UNIVERSIDADE FEDERAL DE MINAS GERAIS

ESCOLA DE ENGENHARIA

CURSO DE GRADUAÇÃO EM ENGENHARIA DE SISTEMAS

JOSE ADONAI ARRUDA DOS SANTOS JUNIOR

APLICAÇÃO DE TÉCNICAS DE APRENDIZADO DE MÁQUINA PARA RECONHECIMENTO DE SINAIS DE LIBRAS EM AMBIENTE EMBARCADO

BELO HORIZONTE

2022

JOSE ADONAI ARRUDA DOS SANTOS JUNIOR

APLICAÇÃO DE TÉCNICAS DE APRENDIZADO DE MÁQUINA PARA RECONHECIMENTO DE SINAIS DE LIBRAS EM AMBIENTE EMBARCADO

Trabalho de Conclusão de Curso II apresentado como requisito parcial à obtenção do título de Engenheiro de Sistemas, da Escola de Engenharia da Universidade Federal de Minas Gerais.

Orientador: Prof. Frederico Gualberto Ferreira Coelho

BELO HORIZONTE

2022

AGRADECIMENTOS

Agradeço primeiramente à minha mãe e a meu falecido pai por toda educação que me deram. Por terem me ensinado que só é possível alcançar nossos sonhos após muito estudo e trabalho duro.

A todos os amigos que acumulei durante esses longos anos de graduação. Tive a incrível sorte de conhecer pessoas incríveis em cada esquina dessa universidade, as quais seria impossível citar todos neste pequeno texto. Certamente cada um que compartilhou algum momento feliz ou triste comigo durante esses anos foram fundamentais para eu chegar até aqui.

Aos projetos extracurriculares em que tiver o prazer de contribuir e aprender: A CPE e ao movimento empresa júnior, pois ser júnior no Brasil é ser gigante pela própria natureza. Ao GEES e ao MUDA UFMG por toda luta em prol do estudante, da ciência e defesa pela universidade pública. A Bateria Engrenada, Grifo e Atletismo por todos ensinamentos, glórias e sofrimentos que o esporte e a música podem proporcionar.

Aos mestres e professores que me guiaram nessa difícil jornada e que trabalham incansavelmente para contribuir com a ciência e com a melhoria da educação no nosso país. Em especial ao professor Frederico Coelho pela orientação neste trabalho e ao professor Frederico Gadelha por fazer parte da banca avaliadora.

Às instituições que me deram uma oportunidade e me fizeram crescer como profissional de engenharia: Unitec, Neocontrol, Órbita e Instituto Eldorado.

A Prof. Ana Liddy e ao servidor Julio Carvalho pela prestatividade em atender e auxiliar cada aluno de Engenharia de Sistemas.

Por fim agradeço a mim mesmo, pela dedicação, por não ter desistido, pelas noites mal dormidas para entregar o melhor trabalho possível e por ter corrido atrás da melhor formação.

"The desire that guides me in all I do is the
desire to harness the forces of nature
to the service of mankind." (Nikola Tesla)

RESUMO

No Brasil, cerca de 10,7 milhões de pessoas possuem algum grau de perda auditiva e enfrentam em sua rotina problemas relacionados à comunicação, já que existe uma reduzida quantidade de pessoas alfabetizadas em Libras. Com o avanço de tecnologias de microcontroladores, aprendizado de máquina e desenvolvimento de software, emergem sistemas que permitem a tradução de línguas de sinais em línguas faladas. Neste contexto, este trabalho apresenta o desenvolvimento de um sistema capaz de solucionar o problema de reconhecimento de gestos da Língua Brasileira de Sinais (Libras). Primeiramente foi elaborada uma revisão sobre o estado da arte em sistemas de reconhecimento de gestos em linguagem de sinais. Foi desenvolvido um protótipo microcontrolado composto por um sensor inercial e sensores de flexão para coleta de dados e, para a tarefa de classificação dos sinais, são explorados modelos de redes neurais artificiais baseados na arquitetura LSTM. São coletados e analisados 11 gestos em Libras além de um gesto neutro, executados por um voluntário, obtendo 50 amostras por gesto. São comparados 2 modelos baseados em redes neurais LSTM, uma rede neural LSTM de única camada e uma rede neural que combina uma camada convolucional à camada LSTM, o modelo ConvLSTM. Também foi comparado um modelo de aprendizado de máquina não baseado em aprendizado profundo. Obteve-se 97,3% de acurácia geral através do modelo ConvLSTM. Além disso, foi examinada a acurácia dos métodos propostos considerando apenas dados do sensor inercial e dos sensores de flexão. Ao final, este estudo também faz análise sobre a implementação de algoritmos baseados em redes neurais em sistemas embarcados. Concluiu-se que redes neurais LSTM são capazes de resolver a tarefa de classificação de gestos em Libras e que o protótipo desenvolvido é valioso para a comunidade surda.

Palavras-chave: Reconhecimento de Gestos, Reconhecimento de Linguagem de Sinais, Sistemas Embarcados, Aprendizado de Máquina Embarcado, Tecnologias Assistivas, Língua Brasileira de Sinais.

ABSTRACT

In Brazil, about 10.7 million people have some degree of hearing loss and face problems related to communication in their routine, since there is a small number of people literate in Libras. Due to the evolution of microcontroller technologies, machine learning and software development, solutions emerge to allow the translation of sign languages into spoken languages. In this context, this work presents the development of a system capable of solving the problem of gesture recognition in Brazilian Sign Language (Libras). First, a review of the state of the art in gesture recognition systems in sign language was elaborated. A embedded system composed of an inertial sensor and bending sensors was developed for data collection and, for the gesture classification task, models of artificial neural networks based on the LSTM architecture are explored. 11 gestures in Libras are collected and analysed in addition to a neutral gesture, performed by a volunteer, obtaining 50 samples per gesture. Two models based on LSTM neural networks, a single-layer LSTM neural network and a neural network that combines a convolutional layer with the LSTM layer, the ConvLSTM model, are compared. A non-deep-learning machine learning model was also compared. An overall accuracy of 97.3% was obtained using the ConvLSTM model. Furthermore, the accuracy of the proposed methods was examined considering only data from the inertial sensor and the bending sensors. In the end, this study also analyses the implementation of algorithms based on neural networks in embedded systems. It was concluded that LSTM neural networks are able to solve the task of classifying gestures in Libras and that the prototype developed is valuable for the deaf community.

Keywords: Gesture Recognition, Sign Language Recognition, Embedded Systems, Embedded Machine Learning, Assistive Technologies, Brazilian Sign Language.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ACRÔNIMOS

LIBRAS	Língua Brasileira de Sinais
PcD	Pessoas com Deficiência
MCU	Microcontroladores
TA	Tecnologia Assistiva
IBGE	Instituto Brasileiro de Geografia e Estatística
EMG	Eletromiografia
ECG	Eletrocardiografia
CI	Circuito Integrado
IoT	*Internet of Things* (do inglês Internet das Coisas)
LSTM	*Long short-term memory* (do inglês memória de curto prazo longa)
FCNN	*Fully Convolutional Neural Network* (do inglês redes neurais completamente convolucionais)
RNN	*Recurrent Neural Networks* (do inglês redes neurais recorrentes)
GAN	*Generative Adversarial Network* (do inglês rede adversária generativa)
TCN	*Temporal Convolutional Newtorks* (do inglês redes convolucionais temporais)
MLP	*Multilayer Perceptron* (do inglês redes perceptron de múltiplas camadas)
TinyML	*Tiny Machine Learning* (do inglês aprendizado de máquina reduzido)
USB	Universal Serial Bus (do inglês Barramento Serial Universal)
MEMS	*Micro-Electro-Mechanical Systems* (do inglês sistemas microeletromecânicos)
KNN	*K Nearest Neighbors* *(*do inglês k Vizinhos mais Próximos*)*
UART	*Universal asynchronous receiver/transmitter* (do inglês Transmissor/receptor assíncrono universal)
RAM	*Random Access Memory* *(*do inglês Memória de acesso aleatório*)*

SUMÁRIO

1 INTRODUÇÃO

De acordo com o censo demográfico de 2010 do Instituto Brasileiro de Geografia e Estatística, a deficiência auditiva atinge 10,7 milhões de pessoas em algum grau, cerca de 5% da população brasileira (IBGE, 2010). (MEC, 2006, p. 19) define a surdez como “perda maior ou menor da percepção normal dos sons”. No Brasil, a Libras foi oficializada como linguagem de comunicação e expressão em 2002 pela lei nº 10.436, exigindo que escolas, faculdades e instituições públicas providenciem intérpretes de Libras para atender pessoas com deficiência auditiva no intuito de estimular a inclusão dos surdos na sociedade (BRASIL, 2002). Porém, esta lei não foi suficiente para solucionar um grande obstáculo para inclusão de PcD auditiva, que é a escassez de ouvintes em Libras, ou pessoas auditivamente saudáveis e capazes de se comunicar na língua de sinais. Para (WESTIN, 2019), a falta de ouvintes e “falantes” da Libras acaba criando ambiente de isolamento na sociedade.

Em 2017, a Libras foi catalogada com 14.500 sinais, e, assim como a linguagem oral, a linguagem de sinais apresenta grande complexidade léxica e semântica (CAPOVILLA, RAPHAEL, et al., 2017). Por exemplo, uma mesma língua de sinais, em uma determinada região, pode apresentar várias diferenças dialetais, bem como diferenças quando expressadas em linguagem metafórica ou sarcástica, o que dificulta ainda mais sua disseminação e aprendizado (DIZEU; CAPORALI, 2005).

A evidente necessidade de interlocução entre pessoas com deficiência (PcD) auditiva e pessoas auditivamente saudáveis cria uma lacuna que justifica o ensino da linguagem de sinais, bem como o desenvolvimento de tecnologias assistivas que facilitem a comunicação de pessoas surdas. Nesse contexto, o decreto de lei nº10.645 de 2021, que dispõe sobre as diretrizes do Plano Nacional de Tecnologia Assistiva, promove um terreno fértil para criação de soluções neste tema (BRASIL, 2021).

As abordagens mais utilizadas em sistemas automáticos de reconhecimento de gestos de mãos em linguagem de sinais são as baseadas em visão computacional ou sensores inerciais. As técnicas baseadas em visão distinguem-se entre as não invasivas, que utilizam uma ou mais câmeras para detecção de regiões de interesse, e as técnicas invasivas, que utilizam marcadores ou luvas para identificação de

movimentos específicos. Considerando as técnicas que utilizam sensores, destacam-se duas categorias: Aquelas que utilizam sensores de medida inercial, como acelerômetro e/ou giroscópio e os que utilizam de dados sensores eletromiográfico (EMG), capazes de detectar sinais de contração muscular, para identificar movimentos nos dedos e nos pulsos (CHEOK; OMAR; JAWARD, 2019).

Devido à complexidade visual dos gestos em linguagem de sinais, grande parte das abordagens de reconhecimento de linguagem de sinais atualmente utiliza técnicas de visão computacional. Porém, essa abordagem pode sofrer por problemas de aquisição de imagem como a obstrução visual de gestos. Além disso, técnicas de aprendizado de máquina para imagens em movimento exigem grande disponibilidade de recursos computacionais para seu processamento. Neste sentido, as abordagens que utilizam de sensores inerciais podem atuar como sistema auxiliar, sendo uma alternativa de rápido processamento e boa acurácia (ER-RADY *et al.*, 2017).

Considerando a discussão levantada, este trabalho tenta responder a seguinte problemática: Como a tecnologia pode ajudar a superar as dificuldades de comunicação entre comunidades com e sem deficiência auditiva?

Algumas soluções tecnológicas já existentes superam em partes o problema da conversão da comunicação escrita ou falada para a linguagem de sinais. Essas soluções serão discutidas mais detalhadamente no capítulo 2 deste trabalho.

1.1 OBJETIVOS

Tendo em vista o problema apresentado, o objetivo principal deste trabalho é avaliar a implementação de algoritmos de aprendizado de máquina na tarefa de reconhecimento de padrões em sinais de comunicação em Libras, utilizando dados de sensores inerciais e de flexão. Considerando a motivação apresentada, os objetivos específicos para desenvolvimento do trabalho proposto são:

- Entender o estado da arte de sistemas de reconhecimento de linguagem de sinais usando tecnologias de aprendizado de máquina;
- Caracterizar os métodos de aprendizado de máquina para classificação de séries temporais;

- Obter, analisar e preparar os dados sensoriais de series temporais em Libras;
- Avaliar a aplicação de modelo de redes neurais artificiais do tipo LSTM para classificação dos dados coletados;
- Analisar a implementação do modelo desenvolvido em sistema embarcado microcontrolado.

1.2 ESTRUTURA DO TRABALHO

Este trabalho será apresentado em 6 capítulos. Primeiramente, a introdução apresenta a motivação, a justificativa e os objetivos do problema a ser resolvido. No capítulo 2 será revisado o estado da arte no tema e os conceitos que serão utilizados no desenvolvimento a seguir. O capítulo 3 apresentará a metodologia e as etapas propostas para alcançar os objetivos propostos. O capítulo 4 mostrará o procedimento de desenvolvimento do *hardware* e *software* que compõem este estudo enquanto no capítulo 5 é feita uma análise da performance dos algoritmos implementados e uma análise sobre a implementação destes algoritmos em sistemas embarcados. Por fim, o capítulo 6 traz uma conclusão sobre os temas discorridos e propostas de melhorias para trabalhos futuros.

2 FUNDAMENTOS TEÓRICOS

2.1 LINGUAGEM DE SINAIS NO BRASIL

De acordo com (EBERHARD; SIMONS; FENNIG, 2021) existem 150 línguas de sinais no mundo. Essas línguas são formadas de acordo com os aspectos culturais e carregam grande influência das línguas faladas majoritariamente na região onde são criadas. Além disso, podem ser influenciadas por outras línguas de sinais dominantes (JEPSEN *et al.*, 2015).

Segundo (PEREIRA, 2011), os gestos em Libras são formados a partir da combinação de cinco parâmetros fonológicos: configuração de mãos, localização, movimento, orientação da palma das mãos e traços não manuais. A configuração de mãos se trata da forma que a mão assume durante a execução dos sinais. A localização é o lugar espacial onde o sinal é formado, geralmente feitos na testa ou em frente ao corpo. O movimento se refere ao percurso realizado pelos dedos, pulso ou antebraço. A orientação da palma da mão indica a direção em que a palma da mão está apontada durante o gesto. Por último, os traços não manuais incluem expressão facial, movimento corporal e olhar do interlocutor.

Os parâmetros são considerados traços distintivos em Libras em que a mudança de apenas um parâmetro durante a execução de um sinal pode alterar completamente o sentido da palavra que se deseja reproduzir.

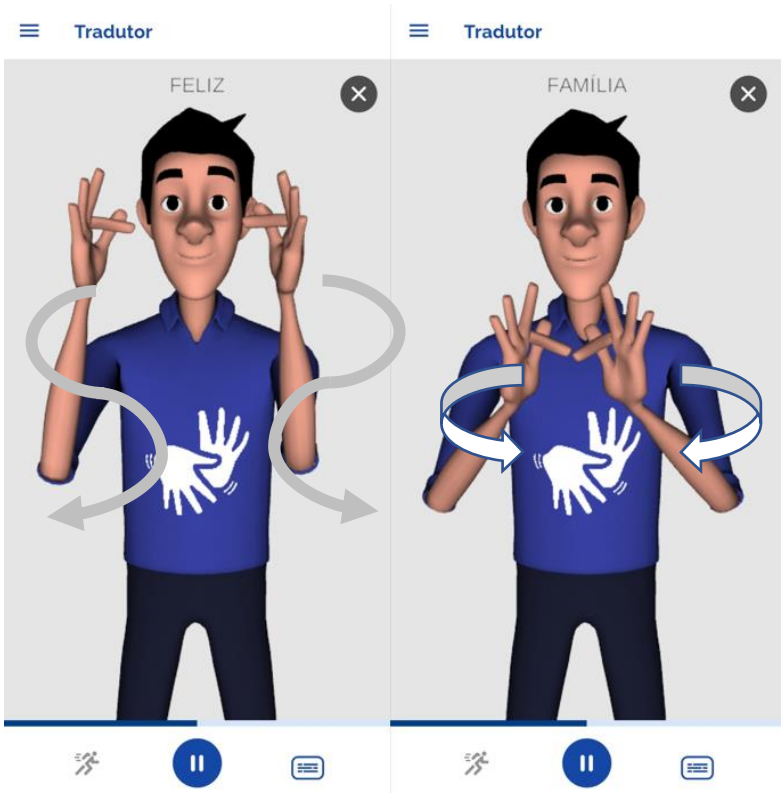

Figura 1 - Representação das palavras 'FELIZ' e 'FAMÍLIA' no *software* VLibras
Fonte: Autoria própria com auxílio de (VLIBRAS, 2010).

A Figura 1 mostra um exemplo de duas palavras que utilizam a mesma configuração de mão, porém com movimento e localização diferentes. A palavra ‘Feliz’

realiza movimento de ondulatório em descida e a palavra ‘Família’ é produzida realizando uma rotação no pulso.

