, comparados com metodologias de estado da arte. Essa análise pode indicar mais pontos positivos ou negativos entre os métodos propostos e as demais metodologias.

7.2 Publicações

Durante o desenvolvimento da tese, algumas publicações de trabalhos foram desenvolvidas, os quais se relacionam com os assuntos abordados.

7.2.1 Publicação em revista

1. F. A. Silva, O. Orang, F. J. Erazo-Costa, P. C. Silva, P. H. Barros, R. P. Ferreira, and F. G. Guimarães. Time series classification using federated convolutional neural networks and image-based representations. IEEE Access, 2025.

7.2.2 Publicações em congressos

1. O. Orang, F. A. da Silva, P. C. Silva, P. H. Barros, H. S. Ramos, and F. G. Guimarães. Traffic forecasting using federated randomized high-order fuzzy cognitive maps. In Brazilian Conference on Intelligent Systems, pages 445–459. Springer, 2024.
2. P. H. Barros, M. Q. Oliveira, O. Orang, F. A. da Silva, F. J. Erazo-Costa, A. T. Bastos, P. C. Silva, G. S. dos Santos, A. A. Loureiro, M. G. Ravetti, et al. Flautim: A federated learning platform using k8s and flower. In Simpósio Brasileiro de Sistemas Multimídia e Web (WebMedia), pages 87–90. SBC, 2024.

3. Petronio C. L. Silva, Omid Orang, Felipe A. R. Silva, Pedro H. Barros, Heitor S. Ramos and Frederico Gadelha Guimaraes. Driver Maneuver Classification Based on Multivariate Fuzzy Time Series. In 2025 IEEE International Conference on Fuzzy Systems. 2025.
4. Omid Orang, Patrícia O. Lucas, Gabriel I. F. Paiva, Petrônio C. L. Silva, Felipe A. R. Silva, Adriano A. Veloso, Frederico G. Guimarães. Causal Graph Fuzzy LLMs: A First Introduction and Applications in Time Series Forecasting. In XVII Congresso Brasileiro de Inteligência Computacional. IEEE, 2025.

REFERÊNCIAS

- AGRAWAL, R.; FALOUTSOS, C.; SWAMI, A. Efficient similarity search in sequence databases. In: SPRINGER. *International conference on foundations of data organization and algorithms*. [S.l.], 1993. p. 69–84.
- AL-MAHASNEH, A. J.; ANAVATTI, S. G.; GARRATT, M. A. The development of neural networks applications from perceptron to deep learning. In: IEEE. *2017 International Conference on Advanced Mechatronics, Intelligent Manufacture, and Industrial Automation (ICAMIMIA)*. [S.l.], 2017. p. 1–6.
- ALEDHARI, M. et al. Federated learning: A survey on enabling technologies, protocols, and applications. *IEEE Access*, IEEE, 2020.
- ASKARI, S.; MONTAZERIN, N. A high-order multi-variable fuzzy time series forecasting algorithm based on fuzzy clustering. *Expert Systems with Applications*, Elsevier, v. 42, n. 4, p. 2121–2135, 2015.
- ÅSTRÖM, K. J. On the choice of sampling rates in parametric identification of time series. *Information Sciences*, Elsevier, v. 1, n. 3, p. 273–278, 1969.
- AZAR, J. et al. Robust iot time series classification with data compression and deep learning. *Neurocomputing*, Elsevier, v. 398, p. 222–234, 2020.
- BAGNALL, A. et al. Transformation based ensembles for time series classification. In: SIAM. *Proceedings of the 2012 SIAM international conference on data mining*. [S.l.], 2012.
- BAGNALL, A. et al. The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances. *Data mining and knowledge discovery*, Springer, v. 31, n. 3, p. 606–660, 2017.
- BAGNALL, A. et al. Time-series classification with cote: the collective of transformation-based ensembles. *IEEE Transactions on Knowledge and Data Engineering*, IEEE, v. 27, n. 9, p. 2522–2535, 2015.
- BAILLY, A. *Time Series Classification Algorithms with Applications in Remote Sensing*. Tese (Doutorado) — Université Rennes 2, 2018.
- BANABILAH, S. et al. Federated learning review: Fundamentals, enabling technologies, and future applications. *Information processing & management*, Elsevier, 2022.
- BARROS, P. H. et al. A novel federated meta-learning approach for discriminating sedentary behavior from wearable data. *IEEE Internet of Things Journal*, 2024.
- BAS, E. et al. Fuzzy-time-series network used to forecast linear and nonlinear time series. *Applied Intelligence*, Springer, v. 43, n. 2, p. 343–355, 2015.
- BATISTA, G. E.; WANG, X.; KEOGH, E. J. A complexity-invariant distance measure for time series. In: SIAM. *Proceedings of the 2011 SIAM international conference on data mining*. [S.l.], 2011. p. 699–710.

