# Capítulo 2

# Tiny ML: Introdução ao Aprendizado de Máquina em Sistemas Embarcados

Ramon Santos Nepomuceno, Luís Fagner de Carvalho da Silva, Dorgival Pereira da Silva Netto, Rafael Perazzo Barbosa Mota Carlos, Carlos Vinicius Gomes Costa Lima e Carlos Julian Menezes Araujo

***Abstract***

*This chapter provides a practical introduction to Machine Learning in Embedded Systems, with a focus on the Edge Impulse platform. It covers fundamental concepts and techniques for developing machine learning models for embedded devices, including practical applications using smartphones and TinyML kits. The practical sessions address data collection, model training, and deployment on real devices, showcasing the technology's applicability. Additionally, the chapter explores theoretical aspects of Tiny Machine Learning, such as Impulse configuration, spectral feature extraction, and classifier training for resource-constrained systems. By the end, readers will be equipped to implement and test machine learning models on microcontrollers and low-power devices, leveraging Edge Impulse to effectively bridge theory and practice.*

***Resumo***

*Este capítulo oferece uma introdução prática ao Aprendizado de Máquina em Sistemas Embarcados com ênfase na plataforma Edge Impulse. O texto explora conceitos fundamentais e técnicas para o desenvolvimento de modelos de aprendizado de máquina em dispositivos embarcados, incluindo o uso de smartphones e kits TinyML. A abordagem prática inclui a coleta de dados, treinamento de modelos e implantação em dispositivos reais, com exemplos que demonstram a aplicabilidade da tecnologia. Além disso, o capítulo aborda aspectos teóricos do Tiny Machine Learning, como a configuração do Impulse, extração de features espectrais e treinamento de classificadores para sistemas com recursos limitados. Ao final, os leitores estarão aptos a implementar e testar modelos de aprendizado de máquina em microcontroladores e dispositivos de baixo consumo de energia, utilizando o Edge Impulse para integrar teoria e prática de forma eficaz.*

## 2.1. Introdução

A crescente interconexão entre dispositivos e sistemas está revolucionando setores como saúde e transporte, onde a capacidade de processar e analisar dados de forma autônoma se tornou essencial. Neste contexto, a computação na borda, ou edge computing, se destaca como uma solução para atender à crescente demanda por análise em tempo real, reduzindo a dependência de conexões de rede instáveis. Este paradigma permite o processamento local de dados nos dispositivos, diminuindo a latência e ampliando as capacidades da Inteligência Artificial (IA) [3].

A aplicação de Inteligência Artificial (AI) em sistemas embarcados, ou edge AI, viabiliza uma ampla gama de aplicações, desde diagnósticos médicos imediatos até a otimização de processos industriais. O processamento local possibilita a realização de inferências e a tomada de decisões em tempo real, melhorando a eficiência, a velocidade e a segurança dos sistemas. Adicionalmente, processar dados localmente contribui para a preservação da privacidade e da segurança, pois informações sensíveis não são transmitidas a servidores externos [18].

O conceito de Tiny Machine Learning (Tiny ML) refere-se à execução de modelos de aprendizado de máquina em dispositivos com recursos limitados, como microcontroladores e sensores . Tiny ML viabiliza a implementação de funcionalidades de IA em hardware com baixo consumo de energia e pouca memória, expandindo significativamente as capacidades da computação na borda [13].

Este capítulo apresenta o desenvolvimento de projetos de IA em sistemas embarcados, introduzindo os fundamentos do aprendizado de máquina e explorando a computação na borda. A plataforma Edge Impulse é utilizada para fornecer um guia prático para a criação e implantação de modelos de aprendizado de máquina em dispositivos reais [6]. Este processo visa demonstrar o potencial da Inteligência Artificial nesse cenário, esclarecendo as vantagens e os desafios dessa área emergente.

## 2.2. Projetos Práticos com a Plataforma Edge Impulse

A Edge Impulse é uma plataforma projetada para facilitar o desenvolvimento e a implementação de modelos de aprendizado de máquina em dispositivos embarcados [6]. Ela oferece ferramentas que permitem que desenvolvedores e engenheiros criem soluções rapidamente, integrando coleta de dados, processamento de sinais e aprendizado de máquina em um único ambiente.

A plataforma possibilita projetos que utilizam diversos sensores, desde projetos mais simples como os que envolvem gestos ou medição de temperatura, até mais complexos, como projetos com áudio e visão computacional. O restante do capítulo será focado no reconhecimento de movimentos usando a plataforma até a implementação em dispositivos mobile e no kit TinyML.

### 2.2.1. Projeto de Reconhecimento de Movimento

Nas seções seguintes, será apresentado um projeto de reconhecimento contínuo de movimentos (Continuous Motion Recognition). O objetivo é identificar e classificar diferentes tipos de movimentos em tempo real. Utilizando sensores como acelerômetros ou

giroscópios para capturar dados de movimento, o sistema detectará padrões específicos e os classificará em categorias predefinidas. Durante o desenvolvimento deste projeto, os principais conceitos relacionados à inteligência artificial em dispositivos embarcados (Edge AI) serão abordados de maneira prática e didática.

#### 2.2.1.1. Criando uma Conta e Iniciando um Projeto

Para começar, acesse `https://www.edgeimpulse.com` e crie uma conta. Uma vez logado, você pode iniciar um novo projeto clicando em *Create New Project*. Como o projeto envolve a detecção de movimento, sugere-se escolher um nome que reflita essa finalidade, como *MotionTrack*. Isso não só ajudará a identificar facilmente o projeto mais tarde, mas também dará um contexto claro sobre seu objetivo. Após nomear o projeto, o *dashboard* será aberto, permitindo gerenciar todas as fases do projeto, desde a coleta de dados até o treinamento e implantação do modelo.

A Figura 2.1 mostra a tela inicial de um projeto na plataforma *Edge Impulse*. O *dashboard* oferece uma visão das funcionalidades e ferramentas disponíveis, integrando os passos necessários para a criação e implantação dos modelos.

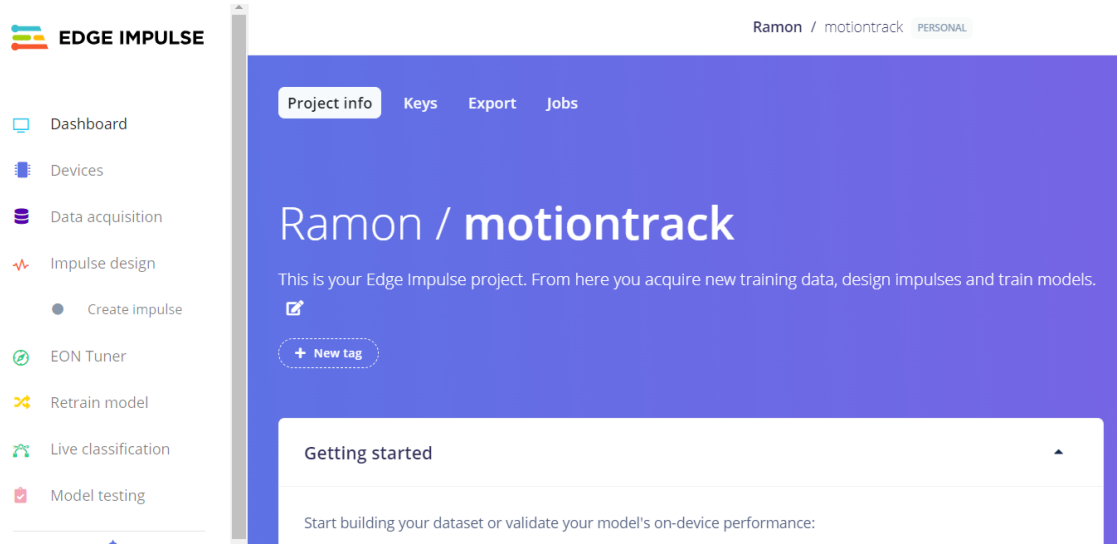


**Figura 2.1. Dashboard da plataforma Edge Impulse.**

#### 2.2.1.2. Conectando Dispositivos e Coletando Dados

Após criar o projeto, clique em *Devices* e selecione *Connect a New Device* para escolher o dispositivo a ser utilizado. Nesta etapa inicial, será utilizado um smartphone como dispositivo alvo. Conecte-o escaneando o código QR exibido na tela, como mostra a Figura 2.2.

A criação da base de dados é um dos primeiros passos no desenvolvimento dos modelos, pois fornece os exemplos necessários para que o modelo aprenda a identificar padrões e tomar decisões. Em termos simples, um modelo de aprendizado de máquina

analisa dados de entrada, reconhece padrões e faz previsões ou decisões com base nesses padrões. Para funcionar corretamente, o modelo deve ser treinado com um conjunto de dados que represente as variáveis e condições que ele encontrará na aplicação real. Durante o treinamento, o modelo ajusta seus parâmetros internos para minimizar o erro entre suas previsões e os resultados reais, melhorando sua precisão ao longo do tempo.