Os gestos em Libras ainda podem ser classificados como estáticos e dinâmicos. Os gestos dinâmicos são aqueles que apresentam componente de movimento em sua execução. Segundo (JEPSEN *et al.*, 2015), 91% dos sinais em Libras são dinâmicos e apenas 9% são estáticos. A Libras ainda apresenta aspectos morfológicos. Palavras mais complexas formadas derivadas de outras palavras, seja por uma combinação de gestos ou pela inclusão de um parâmetro diferente (PEREIRA, 2011).

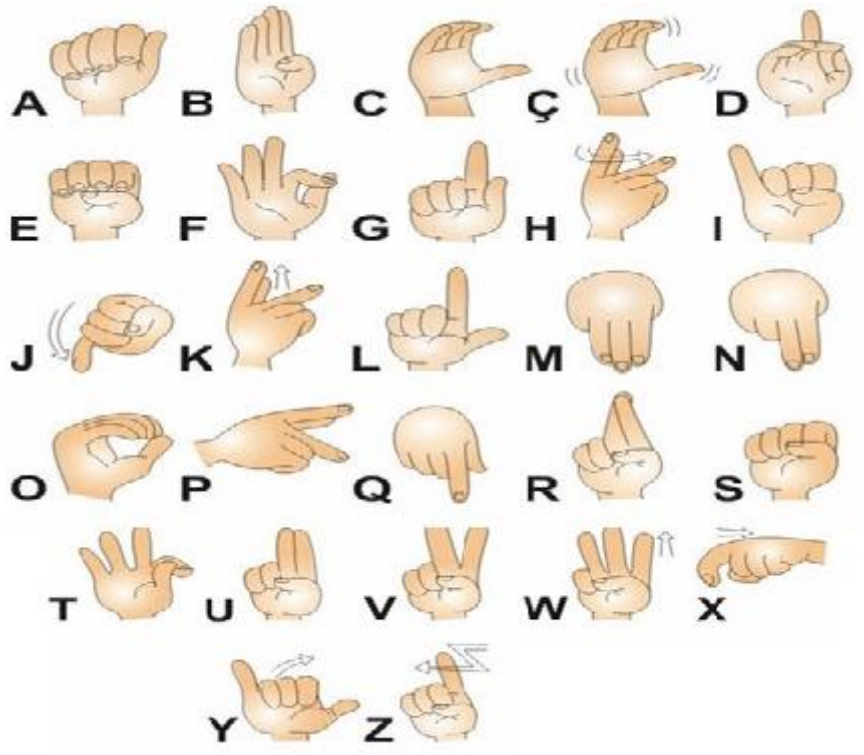

Figura 2 - Alfabeto manual da Libras, composto por 20 sinais estáticos e 6 dinâmicos.
Fonte: (SECCO; BARROS; HERVERTON, 2009).

Neste trabalho procura-se identificar apenas palavras que possam ser diferenciadas através da configuração de mãos, orientação da palma da mão e movimento. Além disso, evita-se palavras que utilizem ambas as mãos para configuração de mãos.

2.2 APRENDIZADO DE MÁQUINA

Aprendizado de máquina é uma subárea do campo de pesquisa de Inteligência Artificial, que é definido como a capacidade de uma máquina em imitar o comportamento inteligente. Algoritmos de aprendizado de máquina são aqueles que adquirem a capacidade de realizar uma tarefa sem serem explicitamente programados para aquela finalidade (GOODFELLOW; BENGIO; COURVILLE, 2016). Algoritmos de aprendizado de máquina podem ser classificados pelo tipo de aprendizagem como aprendizado supervisionado, aprendizado não-supervisionados e aprendizado por reforço. Dentre os algoritmos de aprendizado de máquina clássicos mais conhecidos pode-se citar árvores de decisão, *Naive Bayes,* regressão logística, máquina de vetores de suporte (SVM) e k-vizinhos mais próximos (KNN). Incluída na categoria de aprendizado de máquina estão os algoritmos de Aprendizado Profundo (*Deep Learning*), que utilizam em sua construção unidades chamadas de neurônios artificiais organizados em redes de várias camadas.

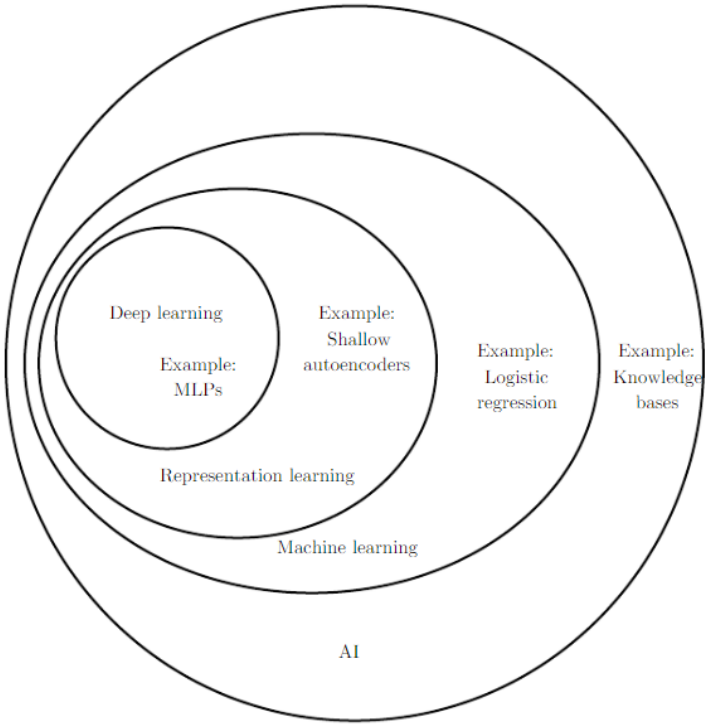

Figura 3 - Diagrama de Venn mostrando a relação entre as categorias de algoritmos de inteligência artificial
Fonte: (GOODFELLOW; BENGIO; COURVILLE, 2016)

2.2.1 Series Temporais

De acordo com (PINHEIRO, 2021), séries temporais são conjuntos de observações de uma ou mais variáveis feitas em sequência ao longo do tempo. A principal caraterística é ordem de apresentação dos dados. A partir do estudo das séries temporais podemos resolver problemas como: previsão de valores futuros; detecção de anomalias; ou reconhecimento de padrões.

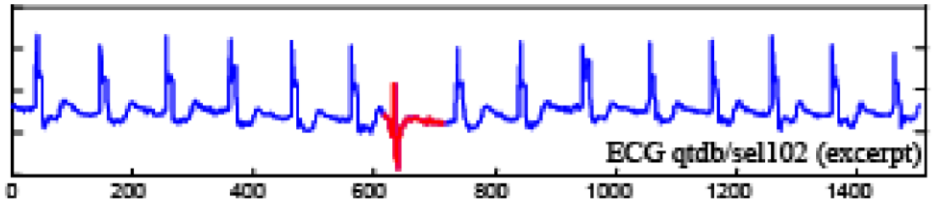

Figura 4 – Dados de exame de eletrocardiograma genérico, um exemplo de série temporal.
Fonte: (CHUAH; FU, [S.d.])

Series temporais podem ser univariadas ou multivariadas. As séries univariadas são aquelas que relacionam apenas uma variável dependente a uma variável independente, geralmente o tempo. Series temporais multivariadas são aquelas que relacionam uma variável independente à várias variáveis dependentes (BLÁZQUEZ-GARCÍA et al., 2021). Este trabalho será desenvolvido considerando series temporais multivariadas. A Figura 5 mostra a diferença dos dois tipos.

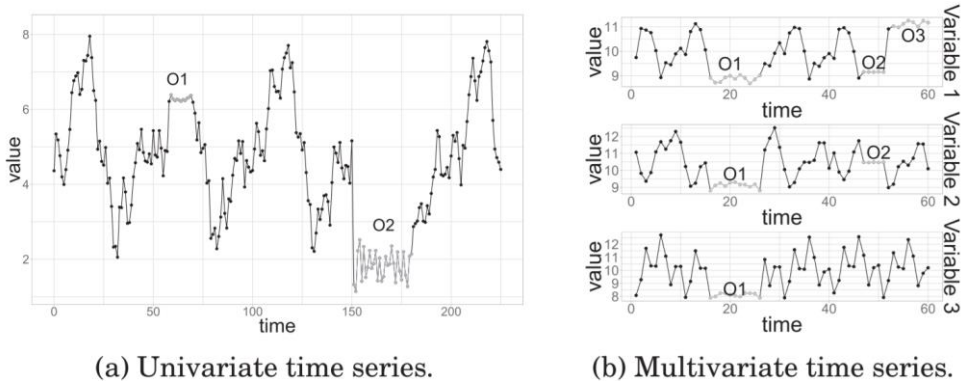

Figura 5 - (a) exemplo de serie temporal univariada e (b) exemplo de serie temporal multivariada
Fonte: (BLÁZQUEZ-GARCÍA *et al.*, 2021)

2.2.2 Redes Neurais Artificiais (RNA)

RNA é uma categoria de modelos de aprendizado de máquina baseada na conexão entre unidades de neurônios artificiais capazes de adaptar seus pesos e realizar tarefas como classificação e regressão. Seu potencial se deve à sua grande capacidade de generalização (BISHOP, 2006).

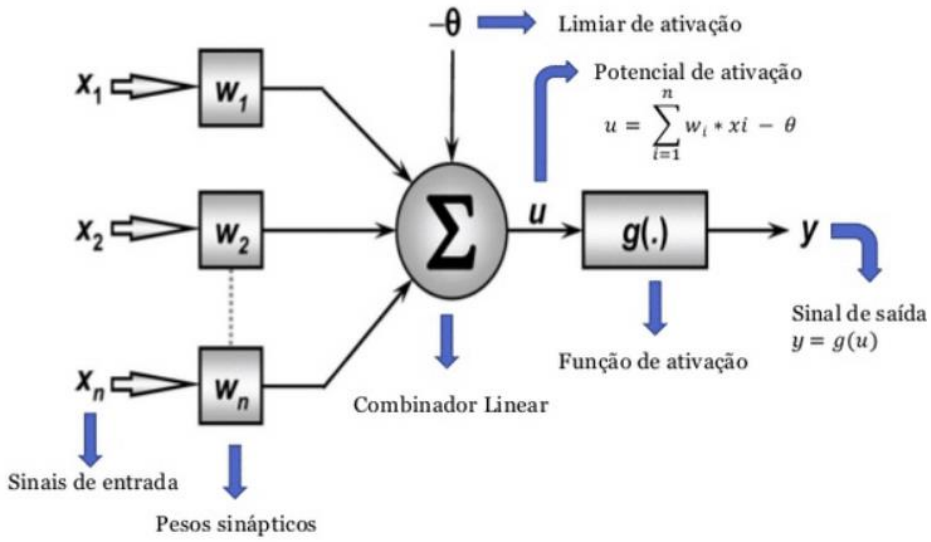

Figura 6 - Representação de neurônio artificial. O sinal de saída é a soma ponderada dos sinais de entrada pelos pesos sinápticos avaliada pela função de avaliação.
Fonte: (BERTOLLO *et al.*, 2020)

Quando isolados, os neurônios artificias possuem pouca utilidade, porém são capazes de resolver diversos problemas da literatura específica quando agrupados em redes *Perceptron* de Múltiplas Camadas (MLP). Pode-se dizer que MLPs são aproximadores universais de funções (GOODFELLOW; BENGIO; COURVILLE, 2016). Quando agrupadas em muitas camadas, essas redes denominam-se redes neurais profundas (DNN).

Os neurônios artificiais são capazes de reter conhecimento em seus pesos sinápticos através do processo de aprendizagem. Este processo é realizado pelo algoritmo de propagação reversa (*backpropagation*), em que a rede calcula novos pesos sinápticos a partir do erro entre o sinal de saída previsto e o sinal observado para aquela iteração (BERTOLLO *et al.*, 2020).

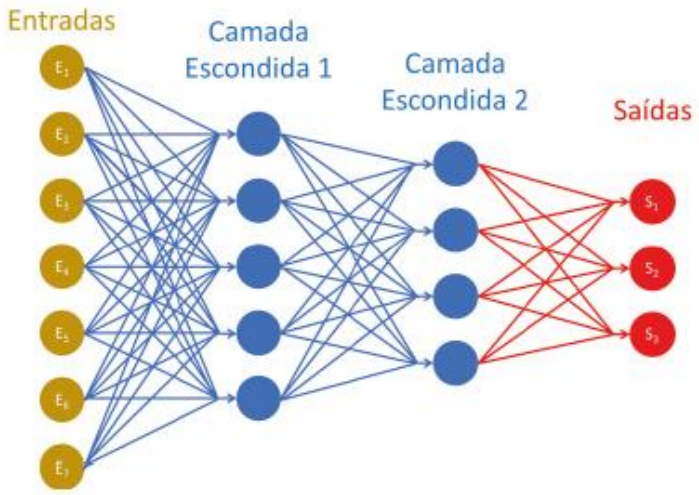

Figura 7 - Representação de uma rede neural de multiplas camadas
Fonte: (OLIVEIRA, 2018)

Assim como outros algoritmos de aprendizado de máquina, as RNA possuem uma etapa de treinamento e uma etapa de validação. Na etapa de treinamento, a saída observada no conjunto de dados é utilizada para cálculo do erro de predição que será utilizado no algoritmo de propagação reversa. Na etapa de validação, a saída observada é utilizada para avaliação da performance do modelo (OLIVEIRA, 2018). Geralmente, RNA dependem de um grande volume de dados para um treinamento eficiente do modelo. Além disso, o algoritmo de propagação reversa tem alto custo computacional e são sensíveis a hiperparâmetros. Essas características fazem com que a aplicação de RNA seja restritiva na solução de alguns problemas.

As funções de ativação são responsáveis por criar uma superfície de decisão nos neurônios artificias, determinando se a informação fornecida na entrada do neurônio é relevante para acionar a próxima camada ou não. Geralmente essas funções introduzem uma não linearidade no sistema. As funções de ativação mais comuns são as: *sigmóide,* tangente hiperbólica*, ReLU, Leaky ReLU e Softmax* (DATA SCIENCE ACADEMY, 2021). A Figura 8 mostra o comportamento das funções de ativação relacionadas anteriormente.

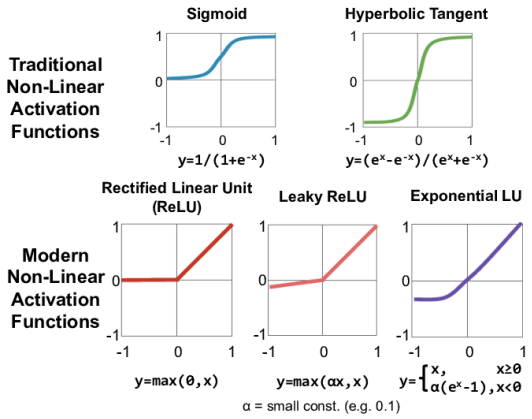

Figura 8 - Funções de ativação mais utilizadas
Fonte: (SZE *et al.*, 2017)

Para se tornarem mais aptas a solucionar problemas específicos, as redes neurais podem se agrupar de diferentes formas. A maneira em que as RNA são estruturadas é chamada de arquitetura. A definição de arquitetura tenta responder às seguintes perguntas: Quantas unidades de neurônios artificias são necessárias? Como essas unidades devem ser conectadas? (GOODFELLOW; BENGIO; COURVILLE, 2016). As classes de arquitetura de redes neurais mais comuns são: redes neurais convolucionais (CNN); redes neurais recorrentes (RNN); redes neurais completamente conectadas (FCNN); *auto-encoders* e redes neurais generativas (GANs) (DATA SCIENCE ACADEMY, 2021). A Figura 9 mostra exemplos de arquiteturas de redes neurais artificiais de múltiplas camadas.

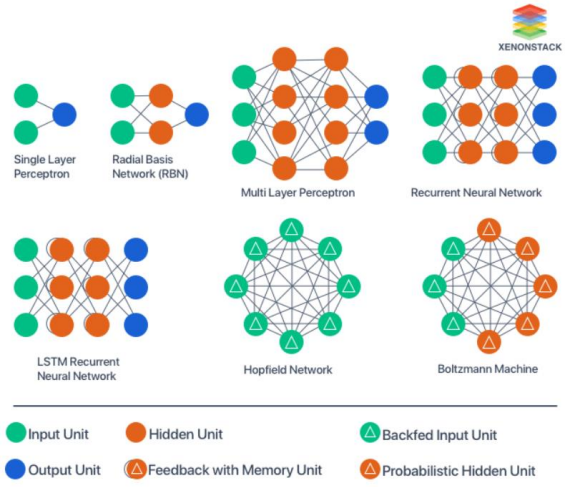

Figura 9 - Exemplo de arquiteturas de redes neurais artificiais
Fonte: (GILL, 2021)

2.2.3 Arquitetura LSTM

As redes neurais LSTM fazem parte de uma classe de RNA conhecida como redes neurais recorrentes (RNN), capazes de processar dados sequenciais (GOODFELLOW; BENGIO; COURVILLE, 2016). Em geral, as vantagens em utilizar as RNN são: a possibilidade de processar entradas de qualquer tamanho; a utilização de informação temporal; seus pesos sinápticos são compartilhados no tempo. Em contra partida temos que seu custo computacional é consideravelmente maior e apresenta dificuldade de acessar informações muito antigas. Apesar da eficiência desse tipo de arquitetura, as RNN simples, mostrada na Figura 10(a) sofrem do problema conhecido como dissipação do gradiente. Como discutido no capítulo 2.2.2 as RNN também utilizam o algoritmo de propagação reversa para treinamento do modelo. Quando a RNN processa uma sequência muito longa, após algum tempo seus neurônios artificiais param de processar os dados devido ao valor do gradiente ser quase nulo (DATA SCIENCE ACADEMY, 2021). Para solucionar esse problema as redes neurais LSTM introduzem em cada célula o conceito de portas lógicas e estado da célula. Esses elementos são responsáveis por controlar o fluxo de informação pela rede e selecionar quando a informação deve ser retida e quando ela deve ser esquecida. A Figura 10(b) mostra uma comparação da célula simples de RNN e da célula de uma rede LSTM.