- BAYDOGAN, M. G.; RUNGER, G.; TUV, E. A bag-of-features framework to classify time series. *IEEE transactions on pattern analysis and machine intelligence*, IEEE, v. 35, n. 11, p. 2796–2802, 2013.
- BERNDT, D. J.; CLIFFORD, J. Using dynamic time warping to find patterns in time series. In: SEATTLE, WA, USA: *KDD workshop*. [S.l.], 1994. v. 10, n. 16, p. 359–370.
- BONAWITZ, K. et al. Federated learning and privacy: Building privacy-preserving systems for machine learning and data science on decentralized data. *Queue*, ACM New York, NY, USA, 2021.
- CARUANA, R.; BALUJA, S.; MITCHELL, T. Using the future to “sort out” the present: Rankprop and multitask learning for medical risk evaluation. *Advances in neural information processing systems*, v. 8, 1995.
- CHAN, K.-P.; FU, A. W.-C. Efficient time series matching by wavelets. In: IEEE. *Proceedings 15th International Conference on Data Engineering (Cat. No. 99CB36337)*. [S.l.], 1999. p. 126–133.
- CHATFIELD, C. *The Analysis of Time Series: An Introduction, Sixth Edition*. CRC Press, 2016. (Chapman & Hall/CRC Texts in Statistical Science). ISBN 9780203491683. Disponível em: <https://books.google.com.br/books?id=qKzyAbdaDFAC>.
- CHELLAPANDI, V. P. et al. Federated learning for connected and automated vehicles: A survey of existing approaches and challenges. *IEEE Transactions on Intelligent Vehicles*, 2023.
- CHEN, L.; NG, R. On the marriage of lp-norms and edit distance. In: *Proceedings of the Thirtieth international conference on Very large data bases-Volume 30*. [S.l.: s.n.], 2004. p. 792–803.
- CHEN, L.; ÖZSU, M. T.; ORIA, V. Robust and fast similarity search for moving object trajectories. In: *Proceedings of the 2005 ACM SIGMOD international conference on Management of data*. [S.l.: s.n.], 2005. p. 491–502.
- CHEN, M.-Y. A high-order fuzzy time series forecasting model for internet stock trading. *Future Generation Computer Systems*, Elsevier, v. 37, p. 461–467, 2014.
- CHEN, S.-M. et al. Forecasting enrollments based on fuzzy time series. *Fuzzy sets and systems*, v. 81, n. 3, p. 311–319, 1996.
- CHENG, C.-H.; CHANG, J.-R.; YEH, C.-A. Entropy-based and trapezoid fuzzification-based fuzzy time series approaches for forecasting it project cost. *Technological Forecasting and Social Change*, Elsevier, v. 73, n. 5, p. 524–542, 2006.
- COOLEY, J. W.; TUKEY, J. W. An algorithm for the machine calculation of complex fourier series. *Mathematics of computation*, JSTOR, v. 19, n. 90, p. 297–301, 1965.
- DAS, G.; GUNOPULOS, D.; MANNILA, H. Finding similar time series. In: SPRINGER. *European Symposium on Principles of Data Mining and Knowledge Discovery*. [S.l.], 1997. p. 88–100.
- DAU, H. A. et al. *The UCR Time Series Classification Archive*. 2018. https://www.cs.ucr.edu/~eamonn/time_series_data_2018/.

