Diagnóstico de falha em rolamentos por meio de Deep Learning

Sergio Leandro P. de Queiroz Campos
Departamento de Eng. de Produção
CEERMA / UFPE
Recife, Brasil
sergio.leandro@ufpe.br

João Vitor Valadares de Moraes
Centro de Informática
MOTOROLA / CIN
Recife, Brasil
jvvm@cin.ufpe.br

***Abstract*— A análise de vibração em rolamentos é uma das formas bem promissoras de identificar a falha em equipamentos em máquinas rotativas. Geralmente, esse acompanhamento é manual. Automatizar esse processo pode garantir uma eficiência e tornar a equipe de manutenção mais eficiente no combate das falhas aumentando a disponibilidade dos equipamentos na planta industrial.**

Keywords*— *Rolamentos, manutenção, deep-learning, confiabilidade, vibração, processamento de sinais

I. OBJETIVOS

Em primeiro ponto, esse trabalho tem como objetivo desenvolver aos autores um senso técnico e analítico voltado a resolver problemas industriais por meio de técnicas oriundas da Inteligência Artificial. Dessa forma, como o objeto de estudo escolhido foram os rolamentos, inicialmente o trabalho visa o aprendizado sobre a análise de vibração dos mesmos e processamento de sinais captados por meio sensores. Quanto a parte do desenvolvimento do modelo de Deep-Learning (DL), esse trabalho visa estudar e analisar o diagnóstico da falha em rolamentos em equipamentos industriais utilizando técnicas de processamento dos sinais de vibração, geralmente captados por meio de acelerômetros, os quais serão tratados e utilizados como espectrogramas que servirão como input para um modelo de Redes Neurais Convolucionais (CNN).

Além disso, durante essa etapa, visa-se estudar diferentes arquiteturas de redes para essa problemática em questão.Por fim, entendemos que para o usuário comum, um modelo implementado simplesmente via código acaba não sendo muito atrativo e interessante. Dessa forma, por meio da biblioteca *stream.lit*, iremos implementar o *deployment* do modelo. Para o teste do modelo e para podermos enxergar como ele funciona na prática, por meio de variáveis aleatórias, iremos simular o processo de captação de dado os quais serão gerados dados genéricos de sinais (em estado degradado e não degradado) e, por fim, aplicar o modelo de CNN desenvolvido anteriormente e poder verificar o seu funcionamento em tempo real, podendo alterar parâmetros relacionados ao processo

II. JUSTIFICATIVA

Como veremos mais à frente na metodologia, esse trabalho, e o desenvolvimento de uma modelagem mais robusta quanto a manutenção de rolamentos em instalações industriais, visto que a falha desses componentes não causa imediatamente enormes custos financeiros a organização devido ao seu baixo custo unitário da peça. Contudo, a falha em um rolamento acarreta na baixa eficiência desse do equipamento o qual ela faz parte podendo ocasionar em maiores problemas como: aumento da amperagem para compensar problemas nos rolamentos, desarmes no equipamento devido a alta amperagem ou até mesmo desgaste nos equipamentos acarretando numa perda da eficiência permanente do equipamento.

Por outro lado, no quesito humano, o desenvolvimento de uma modelagem mais robusta no quesito de detecção de falhas serve também serve como um forte suporte a equipe de manutenção, visto que essa área geralmente é bastante atarefada nas indústrias geralmente e pode realocar profissionais especialista em análise de vibração para atuar em outro segmentos da organização.

