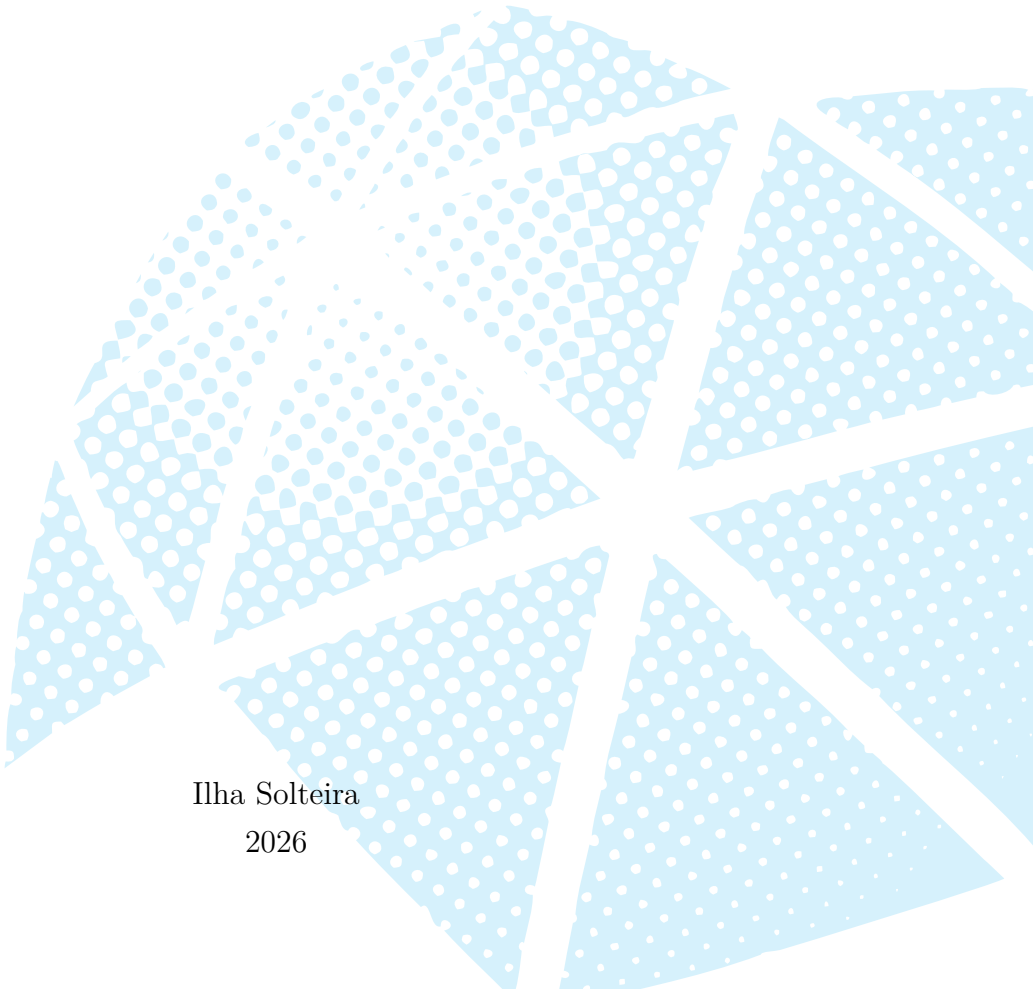
UNIVERSIDADE ESTADUAL PAULISTA "JÚLIO DE MESQUITA FILHO"
CÂMPUS DE ILHA SOLTEIRA

CAROLINA BERTON SANCHES

TÉCNICAS DE APRENDIZADO DE MÁQUINA PARA CLASSIFICAÇÃO DE DECÚBITOS PARA PREVENÇÃO DE LESÕES POR PRESSÃO



Ilha Solteira
2026

UNIVERSIDADE ESTADUAL PAULISTA
"JÚLIO DE MESQUITA FILHO"
Campus de Ilha Solteira

CAROLINA BERTON SANCHES

TÉCNICAS DE APRENDIZADO DE MÁQUINA PARA CLASSIFICAÇÃO DE DECÚBITOS PARA PREVENÇÃO DE LESÕES POR PRESSÃO

Dissertação apresentada à Universidade Estadual Paulista (UNESP), Escola de Engenharia, Ilha Solteira, como parte dos requisitos para obtenção do título de Mestra em Engenharia Mecânica.

Área de Conhecimento: Mecânica dos Sólidos.

Prof. Dr. Amarildo Tabone Paschoalini
Orientador

Prof. Dr. Marcio Antonio Bazani
Co-orientador

Ilha Solteira
2026

FICHA CATALOGRÁFICA
Desenvolvida pela Diretoria Técnica de Biblioteca e Documentação

Sanches, Carolina Berton.
S211t Técnicas de aprendizado de máquina para classificação de decúbitos para prevenção de lesões por pressão / Carolina Berton Sanches. -- Ilha Solteira: [s.n.], 2026
83 f. : il.

Dissertação (mestrado) - Universidade Estadual Paulista. Faculdade de Engenharia de Ilha Solteira. Área de conhecimento: Mecânica dos Sólidos, 2026

Orientador: Amarildo Tabone Paschoalini
Coorientador: Marcio Antonio Bazani
Inclui bibliografia

1. Lesão por pressão. 2. Mapa de pressão. 3. Classificação de decúbitos. 4. Aprendizado de máquina.

Elaborada por Raiane da Silva Santos - CRB-8/9999

UNIVERSIDADE ESTADUAL PAULISTA

Câmpus de Ilha Solteira

CERTIFICADO DE APROVAÇÃO

TÍTULO DA DISSERTAÇÃO: TÉCNICAS DE APRENDIZADO DE MÁQUINA PARA CLASSIFICAÇÃO DE DECÚBITOS PARA PREVENÇÕES DE LESÕES POR PRESSÃO

AUTORA: CAROLINA BERTON SANCHES
ORIENTADOR: AMARILDO TABONE PASCHOALINI
COORIENTADOR: MARCIO ANTONIO BAZANI

Aprovada como parte das exigências para obtenção do Título de Mestra em Engenharia Mecânica, área: Ciências Mecânicas pela Comissão Examinadora:

Prof. Dr. MARCIO ANTONIO BAZANI (Participaçao Presencial)
Departamento de Engenharia Mecânica / UNESP / Câmpus de Ilha Solteira - FEIS

Prof. Dr. APARECIDO CARLOS GONCALVES (Participaçao Presencial)
Departamento de Engenharia Mecânica / UNESP / Câmpus de Ilha Solteira - FEIS

Prof. Dr. EDSON ANTONIO CAPELLO SOUSA (Participaçao Presencial)
Departamento de Engenharia Mecânica / UNESP / Câmpus de Bauru - FEB

Ilha Solteira, 30 de julho de 2026.

AGRADECIMENTOS

Primeiramente, agradeço aos meus pais pelo amor e apoio incondicionais que sempre me dedicaram. Obrigada por estarem presentes e me proporcionarem tudo de melhor em todas as fases da minha vida, sendo meus exemplos e minha base, essenciais para que eu chegasse até aqui.

Ao meu irmão, agradeço pela sensatez constante, por sempre me incentivar a agir de acordo com os meus objetivos, pela amizade e capacidade única de me fazer rir em qualquer momento.

Ao meu namorado, deixo meu agradecimento pelo carinho, paciência e fonte de motivação, sempre me lembrando e acreditando no meu potencial diante dos desafios. Seu auxílio foi fundamental em toda essa trajetória.

Agradeço também aos amigos que sempre estiveram comigo e fizeram parte da minha jornada.

No âmbito acadêmico, agradeço aos meus orientadores pelos ensinamentos e pela confiança depositada em mim desde a graduação, quando apoiaram meus primeiros projetos e direcionaram meus passos na pesquisa.

Por fim, agradeço a toda a equipe do Programa de Pós-Graduação em Engenharia Mecânica e a toda a UNESP, estendendo meu carinho à Ilha Solteira pelo acolhimento ao longo de tantos anos.

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código Financeiro 001. Código de Financiamento 88887.994953/2024-00.

“O sucesso não é a vitória, mas tudo aquilo pelo que você lutou para conquistar.”

Rafael Nadal

RESUMO

Historicamente, as lesões por pressão representam um problema relevante de saúde pública, que geram grandes prejuízos clínicos aos pacientes e impactos econômicos às famílias e instituições. A prevenção dessas lesões é essencial e, embora sejam evitáveis por meio de protocolos preventivos, tem-se ainda espaço para o desenvolvimento e aperfeiçoamento de tecnologias assistivas para o mesmo propósito. Nesse contexto, observa-se que a capacidade de identificar o posicionamento do indivíduo no leito é uma base promissora para aplicação em diversos tipos de sistemas — desde alertas para enfermeiros, até atuação em redistribuição de pressão automática. Alinhado a essa necessidade, este trabalho tem como objetivo geral implementar e avaliar o desempenho de diferentes técnicas de *machine learning* na classificação de decúbitos, a partir de mapas de pressão reais. A metodologia utiliza um conjunto de dados de acesso aberto composto por matrizes de sensores de pressão, registrando 17 posturas de 13 participantes. São explorados três algoritmos como ferramentas de classificação: Florestas Aleatórias, *K-Nearest Neighbors* e Redes Neurais Convolucionais (RNC). Os modelos são avaliados sob duas abordagens de validação, uma baseada na segmentação aleatória do banco de dados e outra na divisão por pacientes, através da estratégia de *Leave-One-Subject-Out*. Os resultados da primeira abordagem indicam acurácias próximas de 99%, enquanto os da segunda variam de 45% a 57%, no entanto, apresentam uma característica mais realista para a generalização com pacientes inéditos. Nesta situação a RNC se destaca com uma acurácia inicial superior e com a vantagem de extrair características sem a necessidade de processamento manual. Com isso, realiza-se a otimização de hiperparâmetros da RNC para o cenário de 17 classes e, o modelo selecionado estabiliza seu desempenho em torno de 80%. Posteriormente, ao mapear os erros decorrentes da semelhança física entre posturas análogas, realiza-se uma nova otimização estrutural na rede para 8 posições mais relevantes, o que eleva a acurácia global para a faixa de 90%. Dessa forma, conclui-se que o método desenvolvido é bastante promissor e estabelece uma possível base tecnológica para diversos tipos de sistemas automatizados que objetivam a prevenção de eventos adversos.

Palavras-chave: Lesão por pressão; mapa de pressão; classificação de decúbitos; aprendizado de máquina.

ABSTRACT

Historically, pressure injuries have represented a significant public health issue, causing substantial clinical harm to patients as well as economic burdens on families and healthcare institutions. Preventing these injuries is essential and, although they are largely avoidable through preventive protocols, there remains considerable opportunity for the development and improvement of assistive technologies aimed at the same purpose. In this context, the ability to identify a patient's body position in bed emerges as a promising foundation for a wide range of applications, from alert systems for healthcare professionals to automated pressure redistribution systems. In line with this need, the main objective of this study is to implement and evaluate the performance of different machine learning techniques for body position classification based on real pressure maps. The proposed methodology uses an open-access dataset composed of pressure sensor matrices acquired from 13 participants performing 17 body positions. Three classification algorithms are investigated: Random Forests, K-Nearest Neighbors, and Convolutional Neural Networks (CNN). The models are evaluated under two validation strategies: random dataset partitioning and pacient-based split using the Leave-One-Subject-Out strategy. The results obtained with the first partitioning approach indicate classification accuracies close to 99%, whereas the second strategy yields accuracies ranging from 45% to 57%, providing a more realistic assessment of the models' ability to generalize to unseen subjects. Under this evaluation, the CNN achieves the highest initial accuracy while offering the additional advantage of automatically extracting relevant features without requiring manual feature engineering. Subsequently, the CNN hyperparameters are optimized for the 17-class scenario, leading to a model that stabilizes its performance at approximately 80% accuracy. After analyzing the classification errors associated with the physical similarity among analogous body positions, a new structural optimization is performed considering the eight most relevant positions, increasing the overall classification accuracy to approximately 90%. Therefore, the proposed method demonstrates considerable potential and establishes a technological foundation for the development of automated systems aimed at preventing pressure injuries and other related adverse events.

Keywords: Pressure injury; pressure map; in-bed posture classification; machine learning.

LISTA DE ILUSTRAÇÕES

Figura 1 – Modelo de Kosiak - relação entre pressão/deformação tecidual, tempo e tolerância do tecido à carga. 16
Figura 2 – Mecanismos fisiopatológicos relatadas que ocasionam LPs. 17
Figura 3 – Pele e tecidos subjacentes em condições normais. 18
Figura 4 – Estágios de I a IV de uma LP. 18
Figura 5 – Exemplos de (a) sobreposição e (b) colchão comerciais. 21
Figura 6 – Exemplo esquemático de um algoritmo de Floresta Aleatória. 24
Figura 7 – Exemplo esquemático de um algoritmo KNN. 26
Figura 8 – Exemplo esquemático de uma arquitetura básica de uma RNC. 29
Figura 9 – Exemplo de tapete de pressão com matriz de sensores piezoresistivos. . 33
Figura 10 – Posturas padrão. 35
Figura 11 – Variações de posturas. 35
Figura 12 – Exemplos de mapas de pressão gerados a partir do *dataset*. 36
Figura 13 – Matriz de Confusão – Floresta Aleatória. 41
Figura 14 – Diagrama de Pareto de Relevância de *Features* – Floresta Aleatória. . . 42
Figura 15 – Matriz de Confusão – *K-Nearest Neighbors*. 43
Figura 16 – Matriz de Confusão – Rede Neural Convolucional. 44
Figura 17 – Matriz de Confusão Global – Floresta Aleatória. 46
Figura 18 – Matriz de Confusão Global – *K-Nearest Neighbors*. 46
Figura 19 – Matriz de Confusão Global – Rede Neural Convolucional. 47
Figura 20 – Comparação de acurácia e desvio padrão para 10 Modelos de RNC (17 Classes). 49
Figura 21 – Matriz de Confusão Global (17 Classes) – RNC Otimizada (Modelo 5). 50
Figura 22 – Comparação de acurácia e desvio padrão para 10 Modelos de RNC (8 Classes). 51
Figura 23 – Matriz de Confusão Global (8 Classes) – RNC Otimizada (Modelo 5). . 52

LISTA DE TABELAS

Tabela 1 – Tabela de referência para a linguagem clínica atual / termos definidos relacionados a superfícies de suporte. 20
Tabela 2 – Dados antropométricos dos participantes. 34
Tabela 3 – Decúbitos e condições experimentais. 34
Tabela 4 – Modelos para busca de melhores hiperparâmetros para RNC. 40
Tabela 5 – Métricas de desempenho por classe para diferentes algoritmos (Segmentação Aleatória). 45
Tabela 6 – Métricas de desempenho por classe para diferentes algoritmos com validação cruzada (Segmentação por Paciente). 48

LISTA DE ABREVIATURAS E SIGLAS

ANVISA	Agência Nacional de Vigilância Sanitária
CNN	*Convolutional Neural Network*
EA	Evento Adverso
EPUAP	*European Pressure Ulcer Advisory Panel*
FN	Falsos Negativos
FP	Falsos Positivos
KNN	*K-Nearest Neighbors*
LOSO	*Leave-One-Subject-Out*
LP	Lesão por Pressão
ML	*Machine Learning*
MS	Ministério da Saúde
MVP	*Minimum Viable Product*
NPIAP	*National Pressure Injury Advisory Panel*
NSP	Núcleo de Segurança do Paciente
OMS	Organização Mundial de Saúde
PPPI	*Pan Pacific Pressure Injury Alliance*
PNSP	Programa Nacional de Segurança do Paciente
PSP	Plano de Segurança do Paciente
ReLU	*Rectified Linear Unit*
RF	*Random Forest*
SNVS	Sistema Nacional de Vigilância Sanitária
SS	Superfícies de Suporte
VN	Verdadeiros Negativos
VP	Verdadeiros Positivos

SUMÁRIO

1 INTRODUÇÃO 11

1.1 Objetivos 13

1.2 Contribuições Esperadas 14

1.3 Organização do Trabalho 14

2 REVISÃO DA LITERATURA 15

2.1 Lesões por Pressão 15

2.1.1 Classificação das LPs 18

2.1.2 Métodos de Prevenção 19

2.2 Aprendizado de Máquina 22

2.2.1 Florestas Aleatórias 22

2.2.2 *K-Nearest Neighbors* 24

2.2.3 Redes Neurais Convolucionais 27

2.2.4 Métricas de Desempenho 29

3 METODOLOGIA 33

3.1 *Dataset* 33

3.2 Aprendizado de Máquina 37

3.2.1 Floresta Aleatória 37

3.2.2 *K-Nearest Neighbors* 38

3.2.3 Rede Neural Convolucional 38

4 RESULTADOS E DISCUSSÕES 41

4.1 Primeira Abordagem: Segmentação Aleatória 41

4.2 Segunda Abordagem: Segmentação por Paciente 45

4.2.1 Otimização de Hiperparâmetros da RNC (17 Posições) 49

4.2.2 Otimização de Hiperparâmetros da RNC (8 Posições) 51

4.3 Comparação entre Abordagens 52

5 CONSIDERAÇÕES FINAIS 54

5.1 Próximas Etapas 55

APÊNDICE A – IMPLEMENTAÇÃO DA GERAÇÃO DOS CONJUNTOS DE DADOS 62

A.1 Geração do conjunto de dados para Redes Neurais Convolucionais . . . 62

A.2 Geração do conjunto de dados para *K-Nearest Neighbors* e Floresta Aleatória 63

APÊNDICE B – IMPLEMENTAÇÃO DO CLASSIFICADOR FLORESTA ALEATÓRIA 67

APÊNDICE C – IMPLEMENTAÇÃO DO CLASSIFICADOR *K-NEAREST NEIGHBORS* 71

APÊNDICE D – IMPLEMENTAÇÃO DO CLASSIFICADOR REDE NEURAL CONVOLUCIONAL . . . 75

1 INTRODUÇÃO

De acordo com *National Pressure Injury Advisory Panel* (NPIAP), *European Pressure Ulcer Advisory Panel* (EPUAP) e *Pan Pacific Pressure Injury Alliance* (PPPI) (National Pressure Injury Advisory Panel; European Pressure Ulcer Advisory Panel; Pan Pacific Pressure Injury Alliance, 2025b), pode-se definir uma lesão por pressão (LP) como um dano localizado na pele ou tecidos moles subjacentes, resultante da pressão prolongada e/ou intensa sobre proeminências ósseas em combinação com cisalhamento, podendo ser também relacionada ao uso de dispositivo médico ou a outro artefato. A tolerância do tecido a essas condições pode depender de outros fatores como microclima, nutrição, perfusão[1] e comorbidades (Coleman *et al.*, 2014).

Historicamente, lesões por pressão são um problema relevante na área da saúde. No século XVI, fatores como pressão e nutrição foram atribuídos como itens relevantes para o problema. Já no século XIX, observou-se que a pressão levava à compressão dos vasos sanguíneos cutâneos, resultando em necrose do tecido. Além disso, houve também maior detalhamento nas descrições de suas manifestações clínicas. Porém, foi apenas no século XX em que os tratamentos e aprimoramentos técnicos foram desenvolvidos (Robert Kwon; Jeffrey E. Janis, 2016).

Atualmente, em consequência dos avanços científicos, os cuidados em saúde melhoraram consideravelmente. No entanto, ainda existem diversos riscos, os quais podem causar sequelas permanentes, maior frequência de visitas aos hospitais, maior tempo de internação ou, até mesmo, a morte. Nesse contexto, a Organização Mundial de Saúde (OMS) traz um conceito que tem ganhado grande importância ao longo das últimas décadas, a "segurança do paciente" (SP), entendida como a redução do risco de dano desnecessário ligado aos cuidados em saúde (Organização Mundial da Saúde, 2016).

Dentro do tema de segurança do paciente, tem-se a definição de "evento adverso" (EA), o qual se refere a um incidente que resulte em dano a um paciente. Esses incidentes constituem um problema de saúde pública e diversos fatores relacionados à deficiência da assistência à saúde tendem a aumentar consideravelmente suas probabilidades de ocorrência (Organização Mundial da Saúde, 2016; Brasil, 2023). Alguns estudos indicam que a maioria dos eventos adversos é evitável, além de constatarem que aproximadamente 28% são atribuíveis à negligência de um profissional de saúde, enquanto 42%

[1] Fluxo de fluido através de um órgão, tecido ou sistema circulatório.

indicam outros fatores (Dubois; Brook, 1988; Bedell *et al.*, 1991; Leape *et al.*, 1993).

Em 2013, o Ministério da Saúde (MS) instituiu no Brasil o Programa Nacional de Segurança do Paciente (PNSP), que busca qualificar o cuidado em saúde por todos os estabelecimentos do território nacional (Brasil, 2013a). Em conjunto com o programa, a Agência Nacional de Vigilância Sanitária (ANVISA) estabeleceu a obrigatoriedade de implantação do Núcleo de Segurança do Paciente (NSP) em serviços de saúde, que auxilia o processo de implantação do Plano de Segurança do Paciente (PSP), sendo a prevenção de lesões por pressão uma das principais ações previstas (Brasil, 2013b).

No período de 2014 a 2022, dentre 1 milhão de incidentes notificados ao Sistema Nacional de Vigilância Sanitária (SNVS), 20,3% corresponderam a casos de lesões por pressão, sendo o segundo tipo de evento mais notificado pelos NSPs. Em relação aos casos de *never events*[2], 72,21% eram decorrentes de lesões por pressão estágio III, enquanto 21,57% apontaram lesões de estágio IV, além de LPs contribuírem diretamente a 1,21% dos óbitos notificados nesse período (Brasil, 2022).

As consequências do desenvolvimento de LPs atingem diretamente o paciente, uma vez que esse problema dificulta o processo de recuperação funcional, causa dor, riscos de graves infecções, sepse[3] ou a morte. No entanto, além dos prejuízos ao paciente, os impactos são também econômicos, gerando custos elevados às instituições, com internações prolongadas, maiores gastos de materiais consumíveis, aumento da carga de trabalho por parte da equipe de saúde, entre outras consequências indiretas (Brasil, 2021; Bergquist-Beringer *et al.*, 2011).

Alguns estudos mostram que despesas relacionadas à hospitalização suplementar, processos judiciais, infecções adquiridas em hospitais, invalidez e despesas médicas, custaram a alguns países entre US$ 6 bilhões e US$ 29 bilhões por ano (Institute of Medicine, 2000; Donaldson, 2002).

Diante desse panorama, fica evidente a necessidade de redução dos danos aos pacientes e a mitigação dos prejuízos econômicos através do aperfeiçoamento da segurança do paciente. Isso reforça a importância de estudos contínuos sobre o tema, que vão desde a constante atualização dos protocolos de enfermagem, até a concepção de tecnologias de suporte. Ferramentas que auxiliem a equipe assistencial por meio de alertas inteligentes

[2] Eventos que nunca deveriam ocorrer em serviços de saúde.