- DEMPSTER, A.; PETITJEAN, F.; WEBB, G. I. Rocket: exceptionally fast and accurate time series classification using random convolutional kernels. *Data Mining and Knowledge Discovery*, Springer, 2020.
- DURU, O.; YOSHIDA, S. Modeling principles in fuzzy time series forecasting. In: IEEE. *2012 IEEE Conference on Computational Intelligence for Financial Engineering & Economics (CIFEr)*. [S.l.], 2012. p. 1–7.
- ECKMANN, J.-P. et al. Recurrence plots of dynamical systems. *World Scientific Series on Nonlinear Science Series A*, World Scientific Publishing, v. 16, p. 441–446, 1995.
- EFENDI, R.; ISMAIL, Z.; DERIS, M. M. Improved weight fuzzy time series as used in the exchange rates forecasting of us dollar to ringgit malaysia. *International Journal of Computational Intelligence and Applications*, World Scientific, v. 12, n. 01, p. 1350005, 2013.
- ESLING, P.; AGON, C. Time-series data mining. *ACM Computing Surveys (CSUR)*, ACM New York, NY, USA, v. 45, n. 1, p. 1–34, 2012.
- FALOUTSOS, C.; RANGANATHAN, M.; MANOLOPOULOS, Y. Fast subsequence matching in time-series databases. *ACM Sigmod Record*, ACM New York, NY, USA, v. 23, n. 2, p. 419–429, 1994.
- FAOUZI, J. Time series classification: A review of algorithms and implementations. *Machine Learning (Emerging Trends and Applications)*, 2022.
- FAWAZ, H. I. et al. Deep learning for time series classification: a review. *Data Mining and Knowledge Discovery*, Springer, v. 33, n. 4, p. 917–963, 2019.
- FU, T.-c. A review on time series data mining. *Engineering Applications of Artificial Intelligence*, Elsevier, v. 24, n. 1, p. 164–181, 2011.
- GAO, D. et al. Hhhfl: Hierarchical heterogeneous horizontal federated learning for electroencephalography. *ArXiv*, 2019.
- GÉRON, A. *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems*. [S.l.]: "O'Reilly Media, Inc.", 2019.
- GRABOCKA, J. et al. Learning time-series shapelets. In: *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. [S.l.: s.n.], 2014. p. 392–401.
- GUENDOUZI, B. S. et al. A systematic review of federated learning: Challenges, aggregation methods, and development tools. *Journal of Network and Computer Applications*, Elsevier, 2023.
- GUO, Y. et al. Deep learning for visual understanding: A review. *Neurocomputing*, Elsevier, v. 187, p. 27–48, 2016.
- HARUTYUNYAN, H. et al. Multitask learning and benchmarking with clinical time series data. *Scientific data*, Nature Publishing Group UK London, v. 6, n. 1, p. 96, 2019.

