UNIVERSIDADE FEDERAL DO CEARÁ
CENTRO DE SOBRAL
DEPARTAMENTO DE ENGENHARIA
CURSO DE GRADUAÇÃO EM ENGENHARIA DA COMPUTAÇÃO

GABRIEL ARAÚJO TEIXEIRA

DIAGNÓSTICO AUTOMATIZADO DE FIBRILAÇÃO ATRIAL EM ECG DE DISPOSITIVOS VESTÍVEIS COM DEEP LEARNING

SOBRAL
2026

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Sistema de Bibliotecas
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

T266d Teixeira, Gabriel.
Diagnóstico Automatizado de Fibrilação Atrial em ECG de Dispositivos Vestíveis com Deep Learning / Gabriel Teixeira. – 2026.
49 f. : il. color.

Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Campus de Sobral, Curso de Engenharia da Computação, Sobral, 2026.
Orientação: Profa. Dra. Jermana Lopes de Moraes.

1. Fibrilação Atrial. 2. Deep Learning. 3. ECG. 4. Smartwatch . I. Título.

CDD 621.39

GABRIEL ARAÚJO TEIXEIRA

DIAGNÓSTICO AUTOMATIZADO DE FIBRILAÇÃO ATRIAL EM ECG DE DISPOSITIVOS VESTÍVEIS COM DEEP LEARNING

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Engenharia da Computação do Centro de Sobral da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Engenharia da Computação.

Aprovada em: 29/01/2026

BANCA EXAMINADORA

Prof. Dr. Jermana Lopes de Moraes (Orientadora)
Universidade Federal do Ceará (UFC)

Prof. Dr. Thiago Iachiley Araújo de Souza
Universidade Federal do Ceará (UFC)

Prof. Mr. Danillo Fernandes do Nascimento
Universidade Federal do Ceará (UFC)

À Deus, à minha família, pelo amor, apoio e incentivo incondicional ao longo de toda a minha trajetória.

AGRADECIMENTOS

A Deus, agradeço primeiramente, por me conceder força, sabedoria e perseverança ao longo de toda a minha jornada acadêmica, permitindo superar os desafios enfrentados durante o desenvolvimento deste trabalho.

À minha família, em especial ao meu pai, Francisco Teixeira Sobrinho, e à minha mãe, Marineide Araújo de Matos, expresso minha profunda gratidão pelo apoio incondicional, incentivo constante e compreensão em todos os momentos da minha trajetória, sendo fundamentais para a realização deste objetivo.

À minha orientadora, Jermana Lopes de Moraes, agradeço sinceramente pela paciência, dedicação e valiosas orientações ao longo de todas as etapas da construção deste trabalho, contribuindo de forma decisiva para o meu amadurecimento acadêmico e científico.

Ao professor Carlos Elmano de Alencar, registro meus agradecimentos pelos ensinamentos, conselhos e contribuições ao longo da trajetória acadêmica, que foram essenciais para minha formação profissional e pessoal.

À Azael Frota Viana Gomes, agradeço pela ajuda e apoio no desenvolvimento deste trabalho.

À Samsung e ao Insight Lab, agradeço pelo suporte técnico, disponibilização de materiais e compartilhamento de conhecimentos fundamentais para o desenvolvimento deste trabalho, possibilitando a aplicação prática dos conceitos estudados e o avanço da pesquisa realizada.

Por fim, agradeço a todos que, direta ou indiretamente, contribuíram para a concretização deste trabalho.

"O sonho é que leva a gente para frente. Se a gente for seguir a razão, fica aquietado, acomodado."

(Ariano Suassuna)

RESUMO

A fibrilação atrial (FA) é a arritmia cardíaca sustentada mais comum na população adulta, associada a risco elevado de AVC, insuficiência cardíaca e mortalidade. Com a difusão de dispositivos vestíveis surge a oportunidade de monitorar continuamente o eletrocardiograma (ECG), possibilitando detecção precoce de arritmias. Neste trabalho foi desenvolvido um sistema baseado em *smartwatch* para, a partir de janelas de ECG de 30 segundos, realizar a classificação automática de ritmos cardíacos por meio de uma rede neural convolucional otimizada (DMKS). A metodologia envolveu o uso da base PhysioNet/CinC Challenge 2017 para treinamento, complementada por outros três repositórios públicos (MIT-BIH Noise Stress Test Database, Motion Artifact Contaminated ECG Database e PhysioNet/CinC 2011) para enriquecimento de ruídos e artefatos. Aplicou-se um pipeline de pré-processamento (padronização de comprimento, filtragem Butterworth com `filtfilt` e normalização Min–Max), geração controlada de exemplos ruidosos e data augmentation para ampliar a diversidade sem alterar a representação de desbalanceamento entre classes. A arquitetura selecionada foi convertida para TensorFlow Lite para implantação on-device. Os resultados demonstram desempenho robusto do classificador, com F1-score global de aproximadamente 0.93 e alta sensibilidade na detecção de episódios de fibrilação atrial (recall ≈ 0.89; F1 ≈ 0.89). A conversão para TensorFlow Lite com quantização pós-treinamento permitiu inferência eficiente em dispositivo móvel, mantendo desempenho comparável à versão não quantizada. Conclui-se que a solução proposta é promissora para triagem automática de FA a partir de sinais obtidos por *smartwatches*, apresentando alta sensibilidade para FA e alto F1-score global.

Palavras-chave: Fibrilação atrial; *Smartwatch*; Eletrocardiograma; Deep learning; TensorFlow Lite.

ABSTRACT

Atrial fibrillation (AF) is the most common sustained cardiac arrhythmia in the adult population and is associated with an increased risk of stroke, heart failure, and mortality. With the widespread adoption of wearable devices, continuous ambulatory monitoring of the electrocardiogram (ECG) has become feasible, enabling early detection of arrhythmias. In this work, a system based on a *smartwatch* was developed to perform automatic classification of cardiac rhythms from 30-second ECG windows using an optimized convolutional neural network (DMKS). The methodology employed the PhysioNet/Computing in Cardiology Challenge 2017 dataset for training, together with a signal processing pipeline including length standardization, Butterworth filtering using `filtfilt`, and Min–Max normalization. Controlled generation of noisy examples and additional data augmentation techniques were applied to increase data diversity while preserving the natural class imbalance observed in real-world scenarios. The selected architecture was converted to TensorFlow Lite for on-device deployment. The results demonstrate robust classifier performance, with an overall F1-score of approximately 0.93 and high sensitivity in the detection of atrial fibrillation episodes (recall ≈ 0.89; F1-score ≈ 0.89). Post-training quantization and conversion to TensorFlow Lite enabled efficient on-device inference while maintaining performance comparable to the non-quantized model. It is concluded that the proposed solution is promising for automatic AF screening from *smartwatch*-acquired signals, achieving high sensitivity for AF and a high overall F1-score.

Keywords: Atrial fibrillation; *Smartwatch*; Electrocardiogram; Deep learning; TensorFlow Lite.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

AVC	Acidente Vascular Cerebral
CNN	Redes Neurais Convolucionais (*Convolutional Neural Networks*)
DMKS	Deep Multi-Kernel Signal
ECG	Eletrocardiograma
EMG	Eletromiografia
F1	*F1-score*
FA	Fibrilação Atrial
FDA	*Food and Drug Administration*
GPU	Unidade de Processamento Gráfico (*Graphics Processing Unit*)
MIT-BIH	Massachusetts Institute of Technology - Beth Israel Hospital
TFLite	TensorFlow Lite

SUMÁRIO

1 INTRODUÇÃO

A fibrilação atrial (FA) é uma das arritmias cardíacas mais comuns, afetando milhões de pessoas em todo o mundo. Caracteriza-se por contrações rápidas e desorganizadas das câmaras atriais do coração, resultando em um ritmo cardíaco irregular e, muitas vezes, acelerado. Essa condição pode levar a complicações graves à saúde, incluindo acidente vascular cerebral (AVC), insuficiência cardíaca e até morte súbita, especialmente quando não diagnosticada e tratada precocemente (ANDERSEN, 2019; ALDUGHAYFIQ *et al.*, 2023).

O eletrocardiograma (ECG) é amplamente reconhecido como o método padrão-ouro para a detecção da fibrilação atrial. Por meio da análise dos sinais elétricos do coração, o ECG permite a identificação de padrões irregulares no ritmo cardíaco, como a ausência de ondas P regulares e intervalos RR variáveis, características marcantes da FA (XIONG *et al.*, 2022). No entanto, a realização do exame tradicional em ambientes clínicos pode limitar o monitoramento contínuo e precoce, especialmente em casos de FA paroxística, em que os episódios são intermitentes e assintomáticos.

Nesse contexto, os dispositivos vestíveis, como *smartwatches* equipados com sensores de ECG, emergem como uma alternativa prática e acessível para o registro contínuo de sinais cardíacos. Esses dispositivos permitem a aquisição de sinais ECG de uma ou mais derivações por meio de sensores integrados, possibilitando o monitoramento remoto e em tempo real do usuário (ZANDE *et al.*, 2023). Grandes empresas como Samsung e Apple já disponibilizam recursos aprovados por agências reguladoras como a Food and Drug Administration (FDA) para monitoramento cardíaco por meio de seus relógios inteligentes (Samsung, 2024; FDA, 2020).

Apesar dos avanços recentes em aprendizado profundo aplicados a sinais cardíacos, existem lacunas claras na literatura: muitos trabalhos demonstram arquiteturas eficazes (por exemplo, Redes Neurais Convolucionais - CNNs e híbridos) em bases públicas bem anotadas, mas poucos avaliam de forma sistemática como escolhas de pré-processamento, filtros digitais e estratégias de *data augmentation* afetam a robustez em sinais originados por *smartwatches*, nem quantificam o custo real de executar esses modelos em dispositivos móveis. O presente trabalho ocupa exatamente esse espaço: são investigadas combinações de técnicas de pré-processamento (padronização do comprimento, filtros *Butterworth* com aplicação *zero-phase*, normalização, *masking*), protocolos de geração de ruído realista e *augmentation* (*stitching, baseline wander,* Eletromiografia — EMG, 60 Hz), e são comparadas arquiteturas CNN multi-escala (inspiradas em K-B2S+/DMKS) para avaliar sua sensibilidade e generalização em ECGs

de origem ambulatorial. Além disso, é demonstrado o caminho completo até a implantação *on-device*, convertendo e otimizando o modelo com TensorFlow Lite e utilizando-o em um aplicativo Android nativo. Com isso, a contribuição deste trabalho combina (i) uma análise experimental cuidadosa das etapas de sinal que antecedem o modelo, (ii) geração e uso de ruídos realistas para aumentar a robustez, (iii) validação prática em *smartwatch* e implantação móvel — aspectos pouco integrados nos estudos anteriores. (FANG *et al.*, 2025; XIONG *et al.*, 2022; ZANDE *et al.*, 2023; ALDUGHAYFIQ *et al.*, 2023; TensorFlow Documentation, 2023; TensorFlow, 2024b)

Para que esses modelos sejam efetivamente utilizados em cenários *real-time* e embarcados, é necessária sua conversão e otimização para execução em ambiente nativo móvel. Nesse sentido, a conversão para TensorFlow Lite possibilita a aplicação de otimizações de quantização pós-treinamento (por exemplo, float16 ou int8), poda e outras transformações que reduzem o tamanho do modelo e a latência de inferência, viabilizando a execução local em aplicativos nativos desenvolvidos em Kotlin/Java para Android. O pipeline completo, incluindo filtros, normalização e mecanismos de masking, deve ser reimplementado de forma eficiente no aplicativo, levando em conta limitações de CPU, memória e consumo energético, além de validar a equivalência do comportamento do modelo após a conversão. Essas práticas e ferramentas para *deployment* em *edge devices* estão descritas na documentação oficial do TensorFlow e em estudos sobre inferência móvel, e têm sido empregadas em trabalhos que demonstram a viabilidade de classificação de sinais coletados por smartwatch*es*. (TensorFlow Documentation, 2023; TensorFlow, 2024b; ZANDE *et al.*, 2023; Samsung, 2024)

Dessa forma, este trabalho propõe o treinamento e validação de um modelo de CNN para detecção de fibrilação atrial utilizando a base de dados PhysioNet/CinC Challenge 2017 como fonte principal, complementada por outros três repositórios públicos (MIT-BIH Noise Stress Test Database, Motion Artifact Contaminated ECG Database e PhysioNet/CinC Challenge 2011) para enriquecimento de exemplos de ruído e artefatos. Essas bases contêm gravações de ECG unipolares rotuladas essenciais para o desenvolvimento de algoritmos robustos de classificação (GOLDBERGER *et al.*, 2000a; MOODY *et al.*, 1999; BEHRAVAN *et al.*, 2015; SILVA *et al.*, 2011). Por fim, espera-se verificar se o modelo é capaz de detectar FA por meio de dados de ECG coletados por um dispositivo Samsung Galaxy Watch 5, possibilitando uma implementação de inferência em tempo real em ambiente *mobile*.

1.1 Objetivos

1.1.1 Objetivo Geral

Verificar se um modelo de CNN é capaz de detectar fibrilação atrial a partir de sinais de ECG, de uma derivação, coletados por smartwatch.

