Uma Arquitetura de Sistema Embarcado Utilizando técnicas de Deep Learning e Sinais de Vibração para Diagnóstico de Falhas em Rolamentos de Motores Elétricos

Luiz A. Pinto * **Marco Antônio de S. L. Cuadros** *
Lucas de O. Soares *

* *Instituto Federal do Espírito Santo, Programa de Pós-Graduação em Engenharia de Controle e Automação, (e-mail: lucasosoares@hotmail.com, marcoantonio@ifes.edu.br, pintoluizalberto@gmail.com).*

Abstract: This work addresses the topic of the development of embedded systems applied to fault detection in electric motor bearings based on vibration signals. The system uses a CNN-M network for feature extraction and a LSTM network for classification, and was embedded in a STMicroelectronics development board B-L475E-IOT01A that uses the microcontroller STM32L475VGT6. To evaluate the system performance, metrics related to classification performance were used, as well as computational metrics of memory requirements and processing time. The results obtained in classification (accuracy in tests of 99.65%) indicate the feasibility of developing embedded systems for fault detection and diagnosis in electric motor bearings. Regarding computational metrics, it was found that the use of CNNs in the feature extraction phase, that require large amounts of parameters for their configuration, can increase memory and processing time requirements.

Resumo: Esse trabalho aborda o tema do desenvolvimento de sistemas embarcados para detecção de falhas em rolamentos de motores elétricos baseados em sinais de vibração. O sistema utiliza uma rede *CNN-M* para extração de características e uma rede *LSTM* para classificação, e foi embarcado em uma placa de desenvolvimento B-L475E-IOT01A da *STMicroelectronics* que utiliza o microcontrolador STM32L475VGT6. Para avaliação do desempenho do sistema foram utilizadas métricas relacionadas ao desempenho na classificação, bem como métricas computacionais de requisitos de memória e tempo de processamento. Os resultados obtidos, (acurácia nos testes de 99,65% indicam a viabilidade do desenvolvimento de sistemas embarcados para detecção e diagnóstico de falhas em rolamentos de motores elétricos. Em relação às métricas computacionais, verificou-se que a utilização de CNNs na fase de extração de características, que requerem grandes quantidades de parâmetros para sua configuração, pode elevar os requisitos de memória e tempo de processamento.

Keywords: Fault diagnosis; machine learning; wavelet transform; feature extraction; vibration analysis; mafaulda; pattern recognition.

Palavras-chaves: Diagnóstico de falhas; aprendizado de máquinas; transformada wavelet; extração de características; análise de vibração; mafaulda; reconhecimento de padrões.

1. INTRODUÇÃO

Motores elétricos são responsáveis por cerca de 40% de todo o consumo de energia elétrica no mundo. Suas boas condições de funcionamento dependem, grandemente, das condições de conservação dos mancais de rolamento. Os mancais de rolamento têm como função dar suporte mecânico às partes rotativas, reduzir o coeficiente de atrito do movimento e garantir a precisão da rotação. Portanto, uma vez que exista um defeito no rolamento, a estabilidade de todo o conjunto pode ser afetada, provocando danos ao equipamento e condições inseguras.

As abordagens atuais mais comumente usadas para detecção e diagnóstico de falhas em rolamentos consistem na análise dos sinais de vibração e na classificação desses por ferramentas de *Machine Learning*. Uma vez que falhas em rolamentos alteram o regime de vibração, a aquisição e a análise desses sinais pode fornecer informações importantes sobre o estado de conservação desses componentes.

Uma variedade de abordagens para elaboração de diagnósticos inteligentes de falhas em rolamentos de motores elétricos tem sido proposta na literatura. Por exemplo, Redes Neurais Artificiais (Lima et al., 2013), Redes Neurais Convolucionais (Eren et al., 2018), processamento de sinais de vibração utilizando a Transformada Wavelet (Liu et al., 2018), e aplicação de outras arquiteturas de redes profundas (Liu et al., 2018) e (Markiewicz et al., 2019). Esses métodos têm atingido altos níveis de precisão

na identificação de defeitos e na estimação da vida útil (*RUL - Rest of Useful Life*). Porém, trazem também custos computacionais cada vez mais elevados, para ganhos cada vez mais marginais em precisão.