- HUANG, D.; LV, X.; ZHANG, Y. Msnet: Multi-task self-supervised network for time series classification. *Pattern Recognition Letters*, Elsevier, v. 191, p. 73–79, 2025.
- ISMAIL, Z.; EFENDI, R. Enrollment forecasting based on modified weight fuzzy time series. *Journal of Artificial Intelligence*, v. 4, n. 1, p. 110–118, 2011.
- ISMAIL, Z.; EFENDI, R.; DERIS, M. M. Application of fuzzy time series approach in electric load forecasting. *New Mathematics and Natural Computation*, World Scientific, v. 11, n. 03, p. 229–248, 2015.
- ITAKURA, F. Minimum prediction residual principle applied to speech recognition. *IEEE Transactions on acoustics, speech, and signal processing*, IEEE, v. 23, n. 1, p. 67–72, 1975.
- JANG, J.-S. R.; SUN, E. M. C.-T. *Neuro-fuzzy and soft computing: a computational approach to learning and machine intelligence*. [S.l.]: Prentice-Hall, 1997.
- JEONG, Y.-S.; JEONG, M. K.; OMITAOMU, O. A. Weighted dynamic time warping for time series classification. *Pattern recognition*, Elsevier, v. 44, n. 9, p. 2231–2240, 2011.
- JIANG, W. Time series classification: Nearest neighbor versus deep learning models. *SN Applied Sciences*, Springer, v. 2, n. 4, p. 1–17, 2020.
- JR, J. C.; JR, C. C. Identification method for fuzzy forecasting models of time series. *Applied Soft Computing*, Elsevier, v. 50, p. 166–182, 2017.
- JURAFSKY, D.; MARTIN, J. H. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. 1st edition. ed. [S.l.]: Prentice Hall PTR, 2020.
- KARIMI-BIDHENDI, S.; MUNSHI, F.; MUNSHI, A. Scalable classification of univariate and multivariate time series. In: IEEE. *2018 IEEE International Conference on Big Data (Big Data)*. [S.l.], 2018. p. 1598–1605.
- KEOGH, E. Fast similarity search in the presence of longitudinal scaling in time series databases. In: IEEE. *Proceedings Ninth IEEE International Conference on Tools with Artificial Intelligence*. [S.l.], 1997. p. 578–584.
- KEOGH, E. et al. Locally adaptive dimensionality reduction for indexing large time series databases. In: *Proceedings of the 2001 ACM SIGMOD international conference on Management of data*. [S.l.: s.n.], 2001. p. 151–162.
- KEOGH, E.; KASETTY, S. On the need for time series data mining benchmarks: a survey and empirical demonstration. *Data Mining and knowledge discovery*, Springer, v. 7, n. 4, p. 349–371, 2003.
- KEOGH, E. et al. Lb.keogh supports exact indexing of shapes under rotation invariance with arbitrary representations and distance measures. In: CITESEER. *Proceedings of the 32nd international conference on Very large data bases*. [S.l.], 2006. p. 882–893.
- KEOGH, E. J.; PAZZANI, M. J. A simple dimensionality reduction technique for fast similarity search in large time series databases. In: SPRINGER. *Pacific-Asia conference on knowledge discovery and data mining*. [S.l.], 2000. p. 122–133.

- KEOGH, E. J.; PAZZANI, M. J. Derivative dynamic time warping. In: SIAM. *Proceedings of the 2001 SIAM international conference on data mining*. [S.l.], 2001. p. 1–11.
- KEOGH, E. J.; SMYTH, P. et al. A probabilistic approach to fast pattern matching in time series databases. In: *Kdd*. [S.l.: s.n.], 1997. v. 1997, p. 24–30.
- KLEIST, C. *Time series data mining methods*. Dissertação (Mestrado) — Humboldt-Universität zu Berlin, Wirtschaftswissenschaftliche Fakultät, 2015.
- KLIR, G.; YUAN, B. *Fuzzy sets and fuzzy logic*. [S.l.]: Prentice hall New Jersey, 1995. v. 4.
- KONEČNÝ, J. et al. Federated optimization: Distributed machine learning for on-device intelligence. *arXiv preprint arXiv:1610.02527*, 2016.
- KRIZHEVSKY, A.; SUTSKEVER, I.; HINTON, G. E. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 2012.
- KUMAR, B. P. et al. Evaluation of federated learning strategies on industrial time series fault classification. In: *Advances in Distributed Computing and Machine Learning: Proceedings of ICADCML 2023*. [S.l.]: Springer, 2023.
- LACASA, L. et al. From time series to complex networks: The visibility graph. *Proceedings of the National Academy of Sciences*, National Acad Sciences, 2008.
- LACASA, L.; NICOSIA, V.; LATORA, V. Network structure of multivariate time series. *Scientific reports*, Nature Publishing Group UK London, 2015.
- LECUN, Y.; BENGIO, Y.; HINTON, G. Deep learning. *nature*, Nature Publishing Group, v. 521, n. 7553, p. 436–444, 2015.
- LECUN, Y.; BENGIO, Y. et al. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, v. 3361, n. 10, p. 1995, 1995.
- LECUN, Y. et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, Ieee, v. 86, n. 11, p. 2278–2324, 1998.
- LEE, M.; SADAIEI, H. J.; SUHARTONO. Improving taie forecasting using fuzzy time series with box-cox power transformation. *Journal of Applied Statistics*, Taylor & Francis, v. 40, n. 11, p. 2407–2422, 2013.
- LEE, M. H.; JAVEDANI, H. et al. A weighted fuzzy integrated time series for forecasting tourist arrivals. In: SPRINGER. *International Conference on Informatics Engineering and Information Science*. [S.l.], 2011. p. 206–217.
- LEVENSHTEIN, V. I. et al. Binary codes capable of correcting deletions, insertions, and reversals. In: SOVIET UNION. *Soviet physics doklady*. [S.l.], 1966. v. 10, n. 8, p. 707–710.
- LI, H.; JIA, R.; WAN, X. Time series classification based on complex network. *Expert Systems with Applications*, Elsevier, 2022.
- LI, S.-T.; CHENG, Y.-C.; LIN, S.-Y. A fcm-based deterministic forecasting model for fuzzy time series. *Computers & Mathematics with Applications*, Elsevier, v. 56, n. 12, p. 3052–3063, 2008.