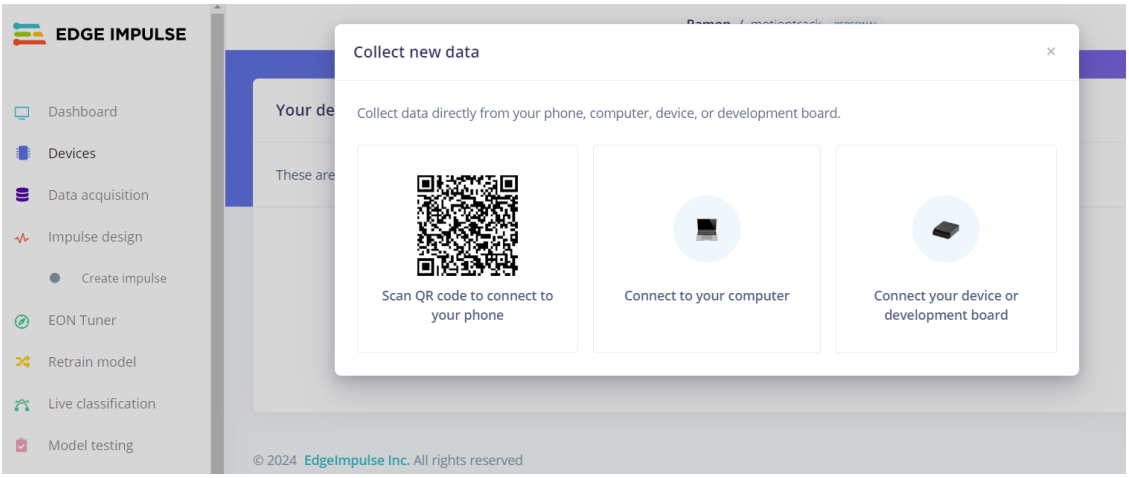


**Figura 2.2. Adicionando dispositivo no Edge Impulse.**

Após conectar o dispositivo, vá para *Data Acquisition*. No campo de seleção do tipo de sensor, escolha *Acelerômetro*. No campo *Label*, insira o tipo de movimento que deseja registrar; comece com *left_right*. Clique em *Start Sampling* e mova o celular da esquerda para a direita por 10 segundos. Repita esse processo 20 vezes para coletar as amostras necessárias.

Depois de completar as coletas para *left_right*, repita o procedimento para os movimentos *up_down* (para cima e para baixo), *circle* (em círculos) e *idle* (com o celular em repouso). Esse processo resultará em um total de 80 amostras, com 20 para cada classe de movimento. A Figura 2.3 exemplifica a tela de coleta de dados.

Após a aquisição das amostras, é necessário dividir os dados em conjuntos de treinamento e teste. O conjunto de treinamento é utilizado para ajustar os parâmetros do modelo, enquanto o conjunto de teste é empregado para avaliar a capacidade do modelo de generalizar para novos dados. Esse processo permite verificar a eficácia do modelo em identificar e classificar amostras não vistas anteriormente e detectar padrões que podem representar maior dificuldade para a inferência correta. Recomenda-se uma divisão de 80% para o conjunto de treinamento e 20% para o conjunto de teste. O Edge Impulse oferece ferramentas automáticas para realizar essa divisão, mas também é possível fazê-la manualmente, clicando nos três pontos de cada amostra e movendo-a para o conjunto de teste.

#### 2.2.1.3. Projetando o Impulso

Após a coleta de dados , é possível criar o Impulse clicando em *Impulse Design*. Um Impulse completo consiste em um pipeline de aprendizado de máquina composto por

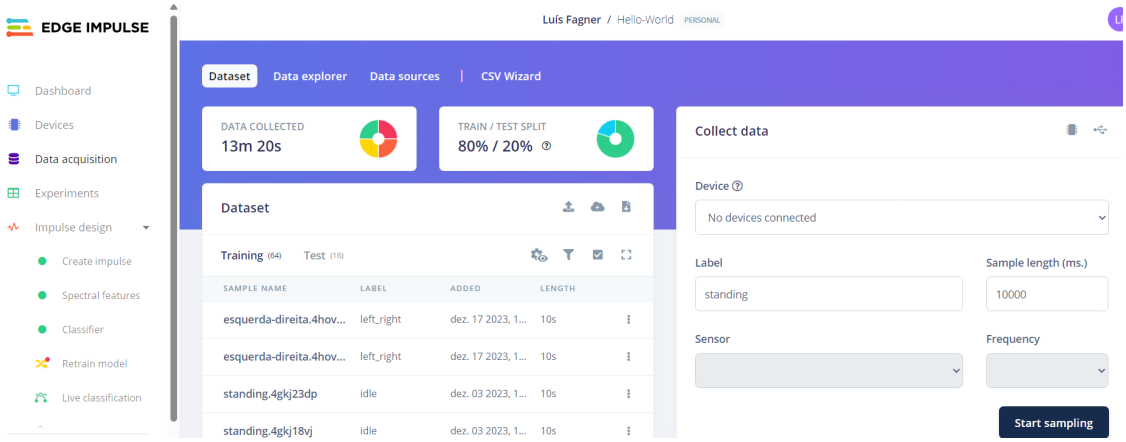


**Figura 2.3. Coletando dados na plataforma Edge Impulse.**

três principais componentes: **Bloco de Entrada**, **Bloco de Processamento** e **Bloco de Aprendizado**, conforme exemplificado na Figura 2.4.

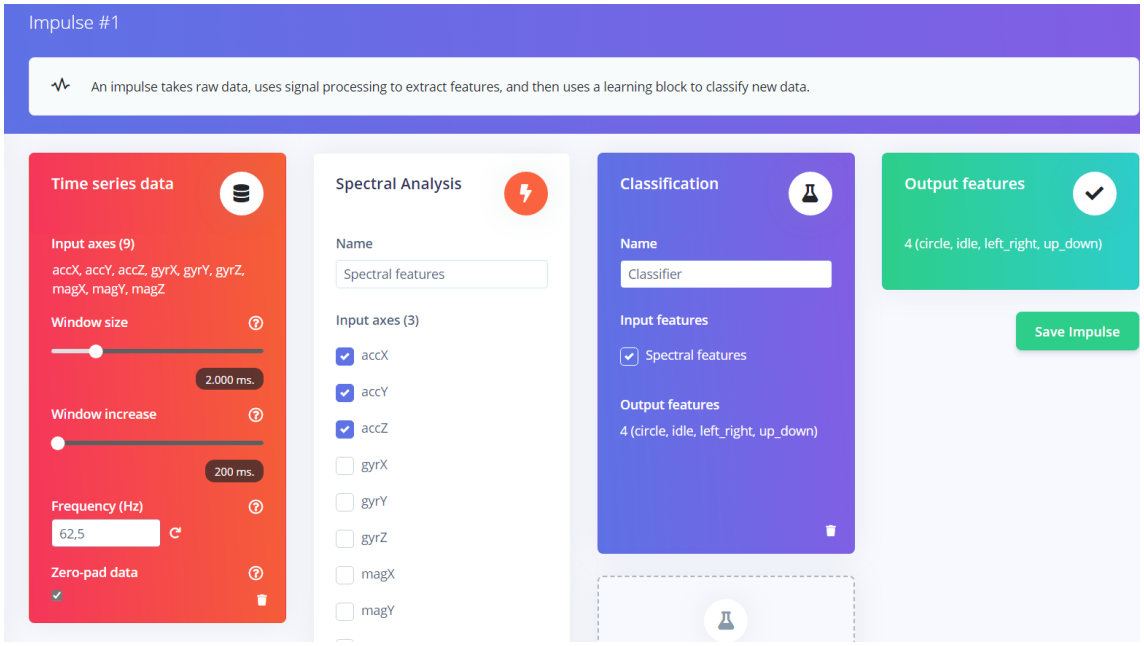


**Figura 2.4. Projeto do Impulse.**

- **Bloco de Entrada (Input Block)**: Define como os dados são coletados e estruturados para o treinamento do modelo. Este bloco especifica o tipo de dados que será utilizado, como séries temporais (áudio, vibração, movimentos) ou imagens. No caso de séries temporais, o bloco indica quais eixos são referenciados a partir do conjunto de dados de treinamento, além de configurar o tamanho da janela, que determina o tamanho das características brutas usadas para o treinamento. O aumento da janela é utilizado para criar artificialmente mais características e fornecer mais informações ao bloco de aprendizado. A frequência dos dados é calculada automaticamente com base nas amostras de treinamento, embora possa ser ajustada, desde

que não seja menor que 0,000016 (menos de 1 amostra a cada 60 segundos). Além disso, o bloco pode adicionar valores zero quando características brutas estão ausentes, garantindo que o modelo receba um fluxo contínuo de dados. Para o projeto em questão, deixe as configurações padrões.

- **Bloco de Processamento (Processing Block)**: Realiza o pré-processamento dos dados para torná-los adequados para a etapa de aprendizado. Este bloco atua como um extrator de características, utilizando operações de Processamento de Sinais Digitais (DSP) para extrair informações relevantes dos dados brutos. Clique em *Add a Processing Block*, e selecione *Spectral Analysis*. Este bloco extrai informações sobre frequência, potência e outras características de um sinal. Filtros passa-baixa e passa-alta podem ser aplicados para remover frequências indesejadas, sendo especialmente úteis para examinar padrões repetitivos como movimentos ou vibrações captados pelo acelerômetro. A escolha do bloco de processamento recomendado é facilitada pela interface do Edge Impulse, que utiliza uma estrela para indicar a opção mais adequada com base nos dados de entrada.