1.1.2 Objetivos Específicos

- Avaliar a qualidade dos dados de ECG coletados pelo smartwatch, identificando ruídos e artefatos que possam comprometer a análise;
- Aplicar diferentes técnicas de filtragem e pré-processamento aos sinais de ECG e avaliar seu impacto no desempenho do modelo;
- Comparar o desempenho de diversas arquiteturas de redes neurais (CNN, CRN, BiLSTM) na detecção de fibrilação atrial;
- Investigar o efeito de técnicas de *data augmentation* (geração de sinais sintéticos, *style transfer*) para mitigar o desequilíbrio entre classes;
- Verificar a performance do modelo de CNN, incluindo métricas de acurácia, sensibilidade, especificidade e *F1-score*;
- Embarcar e utilizar o modelo em ambiente nativo.

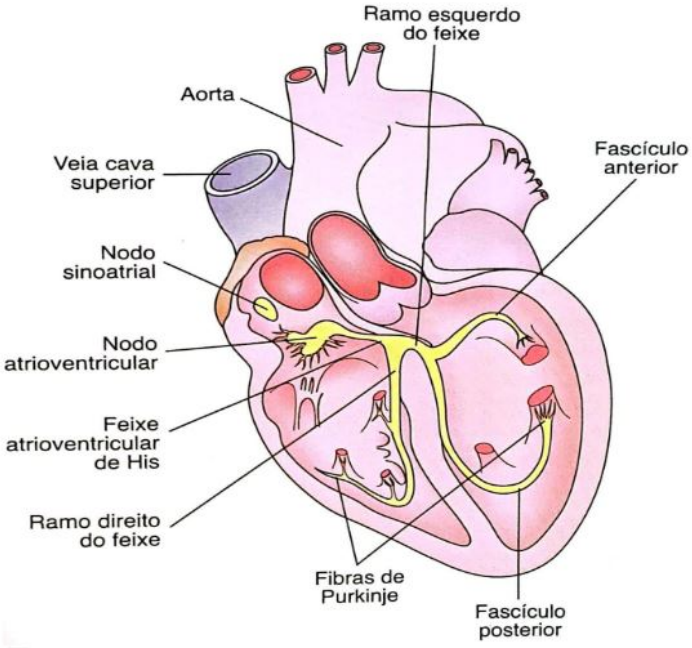
2 FUNDAMENTAÇÃO TEÓRICA

Nesta seção serão apresentados os conceitos fundamentais sobre o eletrocardiograma (ECG) e suas principais características de morfologia. Em cada subseção, serão introduzidos os tópicos de características do ECG, fibrilação atrial (FA), ECG em dispositivos vestíveis e modelos de *deep learning* aplicados à detecção de ritmos cardíacos.

2.1 Eletrocardiograma (ECG)

Para compreender o traçado de um ECG, é fundamental conhecer o mecanismo de condução elétrica do coração. A ativação elétrica cardíaca tem início no nódulo sinoatrial, localizado no átrio direito, que atua como marcapasso fisiológico e gera o impulso elétrico inicial. Esse estímulo propaga-se pelos átrios, alcança o nódulo atrioventricular — onde ocorre um atraso fisiológico — e segue pelo feixe atrioventricular (feixe de His), seus ramos direito e esquerdo e pelas fibras de Purkinje, promovendo a ativação sincronizada dos ventrículos. Os principais componentes desse sistema de condução elétrica, bem como sua relação com a contração cardíaca, estão ilustrados na Figura 1 (ALDUGHAYFIQ *et al.*, 2023).

Figura 1 – Esquema dos principais elementos do sistema de condução elétrica e da contração cardíaca.



Fonte: UFABC Divulga Ciência (2023).

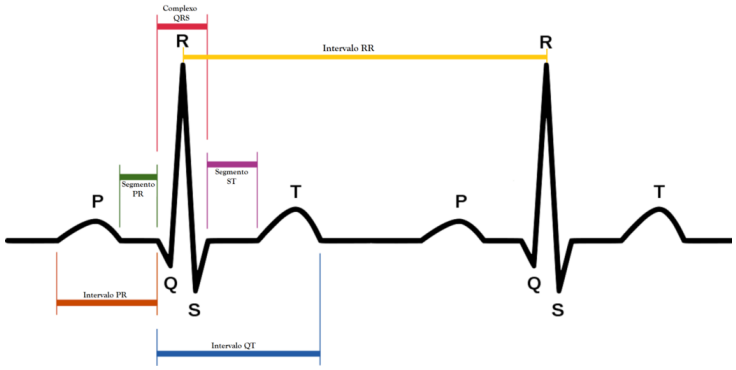
O ECG é um método não invasivo de registro dessa atividade elétrica, obtido por

meio de eletrodos colocados na superfície corporal. Ao medir a diferença de potencial gerada pela despolarização e repolarização das células miocárdicas, o ECG fornece informações sobre ritmo, frequência e condução dos impulsos cardíacos (ANDERSEN, 2019).

A Figura 2 ilustra o traçado padrão de um ECG, que apresenta ondas e complexos denominados P, Q, R, S e T, além dos intervalos e segmentos que os conectam. A onda P corresponde à despolarização atrial, indicando a contração dos átrios. O complexo QRS reflete a rápida despolarização ventricular e está associado à contração dos ventrículos, enquanto a onda T representa o processo de repolarização ventricular, ou relaxamento pós-sístole (XIONG *et al.*, 2022).

O intervalo PR (do início da onda P ao início do complexo QRS) reflete o tempo de condução atrioventricular, enquanto o intervalo QT (início do complexo QRS ao final da onda T) representa a duração total da despolarização e repolarização ventriculares. Dentre as métricas temporais derivadas do traçado, destaca-se o intervalo RR, que corresponde ao tempo transcorrido entre dois picos sucessivos da onda R. Essa medida é fundamental para o cálculo da frequência cardíaca e sua variabilidade, servindo como um indicador direto da regularidade do ritmo sinusal ou da presença de arritmias, como a FA, onde os intervalos RR tornam-se marcadamente irregulares (ANDERSEN, 2019). Alterações nesses intervalos podem indicar bloqueios de ramo, arritmias ou risco de eventos adversos, sendo essenciais para o diagnóstico clínico (FANG *et al.*, 2025). Logo, a análise do traçado de ECG é fundamental para o diagnóstico de anomalias cardíacas, como a FA.

Figura 2 – Ondas, segmentos e intervalos padrões do sinal de ECG, destacando a onda P, o complexo QRS, a onda T e os principais intervalos entre ondas.



Fonte: Mattos (2026).

Para interpretar essas ondas em diferentes regiões do miocárdio e aprimorar a precisão clínica, utilizam-se as derivações do ECG, que capturam a atividade elétrica do coração

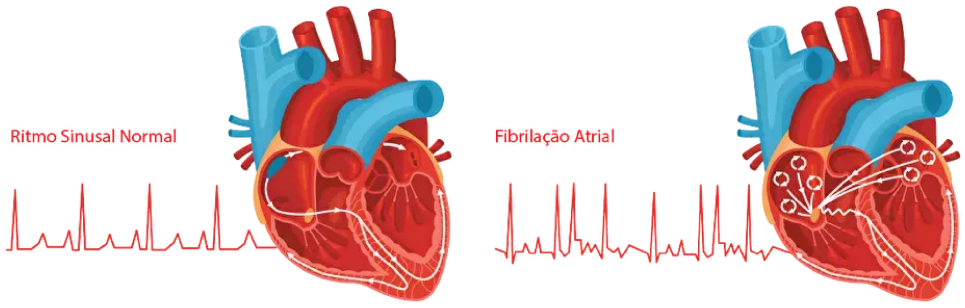
segundo distintos planos de visão. As derivações são classificadas em unipolares e bipolares, sendo as bipolares (I, II e III) obtidas pela diferença de potencial entre dois eletrodos e as unipolares (aVR, aVL, aVF e V1–V6) calculadas em relação a um ponto de referência central. Cada derivação proporciona uma "visão" do coração em diferentes planos, permitindo a localização precisa de alterações na condução elétrica (ALDUGHAYFIQ *et al.*, 2023). Essas diferentes visões do coração permitem identificar alterações morfológicas e temporais do traçado, fundamentais para distinguir ritmos normais de arritmias. Nesse contexto, destaca-se a fibrilação atrial, cujas manifestações eletrocardiográficas específicas são abordadas na próxima seção.

2.2 Fibrilação Atrial (FA)

A FA é uma arritmia caracterizada por descargas elétricas rápidas e desorganizadas nos átrios, resultando em contrações atriais ineficazes e perda da sístole atrial coordenada. Clinicamente, a FA está associada a risco aumentado de tromboembolismo, AVC, insuficiência cardíaca e mortalidade (ANDERSEN, 2019).

A identificação da FA em traçados de ECG baseia-se na ausência de ondas P regulares e na presença de intervalos RR irregulares e imprevisíveis, refletindo resposta ventricular irregular. Em ritmo sinusal, observam-se ondas P claras precedendo cada complexo QRS e intervalos RR constantes; em FA, há atonia atrial e variação aleatória dos intervalos RR, conforme ilustrado na Figura 3 (XIONG *et al.*, 2022).

Figura 3 – Traçado de ECG em ritmo sinusal (esquerda) e fibrilação atrial (direita).



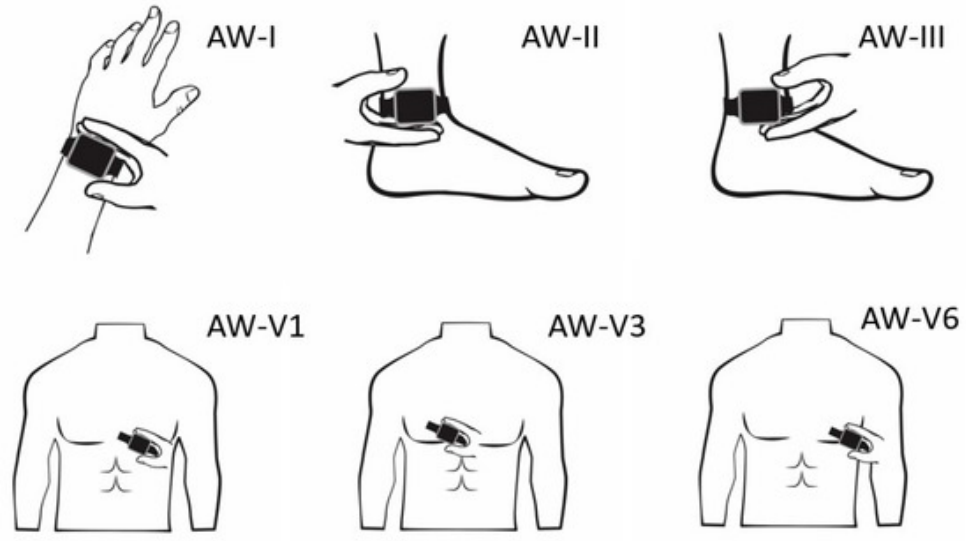
Fonte: Get Smart About AFib (2022).

Nos últimos anos, vários estudos vêm sendo desenvolvidos utilizando modelos de IA para detecção de FA. Dispositivos vestíveis equipados com ECG de derivação única (D1) permitem monitoramento contínuo e detecção precoce de FA em ambiente ambulatorial. Modelos de *deep learning*, como redes neurais convolucionais (CNN) de pequena profundidade, têm demonstrado elevada acurácia na classificação de episódios de FA a partir de batimentos curtos, mesmo em sinais com ruído típico de sensores de pulso vestíveis (FANG *et al.*, 2025).

2.3 ECG em Smartwatches

Os smartwatches modernos, como Apple Watch e Galaxy Watch, contam com ECG de derivação única (semelhante à derivação I) que é captada com dois contatos — um na parte traseira do relógio e outro no botão (coroa ou botão superior) tocado pelo dedo opositor (ZANDE *et al.*, 2023). Estudos de validação mostraram que, ao posicionar o *smartwatch* no tórax ou tornozelo, também é possível registrar derivações precordiais (V1, V3, V6), com boa concordância em relação ao ECG de 12 canais padrão, embora haja leve viés nas amplitudes de R em V1–V6 (ZANDE *et al.*, 2023). A Figura 4 ilustra esses posicionamentos alternativos, detalhando seis diferentes configurações de captação: a posição AW-I refere-se ao uso convencional no pulso para obter a derivação I; as posições AW-II e AW-III representam a coleta realizada com o relógio posicionado no pé ou tornozelo para obter as derivações II e III, respectivamente; e as posições AW-V1, AW-V3 e AW-V6 indicam a aplicação do dispositivo em diferentes pontos do tórax para registrar sinais equivalentes às respectivas derivações precordiais.

Figura 4 – Posicionamento alternativo para gravações precordiais com *smartwatch*.



Fonte: Zande *et al.* (2023).

A Samsung, com seus modelos Galaxy Watch Active2, Watch3, Watch5 e posteriores, recebeu aprovação do FDA e de órgãos reguladores internacionais para o recurso de ECG, disponível por meio do aplicativo Samsung Health Monitor. O recurso auxilia na detecção de FA em adultos acima de 22 anos (FDA, 2020). Entretanto, fatores como interferência por movimento, pele seca ou mau contato podem comprometer a qualidade do sinal (Samsung, 2024).