- LI, X. et al. On the convergence of fedavg on non-iid data. *arXiv preprint arXiv:1907.02189*, 2019.
- LIANG, Z.; WANG, H. Fedtsc: a secure federated learning system for interpretable time series classification. *Proc. VLDB Endow.*, VLDB Endowment, aug 2022.
- LIN, J. et al. A symbolic representation of time series, with implications for streaming algorithms. In: *Proceedings of the 8th ACM SIGMOD workshop on Research issues in data mining and knowledge discovery*. [S.l.: s.n.], 2003. p. 2–11.
- LIN, J.; KHADE, R.; LI, Y. Rotation-invariant similarity in time series using bag-of-patterns representation. *Journal of Intelligent Information Systems*, Springer, v. 39, n. 2, p. 287–315, 2012.
- LIN, R. A. K.-l.; SHIM, H. S. S. K. Fast similarity search in the presence of noise, scaling, and translation in time-series databases. In: CITESEER. *Proceeding of the 21th International Conference on Very Large Data Bases*. [S.l.], 1995. p. 490–501.
- LINES, J. et al. A shapelet transform for time series classification. In: *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*. [S.l.: s.n.], 2012. p. 289–297.
- LINES, J.; TAYLOR, S.; BAGNALL, A. Hive-cote: The hierarchical vote collective of transformation-based ensembles for time series classification. In: IEEE. *2016 IEEE 16th international conference on data mining (ICDM)*. [S.l.], 2016. p. 1041–1046.
- LINES, J.; TAYLOR, S.; BAGNALL, A. Time series classification with hive-cote: The hierarchical vote collective of transformation-based ensembles. *ACM Transactions on Knowledge Discovery from Data*, v. 12, n. 5, 2018.
- LU, D.; WENG, Q. A survey of image classification methods and techniques for improving classification performance. *International journal of Remote sensing*, Taylor & Francis, v. 28, n. 5, p. 823–870, 2007.
- LUCAS, P. O. et al. A tutorial on fuzzy time series forecasting models: recent advances and challenges. *Learn Nonlinear Models*, 2022.
- MAHARAJ, E. A.; D’URSO, P.; CAIADO, J. *Time series clustering and classification*. [S.l.]: Chapman and Hall/CRC, 2019.
- MARTEAU, P.-F. Time warp edit distance with stiffness adjustment for time series matching. *IEEE transactions on pattern analysis and machine intelligence*, IEEE, v. 31, n. 2, p. 306–318, 2008.
- MARTIN, S. *What’s the Difference Between a CNN and an RNN?* 2018. Edge AI and Vision Insights. Disponível em: <https://www.edge-ai-vision.com/2018/09/whats-the-difference-between-a-cnn-and-an-rnn/>.
- MCMAHAN, B. et al. Communication-efficient learning of deep networks from decentralized data. In: PMLR. *Artificial intelligence and statistics*. [S.l.], 2017.
- MCMAHAN, H. B. et al. Federated learning: Strategies for improving communication efficiency. In: *Proceedings of the 29th Conference on Neural Information Processing Systems (NIPS), Barcelona, Spain*. [S.l.: s.n.], 2016.