[3] Disfunção única ou múltipla de órgãos e sistemas, associada à resposta do organismo frente a um agente infeccioso.

de monitoramento, ou mesmo o desenvolvimento de equipamentos automatizados capazes de atuar diretamente no leito, redistribuindo a pressão em regiões críticas, representam frentes de pesquisa cada vez mais necessárias e promissoras.

Diversas podem ser as ferramentas e aplicações voltadas a essa finalidade, contudo, para a grande maioria delas, a capacidade de identificar posições torna-se a base para a maioria dos sistemas. Seja para cronometrar o tempo de permanência em uma posição, mapear zonas de risco sobre proeminências ósseas ou direcionar o inflamento de células de ar para a redistribuição de pressão, o dispositivo precisa, primeiramente, saber como o paciente está posicionado. Sem um reconhecimento postural na camada inicial do sistema, qualquer tomada de decisão subsequente pode ser comprometida.

Para viabilizar essa identificação automatizada em tempo real, as técnicas de inteligência artificial, especialmente as abordagens de aprendizado de máquina (do inglês *machine learning*, ML), são propostas bastante promissoras para essa aplicação, uma vez que possuem como característica a alta capacidade de reconhecer padrões complexos, podendo superar limitações de sistemas puramente baseados em regras rígidas. Dessa forma, o emprego dessas técnicas abre caminhos para o desenvolvimento de soluções preventivas robustas e adaptáveis às variações físicas de cada paciente.

1.1. OBJETIVOS

A partir desse contexto, o presente trabalho tem como objetivo geral implementar e avaliar o desempenho de diferentes técnicas de aprendizado de máquina aplicadas como ferramentas na classificação de decúbitos[4], considerando dados de pressão reais. Para isso, alguns objetivos específicos podem ser listados:

- Analisar e processar dados de pressões reais, obtidos a partir de um *dataset* de acesso aberto;
- Implementar três métodos distintos de aprendizado de máquina para identificação e classificação das posições;
- Comparar os resultados obtidos em cada um dos métodos de ML;
- Selecionar um dos métodos de ML para otimização de seus hiperparâmetros.

[4] Posição horizontal do corpo em repouso.

1.2. CONTRIBUIÇÕES ESPERADAS

A principal contribuição deste trabalho é apresentar um método base que viabilize o desenvolvimento de uma ampla gama de aplicações tecnológicas voltadas à saúde assistiva e à segurança do paciente. Com uma identificação e classificação correta de decúbitos, criam-se diversas possibilidades voltadas à implementação de sistemas auxiliares para equipes de enfermagem.

Dessa forma, espera-se que este estudo colabore com a evolução das pesquisas na área e sirva de apoio para o avanço no desenvolvimento de novas soluções tecnológicas capazes de prevenir eventos adversos e melhorar a qualidade de vida dos pacientes.

1.3. ORGANIZAÇÃO DO TRABALHO

Este trabalho está estruturado em quatro capítulos principais, além desta introdução, organizados da seguinte forma:

O **Capítulo 2** contextualiza a fundamentação teórica da pesquisa, abordando informações relevantes sobre as lesões por pressão, bem como seus métodos tradicionais de prevenção e o panorama de tecnologias correlatas. Adicionalmente, apresenta os princípios das três técnicas de aprendizado de máquina utilizados neste estudo.

O **Capítulo 3** detalha os procedimentos adotados ao longo da pesquisa. São descritas as características do conjunto de dados utilizado, seguidas das etapas de processamento, parametrização e aplicação dos dados nos algoritmos de aprendizado de máquina selecionados.

O **Capítulo 4** expõe as análises e as comparações de desempenho dos modelos a partir de duas abordagens distintas de segmentação do banco de dados para aplicação nos aprendizados de máquina. A primeira, uma divisão aleatória dos dados e a segunda, uma segmentação por paciente. O capítulo se encerra com uma discussão comparativa integrando as métricas resultantes e a capacidade de generalização de cada ferramenta.

Por fim, o **Capítulo 5** sintetiza as principais conclusões obtidas ao longo da pesquisa, destaca as contribuições do método proposto e aponta diretrizes para a continuidade e o aprimoramento deste estudo.

2 REVISÃO DA LITERATURA

Neste capítulo, é apresentada uma breve revisão da literatura sobre a etiologia[5] e classificação da lesão por pressão, seguida de uma descrição dos métodos de prevenção já utilizados e apresentação de pesquisas que fazem uso de diferentes tecnologias para o mesmo objetivo. Por fim, são enumeradas três técnicas de aprendizado de máquina muito utilizadas como ferramentas em diversas aplicações, descrevendo seus princípios básicos de estruturação.

2.1. LESÕES POR PRESSÃO

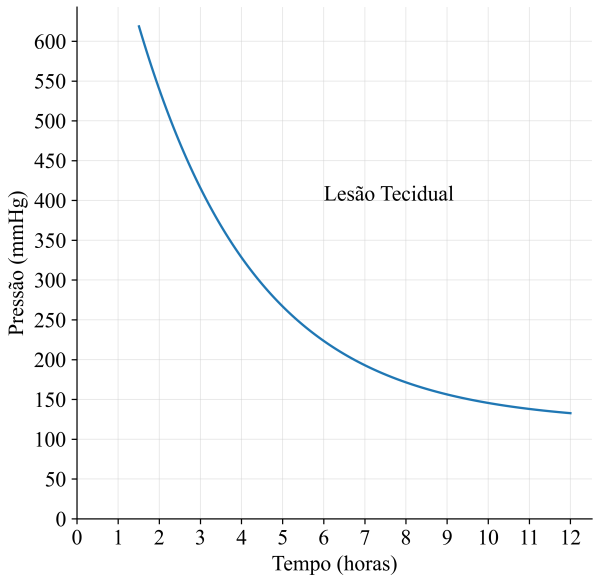
Como mencionado pela definição de lesão por pressão, embora a carga de compressão intensa e/ou prolongada em conjunto com esforços cisalhantes sejam suas principais causas, a LP se destaca por sua etiologia multifatorial. Sendo assim, fatores intrínsecos e extrínsecos ao paciente são associados ao desenvolvimento dessas lesões. Idade, estado mental e nutricional, condição da inervação sensorial, hidratação, nível reduzido de consciência, perfusão tecidual insuficiente, tonicidade muscular e comorbidades, são alguns dos fatores listados como intrínsecos ao paciente (National Pressure Injury Advisory Panel; European Pressure Ulcer Advisory Panel; Pan Pacific Pressure Injury Alliance, 2025b). Quanto às condições rotuladas como extrínsecas, além das cargas de compressão e cisalhamento, estão presentes a fricção, a imobilização e o microclima (umidade e temperatura) (Coleman *et al.*, 2014; Kosiak, 1959; Reswick; Rogers, 1976).

Cargas mecânicas sobre a pele geram uma deformação tecidual, o que pode resultar em lesões diretas ao comprometer funções celulares essenciais, ou indiretas, devido à obstrução de vasos sanguíneos e linfáticos que ocasionam a morte celular. Alguns estudos de referência na área fornecem uma relação entre essa pressão/deformação tecidual, tempo e tolerância dos tecidos à carga. Sendo que essa também pode variar com base nos fatores intrínsecos ao indivíduo, conforme mencionado anteriormente (Kosiak, 1959; Reswick; Rogers, 1976). A seguir tem-se a representação dessa relação na Figura 1.

Ainda não é conhecido com clareza o sistema de tensões e deformações locais dentro dos tecidos deformados e como essa condição causa a morte celular, no entanto, existem

[5] Estudo das causas das doenças.

Figura 1 – Modelo de Kosiak - relação entre pressão/deformação tecidual, tempo e tolerância do tecido à carga.



Fonte: Adaptado de Kirkland-Walsh *et al.* (2015).

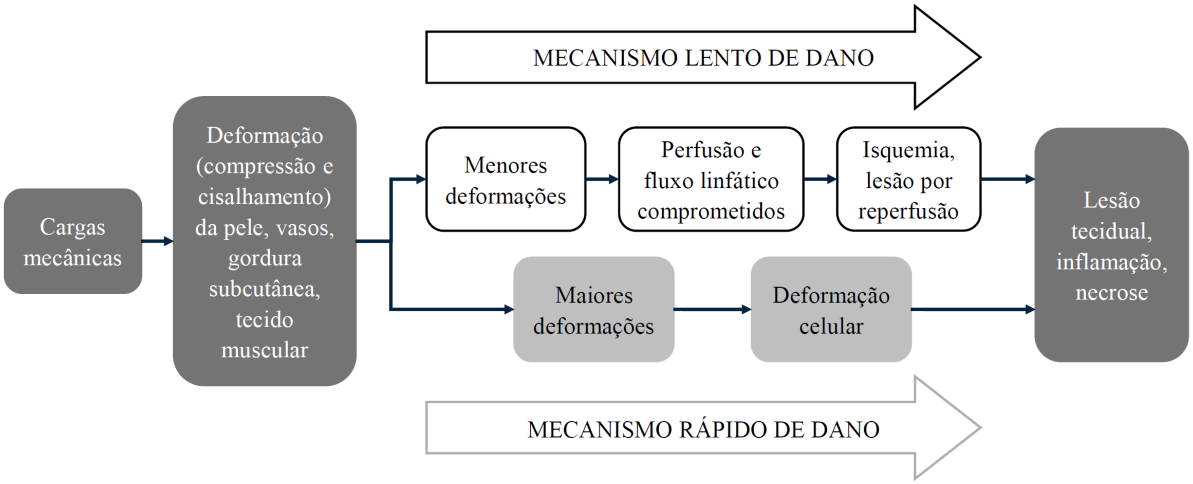
teorias fisiopatológicas que mencionam isquemia[6] localizada, dano pela deformação direta da célula, lesão de reperfusão[7] e função linfática comprometida (Bouten *et al.*, 2003; Gefen *et al.*, 2021). Novos estudos fornecem evidências que os quatro mecanismos patológicos atuam simultaneamente e de maneiras diferentes dentro dos níveis hierárquicos estruturais e funcionais (como vasos, tecidos e células), embora a isquemia e o dano celular direto sejam os maiores agravantes no desenvolvimento das LPs (Loerakker *et al.*, 2011; Gefen *et al.*, 2021).

Enquanto o dano causado por deformação celular direta é relacionado a um processo rápido e causa lesões teciduais profundas, a isquemia pode demorar muito mais tempo para gerar alterações patológicas (Oomens *et al.*, 2015). A Figura 2 ilustra o fluxo dos mecanismos fisiopatológicos que desencadeiam o dano tecidual, divididos de acordo com a intensidade da carga e a velocidade com que a lesão se estabelece.

Pela interpretação da figura, o processo se inicia a partir da aplicação de cargas mecânicas externas sobre o corpo, que resultam na deformação por compressão e cisalhamento da pele, dos vasos sanguíneos, da gordura subcutânea e do tecido muscular. A partir desse ponto inicial, o fluxo fisiopatológico ramifica-se em duas vias distintas de evolução, dependendo da magnitude das forças aplicadas. A primeira via, identifi-

[6] Suspensão ou deficiência da circulação do sangue que irriga um órgão, decorrente de obstrução arterial.
[7] Alterações, funcionais e estruturais, que se tornam aparentes durante o restabelecimento do fluxo após um período de isquemia.

Figura 2 – Mecanismos fisiopatológicos relatadas que ocasionam LPs.



Fonte: Adaptado de National Pressure Injury Advisory Panel, European Pressure Ulcer Advisory Panel e Pan Pacific Pressure Injury Alliance (2025b).

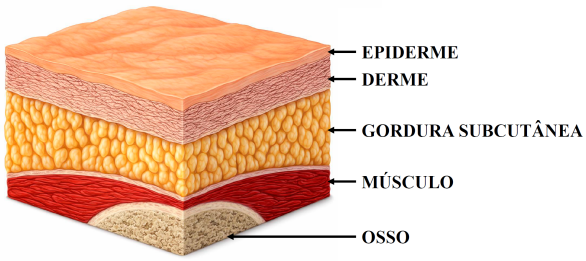
cada como o mecanismo lento de dano, ocorre quando o tecido é submetido a menores deformações, porém mantidas por um período prolongado. Nesse cenário, a perfusão sanguínea e o fluxo linfático locais são comprometidos, impedindo a chegada de oxigênio e nutrientes às células. A permanência dessa condição leva à isquemia e, posteriormente, à lesão por reperfusão quando a carga é aliviada, culminando no desfecho final de lesão tecidual, inflamação e necrose. Por outro lado, a segunda via descreve o mecanismo rápido de dano. Este processo é ativado quando o corpo sofre maiores deformações estruturais causadas por pressões elevadas. Sob essa condição intensa, ocorre a deformação celular direta, na qual a integridade física das células é destruída quase imediatamente. Assim como na via lenta, o resultado final desse estresse severo é o surgimento da lesão tecidual acompanhada de inflamação e necrose local.

É sugerido, também, que o desenvolvimento rápido de lesões profundas pode ser relacionado a cargas de compressão, enquanto lesões mais superficiais são possivelmente associadas a cargas de cisalhamento e à fricção (National Pressure Injury Advisory Panel; European Pressure Ulcer Advisory Panel; Pan Pacific Pressure Injury Alliance, 2025b). Os limites de tolerância dos tecidos para a ocorrência de danos são determinados pelas cargas citadas em conjunto com outros fatores, tais como localização anatômica, morfologia, propriedades dos tecidos, grau de deformação tecidual, microclima da pele e capacidade de reparo individual (Oomens *et al.*, 2015; Gefen *et al.*, 2021; Kottner *et al.*, 2018; Sonenblum *et al.*, 2015).

2.1.1 Classificação das LPs

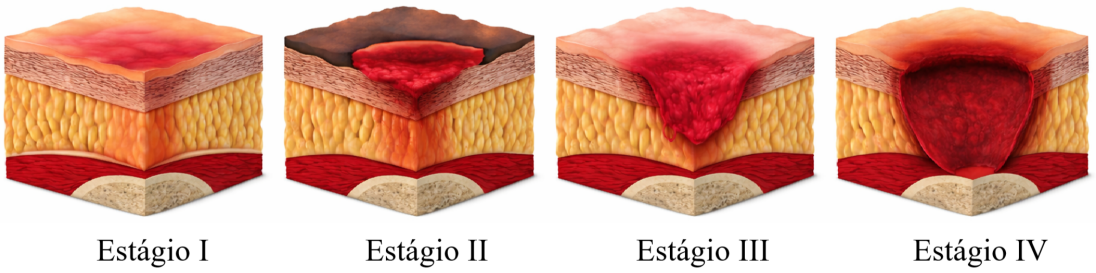
A classificação definida pelas organizações NPIAP e EPUAP (*International Pressure Ulcer Classification System*) varia de acordo com o comprometimento tecidual, que é subdividido em estágios I, II, III, IV, categorias "Não Classificável" e "Tissular Profunda", além das definições adicionais "Relacionada a Dispositivos Médicos" e "Membranas Mucosas". A Figura 3 a seguir, exemplifica uma área de pele e tecidos subjacentes em condições normais, enquanto a Figura 4 demonstra os estágios de I a IV de uma LP.

Figura 3 – Pele e tecidos subjacentes em condições normais.



Fonte: Adaptado de National Pressure Ulcer Advisory Panel, European Pressure Ulcer Advisory Panel e Pan Pacific Pressure Injury Alliance (2014).

Figura 4 – Estágios de I a IV de uma LP.



Fonte: Adaptado de National Pressure Ulcer Advisory Panel, European Pressure Ulcer Advisory Panel e Pan Pacific Pressure Injury Alliance (2014).

Com base nas descrições e ilustrações definidas pelas organizações internacionais, de maneira mais simplificada, uma LP de Estágio I é caracterizada por uma pele intacta com vermelhidão não branqueável em uma área localizada, geralmente, sobre proeminências ósseas. No Estágio II, observa-se a perda de espessura parcial da derme, apresentando-se como úlcera aberta superficial com leito da ferida vermelho/rosa, sem necrose. Durante o Estágio III, tem-se a perda total da espessura do tecido e a gordura subcutânea pode estar visível, porém osso, tendão ou músculo não são expostos. Por fim,

no Estágio IV das lesões por pressão, observa-se o comprometimento total do tecido com a efetiva exposição óssea, muscular ou de tendões. (National Pressure Ulcer Advisory Panel; European Pressure Ulcer Advisory Panel; Pan Pacific Pressure Injury Alliance, 2014).

2.1.2 Métodos de Prevenção

Diante das pesquisas que surgiram ao longo do século XX sobre etiologia e como se dá o processo de formação das lesões por pressão, desenvolveram-se novos métodos e aprimoramentos técnicos para seu tratamento, bem como protocolos e medidas para sua prevenção – uma vez conhecidos seus possíveis mecanismos fisiopatológicos e como intervir em situações de risco para seu desenvolvimento.

Com isso, diversas organizações ao redor do mundo investem e apresentam guias, diretrizes e protocolos de como realizar essas intervenções (Organização Mundial da Saúde, 2016; Haesler *et al.*, 2024). No Brasil, o MS e a ANVISA implementaram o Plano de Segurança do Paciente, tomando como bases as diretrizes internacionais. Algumas das principais medidas adotadas para a prevenção das LPs são a avaliação dos pacientes em risco; o manejo do estado nutricional, incluindo a hidratação; a inspeção e avaliação diária da pele; o manejo da umidade e a redistribuição da pressão, sendo a última considerada uma das intervenções mais críticas e eficientes nesse processo. Tradicionalmente, ela é realizada por meio do protocolo de mudança de decúbito, que consiste na alteração programada da posição do paciente no leito. Recomenda-se clinicamente que essa alternância ocorra em intervalos regulares, tipicamente a cada duas horas, ou conforme o nível de risco avaliado por escalas preditivas.

Ainda no contexto de redistribuição de pressão, as "superfícies de suporte" (SS, do inglês *Support Surfaces*) são componentes essenciais nos processos de prevenção e tratamento. As organizações EPUAP, NPIAP e PPPIA, definem superfícies de suporte como colchões, sobreposições e sistemas integrados especializados, projetados para redistribuir a pressão, reduzir o atrito e cisalhamento, além de auxiliar no controle do microclima (National Pressure Injury Advisory Panel; European Pressure Ulcer Advisory Panel; Pan Pacific Pressure Injury Alliance, 2025a).

Com o surgimento de novas pesquisas e desenvolvimento de produtos destinados a essa aplicação, foi necessária a revisão e criação de termos relacionados a esses equi-

pamentos, a fim de fornecer uma linguagem comum referente a conceitos físicos gerais, características de *design* e desempenho. A Tabela 1 apresenta alguns dos principais termos.

Tabela 1 – Tabela de referência para a linguagem clínica atual / termos definidos relacionados a superfícies de suporte.

Superfície de Suporte	**Componente Material**	**Características de Design**	**Características de Desempenho**
• Apoio ativo; • Apoio reativo.	• Ar; • Célula; • Espuma; • Gel.	• Pressão alternada; • Pressão baixa constante; • Conversível/Adaptável; • Híbrido; • Sistema de leito integrado; • Rotação lateral; • Baixa perda de ar; • Superfície multiregião; • Não-energizado; • Energizado; • Pulsação; • Auxílio de rotação; • Região.	• Envolvimento; • Imersão; • Microclima; • Redistribuição de Pressão.

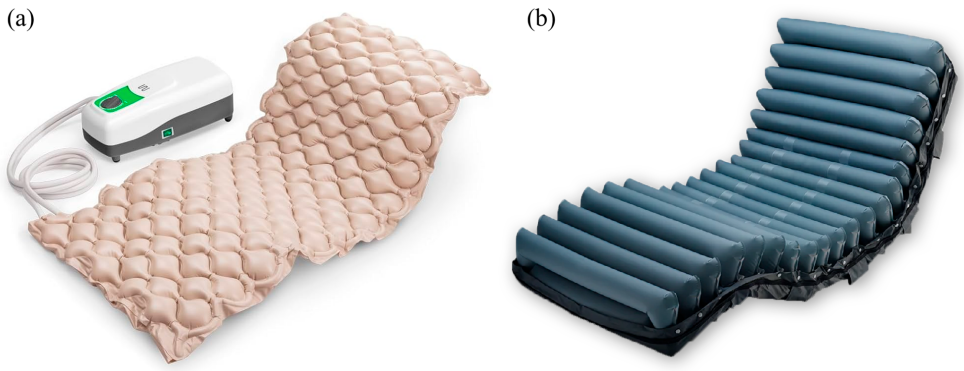
Fonte: Adaptado de National Pressure Injury Advisory Panel, European Pressure Ulcer Advisory Panel e Pan Pacific Pressure Injury Alliance (2025a).

Esses termos normatizados refletem a evolução tecnológica das superfícies de suporte, mostrando que o foco atual vai além da simples escolha de materiais internos como ar, gel ou espuma. As características de desempenho indicam que uma prevenção eficaz exige compreender conceitos como a imersão e o envolvimento do corpo na superfície. Da mesma forma, destaca-se o gerenciamento do microclima, que controla as condições de temperatura e umidade na interface entre a pele e o leito, fatores que reduzem a resistência do tecido ao dano mecânico. Essa diversidade de parâmetros de *design* dá origem aos variados produtos comerciais disponíveis no mercado atual.

Dentre esses produtos comerciais, algumas empresas especializadas no setor médico ao redor do mundo produzem colchões de ar e/ou células, destinados geralmente a ambientes hospitalares, além de sobreposições com os mesmos princípios. Alguns desses aparelhos se enquadram na categoria de SS ativos, porém não energizados (*non-powered*), ou seja, a pressurização e despressurização são feitas manualmente. Outros produtos apresentam pressurização automática (*powered*) com pressão alternada, que permite selecionar frequência e duração da carga em apenas dois canais de ar. Mesmo que essas superfícies de suporte atendam suas funções propostas em prevenir lesões por pressão, ainda percebe-se

uma lacuna que pode ser explorada para o aprimoramento desses aparelhos. A seguir são apresentados pela Figura 5 alguns modelos de produtos pesquisados.

Figura 5 – Exemplos de (a) sobreposição e (b) colchão comerciais.



Fonte: (a) https://tinyurl.com/2nutcvbx, (b) https://tinyurl.com/e9bw928y.

No que se refere às pesquisas acadêmicas sobre o assunto, destacam-se alguns temas de maior relevância como desenvolvimento de *softwares* para monitoramento de pressão e/ou posição para prevenção de lesões (Yousefi *et al.*, 2011c; Yousefi *et al.*, 2011b; Yousefi *et al.*, 2011a; Brunzini *et al.*, 2023; Javid *et al.*, 2023; Hsu; Lo, 2013), apresentando variados métodos de classificação de posições, utilizando técnicas de *machine learning* aplicadas na atualidade. Além de propostas de novos tipos e *designs* de colchões (Ghaderi; Chehre; Fathpour, 2023; Sikka; Garg, 2020), simulações para comparação de eficiências entre diferentes modelos de colchões (Alwasel *et al.*, 2021), estudos para desenvolvimento de bombas de ar mais eficientes para a aplicação apresentada (Krishnakumar *et al.*, 2018), entre outros.