Entre as vantagens dos smartwatches estão o monitoramento contínuo, autocontrole e acessibilidade, promovendo detecção precoce de arritmias em usuários assintomáticos (ALDUGHAYFIQ *et al.*, 2023). Contudo, as limitações incluem a dependência de boas condições de contato, a restrição a uma única derivação e a ausência de avaliação de isquemia e outras anomalias, além da necessidade de confirmação médica em casos suspeitos (XIONG *et al.*, 2022).

Para melhorar a qualidade dos dados, recomenda-se apoio do braço durante a gravação, limpeza e hidratação da pele, treinamento do usuário e filtros de ruído, além de coleta de múltiplas amostras para validação cruzada (Samsung, 2024; ZANDE *et al.*, 2023). Assegurar uma boa captação dos dados, aliado à aplicação de técnicas de pré-processamento, é fundamental para otimizar o desempenho dos modelos de IA que visam à detecção de FA. Nesse contexto, os modelos de *deep learning*, especialmente as redes neurais convolucionais (CNNs), emergem como uma abordagem promissora para extrair padrões complexos dos sinais ECG e realizar a classificação automatizada de ritmos cardíacos.

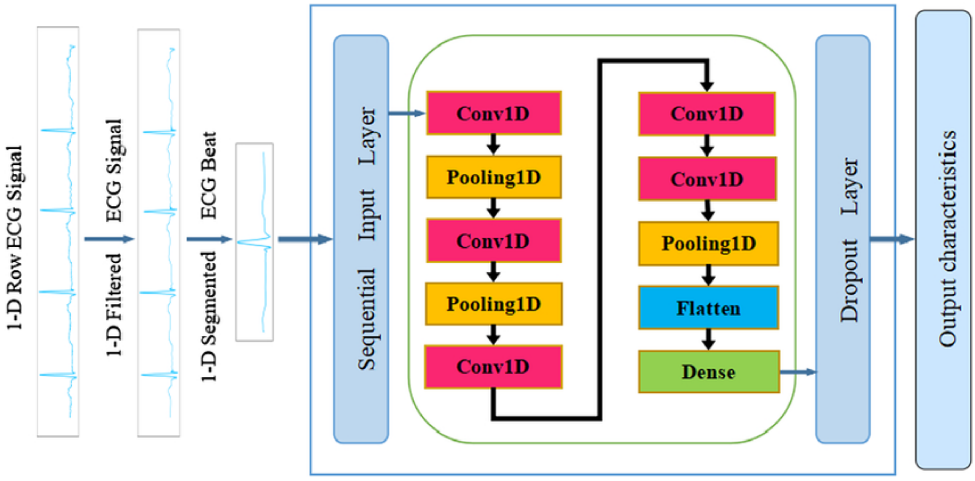
2.4 Modelos de *Deep Learning*

As redes neurais são estruturas computacionais inspiradas no sistema nervoso biológico, compostas por camadas de neurônios artificiais que processam sinais de entrada por meio de funções de ativação e pesos sinápticos ajustáveis (ANDERSEN, 2019). Em uma rede neural profunda (DNN), múltiplas camadas ocultas permitem a extração hierárquica de características complexas, favorecendo o reconhecimento de padrões não lineares em dados de ECG (ANDERSEN, 2019).

As redes convolucionais (CNNs) são um tipo de DNN especialmente eficaz para sinais unidimensionais como o ECG, graças ao uso de filtros (*kernels*) que realizam convoluções ao longo da série temporal. Cada filtro aprende a detectar características locais, como picos R ou padrões de onda, enquanto camadas posteriores agregam essas informações em conceitos globais, como ritmo regular ou irregular (XIONG *et al.*, 2022). A Figura 5 ilustra esse fluxo

de processamento: o *1-D Raw ECG signal* refere-se ao sinal bruto captado pelo sensor, o *1-D Filtered ECG Signal* indica o mesmo sinal após aplicação de filtros digitais (remoção de baseline wander e ruídos de alta frequência) e o *1-D Segmented ECG Beat* representa a divisão em janelas ou batimentos usados como entrada da rede. Na arquitetura mostrada, as camadas `Conv1D` e `Pooling1D` extraem representações locais e reduzem a resolução temporal, e blocos densos finais (Flatten → Dense → Dropout) combinam essas características para produzir as saídas de classificação.

Figura 5 – Arquitetura simplificada de uma CNN aplicada a sinais de ECG.



Fonte: He *et al.* (2022).

As principais vantagens do uso de CNN na detecção de FA incluem a capacidade de extração automática de características relevantes, eliminando a necessidade de engenharia manual de sinais (ANDERSEN, 2019), a robustez a ruídos e variações de amplitude típicas de sinais de *smartwatches* (FANG *et al.*, 2025) e a escalabilidade para processar grandes volumes de dados, aproveitando Unidades de Processamento Gráfico (GPUs) para treinamento rápido (ALDUGHAYFIQ *et al.*, 2023).

Contudo, há desafios a serem superados: a escassez de bases rotuladas específicas para sinais vestíveis, o que dificulta o treinamento supervisionado (XIONG *et al.*, 2022); o desequilíbrio entre batimentos normais e episódios de FA, que exige técnicas de *data augmentation* ou reamostragem para balancear as classes (ZANDE *et al.*, 2023); e a sensibilidade da seleção de hiperparâmetros, como tamanho de *kernel*, profundidade de rede e taxa de aprendizado — ao *overfitting* em conjuntos de dados pequenos (ANDERSEN, 2019).

Pesquisas recentes propuseram soluções como redes híbridas CRN (convolucional + recorrente) para capturar dependências temporais de longo prazo e geradores de estilo (*style*

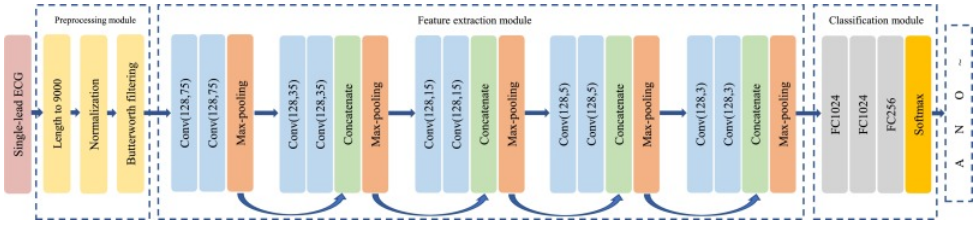
transfer) para criar exemplos sintéticos de batimentos de FA, aumentando a variabilidade do conjunto de treinamento sem necessidade de novos pacientes (XIONG *et al.*, 2022).

Além disso, modelos como o K-B2S+ aplicam técnicas de redução de perda de características por concatenação de mapas de ativação em diferentes camadas convolucionais, melhorando a precisão na classificação de sinais ruidosos de dispositivos vestíveis (FANG *et al.*, 2025). A estrutura detalhada dessa arquitetura é apresentada na Figura 6, sendo organizada em três estágios fundamentais: o pré-processamento, a extração de características e a classificação.

O módulo de pré-processamento é responsável por tratar o sinal bruto de ECG coletado pelo sensor, aplicando filtros digitais (como o filtro Butterworth) para remover ruídos de alta frequência e flutuações da linha de base, além da normalização de amplitude para garantir que os dados estejam em uma escala compatível com a rede neural. Esta etapa é crucial para elevar a relação sinal-ruído e garantir a estabilidade do treinamento, minimizando o impacto de artefatos de contato. Em seguida, o módulo de extração de características utiliza camadas convolucionais 1D sequenciais aliadas a operações de *MaxPooling*. Nessa etapa, destaca-se a estratégia de *big-to-small kernels*, em que as primeiras camadas utilizam filtros maiores para capturar o contexto global e a variabilidade do ritmo, enquanto as camadas posteriores utilizam filtros menores para identificar detalhes morfológicos finos (como a presença ou ausência de ondas P). O *MaxPooling* atua reduzindo a dimensionalidade dos dados e conferindo invariância a pequenas variações temporais do sinal, tornando o modelo mais robusto (FANG *et al.*, 2025).

Por fim, o módulo de classificação recebe as representações de alto nível extraídas e, por meio de camadas densas (*Fully Connected*) e uma camada de saída *softmax*, realiza a predição final da categoria do ritmo cardíaco. A integração desses módulos permite que o modelo K-B2S+ preserve informações essenciais ao longo de toda a rede por meio de conexões de atalho, mitigando a perda de detalhes clínicos em sinais ruidosos típicos de *smartwatches* (FANG *et al.*, 2025).

Figura 6 – Representação modular da arquitetura K-B2S+ englobando etapas de pré-processamento, extração de características e classificação.



Fonte: Fang *et al.* (2025).

Portanto, as CNNs representam uma das abordagens mais promissoras para o diagnóstico automatizado de FA em ECGs de smartwatches, desde que acompanhadas de estratégias robustas de pré-processamento e balanceamento de dados. A seguir, apresenta-se a biblioteca `TensorFlow`: por que escolhê-la para construir e treinar CNNs aplicadas a ECG e como ela facilita a otimização e conversão para `TensorFlow Lite` visando o deployment em dispositivos móveis.

2.5 TensorFlow

O *TensorFlow* é uma plataforma de código-aberto para construção, treino e *deploy* de modelos de aprendizado de máquina desenvolvida pelo Google. Ela combina uma API de baixo nível eficiente com interfaces de alto nível (principalmente `tf.keras`), permitindo tanto experimentação rápida quanto implantação em produção (servidores, dispositivos móveis, web e aceleradores como GPU/TPU). (ABADI *et al.*, 2016; TensorFlow Development Team, 2015).

As principais vantagens de utilizar o *TensorFlow* na construção de modelos são: (i) portabilidade (execução em múltiplas plataformas, incluindo TensorFlow Lite e TF.js), (ii) suporte a treinamento distribuído e aceleração por GPU/TPU, (iii) APIs de alto nível (`tf.keras`) que agilizam a prototipagem e (iv) um ecossistema maduro com ferramentas para visualização (TensorBoard), otimização e *deployment*. Essas características tornam o TensorFlow adequado tanto para pesquisa quanto para aplicações clínicas que necessitam de inferência em produção. (TensorFlow Documentation, 2023; GÉRON, 2019).

No contexto de modelos para ECG coletados por *smartwatches*, o TensorFlow apresenta benefícios práticos: facilidade para implementar arquiteturas 1D (Conv1D), integração direta com ferramentas de pré-processamento em Python, pipelines de *data augmentation*, e caminhos claros para conversão do modelo (TF Lite / ONNX) para execução em dispositivos com recursos limitados. Além disso, `tf.keras` simplifica a experimentação de hiperparâmetros e o uso de *callbacks* (*early stopping*, redução de *learning rate*) importantes para evitar *overfitting* em bases pequenas. (CHOLLET, 2017; GÉRON, 2019).

No treino, recomenda-se utilizar *callbacks* que auxiliem a convergência e evitem *overfitting*, como `EarlyStopping` (para interromper o treino quando a métrica de validação não mais melhorar) e `ReduceLROnPlateau` (para reduzir automaticamente a taxa de aprendizado quando o progresso estagnar). Além disso, é importante aplicar estratégias de balanceamento das classes — por exemplo, atribuição de pesos de classe ou *oversampling* das janelas de FA —

e técnicas de *data augmentation* apropriadas para sinais temporais (*jitter*, escala, deslocamento temporal, *mixup*), de modo a compensar o desbalanceamento entre episódios normais e episódios de FA. Finalmente, durante o treino e a validação deve-se monitorar métricas clínicas relevantes — sensibilidade, especificidade e F1-score — e não apenas acurácia, pois essas métricas refletem melhor o desempenho clínico do classificador. (GÉRON, 2019; CHOLLET, 2017).

Após treinar e validar o modelo, existem diversos caminhos de *deployment* suportados pelo TensorFlow. Uma opção é exportar o modelo no formato `SavedModel` e disponibilizá-lo via servidor de inferência (por exemplo, TensorFlow Serving) ou como uma API REST construída sobre *frameworks* como FastAPI, podendo também utilizar runtimes compatíveis com ONNX/TF para integração em ecossistemas existentes. Para execução em dispositivos móveis ou *edge*, recomenda-se otimizar o modelo e convertê-lo para `TFLite`, fazendo uso de técnicas de quantização (int8, float16) para reduzir latência e consumo de memória. Por fim, se houver necessidade de inferência no navegador, o modelo pode ser convertido para `tfjs`. Essas opções fornecem flexibilidade para transitar do protótipo de pesquisa para soluções de inferência em produção, em diferentes plataformas. (TensorFlow Development Team, 2015; TensorFlow Documentation, 2023).

Em resumo, o TensorFlow oferece um conjunto completo e maduro de ferramentas para pesquisa e produção de modelos de detecção de FA a partir de ECGs de smartwatch, combinando facilidade de prototipagem (`tf.keras`), capacidade de escala e caminhos claros para deployment em dispositivos móveis e servidores clínicos (ABADI *et al.*, 2016; TensorFlow Development Team, 2015; GÉRON, 2019; CHOLLET, 2017).

