DISPOSITIVO PARA RECONHECIMENTO DE PADRÕES SONOROS E PALAVRAS DITAS PARA DEFICIENTES AUDITIVOS

Manuscript template for the REPA Journal

Ben Oliveira Friebus[1]
orcid.org/0009-0007-8131-4683

Marcílio André Félix Feitosa[2]
orcid.org/0000-0001-6654-5220

[1]Escola Politécnica de Pernambuco, Universidade de Pernambuco, Recife, Brasil. E-mail: bof@poli.br

[2]Escola Politécnica de Pernambuco, Universidade de Pernambuco, Recife, Brasil. E-mail: marcilio@poli.br

DOI: 10.25286/repa.v11i4.3983

RESUMO

Este trabalho apresenta o desenvolvimento de um dispositivo baseado em inteligência artificial para pessoas surdas, com capacidade de detectar padrões sonoros de emergência, transcrever fala em texto possibilitando leitura e reconhecer comandos de voz, convertendo-os em estímulos visuais e táteis. O sistema foi implementado utilizando a placa *Raspberry Pi*, empregando a biblioteca *YAMNet*, baseada em redes neurais convolucionais, para a classificação de sons de emergência, e a biblioteca *SpeechRecognition* para o reconhecimento e transcrição de palavras. A resposta do dispositivo ocorre através da exibição da fala em um display LCD, acionamento de LEDs e de um motor de vibração acoplado a uma pulseira, proporcionando uma forma eficiente de comunicação não verbal. Os testes realizados com sons do cotidiano e transcrevendo as palavras faladas confirmaram a eficácia do sistema, operando de maneira simultânea.

PALAVRAS-CHAVE: *YAMNet;* Inteligência Artificial; *SpeechRecognition*

ABSTACT

This work presents the development of an artificial intelligence-based device for deaf individuals, capable of detecting emergency sound patterns, transcribing speech into text for reading, and recognizing voice commands, converting them into visual and tactile stimuli. The system was implemented using the Raspberry Pi board, employing the YAMNet library, based on convolutional neural networks, for emergency sound classification, and the SpeechRecognition library for word recognition and transcription. The device responds through the activation of LEDs and a vibration motor attached to a wristband, providing an efficient form of non-verbal communication. Tests conducted with everyday sounds and speech transcription confirmed the effectiveness of the system, operating simultaneously.

KEY-WORDS: YAMNet; Artificial Intelligence; SpeechRecognition;

1 INTRODUÇÃO

Globalmente, mais de 1,5 bilhão de pessoas experimentam algum grau de perda auditiva. Destes, estima-se que 430 milhões tenham perda auditiva moderada ou grave no ouvido com melhor audição **[1]**. A deficiência auditiva é um desafio que afeta milhões de pessoas em todos os países. De acordo com dados do Instituto Brasileiro de Geografia e Estatística (IBGE), cerca de 5% da população brasileira tem algum tipo de deficiência auditiva, equivalente a mais de 10 milhões de pessoas. Entre elas, aproximadamente 2,7 milhões vivem com surdez profunda, ou seja, não ouvem absolutamente nada **[2]**. Pessoas surdas enfrentam dificuldades consideráveis em diversos aspectos da vida cotidiana, desde a comunicação até a segurança pessoal. A partir de entrevistas com membros da comunidade surda, os achados indicam que os surdos enfrentam dificuldades em situações cotidianas que, em geral, são triviais para os ouvintes.

Para uma pessoa surda, a falta de percepção de alertas sonoros de emergência, como por exemplo sirenes de incêndio, pode levar a situações de extremo perigo. Estudos indicam que a deficiência auditiva está associada a um risco significativamente maior de lesões acidentais. Segundo pesquisa realizada nos Estados Unidos, pessoas com grandes dificuldades auditivas podem ter até o dobro de chances de sofrer acidentes em comparação com aquelas com audição normal **[3]**. Esse risco se deve, principalmente, à dificuldade de perceber sinais sonoros de alerta, como buzinas ou sirenes, sendo mais predominante em ambientes de trabalho.

Nesse contexto, a tecnologia surge como um meio fundamental para mitigar esses desafios e garantir a inclusão social, oferecendo soluções inovadoras que proporcionam meios alternativos para a detecção de alertas sonoros. A inclusão social dessas pessoas é essencial para garantir que elas tenham as mesmas oportunidades e condições de segurança que os ouvintes.

A Tecnologia Assistiva é um campo interdisciplinar que reúne produtos, metodologias e serviços voltados a promover a funcionalidade, autonomia e qualidade de vida de pessoas com deficiência, incapacidades ou mobilidade reduzida **[4]**. Seus recursos vão desde objetos simples, como bengalas, até sistemas sofisticados, como softwares e equipamentos de comunicação alternativa. Para ajudar essas pessoas a se tornarem mais independentes, incluí-las na sociedade e melhorar suas habilidades de convivência no trabalho e na sociedade.