III. METODOLOGIA

A. O caminho da manutenção inteligente

A manutenção é uma das áreas cruciais dentro de uma indústria e muitas vezes quando é esquecida, os custos são imensos para a organização. A manutenção é classicamente dividida de três formas: corretiva, preventiva e preditiva. No primeira, a equipe de manutenção atua como “bombeira” apagando “ incêndios” ao longo da instalação industrial, no segundo caso, se atua de maneira preventiva realizando cronogramas de substituição dos componentes ao longo do tempo evitando com que ocorra esses equipamentos venham a falhar após o tempo de troca dos componentes e a preditiva visa estudar quando essa falha virá a acontecer. No contexto de manutenção 4.0, uma nova etapa e classe de

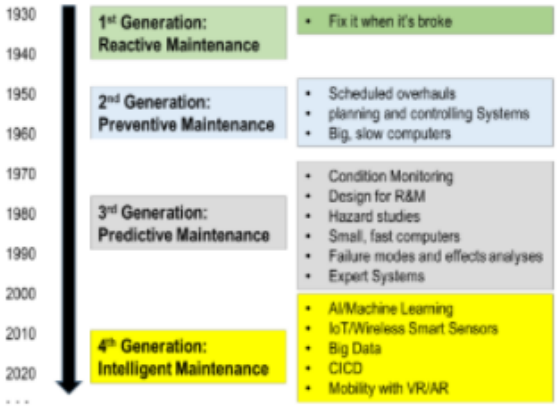

Figura 1. O caminho da manutenção inteligente

manutenção surge a chamada: manutenção inteligente que lida diretamente com a ampla massa de dados que são gerados diariamente por meio de sensores que podem ser captados e tratados em tempo real por meio de Internet das Coisas (IoT) [2]. Podemos perceber esse caminho e avanço ao longo do tempo com base na imagem abaixo:

B. Análise de Falha em Rolamentos

Rolamentos estão presentes em máquinas simples até máquinas mais complexas, quando comparável ao custo da máquina ou de um equipamento produtivo o rolamento representa importância quase que desprezível. Entretanto, um rolamento de poucos reais pode paralisar ou causar uma ineficiência no processo produtivo paralisando equipamentos de milhões de reais.

A falha desses componentes, pode ser decorrente dos seguintes fatores:

1. Sobrecarga.
2. Desbalanceamento.
3. Variações bruscas de temperatura.
4. Lubrificação inadequada.
5. Partículas abrasivas ou corrosivas no lubrificante.
6. Erro de projeto utilizando rolamento inadequado à função.
7. Desgaste pelo uso (fadiga do material).

A falha presente em um rolamento pode ser apresentada tanto internamente, esferas ou roletes, como de maneira externa. Como geralmente os redutores e engrenagens estão apoiados em rolamentos via eixos, as vibrações vão causar danos nos mesmos. Essa vibração pode ser captada por meio de sensores acoplados ao equipamento que iram medir a vibração presente no modelo, normalmente, na indústria técnicos especialistas em análise de vibração são alocados para atividades de monitoramentos dos sinais o que acaba sendo bastante trabalho e passível de falha, visto que depende do humano interpretar sinais em forma gráfica e entender se aquele rolamento se encontra em um estado degradado ou não degradado.

C. Base de dados utilizada: MFPT

O *dataset* utilizado para esse projeto foi organizado, construído por meio do Machinery Failure Prevention (MFPT) society [3]. Para a construção deste *dataset* foram amostrados dados com uma alta frequência de amostragem para três diferentes condições em que os rolamentos poderiam se encontrar: com falhas no *inner race,* no *outer race* e também em boas condições de operação.

Um equipamento de teste em boas boas condições foram coletados dados aceleração coletados para as condições de a 270 libras de carga e uma taxa de amostragem de 97.656 Hz durante seis segundos. No total, dez falhas no *outer race* e sete no *inner race* condições de falha foram rastreadas. Três falhas de *inner race* incluem 270 libras de carga e uma taxa de amostragem de 97.656 Hz por 6 segundos. Sete falhas no *outer race* adicionais foram avaliadas em cargas variadas: 25, 50, 100, 150, 200, 250 e 300 libras. A taxa de amostragem para as falhas foi de 48.828 Hz por três segundos. Sete falhas de *inner race* foram analisadas com cargas variadas de 0, 50, 100, 150, 200, 250 e 300 libras. A taxa de amostragem para as falhas no *inner race* foi de 48.848 Hz por três segundos. Os tipos de falha nos rolamentos podem ser visualizados na figura 2, abaixo, sendo o primeiro rolamento em boas condições, o segundo com falha no *inner race* e o terceiro com falhas no *outer race.* O estado no qual o rolamento se encontra, emitirá um sinal de vibração, captado por meio de acelerômetros, que indicaram um padrão daquele estado, como podemos ver na figura 3, esse sinal bruto capturado desse analisado e tratado de forma com que possamos extrair informação deste e entender se deve ser realizada alguma atividade de manutenção.