2.6 Ambiente Mobile e TensorFlow Lite

O desenvolvimento de aplicações móveis que executam inferência de modelos de aprendizado de máquina exige a integração de ferramentas de desenvolvimento nativas e *runtimes* leves otimizados para dispositivos com recursos limitados. No ecossistema Android, as principais linguagens e tecnologias são Kotlin e Java (com Kotlin sendo atualmente a linguagem recomendada pelo Google), enquanto no iOS as aplicações são normalmente escritas em Swift ou Objective-C; *frameworks cross-platform como Flutter (Dart)* também são largamente usados para acelerar o desenvolvimento multiplataforma. Essas plataformas oferecem APIs específicas para I/O, threads, acesso a sensores (por exemplo, *Bluetooth* para smartwatches) e integração com aceleradores locais (NNAPI no Android), tornando-as adequadas para implantação de

modelos de inferência em tempo real em dispositivos móveis (JEMEROV; ISAKOVA, 2017; Android Developers / TensorFlow, 2024).

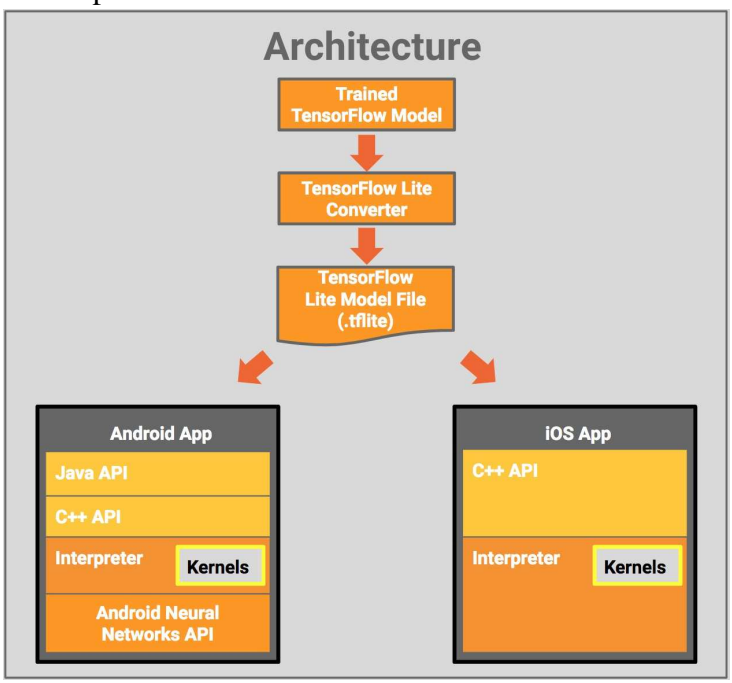
O TensorFlow Lite (TFLite) é o *runtime* e o formato de modelo projetado para executar modelos de TensorFlow (e outros modelos convertidos) em dispositivos móveis e *edge*. O TFLite fornece um conversor para gerar arquivos ‘.tflite‘ (FlatBuffer) a partir de modelos TensorFlow/Keras, além de ferramentas para otimização (quantização, poda, etc.) e delegates para acelerar a inferência usando *hardware* local (GPU delegate, NNAPI, ou delegados específicos do fornecedor). Entre as vantagens estão a redução do tamanho do modelo, compatibilidade com otimizações (post-training quantization e quantization-aware training), e bibliotecas de suporte que facilitam a implantação em Android e iOS (TensorFlow, 2024b; TensorFlow, 2024a).

O fluxo típico para levar uma CNN treinada em TensorFlow para um aplicativo móvel consiste nas seguintes etapas: (1) treinar e validar o modelo em ambiente desktop/servidor usando TensorFlow/Keras; (2) exportar o modelo (SavedModel ou Keras .h5); (3) converter o modelo para o formato TFLite usando o ‘TFLiteConverter‘, aplicando opções de otimização como quantização post-training para reduzir tamanho e latência; (4) testar a inferência localmente com o ‘tflite‘ interpreter; (5) empacotar o arquivo ‘.tflite‘ no app mobile e integrar a inferência via APIs do TensorFlow Lite (Interpreter / Task Library) ou geradores de wrapper do Android Studio. A documentação oficial do conversor e guias de otimização explicam parâmetros comuns e trade-offs entre precisão e desempenho, conforme ilustrado na Figura 7.

Durante a conversão é comum aplicar técnicas de otimização. A quantização pós-treinamento (post-training quantization) reduz o tamanho do modelo e frequentemente melhora a latência em CPU, com perda mínima de acurácia; quando a perda é significativa, a quantization-aware training (QAT) pode ser aplicada durante o treinamento para preservar precisão após quantização (TensorFlow Model Optimization, 2022).

No Android, o modelo ‘.tflite‘ é carregado no app e executado por meio do ‘Interpreter‘ (ou usando a Task Library para casos de visão/áudio/texto com wrappers de alto nível). Para melhorar desempenho, o TensorFlow Lite permite o uso de delegates que aproveitam aceleradores: NNAPI (acessa aceleradores do SoC), GPU delegate e outros delegados de fornecedor; a escolha do delegate depende do dispositivo e do perfil de latência/consumo energético desejado. Também há suporte para conversão com quantização (int8/float16) e para geração de metadados que facilitam integração com ferramentas do Android Studio (ML Model Binding) (Google AI Edge, 2024b; Android Developers, 2025).

Figura 7 – Diagrama de conversão de um modelo treinado em TensorFlow para TensorFlow Lite e uso em um aplicativo móvel.



Fonte: Moroney (2018).

Para iOS, o TensorFlow Lite fornece bindings em Swift/Objective-C, e o fluxo de conversão é análogo; para multiplataforma, bibliotecas como Flutter têm pacotes que permitem carregar modelos TFLite. Ao projetar o app, recomenda-se medir latência e consumo energético em dispositivos reais, testar com delegates específicos e validar a acurácia do modelo quantizado em dados reais colhidos pelo smartwatch/smartphone (TensorFlow, 2024a; Android Developers / TensorFlow, 2024).

Além do caminho clássico de ‘SavedModel -> TFLite‘, o ecossistema Google vem consolidando o L̈ite Runtime(̈LiteRT) e ferramentas de otimização para edge, que ampliam opções de conversão e aceleração. Para aplicações médicas como detecção de FA, é crucial documentar e validar qualquer degradação de desempenho causada por otimizações (especialmente quantização), bem como garantir conformidade regulatória e segurança de dados no app móvel (Google AI Edge, 2024a).

Em suma, a utilização do TensorFlow Lite no ambiente nativo mobile permite executar modelos de CNN com desempenho adequado em dispositivos móveis, desde que se adotem práticas de otimização (quantização, delegates), integração correta com APIs nativas (Kotlin/Java/Swift) e validação detalhada do comportamento do modelo com dados colhidos por *smartwatches*. As referências oficiais e guias práticos são fontes essenciais para implementar esse fluxo de forma segura e eficiente.

2.7 Trabalhos Relacionados

Diversos estudos têm investigado a aplicação de técnicas de *Deep Learning* para a detecção de fibrilação atrial (FA) em sinais de ECG, cada um trazendo avanços e apontando lacunas para pesquisas futuras.

Por exemplo, Aldughayfiq *et al.* (2023) desenvolveram um modelo híbrido composto por uma Rede Neural Convolucional unidimensional (1D CNN) e uma Memória de Longo e Curto Prazo Bidirecional (*Bidirectional Long Short-Term Memory* – BiLSTM) para classificar FA. O método utilizou sinais de eletrocardiograma (ECG) e fotopletismografia (PPG) provenientes da base de dados MIMIC PERform AF. O modelo atingiu 95% de acurácia em testes com sinais de PPG transmissivo. No entanto, o estudo ainda não foca em sinais coletados por smartwatches, o que limita a aplicabilidade ambulatorial dessa abordagem.

Complementando essa linha de pesquisa, Fang *et al.* (2025) propuseram o K-B2S+, uma CNN de *big-to-small kernels* para janelas curtas de ECG de dispositivo vestível. Utilizando a base de dados PhysioNet/Computing in Cardiology Challenge 2017, o modelo obteve acurácia de 93,8%, sensibilidade de 84,3%, especificidade de 92,1% e F1-score médio de 0,854, demonstrando que kernels maiores nas camadas iniciais melhoram a captura de padrões globais. Devido ao seu desempenho promissor, o K-B2S+ foi selecionado como modelo base para este trabalho. No entanto, sua eficácia ainda não foi validada em bases mais heterogêneas, tampouco avaliou-se a generalização a diferentes condições de ruído.

Adicionalmente, Xiong *et al.* (2022) também utilizaram a base de dados PhysioNet/Computing in Cardiology Challenge 2017, combinando redes convolucionais e recorrentes com transferência de estilo (*style transfer*) para gerar ECG sintético e superar o desequilíbrio de classes. Esse método permitiu ganhos de 3 pontos percentuais (p.p.) no F1-score — o que representa um aumento absoluto na média das métricas de precisão e revocação — demonstrando a eficácia da síntese de dados para mitigar o desbalanceamento. Contudo, apesar de utilizarem a mesma base de sinais de dispositivos móveis, o estudo não avalia a generalização do modelo para condições de ruído mais severas e variadas, nem aborda os requisitos para a implantação eficiente em *smartwatches* com recursos computacionais limitados.

Em uma linha voltada à detecção em tempo real, Andersen (2019) apresentaram uma arquitetura CNN+LSTM para detecção de FA em ECG de longa duração, utilizando os conjuntos de dados MIT-BIH AF Database (AFDB) (MOODY; MARK, 1983), MIT-BIH Arrhythmia Database (MITDB) (MOODY; MARK, 2001) e MIT-BIH NSR Database (NSRDB)

(GOLDBERGER *et al.*, 2000b). O modelo obteve sensibilidade de 98,98% e especificidade de 96,95%, todavia, não considera os desafios de captura por *wearables* nem as restrições de processamento em dispositivos móveis.

No que diz respeito à aquisição de sinais, Zande *et al.* (2023) validaram a captação de derivações precordiais com *smartwatch*, mostrando boa concordância com ECG de 12 derivações para V1–V6. Contudo, enfrentaram viés nas amplitudes e não investigaram o impacto em algoritmos de *Deep Learning* treinados exclusivamente com sinais padrão.

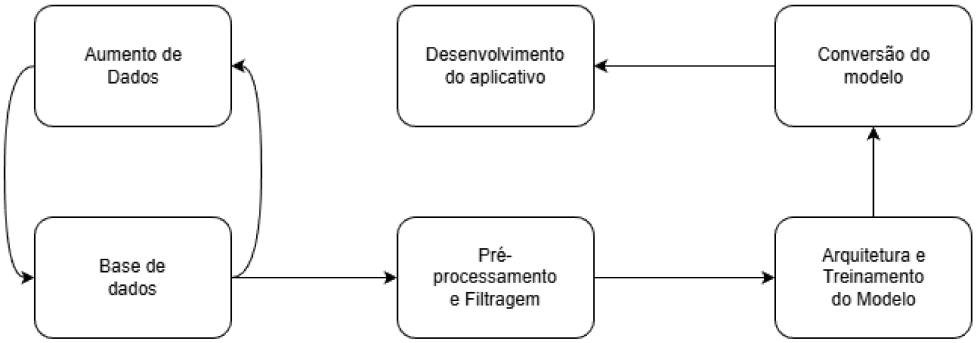
Em síntese, embora a literatura demonstre a viabilidade de CNNs para detecção de FA, persistem lacunas importantes: a falta de bases rotuladas específicas de ECG por *smartwatch*, a necessidade de avaliar a robustez a artefatos de movimento e o desafio de implementar esses modelos em ambiente móvel com restrições de memória e energia. Esta pesquisa buscará preencher tais lacunas, focando na comparação sistemática de protocolos de pré-processamento e no desenvolvimento de um modelo CNN otimizado para classificação de dados coletados por *smartwatch*, visando uma implantação de inferência em tempo real.

3 METODOLOGIA

Este capítulo descreve a metodologia adotada para a construção e avaliação do sistema de detecção de fibrilação atrial a partir de sinais de ECG. O desenvolvimento e treinamento dos modelos foram realizados utilizando a linguagem Python em um computador equipado com processador Intel Core i5, 16 GB de memória RAM e placa de vídeo NVIDIA GTX 1050. Para a etapa de inferência embarcada, o aplicativo móvel foi desenvolvido na linguagem Kotlin e os testes foram conduzidos em um smartphone Samsung Galaxy M54. A coleta dos sinais de ECG para validação em ambiente real utilizou um *smartwatch* Samsung Galaxy Watch 5.

Conforme ilustrado na Figura 8, o fluxo de desenvolvimento do projeto foi organizado em etapas bem definidas: base de dados, aumento de dados, pré-processamento e filtragem, treinamento e avaliação do modelo, conversão do modelo e desenvolvimento do aplicativo móvel para inferência embarcada.

Figura 8 – Fluxo de desenvolvimento.



Fonte: Elaborado pelo autor.