1.1 MOTIVAÇÃO

Iniciou-se este trabalho durante uma conversa com pessoas surdas. Foi destacado o risco de vida no que diz respeito a um morador surdo em um prédio não conseguir ouvir sirenes de emergência durante um incêndio. O dispositivo proposto visa alertá-los do perigo. Outrossim, no trânsito, o Brasil ocupou o terceiro lugar em número de mortes em 2023, de acordo com dados da Organização Mundial da Saúde (OMS). Pessoas com deficiência auditiva enfrentam mais desafios, como não ouvir buzinas ou chamadas de alerta **[5]**. Este projeto busca fornecer acessibilidade e segurança para pessoas com deficiência auditiva, sendo um passo inicial para futuros desenvolvimentos e pesquisas. Além de melhorar a segurança, beneficia a sociedade como um todo, estimulando a inovação em soluções acessíveis.

1.2 OBJETIVO GERAL

O objetivo deste projeto é desenvolver um dispositivo, utilizando inteligência artificial, capaz de transcrever fala em texto, interpretar sons de alarmes de emergência, sirenes de alerta e palavras faladas, tornando essas informações acessíveis para pessoas surdas por meio de vibrações e sinais luminosos. Para isso, será implementado, através da linguagem em *Python*, um sistema de reconhecimento de voz para identificar palavras específicas, como nomes de pessoas. Além do desenvolvimento e integração de um módulo de reconhecimento de sons de emergência utilizando técnicas de processamento de áudio **[6] [7]**.

O dispositivo contará com atuadores que emitirão alertas visuais e táteis em resposta aos sons detectados, garantindo maior acessibilidade.

Além disso, será projetado de forma modular e expansível, permitindo futuras atualizações e adaptações para diferentes tipos de alertas e necessidades dos usuários. Por fim, serão realizados testes em ambientes reais para validar sua eficácia e funcionalidade, assegurando que o dispositivo cumpra seu propósito de proporcionar mais segurança e autonomia às pessoas surdas.

2 REFERENCIAL TEÓRICO

2.1 MACHINE LEARNING

Machine Learning é um subcampo da *IA* e busca tornar as máquinas inteligentes utilizando métodos matemáticos que possibilitam o aprendizado a partir de dados **[8]**.

A ideia central é que, ao observar dados históricos, o sistema possa identificar padrões e fazer previsões sobre novos dados. Isso é possível através dos modelos matemáticos que ajustam parâmetros internos para minimizar erros e melhorar a precisão **[9]**.

De acordo com a Universidade da Califórnia, em *Berkeley,* o sistema de aprendizado de um algoritmo de *Machine Learning* pode ser dividido em três componentes principais **[10]**. O primeiro é o processo de decisão, onde o algoritmo analisa os dados de entrada e gera uma previsão ou classificação com base em padrões identificados.

O segundo componente é a função de erro, responsável por avaliar a qualidade da previsão feita pelo modelo. Quando há exemplos conhecidos, essa função compara as estimativas do modelo com os valores reais, permitindo mensurar o grau de precisão alcançado.

O terceiro e último componente é o processo de otimização do modelo, no qual são realizados ajustes nos pesos para melhorar a adequação do modelo aos dados de treinamento. Esse processo ocorre de forma iterativa: o algoritmo avalia seu desempenho, ajusta os pesos para reduzir os erros detectados e repete essas etapas sucessivamente até que seja atingido um nível aceitável de precisão **[11]**. Esse processo é a base do funcionamento de modelos avançados como o *YAMNet*, utilizado neste projeto para classificação de sons **[12]**.

2.2 FERRAMENTAS DE PROCESSAMENTO DE ÁUDIO UTILIZADAS

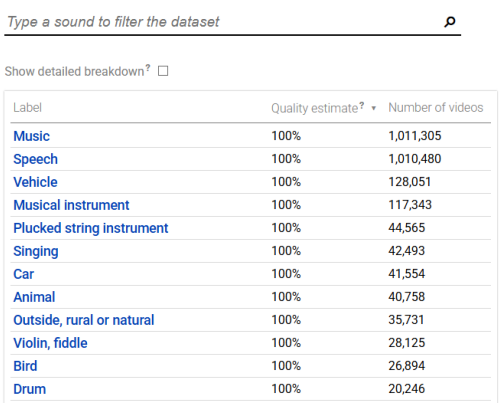
Para compreender adequadamente o funcionamento do dispositivo proposto, é fundamental conhecer as principais bibliotecas e ferramentas em *Python* utilizadas no desenvolvimento do sistema, bem como suas respectivas finalidades.

2.2.1 *YAMNet*

O *YAMNet* (*Yet Another Mobile Network*) é um modelo de rede neural convolucional pré-treinada desenvolvido pelo *Google Research* e disponibilizado através da plataforma *TensorFlow Hub* **[13]**. Ele utiliza o conjunto de dados *AudioSet,* uma enorme coleção de trechos rotulados do *YouTube,* para aprender e identificar 527 categorias distintas de eventos de áudio, conforme mostra a Figura 1.