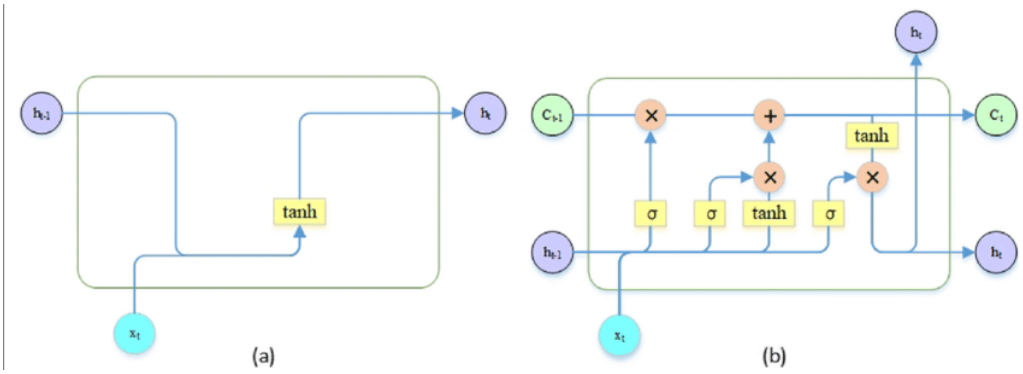

Figura 10 - Representação de uma célula em uma rede neural recorrente simples (a) e em uma rede neural LSTM
Fonte: (ZHU *et al.*, 2019)

As RNN podem ser organizadas em alguns tipos dependendo da aplicação que será utilizada. A Figura 11 mostra alguns dos tipos utilizados na literatura específica. Em problemas de classificação normalmente é utilizado o tipo *many-to-one*, em que, dado uma sequência de características, deseja-se prever apenas um atributo: a classe.

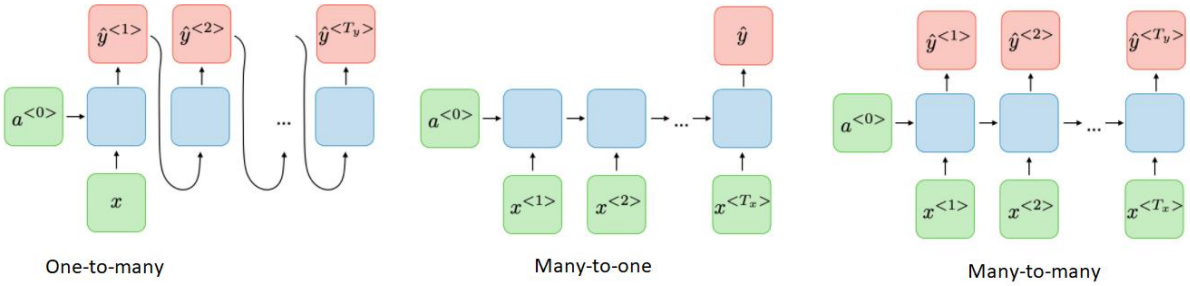

Figura 11 - Diferentes configurações de redes neurais recursivas
Fonte: (AMIDI; AMIDI, 2018)

2.2.4 Classificação de Séries Temporais

Problemas de classificação de series temporais são aqueles nas quais é possível extrair características de uma série temporal para identificá-las como parte de um entre vários grupos pré-definidos, usando dados já conhecidos para treinamento (ISMAIL FAWAZ *et al.*, 2019). Como exemplo de aplicação em casos univariáveis é a identificação de doenças de coração utilizando dados de eletrocardiograma (ECG) (KANANI; PADOLE, 2020).

Existem algumas formas de abordar este tipo de problema em casos que necessitam de múltiplas variáveis. Este capítulo explora alguns métodos mais comuns presentes na literatura específica, focando em métodos que utilizam de Redes Neurais Profundas e mais especificamente LSTM. A Figura 12 mostra um processo genérico para entender o processo de classificação de series temporais de múltiplas variáveis. O problema torna-se cada vez mais complexo dependendo da frequência de amostragem da série temporal e da quantidade de dimensões. Por isso observa-se uma grande preocupação na literatura específica em utilizar métodos que reduzem a dimensionalidade do problema (ISMAIL FAWAZ *et al.*, 2019).

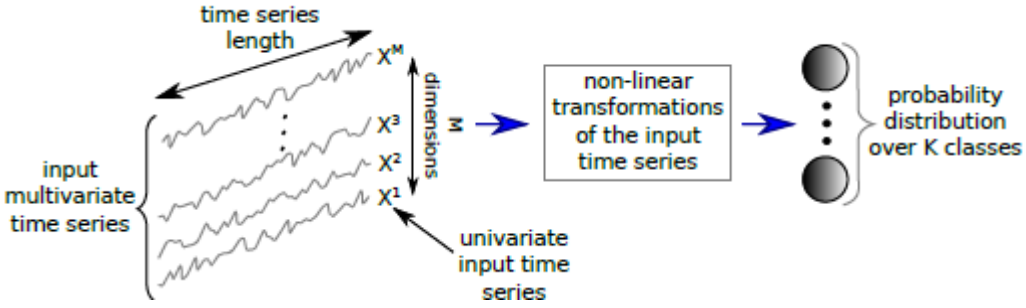

Figura 12 - Processo genérico para classificação de séries temporais
Fonte: (ISMAIL FAWAZ *et al.*, 2019)

Uma implementação usando a redes neurais LSTM é apresentada em (KARIM *et al.*, 2019). O autor apresenta uma arquitetura que combina a classificação feita por uma componente LSTM e uma componente FCN, como mostrado na Figura 13**.**

(KARIM *et al.*, 2018) argumenta que a operação de embaralhamento de dimensão (*dimension shuffle*) pode aumentar a eficiência do modelo quando o número de dimensões (M) é menor que amostras no tempo (Q). Esta operação transpõe a matriz de entrada da rede neural de tamanho QxM para uma matriz de MxQ. Isso reduz a complexidade da rede e reduz o tempo computacional para treino e inferência de amostras.

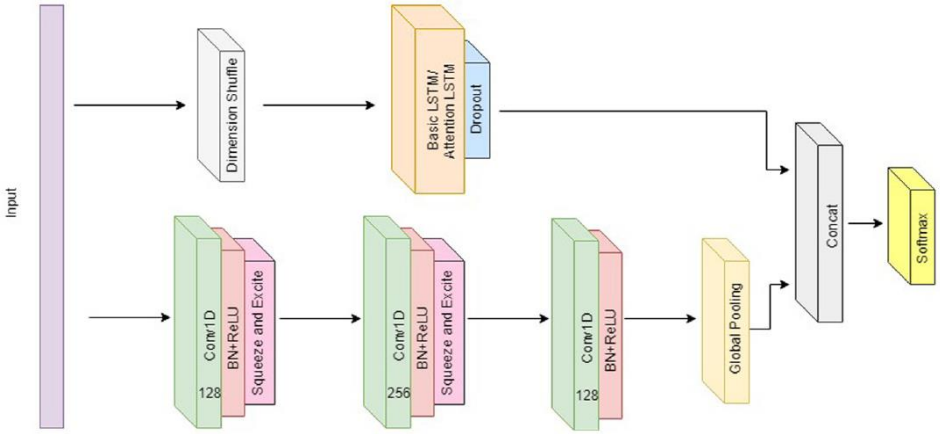

Figura 13 - Arquitetura LSTM combinada com uma arquitetura FCN para solução do problema ade classificação de séries temporais
Fonte: (KARIM *et al.*, 2019)

(SHI *et al.*, 2015) propõe a arquitetura ConvLSTM, que combina uma camada convolucional as camadas LSTM, que capturam melhor as correlações espaço-temporais. A camada convolucional teria o papel de codificar as características de

entrada do modelo e facilitar a interpretação temporal da camada LSTM. Este algoritmo tem boa acurácia em problemas de séries temporais com múltiplas variáveis.

Uma das possíveis soluções para reduzir o tamanho da matriz de entrada da rede neural e consequentemente aumentar a eficiência da solução é utilizar uma camada convolucional dilatada. Esta solução é especialmente útil para series temporais pois mantém a ordem dos dados. Por não dependerem de conexões recorrentes, são mais eficientes que as camadas das redes LSTM.

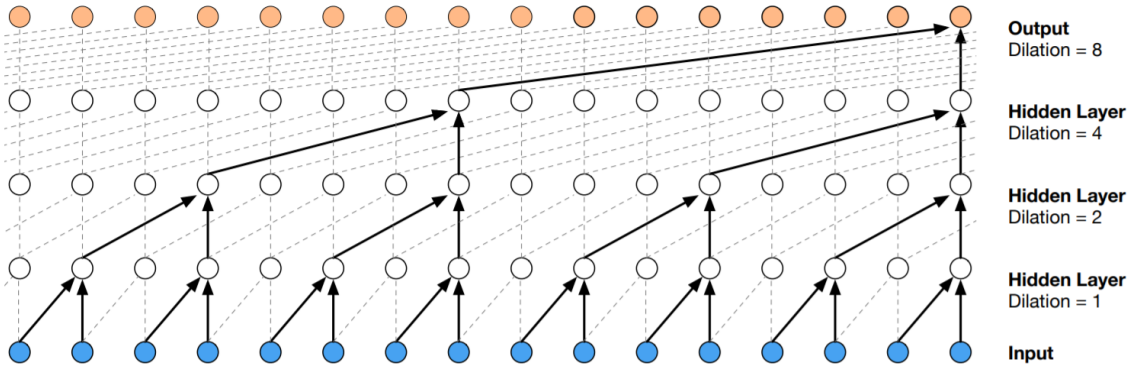

Figura 14 - Representação de uma camada convolucional dilatada
Fonte: (OORD *et al.*, 2016)

As camadas convolucionais dilatadas são a base para as redes convolucionais temporais (TCN), que podem ser utilizadas em conjunto com classificadores baseados em MLP ou outros tipos de classificadores. As redes convolucionais temporais (TCN) são utilizados na literatura em problemas de séries temporais envolvendo muitas dimensões, como vídeos, dados médicos e de IoT (LIN *et al.*, 2019). Essas redes ainda podem ser utilizadas em problemas de predição e detecção de anomalias (BAI; KOLTER; KOLTUN, 2018).

2.3 SISTEMAS EMBARCADOS

De acordo com (ALMEIDA; MORAES; SERAPHIM, 2016), sistemas embarcados são sistemas eletrônicos compostos de um ou mais microcontroladores e periféricos que possuem uma função específica que geralmente não pode ser alterada. Como mostrado pela Figura 15, os sistemas embarcados são compostos por microcontroladores conectados a periféricos e subsistemas, podendo ser de comunicação, sensores, atuadores ou de interface. Como exemplo, impressoras mesmo possuindo um processador que poderia ser usado para fins gerais, tem funcionalidade restrita a atividade de impressão. Uma das características de sistemas embarcado é a restrição de alguns recursos como:

- Físicos: limitação de espaço, interfaces e peso;
- Elétricos: limitação no consumo de energia;
- Computacionais: limitação na capacidade de memória e processamento;
- Mecânicos: temperatura de operação;
- Temporais: tempo de realização de tarefas.

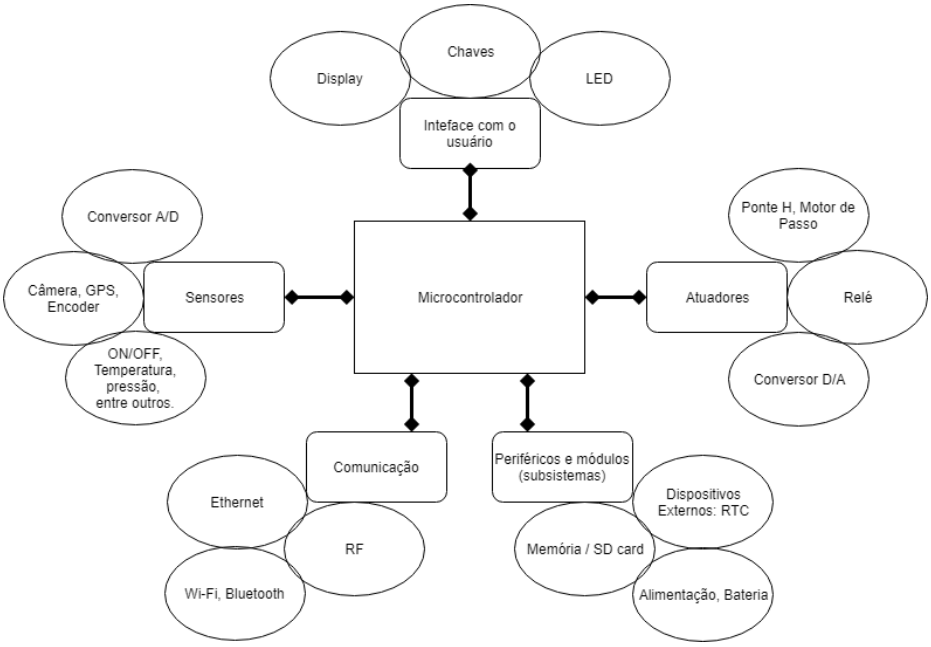

Figura 15 - Exemplo de representação de um sistema embarcado
Fonte: (FERNANDO DELUNO GARCIA, 2018)

Considerados os cérebros dos sistemas embarcados, os microcontroladores são sistemas computadorizados capazes de processar informações complexas, armazenar dados e gerenciar periféricos em um único circuito integrado. Como mostra a Figura 16, um microcontrolador é composto principalmente por uma unidade de processamento central (CPU), uma memória volátil (RAM), uma memória não volátil (ROM), barramentos de comunicação, controlador de energia entre outros componentes periféricos que podem variar a depender do modelo e fabricante (FERNANDO DELUNO GARCIA, 2018).

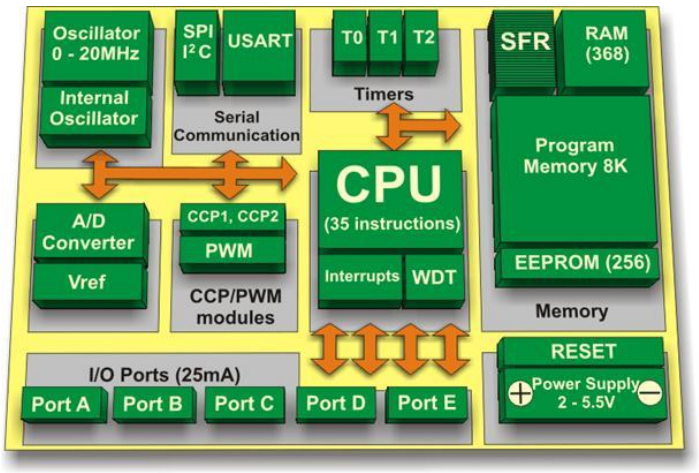

Figura 16 - Arquitetura do microcontrolador PIC16F887. Exemplo de arquitetura de microcontroladores.
Fonte: (SCRIGROUP, 2007)

Esses microcontroladores podem ser programados para tarefas específicas em instruções primárias chamadas de linguagem de máquina. Devido às restrições de recursos discutidas anteriormente, é necessário que os programas armazenados nos microcontroladores sejam otimizados para consumir a menor quantidade de recursos computacionais possível. Em baixo nível, os programas dos microcontroladores são representados em linguagem de montagem, ou Assembly, para facilitar otimizações. Utilizando um compilador, pode-se programar os microcontroladores em linguagem de programação de alto nível, geralmente C/C++ (ALMEIDA; MORAES; SERAPHIM, 2016).

2.3.1 Aprendizado de Máquina Embarcado

Em geral, na solução de um problema envolvendo sistemas embarcados, quando as restrições de *hardware* impedem a implementação de um *software* de alta complexidade, os engenheiros envolvidos no projeto recorrem a duas soluções: reduzir a complexidade do *software* sob a penalização de uma performance pior ou exportar os dados para uma plataforma computacional externa com maior poder computacional, normalmente soluções em nuvem (ALMEIDA; MORAES; SERAPHIM, 2016).