- **Bloco de Aprendizado (Learning Block)**: Treina o modelo de aprendizado de máquina utilizando os dados processados. Este bloco é essencial para ajustar os parâmetros do modelo e permitir que ele faça previsões precisas. Após adicionar o bloco de processamento, o próximo passo é adicionar um bloco de aprendizado. Clique em *Add Learning Block* e selecione *Classification* para adicionar um classificador de rede neural. O bloco de aprendizado pode variar conforme o objetivo do modelo e o tipo de dados no conjunto de treinamento, incluindo algoritmos para classificação, regressão, detecção de anomalias, transferência de aprendizado em imagens, reconhecimento de palavras-chave ou detecção de objetos. Após selecionar o tipo de aprendizado, clique em *Save Impulse* para criar o impulso. O bloco de aprendizado selecionado será configurado para aprender com os dados extraídos no bloco de processamento, permitindo a criação de um modelo que atenda às necessidades específicas do projeto.

#### 2.2.1.4. Gerando *Features Espectrais*

Após configurar o bloco de aprendizado, o próximo passo no projeto de detecção de movimento é gerar as *features espectrais*. Isso é feito clicando em *Spectral Features*, seguido de *Save Parameters* e, finalmente, *Generate Features*. As features espectrais são características extraídas de um sinal no domínio da frequência, que incluem informações sobre frequência, potência e outras propriedades. Esse processo é essencial para identificar padrões repetitivos, como movimentos ou vibrações captados por um acelerômetro, pois permite a identificação de componentes frequenciais significativos que podem não ser evidentes na análise temporal.

A análise espectral, ao converter os dados do domínio do tempo para o domínio da frequência, permite eliminar frequências indesejadas por meio de filtros passa-baixa e passa-alta, aprimorando a qualidade dos dados de entrada para o modelo de aprendizado de máquina [8]. O bloco de features espectrais é particularmente útil para sinais que apre-

sentam padrões irregulares ou transitórios permitindo que o modelo detecte e classifique movimentos com maior precisão.

O bloco de análise espectral oferece várias opções de configuração, como mostrado na Figura 2.5:

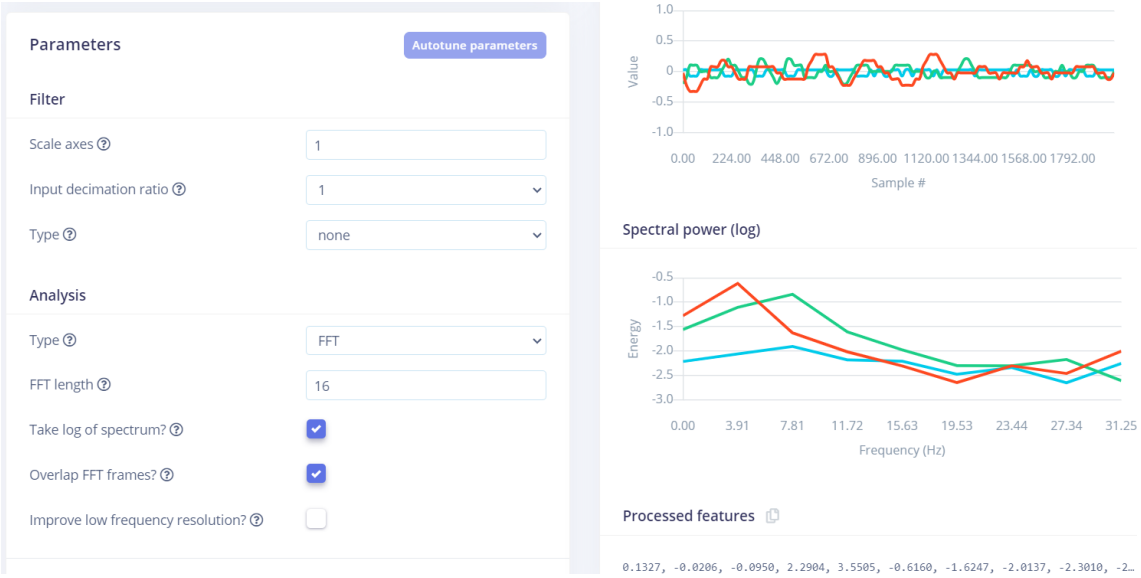


**Figura 2.5. Analise de Features na Edge Impulse.**

- **Filtro**: Antes de calcular a Transformada Rápida de Fourier (FFT) [23], os dados de séries temporais podem ser filtrados para suavizar o sinal ou remover artefatos indesejados. A janela de amostragem utilizada no filtro é ajustada com base nos parâmetros do bloco de entrada.
- **Escala de eixos**: Multiplica todos os valores de entrada brutos por um número específico, ajustando a amplitude dos dados.
- **Razão de dizimação de entrada**: Reduz a taxa de amostragem do sinal para diminuir a quantidade de features, melhorando a resolução de frequência sem aumentar o uso de recursos.
- **Tipo de filtro**: Escolha entre passa-baixa, passa-alta ou sem filtro para aplicar ao dado bruto.
- **Frequência de corte**: Define a frequência de corte em hertz, removendo frequências indesejadas e otimizando o tamanho do modelo.
- **Ordem**: A ordem do filtro Butterworth, determinando a precisão do corte de frequências. Um valor mais alto resulta em um corte mais acentuado, mas aumenta a latência.

A análise de potência espectral é uma abordagem baseada na FFT, e a configuração dos parâmetros, como o tamanho da FFT e a sobreposição dos frames, permite a extração precisa das características desejadas.

- **Tamanho da FFT**: Define a resolução de frequência e a quantidade de bins da FFT. Um tamanho menor reduz a quantidade de features e o tamanho do modelo, enquanto um tamanho maior permite separar mais sinais.
- **Logaritmo do espectro**: Aplica logaritmo aos bins da FFT, expandindo o alcance de sinais de baixa intensidade.
- **Sobreposição de frames da FFT**: Utiliza uma abordagem de *sliding frame* para evitar a perda de eventos transitórios entre frames, sendo ativado por padrão.

Para cada bin da FFT, o valor máximo é mantido como feature, enquanto frequências indesejadas são descartadas, otimizando o modelo para treinamento com menos dados.

#### 2.2.1.5. Treinando o Classificador

Com as features espectrais geradas, o próximo passo é treinar o modelo. Isso é feito clicando em *Classifier* e depois em *Start Training*, como mostrado na Figura 2.6. É nessa etapa que o modelo aprende a diferenciar entre as classes de movimento com base nos dados de entrada processados. Um classificador recebe dados de entrada e fornece uma pontuação de probabilidade que indica a probabilidade de os dados pertencerem a uma classe específica. No contexto do projeto de detecção de movimento, isso significa identificar padrões de movimento a partir dos dados do acelerômetro.

A Edge Impulse oferece vários blocos de treinamento predefinidos, como por exemplo:

- **Classificação (Keras)**: Ideal para diferenciar entre várias classes, como diferentes tipos de movimentos [4].
- **Regressão (Keras)**: Para prever valores contínuos [4].
- **Detecção de Anomalias (K-means, GMM)**: Para identificar padrões incomuns nos dados [5].
- **Classificação de Imagem (Transfer Learning, NVIDIA TAO)**: Para reconhecimento de imagens [12].
- **Detecção de Objetos (MobileNetV2, SSD, FPN, FOMO)**: Para localizar objetos em imagens [17] [25].
- **Reconhecimento de Palavras-chave**: Para identificar palavras específicas em áudio.

Para a tarefa de detecção de movimento, o bloco de *Classifier* é o mais indicado.

O classificador utilizado é uma rede neural, composta por múltiplas camadas de neurônios. Cada neurônio está interconectado com os neurônios da camada subsequente,