A arquitetura do *YAMNet* é otimizada para classificação de eventos acústicos, sendo capaz de identificar classes de sons, como sirenes, alarmes, animais, entre outros. O modelo recebe como entrada sinais de áudio em formato de ondas (*waveforms*), amostrados a 16 kHz, com duração mínima de 0,96 segundos e processa-os para gerar uma distribuição de probabilidades sobre as diversas classes sonoras **[14]**.

Figura 1 - Conjunto de classes do *AudioSet*



Label	Quality estimate?	Number of videos
Music	100%	1,011,305
Speech	100%	1,010,480
Vehicle	100%	128,051
Musical instrument	100%	117,343
Plucked string instrument	100%	44,565
Singing	100%	42,493
Car	100%	41,554
Animal	100%	40,758
Outside, rural or natural	100%	35,731
Violin, fiddle	100%	28,125
Bird	100%	26,894
Drum	100%	20,246

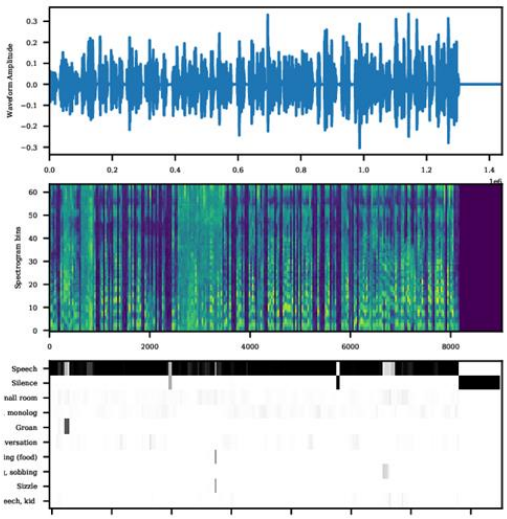
Fonte: [15]

O modelo é construído sobre a arquitetura MobileNet v1, projetada para ser leve e eficiente, o que permite seu uso em dispositivos com capacidade de processamento limitada. Essa eficiência é alcançada através do uso de camadas convolucionais separáveis em profundidade (*depthwise separable convolutions*), que reduzem drasticamente a quantidade de parâmetros sem comprometer a capacidade de generalização do modelo.

A Figura 2, a seguir, demonstra que o áudio, antes de ser processado pela rede, é capturado e convertido em um espectrograma. Que por sua vez, é uma representação visual da distribuição de energia do sinal em função do tempo e da frequência. Posteriormente, esse espectrograma é transformado em uma escala mel, que busca imitar a maneira como o ouvido humano percebe as variações de frequência **[16]**. Por fim, determina-se qual classe apresenta a maior probabilidade de corresponder ao som captado.

Figura 2 - Conjunto de classes do *AudioSet*

DOI: 10.25286/repa.v11i4.3983



Fonte: [17]

Dentre as principais vantagens que motivaram a escolha do *YAMNet* neste projeto, destacam-se o pré-treinamento robusto, que eliminou a necessidade de desenvolver uma nova rede neural do zero, economizando tempo e recursos computacionais; a capacidade de identificar sons críticos de emergência, como sirenes, alarmes de incêndio e alarmes veiculares, de forma confiável; a facilidade de integração, uma vez que o modelo está disponível no *TensorFlow Hub*, podendo ser carregado diretamente no código com poucas linhas de programação, como evidencia a Figura 3; a eficiência computacional, que permite sua execução em dispositivos de baixa potência, como o *Raspberry Pi.*

Figura 3 – Carregamento do *YAMNet*

```
import tensorflow_hub as hub
self.model = hub.load('https://tfhub.dev/google/yamnet/1')
```

Fonte: Do autor.

2.2.2 *SpeechRecognition*

A biblioteca *SpeechRecognition* é um dos *frameworks* mais utilizados para o processamento de reconhecimento automático de fala (*ASR – Automatic Speech Recognition*) em linguagem *Python* **[18]**. Trata-se de uma interface de alto nível que facilita o acesso e a integração com diversos serviços de reconhecimento de voz, tanto locais, quanto baseados em nuvem, como *Google Web Speech API, CMU Sphinx, Microsoft Bing Voice Recognition, IBM Speech to Text*, entre outros **[19]**. Cada uma dessas opções possui características próprias em termos de precisão, suporte a múltiplos idiomas, necessidade de conexão com a *internet* e desempenho.

O *recognizer* do *Google Web Speech API* foi a escolha adotada neste projeto devido a uma combinação de fatores: ele é gratuito, não exige chave de API ou cadastro prévio (os demais *recognizers* exigiam), oferece um nível de precisão muito elevado, sendo um dos melhores desempenhos do mercado. E, mais importante, possui suporte nativo ao idioma português do Brasil (pt-BR) com capacidade de detectar, interpretar, falas com diferentes sotaques e orações coloquiais com elevada confiabilidade.

3 METODOLOGIA

O projeto consistiu no desenvolvimento de um dispositivo capaz de classificar sons de emergência e de transcrever palavras ditas no ambiente, para auxiliar pessoas com deficiência auditiva. O sistema foi implementado utilizando a linguagem de programação *Python* em um *Raspberry Pi*, juntamente com microfone para captura de áudio, modelos de *Machine Learning* e atuadores como LEDs, motor de vibração e *display* LCD.