As principais etapas para o diagnóstico de falhas são: 1. aquisição dos sinais do processo; 2. processamento dos sinais e 3. classificação. Tradicionalmente estas etapas são realizadas em modo *off-line* e remoto. Nesse caso, a etapa de aquisição dos dados é executada de forma separada das outras duas. A aquisição é realizada *on-site*, no motor, e posteriormente em um computador *off-site* as etapas de processamento e diagnóstico são executadas. Diferentemente dos sistemas *off-line* remotos, nos sistemas *on-site* as etapas de aquisição, processamento e classificação ocorrem de forma contínua, no equipamento, permitindo a elaboração de diagnósticos em tempo real, o que possibilita intervenções imediatas para mitigar os efeitos das falhas.

Com a difusão do conceito de *IIoT* (*Industrial Internet of Things*) e de Indústria 4.0, pode-se especular sobre o aumento da utilização de sistemas embarcados microcontrolados e *SBCs* (*Single Board Computers*). Além disso, paralelamente observou-se significantes avanços em conectividade, eficiência energética e capacidade de processamento e de memória dos *hardwares* para sistemas embarcados disponíveis atualmente. Embora sistemas embarcados sejam uma realidade em várias áreas da tecnologia, são raros os trabalhos sobre diagnóstico de falhas desenvolvidos para tais sistemas operando *on-site*, devido a ainda recente convergência da evolução de classificadores e melhorias da capacidade computacional de microcontroladores.

Esse trabalho avalia o desempenho de um sistema embarcado para realizar o diagnóstico de falhas em rolamentos de motores a partir da análise de sinais de vibração. O sistema é constituído por uma rede neural convolucional para a extração de características, e uma rede *LSTM* para a classificação das falhas. Nesse trabalho, diferentemente da maioria dos trabalhos até então publicados sobre diagnóstico de falhas utilizando sistemas embarcados, a avaliação do desempenho, além da acurácia na classificação, considerou métricas relacionadas a demanda computacional e o tempo de processamento, parâmetros que, no caso de sistemas embarcados, podem ser tão relevantes quanto a capacidade de classificação do sistema.

O restante deste trabalho contém as seguintes seções: a Seção 2 apresenta uma breve revisão da literatura sobre diagnóstico de faltas em rolamentos e a implementação desses sistemas utilizando *hardwares* embarcados. A Seção 3 faz uma suscinta revisão da teoria de *CNN* e *LSTM*. A Seção 4 apresenta as ferramentas e métodos utilizados para a realização da pesquisa. Os resultados e a discusão dos mesmos estão apresentados na Seção 5. Por fim, a Seção 6 contém as conclusões do estudo.

2. TRABALHOS CORRELATOS

O desenvolvimento de técnicas de inteligência artificial aplicadas à detecção de falhas tem resultado no aparecimento de vários artigos sobre o tema. Trabalhos abrangendo a aplicação de *Machine Learning* em sistemas embarcados têm sido propostos, porém, ainda são poucos em comparação com o desenvolvimento de soluções propostas para *desktops* e servidores. Em Tzeng (2018) é apresentado um sistema embarcado para captura e pré-processamento de dados de vibração, que então são enviados por uma rede sem fio e monitorados remotamente. Em Hmida and Braham (2016) é implementado um algoritmo de monitoramento de motores, através de sinais de vibração e transformada *wavelet*, embarcado em um microcontrolador.

Em Gongora et al. (2018), uma RNA embarcada é desenvolvida para detectar falhas no rolamento através do sinal da corrente elétrica. Em Eren et al. (2018) os autores apresentam redes neurais convolucionais compactas capazes de classificar falhas em rolamentos através de sinais de vibração, utilizando conjuntos de dados públicos (como o CWRU) para validar o trabalho. Em Markiewicz et al. (2019) os autores apresentam uma solução embarcada completa de baixo consumo para classificação de falhas utilizando redes *Long Short-Term Memory*.

Em Lu et al. (2020) os autores classificam 7 tipos de falha, além do funcionamento normal em motores de indução. Para tal, utilizam 8 motores, cada um em uma condição de falha ou funcionamento normal. Dados de vibração foram adquiridos a uma taxa de 20 kHz e três testes de classificação são realizados em um sistema microcontrolado. No primeiro teste, imagens 28 x 28 são geradas a partir dos sinais e uma CNN com duas camadas convolucionais é utilizada para a classificação. No segundo os autores usam as mesmas imagens 28 x 28 e uma CNN com uma camada convolucional para classificação, e no terceiro teste geram imagens 14 x 14 e as classificam utilizando a CNN com duas camadas convolucionais. As acurácias alcançadas são 100%, 97,5% e 94,13% para cada teste, respectivamente.