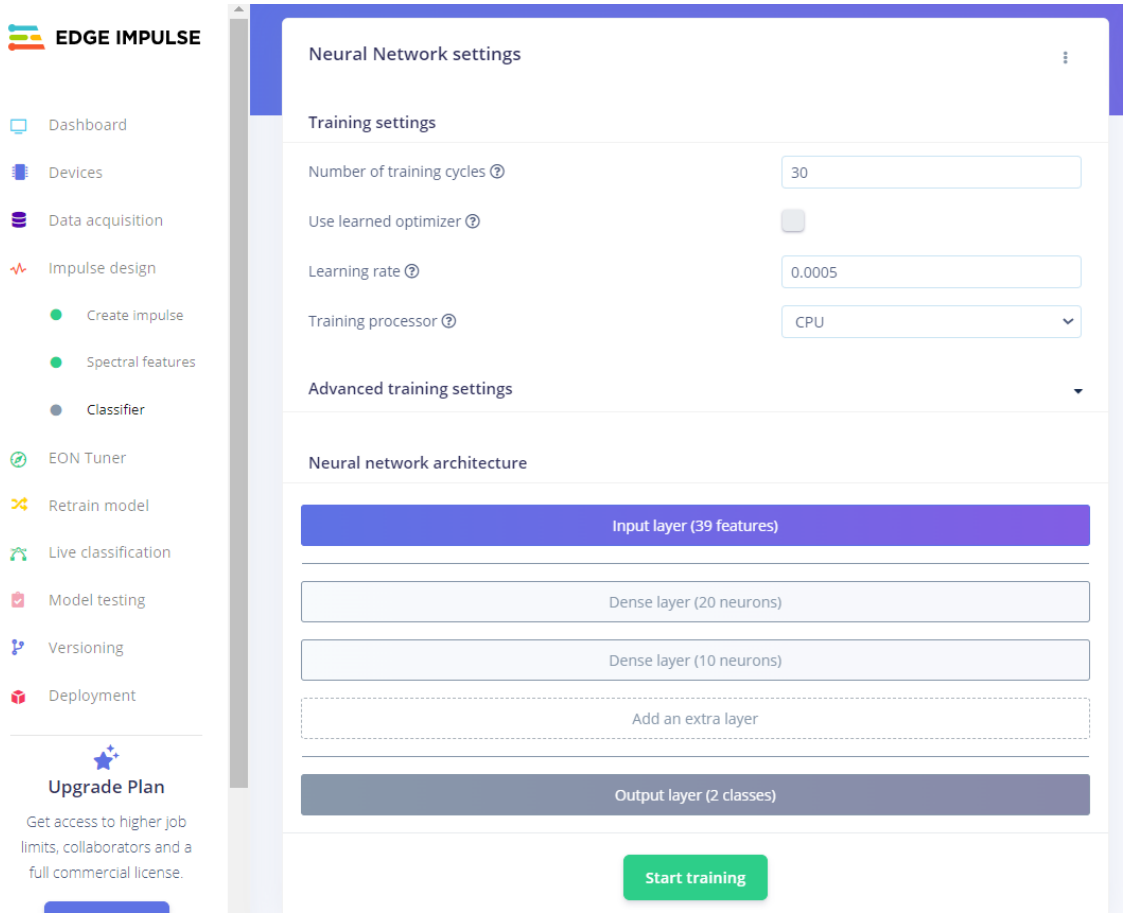


**Figura 2.6. Etapa de Treinamento na Edge Impulse.**

e os pesos dessas conexões são ajustados durante o treinamento. Inicialmente, os pesos são definidos aleatoriamente. Durante o processo de treinamento, a rede é alimentada com um conjunto de dados de treinamento, e a saída gerada pela rede é comparada com as respostas corretas. Com base nessa comparação, os pesos das conexões são ajustados por meio de um processo chamado *backpropagation* [15].

Essa arquitetura permite que a rede aprenda a mapear os dados de entrada para as classes corretas, aprimorando sua capacidade preditiva a cada iteração.

Na página de configuração do classificador, diversas opções estão disponíveis para otimizar o processo de treinamento, por exemplo:

- **Número de Ciclos de Treinamento**: Define quantas vezes o algoritmo percorrerá todos os dados de treinamento para ajustar os parâmetros do modelo.
- **Taxa de Aprendizado**: Controla a quantidade de ajuste dos parâmetros internos do modelo em cada passo. Uma taxa de aprendizado muito alta pode levar a um overfitting rápido.
- **Tamanho do Conjunto de Validação**: Percentual do conjunto de treinamento reservado para validação, geralmente 20%.
- **Tamanho do Lote (Batch Size)**: Número de amostras processadas antes de atualizar o modelo.
- **Autoajuste de Classes**: Foca mais em amostras de classes sub-representadas durante o treinamento para evitar overfitting.

A arquitetura da rede neural recebe as features extraídas como entrada e as processa através de suas camadas. No caso da classificação, a última camada usada é uma camada *softmax* [10], que fornece a probabilidade de pertencimento a uma das classes. Em modo visual, é possível adicionar diferentes tipos de camadas para ajustar a arquitetura conforme necessário.

Para usuários avançados, a Edge Impulse oferece um modo expert, permitindo acessar a API completa do Keras para customizar a arquitetura, ajustar a função de perda, o otimizador e até mesmo implementar *callbacks* para evitar o *overfitting*.

*Overfitting* ocorre quando o modelo se ajusta excessivamente aos dados de treinamento, capturando também o ruído e os detalhes irrelevantes, o que resulta em um desempenho ruim em novos dados. Prevenir o *overfitting* garante que o modelo seja mais generalizável e eficaz em diferentes conjuntos de dados [24].

Durante o treinamento, o painel de saída exibe os logs, permitindo acompanhar o progresso do modelo. Após o treinamento, a performance do modelo pode ser avaliada através de algumas métricas, como mostrado nas Figuras 2.7 e 2.8, são elas:

- **Acurácia de Validação e Função de Perda**: A acurácia de validação e a função de perda são métricas para avaliar o desempenho do modelo. A acurácia de validação mede a proporção de previsões corretas realizadas pelo modelo sobre o conjunto

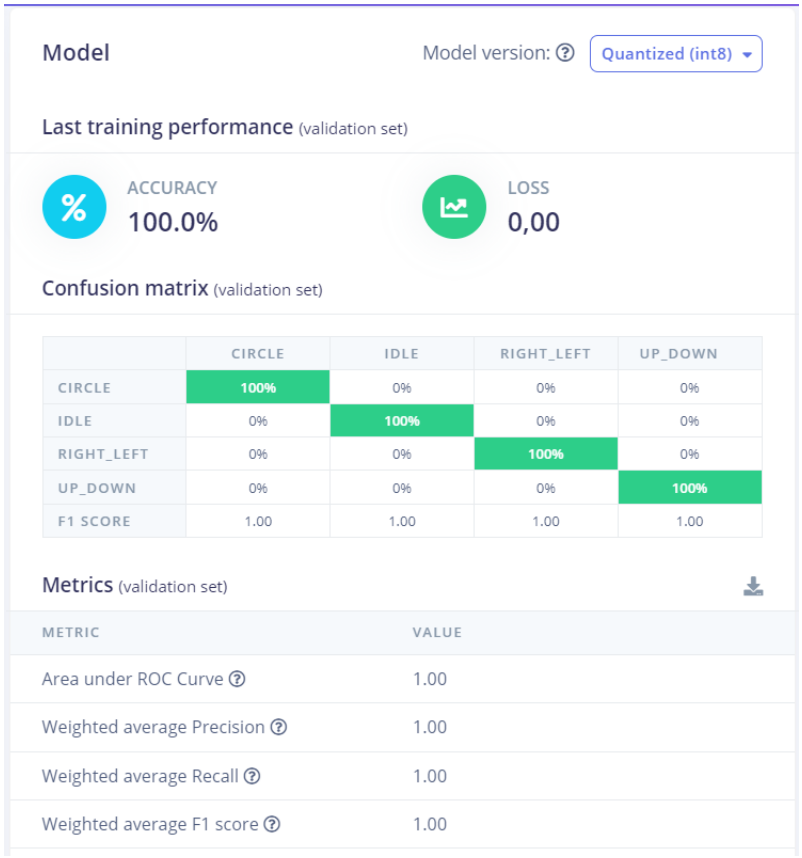


**Figura 2.7. Resultado do treinamento do Impulso.**

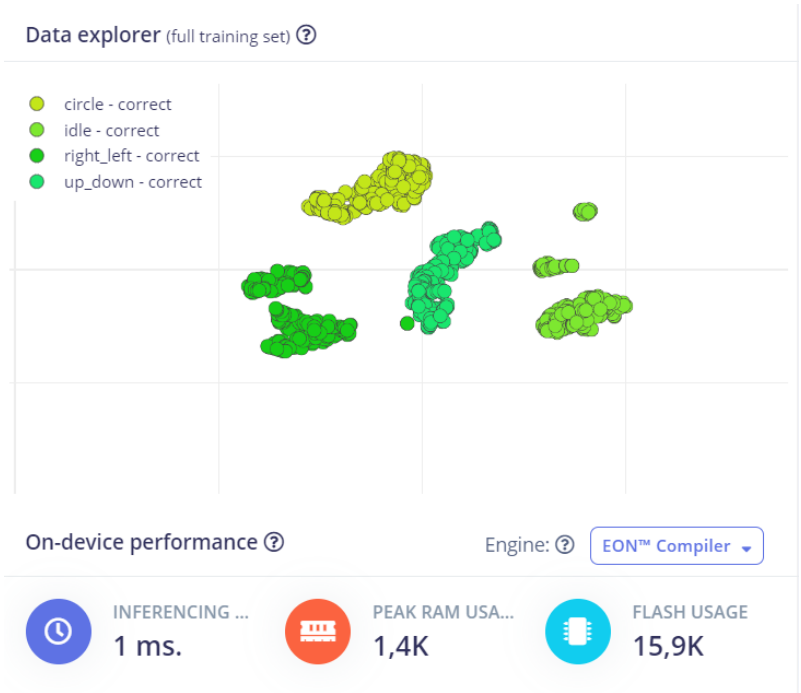


**Figura 2.8. Métricas de consumo no dispositivo.**

de validação, enquanto a função de perda quantifica o erro de previsão, refletindo a diferença entre as previsões do modelo e os valores reais.

- **Matriz de Confusão**: A matriz de confusão é uma ferramenta útil para a avaliação do desempenho do modelo, fornecendo uma representação detalhada das previsões corretas e incorretas. Ela permite a análise das taxas de verdadeiro positivo, falso positivo, verdadeiro negativo e falso negativo, facilitando a compreensão das forças e fraquezas do modelo.