3.1 MATERIAIS UTILIZADOS

3.1.1 *Raspberry Pi*

Para a construção do sistema, foi utilizado como unidade central de processamento o *Raspberry Pi Model 4 B* com 4 GB de RAM. A configuração inicial do dispositivo envolve a instalação do sistema operacional *Raspberry Pi OS* em um cartão MicroSD, permitindo a operação do módulo **[20] [21]**.

Considerando a necessidade de desempenho rápido e alta confiabilidade em dispositivos de emergência, a escolha do *Raspberry Pi* se deve ao seu poder de processamento superior. Equipado com um SoC BCM2711 de 1,5 GHz e cache L2 de 1 MB, o *Raspberry Pi* oferece desempenho mais rápido e preciso comparado ao ESP32, que possui microprocessadores de 240 MHz e 512 kB de SRAM cada. Além disso, o sistema operacional do *Raspberry Pi* é mais versátil, suportando diversas linguagens de programação e permitindo o desenvolvimento de aplicações complexas com inteligência artificial, necessárias para o processamento de áudio **[22]**.

Algumas bibliotecas precisam ser instaladas no *Raspberry Pi* para que o programa funcione corretamente. A Figura 4 mostra o comando utilizado no terminal para realizar a instalação de todas elas.

Figura 4 – Prompt para instalação das bibliotecas

```
sudo apt update && sudo apt install python3-pip python3-dev
libatlas-base-dev portaudio19-dev libffi-dev libssl-dev -y
pip3 install tensorflow==2.12.0 tensorflow_hub numpy sounddevice
RPi.GPIO RPLCD SpeechRecognition pyaudio
```

Fonte: Do autor.

3.1.2 Entrada de áudio

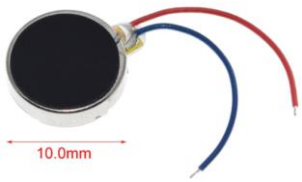
O dispositivo de entrada de áudio utilizado foi um microfone sem fio Onistek On-MC811 **[23]**. Como o modelo utilizado não possui entrada analógica de áudio, foi necessário um adaptador P2 para USB e conectar ao *Raspberry Pi*. Permitindo a captação dos sons para processamento e análise pelo sistema **[24]**.

3.1.3 Atuadores

Para transcrever as palavras captadas pelo microfone, o sistema utilizou um *display* LCD 16x2 com interface I2C, responsável por exibir as informações ao usuário **[25]**. Além disso, foram empregados LEDs indicadores e um motor de vibração modelo 1027, mostrado na Figura 5. Este motor foi acoplado a um transistor 2N2222, operando como chave eletrônica para o acionamento do motor **[26]**. Um resistor de 1 kΩ foi conectado em série com a base do transistor, com a finalidade de limitar a corrente proveniente do GPIO do *Raspberry Pi*, protegendo tanto o transistor quanto o dispositivo.

Adicionalmente, um diodo 1N4007 foi colocado em paralelo ao motor, com o objetivo de proteger o transistor contra picos de tensão reversa gerados pela indutância do motor durante o desligamento **[27]**. E por fim, duas pilhas AA 1,5 V cada, usadas para alimentar o motor de vibração.

Figura 5 – Motor de vibração modelo 1027



Fonte: [28]

Para o acionamento dos LEDs indicadores, foram conectados dois resistores de 470 Ω em paralelo em cada LED, com a função de limitar a corrente a um valor seguro, tipicamente entre 10 mA e 20 mA, evitando sobrecarga e prolongando a vida útil dos LEDs.

3.2 RECONHECIMENTO DE SONS DE EMERGÊNCIA

O reconhecimento de padrões sonoros foi baseado no modelo pré-treinado *YAMNet*, disponibilizado no *TensorFlow Hub*. O *YAMNet*, como visto anteriormente, é uma rede neural convolucional especializada na classificação de eventos sonoros **[13]**. A captura de áudio é feita em tempo real utilizando a biblioteca *SoundDevice*, com taxa de amostragem de 16 kHz, processando blocos de 1,5 segundos. O áudio é ajustado e dividido para que o modelo *YAMNet* possa classificá-lo, indicando a probabilidade de cada classe sonora. Foram consideradas classes de emergência como "*siren*", "*ambulance (siren)*", "*fire alarm*", entre outras como demonstra a Figura 6, previamente mapeadas para alertar o usuário.

Figura 6 – Classes de interesse do *AudioSet*

```
CLASSES_EMERGENCIA = {
    'siren': 'Sirene',
    'ambulance (siren)': 'Sirene',
    'fire engine, fire truck (siren)': 'Sirene',
    'police car (siren)': 'Sirene',
    'reverse beeps': 'Bipe Re',
    'car alarm': 'Alarme',
    'fire alarm': 'Alarme',
    'emergency vehicle': 'Sirene'
}
```

Fonte: Do autor.