- MONTGOMERY, D. C. *Design and analysis of experiments*. [S.l.]: John wiley & sons, 2017.
- MONTGOMERY, D. C.; RUNGER, G. C. *Applied statistics and probability for engineers*. [S.l.]: John wiley & sons, 2010.
- MOTHUKURI, V. et al. A survey on security and privacy of federated learning. *Future Generation Computer Systems*, Elsevier, 2021.
- NATH, S. S. et al. A survey of image classification methods and techniques. In: IEEE. *2014 International conference on control, instrumentation, communication and computational technologies (ICCICCT)*. [S.l.], 2014. p. 554–557.
- NG, K. W.; TIAN, G.-L.; TANG, M.-L. *Dirichlet and related distributions: Theory, methods and applications*. [S.l.]: John Wiley & Sons, 2011.
- NGUYEN, D. C. et al. Federated learning for internet of things: A comprehensive survey. *IEEE Communications Surveys & Tutorials*, IEEE, 2021.
- NIELSEN, A. *Practical time series analysis: prediction with statistics and machine learning*. [S.l.]: "O'Reilly Media, Inc.", 2019.
- PANIGRAHI, S.; BEHERA, H. S. Fuzzy time series forecasting: a survey. *Computational Intelligence in Data Mining*, Springer, p. 641–651, 2020.
- PARAGLIOLA, G. Application of federated learning approaches for time-series classification in ehealth domain. *Procedia Computer Science*, 2022. Knowledge-Based and Intelligent Information & Engineering Systems: Proceedings of the 26th International Conference KES2022.
- PARAGLIOLA, G.; RIBINO, P. Exploring heterogeneous data distribution issues in e-health federated systems. *Biomedical Signal Processing and Control*, Elsevier, 2024.
- PEDRYCZ, W.; GOMIDE, F. *Fuzzy systems engineering: toward human-centric computing*. [S.l.]: John Wiley & Sons, 2007.
- PENHA, D. d. P. et al. Rede neural convolucional aplicada à identificação de equipamentos residenciais para sistemas de monitoramento não-intrusivo de carga. Universidade Federal do Pará, 2018.
- QI, P. et al. Model aggregation techniques in federated learning: A comprehensive survey. *Future Generation Computer Systems*, Elsevier, 2023.
- QI, P. et al. Model aggregation techniques in federated learning: A comprehensive survey. *Future Generation Computer Systems*, 2024.
- RAFIEI, D.; MENDELZON, A. Similarity-based queries for time series data. In: *Proceedings of the 1997 ACM SIGMOD international conference on Management of data*. [S.l.: s.n.], 1997. p. 13–25.
- RAKTHANMANON, T.; KEOGH, E. Fast shapelets: A scalable algorithm for discovering time series shapelets. In: SIAM. *proceedings of the 2013 SIAM International Conference on Data Mining*. [S.l.], 2013. p. 668–676.

- RANI, S. et al. Federated learning for secure iomt-applications in smart healthcare systems: A comprehensive review. *Knowledge-Based Systems*, Elsevier, 2023.
- REZENDE, T. M. Estudo de redes neurais convolucionais para o reconhecimento de sinais de libras. *Researchgate*, Universidade Federal de Minas Gerais, 2019. Disponível em: <http://rgdoi.net/10.13140/RG.2.2.20729.60000>.
- REZENDE, T. M. Reconhecimento automático de sinais da libras: desenvolvimento da base de dados minds-libras e modelos de redes convolucionais. Universidade Federal de Minas Gerais, 2021.
- SADAEI, H. J. et al. A hybrid model based on differential fuzzy logic relationships and imperialist competitive algorithm for stock market forecasting. *Applied Soft Computing*, Elsevier, v. 40, p. 132–149, 2016.
- SADAEI, H. J. et al. Short-term load forecasting method based on fuzzy time series, seasonality and long memory process. *International Journal of Approximate Reasoning*, Elsevier, v. 83, p. 196–217, 2017.
- SADAEI, H. J. et al. Short-term load forecasting by using a combined method of convolutional neural networks and fuzzy time series. *Energy*, Elsevier, v. 175, p. 365–377, 2019.
- SAKOE, H.; CHIBA, S. A similarity evaluation of speech patterns by dynamic programming. In: *Nat. Meeting of Institute of Electronic Communications Engineers of Japan*. [S.l.: s.n.], 1970. p. 136.
- SAKOE, H.; CHIBA, S. Dynamic programming algorithm optimization for spoken word recognition. *IEEE transactions on acoustics, speech, and signal processing*, IEEE, v. 26, n. 1, p. 43–49, 1978.
- SCHÄFER, P. The boss is concerned with time series classification in the presence of noise. *Data Mining and Knowledge Discovery*, Springer, v. 29, n. 6, p. 1505–1530, 2015.
- SCHERER, D.; MÜLLER, A.; BEHNKE, S. Evaluation of pooling operations in convolutional architectures for object recognition. In: SPRINGER. *International conference on artificial neural networks*. [S.l.], 2010. p. 92–101.
- SCHIMKE, S.; VIELHAUER, C.; DITTMANN, J. Using adapted levenshtein distance for on-line signature authentication. In: IEEE. *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004*. [S.l.], 2004. v. 2, p. 931–934.
- SENIN, P.; MALINCHIK, S. Sax-vsm: Interpretable time series classification using sax and vector space model. In: IEEE. *2013 IEEE 13th international conference on data mining*. [S.l.], 2013. p. 1175–1180.
- SHAHEEN, M. et al. Applications of federated learning; taxonomy, challenges, and research trends. *Electronics*, MDPI, v. 11, n. 4, p. 670, 2022.
- SHATKAY, H.; ZDONIK, S. B. Approximate queries and representations for large data sequences. In: IEEE. *Proceedings of the Twelfth International Conference on Data Engineering*. [S.l.], 1996. p. 536–545.