- **Explorador de Características**: O explorador de características permite a visualização da distribuição das variáveis de entrada no espaço de características. Ele mostra quais variáveis foram corretamente classificadas e quais apresentaram dificuldades, oferecendo insights sobre a eficácia do modelo em distinguir entre diferentes classes e a qualidade das informações extraídas.

Com base no dispositivo alvo selecionado na página do Dashboard, estimativas de tempo de inferência, uso máximo de RAM e uso de flash são fornecidas, ajudando a validar se o modelo pode ser executado no dispositivo de destino dentro de suas limitações, como mostrado na Figura 2.8.

Este processo de treinamento do classificador é uma etapa importante no desenvolvimento do modelo de detecção de movimento, garantindo que o modelo possa aprender eficazmente a partir dos dados de treinamento e generalizar para novos dados de forma precisa e eficiente.

#### 2.2.1.6. Implantação do Modelo no Smartphone

Após o treinamento do modelo de aprendizado de máquina, é possível implantá-lo em um *smartphone* para realização de teste. Para isso, acesse a seção *Deployment* no painel do Edge Impulse e selecione *WebAssembly (browser, SIMD)* como o formato de implantação. Em seguida, use a câmera de um smartphone para escanear o QR code, que o redirecionará para uma página onde o modelo estará ativo para teste imediato, como mostrado na Figura 2.9. O uso do WebAssembly oferece algumas vantagens:

1. compatibilidade com todos os principais navegadores, permitindo execução sem instalação adicional;

2. desempenho otimizado para operar de forma rápida e eficiente;

3. portabilidade, eliminando a dependência de plataformas específicas e aumentando a flexibilidade da solução.

Com a implantação no smartphone, o dispositivo se torna um sensor de movimento, demonstrando a aplicabilidade prática do aprendizado de máquina em dispositivos móveis e destacando o potencial para soluções para o mundo real.

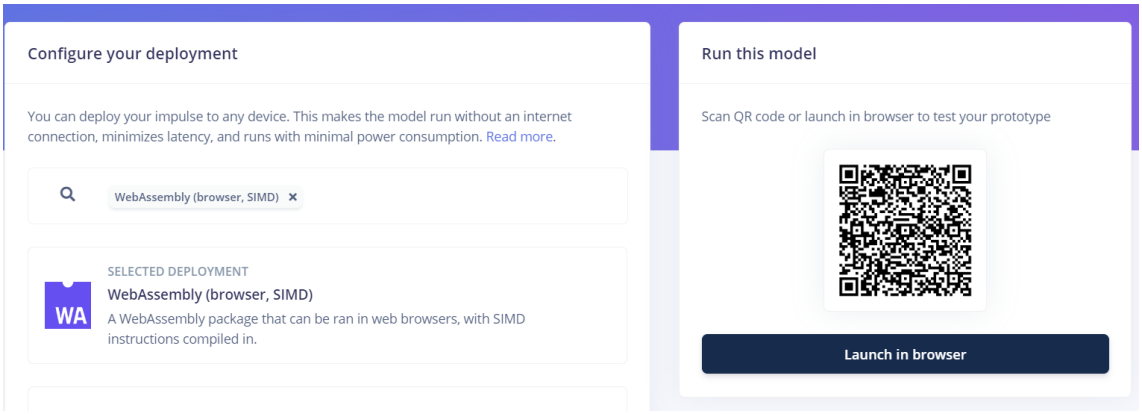


**Figura 2.9. Tela de deploy no smartphone.**

### 2.2.2. Reconhecimento de Movimento com o Kit TinyML

Nessa seção, o processo de detecção de movimento descrito anteriormente será repetido, mas agora utilizando o kit TinyML da Edge Impulse [14]. TinyML refere-se à implementação de modelos de aprendizado de máquina em dispositivos com recursos computacionais limitados, como microcontroladores e sensores integrados [21]. Essa tecnologia permite que algoritmos complexos sejam executados diretamente em dispositivos de borda, possibilitando a coleta e o processamento de dados em tempo real sem a necessidade de conexão com a nuvem.

O kit Arduino Tiny Machine Learning da Edge Impulse, ilustrado na Figura 2.10, é um exemplo dessa tecnologia. Ele inclui um microcontrolador otimizado para machine learning, sensores integrados e um ambiente de desenvolvimento que facilita a criação e a implantação de modelos de aprendizado de máquina diretamente no dispositivo. Este kit é usado em aplicações que requerem baixo consumo de energia, resposta rápida e operação autônoma, como na Internet das Coisas (IoT), automação industrial e dispositivos vestíveis.


**Figura 2.10. Kit Arduino Tiny Machine Learning Edge Impulse.**

Esta seção aborda configuração do kit TinyML, coleta de dados, treinamento do modelo e implantação da solução para reconhecimento de movimento.

#### 2.2.2.1. Conectando o Kit TinyML e Coletando Dados

Para iniciar a coleta de dados com o kit TinyML da Edge Impulse, o primeiro passo é criar um novo projeto na plataforma Edge Impulse. Recomenda-se nomear o projeto de forma descritiva, como "Detecção de Movimento com Arduino Nano 33 BLE Sense".

Com o projeto criado, siga os passos abaixo para conectar o dispositivo e coletar os dados:

1. **Conectando o Dispositivo:** Para conectar o Arduino Nano 33 BLE Sense ao computador, use um cabo micro-USB. Em seguida, pressione o botão RESET duas vezes rapidamente para iniciar o bootloader. O LED da placa deve começar a piscar, indicando que o dispositivo está pronto para a conexão.

2. **Atualizando o Firmware:** A placa não vem com o firmware adequado pré-instalado. Para atualizá-lo, baixe o firmware mais recente da Edge Impulse e descompacte o arquivo. Execute o script de flash apropriado para seu sistema operacional (flash_windows.bat, flash_mac.command ou flash_linux.sh) para atualizar o firmware da placa. Após a conclusão do processo, pressione o botão RESET uma vez para iniciar o novo firmware.

3. **Configurando as Chaves:** Abra um prompt de comando ou terminal e execute o comando edge-impulse-daemon. Isso iniciará um assistente que solicitará o login e a seleção do projeto Edge Impulse. Caso deseje mudar de projeto, adicione a opção –clean ao comando. Em versões recentes dos navegadores Google Chrome e Microsoft Edge, também é possível coletar dados diretamente do seu dispositivo sem a necessidade do CLI da Edge Impulse.

Para a coleta de dados, siga os mesmos passos descritos na Seção 2.2.1.2. No Edge Impulse, clique em *Data Acquisition*. No campo de seleção do tipo de sensor, escolha *Acelerômetro*. No campo *Label*, insira o tipo de movimento que deseja registrar e colete dados para todos os movimentos desejados. Repita o processo para cada tipo de movimento a ser registrado.

### 2.2.3. Projetando Impulso, Gerando as *Features Espectrais* e Treinando o Classificador para o Kit TinyML

Após a coleta de dados para o projeto, os seguintes passos envolvem a configuração do Impulse, a extração das features espectrais e o treinamento do classificador. Para realizar esses procedimentos, consulte as Seções 2.2.1.3, 2.2.1.4, e 2.2.1.5.

Inicialmente, crie o Impulse acessando a opção *Impulse Design*. Adicione um bloco de processamento selecionando *Spectral Analysis*, e em seguida, insira um bloco de aprendizado escolhendo *Classification* para incluir um classificador baseado em rede neural. Após a configuração dos blocos, finalize o processo clicando em *Save Impulse*.

Com o Impulse configurado, prossiga para a geração das features espectrais clicando em *Spectral Features*, seguido de *Save Parameters* e, finalmente, *Generate Features*. Este processo extrai características frequenciais e outras propriedades relevantes do sinal, essenciais para a análise dos dados.

Finalmente, para treinar o modelo, selecione *Classifier* e clique em *Start Training*. O treinamento ajustará o classificador para identificar padrões de movimento com base nas *features espectrais* geradas, conforme ilustrado na Figura 2.6.

#### 2.2.3.1. Implantação no Kit Arduino TinyML

Para implantar o modelo treinado no kit Arduino TinyML, siga os seguintes passos:

1. **Implantação do Modelo:** No Edge Impulse, vá para a seção *Deployment* e selecione o dispositivo *Arduino Nano 33 BLE Sense*. Clique em *Build Model* para compilar o modelo treinado para o dispositivo.

2. **Atualização do Dispositivo:** Após a compilação, baixe o arquivo de firmware gerado e execute o script de flash correspondente ao seu sistema operacional (por exemplo, flash_windows.bat para Windows, flash_mac.command para macOS, ou flash_linux.sh para Linux) para atualizar o firmware no Arduino Nano 33 BLE Sense.

3. **Execução do Modelo:** Após a atualização do firmware, execute o comando edge-impulse-run-impulse no terminal ou prompt de comando do computador ao qual o dispositivo está conectado. O Arduino Nano 33 BLE Sense começará a realizar inferências em tempo real e exibirá os resultados no console.