Por fim, as pesquisas de patentes nacionais e internacionais apresentam também diversas ramificações dentro do tema, sendo apresentados diferentes sistemas de implementação como mecânicos e pneumáticos — com variados *designs* de distribuição do ar —, sistemas de alerta para profissionais da saúde sobre a necessidade de intervenção na posição do paciente, sistemas para monitoramento e controle de pressão de colchões, entre outros.

Diante da variedade de soluções encontradas comercialmente, na literatura e nos registros de patentes, pode-se relacionar que, para diversos tipos de abordagens — seja ela um sistema de alerta para a equipe de enfermagem, um relatório de monitoramento contínuo ou uma superfície de suporte com atuação ativa —, a capacidade de classificar

corretamente as posições dos pacientes no leito, através de técnicas de *machine learning*, torna-se uma ferramenta bastante útil para o desenvolvimento desses sistemas.

2.2. APRENDIZADO DE MÁQUINA

O aprendizado de máquina constitui um subcampo da inteligência artificial voltado para o desenvolvimento de algoritmos capazes de aprender padrões a partir de dados e aprimorar seu desempenho em tarefas preditivas sem a necessidade de programação explícita (Mitchell, 1997). Nas últimas décadas, o aprendizado de máquina tem sido amplamente aplicado em diversos domínios, incluindo a saúde, onde demonstrou grande potencial em tarefas como diagnóstico de doenças, predição de risco, análise de imagens e suporte à decisão (Topol, 2019).

O campo do *machine learning* abrange um conjunto de técnicas bem estabelecidas, cuja escolha depende do tipo de problema a ser abordado, da natureza dos dados disponíveis e dos objetivos da aplicação (Leijnen; Veen, 2020). Entre essas técnicas, destaca-se a Floresta Aleatória, um método de comitê (*ensemble methods*) baseado em árvores de decisão, que combina múltiplos modelos simples para produzir classificações robustas, reduzir *overfitting* e melhorar a capacidade de generalização (Breiman, 2001). Outra técnica amplamente utilizada é o *K-Nearest Neighbors* (KNN), um algoritmo não paramétrico que classifica uma nova instância com base em sua proximidade de exemplos previamente rotulados, mostrando-se particularmente eficaz em tarefas de reconhecimento de padrões e diagnóstico médico (Jain; Duin; Mao, 2000). Por fim, as Redes Neurais Artificiais, em especial as Redes Neurais Convolucionais (RNC), destacam-se por sua capacidade de extrair representações hierárquicas e complexas dos dados, alcançando excelentes resultados em tarefas como análise de imagens médicas e classificação de sinais fisiológicos (LeCun; Bengio; Hinton, 2015; Shen; Wu; Suk, 2017). Essas abordagens ilustram a diversidade de estratégias disponíveis no aprendizado de máquina e, nas próximas subseções, cada uma é descrita com foco em seu funcionamento e aplicações típicas.

2.2.1 Florestas Aleatórias

As Florestas Aleatórias (do inglês *Random Forests*, RF) são classificadas como um algoritmo clássico de aprendizado de máquina supervisionado, sendo utilizadas tanto para

tarefas de classificação quanto de regressão. Esse algoritmo é composto por árvores de decisão, as quais são fáceis de construir e interpretar, entretanto, tendem a apresentar precisão limitada. As *Random Forests*, por sua vez, combinam a simplicidade das árvores de decisão com maior acurácia, oferecendo diversas vantagens (Hastie; Tibshirani; Friedman, 2009). A primeira estrutura de uma árvore de decisão, conhecida como Nó Raiz, é alimentada com o conjunto de dados de treinamento do algoritmo. Posteriormente, ocorre o processo de divisão (*splitting*), que consiste em separar os dados de forma ótima. Métodos comuns utilizados para essa finalidade incluem Impureza de Gini e Entropia (Hastie; Tibshirani; Friedman, 2009). Após essa divisão, formam-se os Nós de Decisão, que fornecem ligações para os Nós Folha, os quais representam os pontos terminais da árvore, onde não há novas divisões.

O processo de construção de uma Floresta Aleatória geralmente começa com a geração de n subconjuntos de dados. Para cada subconjunto, múltiplas árvores de decisão são construídas, com as divisões dos nós baseadas em um número aleatório de variáveis. Uma vez criada a floresta, a estrutura está pronta para analisar dados que não foram utilizados durante sua construção. O processo de avaliação do algoritmo é semelhante à sua aplicação real; contudo, na avaliação, os dados de entrada possuem resultados conhecidos, permitindo medir a acurácia. Em contraste, durante a aplicação prática, os dados de entrada são novos e as previsões do modelo precisam ser confiáveis.

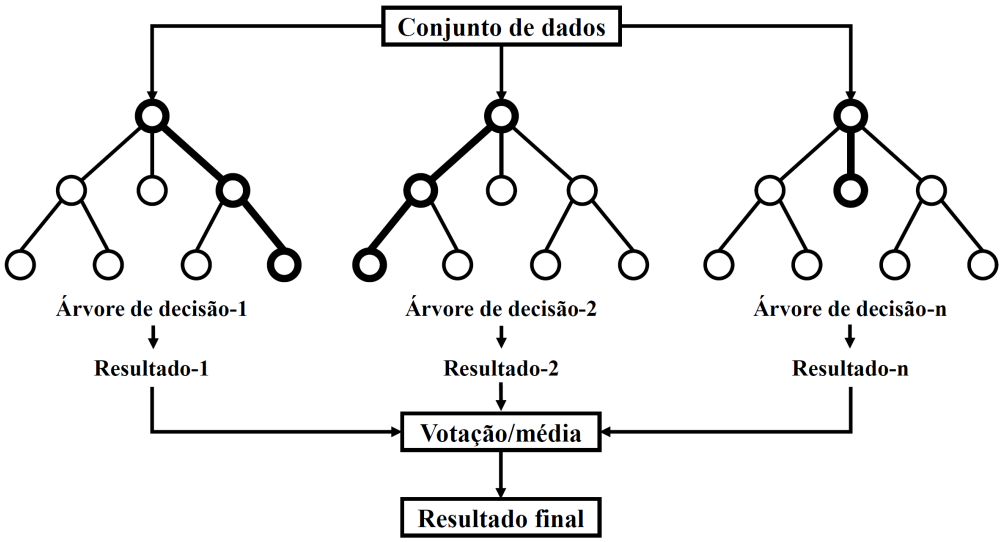
A previsão final em uma *Random Forest* para tarefas de classificação é determinada pelo voto majoritário entre as árvores individuais, enquanto para tarefas de regressão ela é dada pela média das saídas das árvores. Considerando o problema de classificação e uma floresta composta por B árvores de decisão, a classe predita para uma amostra x pode ser expressa por:

$$\hat{y} = \text{moda}\left\{h_b(x)\right\}_{b=1}^{B} \tag{2.1}$$

Em que $\hat{y}$ representa a classe prevista, $h_b(x)$ corresponde à previsão realizada pela b-ésima árvore de decisão e a função *moda* retorna a classe mais frequentemente votada entre todas as árvores da floresta (Breiman, 2001). Além disso, a Figura 6 ilustra o fluxo de processamento dessa técnica, baseada na criação de um comitê de árvores de decisão trabalhando em paralelo.

Pela interpretação da figura, observa-se que fluxo do algoritmo tem início no topo

Figura 6 – Exemplo esquemático de um algoritmo de Floresta Aleatória.



Fonte: Adaptado de Data Science Dojo (2024).

do diagrama com o "Conjunto de dados" original. A partir deste ponto, os dados são divididos e distribuídos para alimentar diferentes preditores em paralelo. Cada *subset* alimenta uma árvore distinta, mapeadas na imagem como desde a "Árvore de decisão-1" e "Árvore de decisão-2" até a "Árvore de decisão-n", garantindo a diversidade do modelo através da amostragem aleatória dos dados e de seus atributos. Na etapa intermediária, cada árvore de decisão processa a informação de maneira independente, seguindo um caminho específico de ramificações — destacado no diagrama pelas linhas pretas mais espessas. Cada uma dessas estruturas gera um diagnóstico isolado, resultando, respectivamente, no "Resultado-1", "Resultado-2" e "Resultado-n". Por fim, todas as respostas individuais são convergidas para uma camada de agregação, indicada pelo bloco "Votação/média". No contexto de problemas de classificação, que se aplica a este trabalho, o algoritmo realiza uma votação por maioria simples, na qual a classe mais votada entre todos os preditores individuais é selecionada, consolidando o "Resultado final" do sistema.

2.2.2 *K-Nearest Neighbors*

O algoritmo *K-Nearest Neighbors* é um algoritmo supervisionado baseado em instâncias, amplamente utilizado em tarefas de classificação e regressão em diversos domínios, sendo comumente aplicado no reconhecimento de padrões. Seu princípio básico assume que objetos semelhantes tendem a estar próximos uns dos outros em um espaço de características.

Para classificar uma nova amostra, o KNN identifica os k exemplos mais próximos no conjunto de treinamento e, em tarefas de classificação, atribui à nova amostra a classe mais frequente entre esses vizinhos. O valor de k é definido pelo usuário e é um hiperparâmetro fundamental, pois afeta diretamente o desempenho do modelo (Mitchell, 1997).

Sua implementação é considerada simples, com cada instância representada como um vetor numérico de características em um espaço multidimensional. A distância entre esses vetores é geralmente medida usando métricas como Euclidiana, Manhattan ou Minkowski (Jain; Duin; Mao, 2000). Considerando dois vetores de características $x = (x_1, x_2, \ldots, x_n)$ e $y = (y_1, y_2, \ldots, y_n)$, a distância Euclidiana é definida por:

$$d_E(x, y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \tag{2.2}$$

A distância Manhattan, também conhecida como distância de bloco ou cidade, é dada por:

$$d_M(x, y) = \sum_{i=1}^{n} |x_i - y_i| \tag{2.3}$$

Já a distância de Minkowski representa uma generalização das métricas anteriores, sendo expressa como:

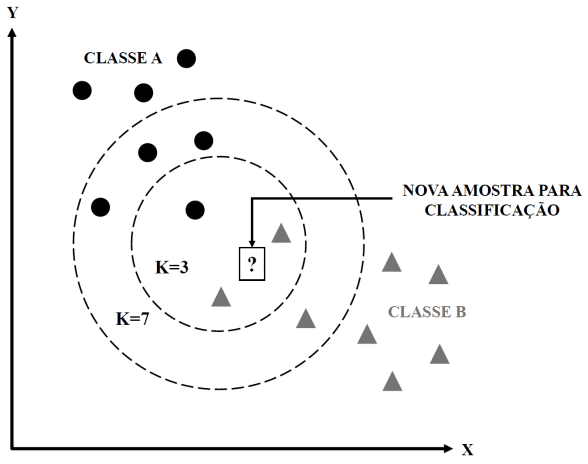
$$d_P(x, y) = \left(\sum_{i=1}^{n} |x_i - y_i|^p\right)^{\frac{1}{p}} \tag{2.4}$$

Em que p corresponde ao parâmetro de ordem da métrica. Quando $p = 1$, obtém-se a distância Manhattan, enquanto para $p = 2$ obtém-se a distância Euclidiana (Jain; Duin; Mao, 2000).

Esse tipo de algoritmo não envolve uma fase explícita de treinamento, visto que apenas armazena os dados de entrada, o que o caracteriza como um método de *lazy learning*. A fase de previsão envolve o cálculo das distâncias entre a nova instância e todas as instâncias do conjunto de dados, o que pode ser computacionalmente custoso para grandes volumes de informações (Dudani, 1976). A Figura 7 ilustra o princípio geométrico desse classificador baseado em instâncias, destacando a função do parâmetro k no processo de tomada de decisão.

A figura demonstra que o funcionamento do algoritmo baseia-se em um plano cartesiano bidimensional, cujos eixos X e Y representam os atributos extraídos das amostras

Figura 7 – Exemplo esquemático de um algoritmo KNN.



Fonte: Adaptado de Shah (2021).

conhecidas. No diagrama, os dados de referência encontram-se previamente mapeados e divididos em dois grupos distintos: a "Classe A", representada pelos círculos pretos, e a "Classe B", identificada pelos triângulos cinzas. O desafio do algoritmo consiste em inferir o rótulo do elemento posicionado no centro das circunferências, indicado como "Nova amostra para classificação". A classificação ocorre calculando a distância entre o ponto desconhecido e seus vizinhos. No primeiro cenário ($k = 3$), delimitado pelo círculo tracejado interno, a região engloba um círculo preto (Classe A) e dois triângulos cinzas (Classe B); por votação majoritária, a nova amostra é classificada como Classe B. Já no segundo cenário ($k = 7$), delimitado pela linha tracejada externa, o raio de busca é expandido e passa a conter quatro círculos pretos e três triângulos cinzas. Por maioria, a decisão é invertida, rotulando o ponto central como Classe A. Dessa forma, a imagem evidencia que a escolha do valor de k determina o comportamento e o diagnóstico final do modelo.

Algumas vantagens do algoritmo KNN incluem sua facilidade de implementação, bom desempenho em conjuntos de dados com fronteiras de decisão bem definidas e o fato de não exigir um treinamento explícito. Além disso, é um método intuitivo e de fácil interpretação. Por outro lado, suas principais desvantagens envolvem o alto custo computacional na fase de inferência, sensibilidade à escolha da métrica de distância e do valor de k, assim como vulnerabilidade a características irrelevantes e à presença de *outliers* (Suguna; Thanushkodi, 2010).

2.2.3 Redes Neurais Convolucionais

As Redes Neurais Convolucionais, ou *Convolutional Neural Networks* (CNN), constituem uma classe especializada de Redes Neurais Artificiais projetadas especificamente para processar dados com topologia em grade, como imagens bidimensionais. Essas redes são compostas por camadas convolucionais que extraem automaticamente características relevantes dos dados, preservando suas relações espaciais.

Diferentemente dos modelos tradicionais de *machine learning*, as RNCs eliminam a necessidade de extração manual de características e são altamente eficazes em tarefas de visão computacional (LeCun; Bengio; Hinton, 2015). A implementação de uma RNC baseia-se na composição sequencial de diferentes tipos de camadas. Inicialmente, tem-se a Camada de Entrada, que recebe os dados brutos — geralmente, imagens representadas como matrizes de *pixels*. Em seguida, Camadas Convolucionais, aplicam filtros (*kernels*) que varrem a imagem para extrair características locais, como bordas e texturas. Matematicamente, a operação de convolução bidimensional aplicada a uma imagem de entrada I utilizando um *kernel* K pode ser representada por:

$$S(i,j) = \sum_m \sum_n I(i+m, j+n)\, K(m,n) \tag{2.5}$$

Em que $S(i,j)$ representa o valor produzido na posição (i,j) do mapa de características resultante. Após a convolução, as Camadas de Ativação introduzem não linearidade no modelo. Entre as funções mais utilizadas está a *Rectified Linear Unit* (ReLU), definida por:

$$f(x) = \max(0, x) \tag{2.6}$$

Essa função preserva valores positivos e substitui valores negativos por zero, contribuindo para reduzir problemas relacionados ao desaparecimento do gradiente durante o treinamento (Goodfellow; Bengio; Courville, 2016). Posteriormente, as Camadas de *Pooling* (ou Subamostragem), reduzem a dimensionalidade das representações enquanto preservam as informações mais relevantes, contribuindo para a invariância espacial. No caso do *Max Pooling*, o valor de saída corresponde ao maior elemento contido em uma região local do mapa de características:

$$y = \max_{(i,j)\in R} x_{ij} \tag{2.7}$$

Em que R representa a região analisada e x_{ij} os valores presentes nessa região. Por fim, as Camadas Totalmente Conectadas, integram as características extraídas para realizar a classificação final (Goodfellow; Bengio; Courville, 2016).

O treinamento das RNCs é realizado por meio do algoritmo de *backpropagation*, o qual ajusta os pesos das conexões para minimizar o erro entre as previsões do modelo e os rótulos reais.

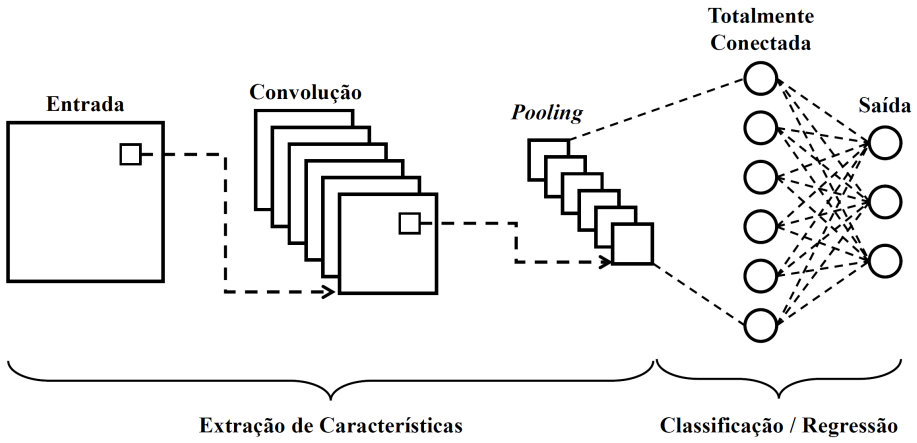
Entre as principais características das RNCs está sua capacidade de aprendizagem hierárquica, na qual camadas mais profundas capturam características progressivamente mais abstratas. Elas também são localmente invariantes à translação, o que significa que podem reconhecer padrões mesmo quando deslocados ligeiramente dentro da imagem. Devido à sua robustez e escalabilidade com grandes conjuntos de dados e arquiteturas profundas, as RNCs são amplamente utilizadas em aplicações práticas, como reconhecimento facial, diagnóstico por imagem médica, direção autônoma, entre outras (Krizhevsky; Sutskever; Hinton, 2012).

Esse método de ML oferece diversas vantagens, incluindo desempenho superior em tarefas visuais, menor necessidade de pré-processamento manual dos dados e forte capacidade de generalização quando treinadas com dados representativos. Entretanto, também apresentam algumas limitações, como altas exigências computacionais, necessidade de grandes volumes de dados rotulados e, dificuldade de interpretar as decisões do modelo. Algumas alternativas são propostas para mitigar essas limitações, como o uso de modelos pré-treinados em grandes bases de dados, realizando ajustes (*fine-tuning*) para tarefas específicas, podendo reduzir a necessidade de extensos conjuntos de dados rotulados. Além disso, técnicas como *dropout* e *batch normalization* ajudam a prevenir *overfitting* e melhorar a generalização do modelo (Litjens *et al.*, 2017).

A Figura 8 ilustra o fluxo sequencial dessa técnica, evidenciando sua divisão em duas etapas principais: a extração de características e a classificação propriamente dita.

Observa-se com a interpretação da figura que a primeira etapa ("Extração de Características") inicia-se com a inserção do bloco de "Entrada", composto pela matriz de dados bidimensional. Sobre essa entrada, são aplicadas operações na camada de "Convolução", onde filtros percorrem a matriz para gerar mapas capazes de identificar padrões

Figura 8 – Exemplo esquemático de uma arquitetura básica de uma RNC.



Fonte: Adaptado de Phung e Rhee (2019).

locais e texturas. Em seguida, os dados gerados passam pela camada de "*Pooling*", responsável por reduzir a dimensionalidade espacial dos mapas de características, diminuindo o custo computacional e retendo apenas as informações mais relevantes. A segunda etapa, denominada "Classificação / Regressão", recebe os mapas compactados vindos do processo anterior. Esses dados são achatados (*flattened*) para alimentar os neurônios da camada "Totalmente Conectada" (*dense layer*). Nessa região, a rede combina os atributos extraídos para realizar o aprendizado estatístico final e mapear as conexões neurais necessárias. O fluxo encerra-se no bloco de "Saída", onde a rede calcula a probabilidade para cada uma das categorias e entrega o resultado definitivo da classificação.

2.2.4 Métricas de Desempenho

As métricas de desempenho são utilizadas para avaliar a capacidade de um modelo em realizar predições corretas e para os classificadores elas são, em sua maioria, obtidas a partir da matriz de confusão. Embora o termo *confusion matrix* tenha sido introduzido originalmente por Miller e Nicely (1955) no contexto de estudos sobre confusões perceptuais entre sons da fala, essa representação foi posteriormente incorporada a diferentes áreas relacionadas à classificação automática. No contexto de *machine learning*, a matriz de confusão constitui uma ferramenta fundamental para a análise dos resultados de classificadores, pois permite analisar a relação entre as classes reais e as classes preditas pelo modelo e identificar não apenas os acertos, mas também os diferentes padrões de erro

(Miller; Nicely, 1955; Sokolova; Lapalme, 2009).

Em uma classificação com N classes, a matriz de confusão é representada por uma matriz quadrada $C \in \mathbb{R}^{N \times N}$, na qual cada elemento C_{ij} corresponde ao número de amostras cuja classe verdadeira é i e que foram classificadas pelo modelo como pertencentes à classe j (Pedregosa *et al.*, 2011).

Na representação gráfica, as linhas correspondem às classes reais, enquanto as colunas representam as classes previstas. Dessa forma, os elementos da diagonal principal representam as classificações corretas, uma vez que a classe real e a classe predita são iguais. Os elementos localizados fora da diagonal representam erros de classificação, indicando que uma amostra pertencente a uma determinada classe foi atribuída incorretamente a outra classe (Pedregosa *et al.*, 2011). Uma matriz de confusão genérica C pode ser expressa da seguinte forma:

$$C = \begin{bmatrix} C_{11} & C_{12} & \cdots & C_{1N} \\ C_{21} & C_{22} & \cdots & C_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ C_{N1} & C_{N2} & \cdots & C_{NN} \end{bmatrix} \tag{2.8}$$

em que C_{ij} representa a quantidade de amostras pertencentes à classe i que foram classificadas como pertencentes à classe j. Assim, quando $i = j$, o elemento pertence à diagonal principal e representa uma classificação correta. Quando $i \neq j$, o elemento representa uma classificação incorreta.

A interpretação da matriz de confusão permite identificar os acertos e os diferentes tipos de erro do classificador. Para cada classe analisada individualmente, considerando-a como positiva e as demais como negativas, as amostras corretamente classificadas correspondem aos verdadeiros positivos (VP). As amostras pertencentes à classe, mas atribuídas a outras classes, correspondem aos falsos negativos (FN), enquanto as amostras de outras classes classificadas como pertencentes à classe analisada representam os falsos positivos (FP). As demais amostras, corretamente identificadas como não pertencentes à classe analisada, correspondem aos verdadeiros negativos (VN).