3.1 Base de Dados

Este trabalho utiliza quatro conjuntos públicos do repositório PhysioNet, selecionados de forma a prover sinais clínicos e exemplos de artefatos necessários para treinar e validar um classificador de ECG com quatro rótulos (Normal, Fibrilação Atrial, Outros e Ruído). A fonte principal é o conjunto da competição PhysioNet/Computing in Cardiology Challenge 2017, que contém 8528 registros de ECG de derivação única (derivação I), coletados por dispositivos portáteis com duração variável entre 9 s e 60 s (aproximadamente 30 s na maior parte dos casos), taxa de amostragem tipicamente de 300 Hz e resolução de 16 bits, no formato WFDB (GOLDBERGER *et al.*, 2000a). No conjunto público de treinamento desta base a distribuição

aproximada por classe é: 5 154 registros rotulados como ritmo sinusal normal (N), 771 registros de fibrilação atrial (FA), 2 557 registros classificados como outros ritmos (O) e 46 registros rotulados como ruído (GOLDBERGER *et al.*, 2000a).

Como complemento para aumentar a representatividade de artefatos e ruídos, foram utilizadas bases específicas do PhysioNet: a Noise Stress Test Database (NSTDB) (MOODY *et al.*, 1999), a Motion Artifact Contaminated ECG Database (MACECGDB) (BEHRAVAN *et al.*, 2015) e o conjunto da competição PhysioNet/CinC 2011 (SILVA *et al.*, 2011). A união dessas bases com os registros de ruído do desafio de 2017 resultou em um conjunto inicial de apenas 551 amostras reais de ruído. Dada a escassez de dados nessa classe crítica para a robustez do modelo, aplicou-se um protocolo intensivo de geração de dados sintéticos (*stitching*, *baseline wander*, ruído muscular e interferência de rede), produzindo 3.545 novos exemplos e totalizando 4.096 amostras de ruído para treinamento.

3.2 Aumento de Dados

Para garantir o balanceamento e a diversidade do conjunto de dados, utilizou-se a combinação de múltiplas bases de dados reais, além de um protocolo específico de aumento de dados (*data augmentation*) para a classe de ruído, dada a sua criticidade e baixa representatividade inicial. As estratégias aplicadas visam melhorar a robustez do modelo a artefatos sem comprometer a autenticidade dos sinais biológicos das classes principais (XIONG *et al.*, 2022).

Especificamente, gera-se ruído sintético por quatro mecanismos principais: costura (*stitching*) (ZIHLMANN *et al.*, 2017; LUO *et al.*, 2017) de dois sinais distintos colados em um ponto aleatório; *baseline wander*, mediante adição de ondas senoidais de baixa frequência (0,1–0,7 Hz); ruído muscular (EMG), obtido adicionando ruído gaussiano proporcional ao desvio padrão do sinal; e interferência de rede elétrica (senoide de 60 Hz) (MOODY *et al.*, 1984). A aplicação desses procedimentos resultou na geração de milhares de exemplos ruidosos (por exemplo, 3 545 ruídos sintéticos obtidos por *stitching*, *baseline wander*, EMG e interferência de rede), complementando o conjunto original e aumentando a robustez do modelo a artefatos reais de aquisição.

Para balancear as classes, utilizou-se a integração de registros provenientes de bases complementares para as classes FA e O, reduzindo o desbalanceamento em relação à classe majoritária (N). O objetivo foi manter uma proporção representativa que evitasse o viés excessivo, mas que ainda refletisse a abundância natural desses exames. A distribuição final das amostras,

detalhando a contribuição de cada fonte e o impacto do processo de aumento de dados na classe de ruído, é apresentada na Tabela 1 (GOLDBERGER *et al.*, 2000a; MOODY *et al.*, 1999; BEHRAVAN *et al.*, 2015; SILVA *et al.*, 2011).

Tabela 1 – Distribuição das amostras por classe e origem

Classe	PhysioNet 2017	Outras Bases (Real)	Data *augmentation* (Sintético)	Total Final
Normal (N)	5.154	26.590	-	31.744
Fibrilação Atrial (FA)	771	3.837	-	4.608
Outros (O)	2.557	7.683	-	10.240
Ruído (R)	46	505	3.545	4.096
Total	**8.528**	**38.615**	**3.545**	**50.688**

Fonte: Elaborado pelo autor.

Essa distribuição orienta as decisões de amostragem e permite a definição de pesos de classe durante o treinamento quando o *oversampling* completo não é desejável (GÉRON, 2019; XIONG *et al.*, 2022).

3.3 Pré-processamento e Filtragem

Antes da aplicação dos modelos de aprendizado profundo, os sinais de ECG passam por um pipeline de processamento para garantir a uniformidade dos dados e mitigar interferências. O fluxo seguido nesta etapa, abrangendo a segmentação, filtragem e normalização, é exemplificado na Figura 9.

Figura 9 – Fluxo de pré-processamento.

Fonte: Elaborado pelo autor.

Inicialmente, cada gravação é segmentada em janelas temporais de 30 s para padronizar a análise do ritmo cardíaco e preservar características de variabilidade. Em seguida, cada janela é ajustada para um comprimento fixo de 8.000 pontos, quando a janela contém mais amostras, ela é truncada; quando contém menos, é preenchida por *zero-padding*. Essa padronização facilita entrada fixa para redes neurais e pipelines de *batch*, além de simplificar a conversão para formatos de inferência embarcada.

A filtragem digital é aplicada para atenuar ruídos e remover componentes indesejados.

O pipeline utiliza filtros Butterworth aplicados em *zero-phase* por meio de `filtfilt` para evitar distorções de fase. As frequências de corte são fixas em 1 Hz (passa-alta) e 30 Hz (passa-baixa) e a ordem do filtro é fixa (ordem 4). Os parâmetros escolhidos são validados empiricamente nos conjuntos de referência e em amostras coletadas por *smartwatch*. (CLIFFORD *et al.*, 2017; ZANDE *et al.*, 2023)

Após a filtragem, aplica-se normalização de amplitude por Min–Max para escalonar os valores ao intervalo $[0, 1]$, reduzindo variações interindividuais e diferenças de ganho entre sensores. A normalização Min–Max melhora a estabilidade numérica durante o treinamento e a inferência, especialmente quando se utiliza ativação sigmoide ou quando se realiza quantização para inferência móvel (GÉRON, 2019; TensorFlow Documentation, 2023).

Para implementação e reprodutibilidade, o pipeline utiliza bibliotecas científicas em Python (por exemplo, `NumPy`, `pandas` e `SciPy`) e bibliotecas especializadas para sinais fisiológicos (`NeuroKit2`, quando aplicável). As funções típicas usadas são `scipy.signal.butter` para projeto do filtro e `scipy.signal.filtfilt` para aplicação *zero-phase*, garantindo mínima distorção temporal das ondas relevantes (ex.: picos R).

3.4 Arquitetura e Treinamento do Modelo

A estrutura detalhada da rede neural convolucional proposta, abrangendo desde a entrada do sinal de ECG até a classificação final, é ilustrada na Figura 10. O diagrama apresenta a organização dos blocos convolucionais, as operações de pooling e as conexões de fusão multi-escala que caracterizam o modelo Dense Multi-Kernel System (DMKS).

A arquitetura adotada neste trabalho, denominada DMKS, é uma rede neural convolucional unidimensional (CNN1D) profunda, projetada para classificação multi-classe de sinais de ECG nas categorias Fibrilação Atrial (FA), Ritmo Sinusal Normal (N), Ruído (R) e Outras Arritmias (O). O modelo prioriza a extração multi-escala de características morfológicas por meio de cinco blocos convolucionais sequenciais e incorpora conexões densas progressivas entre blocos para promover a fusão sinérgica de representações em diferentes resoluções (FANG *et al.*, 2025).

O processamento inicial dos sinais parte de uma camada de *Masking* que ignora valores nulos (*zero-padding*) introduzidos para padronizar janelas em comprimento fixo. Cada um dos cinco blocos convolucionais contém duas camadas Conv1D com 128 filtros e função de ativação ReLU. Os tamanhos de *kernel* são progressivamente reduzidos (75, 35, 15, 5 e 3), de

modo a capturar tanto componentes de baixa frequência e variabilidade de longo prazo (úteis para avaliar irregularidade dos intervalos R-R) quanto detalhes de alta frequência associados à morfologia do QRS. Entre os blocos são aplicadas operações de *pooling* com fatores de redução 10, 10, 5, 5 e 2 para diminuir progressivamente a dimensão temporal. A partir do segundo bloco, as entradas dos estágios são construídas pela concatenação das saídas do bloco imediatamente anterior com as representações de todos os blocos precedentes, formando uma topologia de conexões densas que preserva e redistribui informação ao longo da hierarquia convolucional.

Para viabilizar a fusão de mapas de características com comprimentos temporais distintos devido ao MaxPooling1D, o modelo emprega uma camada de redimensionamento (implementada via camada Lambda com interpolação bilinear) que ajusta os tensores para um comprimento comum antes da concatenação, permitindo que o classificador final integre informações multi-escala de forma coerente. A agregação entre a etapa convolucional e o classificador é feita por meio de GlobalAveragePooling1D, reduzindo a dimensionalidade sem recorrer a *Flatten* e, assim, diminuindo substancialmente a contagem de parâmetros e o risco de *overfitting*.

O bloco de classificação é composto por três camadas densas (256, 128 e 64 neurônios), com aplicação de *Dropout* progressivo (0,5; 0,3; 0,2) para reforçar a capacidade de generalização. A camada de saída utiliza ativação *softmax* para produzir a distribuição de probabilidade sobre as classes alvo, permitindo inferência multi-classe direta.

O treinamento é realizado sobre o conjunto particionado de forma estratificada em 70% treino, 15% validação e 15% teste, preservando as proporções de cada classe nas partições. A função de perda adotada é *Sparse Categorical Crossentropy* e o otimizador é Adam com taxa de aprendizado inicial de 1×10^{-4}. Para mitigar o desequilíbrio entre classes, foi implementado cálculo de pesos de classe balanceados com um teto de ponderação (*cap*) igual a 3.0, estratégia que evita atualizações de gradiente excessivamente ruidosas causadas por classes muito raras e favorece estabilidade durante a otimização (GÉRON, 2019).

Durante o processo de otimização são empregadas políticas de ajuste dinâmico: o treinamento é monitorado por *EarlyStopping* com paciência de 15 épocas e por *ReduceLROnPlateau*, que reduz a taxa de aprendizado por um fator de 0,5 após 10 épocas sem melhora na perda de validação. Além disso, realiza-se *checkpointing* do melhor modelo segundo a métrica definida (por exemplo, F1-score na classe FA), garantindo conservação do estado com melhor desempenho clínico. Quando o *oversampling* global não é desejável, opta-se por amostragem

balanceada em *batch* ou uso de pesos de classe para mitigar viés do classificador em relação à classe majoritária.

A busca e ajuste de hiperparâmetros (por exemplo, número de filtros, tamanhos de *kernel*, profundidade de rede e tamanho do lote) são efetuados por busca em grade ou por métodos bayesianos em espaços maiores de hipótese, avaliando a robustez em múltiplas execuções com diferentes *seeds*. A avaliação final considera métricas clinicamente relevantes, incluindo acurácia, sensibilidade (*recall*), especificidade, precisão, F1-score por classe e AUC-ROC quando aplicável, priorizando sensibilidade e F1 para a classe FA em razão de seu valor diagnóstico (ANDERSEN, 2019). Análises complementares, como curvas de aprendizado, matrizes de confusão e estudos de ablação, são utilizadas para diagnosticar efeitos de *under/overfitting* e para quantificar o impacto das estratégias de pré-processamento e augmentação na robustez do modelo.

3.5 Implantação do Modelo em Aplicativo Mobile

A etapa final do fluxo de desenvolvimento consiste na implantação do classificador treinado em um aplicativo móvel nativo para a plataforma Android. O aplicativo implementa a lógica necessária para comunicar-se com o *smartwatch* (via Bluetooth/BLE), capturar janelas de ECG de 30 s, executar o pré-processamento localmente e submeter os dados ao modelo para inferência em tempo quase real.

3.5.1 Conversão do modelo

O modelo Keras/TensorFlow treinado é exportado inicialmente no formato padrão `SavedModel` e, em seguida, convertido para TensorFlow Lite (`.tflite`) para execução em dispositivos móveis. Durante a conversão realizam-se otimizações de quantização pós-treinamento (por exemplo, `float16` ou `int8`) e poda de operações não essenciais, com o objetivo de reduzir o tamanho do artefato e acelerar a inferência mantendo acurácia aceitável. Quando se aplica quantização inteira (`int8`), utiliza-se um conjunto representativo de amostras para calibrar os pontos de escala, de modo a minimizar a perda de precisão. O arquivo final (`model.tflite`) é testado em comparação com a versão original em termos de acurácia, latência e uso de memória, e são avaliadas opções de delegados de execução (por exemplo, NNAPI, GPU delegate) para aproveitar aceleradores de hardware quando disponíveis. A estratégia adotada segue as boas práticas de *deployment* recomendadas pelo TensorFlow, incluindo salvaguardas para reverter a

quantização caso a degradação de métricas seja inaceitável (TensorFlow Documentation, 2023; TensorFlow, 2024b).