Os autores em (Chen et al., 2020) comparam os desempenhos de duas redes neurais com diferentes *kernels* para extrair descritores de sinais de vibração de rolamentos de motores elétricos. Em uma etapa anterior a classificação, os sinais de vibração form subamostrados a fim de reduzir a quantidade de variáveis. Uma *LSTM* foi utilizada para a classificação das faltas ocorridas no processos, tendo os autores relatado uma acurácia de 98,46%.

3. REFERENCIAL TEÓRICO

Uma breve revisão conceitual das redes *CNN* e *LSTM*, utilizadas para a implementação do sistema é apresentada a seguir.

3.1 Redes Neurais Convolucionais - Convolutional Neural Network (CNN)

Redes Neurais Convolucionais são constituídas por diversas camadas sendo uma de entrada, uma de saída e várias camadas intermediárias alternadas com diferentes funções (Liu et al., 2017).

A Figura 1 apresenta a arquitetura básica de uma CNN. A camada convolucional é a responsável pela extração das informações e sua representação em forma de características numéricas. A camada *Max Pooling* é responsável pela realização de uma subamostragem dividindo a entrada em regiões retangulares menores e calculando o valor representativo para a região considerada, reduzindo a dimensão da imagem para as etapas seguintes. A *Dropout Layer* atribui

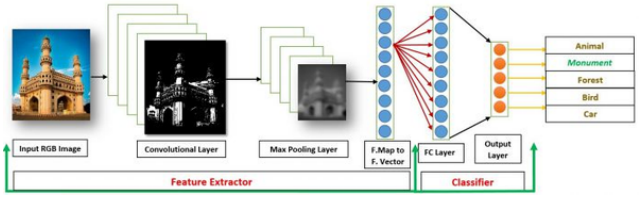

Figura 1. Arquitetura Básica de uma Rede Neural Convolucional. Disponível em: https://www.serpro.gov.br/. Acesso em jun. 2020.

aleatoriamente, durante a etapa de treinamento, valores de entrada para zero conforme uma dada probabilidade. É utilizada para evitar sobreajuste da rede para um dado conjunto de treinamento. A camada totalmente conectada (*Fully Connected Layer*) possui todos os neurônios conectados a todos os neurônios da camada anterior, combinando todas as características aprendidas pelas camadas anteriores para identificar padrões. Rede neurais convolucionais têm sido amplamente utilizadas em aplicações de reconhecimento de padrões e visão computacional (Eren et al., 2018).

3.2 Long Short-Term Memory - LSTM

As redes neurais *Long Short-Term Memory - LSTM* (Hochreiter and Schmidhuber, 1997) são variações da arquitetura das redes neurais recorrentes (RNN), proposta para superar limitações dessas. Uma das características que as diferenciam das arquiteturas convencionais de redes neurais recorrentes, é a sua capacidade de memorizar informações em intervalos arbitrários. A *LSTM* é organizada em uma estrutura em cadeia constituída por blocos de processamento e blocos de memória chamados células, conforme pode ser visto na Figura 2.

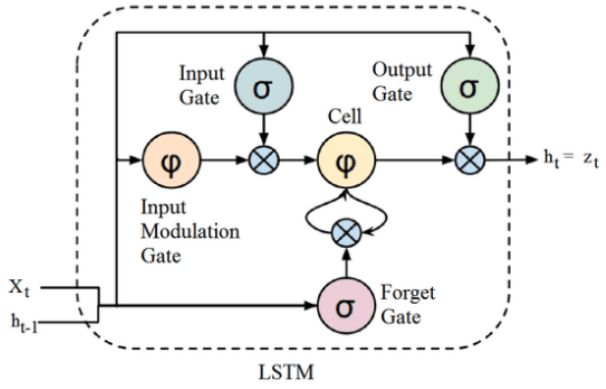

Figura 2. Célula de uma Long Short Term Memory
Fonte: Dias et al. (2018).