Ultimamente, observa-se grande foco na literatura para melhorar as tecnologias aplicadas em ambientes com recurso limitado. Neste sentido, aplicações para MCUs tem ganhado grande atenção devido à sua importância na demanda crescente por dispositivos de Internet das Coisas (IoT) mais eficientes seguros e confiáveis (DUTTA; BHARALI, 2021).

TinyML é um conceito recente que trata da otimização de modelos de aprendizado de máquina implementados em MCUs de baixo consumo (BANBURY *et al.*, 2020). As literaturas específicas que dedicam-se a desenvolver o *TinyML* propõem o desenvolvimento de hardwares, algoritmos e softwares que permitam a aplicação de métodos de aprendizado de máquina em sistemas embarcados.

Para (DUTTA; BHARALI, 2021), entre os benefícios do *TinyML* para sistemas IoT estão:

- Redução da latência em sistemas críticos;
- Redução de custo, uma vez que a computação em nuvem tem alto custo financeiro;
- Aumento da eficiência energética;
- Aumento da confiabilidade e segurança do dispositivo;
- Aumento da autonomia de sistemas de IoT.

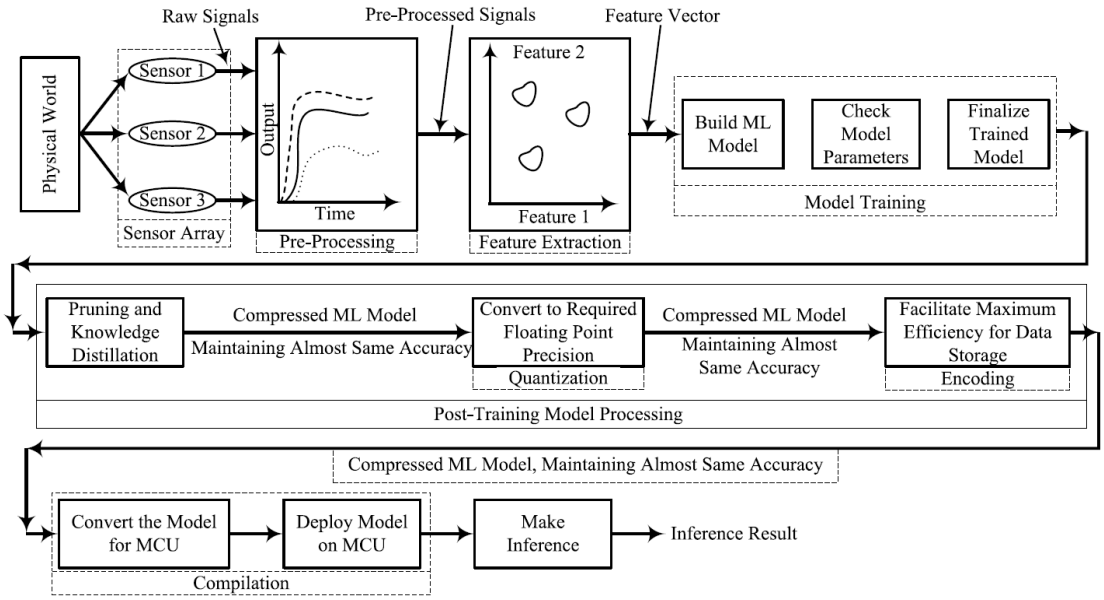

Figura 17 - Processo de treinamento e predição de sistema de aprendizado de máquina embarcado
Fonte: (DUTTA; BHARALI, 2021)

Importante observar que, apesar dos sistemas que utilizam *TinyML* executarem predições nos dispositivos de borda, o treinamento dos modelos de aprendizado de máquina ocorre, em sua maioria, em máquinas com maior capacidade computacional. É dito que o treinamento ocorre de forma *'offline'*. Geralmente esses sistemas possuem um link de conexão com dispositivos de computação em nuvem com poder computacional suficiente para realizar a etapa de treinamento dos modelos (DAVID et al., 2020).

A escolha de dispositivo para implementação de um modelo compatível com *TinyML* depende de diversos fatores como: preço, consumo de energia, tamanho de memória e performance. Para cada projeto pode existir um diferente microcontrolador ideal.

Os próximos capítulos tratarão dos sensores a serem utilizados no sistema embarcado proposto na solução final deste trabalho.

2.3.2 Sensores Inerciais

Para identificação dos gestos de mão dos sinais em Libras, é necessário o uso de sensores eletrônicos capazes de fornecer características capazes de diferenciar os padrões em cada um dos parâmetros que compõem os gestos em Libras. Sensores inerciais são desenvolvidos a partir da tecnologia MEMS. Esta tecnologia origina-se da integração da combinação de conceitos de mecânica e eletrônica para criar dispositivos capazes de detectar mudanças físicas no meio. Sensores desenvolvidos a partir de tecnologia MEMS comportam-se como transdutores, convertendo sinais mecânicos em sinais eletrônicos (SAFFO, 1997).

Os sensores inerciais mais comuns no mercado são os acelerômetros e os giroscópios. O sensor acelerômetro é sensível às forças aplicadas, sendo capaz de medir a aceleração de um objeto a partir da segunda lei de Newton, dada pela equação **Erro! Fonte de referência não encontrada.**, onde *F* é a força aplicada (N), m é a massa do objeto (Kg) e a é a aceleração (m/s^2): (BALBINOT; BRUSAMARELLO, 2000)

$$F = m \cdot a \tag{1}$$

O acelerômetro pode medir a aceleração dinâmica e estática de um corpo. A aceleração estática é consequência da projeção da força gravitacional sobre os eixos do sensor, e a aceleração dinâmica é percebida quando o corpo está em movimento (SHAEFFER, 2013). O giroscópio é o sensor utilizado para medir a velocidade angular de rotação do objeto e é utilizado principalmente para medir rotação constante (SHAEFFER, 2013).

Um dos CIs mais utilizados no mercado para soluções de sensoriamento inercial é o MPU-6050. Ele integra 1 acelerômetro e 1 giroscópio em um mesmo dispositivo e possui 6 eixos de sensibilidade, 3 do acelerômetro e 3 do giroscópio (BALBINOT; BRUSAMARELLO, 2000). A Figura 18 mostra a orientação de cada um dos 6 eixos dos sensores.

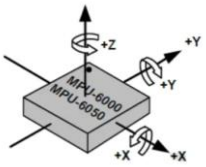

Figura 18 - Orientação dos eixos de sensibilidade e polaridade de rotação do CI MPU-6050.
Fonte: (INVENSENSE, 2013)

No problema a ser resolvido por este trabalho, o acelerômetro é capaz de mensurar as características necessárias para identificação dos parâmetros de orientação da palma da mão, devido a sensibilidade à aceleração estática do corpo, e de movimento, quando percebida uma aceleração dinâmica.

2.3.3 Sensor de flexão

Os sensores de flexão convertem a variação da sua curvatura, por razão da flexão exercida nele, em variação de resistência elétrica. São compostos por poliéster em uma face e tinta de carbono em outra, em que a tinta de carbono é condutiva e possui uma resistência elétrica. A flexão do sensor provoca pequenas rachaduras no substrato contendo a tinta condutiva, aumentando a resistência do material (SAGGIO *et al.*, 2016). A resistência do sensor quando flexionado em 180º é aproximadamente o dobro da resistência do sensor não flexionado (NASCIMENTO *et al.*, 2017). No problema a ser resolvido, os sensores de flexão são úteis para identificação do parâmetro de configuração de mão. Idealmente é utilizado um sensor de flexão para cada dedo, e uma combinação na medição desses sensores é insumo suficiente para caracterização de gestos estáticos (ROCHA SILVA *et al.*, 2017).

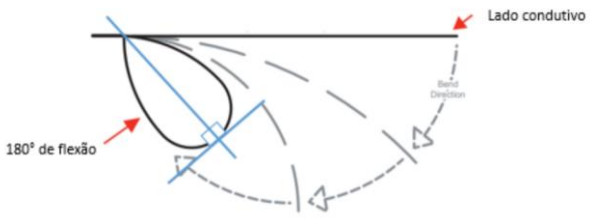

Figura 19 – Representação do sensor de flexão
Fonte: (NASCIMENTO *et al.*, 2017)

2.4 ESTADO DA ARTE NO RECONHECIMENTO AUTOMÁTICO DE LINGUAGEM DE SINAIS

As duas abordagens mais comuns nas pesquisas para reconhecimento de gestos em linguagem de sinais são: a abordagem baseada em imagens, sejam elas estáticas ou em movimento; e as abordagens baseadas em sensores inerciais e de flexão (CHEOK; OMAR; JAWARD, 2019). A Figura 20 apresenta um resumo dos métodos mais comuns para reconhecimento de linguagens de sinais. A abordagem seguida neste trabalho pode ser classificada como luva instrumentada (*Data Gloves*), dentro da categoria de dispositivos baseados em sensores. Neste capítulo são exploradas as principais obras na literatura considerando suas vantagens e desvantagens.

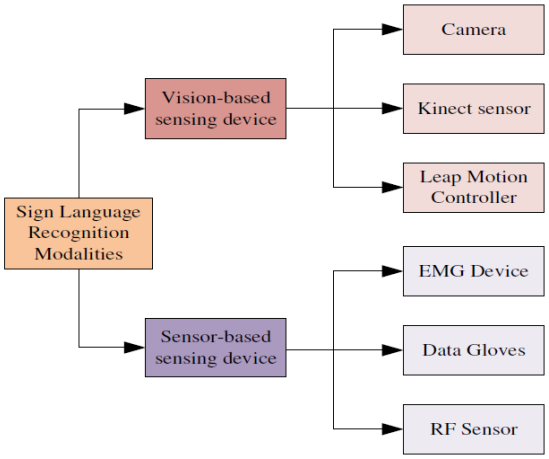

Figura 20 - Modalidades de sistemas de reconhecimento de linguagens de sinais
Fonte: (MADHIARASAN; ROY, 2022)

Dentre as abordagens baseadas em visão computacional para reconhecimento de gestos em linguagens de sinais, é possível destacar alguns estudos. (SAHOO; ARI; GHOSH, 2018) propõe um sistema para reconhecimento de sinais estáticos na língua americana de sinais (ASL) usando a transformada de *wavelet* discreta (DWT) para extrair características da base de dados e posterior classificação. Quanto aos algoritmos de classificação utilizados, (ISLAM *et al.*, 2019) propõe uma configuração baseada na arquitetura de Redes Neurais Convolucionais (CNN) e (MONTEIRO *et al.*, 2016) utiliza um modelo baseado no algoritmo *k*-NN.

Os estudos utilizando câmeras com sensores de profundidade, conhecidos como sensores RGB-D ou Kinect, destacam-se por ter boa acurácia e tornar o sistema mais

robusto. (BESSA CARNEIRO *et al.*, 2016) conta com o auxílio de um sensor RGB-D para classificação de gestos estáticos em Libras, obtendo 89% de acurácia.

Ainda utilizando sensores de profundidade, (REZENDE; ALMEIDA; GUIMARÃES, 2021) propõe a aplicação de classificadores baseados redes neurais convolucionais em um conjunto de dados contendo 20 sinais dinâmicos em Libras gravados por câmera com sensor RGB (2D) e RGB-D (3D). Este estudo obteve acurácia média de 93,3%.

A linguagem de sinais envolve pelo menos três diferentes canais visuais: mãos, face e corpo (ER-RADY *et al.*, 2017). Sistemas baseados em imagem possuem a vantagem de entender o gesto como um todo, separando cada componente do gesto em um subsistema diferente (CHEOK; OMAR; JAWARD, 2019). Ademais, esse tipo de solução é não invasivo e possui menor custo (AHMED *et al.*, 2018). Por outro lado, enfrenta alguns desafios para obter informações importantes das imagens, como: oclusão, distorção e variações na iluminação (GHANEM; CONLY; ATHITSOS, 2017).

Considerando os trabalhos que utilizam sensores inerciais e de flexão, pode-se citar o estudo de (CHU *et al.*, 2020). Este estudo pesquisa o reconhecimento de sequencias gestuais genéricas usando apenas dados de acelerômetro e comparando diferentes configurações da rede neural *PairNet*, que consiste em um tipo de rede neural da categoria das redes neurais convolucionais. Este estudo obteve acurácia média de 92,36%.

O estudo de (ROCHA SILVA *et al.*, 2017) faz uma comparação de diferentes configurações do dispositivo embarcado a fim de descobrir o posicionamento dos sensores que traz melhores resultados. Os autores comparam 3 configurações de sensores e diferentes modelos de redes neurais para classificação de letras em Libras, obtendo 95,2% de acurácia no melhor caso.

Já (SHAHEEN; MEHMOOD, 2018) discutem uma proposta de luvas para reconhecimento de gestos na língua paquistanesa de sinais que utiliza ambas as mãos para obter resultados mais precisos. Este estudo obteve acurácia de 93,4%.

Dentre os trabalhos que baseiam-se em sensores inerciais e sistemas embarcados, grande parte destinam-se a solucionar o problema de reconhecimento de gestos estáticos, porém, (DIAS, 2020) propõe um procedimento para identificação de métodos dinâmicos baseado em redes *Perceptron* de Múltiplas Camadas (MLP), utilizando uma luva instrumentada com sensores de flexão, de contato e inerciais. Este estudo obteve uma acurácia de 98,96%.

Além dos estudos citados, existe uma solução comercial capaz de rastrear movimentos de mão e registrá-los em ambiente virtualizado (CYBERGLOVE SYSTEMS, 2017). Este produto possui de 18 a 22 sensores para captura de movimentos e conexão sem fio. Possui aplicação direcionada a sistemas de animação digital e sistemas industriais. Uma imagem do produto é mostrada na Figura 21. Observa-se que é uma solução completa para extração de características para desenvolvimento do modelo classificador, porém possui alto custo financeiro.

Figura 21 - Produto *CyberGlove*. Sistema para detecção de movimentos nas mãos
Fonte: (CYBERGLOVE SYSTEMS, 2017)

Os métodos baseados em sensores têm o benefício de não depender de alto poder de processamento. Por isso são interessantes para implementação embarcada. Porém possuem custo superior em relação aos métodos baseados em visão computacional, já que dependem do desenvolvimento de *hardware* dedicado. (GHANEM; CONLY; ATHITSOS, 2017). A Tabela 1 resume as principais vantagens e desvantagens de cada método.

Método	Baseado em visão computacional	Baseado em sensores
Dispositivo de Captura	Camera 2D (RGB) ou 3D (RGB-D)	Sensores Inerciais, de flexão, de movimento ou EMG
Custo	Baixo custo de desenvolvimento	Alto custo de desenvolvimento (protótipos e sensores)
Eficiência	Baixa (exige maior capacidade de processamento)	Alta (exige menor capacidade de processamento)
Obstáculos	Iluminação, oclusão e ruído de cor	Distúrbios do ambiente (movimento inesperado das mãos) e ruídos eletrônicos
Limitações	Desafios referentes a extração de características	Não adequados para aplicações em tempo real
Vantagens	Possibilidade de capturar mais características inerentes ao sinal (informação facial, postura corporal)	Menor complexidade na extração de características. Melhor performance.

Tabela 1 - Comparação entre os métodos de reconhecimento de linguagem de sinais
Fonte: Adaptado de (MADHIARASAN; ROY, 2022)

3 METODOLOGIA

O modelo CRISP-DM, mostrado pela Figura 22, é um dos métodos mais utilizados para estruturar problemas de ciência de dados e aprendizado de máquina (GOSMAR, 2020). O modelo consiste em dividir o desenvolvimento de um trabalho de aprendizado de máquina em seis etapas:

- Entendimento do negócio: entender os objetivos e requisitos do projeto;
- Entendimento dos dados: coletar, descrever e explorar as características dos dados a serem utilizados;
- Preparação dos dados: quaisquer transformações e processamento nos dados para facilitar a utilização dos mesmos nas próximas fases;
- Modelagem: fase em que é selecionado, desenvolvido e treinado o modelo de aprendizado de máquina a ser utilizado pelo sistema;
- Avaliação: testar e avaliar se o modelo desenvolvido atende às expectativas de negócio estabelecidas na primeira fase;
- Implementação: o modelo desenvolvido é implementado no sistema final.

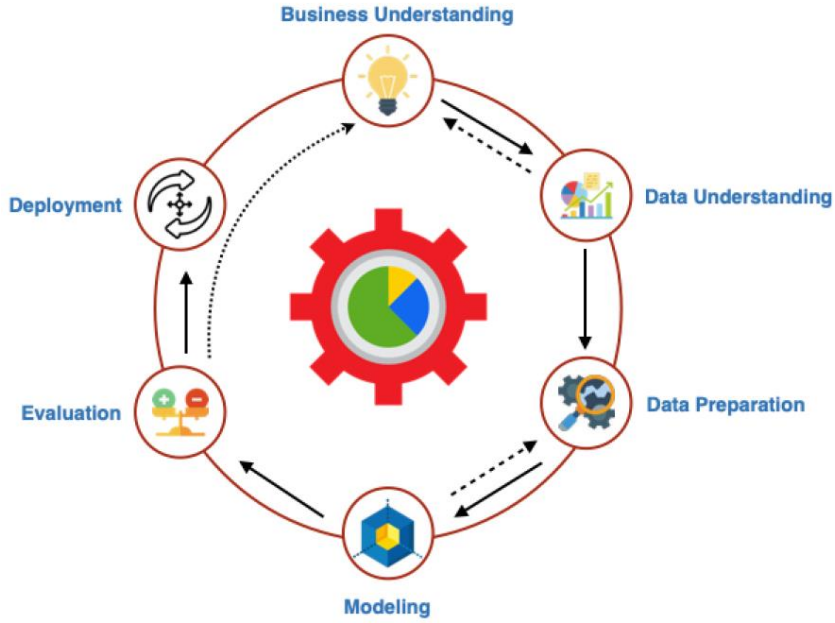

Figura 22 - Fluxograma das diferentes fases do modelo CRISP-DM
Fonte: (GOSMAR, 2020).