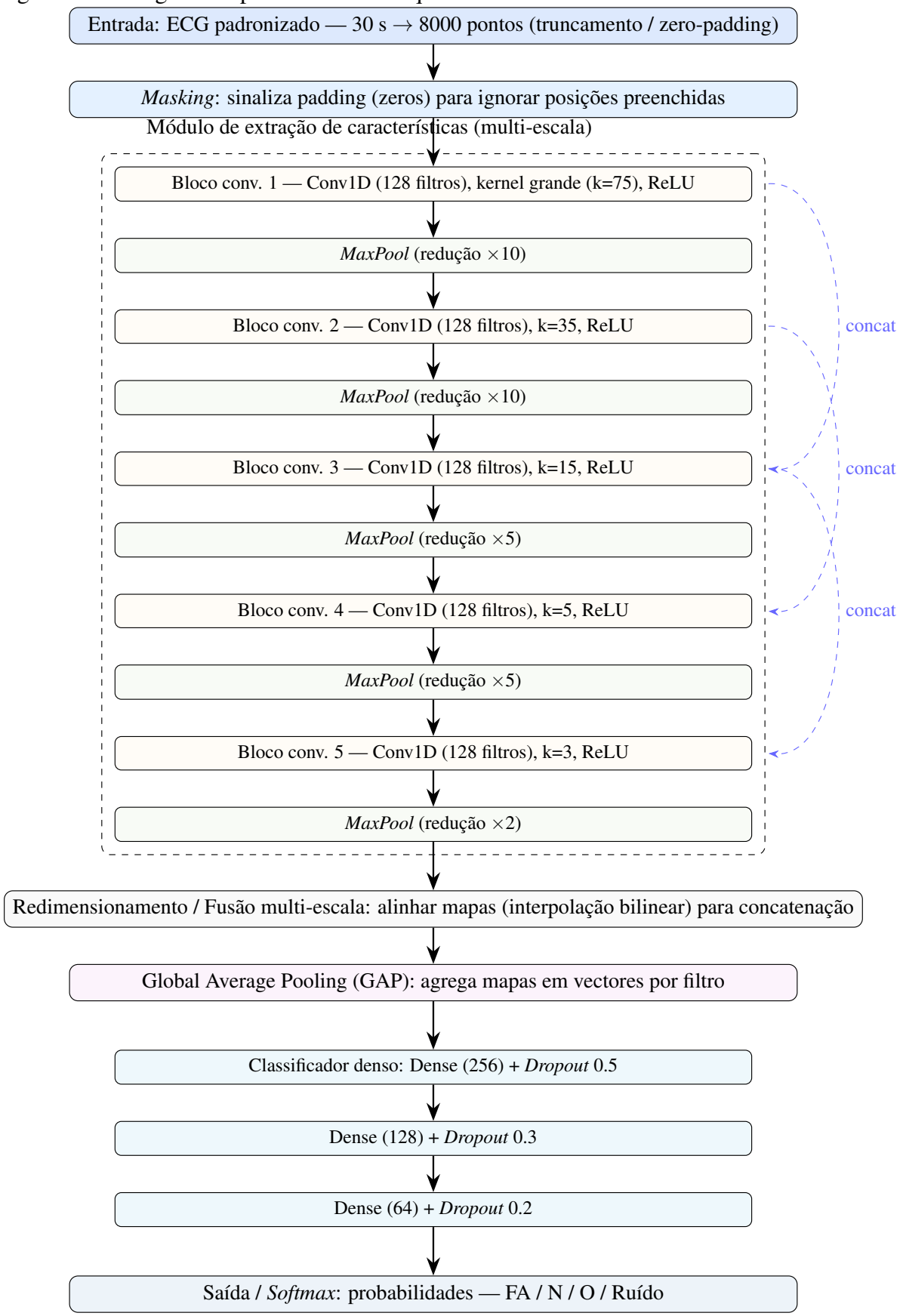
3.5.2 Desenvolvimento do aplicativo

O aplicativo móvel é desenvolvido em Kotlin e incorpora todo o pipeline de pré-processamento reimplementado em código nativo. As etapas executadas pelo aplicativo incluem:

- **Padronização:** zero-padding ou recorte para um comprimento fixo de 8.000 pontos;
- **Filtragem:** aplicação de filtros Butterworth equivalente ao comportamento do `filtfilt` (filtro de fase zero);
- **Normalização:** escalonamento Min–Max para o intervalo $[0, 1]$;
- **Mascaramento:** sinalização de regiões de padding para o interpretador.

A captura do ECG via *smartwatch* ocorre por conexão BLE e os dados são recebidos em uma corrotina separada da interface, garantindo que a UI permaneça responsiva. A inferência é realizada pela API do TensorFlow Lite (`Interpreter`), carregando o `model.tflite` e executando a predição sobre o vetor já pré-processado. Para reduzir latência e consumo de energia, a inferência é executada em uma corrotina ou thread de *background*, e os resultados (probabilidades por classe) são post-processados por limiares e regras de negócio antes de gerar notificações ao usuário. O aplicativo também inclui mecanismos de log e telemetria para coleta de métricas de latência e uso de recursos em campo, permitindo iterações futuras na otimização do modelo e do pipeline de processamento (TensorFlow Documentation, 2023; TensorFlow, 2024b).

Figura 10 – Diagrama representativo da arquitetura do modelo DMKS.



Fonte: Elaborado pelo autor.

Nota: **Conv1D** = convolução unidimensional; Pool = redução temporal; GAP = Global Average Pooling; concat = concatenação de mapas (fusões multi-escala); *Dropout* = regularização; *Masking* = ignorar padding.

4 RESULTADOS

Este capítulo apresenta os resultados obtidos em cada etapa descrita na Metodologia (Capítulo 3). Cada seção descreve o que foi medido, os resultados principais e indica os locais onde as figuras e tabelas devem ser inseridas para completar a apresentação dos achados.

4.1 Avaliação do Modelo

Ao comparar o desempenho do modelo proposto **DMKS** com o modelo de referência **K-B2S+** (FANG *et al.*, 2025), as métricas por classe revelam a eficácia da abordagem multi-escala. A Tabela 2 detalha os resultados para o modelo proposto, enquanto a Tabela 3 apresenta as métricas reportadas para o modelo de referência. Nas tabelas, as siglas seguem a codificação estabelecida na Tabela 1: além da Fibrilação Atrial (FA) e Ruído (R), utiliza-se **N** para Ritmo Sinusal Normal e **O** para Outros ritmos.

A consolidação dessa comparação é apresentada na Tabela 4, onde se observa uma superioridade consistente do DMKS, com ganhos expressivos na sensibilidade para a classe FA e no *F1-score* global. Complementarmente, a matriz de confusão (Figura 11) permite observar a distribuição dos acertos e erros de classificação. Observa-se uma excelente capacidade de discriminação da classe FA, com confusões residuais concentradas principalmente entre as classes Outros e Ruído, sugerindo similaridade morfológica em determinados segmentos.

Tabela 2 – Métricas por classe do modelo DMKS

Classe	**Precisão**	**Revocação**	**F1-Score**	**Suporte**
FA	0.90	0.89	0.89	691
N	0.98	0.94	0.96	4762
O	0.83	0.89	0.86	1536
Ruído	0.89	0.99	0.94	615

Fonte: Elaborado pelo autor.

Tabela 3 – Métricas por classe do modelo K-B2S+

Kernel	**Sen**	**Spe**	**F1-FA**	**F1-N**	**F1-O**	**F1-Global**
115	0.847	0.919	0.767	0.917	0.785	0.823
95	**0.861**	0.919	0.819	0.920	0.784	0.840
75	0.843	**0.921**	**0.842**	0.922	**0.799**	**0.854**
55	0.842	0.916	0.821	0.922	0.794	0.846
35	0.848	0.920	0.800	**0.925**	0.780	0.835

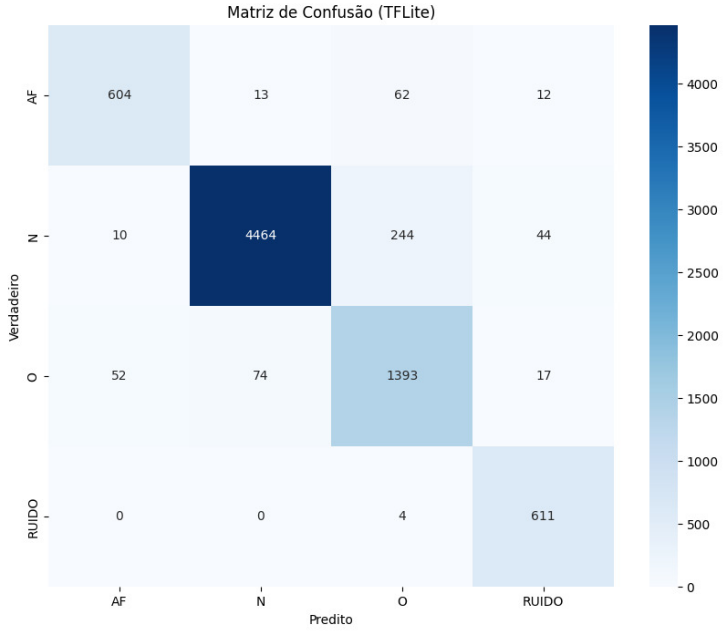
Fonte: (FANG *et al.*, 2025)

Tabela 4 – Comparação de desempenho: DMKS (proposto) vs K-B2S+ (referência)

Métrica / Classe	**DMKS**	**K-B2S+**	**Ganho (Δ)**
F1-score FA	**0.89**	0.842	+0.048
Revocação FA	**0.89**	0.843	+0.047
F1-score Normal	**0.96**	0.922	+0.038
F1-score Outros	**0.86**	0.799	+0.061
F1-score Global	**0.93**	**0.854**	**+0.076**

Fonte: Elaborado pelo autor com base em (FANG *et al.*, 2025).

Figura 11 – Matriz de confusão do modelo.



Fonte: Elaborado pelo autor.

A análise técnica das métricas obtidas após a quantização e conversão do modelo fundamenta a viabilidade da implantação proposta. A redução de 47,7% no espaço ocupado em memória (de 14,9 MB para 7,8 MB) é um fator determinante para a portabilidade, permitindo que o modelo seja executado em dispositivos com recursos limitados. Paralelamente, a otimização resultou em uma redução de 37,7% na latência de inferência (atingindo média de 23,43 ms), garantindo um processamento em tempo quase real que não compromete a responsividade da interface do usuário. O fato de que essa economia de recursos foi alcançada mantendo-se métricas de classificação elevadas (como o *F1-score* de 0,89 para a classe FA) valida a robustez do DMKS e justifica sua adoção em cenários de monitoramento contínuo via dispositivos vestíveis. Os detalhes dessa análise são apresentados na Tabela 5 e na Tabela 6, que mostram os ganhos de eficiência e a manutenção da acurácia pós-conversão.

4.2 Implantação em ambiente móvel

Para a implantação foi convertida a versão Keras/TensorFlow do modelo para o formato TensorFlow Lite (`.tflite`) com otimizações de quantização pós-treinamento (`int8`) para reduzir tamanho e acelerar inferência. As medições indicaram que a quantização reduziu substancialmente o tamanho do modelo e o tempo de inferência, mantendo acurácia comparável em relação à versão não quantizada. Conforme observado na Tabela 5, a versão convertida apresentou uma latência média de 23,43 ms para a execução do modelo, o que demonstra um excelente tempo de resposta. É importante ressaltar que essas métricas foram obtidas em ambiente de simulação (PC/Windows) e podem sofrer variações quando executadas em hardware móvel devido a diferenças de arquitetura e otimizações de hardware (GPU/DSP).

Assim, foi desenvolvido um aplicativo para Android no qual o modelo foi implantado. O aplicativo desenvolvido oferece uma interface intuitiva para o monitoramento e gerenciamento das predições. Na Figura 12, é possível observar o histórico de processamentos realizados, indicando a classe predita e a data da coleta. A listagem detalhada de todas as coletas de sinal de ECG armazenadas localmente é apresentada na Figura 13. Por fim, a Figura 14 ilustra a tela de resultado da classificação em tempo real. Nessa tela, os termos exibidos correspondem às classes do modelo: **Ritmo Sinusal** representa a classe Normal (N), enquanto **Outra** refere-se à classe Outros (O). O sistema exibe a categoria detectada acompanhada de sua respectiva probabilidade, fornecendo um *feedback* imediato ao usuário.