Os blocos de áudio são processados pelo modelo, convertidos em espectogramas e então são comparados com as classes de interesse previamente selecionadas. Quando uma classe obtiver um valor de probabilidade acima do "LIMIAR_CONFIANÇA" uma sequência de ações é executada automaticamente pelo sistema.

O LED vermelho é acionado, permanecendo aceso por 3 segundos, com a finalidade de indicar visualmente a detecção de um evento relevante.

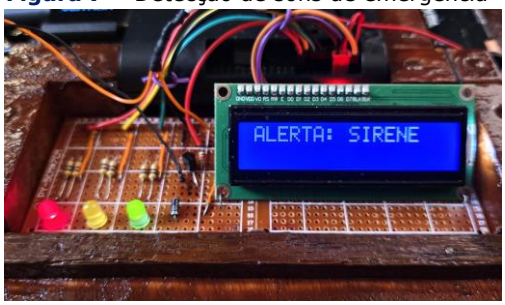
Ao mesmo tempo, o *display* LCD exibe a mensagem de "ALERTA:", seguida pelo nome da classe detectada, informando ao usuário sobre o tipo específico de som identificado pelo sistema, como mostra a Figura 7.

E, simultaneamente, o pino GPIO do *Raspberry Pi*, que está conectado à base do transistor, é ajustado para nível lógico alto (1), o que provoca a saturação do transistor e, consequentemente, o fechamento do circuito, permitindo o acionamento do motor de vibração. O motor, então, executa três pulsos sucessivos, cada um com duração de 0,7 segundos, promovendo uma sinalização tátil que

DOI: 10.25286/repa.v11i4.3983

complementa o alerta visual, garantindo a efetiva notificação ao usuário.

Figura 7 – Detecção de sons de emergência



Fonte: Do autor.

3.3 RECONHECIMENTO DE FALA

Simultaneamente ao reconhecimento de sons de emergência, o programa contém um módulo de reconhecimento de fala, utilizando a biblioteca *SpeechRecognition.*

Seu funcionamento é simples. O processo iniciou-se com a criação de um objeto da classe "*Recognizer()*", responsável por fornecer os métodos necessários para o processamento do áudio e a realização do reconhecimento de fala.

Em seguida, utilizou-se a função "*Microphone()*" para acessar o dispositivo físico de entrada, no caso, um microfone conectado via adaptador P2 para USB.

Antes de iniciar a escuta propriamente dita, foi fundamental realizar um ajuste automático para o ruído ambiente por meio da função *"adjust_for_ambient_noise()"*, que calibra a sensibilidade do microfone com base nas condições sonoras do local. Esse ajuste foi configurado com uma duração de 0,5 segundos, um tempo suficiente para que o sistema reconhecesse e compensasse eventuais ruídos constantes ou interferências do ambiente, assegurando uma captação mais limpa e precisa da fala.

Após essa calibração, o sistema realizou a escuta ativa com a função *"listen()"*, que aguarda até que um som é detectado, capturando então a amostra de áudio correspondente e armazenando-a em uma variável para posterior processamento. Esta função recebe como parâmetro o *"timeout"* definido em 0,5 segundos, significa o tempo pela detecção de fala. Caso não haja detecção de palavras nesse intervalo, a escuta será interrompida, e uma nova tentativa será iniciada para garantir um sistema rápido e responsivo.

Outro parâmetro, ainda dentro da função *"listen()"*, importante de evidenciar é o *"phrase_time_limit"* que limita a duração máxima da gravação em 5 segundos, evitando que o sistema processe áudios excessivamente longos comprometendo a responsividade ao detectar palavras de interesse.

Por fim, essa amostra de áudio é enviada para a API de reconhecimento de fala do *Google* utilizando o método *"recognize_google()"*, que retornou como resultado a transcrição automática da fala detectada. Esse método foi configurado com o parâmetro *language='pt-BR'*, assegurando que o reconhecimento fosse realizado em português brasileiro. A Figura 8 retrata um trecho do código contendo toda essa configuração. A partir daí, todas as palavras contidas no áudio são transcritas e exibidas no *display* LCD em tempo real, como comprova a Figura 8.

Figura 8 – Transcrição em tempo real da fala

Fonte: Do autor.

No sistema implementado, a palavra "João" foi definida como um exemplo genérico para representar um usuário surdo e servir como comando de ativação. Sempre que, durante a transcrição do áudio captado pelo microfone, for detectada a presença dessa palavra, o dispositivo executará: primeiramente, o LED amarelo será acionado como sinal visual.

Simultaneamente, o Raspberry Pi enviará um sinal pelo GPIO, que saturará a base do transistor, fechando o circuito e ligando do motor de vibração. O motor, então, emitirá dois pulsos curtos, cada um com 0,2 segundos de duração, funcionando como alerta tátil para o usuário.

E, no mesmo instante, o sistema exibirá a palavra "João" no *display* LCD, informando textualmente o comando detectado. A Figura 9 demonstra o funcionamento neste caso.

Figura 9 – Detecção da palavra João durante a fala

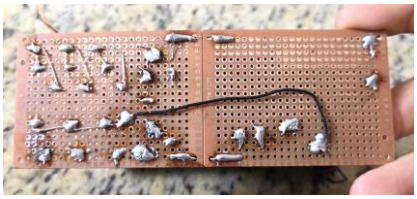
Fonte: Do autor.