Em problemas de classificação multiclasses, a matriz de confusão pode ser analisada individualmente para cada classe. Essa representação permite derivar métricas específicas para cada classe, como precisão, sensibilidade e *F1-score*, possibilitando uma avaliação mais detalhada do desempenho do classificador em comparação com a métrica global.

Além da análise dos valores absolutos, a matriz de confusão pode ser normalizada em relação ao número de amostras de cada classe. Essa normalização expressa os valores como proporções ou percentuais, facilitando a comparação do desempenho entre classes que possuem diferentes quantidades de amostras. Essa abordagem é particularmente útil em conjuntos de dados desbalanceados, nos quais os valores absolutos podem favorecer visualmente as classes com maior número de exemplos (Pedregosa *et al.*, 2011).

Os elementos da matriz de confusão constituem, portanto, a base para o cálculo de diferentes métricas de desempenho. A acurácia (*accuracy*) representa a proporção de classificações corretas realizadas pelo modelo em relação ao número total de amostras avaliadas, conforme demonstrado:

$$Acurácia = \frac{VP + VN}{VP + VN + FP + FN} \tag{2.9}$$

Essa métrica fornece uma visão geral do desempenho do classificador, sendo amplamente utilizada em problemas de classificação com conjuntos de dados balanceados. Valores elevados de acurácia indicam que o modelo apresenta boa capacidade de classificação global. Entretanto, essa métrica pode ser insuficiente quando aplicada a conjuntos de dados desbalanceados, pois um modelo pode apresentar alta acurácia mesmo tendo desempenho insatisfatório na identificação da classe minoritária. Por esse motivo, é comum que a acurácia seja analisada em conjunto com outras métricas, como precisão, sensibilidade e *F1-score*, proporcionando uma avaliação mais abrangente do desempenho do modelo (Hastie; Tibshirani; Friedman, 2009).

A métrica de precisão (*precision*) mede a confiabilidade das previsões positivas realizadas pelo modelo. É calculada pela razão entre o número de verdadeiros positivos e o total de previsões positivas realizadas pelo modelo:

$$Precisão = \frac{VP}{VP + FP} \tag{2.10}$$

A sensibilidade (*recall*) mede a capacidade do modelo de identificar corretamente todas as instâncias positivas existentes. E é definida pela razão entre o número de verdadeiros positivos e o total de instâncias realmente positivas:

$$Sensibilidade = \frac{VP}{VP + FN} \tag{2.11}$$

Já o *F1-score* corresponde à média harmônica entre precisão e sensibilidade, equilibrando ambas as métricas e tornando-se útil em contextos com classes desbalanceadas, sendo calculado da seguinte forma:

$$F1 = 2 \cdot \frac{Precisão \cdot Sensibilidade}{Precisão + Sensibilidade} \tag{2.12}$$

3 METODOLOGIA

Neste capítulo são descritas as etapas metodológicas do trabalho, divididas em duas seções. A Seção 3.1 descreve as características do conjunto de dados experimentais utilizado como base para o trabalho. Na Seção 3.2 são apresentados os processamentos e aplicação dos dados em cada uma das técnicas de aprendizado de máquina.

3.1. *DATASET*

Neste trabalho, utiliza-se um conjunto de dados de pressão de acesso aberto, fornecido por um grupo de pesquisadores do laboratório *Quality-of-Life Technology* (QoLT) da Universidade do Texas, publicado no repositório *PhysioNet* (Pouyan *et al.*, 2017).

Para o experimento em questão, usou-se o equipamento Vista Medical FSA Soft-Flex 2048, composto por uma matriz de sensores de pressão piezoresistivos de 32×64, que apresenta valores de saída em uma faixa de 0 a 1000. A Figura 9 mostra um tapete de pressão similar ao modelo escolhido.

Figura 9 – Exemplo de tapete de pressão com matriz de sensores piezoresistivos.



Fonte: https://tinyurl.com/7zn9aj2h.

O procedimento envolve o posicionamento do tapete de pressão sobre um colchão comum e coleta de aproximadamente dois minutos de sinal para cada posição selecionada, com uma taxa de amostragem de 1 Hz. São coletados dados de pressão de treze participantes em oito posturas padrão e nove posições adicionais. As informações antropométricas dos participantes são exibidas pela Tabela 2, enquanto os tipos de decúbitos avaliados e as variações experimentais são apresentados pela Tabela 3.

Como pode-se observar, dentre as 17 posturas listadas, quatro são posições com apoio sobre o lado direito do corpo dos participantes, quatro sobre o ladro esquerdo e nove

Tabela 2 – Dados antropométricos dos participantes.

Índice	**Idade**	**Altura (cm)**	**Peso (kg)**
1	19	175	87
2	23	183	85
3	23	183	100
4	24	177	70
5	24	172	66
6	26	169	83
7	27	179	96
8	27	186	63
9	30	174	74
10	30	174	79
11	30	176	91
12	33	170	78
13	34	174	74

Fonte: Adaptado de Pouyan *et al.* (2017).

Tabela 3 – Decúbitos e condições experimentais.

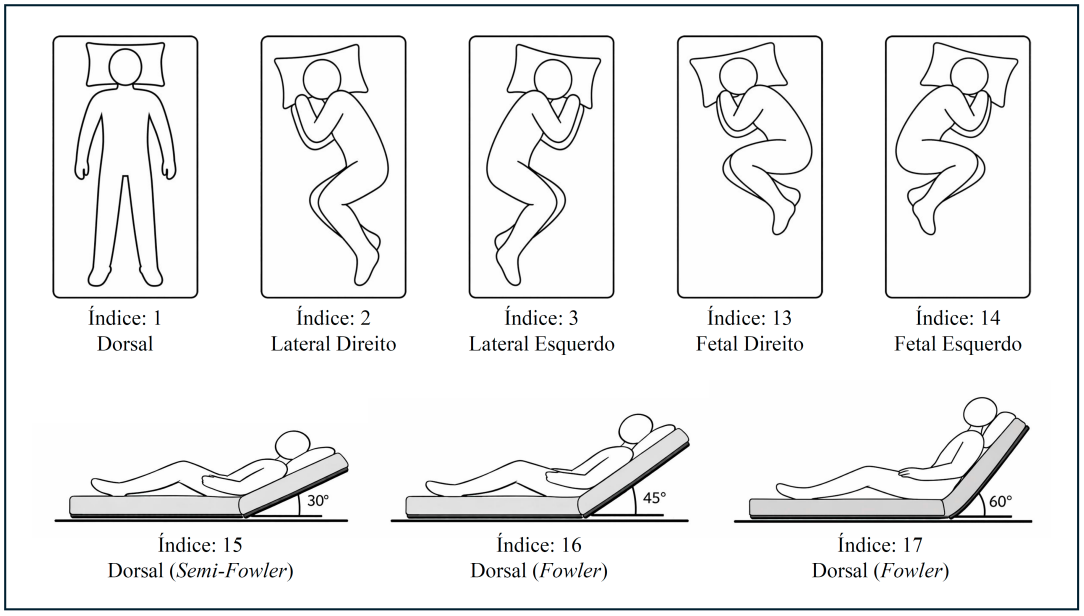
Índice	**Decúbito**	**Inclinação**	**Rotação Corporal**
1	Dorsal	0°	0°
2	Lateral Direito	0°	0°
3	Lateral Esquerdo	0°	0°
4	Lateral Direito	0°	30° (1 calço)
5	Lateral Direito	0°	60° (2 calços)
6	Lateral Esquerdo	0°	30° (1 calço)
7	Lateral Esquerdo	0°	60° (2 calços)
8	Dorsal	0°	0°
9	Dorsal	0°	0°
10	Dorsal (Dorsal Recumbente)	0°	0°
11	Dorsal	0°	0°
12	Dorsal	0°	0°
13	Fetal Direito	0°	0°
14	Fetal Esquerdo	0°	0°
15	Dorsal (*Semi-Fowler*)	30°	0°
16	Dorsal (*Fowler*)	45°	0°
17	Dorsal (*Fowler*)	60°	0°

Fonte: Adaptado de Pouyan *et al.* (2017).

em decúbito dorsal (ou supino), apresentando diversas variações, como adição de rotação corporal por meio de calços, reposicionamento de membros superiores e/ou inferiores, além de inclinações de tronco em 30°, 45° e 60°. Para maior entendimento das posturas mencionadas, têm-se as Figuras 10 e 11 que ilustram cada uma delas.

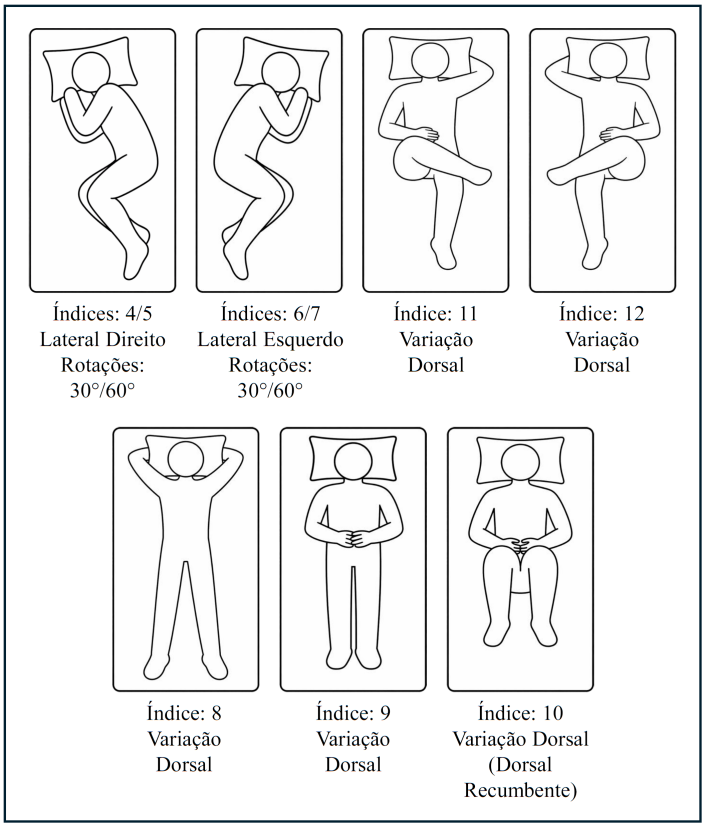
A partir dos dados originais, plotam-se mapas de pressão referentes a cada segundo de coleta, para todas as posições de cada um dos participantes. Dessa forma, obtêm-se aproximadamente 120 *frames* por postura. A seguir, têm-se exemplos de alguns dos *frames*

Figura 10 – Posturas padrão.



Fonte: Elaborado pelo autor.

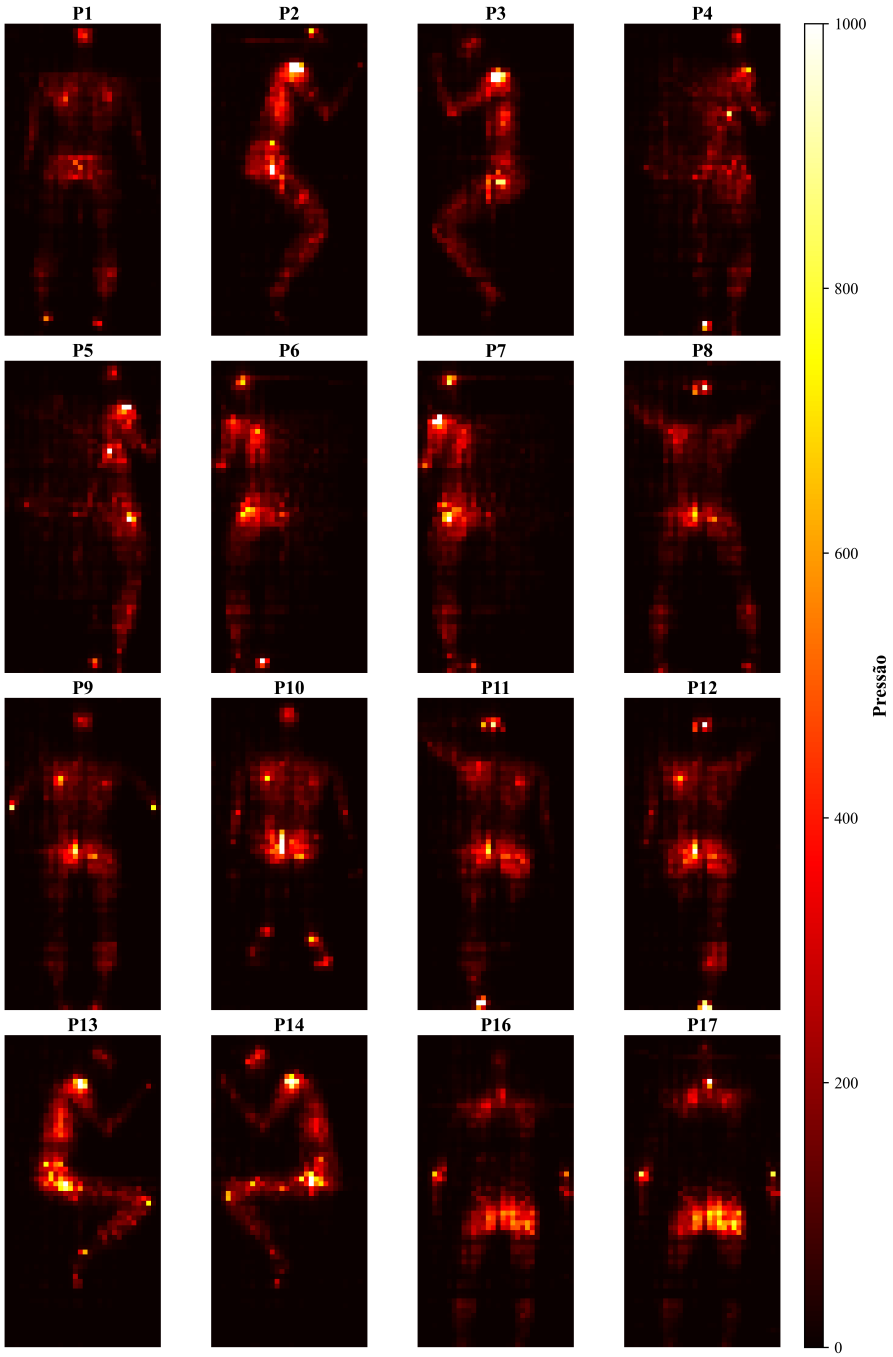
Figura 11 – Variações de posturas.



Fonte: Elaborado pelo autor.

gerados a partir do banco de dados apresentados pela Figura 12.

Figura 12 – Exemplos de mapas de pressão gerados a partir do *dataset*.



Fonte: Elaborado pelo autor.

Após a visualização dos dados, são selecionados *frames* para descarte, uma vez que se observa a falta de estabilização do sistema de sensores no início das coletas. Nos instantes iniciais as capturas contém, em quase sua totalidade, valores estáticos e, por essa razão, esses *frames* são excluídos do banco de dados.

Assim, com os dados inicialmente organizados, é possível avançar para a etapa

de processamento das informações para cada algoritmo de aprendizado de máquina. As próximas seções explicitam cada um dos métodos utilizados no trabalho como ferramentas para o avanço da pesquisa até seu objetivo final.

Além disso, o trabalho é realizado em um laptop equipado com um processador Ryzen 7-5800H operando a 3,2 GHz e 24 GB de RAM DDR4, sem o uso de GPU dedicada para o treinamento, com sistema operacional Windows 11. Todas as etapas, desde o carregamento e pré-processamento dos dados, até o treinamento e avaliação, são executadas em Python, com o suporte das bibliotecas Scikit-Learn, TensorFlow, Pandas, NumPy e Matplotlib.

3.2. APRENDIZADO DE MÁQUINA

Para todos os métodos de aprendizado de máquina testados, são apresentadas duas abordagens para a divisão do banco de dados, sendo a "Segmentação Aleatória" e a "Segmentação por Pacientes". A justificativa para essa divisão é melhor detalhada no Capítulo 4 e descreve cronologicamente as análises e decisões tomadas com base nas observações dos resultados encontrados. Esta Seção 3.2 é dedicada à descrição das características dos algoritmos implementados.

3.2.1 Floresta Aleatória

Durante a etapa de preparação de dados antes da implementação do algoritmo, é necessário realizar o processo de *feature extract*, que consiste na transformação de dados originais em um conjunto estruturado de variáveis descritivas que representem de forma compacta e discriminativa os padrões relevantes do problema. Para o atual trabalho são extraídas as seguintes métricas: pressão média, pressão máxima, coordenadas de pressão média, coordenadas de centro de pressão, percentual de área ocupada e, por fim, razão de aspecto. O Código A.2 utilizado nesta etapa é apresentado no Apêndice A.

Inicialmente, para primeira abordagem, o conjunto de dados é dividido aleatoriamente em 80% para treinamento e 20% para teste, utilizando uma semente fixa para garantir reprodutibilidade dos resultados. O modelo é configurado com 100 estimadores, ou seja, 100 árvores de decisão independentes são geradas e combinadas para realizar a classificação final. O uso de múltiplas árvores permite reduzir a variância do modelo,

tornando-o menos suscetível a sobreajuste. Após o particionamento dos dados, o modelo é treinado com os dados de entrada previamente extraídos e normalizados.

Já para a segunda abordagem, o conjunto de dados é segmentado por pacientes, utilizando a estratégia de *Leave-One-Subject-Out* (LOSO). Toda a estrutura do modelo é mantida a mesma porém, agora, faz-se uso da técnica de *k-folds*. Dessa forma, devido à presença de 13 pacientes no banco de dados, executam-se 13 ciclos de treinamento independentes, contando com 12 pacientes no conjunto de treino e 1 paciente remanescente no conjunto de teste. Com isso, o Código B.1 utilizado para implementação da Floresta Aleatória é apresentado no Apêndice B.

3.2.2 *K-Nearest Neighbors*

Para a etapa de preparação dos dados, anterior à implementação do algoritmo KNN, é realizado o mesmo processo de extração de características (*feature extract*) utilizado para a floresta aleatória. Sendo que os parâmetros extraídos também são os mesmos: pressão média, pressão máxima, coordenadas de pressão média, coordenadas de centro de pressão, percentual de área ocupada e razão de aspecto. O Código A.2 igualmente utilizado nesta etapa é apresentado no Apêndice A.

Para a primeira abordagem, segmenta-se os dados da mesma maneira descrita pela Seção 3.2.1, divididos aleatoriamente em 80% para referência e 20% para teste, utilizando uma semente fixa para garantir a reprodutibilidade dos resultados. Neste estudo, o valor de k é definido como 11, o que significa que a decisão de classe é tomada com base na maioria das classes entre os onze vizinhos mais próximos no espaço das *features*.

Já para a segunda abordagem, o conjunto de referência é composto por dados de 12 pacientes, enquanto os dados do paciente remanescente são separados para teste, novamente, aplicando *k-folds*. Além disso, a estrutura do algoritmo permanece inalterada. Dessa forma, o Código C.1 utilizado para implementação do classificador *K-Nearest Neighbors* é apresentado no Apêndice C.

3.2.3 Rede Neural Convolucional

Uma etapa de preparação de dados é feita antes da implementação do algoritmo. Como mencionado na caracterização de uma RNC, seus dados para a camada de entrada,

geralmente, são imagens representadas por matrizes de *pixels*. Com base nisso, nessa etapa realiza-se a compactação dos dados em um *dataframe* contendo todas as matrizes das imagens geradas para as dezessete posições, dos treze pacientes. O Código A.1 utilizado nesta etapa é apresentado no Apêndice A.

A rede desenvolvida foi construída com uma arquitetura sequencial composta por três camadas convolucionais intercaladas com camadas de *pooling*. A primeira camada convolucional possui dois filtros 3×3, função de ativação ReLU e recebe como entrada matrizes bidimensionais de dimensão 32×64, com um único canal (escala de cinza). Em seguida, aplica-se uma camada de *MaxPooling* 2×2, que reduz pela metade as dimensões espaciais da saída da convolução. O mesmo padrão se repete na segunda camada convolucional, com dois filtros e *pooling* 2×2. A terceira camada convolucional aumenta a profundidade para quatro filtros, mantendo o *kernel* 3×3 e ativação ReLU seguida, novamente, por *pooling* 2×2. Após as etapas convolucionais e de redução de dimensionalidade, a saída é achatada (*Flatten*) e ligada a uma camada totalmente conectada (*Dense*) com oito neurônios e ativação ReLU. A camada final é uma camada densa com função de ativação *Softmax*, cujo número de neurônios corresponde ao número de classes distintas no problema, permitindo a classificação multiclasses com base nas probabilidades de pertença a cada categoria. Esta configuração é escolhida por sua leveza computacional, adequada ao conjunto de dados processados, e por oferecer uma estrutura suficientemente profunda para capturar padrões espaciais relevantes nos mapas de pressão utilizados.

Durante a primeira abordagem, o treinamento da rede convolucional é conduzido utilizando o conjunto de dados previamente normalizado e codificado, dividido em subconjuntos de treinamento e teste na proporção de 80% e 20%, respectivamente, com uma semente fixa para garantir reprodutibilidade. Durante o treinamento, é aplicada a técnica de *early stopping* para prevenção de *overfitting*. O critério de monitoramento adotado é a perda da validação, com uma paciência de dez épocas, o que significa que o treinamento é interrompido caso não haja melhora na métrica monitorada por dez épocas consecutivas. Além disso, os melhores pesos da rede são restaurados automaticamente ao final do processo. A rede é treinada por até cinquenta épocas com um *batch size* de trinta e duas amostras, sendo 20% do conjunto de treinamento reservado internamente para validação, o que permite acompanhamento contínuo do desempenho da rede durante o aprendizado, sem exposição ao conjunto de teste.

Já para a segunda abordagem, a estrutura da rede e seus critérios continuam iguais,

porém a divisão do banco de dados é alterada. Utilizando a técnica de LOSO, têm-se os dados de 12 pacientes no conjunto de treino, sendo 20% desses dados reservados para validação, e *frames* de um paciente remanescente utilizado para teste. E, novamente, o processo de validação cruzada é aplicado com divisão em *k-folds* pelo número de pacientes disponíveis.

Posteriormente, para a busca de melhores hiperparâmetros mantem-se a rede com apenas uma camada densa (*dense layer*). A estrutura base é montada com 3 camadas convolucionais em sequência, utilizando filtros de tamanho fixo em 3×3. A partir dessa topologia, são definidos dez modelos candidatos (de M01 a M10) para avaliar o impacto do espessamento da rede. Nessa escala, a quantidade de filtros por camada variou de $[2, 2, 4]$ no modelo mais simples até $[128, 256, 512]$ na configuração mais complexa, enquanto o número de neurônios na camada densa escalou de 8 a 512. Os demais parâmetros, como funções de ativação, taxa de aprendizado (*learning rate*) e operações de subamostragem (*maxpooling*), permanecem inalterados. Em um primeiro momento, o procedimento é feito para 17 classes e, posteriormente, para 8 classes. A Tabela 4 apresenta a estrutura de cada um dos modelos e o Código D.1 utilizado para implementação dessa rede é apresentado no Apêndice D.