- SILVA, P. C. d. L. Scalable models for probabilistic forecasting with fuzzy time series. 2019.
- SILVA, V. F. et al. Time series analysis via network science: Concepts and algorithms. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, Wiley Online Library, 2021.
- SINGH, P. A brief review of modeling approaches based on fuzzy time series. *International Journal of Machine Learning and Cybernetics*, Springer, v. 8, n. 2, p. 397–420, 2017.
- SMYTH, P.; KEOGH, E. Clustering and mode classification of engineering time series data. In: CITESEER. *Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining*. [S.l.], 1997. p. 24–30.
- SONG, Q.; CHISSOM, B. S. Forecasting enrollments with fuzzy time series. ERIC, 1991.
- SONG, Q.; CHISSOM, B. S. Fuzzy time series and its models. *Fuzzy sets and systems*, Elsevier, v. 54, n. 3, p. 269–277, 1993.
- SONG, Q.; CHISSOM, B. S. Fuzzy time series and its models. *Fuzzy sets and systems*, v. 54, n. 3, p. 269–277, 1993.
- SONG, Q.; CHISSOM, B. S. Forecasting enrollments with fuzzy time series—part ii. *Fuzzy sets and systems*, Elsevier, v. 62, n. 1, p. 1–8, 1994.
- STEFAN, A.; ATHITSOS, V.; DAS, G. The move-split-merge metric for time series. *IEEE transactions on Knowledge and Data Engineering*, IEEE, v. 25, n. 6, p. 1425–1438, 2012.
- SUN, B. et al. Prediction of stock index futures prices based on fuzzy sets and multivariate fuzzy time series. *Neurocomputing*, Elsevier, v. 151, p. 1528–1536, 2015.
- SUPRIYA, S. et al. Weighted visibility graph with complex network features in the detection of epilepsy. *IEEE access*, IEEE, 2016.
- TALARPOSHTI, F. M. et al. Stock market forecasting by using a hybrid model of exponential fuzzy time series. *International Journal of Approximate Reasoning*, Elsevier, v. 70, p. 79–98, 2016.
- TRUEX, S. et al. A hybrid approach to privacy-preserving federated learning. In: *Proceedings of the 12th ACM workshop on artificial intelligence and security*. [S.l.: s.n.], 2019.
- VARGAS, A. C. G.; PAES, A.; VASCONCELOS, C. N. Um estudo sobre redes neurais convolucionais e sua aplicação em detecção de pedestres. In: *Proceedings of the XXIX Conference on Graphics, Patterns and Images*. [S.l.: s.n.], 2016. v. 1, n. 4.
- VLACHOS, M.; KOLLIOS, G.; GUNOPULOS, D. Discovering similar multidimensional trajectories. In: IEEE. *Proceedings 18th international conference on data engineering*. [S.l.], 2002. p. 673–684.
- WAGNER, R. A.; FISCHER, M. J. The string-to-string correction problem. *Journal of the ACM (JACM)*, ACM New York, NY, USA, v. 21, n. 1, p. 168–173, 1974.
- WANG, J. et al. Bag-of-words representation for biomedical time series classification. *Biomedical Signal Processing and Control*, Elsevier, v. 8, n. 6, p. 634–644, 2013.