Por fim, se o usuário dizer a frase "encerrar programa", o dispositivo reconhece e entra na rotina de "encerrar graciosamente", todos os LEDs e o motor são desligados, as portas GPIO são liberadas com "*GPIO.cleanup()*", o *display* exibe a frase "Sistema *Offline*", e os recursos do áudio são finalizados, garantindo a segurança do sistema.

3.4 MONTAGEM DO CIRCUITO

O circuito foi montado soldando os componentes em placas de protótipo padrão ilha, conforme mostra a Figura 10.

Figura 10 – Vista traseira da placa



Fonte: Do autor.

Inicialmente, foi realizada a conexão dos LEDs indicadores com 2 resistores em paralelo, para limitar a corrente, de 470 Ω cada. O LED verde, responsável por indicar o estado ativo do sistema, foi conectado ao GPIO17, com resistor limitador adequado para proteção do componente. O LED amarelo, correspondente à detecção de voz, especificamente à palavra "João", foi ligado ao GPIO27. Por fim, o LED vermelho, destinado à sinalização de sons de emergência, foi conectado ao GPIO22.

Em seguida, foi realizada a instalação do motor de vibração modelo 1027, com corrente de 90 mA. Para permitir seu acionamento sem sobrecarregar o *Raspberry Pi*, foi utilizado o transistor 2N2222, configurado como chave eletrônica e 2 pilhas AA de 1,5V em série. O fio positivo do motor foi conectado diretamente as pilhas, enquanto o fio negativo foi ligado ao coletor do transistor. O emissor do 2N2222 foi conectado ao GND do *Raspberry Pi*, e a base recebeu o controle do GPIO23, através de um resistor de 1kΩ, garantindo o ajuste adequado da corrente. Foi adicionado também um diodo de proteção 1N4007 em paralelo com o motor, para evitar danos causados pela corrente reversa gerada durante as comutações.

O motor de vibração 1027 foi acoplado a uma pulseira de borracha, mostrado na Figura 11, permitindo que o usuário perceba fisicamente os sinais de emergência de forma imediata.

Figura 11 – Motor de vibração acoplado na pulseira

Fonte: Do autor.

O *display* LCD 16x2, com interface I2C, foi integrado ao sistema para exibir as mensagens de status e alertas. O módulo foi conectado aos pinos SDA e SCL do *Raspberry Pi* (GPIO2 e GPIO3, respectivamente), além das conexões de alimentação em 5V e GND.

E, finalmente, o microfone Onistek ON-MC811 utilizado para captar sinais de voz, foi integrado ao sistema via a biblioteca *SpeechRecognition*, operando conectado à entrada USB do *Raspberry Pi*.

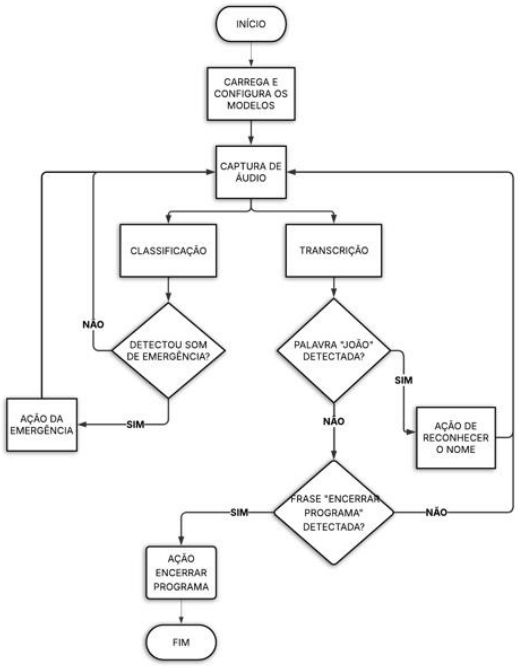
3.4 FLUXOGRAMA DO FUNCIONAMENTO

O fluxograma apresentado na Figura 12 descreve a arquitetura funcional do dispositivo, destacando as principais etapas de execução do sistema. Como o projeto faz uso de múltiplas *threads*, o fluxo de operação ocorre de maneira paralela: enquanto uma rotina é responsável pelo monitoramento contínuo de sons ambientes por meio da rede neural *YAMNet*, outra executa o reconhecimento de comandos de voz com a biblioteca *SpeechRecognition.* Estas rotinas, que funcionam de forma assíncrona e concorrente, interagem com o núcleo principal do sistema, que é responsável pela ativação dos LEDs, motor de

DOI: 10.25286/repa.v11i4.3983

vibração e *display* LCD, conforme os eventos detectados.

Figura 12 – Fluxograma completo



Fonte: Do autor.