Tabela 4 – Modelos para busca de melhores hiperparâmetros para RNC.

Modelo	**Filtros**	**Camada Densa**	**Nº de parâmetros treináveis**	
			17 classes	**8 classes**
M01	[2,2,4]	8	603	522
M02	[4,8,16]	16	5157	5004
M03	[8,16,32]	32	19 289	18 992
M04	[16,32,64]	64	75 537	74 952
M05	[32,64,128]	64	222 865	222 280
M06	[32,64,128]	128	322 321	321 160
M07	[64,128,256]	128	872 273	871 112
M08	[64,128,256]	256	1 267 473	1 265 160
M09	[128,256,512]	256	3 449 617	3 447 304
M10	[128,256,512]	512	5 027 345	5 022 728

Fonte: Elaborado pelo autor.

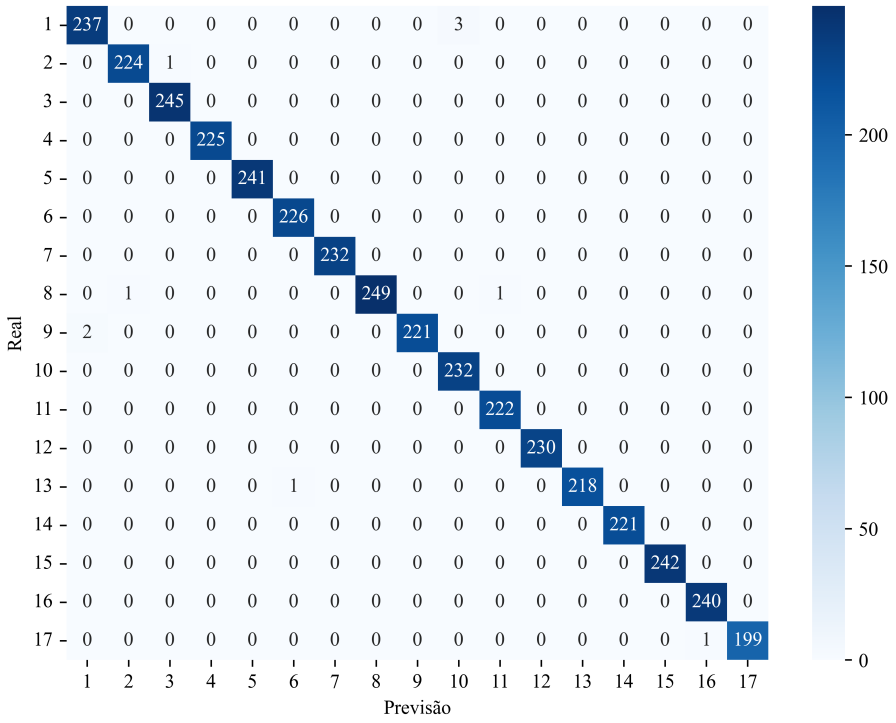
4 RESULTADOS E DISCUSSÕES

Neste capítulo são apresentados os resultados obtidos da aplicação de técnicas de aprendizado de máquina para a classificação de decúbitos, analisando e comparando as métricas de desempenho, acurácia e características gerais de cada uma das ferramentas utilizadas. Na Seção 4.1 são apresentados os resultados da primeira abordagem, que utiliza a separação aleatória do banco de dados para treinamento e teste. A Seção 4.2 aborda os resultados obtidos com a segmentação dos dados estruturada por paciente. Por fim, a Seção 4.3 sintetiza as discussões entre as abordagens apresentadas.

4.1. PRIMEIRA ABORDAGEM: SEGMENTAÇÃO ALEATÓRIA

Após a etapa de treinamento dos três métodos utilizando a primeira abordagem, os dados separados para teste são utilizados para avaliar o desempenho individual de cada algoritmo. A seguir, são apresentadas as matrizes de confusão para cada modelo, fornecendo uma representação visual das respectivas predições. Inicialmente, a Figura 13 ilustra a matriz de confusão obtida pelo algoritmo de Floresta Aleatória.

Figura 13 – Matriz de Confusão – Floresta Aleatória.

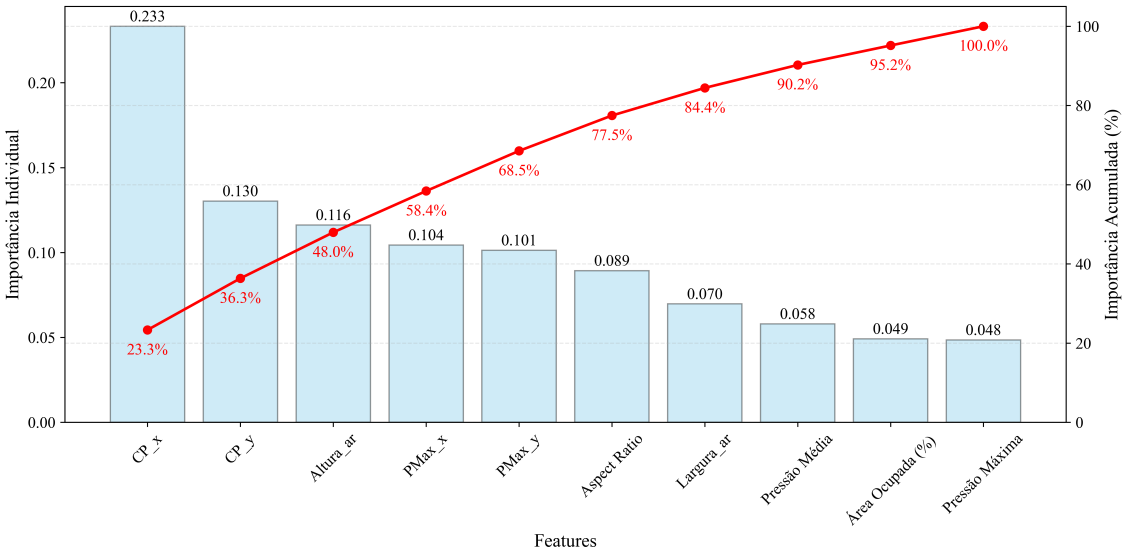


Fonte: Elaborado pelo autor.

A acurácia do primeiro modelo atinge 99,74% e, a partir da matriz de confusão apresentada, observa-se um desempenho geral positivo na tarefa de classificação multiclasse, com predominância de previsões corretas ao longo da diagonal principal. Os erros isolados, caracterizados por classificações incorretas entre algumas classes, sugerem uma sobreposição de características nas variáveis extraídas; no entanto, mantém um comportamento consistente na maioria das classes.

Além da matriz de confusão, o algoritmo de RF possibilita a análise das características (*features*) mais relevantes para a construção das árvores de decisão, as quais estão diretamente associadas à importância da informação para o processo de classificação. Para visualizar essa relevância, é possível utilizar o diagrama de Pareto apresentado na Figura 14.

Figura 14 – Diagrama de Pareto de Relevância de *Features* – Floresta Aleatória.

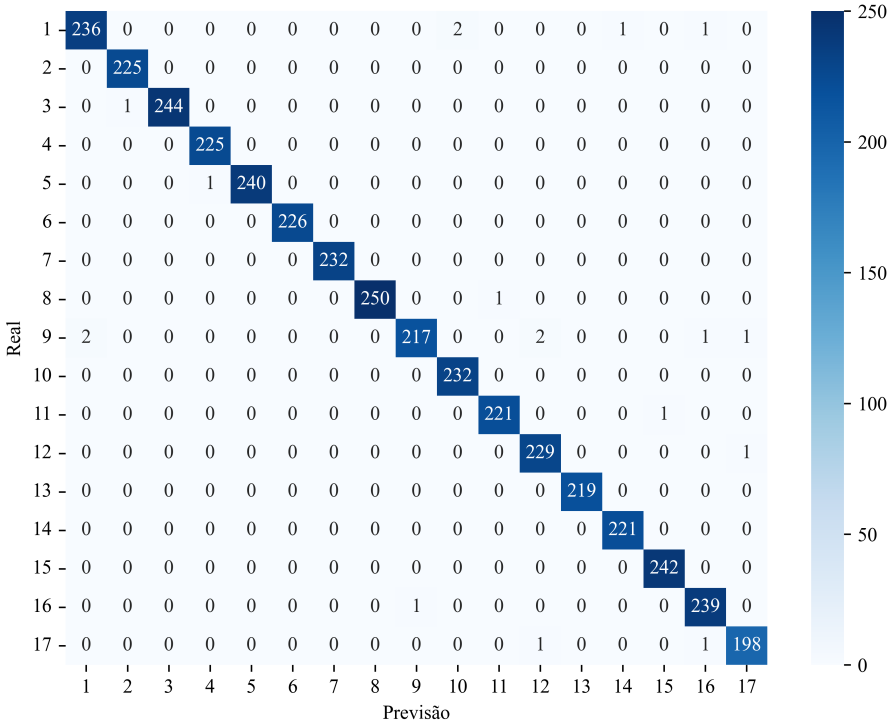


Fonte: Elaborado pelo autor.

Analisando o diagrama de Pareto, constata-se um maior destaque para as duas primeiras *features*, "CP_x" e "CP_y", que descrevem a posição do centro de pressão de cada *frame* e, juntas, totalizam uma importância acumulada de aproximadamente 36%. Esse percentual significativo corrobora com as expectativas, visto que o centro de pressão carrega informações espaciais relevantes a respeito da distribuição geométrica do corpo do paciente para as diferentes configurações de decúbito.

Em relação ao método KNN, a matriz de confusão correspondente é apresentada na Figura 15.

Figura 15 – Matriz de Confusão – *K-Nearest Neighbors*.



Fonte: Elaborado pelo autor.

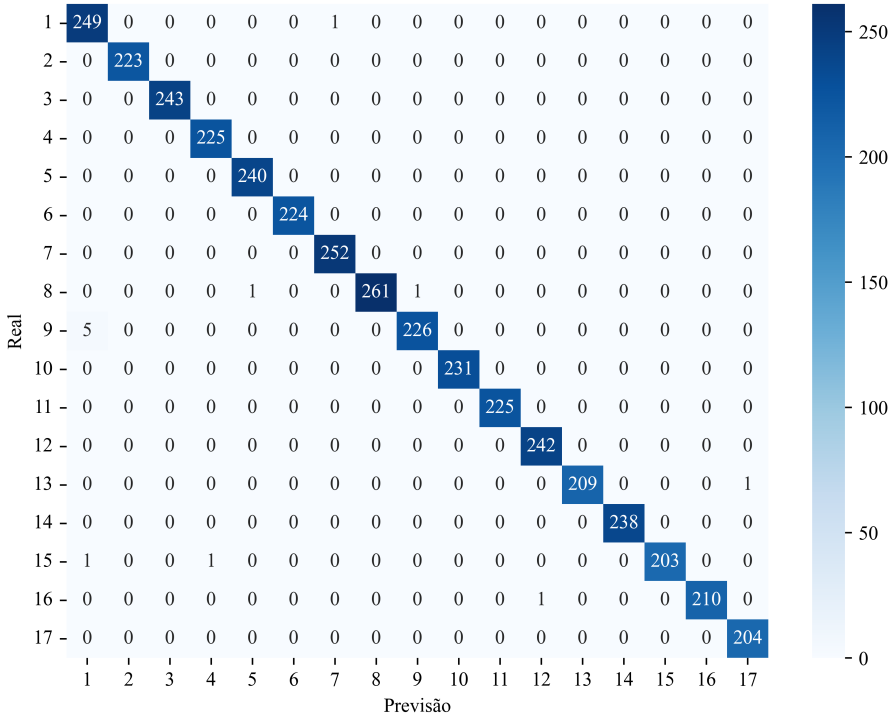
Identifica-se um desempenho igualmente consistente e equilibrado na tarefa de classificação para este segundo modelo. Assim como observado no caso anterior da RF, a maior parte das predições concentra-se de forma nítida ao longo da diagonal principal da matriz de confusão. Essa distribuição denota um elevado nível de acurácia entre as classes analisadas, a qual atinge 99,54%.

Por fim, apresenta-se na Figura 16 a matriz de confusão referente à Rede Neural Convolucional.

Com base nos resultados expostos, o modelo convolucional demonstra um bom desempenho na tarefa de classificação. Sua acurácia final no conjunto de teste é de 99,69%, com as predições majoritariamente alinhadas na diagonal principal, o que certifica que a maior parte das amostras é correlacionada com sucesso às suas respectivas classes. A presença de apenas alguns elementos fora da diagonal principal reflete uma baixa taxa de erro, evidenciando que a RNC obtém sucesso em abstrair os padrões espaciais para as dezessete classes avaliadas, sem a necessidade de uma etapa prévia de extração manual de características.

Complementando a avaliação visual das matrizes de confusão, a Tabela 5 compila algumas métricas estatísticas para a comparação quantitativa entre as técnicas aplicadas,

Figura 16 – Matriz de Confusão – Rede Neural Convolucional.



Fonte: Elaborado pelo autor.

sendo reportados os valores de precisão, sensibilidade e *F1-score*.

A partir dos dados apresentados pela tabela, nota-se que os três algoritmos alcançam desempenhos semelhantes em todas as métricas, com valores de modo geral muito próximos de 1,00, o que indica que os erros dos modelos são mínimos. Contudo, a análise aprofundada desse cenário inicial indica que a divisão puramente aleatória do *dataset* tende a mascarar algumas limitações. Embora o particionamento randômico seja uma prática padrão na literatura de aprendizado de máquina (Goodfellow; Bengio; Courville, 2016), em contextos com volume limitado de dados e baixa variabilidade amostral, essa metodologia pode acarretar o fenômeno de vazamento de dados (*data leakage*) entre os subconjuntos de treino e teste (Litjens *et al.*, 2017). Como consequência direta, as métricas de performance são artificialmente superestimadas, não refletindo a capacidade adaptativa dos modelos diante de dados não vistos.

Dessa forma, a fim de realizar uma avaliação mais rigorosa e alinhada à aplicação real—que consiste na classificação de posições para pacientes inéditos no *dataset*—, analisa-se uma segunda abordagem, a qual realiza a segmentação do banco de dados por pacientes, sendo os resultados obtidos com essa metodologia apresentados na subseção seguinte.

Tabela 5 – Métricas de desempenho por classe para diferentes algoritmos (Segmentação Aleatória).

	Floresta Aleatória			**k-NN**			**RNC**		
Classe	Prec.	Rev.	F1	Prec.	Rev.	F1	Prec.	Rev.	F1
1	0.99	0.99	0.99	0.99	0.98	0.99	0.98	1.00	0.99
2	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
3	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
4	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
5	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
6	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
7	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
8	1.00	0.99	1.00	1.00	1.00	1.00	1.00	0.99	1.00
9	1.00	0.99	1.00	1.00	0.97	0.98	1.00	0.98	0.99
10	0.99	1.00	0.99	0.99	1.00	1.00	1.00	1.00	1.00
11	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
12	1.00	1.00	1.00	0.99	1.00	0.99	1.00	1.00	1.00
13	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
14	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
15	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.99	1.00
16	1.00	1.00	1.00	0.99	1.00	0.99	1.00	1.00	1.00
17	1.00	0.99	1.00	0.99	0.99	0.99	1.00	1.00	1.00

Fonte: Elaborado pelo autor.

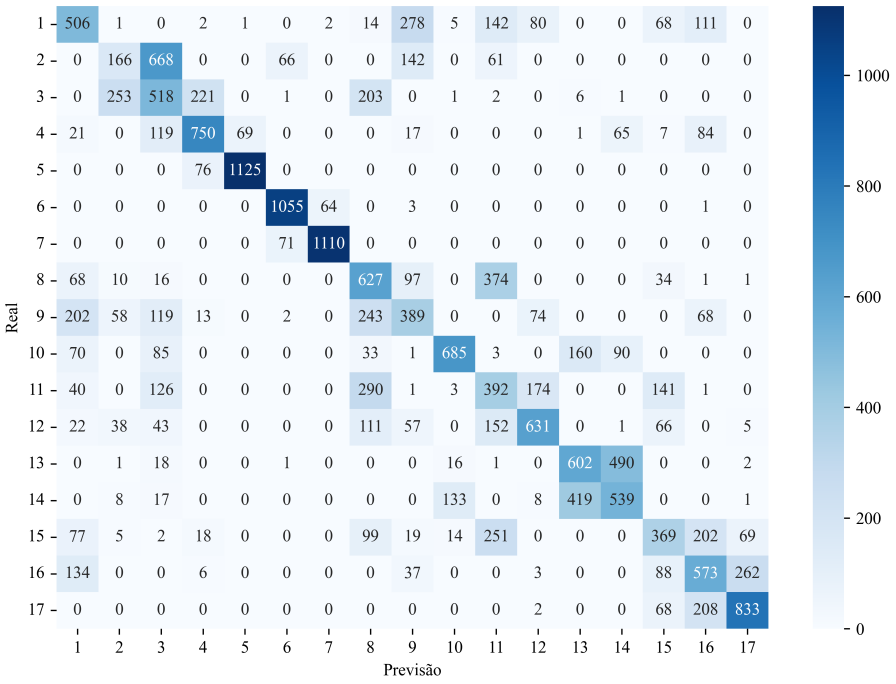
4.2. SEGUNDA ABORDAGEM: SEGMENTAÇÃO POR PACIENTE

Nesta seção, avaliam-se os resultados obtidos por meio de uma metodologia de validação mais rigorosa, fundamentada na estratégia *Leave-One-Subject-Out*. Diferente da abordagem anterior, este procedimento previne que amostras de um mesmo indivíduo apareçam simultaneamente nos conjuntos de treinamento e de teste, fornecendo uma análise mais realista sobre a capacidade de generalização dos classificadores para novos pacientes.

Conforme estabelecido, o processo de validação cruzada opera com a divisão em *k-folds* mapeada pelo número de indivíduos disponíveis. Contando com 13 pacientes mapeados no *dataset*, executam-se 13 ciclos de treinamento independentes para cada algoritmo, de forma que, a cada iteração, 12 pacientes compõem o conjunto de treino e o indivíduo remanescente atua como o conjunto de teste. Ao final, as matrizes de confusão resultantes de cada partição são sobrepostas para compor uma matriz de desempenho global cumulativa. As Figuras 17, 18 e 19 exibem esses resultados para os métodos RF, KNN e RNC, respectivamente.

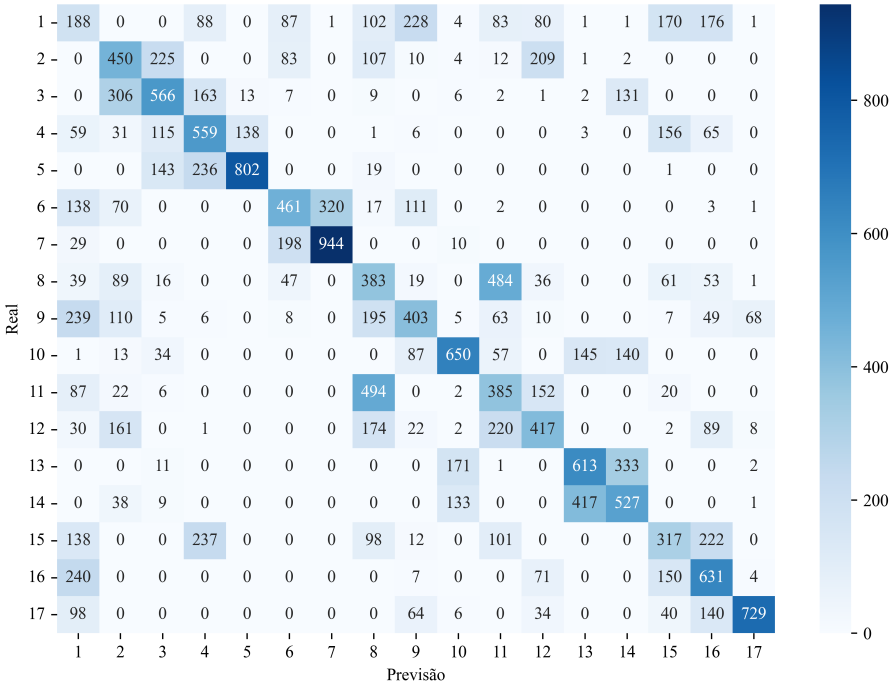
A análise comparativa entre as matrizes de confusão obtidas com LOSO revela tendências similares entre os três métodos avaliados. Em todos os casos, a diagonal

Figura 17 – Matriz de Confusão Global – Floresta Aleatória.



Fonte: Elaborado pelo autor.

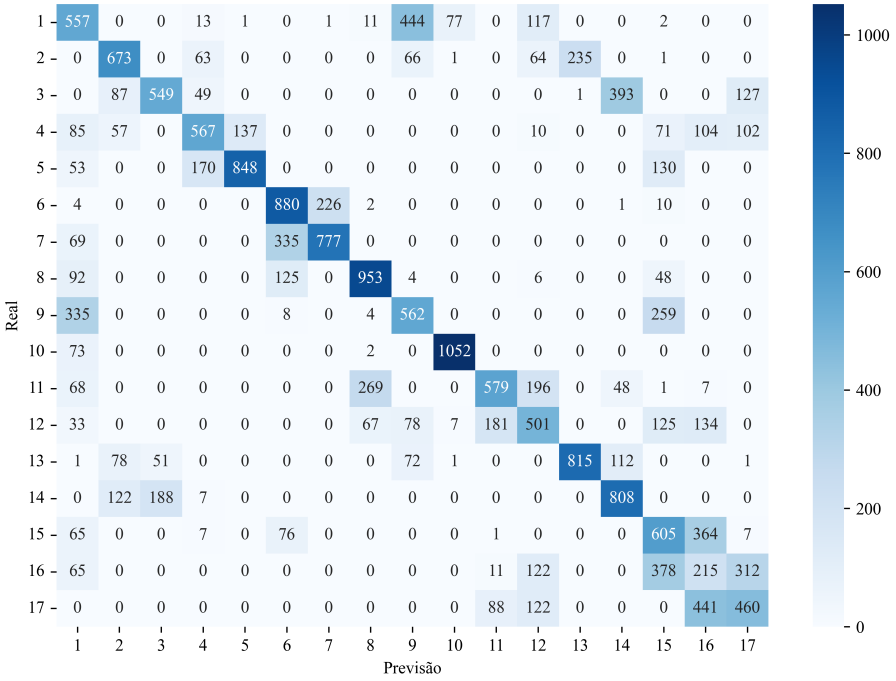
Figura 18 – Matriz de Confusão Global – *K-Nearest Neighbors*.