A informação é retida pelas células, e as manipulações de memória são realizadas por três tipos de portas *(gates)*. Na *Forget Gate* as informações que não são mais úteis para um determinado estado da célula são removidas. Se a saída desse bloco for 0, a informação é esquecida, se for 1, a informação é retida para uso futuro. Na *Input Gate* ocorre a adição de informações úteis ao estado da célula. A *Output Gate*, por sua vez, tem a função de extrair informações úteis do estado atual da célula para serem apresentadas como uma saída.

4. METODOLOGIA

Nesta seção está descrito o conjunto de dados utilizado para a realização dos experimentos, os modelos de extração de características e classificação das falhas, e as características do *hardware* utilizados para embarcar o sistema.

4.1 O Conjunto de Dados CWRU

O conjunto de sinais de vibração *CWRU*, foi gerado nos laboratórios da *Case Western Reserve University* em Ohio, EUA e pode ser obtido livremente no repositório de dados da referida universidade. Para a sua aquisição, experimentos foram realizados em uma bancada de teste equipada com um motor elétrico de 2 hp, conforme mostrado na Figura 3. Utilizando eletroerosão (EDM - *electro-discharge machining*), foram introduzidas falhas com dimensões entre 0,007 e 0,028 polegadas na pista interna, pista externa e nas esferas dos rolamentos. Os rolamentos defeituosos foram reinstalados e foram coletados 10 segundos de sinal de vibração no motor com cargas variando entre 0 e 3 hp acopladas em seu eixo (velocidades do motor entre 1797 e 1720 rpm).

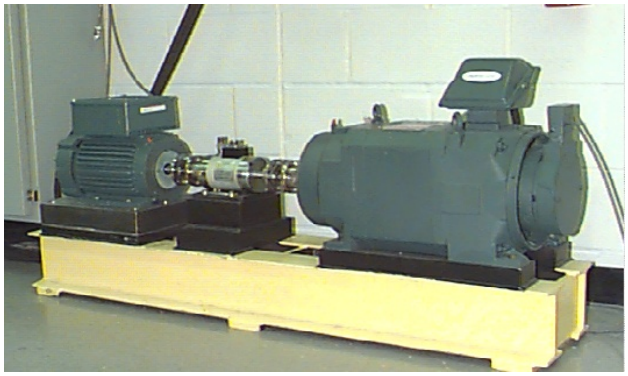

Figura 3. Equipamento utilizado na aquisição de dados do conjunto CWRU.
Fonte: Case Western Reserve University

O conjunto completo é constituído por 161 arquivos contendo os sinais de vibração coletados em 3 posições diferentes (lado do eixo, lado do ventilador e base), bem como as velocidade de rotação do motor em que os sinais foram coletados. Para a realização desse trabalho, foram utilizados os sinais de vibração coletados pelo sensor posicionado no eixo do motor, a 48 kHz, para todas as falhas com dimensão de 0,021 polegadas no rolamento do eixo do motor, totalizando 24 sinais cobrindo 6 posições de defeitos e 4 cargas/velocidades. Os sinais foram agrupados de acordo com a posição dos defeitos, independente da carga/velocidade de rotação, originando 6 classes diferentes.

A estrutura do modelo para detecção e diagnóstico das falhas é constituído por dois módulos. O módulo de extração de descritores, implementado por uma Rede Neural Convolucional, e o módulo de classificação, que consiste em uma rede *Long Short Time Memory (LSTM).*

4.2 Rede extratora de características CNN-M

A rede neural convolucional utilizada no módulo de extração de características está mostrada na Figura 4. A

Tabela 1. Classes do CWRU.

Classe	Posição do defeito
$C6$	Sem defeito, condição normal
$C5$	Esfera
$C4$	Pista interna
$C3$	Pista externa ortogonal
$C2$	Pista externa centralizado
$C1$	Pista externa oposto

rede *CNN-M* é uma *CNN multi-scale* proposta por Chen et al. (2020), composta por duas *CNNs* unidimensionais em paralelo, *CNN-H* e *CNN-L*. A *CNN-L* que tem campos receptivos maiores, é a responsável pela extração de características relacionadas à baixas frequências. Por sua vez, a *CNN-H* tem campos receptivos menores para extrair características de frequências mais altas do sinal de vibração. As características extraídas pelas *CNNs*, separadamente, são combinadas através de uma camada de multiplicação ponto a ponto entre os vetores de saída da *CNN-L* e *CNN-H*. Os parâmetros da configuração dessa rede estão mostrados na Tabela 2.