Com o modelo treinado implantado no kit TinyML, o dispositivo está pronto para realizar inferências em tempo real, demonstrando a capacidade de detecção de movimento diretamente na borda. Este processo ilustra como tecnologias de aprendizado de máquina podem ser integradas em sistemas embarcados para aplicações eficientes e autônomas. No entanto, para entender plenamente as possibilidades e os desafios do TinyML, é essencial aprofundar-se nos conceitos fundamentais e nas abordagens que tornam essa tecnologia viável. A Seção 2.3 explora esses conceitos em detalhes, proporcionando uma base sólida para o desenvolvimento e a aplicação de soluções TinyML.

## 2.3. Conceitos Fundamentais e Abordagens em TinyML

No campo da tecnologia, termos como *Edge AI* e *TinyML* tornaram-se comuns. Para compreender plenamente essas inovações e suas aplicações, é essencial explorar os conceitos fundamentais que as sustentam. Esta seção, fundamentada no trabalho apresentado em 19, aborda os seguintes temas centrais: Sistemas Embarcados, Computação na Borda e Internet das Coisas, Inteligência Artificial, Aprendizado de Máquina, Inteligência Artificial na Borda, Aprendizado de Máquina Embarcado e Tiny Machine Learning, e Processamento Digital de Sinais. Ao entender esses conceitos, será possível apreciar melhor a complexidade e o potencial transformador do TinyML.

### 2.3.1. Sistemas Embarcados

Os sistemas embarcados são computadores projetados especificamente para executar funções dedicadas em diversos dispositivos físicos, como fones de ouvido Bluetooth e unidades de controle de motores em veículos modernos [11]. O software embarcado, que opera nestes sistemas, pode variar desde simples códigos em microcontroladores de relógios digitais até sistemas operacionais mais sofisticados, como o Linux integrado em smart TVs. Ao contrário dos computadores de uso geral, como laptops e smartphones, os sistemas embarcados são desenvolvidos para realizar tarefas específicas e dedicadas.

A relevância dos sistemas embarcados na tecnologia moderna é evidente. Em 2020, estima-se que mais de 28 bilhões de microcontroladores foram distribuídos globalmente, destacando a onipresença desses sistemas em nossa vida cotidiana, desde residências e veículos até fábricas e cidades [9]. Esta ubiquidade sugere que é raro estarmos a mais de alguns metros de um sistema embarcado.

Estes sistemas frequentemente enfrentam limitações impostas pelos ambientes em que operam, como a necessidade de eficiência energética e restrições de memória ou velocidade de *clock*. O desenvolvimento de software para sistemas embarcados envolve a criação de soluções que maximizam a funcionalidade dentro dessas limitações, ressaltando a complexidade e a importância do trabalho dos engenheiros especializados na área.

Neste capítulo, a discussão sobre sistemas embarcados se conecta diretamente ao desenvolvimento e à aplicação de técnicas de aprendizado de máquina, particularmente no contexto do TinyML. A capacidade de implementar modelos de aprendizado de máquina em sistemas embarcados permite a realização de análises avançadas e a execução de tarefas inteligentes diretamente em dispositivos com recursos limitados. A compreensão das restrições e capacidades dos sistemas embarcados é essencial para o desenvolvimento eficaz de soluções que aproveitam o processamento local para oferecer desempenho eficiente e respostas em tempo real.

### 2.3.2. Computação na Borda e Internet das Coisas

A evolução das redes de computadores tem alternado entre centralização e descentralização do processamento ao longo do tempo. Nos primórdios, o processamento era centralizado em grandes mainframes, com terminais periféricos assumindo funções específicas. A introdução dos computadores pessoais levou a uma maior descentralização, permitindo que dispositivos individuais realizassem tarefas de forma independente de um servidor central.

Com o surgimento da internet e das soluções baseadas na nuvem, houve um retorno à centralização dos processos. No entanto, a expansão da Internet das Coisas (IoT) trouxe uma nova perspectiva, com uma vasta rede de dispositivos interconectados [7].

Esses dispositivos situados na borda da rede utilizam a computação na borda (Edge Computing), uma abordagem que permite o processamento local dos dados. Esse método possibilita a coleta e análise de dados diretamente nos dispositivos, a tomada de decisões em tempo real e a comunicação entre eles, minimizando a necessidade de processamento e armazenamento na nuvem [2].

Os dispositivos de borda desempenham um papel fundamental na conexão entre o

mundo físico e a internet. Dispositivos móveis, como smartphones e tablets, são exemplos de tecnologias de borda que incorporam inteligência artificial para oferecer funcionalidades avançadas, como reconhecimento de voz e fotografia inteligente.

### 2.3.3. Inteligência Artificial

A Inteligência Artificial (IA) refere-se ao campo da ciência da computação que busca desenvolver sistemas capazes de executar tarefas que normalmente requerem inteligência humana. Este campo é vasto e complexo, abrangendo desde a resolução de problemas até a otimização de processos por meio de máquinas e algoritmos [16].

O conceito de inteligência é multifacetado e difícil de definir de maneira universal. Por exemplo, organismos como o mofo do limo demonstram capacidades de resolução de problemas complexos, como percorrer labirintos, apesar de não possuírem um sistema nervoso central. Uma definição prática de inteligência poderia ser a habilidade de escolher e executar a ação apropriada em situações específicas.

Atividades cotidianas, como acionar uma torneira para lavar as mãos ou frear um veículo para evitar uma colisão, podem parecer simples, mas exigem um nível de inteligência para serem realizadas de maneira eficaz. Enquanto a inteligência humana geral é adaptativa e pode lidar com uma ampla gama de tarefas, a IA geralmente é projetada para tarefas específicas.

Desenvolver uma inteligência artificial com a mesma amplitude e flexibilidade da inteligência humana é uma tarefa complexa e desafiadora. No entanto, a criação de sistemas de IA especializados para tarefas concretas é não apenas viável, mas amplamente realizada. Assim, a IA pode ser vista como a implementação de sistemas artificiais que tomam decisões baseadas em dados e utilizam técnicas de aprendizado de máquina para melhorar seu desempenho ao longo do tempo.

### 2.3.4. Aprendizado de Máquina

O aprendizado de máquina (ML) é uma abordagem que permite descobrir padrões e tendências em dados, automatizando esse processo por meio de algoritmos.Embora frequentemente associado à inteligência artificial (IA), o aprendizado de máquina (ML) é, na verdade, uma subárea da IA. Enquanto a IA engloba uma variedade de técnicas e abordagens para simular a inteligência humana, o ML é especificamente voltado para a criação de algoritmos que aprendem e fazem previsões com base em dados. Dessa forma, o ML se insere dentro do escopo mais amplo da IA, complementando e expandindo suas capacidades [1].

Para ilustrar o conceito de aprendizado de máquina, considere o exemplo de um sistema de reconhecimento de movimento contínuo descrito neste Capítulo. Neste cenário, um dispositivo, como um smartphone ou um wearable, é equipado com sensores de movimento, como acelerômetros e giroscópios. O objetivo é identificar e classificar diferentes tipos de movimentos, como caminhada, corrida ou gestos específicos, com base nos dados capturados pelos sensores.

Primeiramente, dados são coletados a partir de um conjunto de movimentos realizados por indivíduos. Esses dados incluem medidas contínuas de aceleração e rotação em

diferentes eixos. O objetivo é extrair padrões que permitam distinguir entre os diversos tipos de movimentos. A análise manual desses dados pode ser desafiadora e exigir técnicas avançadas para identificar os padrões relevantes. É aqui que o aprendizado de máquina se mostra valioso.

Com o uso de ML, os dados coletados são processados por um algoritmo de treinamento, que busca mapear a relação entre os sinais dos sensores e os tipos de movimentos correspondentes, criando um modelo. Se o treinamento for bem-sucedido, o modelo será capaz de identificar e classificar novos movimentos com base nas informações capturadas pelos sensores.

A capacidade do modelo de aplicar seu conhecimento a novos dados é chamada de generalização. Durante a fase de treinamento, o modelo aprende a identificar características específicas de cada atividade, permitindo que o rastreador interprete novos dados com precisão. Embora o ML utilize lógica semelhante à lógica condicional, ele oferece uma abordagem mais precisa e eficiente para a análise de dados complexos.

Existem diversos algoritmos de aprendizado de máquina, cada um com suas próprias vantagens e desvantagens. O aprendizado de máquina é particularmente valioso quando se trata de dados que são difíceis de analisar manualmente.

### 2.3.5. Inteligência Artificial na Borda

A inteligência artificial na borda, ou Edge AI, representa a integração de capacidades de IA em dispositivos localizados na borda da rede. Estes sistemas embarcados, equipados com sensores, capturam dados ambientais e permitem a análise desses dados sem necessidade de transmitir grandes volumes para servidores centrais. Embora a coleta de dados seja intensa, a informação isolada de cada sensor pode ser insuficiente por si só [20].

No passado, os dispositivos da Internet das Coisas (IoT) atuavam principalmente como coletores de dados, enviando informações para processamento centralizado. No entanto, a transmissão contínua de grandes quantidades de dados pode ser cara e energeticamente ineficiente, especialmente para dispositivos IoT movidos a bateria. Isso frequentemente resulta no descarte de dados valiosos devido à incapacidade de transmitir e processar tudo de maneira eficaz.