4 RESULTADOS E DISCUSSÕES

Algumas estratégias foram adotadas para otimizar o funcionamento do sistema. Uma delas foi a escolha por bibliotecas prontas, como *YAMNet* e *SpeechRecognition* com API do *Google*, eliminou a necessidade de treinar modelos, economizando tempo e recursos computacionais. Outra estratégia essencial foi a simultaneidade entre os processos de detecção sonora e processamento de fala, tornando a estrutura de execução paralela no desenvolvimento do sistema. Este modelo de execução se mostrou eficiente, permitindo que diferentes sinais de entrada (sons de emergencia e palavras ditas) fossem processados de maneira independente, com respostas rápidas e coordenadas por meio dos atuadores.

Na parte eletrônica, uma pilha externa foi implementada para alimentar o motor de vibração, já que o *Raspberry Pi* não fornecia corrente suficiente para acioná-lo.

No código, o tempo de análise do *YAMNet* é um parâmetro fundamental que impacta diretamente o equilíbrio entre precisão e velocidade na detecção. Um tempo maior de análise tende a gerar respostas mais precisas, mas também mais lentas. Por outro lado, um tempo menor proporciona respostas mais rápidas, porém com possível redução na precisão. Após a realização de diversos testes, optou-se por reduzir esse tempo de análise para 1,5 segundos, o que possibilitou acelerar a resposta do sistema sem comprometer a confiabilidade dos resultados.

Além disso, ajustar o tempo de escuta foi essencial para tornar o reconhecimento de fala mais eficiente. Se o tempo configurado fosse muito curto, o sistema poderia interromper a captação no meio de uma fala contínua, cortando alguma palavra antes de ser completamente detectada. Por outro lado, se o tempo de escuta fosse configurado como muito longo, poderia ocorrer um atraso na resposta do sistema. Por exemplo, se a pessoa dissesse "João" no início de uma frase e demorasse a concluir o restante, o reconhecimento só seria processado após o sistema detectar um breve silêncio maior que 0,5 segundos (configurado como o *timeout*). Isso faria com que a resposta, como o acionamento do LED, *display* e do motor, fosse percebida com atraso. Assim, por meio de testes, a escolha de 5 segundos gerou tempo de escuta suficiente para captar toda a frase sem interrupções, e, também curto o bastante para permitir respostas rápidas. Podendo reduzir possíveis atrasos, caso o sistema detecte palavras de forma ininterrupta.

Por último, o uso do *display* LCD via I2C garantiram uma montagem compacta e com menos fiação.

Caso o leitor tenha interesse em ver o sistema em funcionamento ou o código desenvolvido, pode acessar "https://tinyurl.com/2m4zcdv5".

5 CONCLUSÃO

Este projeto busca oferecer uma solução tecnológica inovadora para ampliar a segurança e a acessibilidade de pessoas surdas. Além de reduzir os riscos associados à falta de percepção de sons críticos, o dispositivo contribui para a autonomia dos usuários, promovendo uma maior inclusão social.

Os resultados obtidos demonstraram que a solução proposta é satisfatória, pois o dispositivo apresentou respostas rápidas na detecção de sons de emergência e precisão na transcrição de fala, mesmo em ambientes ruidosos. Além disso, o sistema é capaz de executar essas funções de forma simultânea, sem comprometer o desempenho, garantindo maior estabilidade e eficiência. As otimizações realizadas, tanto no

software quanto no circuito eletrônico, tornaram as respostas rápidas e confiáveis.

Como sugestão para trabalhos futuros, propõe-se a integração deste sistema em um *smartwatch*, que, aliado a um aplicativo dedicado, seja capaz de realizar a mesma classificação de sons e reconhecimento de palavras-chave, acionando vibrações e notificações diretamente no pulso do usuário. Essa evolução poderia ampliar ainda mais a autonomia e a qualidade de vida das pessoas surdas, oferecendo uma solução discreta, portátil e sempre acessível.

REFERÊNCIAS

[1] ORGANIZAÇÃO PAN-AMERICANA DA SAÚDE. **Saúde auditiva.** Disponível em: https://tinyurl.com/efju7ps4. Acesso em: 26 mai. 2025.

[2] JORNAL USP. **Mais de 10 milhões de brasileiros apresentam algum grau de surdez.** Disponível em: https://tinyurl.com/5dp6p9aj. Acesso em: 26 mai. 2025.

[3] MICROSOM. **Pessoas com deficiência auditiva correm risco maior de sofrerem lesões acidentais.** Disponível em: https://tinyurl.com/5n7wsara. Acesso em: 02 jun. 2025.

[4] ASSISTIVA. **Tecnologia assistiva.** Disponível em: https://tinyurl.com/yp9uxpm3. Acesso em: 15 mai. 2025.

[5] METRÓPOLES. **Ranking trágico: Brasil é 3º país que mais registra mortes no trânsito.** Disponível em: https://tinyurl.com/yeykjbuz. Acesso em: 27 mai. 2025.

[6] SILVA, Lucas et al. **Reconhecimento de voz offline utilizando modelos pré-treinados e TensorFlow Lite.** Disponível em: https://tinyurl.com/yftrwfyf. Acesso em: 09 abr. 2025.