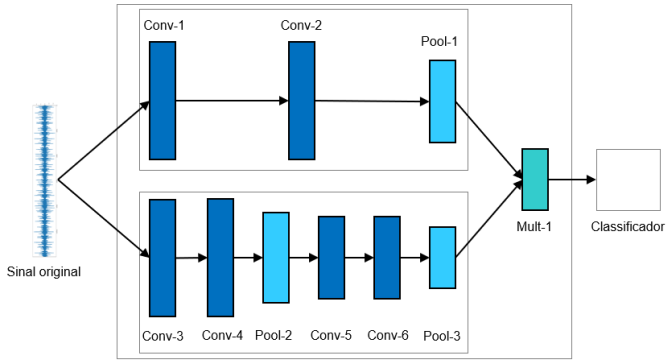

Figura 4. Estrutura da CNN-M.
Fonte: Autoria própria

Tabela 2. Parâmetros de Configuração da Rede CNN-M

Camada	Nº Filtros	Kernel/Stride	F Ativação
Convolucional-1	50	20/2	tanh
Convolucional-2	30	10/2	tanh
Pooling-1		2/2	
Convolucional-3	50	6/1	tanh
Convolucional-4	40	6/1	tanh
Pooling-2		2/2	
Convolucional-5	30	6/1	tanh
Convolucional-6	30	6/2	tanh
Pooling-3		2/2	
Total de Parâmetros			41.130

4.3 Rede classificadora LSTM

O módulo de classificação é implementado por uma rede *LSTM* hierárquica seguida por uma camada totalmente conectada de saída, cuja estrutura é ilustrada na Figura 5. A Tabela 3 mostra o número de parâmetros utilizados para a implementação dessa rede.

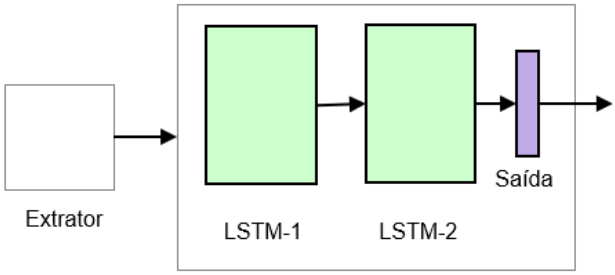

Figura 5. Estrutura da LSTM.
Fonte: Autoria própria

Tabela 3. Parâmetros da Rede LSTM

Camada	Unidades	Função Ativ.
LSTM-1	60	ReLU
LSTM-2	30	ReLU
Saída	10	
Total de parâmetros		32.226

4.4 Hardware - O Sistema Microcontrolado

Como este trabalho não se limitou a implementar técnicas e modelos para serem executados em computadores *desktop*, foi necessário utilizar um sistema computacional consistindo de *hardware*, *firmware* e *softwares* de desenvolvimento para microcontroladores STM32.

Para realizar a classificação utilizando os modelos apresentados, foi escolhida a placa de desenvolvimento B-L475E-IOT01A da *STMicroelectronics* (Figura 6), que utiliza o microcontrolador STM32L475VGT6. Este microcontrolador conta com 1 *Mbyte* de memória *flash*, 128 *Kbyte* de memória SRAM, clock principal de até 80 MHz, unidade de ponto flutuante (FPU) padrão IEEE 754, e tem um microprocessador *Arm Cortex-M4* de ultra-baixo consumo e alto desempenho, utilizando arquitetura ARMv7E-M com extensão de instruções para processamento digital de sinais (DSP).

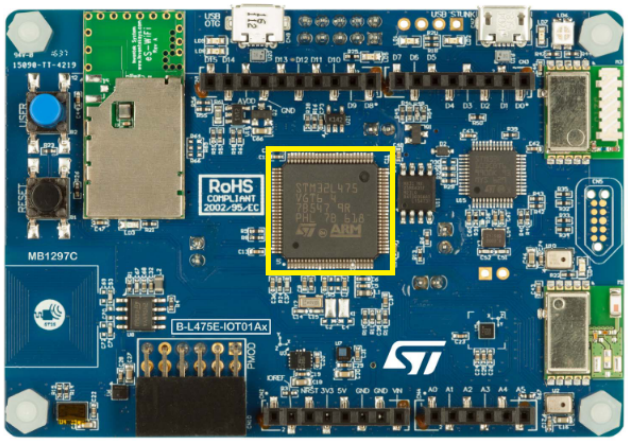

Figura 6. Placa de desenvolvimento B-L475E-IOT01A.
Fonte: Adaptado de STMicroeletronics