A abordagem Edge AI enfrenta essas limitações processando dados diretamente no dispositivo onde são gerados. Esse método permite a tomada de decisões localmente, minimizando a necessidade de envio de dados para servidores distantes e reduzindo assim custos e consumo de energia.

A inteligência artificial aplicada na borda pode variar desde a implementação de regras simples até o uso de técnicas avançadas de aprendizado profundo. Essa estratégia não apenas melhora a eficiência ao tomar decisões próximas à origem dos dados, mas também se encaixa bem em arquiteturas de computação distribuída, onde as operações são distribuídas entre dispositivos na borda, gateways locais e servidores na nuvem.

### 2.3.6. Aprendizado de Máquina Embarcado e Tiny Machine Learning

O conceito de aprendizado de máquina embarcado refere-se à implementação de modelos de aprendizado de máquina em dispositovos projetados para operar com recursos limita-

dos. A abordagem conhecida como Tiny Machine Learning (TinyML) expande essa ideia para ambientes com hardware extremamente restrito, como microcontroladores, processadores digitais de sinal (DSPs) e pequenas FPGAs (Field Programmable Gate Arrays) [22].

Em geral, o aprendizado de máquina embarcado foca na fase de inferência, que consiste em processar dados de entrada e gerar previsões ou classificações. Por exemplo, pode-se usar um modelo para reconhecer atividades físicas com base nas leituras de um acelerômetro. A fase de treinamento, que exige consideráveis recursos computacionais, ainda é realizada em máquinas mais potentes, como computadores pessoais ou servidores.

Os sistemas embarcados enfrentam desafios notáveis, especialmente em termos de memória. A execução de modelos de aprendizado de máquina pode ser limitada pela quantidade de memória somente leitura (ROM) disponível para armazenar o modelo e pela memória de acesso aleatório (RAM) necessária para processar dados temporários durante a inferência. Além disso, o poder computacional restrito desses sistemas pode dificultar a implementação de modelos complexos.

Recentemente, avanços em técnicas de otimização têm permitido a execução eficiente de modelos de aprendizado de máquina em hardware com recursos limitados e baixo consumo de energia. Esses avanços possibilitam a operação de modelos complexos nesses dispositivos, tornando a inteligência artificial mais acessível em ambientes com restrições de hardware.

## 2.4. Quando Utilizar Edge AI

A relevância do Edge AI torna-se evidente ao analisar o ambiente tecnológico atual, onde a conectividade é amplamente disponível, mesmo em locais remotos. Imagine, por exemplo, uma manhã em um parque nacional distante, onde a música é transmitida pelo smartphone e as fotos são enviadas sem dificuldades. Dada a ubiquidade das conexões de dados, qual seria a vantagem de aplicar inteligência artificial diretamente em dispositivos de borda, se servidores poderosos estão acessíveis?

Edge AI oferece soluções práticas que podem melhorar significativamente a eficiência e a funcionalidade dos sistemas tecnológicos. Para compreender melhor os benefícios do Edge AI, podemos recorrer ao acrônimo BLERP, que representa: *Bandwidth*, *Latency*, *Economics*, *Reliability* e *Privacy*. Este acrônimo foi descrito em detalhes na referência [19], destacando as principais razões para implementar inteligência na borda.

### 2.4.1. Largura de Banda

A largura de banda limitada é um desafio significativo para dispositivos IoT, que frequentemente geram mais dados do que podem transmitir devido a restrições na capacidade de rede. Por exemplo, considere um sistema de monitoramento ambiental que utiliza sensores para medir a qualidade do ar em uma área remota. Esses sensores podem coletar dados sobre poluentes atmosféricos, mas a capacidade de transmitir todos esses dados para um centro de análise pode ser restrita.

Se a largura de banda for insuficiente, a maior parte dos dados coletados pode ser descartada, mesmo que contenham informações importantes sobre tendências ou padrões. Embora o envio dos dados para um servidor na nuvem permita análises aprofundadas,

essa abordagem não é sempre viável devido às limitações de conectividade e consumo de energia.

Em muitos cenários, o consumo de energia associado à transmissão de dados é elevado, o que reduz a duração da bateria dos dispositivos IoT. Por exemplo, a comunicação contínua com servidores pode esgotar rapidamente a bateria de um sensor ambiental que coleta dados em tempo real. Alguns algoritmos de aprendizado de máquina podem ser intensivos em processamento, mas frequentemente consomem menos energia do que a transmissão constante de grandes volumes de dados.

A solução para esses desafios é a inteligência artificial na borda (Edge AI). Com a Edge AI, é possível realizar a análise dos dados diretamente no dispositivo, sem a necessidade de enviar todas as informações para a nuvem. Isso permite que o dispositivo identifique e reaja a padrões importantes, como picos de poluição, com a largura de banda disponível de maneira mais eficiente. Dessa forma, o sistema pode emitir alertas locais com base na análise realizada no próprio dispositivo.

Além disso, para dispositivos que operam em locais sem conectividade de rede, a Edge AI oferece uma solução crucial. Esses dispositivos podem realizar análise e tomada de decisão no local, tornando viável a implementação de aplicações avançadas em áreas anteriormente inacessíveis.

### 2.4.2. Latência

O tempo necessário para transmitir dados entre um dispositivo e um servidor pode ser considerável, mesmo em redes com alta largura de banda. A latência, que é o intervalo entre o envio e o recebimento dos dados, pode variar de milissegundos a minutos ou até horas, dependendo da distância e da infraestrutura envolvida. Em sistemas de comunicação via satélite, por exemplo, a latência pode ser significativamente maior, complicando a interação em tempo real.

Para muitas aplicações, especialmente aquelas que demandam respostas rápidas, a latência elevada pode ser um problema crítico. Por exemplo, em sistemas de controle remoto de drones, a capacidade de responder rapidamente a comandos é essencial para a navegação segura e precisa. Se o tempo de resposta for muito longo, o controle sobre o drone pode ser comprometido, afetando a eficácia da operação.

A Edge AI surge como uma solução eficaz para esses desafios de latência, ao processar e analisar dados localmente no próprio dispositivo. Um exemplo notável é o uso de Edge AI em veículos autônomos, onde sistemas de IA a bordo permitem uma resposta imediata a eventos, como a detecção repentina de obstáculos. Isso garante que o veículo possa ajustar sua trajetória instantaneamente, sem depender da comunicação com servidores distantes.

### 2.4.3. Economia

O custo de conectar dispositivos e transmitir dados pode ser elevado, especialmente quando se considera a infraestrutura necessária para suportar essa conectividade. Dispositivos que transmitem grandes quantidades de dados, particularmente em áreas remotas ou de difícil acesso, podem gerar despesas significativas. O aumento da largura de banda necessária

para transmitir dados também contribui para esses custos, sendo um desafio crucial para dispositivos que operam em locais isolados e dependem de conexões caras como satélites.

Uma solução eficaz para mitigar esses custos é a implementação de Edge AI, que processa dados diretamente no local de captura. Ao realizar o processamento localmente, em vez de enviar constantemente grandes volumes de dados para servidores externos ou para a nuvem, os dispositivos podem reduzir significativamente os custos associados à transmissão e ao armazenamento de dados.

Considere o exemplo de uma rede de sensores usados para monitorar a qualidade da água em um sistema de distribuição. Esses sensores podem detectar variações importantes na composição da água e gerar grandes volumes de dados continuamente. Tradicionalmente, enviar todos esses dados para análise em tempo real poderia ser caro e ineficiente, especialmente se a rede se estender por grandes distâncias.

Com Edge AI, os sensores podem realizar análises preliminares dos dados localmente e apenas relatar informações críticas ou anômalas. Isso reduz a necessidade de transmissão contínua de grandes quantidades de dados e minimiza o consumo de largura de banda. Como resultado, os custos operacionais são reduzidos, e a eficiência do sistema de monitoramento é aprimorada.

Essa abordagem também é vantajosa em situações em que a conectividade é limitada ou intermitente, permitindo que o sistema opere de forma independente e apenas compartilhe informações essenciais quando necessário.

### 2.4.4. Confiabilidade

A confiabilidade dos sistemas que utilizam inteligência artificial pode ser significativamente afetada pela dependência de conexões contínuas com a nuvem. Quando um dispositivo depende de uma rede para operar, ele está sujeito a uma série de variáveis externas, incluindo a integridade da comunicação sem fio e a estabilidade dos servidores na internet. Cada um desses elementos representa uma possível fonte de falha.

Quando um dispositivo se baseia em uma conexão constante para realizar suas funções, o sistema pode enfrentar problemas se a conectividade falhar. Por exemplo, um dispositivo inteligente de controle de temperatura em um edifício que depende da nuvem para ajustar as configurações pode falhar em manter a temperatura adequada se a conexão for interrompida, resultando em desconforto e possivelmente danos.

Por outro lado, um sistema embarcado que incorpora inteligência artificial localmente é menos vulnerável a essas falhas de comunicação. Um bom exemplo é um dispositivo de segurança em uma casa, que monitora e reage a movimentos suspeitos. Se esse dispositivo for totalmente autônomo e não depender de comunicação constante com um servidor remoto, ele continuará a operar efetivamente mesmo se houver uma interrupção na conexão com a internet. Isso é crucial em situações em que a continuidade do serviço é essencial para a segurança.