Fonte: Elaborado pelo autor.

principal permanece bem destacada, evidenciando que os algoritmos mantêm a capacidade de discriminação postural mesmo diante de pacientes inéditos. No entanto, nota-se um aumento considerável na ocorrência de classificações errôneas em comparação ao cenário

Figura 19 – Matriz de Confusão Global – Rede Neural Convolucional.



Fonte: Elaborado pelo autor.

de segmentação aleatória dos dados. Além de erros aleatórios e dispersos, observa-se a formação de agrupamentos nítidos de confusão em torno de blocos específicos da diagonal principal, sinalizando dificuldades no isolamento de determinadas posições.

Além disso, a equivalência de resultados entre os três modelos mostra que tanto as características extraídas manualmente para RF e KNN, quanto as *features* aprendidas automaticamente pela RNC diretamente dos mapas de pressão brutos, são eficazes. Nesse cenário, pode-se destacar uma ligeira vantagem da rede neural devido à sua maior acurácia global de 57,70%, superando o desempenho obtido pela RF, de 55,38% e, de forma mais expressiva, o resultado apresentado pelo KNN, de 45,76%.

Para fins de detalhamento numérico por classe sob validação cruzada (*GroupK-Fold*), as métricas de desempenho são apresentadas na Tabela 6.

A análise conjunta das métricas indica um comportamento equilibrado entre a acurácia global e o $F1$-*score* para os modelos RF e RNC, o que denota estabilidade interna diante da distribuição balanceada das classes no *dataset*. No caso do algoritmo RF, a acurácia e o $F1$ apresentam valores muito próximos. O KNN, por outro lado, obtém o menor desempenho preditivo geral, acompanhado por uma sutil distorção na precisão média que sugere maior tendência à geração de falsos positivos. Já a RNC consolida a melhor performance entre os três métodos em todas as métricas globais e de validação

Tabela 6 – Métricas de desempenho por classe para diferentes algoritmos com validação cruzada (Segmentação por Paciente).

	Floresta Aleatória			**k-NN**			**RNC**		
Classe	Prec.	Rev.	F1	Prec.	Rev.	F1	Prec.	Rev.	F1
1	0.44	0.42	0.43	0.15	0.16	0.15	0.37	0.46	0.41
2	0.31	0.15	0.20	0.35	0.41	0.38	0.66	0.61	0.63
3	0.30	0.43	0.35	0.50	0.47	0.48	0.70	0.46	0.55
4	0.69	0.66	0.68	0.43	0.49	0.46	0.65	0.50	0.56
5	0.94	0.94	0.94	0.84	0.67	0.74	0.86	0.71	0.78
6	0.88	0.94	0.91	0.52	0.41	0.46	0.62	0.78	0.69
7	0.94	0.94	0.94	0.75	0.80	0.77	0.77	0.66	0.71
8	0.39	0.51	0.44	0.24	0.31	0.27	0.73	0.78	0.75
9	0.37	0.33	0.35	0.42	0.35	0.38	0.46	0.48	0.47
10	0.80	0.61	0.69	0.65	0.58	0.61	0.92	0.93	0.93
11	0.28	0.34	0.31	0.27	0.33	0.30	0.67	0.50	0.57
12	0.65	0.56	0.60	0.41	0.37	0.39	0.44	0.44	0.44
13	0.51	0.53	0.52	0.52	0.54	0.53	0.78	0.72	0.75
14	0.45	0.48	0.47	0.46	0.47	0.47	0.59	0.72	0.65
15	0.44	0.33	0.38	0.34	0.28	0.31	0.37	0.54	0.44
16	0.46	0.52	0.49	0.44	0.57	0.50	0.17	0.19	0.18
17	0.71	0.75	0.73	0.89	0.66	0.76	0.46	0.41	0.43
Média	0.56	0.55	0.55	0.48	0.46	0.47	0.60	0.58	0.59
Desvio	0.23	0.23	0.23	0.20	0.16	0.18	0.20	0.18	0.18

Fonte: Elaborado pelo autor.

cruzada avaliadas.

A avaliação individualizada por classe revela assimetrias importantes no comportamento dos modelos. Algumas posições apresentam excelente responsividade preditiva, com destaque para determinadas classes nos modelos RF e RNC que alcançam altos índices de $F1$-*score*. Em contrapartida, certas posições dificultam o desempenho dos classificadores, resultando em métricas relativamente baixas, especialmente no KNN e na RF. Quanto à estabilidade ao longo das 17 classes, expressa pelo desvio padrão, o modelo de Floresta Aleatória exibe a maior dispersão nos resultados, enquanto a RNC apresenta oscilações mais moderadas e consistentes entre as classes.

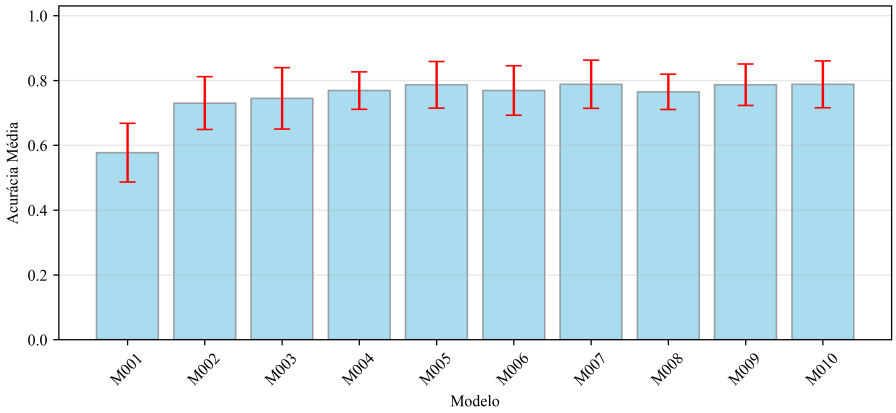
Com base nesse panorama, determina-se a seleção da Rede Neural Convolucional para a continuidade do estudo. Esta decisão fundamenta-se em três pilares: primeiro, a RNC obtém o maior índice de acurácia global absoluta; segundo, por tratar-se essencialmente de um problema de classificação de padrões baseados em arranjos bidimensionais de pressão (análogos a imagens), a arquitetura convolucional alinha-se intrinsecamente à natureza física do problema; terceiro, o modelo elimina a dependência de processos manuais de extração de atributos (*feature extracting*), automatizando o aprendizado de

características hierárquicas.

4.2.1 Otimização de Hiperparâmetros da RNC (17 Posições)

Definida a RNC como a abordagem escolhida para continuidade do estudo, inicia-se a busca pela otimização de seus hiperparâmetros estruturais. Com o propósito de delimitar o espaço de busca e reduzir o custo computacional, restringe-se a arquitetura a uma única camada densamente conectada (*dense layer*), não sendo consideradas arquiteturas profundas. Define-se uma topologia base composta por três camadas convolucionais sequenciais, com filtros de dimensão fixa de 3×3, variando-se exclusivamente o número de filtros por camada, em uma faixa entre 2 e 512. Paralelamente, o número de neurônios na camada densa varia de 8 a 512. Variáveis como a taxa de aprendizado (*learning rate*), as funções de ativação e as operações de subamostragem (*maxpooling*) mantêm-se as mesmas da abordagem anterior. A Figura 20 apresenta o desempenho e o desvio padrão associados aos 10 modelos mais representativos obtidos nesse mapeamento.

Figura 20 – Comparação de acurácia e desvio padrão para 10 Modelos de RNC (17 Classes).

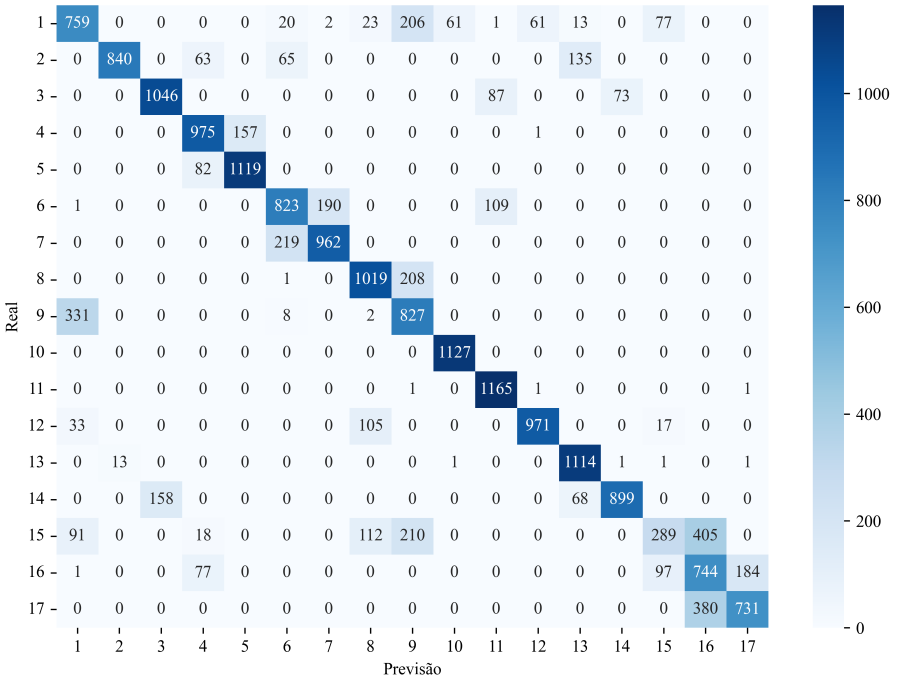


Fonte: Elaborado pelo autor.

A partir do perfil comparativo ilustrado na Figura 20, evidencia-se um comportamento de estabilização na métrica de acurácia à medida que se expande a quantidade de parâmetros treináveis dos modelos. A partir do segundo modelo, as taxas de acerto mantêm-se entre 70% e 80%, independente do aumento de parâmetros treináveis. Diante desse comportamento, o Modelo 5 consolida-se como o candidato ideal para a operação, uma vez que representa um ponto de equilíbrio, exibindo uma quantidade intermediária de parâmetros e mantendo elevada acurácia e um desvio padrão comparativamente re-

duzido. A Figura 21 retrata a matriz de confusão global acumulada do Modelo 5 sob a estratégia LOSO para as 17 classes.

Figura 21 – Matriz de Confusão Global (17 Classes) – RNC Otimizada (Modelo 5).



Fonte: Elaborado pelo autor.

A matriz de confusão apresenta uma diagonal principal melhor definida em relação às outras apresentadas anteriormente, as quais não possuem uma otimização dos hiperparâmetros. Outro ponto observado é a menor dispersão existente, embora seja possível identificar um padrão de agrupamento de erros em classes similares presentes nas duas matrizes das RNCs, da Figura 19 e da Figura 21. Ao voltar nas imagens referentes aos decúbitos (Figura 10 e Figura 11) e aos *frames* (Figura 12), pode-se verificar uma possível causa de certas confusões entre algumas classes específicas. Têm-se por exemplo, os conjuntos 4/5 e 6/7 muito próximos entre si e, por constituírem quatro variações dos decúbitos laterais (com rotação corporal), podem ser confundidos com as posturas padrões 2 e 3, respectivamente. As posições 8, 9 e 10 também por serem variações da postura dorsal padrão, se confundem com a posição 1. Além disso, existem alguns erros referentes aos índices 15 e 16, que, apesar de não serem variações diretas (*Semi-Fowler* e *Fowler*), são também decúbitos dorsais com alterações em suas inclinações.

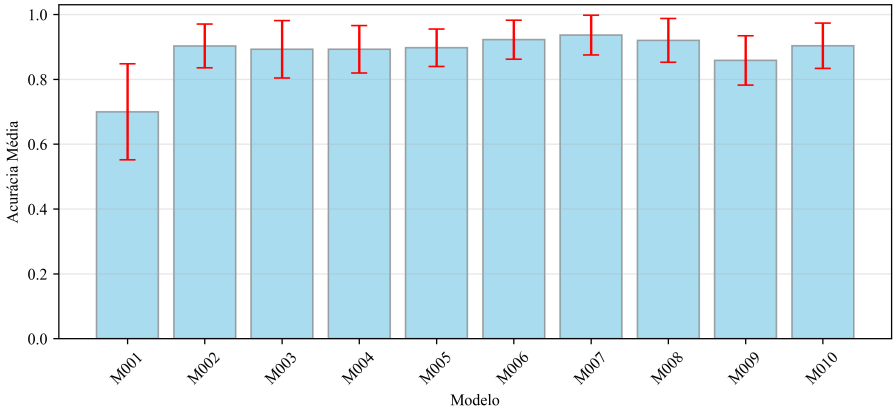
A partir dessa análise, propõe-se remover as posições identificadas como potenciais pontos de confusão para a rede neural. O próximo teste visa avaliar se a sobreposição de características entre as classes constitui o principal obstáculo para o modelo. Dessa

forma, a subseção seguinte apresenta o desempenho obtido com a nova configuração de apenas 8 posições de classificação.

4.2.2 Otimização de Hiperparâmetros da RNC (8 Posições)

A partir da nova organização com 8 posições para classificação, correspondendo aos índices 1, 2, 3, 11, 12, 13, 14 e 17; são apresentados os resultados obtidos para tal configuração. Inicialmente, realiza-se novamente uma busca por melhores hiperparâmetros, da mesma maneira da subseção anterior, otimizando parâmetros diretamente relacionados à estruturação da rede, porém agora com apenas 8 classes. A Figura 22 ilustra a variação de desempenho ao longo dos modelos abordados.

Figura 22 – Comparação de acurácia e desvio padrão para 10 Modelos de RNC (8 Classes).

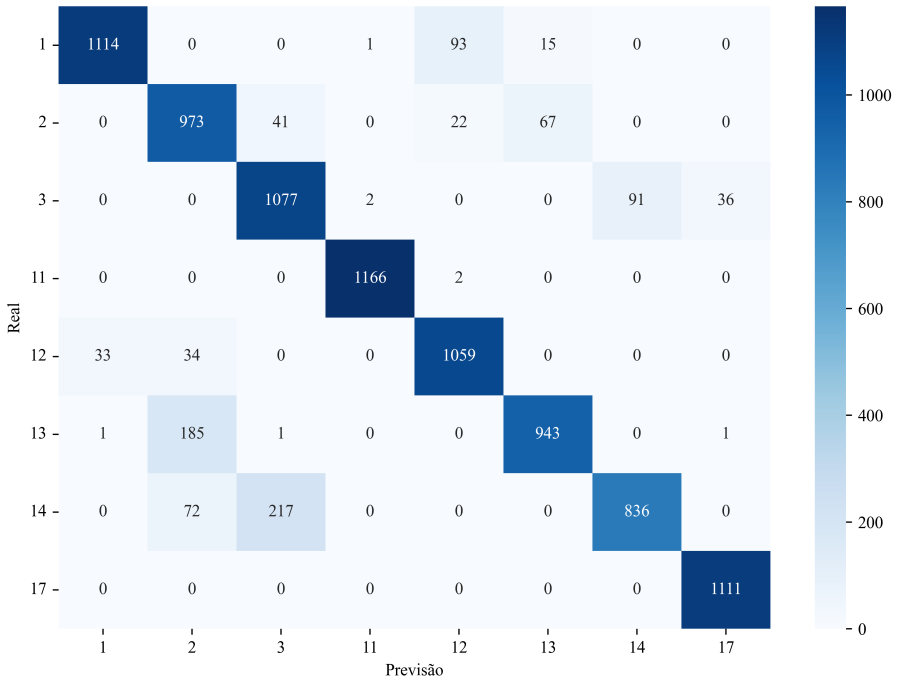


Fonte: Elaborado pelo autor.

Assim como ocorre no cenário de 17 classes, os índices de acurácia se estabilizam logo nos primeiros modelos testados. No entanto, a redução no número de classes gera um ganho visível nos resultados, elevando as acurácias médias para patamares próximos a 90%. Esse aumento possivelmente reforça a hipótese de que a maior dificuldade dos modelos está na sobreposição dos padrões de pressão entre classes parecidas. Buscando um equilíbrio entre simplicidade e desempenho, o Modelo 5 se mostra novamente uma escolha vantajosa, por conciliar uma alta acurácia com um desvio padrão baixo e uma rede de tamanho intermediário. A Figura 23 apresenta a matriz de confusão para essa configuração.

A análise da nova matriz de confusão global mostra uma melhora clara no padrão de acertos. A diagonal principal aparece bem definida e sem os antigos agrupamentos de

Figura 23 – Matriz de Confusão Global (8 Classes) – RNC Otimizada (Modelo 5).



Fonte: Elaborado pelo autor.

erros que dificultam a classificação. Os erros restantes são mais dispersos e reduzidos, o que indica um bom desempenho do classificador para aplicações práticas. Com isso, o Modelo 5 alcança uma acurácia global consolidada de 90,06%, mostrando-se uma alternativa promissora para a classificação de decúbitos e suas possíveis aplicações futuras.

4.3. COMPARAÇÃO ENTRE ABORDAGENS

Esta seção sintetiza as principais discussões abordadas pelo estudo, delineando um panorama comparativo entre as abordagens e a evolução dos modelos.

Inicialmente, os resultados da primeira abordagem indicam índices de acurácia global muito elevados para todos os algoritmos analisados, aproximando-se de 99%. Contudo, a análise crítica desse cenário sugere que a segmentação aleatória pode não ser a escolha mais adequada para a situação, devido à grande similaridade dos dados, podendo acarretar em um vazamento de dados entre os conjuntos de treinamento e teste. Como consequência, o desempenho dos modelos tende a ser superestimado, mascarando algumas limitações existentes.

Para uma abordagem mais criteriosa é utilizada a estratégia de *Leave-One-Subject-Out*. Ao testar os algoritmos com pacientes inéditos, observa-se uma redução notável nas

acurácias globais dos três métodos de ML. Apesar dessa redução, as matrizes de confusão ainda exibem uma diagonal principal visível, embora acompanhada por agrupamentos de erros concentrados em classes específicas. Nesse cenário, a RNC apresenta um desempenho inicial ligeiramente superior aos demais, além de possuir a vantagem de eliminar a necessidade de uma etapa independente para a extração manual de características do banco de dados.

Diante disso, a RNC é selecionada como a técnica para a continuidade da pesquisa. O processo de otimização de hiperparâmetros permite refinar a estrutura do modelo e obter resultados melhores que os anteriores. Entre as configurações mapeadas para o cenário de 17 classes, o Modelo 5 destaca-se como uma alternativa vantajosa, conciliando uma boa acurácia, em torno de 80%, utilizando uma quantidade reduzida de parâmetros treináveis em comparação aos demais modelos avaliados.

Por fim, o mapeamento dos erros remanescentes da RNC indica que as maiores dificuldades de classificação são possivelmente associadas à semelhança física entre certas posições e com a otimização de hiperparâmetros, para 8 posições, essa suposição é reforçada. Nessa etapa final, o Modelo 5 novamente se mostra adequado e os resultados indicam uma melhora considerável de acurácia, agora em torno de 90%. A nova matriz de confusão apresenta uma diagonal principal bem definida e minimiza significativamente os antigos agrupamentos de erros, posicionando a abordagem proposta como uma alternativa promissora para a classificação de decúbitos em aplicações futuras.

5 CONSIDERAÇÕES FINAIS

Conforme discutido no início deste estudo, a segurança do paciente e a mitigação de eventos adversos, como as lesões por pressão, constituem um dos maiores desafios da saúde pública. O desenvolvimento dessas lesões gera graves prejuízos clínicos aos indivíduos e impactos econômicos profundos às instituições hospitalares. Nesse contexto, o desenvolvimento de tecnologias assistivas atua como um pilar essencial para auxiliar as equipes de enfermagem no monitoramento contínuo e na gestão de riscos. Identifica-se também que a base para diversos sistemas automatizados de alertas, monitoramento ou redistribuição de pressão, está na capacidade de identificar a postura adotada pelo paciente no leito.

Alinhado a essa necessidade, o presente trabalho cumpre seu objetivo geral ao implementar e avaliar o desempenho de técnicas de *machine learning* voltadas à classificação de decúbitos, a partir de mapas de pressão reais. O estudo das metodologias demonstra que a segmentação puramente aleatória dos dados, embora comum, tende a superestimar o desempenho dos classificadores devido ao possível vazamento de dados. Em contrapartida, a segmentação por paciente oferece uma perspectiva mais realista e criteriosa para o cenário, expondo as dificuldades reais de generalização para indivíduos inéditos no sistema.

Nesse contexto, a Rede Neural Convolucional destaca-se em relação aos demais métodos, pois, além de alcançar um desempenho inicial superior, a arquitetura convolucional confirma sua aptidão para o problema de extrair características relevantes diretamente dos dados bidimensionais de pressão, eliminando processos manuais de *feature extration*. A etapa subsequente de otimização de hiperparâmetros consolida o modelo como uma solução promissora para o cenário de 17 classes, por equilibrar alta acurácia e uma estrutura relativamente enxuta de parâmetros treináveis.

Por fim, o mapeamento das similaridades físicas entre diferentes posições conduz uma nova otimização de hiperparâmetros, com as 8 classes mais relevantes. Essa reestruturação refina significativamente o padrão de acertos do modelo. Diante disso, conclui-se que o método desenvolvido se mostra promissor, estabelecendo uma base para o desenvolvimento de sistemas automatizados voltados à prevenção de lesões por pressão.

5.1. PRÓXIMAS ETAPAS

Como continuidade deste estudo e visando o aprimoramento da tecnologia proposta, listam-se algumas etapas para trabalhos futuros:

- **Mapeamento de regiões críticas e tomada de decisão:** Implementar um algoritmo capaz de identificar e segmentar as áreas críticas ao desenvolvimento de LPs, para cada posição classificada. A partir desse mapeamento, integrar um sistema de tomada de decisão para atuar no tipo de prevenção recomendada, seja estimar o tempo de permanência na posição e o risco de lesão, gerando alertas para a equipe de saúde, ou na própria atuação de um sistema de redistribuição de pressões;

- **Exploração de novas aplicações assistivas:** Expandir a aplicabilidade do modelo para além do monitoramento e emissão de alertas tradicionais. Sugere-se o estudo de sua integração com sistemas de atuação física, como colchões pneumáticos capazes de inflar ou desinflar células de ar de forma ativa e personalizada com base no decúbito identificado;

- **Adição de dados e contextualização do paciente:** Incorporar variáveis clínicas e características intrínsecas do paciente (como peso, índice de massa corporal, idade, entre outros) como entradas adicionais na arquitetura da RNC. Essa fusão de dados permite que a rede pondere a relevância dos mapas de pressão em conjunto com o perfil de vulnerabilidade do indivíduo, tornando o diagnóstico de risco mais preciso e individualizado.

REFERÊNCIAS

ALWASEL, A. *et al.* Bedsores Management: Efficiency Simulation of a New Mattress Design. **Healthcare**, v. 9, n. 12, p. 1701, dez. 2021. ISSN 2227-9032. Number: 12 Publisher: Multidisciplinary Digital Publishing Institute. Disponível em: ⟨https://www.mdpi.com/2227-9032/9/12/1701⟩.