Tabela 5 – Comparação de desempenho: Modelo Original vs. Convertido

Atributo	**Modelo Original (Keras)**	**Modelo Convertido (TFLite)**	**Redução**
Total de Parâmetros	3.902.788	3.902.788	-
Tamanho do Arquivo	14,9 MB	7,8 MB	47,7%
Latência (PC/Windows)	37,60 ms	23,43 ms	37,7%

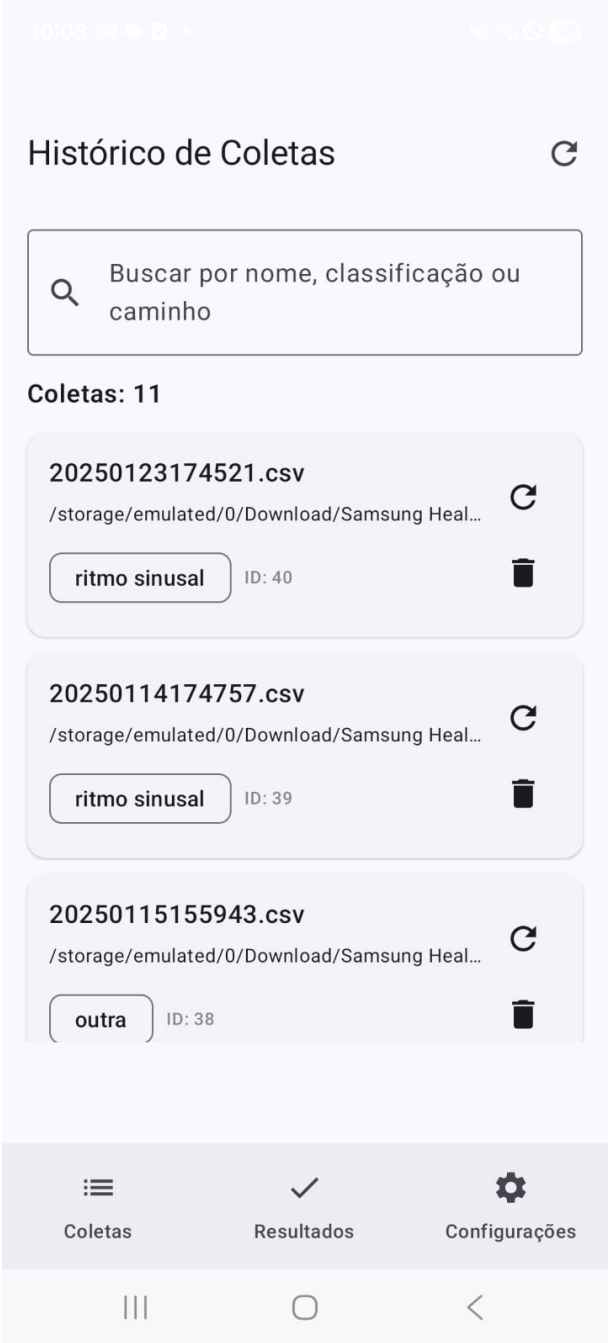
Fonte: Elaborado pelo autor.

Tabela 6 – Métricas por classe do modelo DMKS — pós-conversão (TFLite int8)

Classe	**Precisão**	**Revocação**	**F1-Score**	**Suporte**
FA	0.9069	0.8741	0.8902	691
N	0.9809	0.9374	0.9587	4762
O	0.8180	0.9069	0.8601	1536
Ruído	0.8933	0.9935	0.9407	615

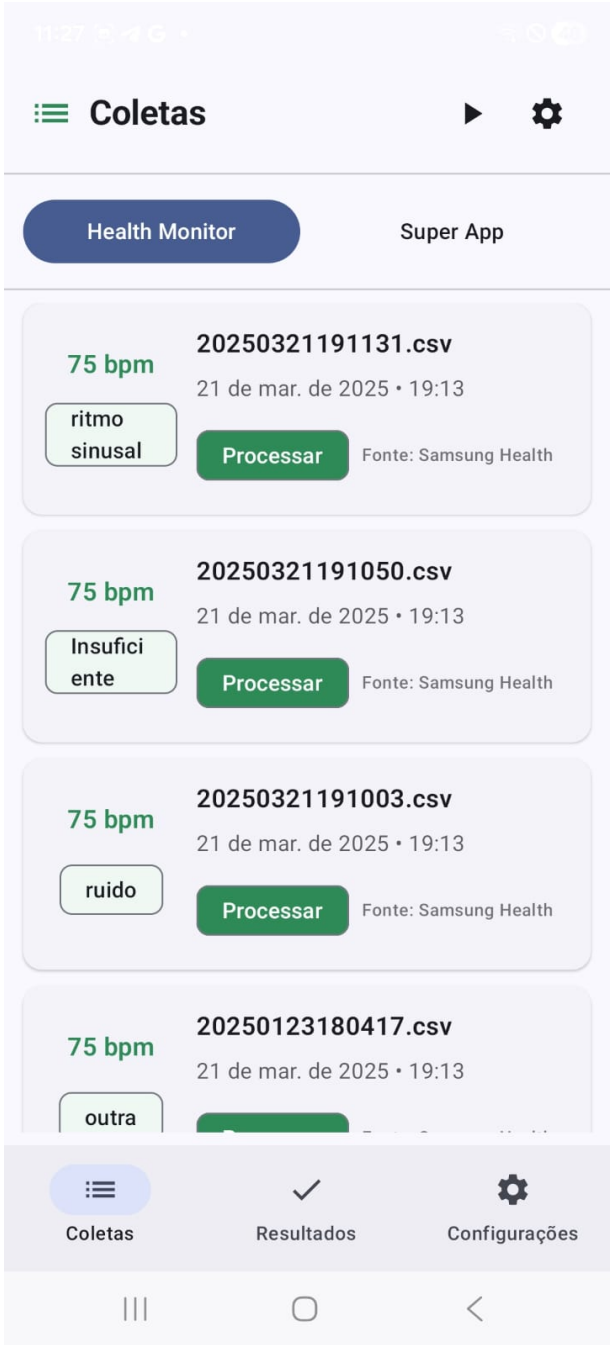
Fonte: Elaborado pelo autor.

Figura 12 – Tela de histórico de processamentos.



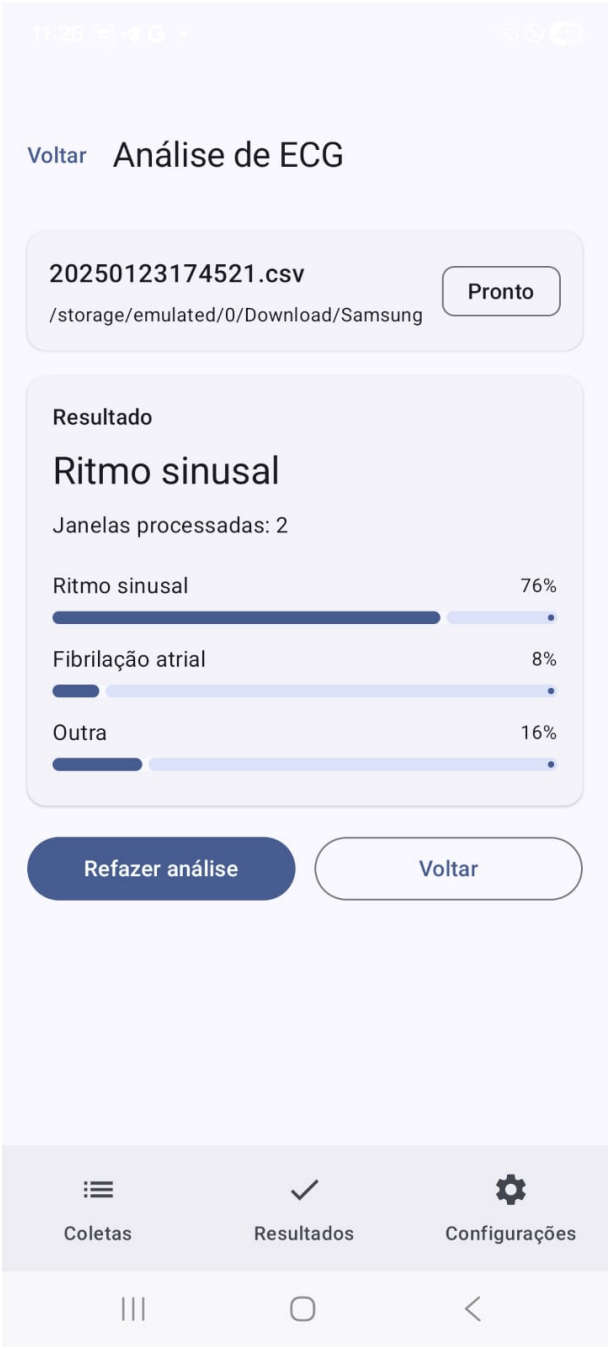
Fonte: Elaborado pelo autor.

Figura 13 – Tela de listagem de coletas.



Fonte: Elaborado pelo autor.

Figura 14 – Tela de classificação do modelo.



Fonte: Elaborado pelo autor.

5 DISCUSSÃO

Os resultados obtidos confirmam que a combinação de um pré-processamento rigoroso, estratégias de aumento de dados e uma arquitetura CNN otimizada (K-B2S+/DMKS) fornece desempenho elevado na detecção de fibrilação atrial (FA) a partir de janelas curtas de ECG (30 s). A alta precisão e revocação observadas para FA indicam que o modelo captura bem os sinais patognomônicos relevantes mesmo na presença de ruídos simulados, o que corrobora trabalhos anteriores que ressaltam a importância de kernels multi-escala e *augmentation* para sinais vestíveis (XIONG *et al.*, 2022; FANG *et al.*, 2025; ALDUGHAYFIQ *et al.*, 2023). Cumpre notar que, embora o modelo DMKS tenha apresentado métricas superiores ao K-B2S+, ambos não foram treinados sob a mesma base de dados consolidada, de modo que, em trabalhos futuros, é fundamental submetê-los ao mesmo conjunto de dados para verificar a eficácia relativa de forma rigorosa. A inclusão de ruídos sintéticos controlados (*stitching*, *baseline wander*, EMG e 60 Hz) ampliou a variabilidade do conjunto e contribuiu para uma maior robustez do classificador contra artefatos realistas, reduzindo em parte o viés para classes majoritárias após balanceamento.

Nem todas as classes, contudo, alcançam desempenho igualmente elevado. As métricas inferiores nas classes *Fibrilação Atrial* e *Outros* apontam limitações relevantes: a similaridade morfológica entre determinadas arritmias e a FA ainda confunde o classificador, o que reduz a sensibilidade e a precisão para essas categorias. Isso indica que, embora o *augmentation* sintético tenha sido altamente eficaz para conferir robustez à classe de *Ruído*, é necessário incorporar mais amostras reais e diversificadas de ritmos não-sinusais para treinar representações que generalizem melhor às condições de campo e permitam uma distinção mais clara entre a FA e outras arritmias clinicamente semelhantes (CLIFFORD *et al.*, 2017; ZANDE *et al.*, 2023).

A implantação do modelo no ecossistema Android, utilizando a conversão para TensorFlow Lite e técnicas de quantização pós-treinamento, apresentou resultados altamente satisfatórios. Conforme detalhado nos resultados, a latência média de inferência do modelo foi reduzida para 23,43 ms, o que demonstra a viabilidade técnica do DMKS para inferência em tempo quase real em dispositivos móveis. Embora a latência total inclua o custo do pré-processamento (filtragem digital, normalização e *masking*), os tempos obtidos indicam que o *pipeline* proposto é eficiente o suficiente para monitoramento contínuo sem comprometer a responsividade do dispositivo (TensorFlow, 2024b; TensorFlow Documentation, 2023). Assim, a arquitetura mostra-se robusta não apenas em termos de acurácia clínica, mas também em

desempenho computacional para ambientes com recursos limitados.

Um apontamento crítico levantado pelo trabalho é a escassez de bases públicas representativas de sinais obtidos especificamente por *smartwatches*. A maioria das bases disponíveis (mesmo as utilizadas aqui) provém de dispositivos portáteis ou de registros clínicos com características de aquisição distintas das de *wearables* comerciais; isso limita a generalização e impede avaliações realistas de campo. Há, portanto, necessidade urgente de criar e compartilhar *datasets* anotados de ECG por *smartwatch* com metadados ricos (posição do relógio, atividade do usuário, qualidade do contato, temperatura, *firmware*), pois tais informações são fundamentais para modelar e mitigar fontes reais de variabilidade observadas em ambiente ambulatorial (GOLDBERGER *et al.*, 2000a; ZANDE *et al.*, 2023; SILVA *et al.*, 2011).

Os filtros digitais mostraram-se essenciais para melhorar a relação sinal-ruído e evidenciar morfologias relevantes (picos R, morfologia do QRS, ausência de ondas P regulares). Contudo, filtros não resolvem todos os problemas: quando a captação é severamente comprometida (contato ruim, movimento extremo, influência térmica do sensor), a informação clínica pode simplesmente não existir no registro e o processamento não consegue recuperar o que não foi originalmente aquisitado (CLIFFORD *et al.*, 2017; ZANDE *et al.*, 2023; Samsung Support, 2023).

As limitações de *hardware* dos *smartwatches* emergem, portanto, como uma barreira prática: eletrodos pequenos, variação térmica, deslocamento no pulso, suor, interferência de outras aplicações e contaminação por movimento reduzem consistentemente a qualidade da captação. Mesmo os melhores *pipelines* de software têm alcance limitado quando a qualidade física do sinal é baixa. Isso sugere que avanços paralelos em projeto de sensores (eletrodos com melhor acondicionamento, aquisição multi-ponto, algoritmos de compensação de movimento embarcados) e em boas práticas de uso (instruções ao usuário para melhorar contato) são tão importantes quanto as melhorias algorítmicas para elevar a sensibilidade e especificidade alcançáveis em campo (Samsung Support, 2023; FDA, 2020; ZANDE *et al.*, 2023).

Com base nesses achados, recomendam-se alguns caminhos para pesquisas e desenvolvimentos futuros. Primeiramente, fomentar a criação colaborativa de bases anotadas específicas para *smartwatches*, incluindo metadados de aquisição; em segundo lugar, explorar abordagens multimodais (ECG + PPG + acelerômetro) para mitigar artefatos de movimento; e por fim, investigar avanços no desenvolvimento de sensores e técnicas de aquisição, como o uso de novos materiais para eletrodos e sistemas de captura multi-ponto, visando mitigar a

interferência de ruídos e artefatos diretamente na origem do sinal. Além disso, qualquer sistema que forneça alertas clínicos precisa de validação prospectiva com supervisão médica e avaliação regulatória antes do uso em larga escala.