Este trabalho propõe-se a analisar a utilização de algoritmos de aprendizado de máquina baseados em redes neurais para classificação de sinais em Libras. Estes sinais podem ser capturados como séries temporais multivariadas. Para realizar essa captura de dados é necessária a construção de um protótipo que contemple os sensores necessários para identificar características suficientes para diferenciar os diferentes sinais. Além disso, esta proposta inclui rotinas de coleta, extração de características, armazenamento e processamento dos dados, classificação e análise dos resultados obtidos. Para posterior implementação do modelo desenvolvido em sistema embarcado, é necessário que o microcontrolador escolhido atenda requisitos relacionados a capacidade de memória e velocidade de processamento. Neste sentido, os estudos relacionados na seção 332.4 foram essenciais para auxiliar nas definições de projeto e de arquitetura que serão apresentados no próximo capítulo.

O desenvolvimento do trabalho pode então ser dividido nas seguintes etapas, correlacionando com o método apresentado pela Figura 22 - Fluxograma das diferentes fases do modelo CRISP-DMFigura 22:

1. Revisão da literatura e definições do projeto (Entendimento do negócio)
2. Prototipação do sistema embarcado (Entendimento dos dados)
3. Aquisição de Dados (Entendimento dos dados)
4. Pré-processamento de dados (Preparação dos dados)
5. Classificação de dados (Modelagem)
6. Avaliação dos resultados
7. Implementação

O tópico 1 de revisão de literatura foi abordado no capítulo 2 enquanto as definições de projeto serão definidas no próximo capítulo. Os tópicos 2 a 5 são discorridos nas seções 4.2, 4.3, 4.4 e 4.5 respectivamente e os tópicos 6 e 7 são discutidos no capítulo 5.

Foi escolhida a implementação do modelo de aprendizado de máquina em sistema embarcado. Outra opção possível para este projeto seria a transferência dos dados coletados para outra plataforma com maior poder de processamento, por exemplo um computador pessoal, *smartphone* ou computador em nuvem. A partir dos pontos levantados na seção 2.3.1, podem-se citar como benefícios da decisão tomada a redução da latência para predição de uma classe, já que desconsidera a latência decorrente da transferência de dados entre plataformas, além da independência do sistema, que em outro caso necessitaria de conexão constante com outra plataforma. Além disso, esta opção evita o dispêndio de maior valor financeiro para desenvolvimento do sistema. Observa-se também que o processo de treinamento do modelo, que depende de maior poder de processamento, não tem a necessidade de ser realizado em ambiente embarcado.

4 DESENVOLVIMENTO

Este capítulo apresenta as definições e implementações de *hardware* e *software* necessárias para alcançar os objetivos propostos na seção 1.1.

4.1 DEFINIÇÕES DO PROJETO

Foram definidos 11 sinais em Libras para estudo e classificação neste trabalho, são eles:

- Oi
- Tudo bem
- Bom dia
- Boa tarde
- Boa noite
- Prazer em conhecer
- Por favor
- Obrigado

- De nada
- Desculpa
- Tchau

Estes sinais são utilizados principalmente em situações de apresentação. Apesar de conterem traços manuais em ambas as mãos, são possíveis de serem diferenciados utilizando apenas uma mão. Além dos sinais selecionados, foi incluído uma classe neutra, chamada neste trabalho de classe de ‘descanso’. Este sinal corresponde ao posicionamento estático da mão sobre uma superfície plana. A adição desta classe tem como objetivo reconhecer classes que estão fora do conjunto de sinais definidos anteriormente. Ao final, têm-se um conjunto de 12 classes para serem treinadas na construção do modelo classificador. Foi definido para coleta de dados, uma quantidade de 50 amostras de cada classe executadas por 1 voluntário. De acordo com estudos anteriores, este é um volume de amostras suficiente para a tarefa de classificação proposta (DIAS, 2020). Para início da coleta de dados é utilizado um botão de pressão. A janela de coleta de cada amostra é definida como 3 segundos, tempo suficiente para execução completa dos de Libras definidos nesta seção.

Nos testes iniciais para aquisição de dados, verificou-se que os dados do sensor giroscópio tem pouca variabilidade durante a execução dos gestos, portanto foram ignorados nesta e nas próximas etapas do projeto. Dados com pouca variabilidade podem acrescentar uma complexidade desnecessária ao modelo de classificação.

A etapa de classificação foi desenvolvida usando a ferramenta *Google Colaboratory* em linguagem de programação *Python.* Os modelos de aprendizado de máquina propostos foram desenvolvidos com auxílio do framework Tensorflow (MARTIN ABADI *et al.*, 2015). O *firmware* implementado no protótipo foi desenvolvido utilizando a linguagem C++ em ambiente de desenvolvimento *Arduino IDE*. A coleta e armazenamento de dados foi feito em ambiente Windows 11, em linguagem *Python* utilizando a plataforma de desenvolvimento Visual Studio Code. Para implementação embarcada do modelo desenvolvido foi utilizada a versão leve do *framework Tensorflow*, chamada de *Tensorflow Lite Micro* (TFLM), desenvolvido especificamente para serem executadas em arquiteturas de microcontroladores utilizando a linguagem de programação C++ (DAVID *et al.*, 2020b). A placa de desenvolvimento utilizada para coleta de dados e implementação embarcada do modelo foi a Heltec WiFi LoRa 32 V2

(HELTEC, 2022) equipada com módulo Esp32, que possui microcontrolador *dual-core* de 240MHz, 448KB de memória de programa, 512KB de memória RAM, protocolo de comunicação I2C e UART, 34 portas programáveis de entrada e saída para uso geral (GPIO) e conversor analógico digital (ADC) de 12 bits (ESPRESSIF, 2022). Todo o código desenvolvido neste projeto está disponível na plataforma *Github*.

4.2 DESENVOLVIMENTO DO PROTÓTIPO

Para realizar a coleta de dados necessária para classificação dos gestos propostos, desenvolveu-se um protótipo de luva instrumentada utilizando sensores de flexão e inerciais. O protótipo foi desenvolvido para ser utilizado na mão direita. Os sensores de flexão foram posicionados nos dedos polegar, indicador e mínimo, enquanto o sensor inercial foi posicionado na região central no dorso da mão, região onde existe maior movimentação. Também foi incluído um botão de pressão no dorso da mão, que serviu para indicar o momento de início da coleta de dados de um gesto. A Figura 23 apresenta uma foto do protótipo desenvolvido e indicações de posicionamento dos componentes utilizados.

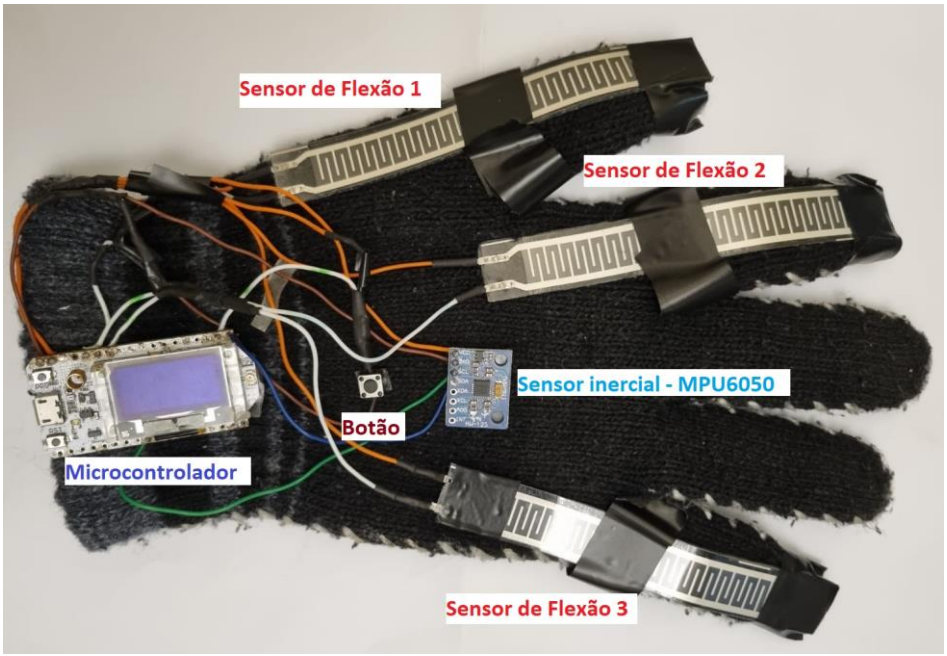

Figura 23 - Protótipo para coleta de dados
Fonte: Elaborada pelo autor

A Figura 24 apresenta o esquemático eletrônico do protótipo e mostra com mais detalhes as conexões dos componentes. O esquemático foi desenvolvido utilizando o software *Fritzing* (FRITZING, 2022). Os sensores de flexão são utilizados conectando-os a resistores de 10kΩ formando divisores de tensão. Os sensores são conectados nas entradas do conversor analógico-digital (ADC) do microcontrolador que lê a tensão resultante do divisor de tensão implementado. Como explicado na seção 2.3.3, a resistência do sensor de flexão aumenta quando este é flexionado. A resistência do sensor é calculada no *firmware* do protótipo através de divisão de tensão, dada pela equação *(2)*, onde R_1é a resistência do sensor de flexão, R_2é o resistor fixo de 10kΩ, V_{in}é a tensão de operação do microcontrolador e V_{adc}é a tensão medida pelo ADC.

$$R_1 = \left(\frac{V_{cc}}{V_{adc}} - 1\right) \cdot R_2 \qquad (2)$$

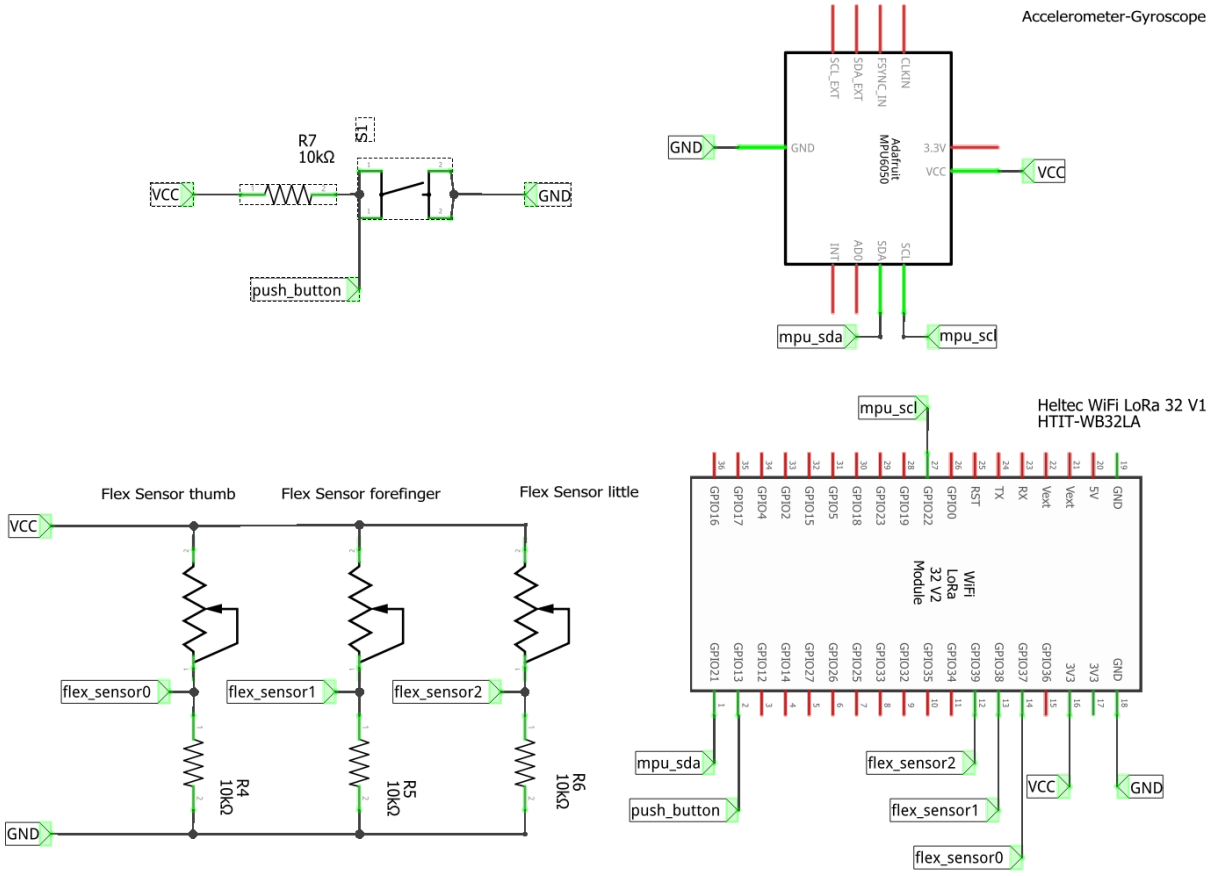

Figura 24 - Esquemático do protótipo
Fonte: Elaborado pelo autor

Não foram incluídos sensores de flexão nos dedos médio e anelar no protótipo, pois, dentre os gestos em Libras selecionados para classificação, a flexão destes dedos não interfere na diferenciação das classes. Além disso, a redução do número de sensores reduz a complexidade do *hardware* e do *firmware* desenvolvidos.

4.3 AQUISIÇÃO DE DADOS

O processo de aquisição de dados do protótipo para o computador é feito através do protocolo de comunicação serial UART a uma taxa de transmissão de 115200 *bauds*. A alimentação do protótipo e a comunicação física com o computador é feita por meio da porta do Barramento Serial Universal (USB). O protótipo é programado com *firmware* que coleta 3 segundos de dados em uma frequência de 20Hz assim que pressionado o botão. No lado do computador, um código na linguagem *python* captura os dados da porta de comunicação dedicada, decodifica os dados e armazena em arquivo formato *.txt* em pastas destinadas a cada um dos gestos coletados. A Figura 25 mostra o diagrama de estados dos *firmwares* do protótipo e do programa *python*, enquanto a Figura 26 mostra o esquema de coleta de dados.

Em uma camada de abstração de nível maior, os dados são transmitidos em uma mensagem de texto do tipo *string* no formato “accX, accY, accZ, flex0, flex1, flex2”, onde *accX*, *accY* e *accZ*, representam os valores brutos dos eixos do acelerômetro (que variam entre -10 e 10) e *flex0*, *flex1* e *flex2*, os valores brutos dos sensores de flexão nos dedos polegar, indicador e mínimo, respectivamente (que variam entre 0 e 4000). Os dados dos sensores de flexão são maiores quando os sensores são mais flexionados.

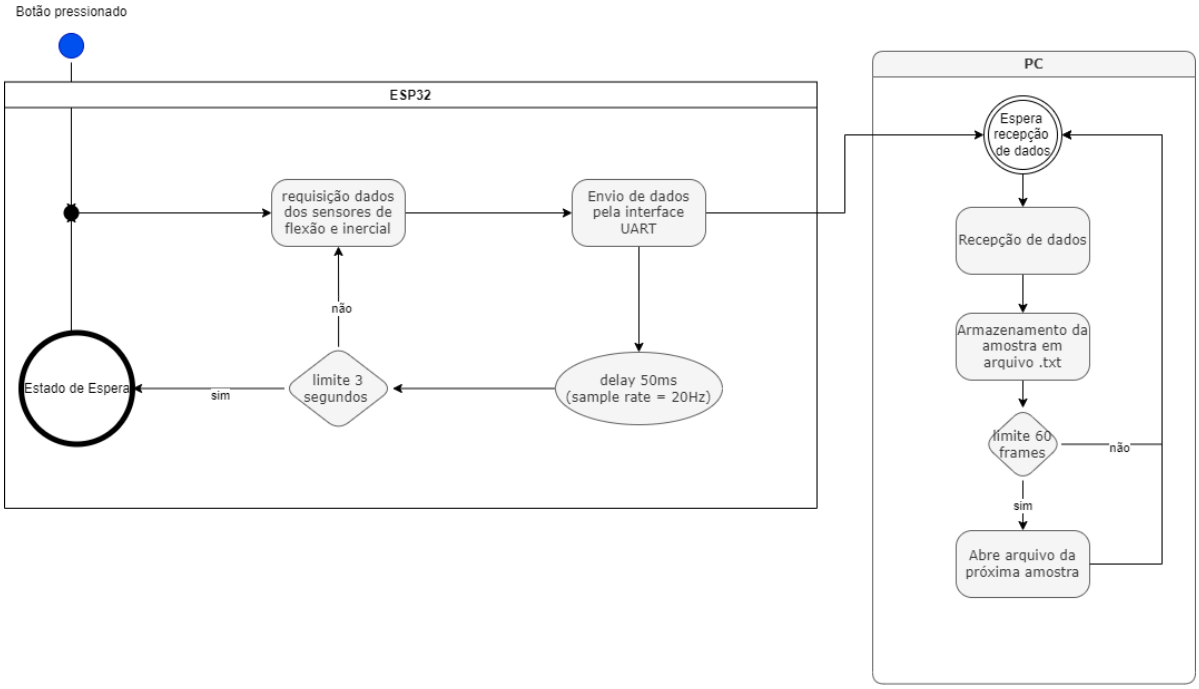

Figura 25 - Diagramas de estados dos programas para coleta de dados
Fonte: Elaborado pelo autor.