A confiança em sistemas que operam localmente elimina muitas das incertezas associadas à dependência de uma infraestrutura de rede externa. Isso garante que o sistema possa manter suas funções críticas, independentemente de problemas de conectividade, aumentando a segurança e a confiabilidade geral.

### 2.4.5. Privacidade

A crescente integração de tecnologias inteligentes em nosso cotidiano frequentemente implica uma troca entre a conveniência e a proteção dos dados pessoais. Muitos usuários aceitam compartilhar suas informações para beneficiar-se de funcionalidades avançadas oferecidas por dispositivos conectados, aceitando que suas informações sejam processadas por servidores remotos na nuvem.

Para alguns dispositivos, como um rastreador de exercícios que envia dados de desempenho para análise, a preocupação com a privacidade pode ser mínima. No entanto, para outros casos, a proteção da privacidade é crucial. Um exemplo seria um assistente doméstico inteligente que coleta dados de áudio contínuos. Mesmo com a promessa de conveniência, muitos usuários podem se sentir desconfortáveis com a ideia de ter gravações de suas conversas pessoais transmitidas e armazenadas em servidores externos.

A tecnologia Edge AI fornece uma solução eficaz para mitigar essas preocupações. Em vez de enviar continuamente dados sensíveis para a nuvem, dispositivos equipados com inteligência local podem realizar análises diretamente no próprio dispositivo. Por exemplo, um dispositivo de monitoramento de saúde pode analisar sinais vitais e detectar anomalias sem enviar informações para fora do dispositivo. Se uma anomalia for detectada, o sistema pode alertar o usuário ou o profissional de saúde de forma segura, sem comprometer dados pessoais.

Esse modelo de processamento local protege a privacidade ao evitar a transmissão de dados sensíveis, limitando o risco de exposição e possíveis abusos. Ao manter as informações dentro do dispositivo, a Edge AI garante que dados pessoais sejam utilizados de maneira mais segura e controlada, respeitando a privacidade dos usuários.

## 2.5. Considerações Finais

Neste capítulo, foram explorados os princípios fundamentais e as abordagens técnicas que sustentam a tecnologia TinyML, destacando como a integração da inteligência artificial (IA) em sistemas embarcados pode transformar diversos setores, oferecendo soluções mais autônomas e eficientes. Inicialmente, discutimos a aplicação prática de TinyML, através de projetos com smartphones e kits TinyML, demonstrando como esses dispositivos podem ser utilizados para tarefas como a detecção de movimento. Estes exemplos práticos ilustram a viabilidade e o impacto real da tecnologia em cenários do dia a dia.

O capítulo avançou para uma análise aprofundada dos desafios e benefícios associados à Edge AI. Foi abordado como a tecnologia pode superar limitações críticas de largura de banda, latência e custos. A capacidade de realizar processamento local é essencial para minimizar a transmissão de dados, economizando largura de banda e reduzindo os custos operacionais. Isso é particularmente importante em ambientes com conectividade limitada, onde a análise local permite a continuidade das operações e aumenta a resiliência dos sistemas.

A redução da latência foi outro ponto crucial discutido, destacando como a Edge AI permite respostas quase instantâneas ao processar dados diretamente no dispositivo. Isso é fundamental para aplicações que requerem decisões rápidas e precisas, como em veículos autônomos e sistemas de controle industrial, onde a rapidez e a eficácia das

respostas são essenciais para a segurança e a eficiência.

Além dos desafios econômicos, a integração de Edge AI foi analisada sob a perspectiva da redução dos custos de infraestrutura e processamento na nuvem. A análise local não só reduz a necessidade de servidores e largura de banda, mas também torna a tecnologia mais acessível e econômica. Em contextos de conectividade limitada ou inexistente, essa abordagem é crucial para a operação contínua e para a resiliência a falhas de comunicação.

A confiabilidade dos sistemas embarcados também foi abordada, ressaltando que a análise local reduz a vulnerabilidade a falhas de conectividade e problemas de rede. A Edge AI melhora a robustez das aplicações, proporcionando maior estabilidade e segurança para sistemas críticos.

No tocante à privacidade, a Edge AI oferece uma solução ao minimizar a transmissão de dados pessoais para a nuvem. Isso reduz o risco de exposição e abusos, especialmente em áreas sensíveis como segurança, saúde e educação, onde a proteção de dados é essencial.

A integração dos conceitos e práticas discutidos neste capítulo oferece uma base para a compreensão e aplicação da tecnologia TinyML. Os projetos práticos com smartphones e kits TinyML demonstram a aplicabilidade da tecnologia, enquanto as análises dos desafios e benefícios ressaltam a importância da Edge AI na evolução das tecnologias embarcadas.

À medida que a tecnologia avança, espera-se que o TinyML continue a expandir as fronteiras da inovação tecnológica, oferecendo novas oportunidades para desenvolvimento e aplicação em diversos setores. Compreender as nuances da Edge AI é essencial para profissionais e entusiastas que buscam se destacar nas inovações tecnológicas. Espera-se que o conhecimento adquirido inspire novas ideias e soluções, contribuindo para um futuro em que a inteligência artificial se integre de maneira eficiente e segura em no cotidiano.

## Referências


[1] Yoshua Bengio. *Deep Learning*. Adaptive Computation and Machine Learning series. MIT Press, London, England, November 2016.

[2] Jie Cao, Quan Zhang, and Weisong Shi. *Edge Computing: A Primer*. Springer International Publishing, Cham, 2018.

[3] Gonçalo Carvalho, Bruno Cabral, Vasco Pereira, and Jorge Bernardino. Edge computing: current trends, research challenges and future directions. *Computing*, 103(5):993–1023, May 2021.

[4] François Chollet et al. Keras. `https://keras.io`, 2015.

[5] J. A. Hartigan and M. A. Wong. A k-means clustering algorithm. *JSTOR: Applied Statistics*, 28(1):100–108, 1979.

[6] Shawn Hymel, Colby R. Banbury, Daniel Situnayake, Alex Elium, Carl Ward, Matthew Kelcey, Mathijs Baaijens, Mateusz Majchrzycki, Jenny Plunkett, David

Tischler, Alessandro Grande, Louis Moreau, Dmitry Maslov, Arthur Beavis, Jan Jongboom, and Vijay Janapa Reddi. Edge impulse: An mlops platform for tiny machine learning. *ArXiv*, abs/2212.03332, 2022.

[7] F Khodadadi, A V Dastjerdi, and R Buyya. Internet of things: An overview. In *Internet of Things*, pages 3–27. Elsevier, 2016.

[8] Richard G Lyons. *Understanding digital signal processing*. Prentice Hall, Philadelphia, PA, 3 edition, November 2010.

[9] MarketsandMarkets. Global microcontroller market: Analysis and forecast. Technical report, MarketsandMarkets, 2021.

[10] A. Martins and R. Astudillo. From softmax to sparsemax: A sparse model of attention and multi-label classification. In *International Conf. on Machine Learning - ICML*, volume 48, pages 1614–1623, June 2016.

[11] K C S Murti. *Design principles for embedded systems*. Transactions on Computer Systems and Networks. Springer, Singapore, Singapore, 1 edition, September 2021.

[12] S.J. Pan and Q. Yang. A Survey on Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, 2010.

[13] Visal Rajapakse, Ishan Karunanayake, and Nadeem Ahmed. Intelligence at the extreme edge: A survey on reformable tinyml. *ACM Comput. Surv.*, 55(13s), jul 2023.

[14] Partha Pratim Ray. A review on TinyML: State-of-the-art and prospects. *J. King Saud Univ. - Comput. Inf. Sci.*, 34(4):1595–1623, April 2022.

[15] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986.

[16] Stuart Russell and Peter Norvig. *Artificial intelligence*. Pearson, Upper Saddle River, NJ, 4 edition, November 2020.

[17] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks, 2018. cite arxiv:1801.04381.

[18] Raghubir Singh and Sukhpal Singh Gill. Edge AI: A survey. *Internet of Things and Cyber-Physical Systems*, 3:71–92, 2023.

[19] Weixing Su, Linfeng Li, Fang Liu, Maowei He, and Xiaodan Liang. Ai on the edge: a comprehensive review. *Artif. Intell. Rev.*, 55(8):6125–6183, dec 2022.

[20] Xiaofei Wang, Han Yu, Chunyan Miao, and Qiang Yang. *Edge AI: Convergence of Edge Computing and Artificial Intelligence*. Springer, 2020.

[21] Pete Warden. *Tiny ML*. O'Reilly Media, Sebastopol, CA, January 2020.

[22] Pete Warden. *Tiny ML*. O'Reilly Media, Sebastopol, CA, January 2020.

[23] William H. Press, Saul A Teukolsky, William T Vetterling, and Brian P Flannery. *Numerical recipes 3rd edition*. Cambridge University Press, Cambridge, England, 3 edition, September 2007.

[24] Xue Ying. An overview of overfitting and its solutions. *Journal of Physics: Conference Series*, 1168(2):022022, feb 2019.

[25] Orr Zohar, Kuan-Chieh Wang, and Serena Yeung. Prob: Probabilistic objectness for open world object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11444–11453, June 2023.