6 CONCLUSÃO

Este trabalho apresentou o desenvolvimento e a avaliação de um sistema automatizado para detecção de fibrilação atrial a partir de sinais de eletrocardiograma de derivação única, com foco em aplicações vestíveis e em ambiente móvel.

A metodologia proposta integrou dados públicos das bases PhysioNet/Computing in Cardiology Challenge 2017, PhysioNet/Computing in Cardiology Challenge 2011, Noise Stress Test Database (NSTDB) e Motion Artifact Contaminated ECG Database (MACECGDB), assegurando a robustez estatística do treinamento e a diversidade na representação de artefatos. O pipeline de pré-processamento, composto por padronização do comprimento dos sinais, filtragem Butterworth passa-alta e passa-baixa com filtragem zero-phase e normalização Min–Max, mostrou-se eficaz na redução de ruídos e na padronização das entradas para os modelos.

Para mitigar o desbalanceamento entre classes e aumentar a robustez frente a artefatos comuns em sinais vestíveis, foram aplicadas técnicas de aumento de dados baseadas na geração de ruídos realistas, incluindo costura de sinais, flutuação de linha de base, ruído muscular e interferência de rede elétrica. Essas estratégias contribuíram para melhorar a capacidade de generalização do classificador, especialmente na detecção da classe de fibrilação atrial.

A arquitetura K-B2S+, devido ao seu excelente compromisso entre desempenho e complexidade computacional, serviu como base para o desenvolvimento do modelo DMKS proposto neste trabalho. Os resultados consolidados do DMKS no conjunto de teste demonstraram elevada precisão, revocação e F1-score para a classe de fibrilação atrial, consolidando sua eficácia para o diagnóstico automatizado. Embora a análise da matriz de confusão tenha revelado que os erros residuais de classificação se concentram nas classes Outros e Ruído, o desempenho geral do DMKS valida a integração de kernels multi-escala com a estratégia de aumento de dados sintéticos para a mitigação de artefatos.

A conversão do modelo para o formato TensorFlow Lite e sua integração em um aplicativo móvel desenvolvido em Kotlin demonstraram a viabilidade da implantação do sistema em dispositivos Android. A inferência local, aliada à execução do pré-processamento no próprio aplicativo, apresentou latência compatível com aplicações em tempo quase real, reforçando o potencial de uso do sistema como ferramenta de apoio à triagem e monitoramento contínuo de arritmias em ambiente ambulatorial.

Como trabalhos futuros, destacam-se a ampliação da base de dados reais coletadas por *smartwatches* incluindo metadados contextuais de aquisição, a exploração de abordagens

multimodais integrando sinais de acelerômetro e fotopletismografia (PPG) para melhoria da robustez contra artefatos de movimento, além da realização de estudos de validação clínica prospectiva sob supervisão médica. Em síntese, os resultados obtidos demonstram que a abordagem proposta é tecnicamente viável e promissora para aplicações de detecção automática de fibrilação atrial em sistemas móveis e vestíveis, contribuindo para o avanço de soluções de saúde digital baseadas em inteligência artificial.

REFERÊNCIAS

ABADI, M.; AGARWAL, A.; BARHAM, P.; BREVDO, E.; CHEN, Z.; CITRO, C.; CORRADO, G. S.; DAVIS, A.; DEAN, J.; DEVIN, M.; GHEMAWAT, S.; GOODFELLOW, I.; HARP, A.; IRVING, G.; ISARD, M.; JIA, Y.; JOZEFOWICZ, R.; KAISER, L.; KUDLUR, M.; LEVENBERG, J.; MANE, D.; MONGA, R.; MOORE, S.; MURRAY, D.; OLAH, C.; SCHUSTER, M.; SHLENS, J.; STEINER, B.; SUTSKEVER, I.; TALWAR, K.; TUCKER, P.; VANHOUCKE, V.; VASUDEVAN, V.; VIEGAS, F.; VINYALS, O.; WARDEN, P.; WATTENBERG, M.; WICKE, M.; YU, Y.; ZHENG, X. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. **arXiv preprint arXiv:1603.04467**, 2016. Disponível em: <https://arxiv.org/abs/1603.04467>.

ALDUGHAYFIQ, B.; ASHFAQ, F.; JHANJHI, N. Z.; HUMAYUN, M. A deep learning approach for atrial fibrillation classification using multi-feature time series data from ECG and PPG. **Diagnostics**, v. 13, n. 2442, 2023.

ANDERSEN, R. S. A deep learning approach for real-time detection of atrial fibrillation. **Expert Systems with Applications**, v. 115, p. 465–473, 2019.

Android Developers. **NNAPI Migration Guide**. 2025. Acesso em: 25 jul. 2025. Disponível em: <https://developer.android.com/ndk/guides/neuralnetworks/migration-guide>.

Android Developers / TensorFlow. **Build a handwritten digit classifier app with TensorFlow Lite**. 2024. Acesso em: 25 jul. 2025. Disponível em: <https://developer.android.com/codelabs/digit-classifier-tflite>.

BEHRAVAN, V. *et al.* **Motion Artifact Contaminated ECG Database (MacECGDB)**. 2015. <https://physionet.org/content/macecgdb/1.0.0/>. Versão 1.0.0. Acesso em: 04 jan. 2026.

CHOLLET, F. **Deep Learning with Python**. 1st. ed. Shelter Island, NY: Manning Publications Co., 2017. ISBN 9781617294433.

CLIFFORD, G. D.; LIU, C.; MOODY, B.; LAW, L. H.; SILVA, I.; LI, Q.; JOHNSON ALLAN E. W.; MARK, R. G. AF classification from a short single-lead ECG recording: The physionet/computing in cardiology challenge 2017. In: **Proceedings of the 2017 Computing in Cardiology (CinC)**. IEEE, 2017. p. 1–4. Disponível em: <https://doi.org/10.22489/CinC.2017.065-469>.

FANG, B.; YU, Z.; ZHANG, L.; TENG, Y.; CHEN, J. K-b2s+: A one-dimensional cnn model for af detection with short single-lead ecg waves from wearable devices. **Digital Communications and Networks**, v. 11, p. 613–621, 2025.

FDA. **K201168: Samsung ECG Monitor App – FDA Summary**. 2020. Acesso em: 04 jul. 2025. Disponível em: <https://www.accessdata.fda.gov/cdrh_docs/pdf20/K201168.pdf>.

GÉRON, A. **Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow**. 2. ed. Beijing / Sebastopol: O'Reilly Media, 2019. ISBN 978-1492032649.

Get Smart About AFib. **Electrocardiogram illustration showing atrial fibrillation pattern**. 2022. Acesso em: 25 jul. 2025. Disponível em: <https://getsmartaboutafib.net/pt-br/publico-geral/tenho-fibrilacao-atrial/que-e-fibrilacao-atrial>.

GOLDBERGER, A. L.; AMARAL, L. A. N.; GLASS, L.; HAUSDORFF, J. M.; IVANOV, P. C.; MARK, R. G.; PATEL, V.; STANLEY, H. E. **PhysioBank, PhysioToolkit and PhysioNet: Components of a New Research Resource for Complex Physiologic Signals**. 2000. <https://physionet.org/>. Circulation [Online] 101(23):e215–e220. RRID:SCR_007345. Acesso em: 25 jul. 2025.

GOLDBERGER, A. L.; AMARAL, L. A. N.; GLASS, L.; HAUSDORFF, J. M.; IVANOV, P. C.; MARK, R. G.; PATEL, V.; STANLEY, H. E. Physiobank, physiotoolkit and physionet: Components of a new research resource for complex physiologic signals. **Circulation**, v. 101, n. 23, p. e215–e220, 2000.

Google AI Edge. **LiteRT overview**. 2024. Acesso em: 25 jul. 2025. Disponível em: <https://ai.google.dev/edge/litert>.

Google AI Edge. **TensorFlow Lite Task Library**. 2024. Acesso em: 25 jul. 2025. Disponível em: <https://ai.google.dev/edge/litert/libraries/task_library/overview>.

HE, Z.; CHEN, Y.; ZHANG, D.; YIN, W.; KARIMI, H. R. A new intelligent ECG recognition approach based on CNN and improved ALO-SVM. **Signal, Image and Video Processing**, v. 16, p. 2073–2081, 2022.

JEMEROV, D.; ISAKOVA, S. **Kotlin in Action**. Shelter Island, NY: Manning Publications, 2017. 2ª ed. recomendada para desenvolvimento Android com Kotlin.

LUO, K.; LI, J.; BIN, J.; QIU, S. G.; CAI, W.; SUN, C. A deep learning approach for atrial fibrillation detection from single-lead short ECG recordings. In: **Proceedings of the 2017 Computing in Cardiology (CinC)**. Rennes, France: IEEE, 2017. p. 1–4.

MATTOS, L. **Coração | Complexo estimulante – Eletrocardiograma**. 2026. Acesso em: 30 jan. 2026. Disponível em: <https://anatomia-papel-e-caneta.com/coracao-complexo-estimulante-eletrocardiograma/>.

MOODY, G. B.; MARK, R. G. A new method for detecting atrial fibrillation using r-r intervals. **Computers in Cardiology**, v. 10, p. 227–230, 1983.

MOODY, G. B.; MARK, R. G. The impact of the mit-bih arrhythmia database. **IEEE Engineering in Medicine and Biology Magazine**, v. 20, n. 3, p. 45–50, 2001.

MOODY, G. B.; MULDROW, W. E.; MARK, R. G. A noise stress test for arrhythmia detectors. **Computers in Cardiology**, v. 11, p. 381–384, 1984.

MOODY, G. B.; MULDROW, W. E.; MARK, R. G. **MIT-BIH Noise Stress Test Database (NSTDB)**. 1999. <https://physionet.org/content/nstdb/1.0.0/>. Versão 1.0.0. Acesso em: 04 jan. 2026.

MORONEY, L. **Using TensorFlow Lite on Android**. 2018. Acesso em: 30 jan. 2026. Disponível em: <https://blog.tensorflow.org/2018/03/using-tensorflow-lite-on-android.html>.

Samsung. **Monitor your heart rate and blood pressure with the Galaxy Watch series**. 2024. Acesso em: 04 jul. 2025. Disponível em: <https://www.samsung.com/uk/support/mobile-devices/measure-your-ecg-with-the-galaxy-watch-series/>.

Samsung Support. **Meça o seu ECG com a série Galaxy Watch**. 2023. <https://www.samsung.com/africa_pt/support/mobile-devices/meca-o-seu-ecg-com-a-serie-galaxy-watch/>. Acesso em: 25 jul. 2025.

SILVA, I.; MOODY, G. B.; CELI, L. A. **Improving the Quality of ECGs Collected using Mobile Phones: The PhysioNet/Computing in Cardiology Challenge 2011**. 2011. <https://physionet.org/content/challenge-2011/1.0.0/>. Versão 1.0.0. Acesso em: 04 jan. 2026.

TensorFlow. **Convert a TensorFlow model to TensorFlow Lite**. 2024. Acesso em: 25 jul. 2025. Disponível em: <https://www.tensorflow.org/lite/convert>.

TensorFlow. **TensorFlow Lite Guide**. 2024. Acesso em: 25 jul. 2025. Disponível em: <https://www.tensorflow.org/lite/guide>.

TensorFlow Development Team. **TensorFlow**. 2015. <https://www.tensorflow.org/>. Acesso em: 25 jul. 2025.

TensorFlow Documentation. **TensorFlow Guide — Keras and Core**. 2023. <https://www.tensorflow.org/guide>. Acesso em: 25 jul. 2025.

TensorFlow Model Optimization. **Post-training quantization — Model Optimization**. 2022. Acesso em: 25 jul. 2025. Disponível em: <https://www.tensorflow.org/model_optimization/guide/quantization/post_training>.

UFABC Divulga Ciência. **Entendendo o funcionamento do coração humano – Parte 2: o sistema elétrico**. 2023. Acesso em: 27 jul. 2025. Disponível em: <https://ufabcdivulgaciencia.proec.ufabc.edu.br/2023/11/09/entendendoo-funcionamento-2/>.

XIONG, Z.; STILES, M. K.; GILLIS, A. M.; ZHAO, J. Enhancing the detection of atrial fibrillation from wearable sensors with neural style transfer and convolutional recurrent networks. **Computers in Biology and Medicine**, v. 146, p. 105551, 2022.

ZANDE, J. van der; STRIK, M.; DUBOIS, R.; PLOUX, S.; ALRUB, S. A.; CAILLOL, T.; NASARRE, M.; DONKER, D. W.; OPPERSMA, E.; BORDACHAR, P. Using a smartwatch to record precordial electrocardiograms: A validation study. **Sensors**, v. 23, n. 5, p. 2555, 2023.

ZIHLMANN, M.; PEREKRESTENKO, D.; TSCHANNEN, M. Convolutional recurrent neural networks for electrocardiogram classification. In: **Proceedings of the 2017 Computing in Cardiology (CinC)**. Rennes, France: IEEE, 2017. p. 1–4.