BEDELL, S. E. *et al.* Incidence and characteristics of preventable iatrogenic cardiac arrests. **JAMA**, v. 265, n. 21, p. 2815–2820, jun. 1991. ISSN 0098-7484.

BERGQUIST-BERINGER, S. *et al.* The Reliability of the National Database of Nursing Quality Indicators Pressure Ulcer Indicator: A Triangulation Approach. **Journal of Nursing Care Quality**, v. 26, n. 4, p. 292, dez. 2011. ISSN 1057-3631. Disponível em: ⟨https://journals.lww.com/jncqjournal/abstract/2011/10000/the_reliability_of_the_national_database_of.2.aspx⟩.

BOUTEN, C. V. *et al.* The etiology of pressure ulcers: Skin deep or muscle bound? **Archives of Physical Medicine and Rehabilitation**, v. 84, n. 4, p. 616–619, abr. 2003. ISSN 00039993. Disponível em: ⟨https://linkinghub.elsevier.com/retrieve/pii/S0003999302048566⟩.

Brasil. **PORTARIA GM/MS nº 529, de 1º de abril de 2013 — Institui o Programa Nacional de Segurança do Paciente (PNSP) — Ministério da Saúde**. Brasília, DF: [s.n.], 2013. Diário Oficial da União: Seção 1. Disponível em: ⟨https://bvsms.saude.gov.br/bvs/saudelegis/gm/2013/prt0529_01_04_2013.html⟩.

Brasil. **RESOLUÇÃO DA DIRETORIA COLEGIADA DA ANVISA nº 36, de 25 de julho de 2013 — Institui ações para a Segurança do Paciente em Serviços de Saúde e dá outras providências — Agência Nacional de Vigilância Sanitária - Anvisa**. Brasília, DF: [s.n.], 2013. Diário Oficial da União: Seção 1. Disponível em: ⟨https://bvsms.saude.gov.br/bvs/saudelegis/anvisa/2013/rdc0036_25_07_2013.html⟩.

Brasil. **PROTOCOLO DE ÚLCERA POR PRESSÃO — Agência Nacional de Vigilância Sanitária - Anvisa ; Ministério da Saúde ; Fundação Oswaldo Cruz (Fiocruz)**. Brasília, DF: [s.n.], 2021. Disponível em: ⟨https://www.gov.br/anvisa/pt-br/centraisdeconteudo/publicacoes/servicosdesaude/publicacoes/protocolo-de-ulcera-por-pressao/view⟩.

Brasil. Boletim Segurança do Paciente e Qualidade em Serviços de Saúde nº 29: Incidentes Relacionados à Assistência à Saúde – 2014 a 2022. 2022. Disponível em: ⟨https://www.gov.br/anvisa/pt-br/centraisdeconteudo/publicacoes/servicosdesaude/boletins-e-relatorios-das-notificacoes-de-iras-e-outros-eventos-adversos-1/BR_2014__2022.pdf⟩.

Brasil. **NOTA TÉCNICA GVIMS/GGTES/Anvisa nº 05/2023 - Práticas de Segurança do Paciente em Serviços de Saúde: Prevenção de Lesão por Pressão — Agência Nacional de Vigilância Sanitária - Anvisa**. 2023. Disponível em: ⟨https://www.gov.br/anvisa/pt-br/centraisdeconteudo/publicacoes/servicosdesaude/notas-tecnicas/notas-tecnicas-vigentes/nota-tecnica-gvims-ggtes-anvisa-no-05-2023-praticas-de-seguranca-do-paciente-em-servicos-de-saude-prevencao-de-lesao-por-pressao/view⟩.

BREIMAN, L. Random Forests. **Machine Learning**, v. 45, p. 5–32, out. 2001.

BRUNZINI, A. *et al.* DEVELOPMENT OF A CLASSIFIER AND A SIMULATOR TO SUPPORT THE DESIGN OF AN ANTI-DECUBITUS ACTIVE MATTRESS. **Proceedings of the Design Society**, v. 3, p. 1445–1454, jul. 2023. ISSN 2732-527X. Disponível em: ⟨https://www.cambridge.org/core/product/identifier/S2732527X23001451/type/journal_article⟩.

COLEMAN, S. *et al.* A new pressure ulcer conceptual framework. **Journal of Advanced Nursing**, v. 70, n. 10, p. 2222–2234, 2014. ISSN 1365-2648. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/jan.12405. Disponível em: ⟨https://onlinelibrary.wiley.com/doi/abs/10.1111/jan.12405⟩.

Data Science Dojo. **Entendendo o Algoritmo Random Forest: Um Guia Abrangente**. 2024. Disponível em: ⟨https://datasciencedojo.com/blog/random-forest-algorithm/⟩.

DONALDSON, L. An organisation with a memory. **Clinical Medicine**, v. 2, n. 5, p. 452–457, 2002. ISSN 1470-2118. Disponível em: ⟨https://www.sciencedirect.com/science/article/pii/S147021182403879X⟩.

DUBOIS, R. W.; BROOK, R. H. Preventable Deaths: Who, How Often, and Why? **Annals of Internal Medicine**, v. 109, n. 7, p. 582–589, out. 1988. ISSN 0003-4819. Publisher: American College of Physicians. Disponível em: ⟨https://www.acpjournals.org/doi/10.7326/0003-4819-109-7-582⟩.

DUDANI, S. A. The Distance-Weighted k-Nearest-Neighbor Rule. **IEEE Transactions on Systems, Man, and Cybernetics**, SMC-6, n. 4, p. 325–327, abr. 1976. ISSN 0018-9472. Disponível em: ⟨http://ieeexplore.ieee.org/document/5408784/⟩.

GEFEN, A. *et al.* Our contemporary understanding of the aetiology of pressure ulcers/pressure injuries. **International Wound Journal**, v. 19, n. 3, p. 692–704, ago. 2021. ISSN 1742-4801. Disponível em: ⟨https://pmc.ncbi.nlm.nih.gov/articles/PMC8874092/⟩.

GHADERI, M.; CHEHRE, H.; FATHPOUR, H. Design and Implementation of a Novel Double-Layered Wavy Mattress for the Prevention and Recovery of Pressure Ulcers: A Feasibility Study. **Journal of Biomedical Physics & Engineering**, v. 13, n. 5, p. 489–494, out. 2023. ISSN 2251-7200. Disponível em: ⟨https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10589688/⟩.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. **Deep Learning**. [S.l.]: MIT Press, 2016. ⟨http://www.deeplearningbook.org⟩.

HAESLER, E. *et al.* Protocol for the Development of the Fourth Edition of the Prevention and Treatment of Pressure Ulcers/Injuries: Clinical Practice Guideline Using GRADE Methods. **Advances in Skin & Wound Care**, v. 37, n. 3, p. 136–146, mar. 2024. ISSN 1538-8654, 1527-7941. Disponível em: ⟨https://journals.lww.com/10.1097/ASW.0000000000000079⟩.

HASTIE, T.; TIBSHIRANI, R.; FRIEDMAN, J. **The Elements of Statistical Learning**. New York, NY: Springer, 2009. (Springer Series in Statistics). ISBN 978-0-387-84857-0 978-0-387-84858-7. Disponível em: ⟨http://link.springer.com/10.1007/978-0-387-84858-7⟩.

HSU, H.-C.; LO, R.-C. **A New Mattress Development Based on Pressure Sensors for Body-contouring Uniform Support**. arXiv, 2013. ArXiv:1308.2196 [cs]. Disponível em: ⟨http://arxiv.org/abs/1308.2196⟩.

Institute of Medicine. **TO ERR IS HUMAN: BUILDING a SAFER HEALTH SYSTEM**. Washington, DC: THE NATIONAL ACADEMIES PRESS, 2000. Disponível em: ⟨https://nap.nationalacademies.org/catalog/9728/to-err-is-human-building-a-safer-health-system⟩.

JAIN, A.; DUIN, R.; MAO, J. Statistical Pattern Recognition: A Review. **IEEE Trans. Pattern Anal. Mach. Intell.**, v. 22, p. 4–37, jan. 2000.

JAVID, S. S. *et al.* Technology-Assisted Pressure Ulcer Prevention: The Automatic Mattress. **International Journal of Health Technology and Innovation**, v. 2, n. 02, p. 16–23, ago. 2023. ISSN 2583-8547. Number: 02. Disponível em: ⟨http://www.ijht.org.in/index.php/ijhti/article/view/91⟩.

KIRKLAND-WALSH, H. *et al.* Pressure Mapping Comparison of Four OR Surfaces. **AORN journal**, v. 102, p. 61.e1–9, jul. 2015.

KOSIAK, M. Etiology and pathology of ischemic ulcers. **Archives of physical medicine and rehabilitation**, v. 40 2, p. 62–9, 1959. Disponível em: ⟨https://api.semanticscholar.org/CorpusID:31917815⟩.

KOTTNER, J. *et al.* Microclimate: A critical review in the context of pressure ulcer prevention. **Clinical Biomechanics**, v. 59, p. 62–70, nov. 2018. ISSN 0268-0033, 1879-1271. Publisher: Elsevier. Disponível em: ⟨https://www.clinbiomech.com/article/S0268-0033(18)30257-2/fulltext⟩.

KRISHNAKUMAR, R. *et al.* Anti-Decubitus Air Pump for Air Mattress System. v. 3, n. 3, 2018.

KRIZHEVSKY, A.; SUTSKEVER, I.; HINTON, G. E. ImageNet Classification with Deep Convolutional Neural Networks. In: PEREIRA, F. *et al.* (Ed.). **Advances in Neural Information Processing Systems**. Curran Associates, Inc., 2012. v. 25. Disponível em: ⟨https://proceedings.neurips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf⟩.

LEAPE, L. L. *et al.* Preventing Medical Injury. **QRB - Quality Review Bulletin**, v. 19, n. 5, p. 144–149, maio 1993. ISSN 0097-5990. Disponível em: ⟨https://www.sciencedirect.com/science/article/pii/S009759901630608X⟩.

LECUN, Y.; BENGIO, Y.; HINTON, G. Deep Learning. **Nature**, v. 521, p. 436–44, maio 2015.

LEIJNEN, S.; VEEN, F. The Neural Network Zoo. **Proceedings**, v. 47, p. 9, maio 2020.

LITJENS, G. *et al.* A Survey on Deep Learning in Medical Image Analysis. **Medical Image Analysis**, v. 42, p. 60–88, dez. 2017. ISSN 13618415. ArXiv:1702.05747 [cs]. Disponível em: ⟨http://arxiv.org/abs/1702.05747⟩.

LOERAKKER, S. *et al.* The effects of deformation, ischemia, and reperfusion on the development of muscle damage during prolonged loading. **Journal of Applied Physiology**, v. 111, n. 4, p. 1168–1177, out. 2011. ISSN 8750-7587. Publisher: American Physiological Society. Disponível em: ⟨https://journals.physiology.org/doi/full/10.1152/japplphysiol.00389.2011⟩.

MILLER, G. A.; NICELY, P. E. An analysis of perceptual confusions among some english consonants. **The Journal of the Acoustical Society of America**, v. 27, n. 2, p. 338–352, mar 1955. Disponível em: ⟨https://doi.org/10.1121/1.1907526⟩.

MITCHELL, T. **Machine Learning**. McGraw-Hill Education, 1997. (McGraw-Hill international editions - computer science series). ISBN 978-0-07-042807-2. Disponível em: ⟨https://books.google.com.br/books?id=xOGAngEACAAJ⟩.

National Pressure Injury Advisory Panel; European Pressure Ulcer Advisory Panel; Pan Pacific Pressure Injury Alliance. Full Body Support Surfaces for Prevention of Pressure Injuries. In: **Prevention and Treatment of Pressure Ulcers/Injuries: Clinical Practice Guideline.** Emily Haesler, 2025. The International Guideline: Fourth Edition. Disponível em: ⟨https://internationalguideline.com/surfaces⟩.

National Pressure Injury Advisory Panel; European Pressure Ulcer Advisory Panel; Pan Pacific Pressure Injury Alliance. Pressure Ulcers/Injuries: Definition ad Etiology. In: **Prevention and Treatment of Pressure Ulcers/Injuries: Clinical Practice Guideline.** Emily Haesler, 2025. The International Guideline: Fourth Edition. Disponível em: ⟨https://static1.squarespace.com/static/6479484083027f25a6246fcb/t/67bef722cbe306395dbdc531/1740568356746/2025-Guideline-Etiology-25Feb2025.pdf⟩.

National Pressure Ulcer Advisory Panel; European Pressure Ulcer Advisory Panel; Pan Pacific Pressure Injury Alliance. **International NPUAP/EPUAP Pressure Ulcer Classification System (2009,2014)**. Emily Haesler, 2014. Disponível em: ⟨https://static1.squarespace.com/static/61e64d59c5eb655d91a71ce0/t/69525328e3177657c71fb749/1767002920595/Classification+System+Multicultural.pdf⟩.

OOMENS, C. W. J. *et al.* Pressure Induced Deep Tissue Injury Explained. **Annals of Biomedical Engineering**, v. 43, n. 2, p. 297–305, fev. 2015. ISSN 1573-9686. Disponível em: ⟨https://doi.org/10.1007/s10439-014-1202-6⟩.

Organização Mundial da Saúde. **Guia Curricular de Segurança do Paciente da Organização Mundial da Saúde: Edição Multiprofissional**. [S.l.]: Autografia, 2016. ISBN 978-85-5526-850-2.

PEDREGOSA, F. *et al.* Scikit-learn: Machine learning in python. **Journal of Machine Learning Research**, v. 12, p. 2825–2830, 2011. Disponível em: ⟨https://jmlr.org/papers/v12/pedregosa11a.html⟩.

PHUNG, V.; RHEE, E. A High-Accuracy Model Average Ensemble of Convolutional

Neural Networks for Classification of Cloud Image Patches on Small Datasets. **Applied Sciences**, v. 9, p. 4500, out. 2019.

POUYAN, M. B. *et al.* A pressure map dataset for posture and subject analytics. In: **2017 IEEE EMBS International Conference on Biomedical & Health Informatics (BHI)**. [s.n.], 2017. p. 65–68. Disponível em: ⟨https://ieeexplore.ieee.org/document/7897206⟩.

RESWICK, J. B.; ROGERS, J. E. Experience at rancho los amigos hospital with devices and techniques to prevent pressure sores. In: KENEDI, C. J.; KENEDI, R. M. (Ed.). **Bed Sore Biomechanics**. London: Palgrave, 1976, (Strathclyde Bioengineering Seminars).

Robert Kwon; Jeffrey E. Janis. **Pressure sores**. 2016. Disponível em: ⟨https://plasticsurgerykey.com/pressure-sores/⟩.

SHAH, R. **Introduction to k-Nearest Neighbors (kNN) Algorithm**. 2021. Disponível em: ⟨https://ai.plainenglish.io/introduction-to-k-nearest-neighbors-knn-algorithm-e8617a448fa8⟩.

SHEN, D.; WU, G.; SUK, H.-I. Deep Learning in Medical Image Analysis. **Annual Review of Biomedical Engineering**, v. 19, p. 221–248, jun. 2017. ISSN 1545-4274.

SIKKA, M. P.; GARG, S. Functional textiles for prevention of pressure ulcers – a review. **Research Journal of Textile and Apparel**, v. 24, n. 3, p. 185–198, jan. 2020. ISSN 1560-6074. Publisher: Emerald Publishing Limited. Disponível em: ⟨https://doi.org/10.1108/RJTA-10-2019-0047⟩.

SOKOLOVA, M.; LAPALME, G. A systematic analysis of performance measures for classification tasks. **Information Processing & Management**, v. 45, n. 4, p. 427–437, 2009. Disponível em: ⟨https://doi.org/10.1016/j.ipm.2009.03.002⟩.

SONENBLUM, S. E. *et al.* 3D anatomy and deformation of the seated buttocks. **Journal of Tissue Viability**, v. 24, n. 2, p. 51–61, maio 2015. ISSN 0965-206X. Disponível em: ⟨https://www.sciencedirect.com/science/article/pii/S0965206X15000364⟩.

SUGUNA; Thanushkodi. An Improved k-Nearest Neighbor Classification Using Genetic Algorithm. **International Journal of Computer Science Issues**, v. 7, jul. 2010.

TOPOL, E. J. High-performance medicine: the convergence of human and artificial intelligence. **Nature Medicine**, v. 25, n. 1, p. 44–56, jan. 2019. ISSN 1546-170X. Publisher: Nature Publishing Group. Disponível em: ⟨https://www.nature.com/articles/s41591-018-0300-7⟩.

YOUSEFI, R. *et al.* Bed posture classification for pressure ulcer prevention. **Conference proceedings : ... Annual International Conference of the IEEE Engineering in Medicine and Biology Society. IEEE Engineering in Medicine and Biology Society. Conference**, v. 2011, p. 7175–8, ago. 2011. Disponível em: ⟨https://doi.org/10.1109/IEMBS.2011.6091813⟩.

YOUSEFI, R. *et al.* Posture and Limb Detection for Pressure Ulcer Prevention. v. 5, p. 77, maio 2011. Disponível em: ⟨https://www.researchgate.net/publication/228834792

_Posture_and_Limb_Detection_for_Pressure_Ulcer_Prevention⟩.

YOUSEFI, R. *et al.* A smart bed platform for monitoring & Ulcer prevention. In: **2011 4th International Conference on Biomedical Engineering and Informatics (BMEI)**. [s.n.], 2011. v. 3, p. 1362–1366. ISSN: 1948-2922. Disponível em: ⟨https://ieeexplore.ieee.org/document/6098589/metrics#metrics⟩.

APÊNDICE A – IMPLEMENTAÇÃO DA GERAÇÃO DOS CONJUNTOS DE DADOS

Este apêndice apresenta os algoritmos desenvolvidos para a construção dos conjuntos de dados utilizados no treinamento e avaliação dos modelos de aprendizado de máquina empregados no trabalho. Os códigos são reorganizados e documentados para evidenciar as etapas metodológicas essenciais da implementação. São omitidas rotinas auxiliares relacionadas à depuração, visualização gráfica e verificações intermediárias, por não interferirem na reprodução da metodologia proposta.

A.1. GERAÇÃO DO CONJUNTO DE DADOS PARA REDES NEURAIS CONVOLUCIONAIS

O Código A.1 apresenta o procedimento utilizado para construir o conjunto de dados destinado ao treinamento da RNC. Os mapas de pressão são carregados, reconstruídos em matrizes bidimensionais de dimensão 64×32, os dois primeiros *frames* de cada aquisição são descartados, aplica-se a limitação dos valores máximos de pressão (`clip` em 1000) e, por fim, os dados são armazenados em um *DataFrame*.

Código A.1 – Geração do conjunto de dados para RNC.

```
import os
import numpy as np
import pandas as pd

# ================================
# Configuracao do experimento
# ================================
path = "dados␣experimentais/experiment-i/"
pacientes = range(1, 14)
posicoes = range(1, 18)
df_cnn = pd.DataFrame(columns=["Frame", "Posicao", "Paciente"])

# ================================
# Construcao do conjunto de dados
# ================================
for paciente in pacientes:
for posicao in posicoes:
dados_brutos = pd.read_csv(
f"{path}S{paciente}/{posicao}.txt",
header=None,
sep="\t"
)

# Remove a coluna vazia introduzida durante a aquisicao.
dados_limpos = dados_brutos.iloc[:, :2048]

# Reconstroi cada registro como um mapa bidimensional de pressao.
```

```
frames = dados_limpos.to_numpy().reshape((-1, 64, 32))
for indice, frame in enumerate(frames):

# Descarta os dois primeiros frames.
if indice < 2:
continue

# Limita os valores maximos de pressao.
frame = np.clip(frame, None, 1000)

registro = pd.DataFrame({
        "Frame": [frame],
        "Posicao": [posicao],
        "Paciente": [paciente]
})

df_cnn = pd.concat([df_cnn, registro], ignore_index=True)

# ================================
# Salvamento do conjunto de dados
# ================================
os.makedirs("DataFrames/Exp_1/", exist_ok=True)
df_cnn.to_hdf(
"DataFrames/Exp_1/DataFrame_CNN_clipp.h5",
key="df",
mode="w",
index=False
)
```

A.2. GERAÇÃO DO CONJUNTO DE DADOS PARA *K-NEAREST NEIGHBORS* E FLORESTA ALEATÓRIA

O Código A.2 apresenta o algoritmo utilizado para construir o conjunto de dados empregado pelos classificadores *K-Nearest Neighbors* e Floresta Aleatória. Diferentemente da Rede Neural Convolucional, que utiliza diretamente os mapas bidimensionais de pressão como entrada, esses modelos recebem um vetor de características extraído de cada *frame* adquirido. Para cada mapa de pressão são calculados descritores estatísticos e geométricos, incluindo pressão média, pressão máxima, coordenadas da pressão máxima, centro de pressão, área ocupada e razão de aspecto da região ativa. Após a extração dessas características, os resultados são organizados em um *DataFrame* e armazenados para utilização nas etapas de treinamento e validação dos classificadores.