4.5 Software e Firmware

A classificação das falhas foi realizada por um *firmware* embarcado no microcontrolador. Para isso, foi utilizada a biblioteca *CMSIS-NN* disponibilizada pela própria Arm, e

a ferramenta *X-Cube-AI* da *STMicroelectronics*, integrada ao ambiente de desenvolvimento STM32CubeIDE para microcontroladores da família STM32. Para a classificação das falhas, o *firmware* executa a seguinte rotina básica: recebimento dos dados referentes ao sinal que deve ser classificado -> extração de características do sinal -> classificação -> envio do resultado da classificação.

4.6 Teste Multi-plataforma

Os testes realizados envolveram duas etapas: o treinamento e a classificação. Devido a limitação de recursos computacionais do microcontrolador, o treinamento do modelo, etapa com maior demanda, foi realizado em um computador *desktop*, e para tal, foram utilizadas funções implementadas na linguagem *Python*. Posteriormente, após o treinamento, estas redes foram transformadas e inseridas no *firmware* utilizando as ferramentas mencionadas na subseção anterior. A Figura 7 apresenta a rotina executada nessa fase do projeto. Em um computador *desktop* (1) todos os sinais (a) são divididos em treinamento (b) e teste (c). Os dados de treinamento são processados e utilizados para treinar os classificadores (d). O *firmware* com a rede treinada (e) é embarcada no microcontrolador (2), que recebe os dados de teste e os classifica. Finalmente o resultado da classificação é enviado ao desktop para cálculo das métricas de desempenho (f).

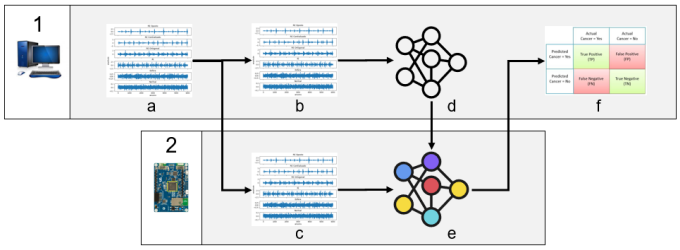

Figura 7. Fluxo de treinamento e teste multi-plataforma.
Fonte: Autoria própria

É importante lembrar que em aplicações reais a aquisição dos dados seria realizada pelo próprio dispositivo embarcado posicionado junto ao motor elétrico monitorado, e que o resultado da classificação seria transmitido via comunicação remota, preferencialmente sem fio, eliminando por completo a necessidade de um computador *desktop* na rotina de identificação dos defeitos.

4.7 Treinamento

Os sinais brutos foram particionados em segmentos de 2000 pontos que constituíram as amostras que alimentaram a rede *CNN-M* para extração das características. Na fase de extração de características foram obtidas 5.760 amostras, sendo 960 por classe. Para o treinamento dos modelos foi utilizada a técnica *Holdout*, tendo as amostras sido particionadas em amostras de treino (70%) e teste (30%). A Figura 8 mostra o desempenho na fase de treino do modelo *LSTM*. O treinamento foi realizado em 20 épocas, com *batches* de 10 amostras. Pode-se notar que o valor da acurácia no final das 20 épocas é próximo da unidade. O treinamento foi realizado em um computador *desktop* com 24 Gb de memória *RAM* e um processador *AMD Ryzen 5 2400G*.