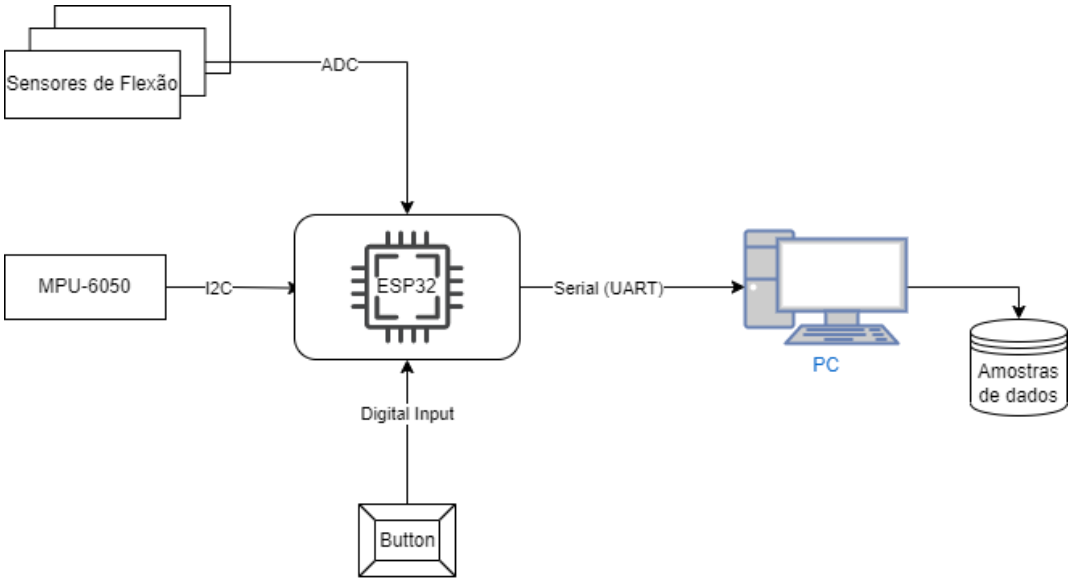

Figura 26 - Esquema de coleta de dados
Fonte: Elaborado pelo autor

4.4 PRÉ-PROCESSAMENTO DE DADOS

Após a aquisição de dados, é necessário entender o comportamento dos dados de cada amostra e prepará-los para utilização nos métodos de aprendizado de máquina escolhidos. Para entender a distribuição de dados entre as variáveis, as amostras foram agrupadas em 6 canais. A Figura 27 mostra o histograma de frequência dos dados por canal. Observa-se que, para os conjuntos de gestos selecionados, em grande parte das amostras os sensores de flexão possuem valor 0. Os dados do acelerômetro apresentam maior extensão de distribuição, principalmente no eixo X.

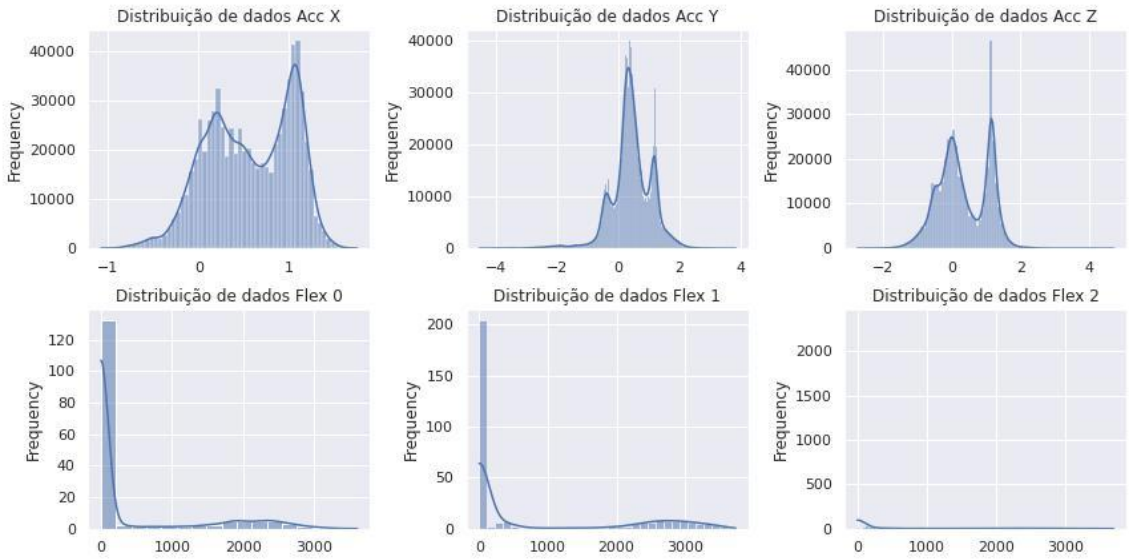

Figura 27 - Distribuição de dados por canal
Fonte: Elaborado pelo autor

Para utilização destes dados nos modelos de classificação, foram feitas duas transformações. A padronização e a normalização por máximo absoluto. O objetivo destas transformações é colocar todas as variáveis na mesma ordem de grandeza. A padronização é feita através da fórmula *z-score*, dada pela equação *(3)*, onde z é a variável padronizada, x é a variável não padronizada, μ é a média e σ é o desvio padrão do conjunto de dados. A normalização é feita dividindo o valor do conjunto de dados pelo máximo absoluto de cada variável. Ao final, têm-se o conjunto de dados de cada variável no intervalo [-1, 1] com média 0, como mostrado na Figura 28.

$$z = \frac{x - \mu}{\sigma} \qquad (3)$$

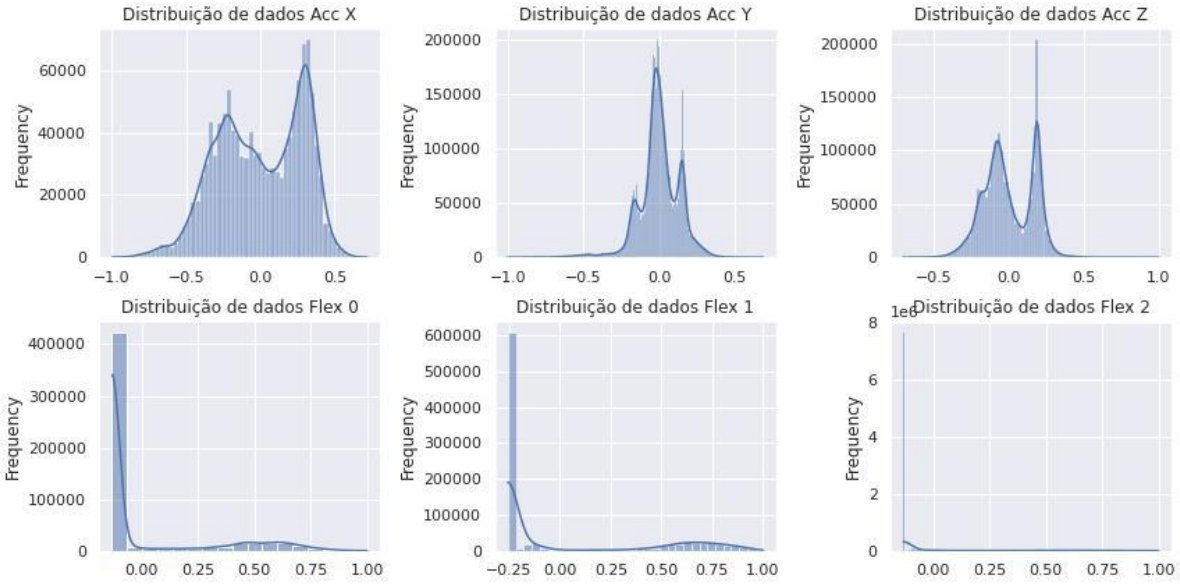

Figura 28 - Distribuição de dados por canal após padronização e normalização
Fonte: Elaborado pelo autor

Para entender a representação dos dados dos sinais coletados, foi elaborada a Figura 29, que apresenta uma amostra aleatória de cada sinal selecionado para classificação ao lado de uma captura de tela da execução do sinal usando o aplicativo Handtalk (HANDTALK, 2012).

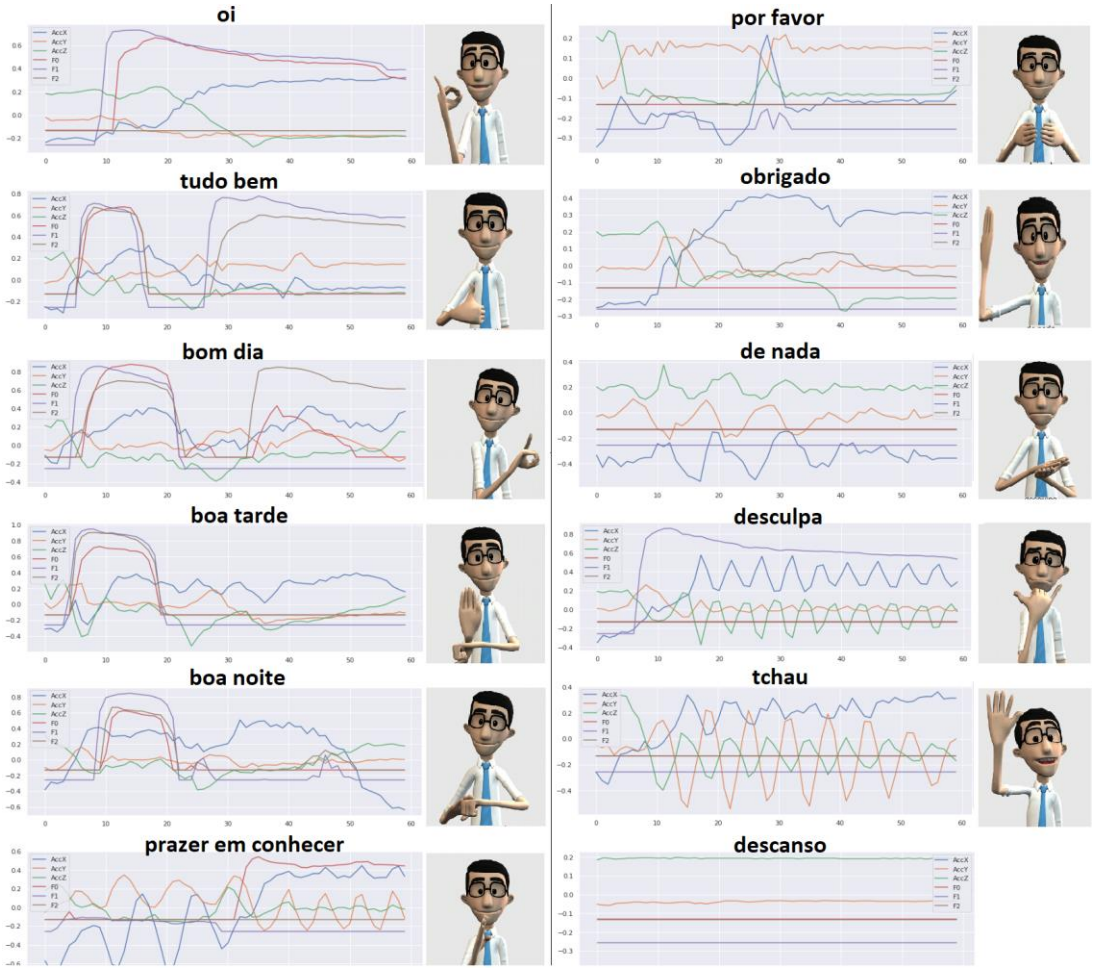

Figura 29 - Representação dos dados coleados para cada gesto escolhido para classificação e imagem da transcrição do gesto feita pelo aplicativo (HANDTALK, 2012)
Fonte: Elaborado pelo autor com adaptações de (HANDTALK, 2012)

4.5 CLASSIFICADOR

Como explicado na seção 1.1, este trabalho faz uma avaliação da aplicação de redes neurais artificiais do tipo LSTM para classificação de gestos em Libras. Foram utilizadas 2 arquiteturas de redes neurais derivadas de LSTM, uma rede neural LSTM simples com 20 neurônios e uma rede neural LSTM combinada com uma camada convolucional a fim de obter melhores resultados. Este segundo modelo é chamado neste trabalho de rede ConvLSTM (SHI *et al.*, 2015) e foi apresentada na seção 2.2.4. Para fins de comparação de resultados, foi escolhido um método de classificação de séries temporais de menor complexidade, que não utiliza arquitetura baseada em

redes neurais artificiais para servir como linha de base. O método KNN (*K-Nearest Neighbors*) foi escolhido para este caso.

Espera-se que o modelo ConvLSTM obtenha melhor resultado, já que a camada convolucional que precede a camada LSTM tem como objetivo extrair características da série temporal que facilitem a tarefa de classificação da camada LSTM.

Para comparação da performance dos modelos escolhidos, foi utilizado o método de validação cruzada *k*-partições com 5 partições de validação. A métrica de acurácia média foi utilizada para determinar o modelo de melhor desempenho. Durante o processo de validação cruzada, os modelos baseados em redes neurais são treinados por 150 épocas e o parâmetro de tamanho do *batch* escolhido foi 16. Além da comparação por validação cruzada, os modelos foram avaliados dividindo os dados em 80% para treinamento e 20% para validação. Desta forma é possível avaliar a evolução no processo de treinamento através da curva de *Loss* por época e o resultado da classificação do modelo para cada uma das amostras de teste por meio da matriz de confusão. Para entender o efeito dos sensores na classificação dos gestos, foi realizada a classificação dos modelos através da validação cruzada analisando separadamente apenas o conjunto de dados dos 3 canais provenientes do acelerômetro e o conjunto de dados proveniente dos sensores de flexão dos dedos.

Neste capítulo foram relatadas as etapas de desenvolvimento do protótipo, coleta, processamento e classificação dos gestos. Na sequência, serão apresentados e analisados os resultados obtidos pelos modelos classificadores.

5 ANÁLISE E DISCUSSÃO DE RESULTADOS

Neste capítulo serão apresentados e analisados os resultados obtidos após os testes com os classificadores selecionados, considerando a situação completa, onde são utilizados os 6 canais de dados dos sensores como variáveis entrada dos modelos. Além disso, são analisados os resultados dos modelos considerando apenas os 3 canais referentes ao acelerômetro e os 3 canais referentes aos sensores de flexão. Para cada análise feita serão apresentados os resultados dos 3 modelos

apresentados anteriormente na tarefa de classificação dos 12 padrões definidos na coleta de dados.

5.1 COMPARAÇÃO DOS MODELOS

A Figura 30 - Acurácia média de cada modelo avaliadoFigura 30 apresenta a acurácia final média obtida por cada um dos modelos após validação cruzada. Como esperado, o modelo ConvLSTM alcançou o melhor desempenho dentre os modelos selecionados, obtendo 97,3% de acurácia. Isso mostra que inclusão de uma camada convolucional junto à camada LSTM melhora a capacidade dessas redes em classificar séries temporais multivariadas, preservando a capacidade de interpretar sinais sequenciais. O modelo KNN, apesar de servir como linha de base e ter pior desempenho dentre os modelos selecionados, ainda obtém desempenho significativo considerando o problema apresentado.