Código A.2 – Geração do conjunto de dados para KNN e Floresta Aleatória.

```
import os
import numpy as np
import pandas as pd

# ================================
```

```
# Funcoes para extracao das caracteristicas
# ================================
def calcular_cp(matriz):
# Calcula o centro de pressao utilizando a media ponderada das
    intensidades do mapa de pressao.
matriz = np.asarray(matriz)
altura, largura = matriz.shape
y, x = np.indices((altura, largura))
total = np.sum(matriz)

if total == 0:
return np.nan, np.nan

cp_x = np.sum(x * matriz) / total
cp_y = np.sum(y * matriz) / total
return cp_x, cp_y


def coordenada_max(matriz):
# Retorna as coordenadas do sensor correspondente ao maior valor de
    pressao.
return tuple(np.unravel_index(np.argmax(matriz), matriz.shape))


def calcular_area_ocupada(matriz, limiar=50):
# Calcula a porcentagem da area ocupada considerando apenas sensores com
     pressao superior ao limiar.
sensores_ativos = np.sum(matriz > limiar)
return (sensores_ativos / matriz.size) * 100


def calcular_aspecto_area_ativa(matriz, limiar=50):
# Determina as dimensoes da regiao ativa e calcula sua razao de aspecto.
matriz_binaria = matriz > limiar

if not np.any(matriz_binaria):
return 0, 0, np.nan

linhas_ativas = np.any(matriz_binaria, axis=1)
colunas_ativas = np.any(matriz_binaria, axis=0)
y_min, y_max = np.where(linhas_ativas)[0][[0, -1]]
x_min, x_max = np.where(colunas_ativas)[0][[0, -1]]
largura = x_max - x_min + 1
altura = y_max - y_min + 1
aspecto = largura / altura
return largura, altura, aspecto


# ================================
# Configuracao do experimento
# ================================
path = "dados␣experimentais/experiment-i/"
pacientes = range(1, 14)
posicoes = range(1, 18)

df_features = pd.DataFrame(columns=[
"Pressao␣Media",
```

```
"Pressao␣Maxima",
"PMax_x",
"PMax_y",
"CP_x",
"CP_y",
"Area␣Ocupada␣(%)",
"Largura_ar",
"Altura_ar",
"Aspect␣Ratio",
"Posicao",
"Paciente"
])

# ================================
# Construcao do conjunto de dados
# ================================
for paciente in pacientes:
for posicao in posicoes:
dados_brutos = pd.read_csv(
f"{path}S{paciente}/{posicao}.txt",
header=None,
sep="\t"
)

# Remove a coluna vazia introduzida durante a aquisicao.
dados_limpos = dados_brutos.iloc[:, :2048]

# Reconstroi cada registro como um mapa bidimensional de pressao (64 x
    32 sensores).
frames = dados_limpos.to_numpy().reshape((-1, 64, 32))
for indice, frame in enumerate(frames):
# Descarta os dois primeiros frames de cada aquisicao.
if indice < 2:
continue

# Limita os valores maximos de pressao.
frame = np.clip(frame, None, 1000)

# ================================
# Extracao das caracteristicas
# ================================
pressao_media = np.mean(frame)
pressao_maxima = np.max(frame)
cp_x, cp_y = calcular_cp(frame)
pmax_y, pmax_x = coordenada_max(frame)
area_ocupada = calcular_area_ocupada(frame)
largura_area, altura_area, aspecto = \
calcular_aspecto_area_ativa(frame)

# ================================
# Armazena as caracteristicas extraidas
# ================================
registro = pd.DataFrame({
        "Pressao␣Media": [pressao_media],
        "Pressao␣Maxima": [pressao_maxima],
        "PMax_x": [pmax_x],
        "PMax_y": [pmax_y],
```

```
            "CP_x": [cp_x],
            "CP_y": [cp_y],
            "Area␣Ocupada␣(%)": [area_ocupada],
            "Largura_ar": [largura_area],
            "Altura_ar": [altura_area],
            "Aspect␣Ratio": [aspecto],
            "Posicao": [posicao],
            "Paciente": [paciente]

    })

    df_features = pd.concat(
    [df_features, registro],
    ignore_index=True
    )

    # ================================
    # Salvamento do conjunto de dados
    # ================================
    os.makedirs("DataFrames/Exp_1/", exist_ok=True)
    df_features.to_hdf(
    "DataFrames/Exp_1/DataFrame_FE.h5",
    key="df",
    mode="w",
    index=False
    )
```

Os algoritmos apresentados nesta seção correspondem às implementações efetivamente utilizadas para a geração do conjunto de dados empregado pelos classificadores *K-Nearest Neighbors* e Floresta Aleatória. São omitidos apenas procedimentos auxiliares relacionados à inspeção visual dos dados, mensagens de depuração e verificações intermediárias, preservando integralmente as etapas necessárias para reproduzir a metodologia proposta.

APÊNDICE B – IMPLEMENTAÇÃO DO CLASSIFICADOR FLORESTA ALEATÓRIA

Este apêndice apresenta a implementação final utilizada no treinamento e na avaliação do classificador Floresta Aleatória. Durante o desenvolvimento do trabalho são avaliadas versões simplificadas do algoritmo, utilizando diferentes estratégias de particionamento dos dados. Entretanto, por representar a versão final da implementação empregada na obtenção dos resultados apresentados nesta dissertação, opta-se por documentar apenas a implementação baseada na estratégia *Leave-One-Subject-Out*, implementada por meio do *GroupKFold*, da qual as demais implementações constituem casos particulares.

Código B.1 – Treinamento e avaliação do classificador Floresta Aleatória utilizando a estratégia *Leave-One-Subject-Out* implementada por meio de *GroupKFold*.

```
import numpy as np
import pandas as pd

from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import GroupKFold
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.metrics import (
accuracy_score,
precision_score,
recall_score,
f1_score
)

# ================================
# Configuracao do experimento
# ================================
path_dataset = "DataFrames/Exp_1/DataFrame_FE.h5"
RANDOM_STATE = 42
RF_PARAMS = {
        "n_estimators": 100,
        "random_state": RANDOM_STATE,
        "max_depth": None,
        "max_features": "sqrt",
        "min_samples_split": 2,
        "min_samples_leaf": 1,
        "bootstrap": True,
        "n_jobs": -1
}

# ================================
# Carregamento do conjunto de dados
# ================================
df_rf = pd.read_hdf(path_dataset, key="df")
# Remove registros invalidos eventualmente presentes no conjunto de
    dados.
linhas_invalidas = np.where(df_rf == "largura")[0]
df_rf = (
df_rf
```

```
.drop(index=linhas_invalidas)
.reset_index(drop=True)
)

# ================================
# Preparacao dos atributos e rotulos
# ================================
feature_columns = [
"Pressao␣Media",
"Pressao␣Maxima",
"PMax_x",
"PMax_y",
"CP_x",
"CP_y",
"Area␣Ocupada␣(%)",
"Largura_ar",
"Altura_ar",
"Aspect␣Ratio"

]

X = df_rf[feature_columns].values
y = df_rf["Posicao"]
grupos = df_rf["Paciente"]

# ================================
# Codificacao dos rotulos
# ================================
encoder = LabelEncoder()
y_codificado = encoder.fit_transform(y)

# ================================
# Padronizacao dos atributos
# ================================
# Embora o Random Forest nao requeira normalizacao, os atributos foram
    padronizados para manter a consistencia metodologica entre os
    classificadores avaliados no trabalho.
scaler = StandardScaler()
X_padronizado = scaler.fit_transform(X)

# ================================
# Configuracao da estrategia de validacao
# ================================
# A estrategia LOSO e implementada utilizando o identificador de cada
    participante como grupo, garantindo que todas as amostras de um
    mesmo individuo sejam destinadas exclusivamente ao conjunto de
    treinamento ou ao conjunto de teste em cada iteracao.
validacao_loso = GroupKFold(
n_splits=len(np.unique(grupos))
)

# ================================
# Configuracao do classificador Floresta Aleatoria
# ================================
modelo_rf = RandomForestClassifier(**RF_PARAMS)

# ================================
```

```
# Inicializacao das estruturas para armazenamento dos resultados
# ================================
accuracy_por_fold = []
precision_por_fold = []
recall_por_fold = []
f1_por_fold = []
rotulos_reais = []
rotulos_preditos = []
pacientes_avaliados = []

# ================================
# Processo de treinamento e validacao
# ================================
# Cada iteracao considera um participante distinto como conjunto de
    teste (LOSO).
for indice_fold, (train_index, test_index) in enumerate(
validacao_loso.split(X_padronizado, y_codificado, grupos),
start=1
):

# ================================
# Separacao dos conjuntos de treinamento e teste
# ================================
X_train = X_padronizado[train_index]
X_test  = X_padronizado[test_index]
y_train = y_codificado[train_index]
y_test  = y_codificado[test_index]
grupo_teste = grupos.iloc[test_index].unique()[0]

# ================================
# Treinamento do classificador
# ================================
modelo_rf.fit(X_train, y_train)

# ================================
# Predicao das amostras de teste
# ================================
y_pred = modelo_rf.predict(X_test)

# ================================
# Avaliacao do desempenho
# ================================
accuracy_fold = accuracy_score(
y_test,
y_pred
)

precision_fold = precision_score(
y_test,
y_pred,
average="weighted",
zero_division=0
)

recall_fold = recall_score(
y_test,
y_pred,
```

```
average="weighted",
zero_division=0
)

f1_fold = f1_score(
y_test,
y_pred,
average="weighted",
zero_division=0
)

# ================================
# Armazenamento das metricas obtidas
# ================================
accuracy_por_fold.append(accuracy_fold)
precision_por_fold.append(precision_fold)
recall_por_fold.append(recall_fold)
f1_por_fold.append(f1_fold)
rotulos_reais.extend(y_test)
rotulos_preditos.extend(y_pred)
pacientes_avaliados.append(grupo_teste)

# ================================
# Consolidacao das metricas
# ================================
accuracy_media = np.mean(accuracy_por_fold)
precision_media = np.mean(precision_por_fold)
recall_media = np.mean(recall_por_fold)
f1_media = np.mean(f1_por_fold)

# As metricas consolidadas representam o desempenho medio do
    classificador considerando todas as iteracoes da estrategia LOSO.
resultados = {

        "Accuracy": accuracy_media,
        "Precision": precision_media,
        "Recall": recall_media,
        "F1-score": f1_media

}
```

O algoritmo apresentado neste apêndice corresponde à implementação final utilizada no treinamento e na avaliação do classificador Floresta Aleatória. São omitidas apenas rotinas auxiliares relacionadas à geração de gráficos, armazenamento de resultados intermediários, medição de tempo de execução, mensagens de depuração e demais procedimentos que não interferem na reprodução da metodologia empregada.

APÊNDICE C – IMPLEMENTAÇÃO DO CLASSIFICADOR *K-NEAREST NEIGHBORS*

Este apêndice apresenta a implementação final utilizada no treinamento e na avaliação do classificador *K-Nearest Neighbors*. Durante o desenvolvimento do trabalho são avaliadas versões simplificadas do algoritmo, utilizando diferentes estratégias de particionamento dos dados. Entretanto, por representar a versão final da implementação empregada na obtenção dos resultados apresentados nesta dissertação, opta-se por documentar apenas a implementação baseada na estratégia *Leave-One-Subject-Out*, implementada por meio do *GroupKFold*, da qual as demais implementações constituem casos particulares.

Código C.1 – Implementação do treinamento e avaliação do classificador K-Nearest Neighbors.

```
import numpy as np
import pandas as pd

from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import GroupKFold
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.metrics import (
accuracy_score,
precision_score,
recall_score,
f1_score
)

# ================================
# Configuracao do experimento
# ================================
path_dataset = "DataFrames/Exp_1/DataFrame_FE.h5"
KNN_PARAMS = {
        "n_neighbors": 11
}

# ================================
# Carregamento do conjunto de dados
# ================================
df_knn = pd.read_hdf(path_dataset, key="df")
# Remove registros invalidos eventualmente presentes no conjunto de
    dados.
linhas_invalidas = np.where(df_knn == "largura")[0]
df_knn = (
df_knn
.drop(index=linhas_invalidas)
.reset_index(drop=True)
)

# ================================
# Preparacao dos atributos e rotulos
# ================================
feature_columns = [
```

```
"Pressao␣Media",
"Pressao␣Maxima",
"PMax_x",
"PMax_y",
"CP_x",
"CP_y",
"Area␣Ocupada␣(%)",
"Largura_ar",
"Altura_ar",
"Aspect␣Ratio"

]

X = df_knn[feature_columns].values
y = df_knn["Posicao"]
grupos = df_knn["Paciente"]

# ================================
# Codificacao dos rotulos
# ================================
encoder = LabelEncoder()
y_codificado = encoder.fit_transform(y)

# ================================
# Padronizacao dos atributos
# ================================
# A normalizacao dos atributos e necessaria para o classificador KNN,
    uma vez que o algoritmo utiliza medidas de distancia entre as
    amostras para realizar a classificacao.
scaler = StandardScaler()
X_padronizado = scaler.fit_transform(X)

# ================================
# Configuracao da estrategia de validacao
# ================================
# A estrategia LOSO e implementada utilizando o identificador de cada
    participante como grupo, garantindo que todas as amostras de um
    mesmo individuo sejam destinadas exclusivamente ao conjunto de
    treinamento ou ao conjunto de teste em cada iteracao.
validacao_loso = GroupKFold(
n_splits=len(np.unique(grupos))
)

# ================================
# Inicializacao das estruturas para armazenamento dos resultados
# ================================
accuracy_por_fold = []
precision_por_fold = []
recall_por_fold = []
f1_por_fold = []
rotulos_reais = []
rotulos_preditos = []
pacientes_avaliados = []

# ================================
# Processo de treinamento e validacao
# ================================
```

```
# Cada iteracao considera um participante distinto como conjunto de
    teste (LOSO).
for indice_fold, (train_index, test_index) in enumerate(
validacao_loso.split(X_padronizado, y_codificado, grupos),
start=1
):

# ================================
# Separacao dos conjuntos de treinamento e teste
# ================================
X_train = X_padronizado[train_index]
X_test = X_padronizado[test_index]
y_train = y_codificado[train_index]
y_test = y_codificado[test_index]
grupo_teste = grupos.iloc[test_index].unique()[0]

# ================================
# Inicializacao e treinamento do classificador
# ================================
modelo_knn = KNeighborsClassifier(**KNN_PARAMS)
modelo_knn.fit(X_train, y_train)

# ================================
# Predicao das amostras de teste
# ================================
y_pred = modelo_knn.predict(X_test)

# ================================
# Avaliacao do desempenho
# ================================
accuracy_fold = accuracy_score(
y_test,
y_pred
)

precision_fold = precision_score(
y_test,
y_pred,
average="weighted",
zero_division=0
)

recall_fold = recall_score(
y_test,
y_pred,
average="weighted",
zero_division=0
)

f1_fold = f1_score(
y_test,
y_pred,
average="weighted",
zero_division=0
)

# ================================
```

```
# Armazenamento das metricas obtidas
# ================================
accuracy_por_fold.append(accuracy_fold)
precision_por_fold.append(precision_fold)
recall_por_fold.append(recall_fold)
f1_por_fold.append(f1_fold)
rotulos_reais.extend(y_test)
rotulos_preditos.extend(y_pred)
pacientes_avaliados.append(grupo_teste)

# ================================
# Consolidacao das metricas
# ================================
accuracy_media = np.mean(accuracy_por_fold)
precision_media = np.mean(precision_por_fold)
recall_media = np.mean(recall_por_fold)
f1_media = np.mean(f1_por_fold)

# As metricas consolidadas representam o desempenho medio do
    classificador considerando todas as iteracoes da estrategia LOSO.
metricas_finais = {

        "Accuracy": accuracy_media,
        "Precision": precision_media,
        "Recall": recall_media,
        "F1-score": f1_media

}
```

O algoritmo apresentado neste apêndice corresponde à implementação final utilizada no treinamento e na avaliação do classificador *K-Nearest Neighbors*. São omitidas apenas rotinas auxiliares relacionadas à geração de gráficos, armazenamento de resultados intermediários, medição de tempo de execução, mensagens de depuração e demais procedimentos que não interferem na reprodução da metodologia empregada.

APÊNDICE D – IMPLEMENTAÇÃO DO CLASSIFICADOR REDE NEURAL CONVOLUCIONAL

Este apêndice apresenta a implementação final utilizada no treinamento e na avaliação da Rede Neural Convolucional. Durante o desenvolvimento do trabalho são avaliadas diferentes arquiteturas da rede, bem como versões simplificadas da implementação utilizando estratégias distintas de particionamento dos dados. Entretanto, por representar a versão final empregada na obtenção dos resultados apresentados nesta dissertação, optase por documentar apenas a implementação baseada na estratégia *Leave-One-Subject-Out*, implementada por meio do *GroupKFold*, contemplando também a definição da arquitetura da rede, o processo de treinamento, a utilização do mecanismo de *Early Stopping* e a comparação entre diferentes configurações de hiperparâmetros, da qual as demais implementações constituem casos particulares.

Código D.1 – Implementação do treinamento e avaliação da Rede Neural Convolucional.

```
import random
import numpy as np
import pandas as pd
import tensorflow as tf

from sklearn.model_selection import GroupKFold, train_test_split
from sklearn.preprocessing import LabelEncoder

from tensorflow.keras.utils import to_categorical
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import (
Conv2D,
MaxPooling2D,
Flatten,
Dense
)
from tensorflow.keras.callbacks import EarlyStopping

# ================================
# Configuracao do experimento
# ================================
SEED = 42
random.seed(SEED)
np.random.seed(SEED)
tf.random.set_seed(SEED)
tf.keras.utils.set_random_seed(SEED)
tf.config.experimental.enable_op_determinism()

# ================================
# Carregamento do conjunto de dados
# ================================
path_dataset = "DataFrames/Exp_1/"
df_cnn = pd.read_hdf(
path_dataset + "DataFrame_CNN_clipp.h5",
key="df"
```

```
)

# ================================
# Preparacao dos atributos e rotulos
# ================================
X = np.stack(df_cnn["Frame"].to_numpy())
X = np.expand_dims(
X,
axis=-1
)

X = X / 1000.0
y = df_cnn["Posicao"]
grupos = df_cnn["Paciente"]

# ================================
# Codificacao dos rotulos
# ================================
encoder = LabelEncoder()
y_codificado = encoder.fit_transform(y)
y_categorico = to_categorical(
y_codificado
)

# ================================
# Configuracao das arquiteturas avaliadas
# ================================
# Cada configuracao representa uma arquitetura candidata, variando a
    quantidade de filtros das camadas convolucionais e o numero de
    neuronios da camada totalmente conectada.
configuracoes_modelos = {

        "M001": {
                "filtros": [2, 2, 4],
                "dense": 8
        },
        "M002": {
                "filtros": [4, 8, 16],
                "dense": 16
        },
        "M003": {
                "filtros": [8, 16, 32],
                "dense": 32
        },
        "M004": {
                "filtros": [16, 32, 64],
                "dense": 64
        },
        "M005": {
                "filtros": [32, 64, 128],
                "dense": 64
        },
        "M006": {
                "filtros": [32, 64, 128],
                "dense": 128
        },
        "M007": {
```

```
                "filtros": [64, 128, 256],
                "dense": 128
        },
        "M008": {
                "filtros": [64, 128, 256],
                "dense": 256
        },
        "M009": {
                "filtros": [128, 256, 512],
                "dense": 256
        },
        "M010": {
                "filtros": [128, 256, 512],
                "dense": 512
        }

}

# As configuracoes apresentadas sao utilizadas na construcao automatica
    das arquiteturas avaliadas durante os experimentos.

# ================================
# Definicao da arquitetura da Rede Neural Convolucional
# ================================
def criar_modelo(filtros, dense_neurons):
modelo = Sequential()

# ================================
# Primeira camada convolucional
# ================================
modelo.add(
Conv2D(
filtros[0],
kernel_size=(3, 3),
activation="relu",
padding="same",
input_shape=X.shape[1:]
)
)
modelo.add(
MaxPooling2D(pool_size=(2, 2))
)

# ================================
# Segunda camada convolucional
# ================================
modelo.add(
Conv2D(
filtros[1],
kernel_size=(3, 3),
activation="relu",
padding="same"
)
)
modelo.add(
MaxPooling2D(pool_size=(2, 2))
)
```

```
# ================================
# Terceira camada convolucional
# ================================
modelo.add(
Conv2D(
filtros[2],
kernel_size=(3, 3),
activation="relu",
padding="same"
)
)
modelo.add(
MaxPooling2D(pool_size=(2, 2))
)
modelo.add(
Flatten()
)

# ================================
# Camada totalmente conectada
# ================================
modelo.add(
Dense(
dense_neurons,
activation="relu"
)
)

# ================================
# Camada de saida
# ================================
modelo.add(
Dense(
len(encoder.classes_),
activation="softmax"
)
)

# ================================
# Compilacao do modelo
# ================================
modelo.compile(
optimizer="adam",
loss="categorical_crossentropy",
metrics=["accuracy"]
)
return modelo

# ================================
# Configuracao da estrategia de validacao
# ================================
validacao_loso = GroupKFold(
n_splits=len(np.unique(grupos))
)

# ================================
```

```
# Inicializacao das estruturas para armazenamento dos resultados
# ================================
metricas_modelos = []

# ================================
# Avaliacao das arquiteturas candidatas
# ================================
for nome_modelo, parametros in configuracoes_modelos.items():
accuracy_por_fold = []
precision_por_fold = []
recall_por_fold = []
f1_por_fold = []

# ================================
# Estrategia Leave-One-Subject-Out (LOSO)
# ================================
# Cada iteracao considera um participante distinto como conjunto de
    teste.
for indice_fold, (train_index, test_index) in enumerate(
validacao_loso.split(
X,
y_codificado,
grupos
),
start=1
):

# ================================
# Separacao dos conjuntos de treinamento e teste
# ================================
X_train = X[train_index]
X_test = X[test_index]
y_train = y_categorico[train_index]
y_test = y_categorico[test_index]
grupo_teste = grupos.iloc[test_index].unique()[0]

# ================================
# Separacao do conjunto de validacao
# ================================
X_train,
X_val,
y_train,
y_val = train_test_split(
X_train,
y_train,
test_size=0.20,
random_state=SEED,
stratify=np.argmax(y_train, axis=1)
)

# ================================
# Construcao da arquitetura corrente
# ================================
modelo = criar_modelo(
parametros["filtros"],
parametros["dense"]
)
```

```
# ================================
# Configuracao do Early Stopping
# ================================
early_stopping = EarlyStopping(
monitor="val_loss",
patience=10,
restore_best_weights=True
)

# ================================
# Treinamento da Rede Neural Convolucional
# ================================
modelo.fit(
X_train,
y_train,
validation_data=(X_val, y_val),
epochs=60,
batch_size=32,
callbacks=[early_stopping],
verbose=0
)

# ================================
# Avaliacao do modelo
# ================================
y_pred_prob = modelo.predict(
X_test,
verbose=0
)

y_pred = np.argmax(
y_pred_prob,
axis=1
)

y_true = np.argmax(
y_test,
axis=1
)

# ================================
# Calculo das metricas de desempenho
# ================================
accuracy_fold = accuracy_score(
y_true,
y_pred
)

precision_fold = precision_score(
y_true,
y_pred,
average="weighted",
zero_division=0
)

recall_fold = recall_score(
```

```
y_true,
y_pred,
average="weighted",
zero_division=0
)

f1_fold = f1_score(
y_true,
y_pred,
average="weighted",
zero_division=0
)

# ================================
# Armazenamento das metricas do fold
# ================================
accuracy_por_fold.append(accuracy_fold)
precision_por_fold.append(precision_fold)
recall_por_fold.append(recall_fold)
f1_por_fold.append(f1_fold)

# ================================
# Consolidacao das metricas da arquitetura avaliada
# ================================
metricas_modelos.append({

        "Modelo": nome_modelo,
        "Accuracy": np.mean(accuracy_por_fold),
        "Precision": np.mean(precision_por_fold),
        "Recall": np.mean(recall_por_fold),
        "F1-score": np.mean(f1_por_fold)

})

# ================================
# Selecao da melhor arquitetura
# ================================
resultados = pd.DataFrame(metricas_modelos)
```

O algoritmo apresentado neste apêndice corresponde à implementação final utilizada no treinamento e na avaliação da Rede Neural Convolucional. São omitidas apenas rotinas auxiliares relacionadas ao armazenamento de modelos, geração de gráficos, exportação de arquivos, registros de execução e demais procedimentos que não interferem na reprodução da metodologia.