- WANG, Z.; OATES, T. Imaging time-series to improve classification and imputation. *arXiv preprint arXiv:1506.00327*, 2015.
- WANG, Z.; YAN, W.; OATES, T. Time series classification from scratch with deep neural networks: A strong baseline. In: IEEE. *2017 International joint conference on neural networks (IJCNN)*. [S.l.], 2017. p. 1578–1585.
- WOLPERT, D. H.; MACREADY, W. G. No free lunch theorems for optimization. *IEEE transactions on evolutionary computation*, IEEE, v. 1, n. 1, p. 67–82, 1997.
- WU, C. et al. Spatio-temporal heterogeneous federated learning for time series classification with multi-view orthogonal training. In: *Proceedings of the 32nd ACM International Conference on Multimedia*. [S.l.: s.n.], 2024. p. 2613–2622.
- WU, Y.-L.; AGRAWAL, D.; ABBADI, A. E. A comparison of dft and dwt based similarity search in time-series databases. In: *Proceedings of the ninth international conference on Information and knowledge management*. [S.l.: s.n.], 2000. p. 488–495.
- XING, H. et al. An efficient federated distillation learning system for multi-task time series classification. *IEEE Transactions on Instrumentation and Measurement*, 2022.
- XING, H. et al. An efficient federated distillation learning system for multitask time series classification. *IEEE Transactions on Instrumentation and Measurement*, IEEE, 2022.
- XING, Z.; PEI, J.; KEOGH, E. A brief survey on sequence classification. *ACM Sigkdd Explorations Newsletter*, ACM New York, NY, USA, v. 12, n. 1, p. 40–48, 2010.
- YANG, Q.; WU, X. 10 challenging problems in data mining research. *International Journal of Information Technology & Decision Making*, World Scientific, v. 5, n. 04, p. 597–604, 2006.
- YE, L.; KEOGH, E. Time series shapelets: a new primitive for data mining. In: *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*. [S.l.: s.n.], 2009. p. 947–956.
- YE, L.; KEOGH, E. Time series shapelets: a novel technique that allows accurate, interpretable and fast classification. *Data mining and knowledge discovery*, Springer, v. 22, n. 1, p. 149–182, 2011.
- YI, B.-K.; FALOUTSOS, C. Fast time sequence indexing for arbitrary lp norms. Carnegie Mellon University, 2000.
- YI, B.-K.; JAGADISH, H. V.; FALOUTSOS, C. Efficient retrieval of similar time sequences under time warping. In: IEEE. *Proceedings 14th International Conference on Data Engineering*. [S.l.], 1998. p. 201–208.
- YOLCU, O. C.; LAM, H.-K. A combined robust fuzzy time series method for prediction of time series. *Neurocomputing*, Elsevier, v. 247, p. 87–101, 2017.
- YOUNIS, R. et al. Flames2graph: An interpretable federated multivariate time series classification framework. In: *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. New York, NY, USA: Association for Computing Machinery, 2023. ISBN 9798400701030.

- ZADEH, L. A. Fuzzy sets. *Information and control*, Elsevier, v. 8, n. 3, p. 338–353, 1965.
- ZHANG, C. et al. A survey on federated learning. *Knowledge-Based Systems*, 2021.
- ZHANG, Y.; YANG, Q. A survey on multi-task learning. *IEEE transactions on knowledge and data engineering*, IEEE, v. 34, n. 12, p. 5586–5609, 2021.
- ZHENG, Y. et al. Time series classification using multi-channels deep convolutional neural networks. In: SPRINGER. *International conference on web-age information management*. [S.l.], 2014. p. 298–310.
- ZHIYU, L.; WANG, H. *FedST: Federated Shapelet Transformation for Interpretable Time Series Classification*. 2023.
- ZHU, J.; MA, X.; BLASCHKO, M. B. Confidence-aware personalized federated learning via variational expectation maximization. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. [S.l.: s.n.], 2023.
- ZOU, Y. et al. Complex network approaches to nonlinear time series analysis. *Physics Reports*, Elsevier, 2019.