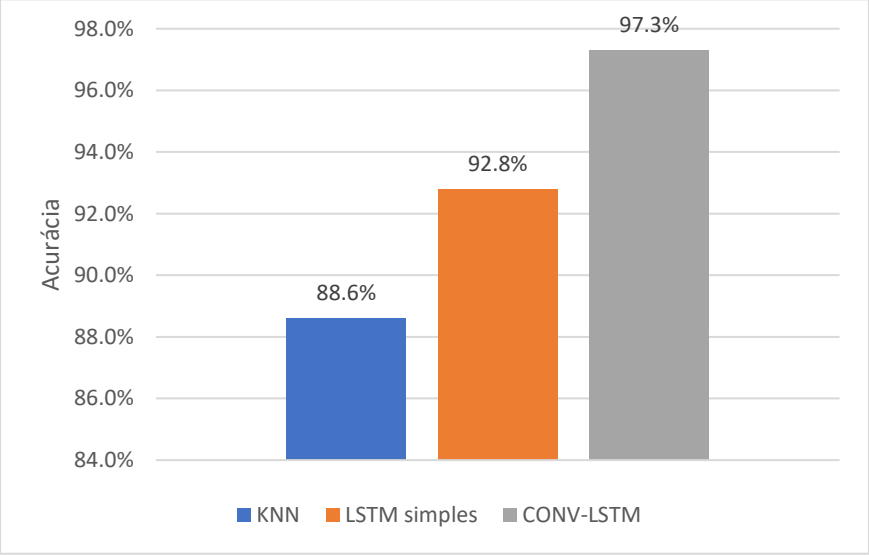

Figura 30 - Acurácia média de cada modelo avaliado
Fonte: Elaborado pelo autor

Analisando o resultado dos modelos utilizando 80% dos dados para treino e 20% para teste, pode-se perceber outras características inerentes ao treinamento dos modelos analisados. A Figura 31 mostra a matriz de confusão obtida na classificação dos dados de teste para cada modelo. É possível perceber que algumas classes

parecidas são classificadas erroneamente pelos modelos LSTM simples e KNN. No caso do modelo KNN, a classe do gesto *'de_nada'* é predita como a classe *'descanso'*. Relembrando a Figura 29, ambas classes apresentam os canais referentes aos sensores flexores como sem flexão e a palma da mão virada para baixo. Já o modelo LSTM simples prevê a maioria das amostras da classe *'descanso'* como *'de_nada'*.

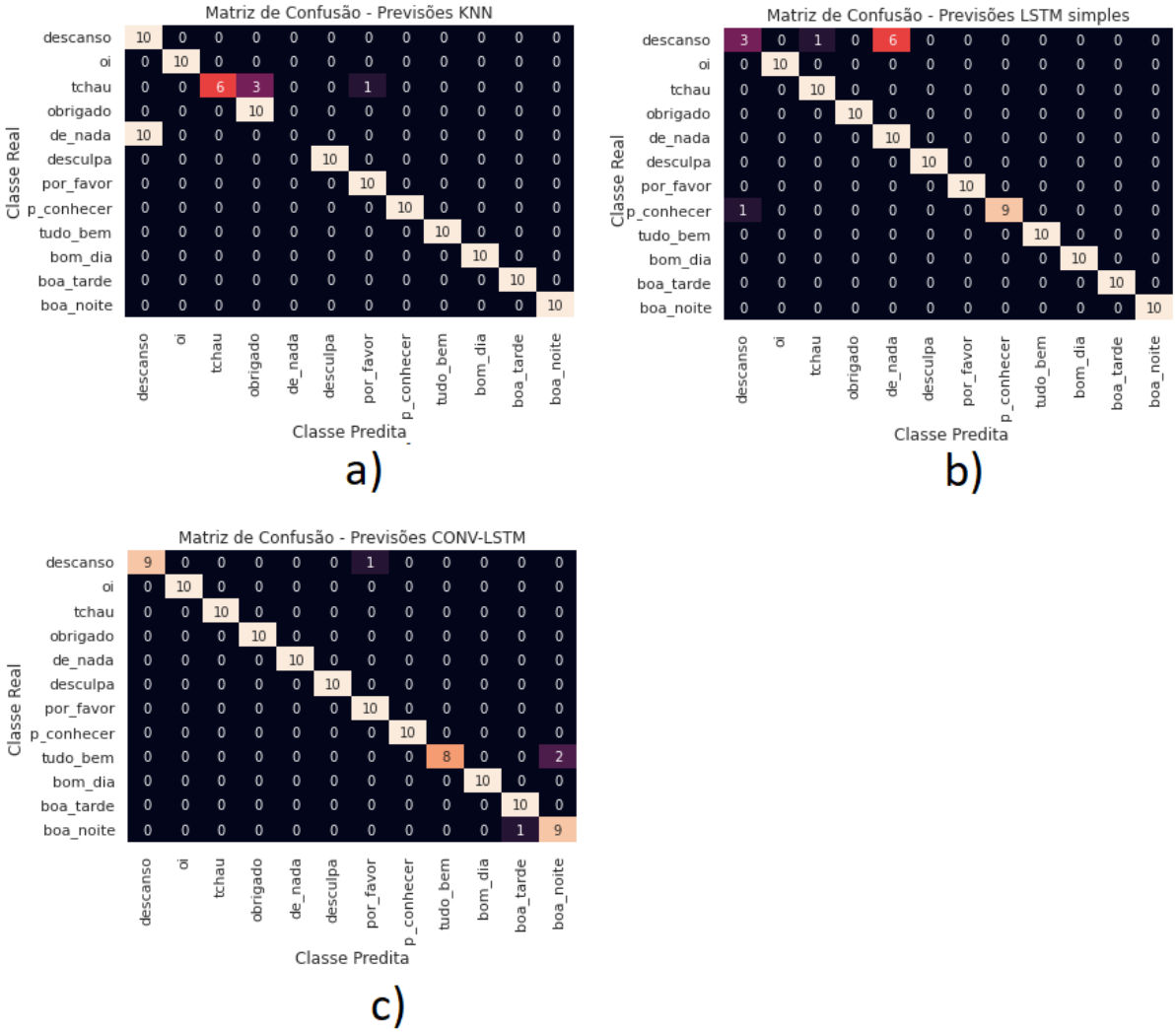

Figura 31 - Matriz de confusão obtida após treinamento de cada modelo utilizando divisão de treino e teste 80/20. Em **a)** o modelo KNN, **b)** o modelo LSTM simples e em **c)** o modelo ConvLSTM
Fonte: Elaborado pelo autor

A Figura 32 apresenta a evolução da função de *Loss* e da acurácia para o conjunto de treino e teste no processo de treinamento dos modelos LSTM simples e ConvLSTM. Observa-se que o modelo ConvLSTM é capaz de convergir mais rápido e com maior estabilidade do que o modelo LSTM simples.

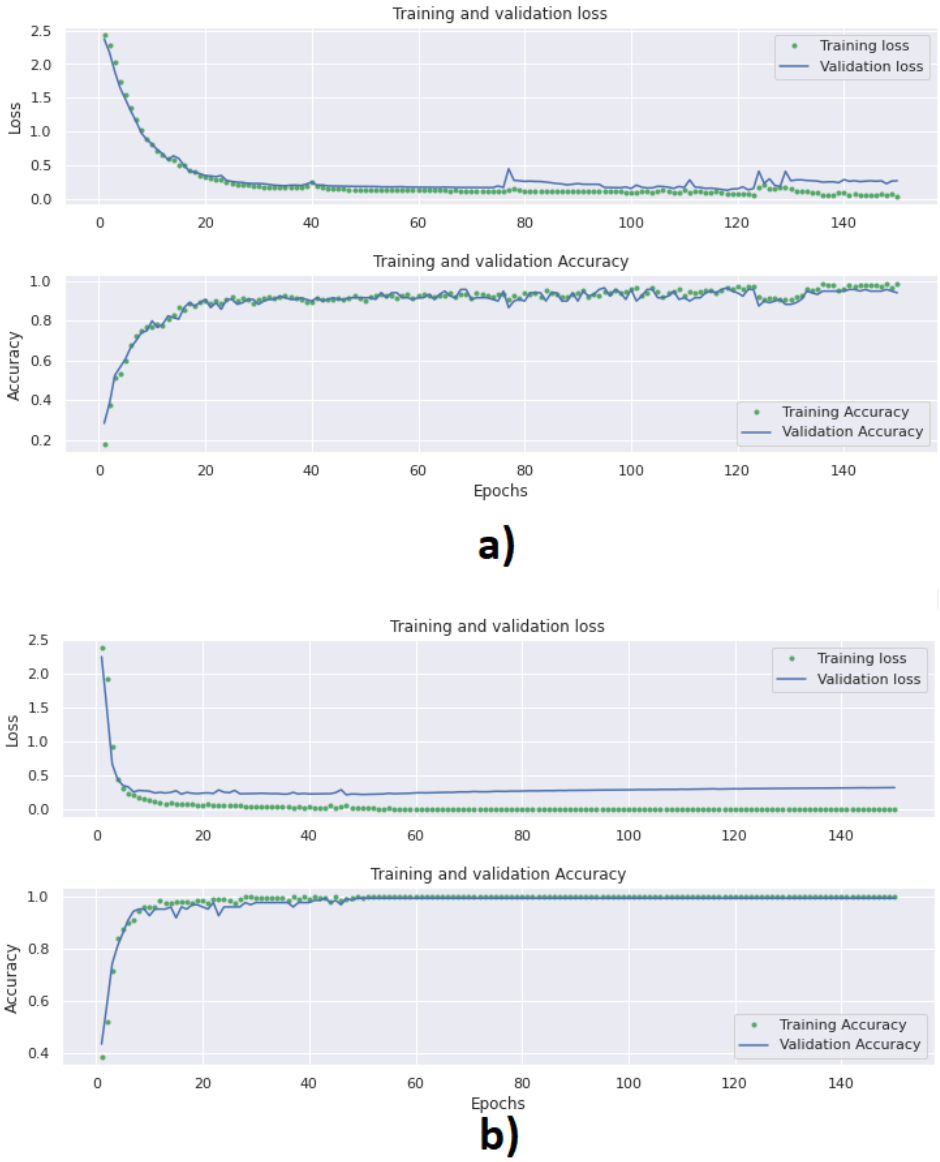

Figura 32 - Evolução da função de *Loss* e acurácia em cada época do processo de treinamento dos modelos. Em **a)** o modelo LSTM simples e em **b)** o modelo ConvLSTM
Fonte: Elaborado pelo autor

5.2 COMPARAÇÃO DOS MODELOS POR TIPO DE SENSOR

Para entender a contribuição de cada sensor como característica de entrada para classificação, foi aplicado o método de validação cruzada primeiramente utilizando apenas os dados do sensor acelerômetro e depois dos sensores de flexão. A Figura 33 apresenta os resultados de acurácia média para a classificação usando os dados do acelerômetro e a Figura 34 a acurácia média para a classificação usando os dados dos sensores de flexão.

Como esperado, para ambos os casos, o modelo ConvLSTM obteve melhor desempenho, com 95,8% no caso utilizando apenas dados do acelerômetro e 71,8% utilizando apenas dados dos sensores de flexão, seguido do modelo LSTM simples e do KNN. Porém, observa-se que os resultados utilizando dados do sensor acelerômetro são consideravelmente superiores aos resultados obtidos que utilizam apenas sensores de flexão, sendo comparáveis com os resultados considerando as variáveis combinadas. Isso mostra que os dados do acelerômetro têm maior peso no processo de classificação. Os sensores de flexão possuem importância consideravelmente menor em relação ao acelerômetro, porém ainda relevantes.

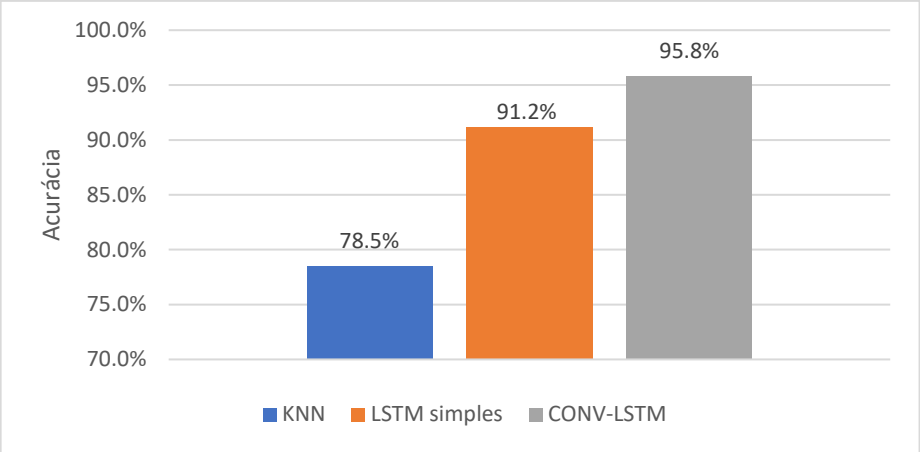

Figura 33 - Acurácia média de cada modelo avaliado considerando apenas dados do acelerômetro
Fonte: Elaborado pelo autor.

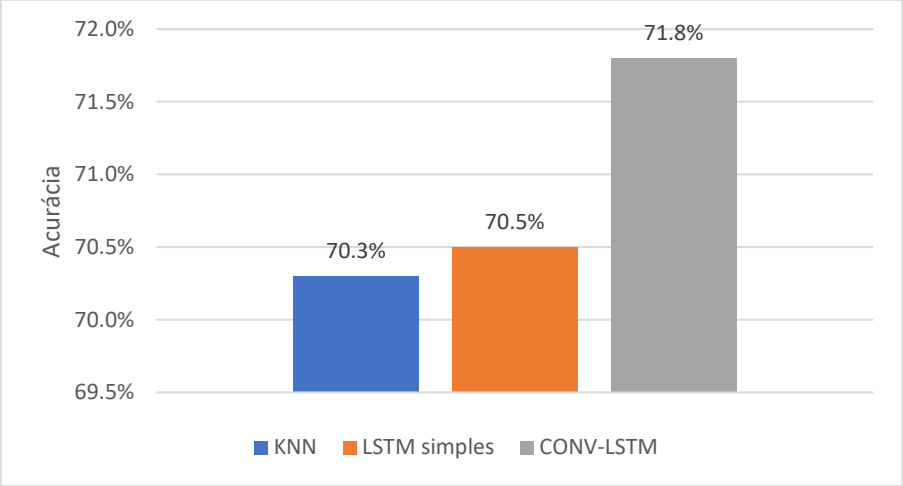

Figura 34 - Acurácia média de cada modelo avaliado considerando apenas dados dos sensores de flexão
Fonte: Elaborado pelo autor.

5.3 IMPLEMENTAÇÃO DO MODELO EM AMBIENTE EMBARCADO

Como mostrado na Figura 17, os próximos passos para implementação do modelo de aprendizado de máquina em um microcontrolador são, a quantização, a codificação, a conversão do modelo para implementação em arquitetura embarcada. O processo de quantização foi ignorado neste processo para simplificação da implementação. Com a utilização da biblioteca *Tensorflow Lite Micro*, o processo de codificação resulta em um arquivo do tipo *tflite* que guarda as informações do modelo e a conversão resulta em um arquivo de cabeçalho (*.h*) para ser utilizado pelo *firmware* do microcontrolador. Dentre os modelos desenvolvidos, ambos baseados em redes neurais artificiais obtiveram bom resultado. Para simplificação do processo e maior velocidade na predição, foi escolhido o modelo mais simples, o LSTM simples, para implementação no sistema embarcado. O modelo resultante após a conversão para implementação no sistema embarcado possui 27kB de tamanho.

A Figura 35 mostra um fluxograma do *firmware* implementado no microcontrolador, dividindo-o em fase de *setup* (configuração), em que são definidos, inicializados e configurados todas bibliotecas e periféricos utilizados, e fase de *loop* (ciclo), que é executado continuamente enquanto o microcontrolador estiver ligado. Este ciclo executa a tarefa de aquisição de dados quando o botão é pressionado, realiza o pré-processamento que consiste na normalização de dados e executa o processo de predição do modelo de aprendizado de máquina treinado. Para exibição do resultado, o programa aproveita da conexão USB com o computador e retorna o resultado obtido através da interface serial (UART), como mostrado na Figura 36. A etapa de predição demora em torno de 30 milissegundos para retornar um resultado, tempo relativamente baixo para a complexidade do modelo.

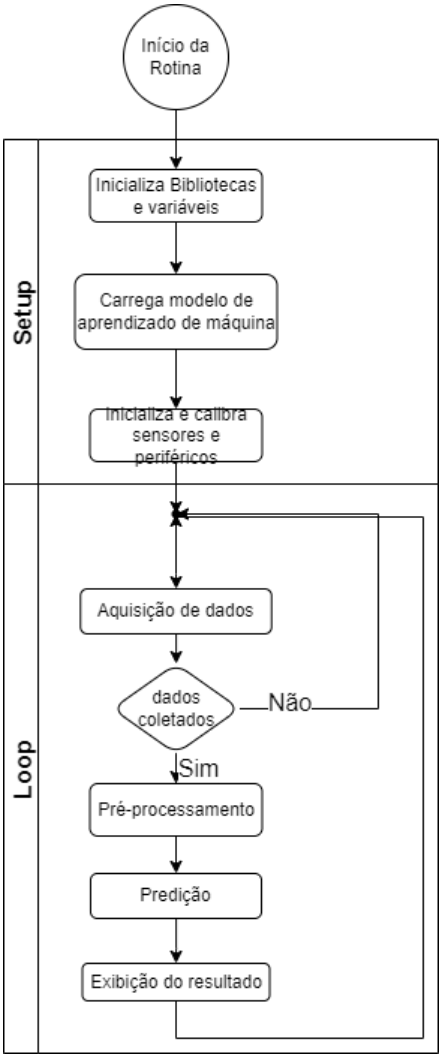

Figura 35 - Fluxograma simplificado do *firmware* de predição implementado no microcontrolador
Fonte: Elaborado pelo autor

```
1:15:03.387 -> Gesto captado
1:15:03.433 -> Classe: oi
1:15:03.433 -> Tempo de predicao: 30.770ms
1:15:03.433 ->
1:15:03.433 ->
1:15:05.350 -> Início da coleta de dados
1:15:08.410 -> Gesto captado
1:15:08.456 -> Classe: tchau
1:15:08.456 -> Tempo de predicao: 30.632ms
1:15:08.456 ->
1:15:08.456 ->
1:15:10.963 -> Início da coleta de dados
1:15:13.895 -> Gesto captado
1:15:13.945 -> Classe: descanso
1:15:13.945 -> Tempo de predicao: 30.624ms
```

Figura 36 - Exemplo da exibição do resultado de algumas predições
Fonte: Elaborado pelo autor

6 CONCLUSÃO

Esta pesquisa concluiu seu objetivo de apresentar um método capaz de coletar, analisar e classificar gestos em Libras. Concluiu-se que redes neurais do tipo LSTM são capazes de resolver a tarefa de classificação de séries temporais multivariadas com dados de linguagens de sinais e que existe ganho de desempenho quando utilizadas camadas convolucionais junto as camadas LSTM para esta tarefa.