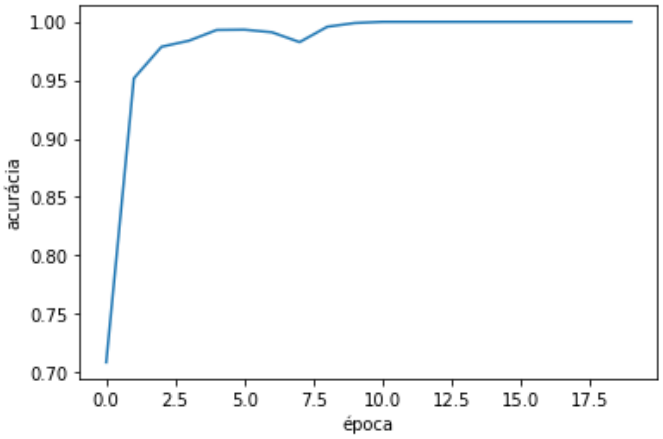

Figura 8. Acurácia de treino do modelo.
Fonte: Autoria própria

4.8 Métricas de Avaliação de Desempenho

Para a avaliação do desempenho do sistema foram consideradas métricas relacionadas à classificação, e métricas computacionais. Para a avaliação da classificação foi utilizada a acurácia. No contexto de sistemas embarcados microcontrolados, os recursos computacionais são escassos, sendo um fator limitante na viabilidade de modelos de classificação. Dessa forma, além das métricas de classificação, métricas de desempenho computacional dos modelos são tão importantes quanto e, dependendo da aplicação, mais relevantes do que as métricas de classificação. Quando a aplicação é limitada mais por preço, tamanho, ou consumo de energia do *hardware* do que pela acurácia de classificação, é importante ter métricas que auxiliem na escolha dos melhores algorítimos dentro daquelas limitações. As métricas computacionais aplicadas nesse trabalho se baseiam no *framework* desenvolvido pela *STMicroelectronics* para aplicação de inteligência artifical. Os parâmetros de desempenho do *STM32 Cube AI* são capacidade de armazenamento em *KiB*, complexidade computacional (chamada então de *MACC - multiply-accumulate complexity*), e tempo.

5. RESULTADOS

Nesta seção os resultados dos testes são apresentados em termos das métricas de desempenho na classificação, bem como em termos das métricas computacionais.

5.1 Resultados em Termos das Métricas de Classificação

A Figura 9 mostra a matriz de confusão da etapa de testes. O valor da acurácia foi 99,65%. Esse resultado mostra que a combinação da rede *CNN-M* para extração de caracerísticas com a *LSTM* para classificação produz um modelo com elevada capacidade de classificação das falhas. De fato, o bom desempenho da *CNN* na etapa de extração de características que, de forma geral, extrai as variáveis com melhor capacidade de descrição das falhas, favorece o desempenho da rede *LSTM* na etapa de classificação. Pode-se observar a partir da matriz de confusão que o modelo teve bom desempenho em todas as condições de funcionamento.

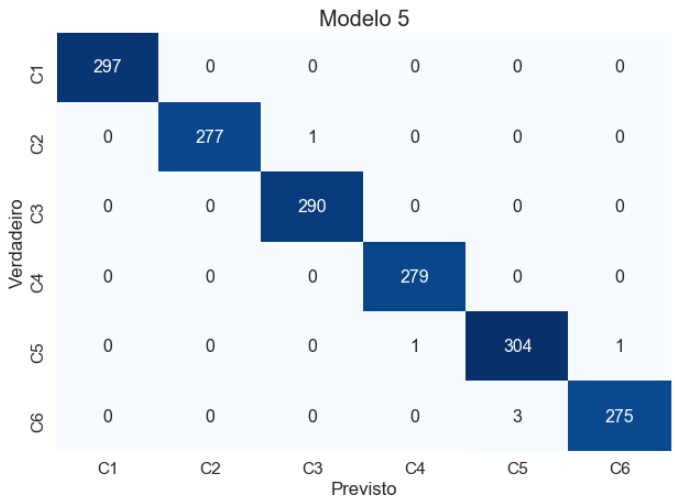

Figura 9. Matriz de confusão.
Fonte: Autoria própria

5.2 Resultados em Termos das Métricas Computacionais

Com o modelo embarcado no microcontrolador as métricas de demanda computacional foram extraídas para o modelo completo (módulo de extração de características e módulo de classificação) e para cada módulo individualmente (*CNN-M* e *LSTM*). Além disso, o tempo necessário para a classificação de cada amostra foi medida. Todos os valores das métricas computacionais foram calculadas utilizando a ferramenta STM32 Cube AI, e estão registrados na Tabela 4.