[7] UNIVERSIDADE ESTADUAL DE MATO GROSSO DO SUL. **Projeto de Fim de Curso - PFC 183.** Disponível em: https://tinyurl.com/9e55x5at. Acesso em: 11 abr. 2025.

[8] IBM. **Machine learning.** Disponível em: https://tinyurl.com/ymc7f6yh. Acesso em: 13 abr. 2025.

[9] CARVALHO, Bruno. **Inteligência artificial na medicina: desafios e oportunidades.** Disponível em: https://tinyurl.com/yc5hytre. Acesso em: 13 abr. 2025.

[10] UNIVERSITY OF CALIFORNIA BERKELEY. **What is machine learning?** Disponível em: https://tinyurl.com/95bhhj4k. Acesso em: 15 abr. 2025.

[11] UNIVERSIDADE FEDERAL DE SANTA CATARINA. **Reconhecimento de fala com redes neurais convolucionais.** Disponível em: https://tinyurl.com/y7euwp86. Acesso em: 24 abr. 2025.

[12] INSTITUTO FEDERAL DO PARANÁ. **Reconhecimento de fonemas com redes neurais convolucionais.** Disponível em: https://tinyurl.com/3sj2yhrc. Acesso em: 03 jun. 2025.

[13] TENSORFLOW. **YAMNet tutorial.** Disponível em: https://tinyurl.com/52pu59ew. Acesso em: 20 mai. 2025.

[14] TENSORFLOW. **YAMNet README.** Disponível em: https://tinyurl.com/5f76rufa. Acesso em: 20 mai. 2025.

[15] GOOGLE RESEARCH. **AudioSet Dataset.** Disponível em: https://tinyurl.com/4sh2yh7n. Acesso em: 19 abr. 2025.

[16] DESHPANDE, Shreyas. **Audio Deep Learning made simple (Part 2): Why mel spectrograms perform better.** Disponível em: https://tinyurl.com/3d554zmz. Acesso em: 27 abr. 2025.

[17] GOOGLE RESEARCH. **YAMNet output of a sample of BioSpeech with the temporal representation.** Disponível em: https://tinyurl.com/7ffevsya. Acesso em: 03 jun. 2025.

[18] PYPI. **SpeechRecognition.** Disponível em: https://tinyurl.com/5n968dv5. Acesso em: 21 abr. 2025.

[19] REAL PYTHON. **Python Speech Recognition.** Disponível em: https://tinyurl.com/ycafx8t6. Acesso em: 25 abr. 2025.

[20] RASPBERRY PI FOUNDATION. **Raspberry Pi 4 Model B.** Disponível em: https://tinyurl.com/3fns8adk. Acesso em: 15 mai. 2025.

[21] RASPBERRY PI FOUNDATION. **Raspberry Pi Software.** Disponível em:

DOI: 10.25286/repa.v11i4.3983

https://tinyurl.com/muzdszrd. Acesso em: 15 mai. 2025.

[22] ELPROCUS. **Diferença entre ESP32 e Raspberry Pi.** Disponível em: https://tinyurl.com/mpaphmuy. Acesso em: 15 mai. 2025.

[23] ONISTEK. **Microfone sem fio MC811.** Disponível em: https://tinyurl.com/53twfcph. Acesso em: 17 mai. 2025.

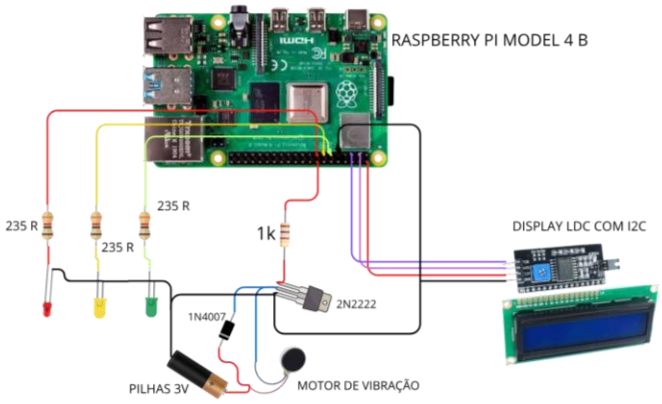
[24] AMAZON. **Placa de som externa 7.1 canais ST3-5.** Disponível em: https://tinyurl.com/2hyehj48. Acesso em: 17 mai. 2025.

[25] BLOG DA ROBÓTICA. **Como utilizar o display LCD 16x02 com módulo I2C no Arduino.** Disponível em: https://tinyurl.com/y4sk8ey4. Acesso em: 21 mai. 2025.

[26] MICRO ELECTRONICS. **Datasheet 2N2222.** Disponível em: https://tinyurl.com/msuxxy54. Acesso em: 26 mai. 2025.

[27] SURGE. **Datasheet 1N4007.** Disponível em: https://tinyurl.com/n97bmwt9. Acesso em: 26 mai. 2025.

[28] INSTITUTO DIGITAL. **Motor de vibração 1027 para Arduino.** Disponível em: https://tinyurl.com/39ynhuye. Acesso em: 29 mai 2025.



ANEXO 1 – Equipamento completo

ANEXO 2 – Esquema elétrico