Através da revisão bibliográfica, apresentada no capítulo 2, foi possível entender a complexidade inerente as linguagens de sinais e a Libras, além de estudar as pesquisas dedicadas ao reconhecimento automático de linguagens de sinais desenvolvidas até o momento. Percebe-se que, apesar de haver farta literatura no tema de reconhecimento de linguagens de sinais, ainda existe grande espaço a ser explorado no reconhecimento de Libras. No capítulo 3 foi apresentada a metodologia proposta para desenvolvimento do trabalho e as etapas para alcançar os objetivos definidos. No capítulo 4 foi documentado todo o desenvolvimento do *hardware* e *software* que culmina na avaliação dos modelos de classificação, apresentada no capítulo 5. Considerando as amostras coletadas, o modelo com melhor desempenho foi o ConvLSTM, que apresentou acurácia de 97,3%, comparável a outras literaturas estudadas anteriormente.

Ao longo do desenvolvimento do trabalho, foram identificados pontos de melhoria que podem expandir a usabilidade do protótipo e aumentar a acurácia do modelo, que não foram implementados aqui por limitações de tempo e/ou recurso. Estes pontos são colocados como propostas de continuidade do projeto. Pelo lado do classificador, acredita-se que seja possível alcançar uma melhor acurácia através da utilização de algumas técnicas como, a busca de hiperparâmetros, *dropout* e *data augmentation.* Ademais, este trabalho teve foco na implementação de classificadores baseados na rede neural do tipo LSTM, sendo implementados apenas dois modelos baseados nesta arquitetura. Desta forma, a implementação de outros modelos de arquitetura baseadas em redes neurais artificiais é uma sugestão de desenvolvimento para trabalhos futuros.

Observado o *hardware* desenvolvido, é possível encontrar ainda mais pontos de melhoria. Na seção 5.2 foi analisado que os dados dos sensores de flexão têm pouca influência na classificação dos sinais frente aos dados do acelerômetro. Porém, é possível que, adicionando um maior conjunto de sinais, estes sensores tenham maior influência. Além disso, foi notado grande ruído nos dados deste sensor, por isso acredita-se que um filtro eletrônico analógico possa auxiliar na maior estabilidade das amostras de dados. Outro ponto refere-se à coleta de dados e alimentação do protótipo, que optou pela conexão com fio de interface USB entre o protótipo e o computador, porém, observou-se que esta conexão gerou uma pequena limitação na execução dos sinais. É proposto que, em próximas versões deste protótipo, seja implementada uma conexão sem fio.

Outras implementações sugeridas para trabalhos futuros são: O aumento no vocabulário de sinais, a coleta de dados com maior número de voluntários e o aumento da quantidade de amostras por sinal.

Quanto à implementação do modelo em ambiente microcontrolado, destaca-se o tempo de predição de aproximadamente 30 milissegundos, intervalo de tempo pequeno em relação ao tempo de aquisição de dados de 3 segundos. Observa-se também que o tamanho do modelo resultante, de apenas 27KB, é bem inferior a memória de programa de 448KB disponível no Esp32. Conclui-se então que o microcontrolador opera bem abaixo do seu limite, possibilitando a implementação de modelos maiores e mais complexos.

O desenvolvimento de protótipos que envolvem *firmware* e *hardware*, por si só é uma atividade desafiadora e multidisciplinar, envolve experiência e habilidade tanto na construção física quanto no desenvolvimento do software. A implementação deste trabalho exigiu domínio de teorias envolvendo sistemas embarcados, eletrônica, desenvolvimento de software e aprendizado de máquina. Apesar de todas melhorias necessárias para considerar este um produto funcional, acredita-se que o protótipo elaborado neste trabalho é possível de ser considerado como mínimo produto viável capaz de auxiliar em demandas da comunidade surda no Brasil bem como auxiliar no processo de aprendizado em Libras.

REFERÊNCIAS

AHMED, M. A. *et al.* **A Review on Systems-Based Sensory Gloves for Sign Language Recognition State of the Art between 2007 and 2017.** *Sensors*, v. 18, n. 7, p. 2208, 9 jul. 2018.

ALMEIDA, R. M. A.; MORAES, C. H. V.; SERAPHIM, T. F. P. ***Programação de Sistemas Embarcados: Desenvolvendo software para microcontroladores em linguagem C*****.** 1ª ed. ed. Rio de Janeiro: Elsevier, 2016.

AMIDI, S.; AMIDI, A. ***Deep Learning Cheatsheet***. Disponível em: <https://stanford.edu/~shervine/teaching/cs-230/>. Acesso em: 22 jan. 2022.

BAI, S.; KOLTER, J. Z.; KOLTUN, V. **An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling.** 3 mar. 2018.

BALBINOT, A.; BRUSAMARELLO, V. J. ***Instrumentação e Fundamentos de Medidas*****.** Grupo Gen-LTC, 2000.

BANBURY, C. R. *et al.* **Benchmarking TinyML Systems: Challenges and Direction**. 2020.

BERTOLLO, D. *et al.* **GESTÃO DO CONHECIMENTO E REDES NEURAIS ARTIFICIAIS UMA ANÁLISE BIBLIOMÉTRICA NA BASE SCOPUS.** ***Propriedade intelectual: estudos propectivos e inovação tecnológic****a*. Associação Acadêmica de Propriedade Intelectual - API, 2020. p. 199–210.

BESSA CARNEIRO, S. *et al.* **Static gestures recognition for Brazilian Sign Language with kinect sensor**. out. 2016, IEEE, out. 2016. p. 1–3.

BISHOP, C. M. ***Pattern Recognition and Machine Learning (Information Science and Statistics)*****.** Berlin, Heidelberg: Springer-Verlag, 2006.

BLÁZQUEZ-GARCÍA, A. *et al.* **A Review on Outlier/Anomaly Detection in Time Series Data**. *ACM Computing Surveys*, v. 54, n. 3, p. 1–33, jun. 2021.

BRASIL. ***Lei Nº 10.436, de 24 de abril de 2002***. Disponível em: <http://www.planalto.gov.br/ccivil_03/Leis/2002/L10436.htm>. Acesso em: 11 jan. 2022.

CHEOK, M. J.; OMAR, Z.; JAWARD, M. H. **A review of hand gesture and sign language recognition techniques.** ***International Journal of Machine Learning and Cybernetics***, v. 10, n. 1, p. 131–153, 8 jan. 2019.

CHU, Y.-C. *et al.* **Recognition of Hand Gesture Sequences by Accelerometers and Gyroscopes.** ***Applied Sciences***, v. 10, n. 18, p. 6507, 18 set. 2020.

CHUAH, M. C.; FU, F. ECG Anomaly Detection **via Time Series Analysis.** ***Frontiers of High Performance Computing and Networking ISPA 2007 Workshops*****. Berlin, Heidelberg:** Springer Berlin Heidelberg, 2007.

CYBERGLOVE SYSTEMS. ***CyberGlove® II Wireless Glove***. Disponível em: <https://static1.squarespace.com/static/559c381ee4b0ff7423b6b6a4/t/5602fbd2e4b07ebf58d47ecd/1443036114091/CyberGloveII_Brochure.pdf>. Acesso em: 19 jan. 2022.

DATA SCIENCE ACADEMY. ***Deep Learning Book***. Disponível em: <https://www.deeplearningbook.com.br/>. Acesso em: 9 jan. 2021.

DAVID, R. *et al.* **TensorFlow Lite Micro: Embedded Machine Learning on TinyML Systems**. 16 out. 2020b.

DIAS, T. S. ***LUVA INSTRUMENTADA PARA RECONLUVA INSTRUMENTADA PARA RECONHECIMENTO DE PADRÕES DE GESTOS EM LIBRASHECIMENTO DE PADRÕES DE GESTOS EM LIBRAS***. 2020. Universidade Tecnológica Federal do Paraná, Curitiba, 2020.

DUTTA, DR. L.; BHARALI, S. **TinyML Meets IoT: A Comprehensive Survey.** ***Internet of Things***, v. 16, p. 100461, dez. 2021.

EBERHARD, D. M.; SIMONS, G. F.; FENNIG, C. D**.** ***Ethnologue: Languages of the World*****.** 24ª ed. Dallas, Texas: SIL International, 2021. Disponível em: <https://www.ethnologue.com/>. Acesso em: 9 jan. 2022.

ER-RADY, A. *et al.* **Automatic sign language recognition: A survey**. maio 2017, [S.l.]: IEEE, maio 2017. p. 1–7.

ESPRESSIF. ***ESP32 Datasheet***. Disponível em: <https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf>. Acesso em: 29 jun. 2022.

FERNANDO DELUNO GARCIA. ***Introdução aos sistemas embarcados e microcontroladores*****.** Disponível em: <https://bit-by-bit.gitbook.io/embedded-systems/introducao-aos-sistemas-embarcados>. Acesso em: 19 jan. 2022.

FRITZING. ***Software Fritzing***. Disponível em: <https://fritzing.org/>. Acesso em: 29 jun. 2022.

GHANEM, S.; CONLY, C.; ATHITSOS, V. **A Survey on Sign Language Recognition Using Smartphones**. 21 jun. 2017, New York, NY, USA: ACM, 21 jun. 2017. p. 171–176.

GILL, N. S. ***Artificial Neural Networks Applications and Algorithms*.** Disponível em: <https://www.xenonstack.com/blog/artificial-neural-network-applications>. Acesso em: 18 jan. 2022.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. ***Deep Learning***. MIT Press, 2016.

GOSMAR, D. ***Machine Learning: The Sixth Chakra of Artificial Intelligence***. [S.l.]: Independently Published, 2020.

HANDTALK. ***Hand talk***. Disponível em: <https://www.handtalk.me/br>. Acesso em: 24 jan. 2022.

HELTEC. ***Heltec WiFi LoRa 32 (V2) especifications***. Disponível em: <https://heltec.org/project/wifi-lora-32/>. Acesso em: 29 jun. 2022.

IBGE. *Censo demográfico : 2010 :* ***características gerais da população, religião e pessoas com deficiência***. Rio de Janeiro: [s.n.], 2010.

INVENSENSE. ***MPU-6000 and MPU-6050 Product Specification***. Disponível em: <https://invensense.tdk.com/wp-content/uploads/2015/02/MPU-6000-Datasheet1.pdf>. Acesso em: 14 jan. 2022.

ISLAM, MD. Z. *et al.* **Static Hand Gesture Recognition using Convolutional Neural Network with Data Augmentation**. IEEE, maio 2019. p. 324–329.

ISMAIL FAWAZ, H. *et al.* **Deep learning for time series classification: a review.** ***Data Mining and Knowledge Discovery***, v. 33, n. 4, p. 917–963, 2 jul. 2019.

JEPSEN, J. B. *et al.* ***Sign Languages of the World: A Comparative Handbook***. [S.l.]: De Gruyter Mouton, 2015.

KANANI, P.; PADOLE, M. ECG ***Heartbeat Arrhythmia Classification Using Time-Series Augmented Signals and Deep Learning Approach***. Procedia Computer Science, 2020.

KARIM, F. *et al.* **LSTM Fully Convolutional Networks for Time Series Classification.** *IEEE Access*, v. 6, p. 1662–1669, 2018.

LEA, C. *et al.* **Temporal Convolutional Networks for Action Segmentation and Detection**. jul. 2017, [S.l.]: IEEE, jul. 2017. p. 1003–1012.

LIN, L. *et al.* **Medical Time Series Classification with Hierarchical Attention-based Temporal Convolutional Networks: A Case Study of Myotonic Dystrophy Diagnosis**. 27 mar. 2019.

MADHIARASAN, DR. M.; ROY, PROF. P. P. **A Comprehensive Review of Sign Language Recognition: Different Types, Modalities, and Datasets**. 7 abr. 2022.

MARTIN ABADI *et al.* ***TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems***. Disponível em: https://www.tensorflow.org/, 2015.

MEC. ***Saberes e Práticas da Inclusão***. Brasília, 2006.

MONTEIRO, C. *et al.* ***Um sistema de baixo custo para reconhecimento de gestos em LIBRAS utilizando visão computacional***. Sociedade Brasileira de Telecomunicações, 2016.

NASCIMENTO, L. *et al.* ***METODOLOGIA DE CARACTERIZAÇÃO DE UM SENSOR DE FLEXÃO***. set. 2017, [S.l: s.n.], set. 2017.

OLIVEIRA, R. F. DE. ***Inteligência artificial***. Londrina: Editora e Distribuidora Educacional S.A., 2018.

OORD, A. VAN DEN *et al.* **WaveNet: A Generative Model for Raw Audio**. 12 set. 2016.

PEREIRA, M. C. *Libras: Conhecimento Além dos Sinais*. 1ª ed. Pearson Universidades, 2011.

PINHEIRO, N. ***Introdução a Series Temporais — Parte 1***. Disponível em: <https://medium.com/data-hackers/series-temporais-parte-1-a0e75a512e72>. Acesso em: 17 jan. 2022.

REZENDE, T. M.; ALMEIDA, S. G. M.; GUIMARÃES, F. G. **Development and validation of a Brazilian sign language database for human gesture recognition.** ***Neural Computing and Applications***, v. 33, n. 16, p. 10449–10467, 1 ago. 2021.

ROCHA SILVA, B. C. *et al.* **Methodology and comparison of devices for recognition of sign language characters**. maio 2017, [S.l.]: IEEE, maio 2017. p. 1–6.

SAFFO, P. **Sensors: The next wave of innovation**. *Communications of the ACM*, v. 40, n. 2, p. 92–97, 1997.

SAGGIO, G. *et al.* **Resistive flex sensors: a survey.** ***Smart Materials and Structures***, v. 25, n. 1, p. 013001, 1 jan. 2016.

SAHOO, J. P.; ARI, S.; GHOSH, D. K. **Hand gesture recognition using DWT and *F* -ratio based feature descriptor**. *IET Image Processing*, v. 12, n. 10, p. 1780–1787, out. 2018.

SCRIGROUP. ***Microcontrolador PIC16F887***. Disponível em: <https://www.scrigroup.com/calculatoare/hardware/Microcontroler-PICF21428.php>. Acesso em: 22 jan. 2022.

SECCO, R.; BARROS, L. F. DA S. B.; HERVERTON, M. **Ambiente Interativo para Aprendizagem em LIBRAS Gestual e Escrita**. nov. 2009, [S.l: s.n.], nov. 2009.

SHAEFFER, D. K. **MEMS inertial sensors: A tutorial overview**. *IEEE Communications Magazine*, v. 51, n. 4, p. 100–109, abr. 2013.

SHAHEEN, H.; MEHMOOD, T. T**alking Gloves: Low-Cost Gesture Recognition System for Sign Language Translation**. jul. 2018, [S.l.]: IEEE, jul. 2018. p. 219–224.

SHI, X. *et al.* **Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting**. 12 jun. 2015.

SPECTRASYMBOL. ***Spectra symbol flex sensor datasheet***. Disponível em: <https://www.sparkfun.com/datasheets/Sensors/Flex/flex22.pdf>. Acesso em: 22 jan. 2022.

SZE, V. *et al.* **Efficient Processing of Deep Neural Networks: A Tutorial and Survey**. *Proceedings of the IEEE*, v. 105, n. 12, p. 2295–2329, dez. 2017.

VLIBRAS. ***VLibras***. Disponível em: <https://www.gov.br/governodigital/pt-br/vlibras>.

WESTIN, R. ***Baixo alcance da língua de sinais leva surdos ao isolamento***. Disponível em: <https://www12.senado.leg.br/noticias/especiais/especial-cidadania/baixo-alcance-da-lingua-de-sinais-leva-surdos-ao-isolamento>. Acesso em: 11 jan. 2022.

ZHU, Y. *et al.* **A novel approach to workload prediction using attention-based LSTM encoder-decoder network in cloud environment.** *EURASIP Journal on Wireless Communications and Networking*, v. 2019, n. 1, p. 274, 17 dez. 2019.