Tabela 4. Métricas computacionais - módulos

	ROM (KiB)	RAM (KiB)	MACC	tempo (ms)
CNN-M	160,66	62,97	5.364.080	2791,5
LSTM	127,88	11,29	906.630	171,7
Modelo	288,54	74,26	6.270.710	2981,8

Analisando os valores das métricas computacionais observa-se que, se por um lado, a utilização da rede *CNN-M* na etapa de extração de característica pode assegurar a extração de características com melhor capacidade de classificação, quando comparado aos métodos estatísticos, por outro lado, devido a grande quantidade de parâmetros necessários para a configuração da rede, o valor da *MACC*, e consequentemente do tempo computacional dessa etapa foi comparativamente alto em relação ao desempenho da rede *LSTM*.

6. CONCLUSÃO

Os resultados obtidos, considerando a métrica de classificação e as métricas computacionais relacionadas aos requisitos de tempo de processamento e memória, mostram ser viável a utilização de microcontroladores com técnicas de *Deep Learning* para classificação de defeitos em rolamentos de motores utilizando sinais de vibração em sistemas embarcados. A combinação das redes *CNN* e *LSTM* para implementação do sistema de classificação demonstrou-se eficiente para a classificação das falhas do sistema, tendo apresentado uma acurácia de 99,65%.

Outro aspecto importante a ser considerado é o alto valor de *MACC* e do tempo de processamento na fase de extração de características como resultado da utilização da *CNN-M*. Trabalhos futuros poderão considerar a utilização de técnicas estatísticas convencionais para essa tarefa, o que, se espera, reduzirão o valor da *MACC* e do tempo de processamento, sem redução significativa do desempenho do modelo na classificação.

REFERÊNCIAS

Chen, X., Zhang, B., and Gao, D. (2020). Bearing fault diagnosis base on multi-scale cnn and lstm model. *Journal of Intelligent Manufacturing*.

Dias, P.F., Pinto, L.A., Cavalieri, D.C., and Pereira, F.G. (2018). Utilização de Técnicas de Deep Learning para Reidentificação de Pessoas. João Pessoa, Paraíba, Brasil. doi:10.20906/CPS/CBA2018-0825. URL `http://www.swge.inf.br/proceedings/paper/?P=CBA2018-0825`.

Eren, L., Ince, T., and Kiranyaz, S. (2018). A generic intelligent bearing fault diagnosis system using compact adaptive 1d cnn classifier. *Journal of Signal Processing Systems*. doi:10.1007/s11265-018-1378-3.

Gongora, W.S., Goedtel, A., Castoldi, M.F., Oliveira da silva, S.A., and Da Silva, I.N. (2018). Embedded system to detect bearing faults in line-connected induction motors. In *2018 XIII International Conference on Electrical Machines (ICEM)*, 1841–1847.

Hmida, M.A. and Braham, A. (2016). Arm based rswpt implementation for embedded condition monitoring of induction motor. In *IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society*, 1464–1469.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8), 1735–1780.

Lima, A.A.d., Prego, T.d.M., Netto, S.L., da Silva, E.A., Gutierrez, R.H., Monteiro, U.A., Troyman, A.C., Silveira, F.J.d.C., and Vaz, L. (2013). On fault classification in rotating machines using fourier domain features and neural networks. In *2013 IEEE 4th Latin American Symposium on Circuits and Systems (LASCAS)*, 1–4. IEEE.

Liu, J., Gao, X., Bao, N., Tang, J., and Wu, G. (2017). Deep convolutional neural networks for pedestrian detection with skip pooling. 2056–2063. IEEE. doi:10.1109/IJCNN.2017.7966103.

Liu, R., Yang, B., Zio, E., and Chen, X. (2018). Artificial intelligence for fault diagnosis of rotating machinery: A review. *Mechanical Systems and Signal Processing*, 108, 33–47.

Lu, S., Qian, G., He, Q., Liu, F., Liu, Y., and Wang, Q. (2020). In situ motor fault diagnosis using enhanced convolutional neural network in an embedded system. *IEEE Sensors Journal*, 20(15), 8287–8296.

Markiewicz, M., Wielgosz, M., Bocheński, M., Tabaczyński, W., Konieczny, T., and Kowalczyk, L. (2019). Predictive maintenance of induction motors using ultra-low power wireless sensors and compressed recurrent neural networks. *IEEE Access*, 7, 178891–178902.

Tzeng, C. (2018). Vibration detection and analysis of wind turbine based on a wireless embedded microcontroller system. In *2018 IEEE International Conference on Applied System Invention (ICASI)*, 133–136.