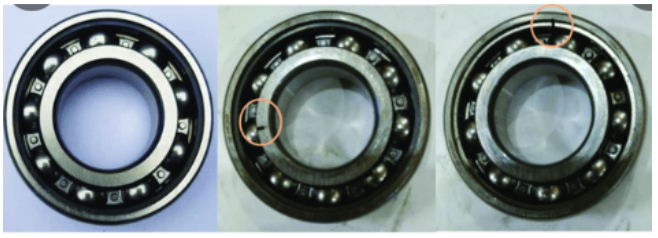

Figura 2. Tipos de falha em rolamentos.

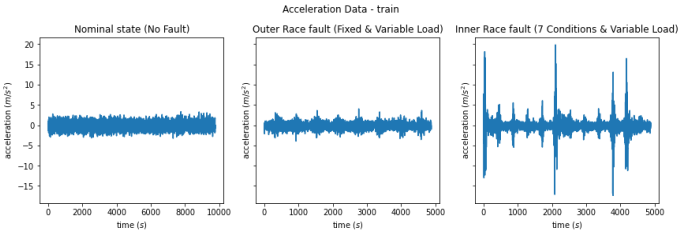

Figura 2. Sinais brutos de aceleração por estado de falha.

D. Metodología tradicional x Metodología por feature learning

De acordo com [1], o diagnóstico de falha por meio de análise de vibração análise de vibração em rolamentos por meio de algoritmos de classificação pode ser realizada de duas formas distintas. A primeira é dita tradicional, por meio dos dados brutos captados por meio de acelerômetros, são captadas informações da distribuição desses dados. Essas informações são captadas por meio de métricas que serão geradas pelo modelo, tais como: média, variância, média quadrática, amplitude, kurtosis, skewness entre outras métricas e momentos estatísticos que podem ser gerados. Essas métricas são construídas por meio de *chunks* da base de dados que devem ser definidos pelo analista que está desenvolvendo o trabalho, vale ressaltar que estão completamente ligados da taxa de amostragem do sensor utilizado. Após o processo de serem calculados as *features*, temos um dataset que pode ser utilizado por meio de algoritmos tradicionais de *machine learning*. Por outro lado, outra abordagem que vem se mostrado bastante promissora nesse campo, é a utilização de métodos tempo-frequência, como STFT e WT os quais veremos a seguir na próxima sessão, que vem se mostrando mais eficientes nesse processo. Dessa forma, primeiramente os dados são organizados em forma de *chunks,* da mesma forma que os modelos tradicionais, após esse processo para cada *chunk* é realizada uma STFT ou WT que irão retornar de forma gráfica espectrogramas ou escalogramas. Por meio dessas imagens geradas ao longo do processo, podemos utilizar uma CNN que irá fazer uma extração de características das imagens e após esse processo irá realizar a classificação da imagem.

E. Métodos de tempo-frequência

Metódos de tempo-frequência representam os sinais tanto no tempo quanto no domínio da frequência tendo suas representações mais comuns um espectrograma e escalogramas. O primeiro, trata-se de uma representação no do domínio tempo-frequência utilizando, *short-time fourier transform (STFT),* no qual no eixo x temos o tempo e no y a frequência, como podemos observar por meio da figura 3. Quanto ao segundo, os escalogramas representam a visualização gráfica de uma *wavelet transform (WT)* o qual traz uma representação linear do tempo-frequência, por conta do aspecto que essa transformada apresenta a variável de escala em sua composição ela acaba sendo eficiente para sinais transientes e não-estacionários, como podemos ver na figura 4.

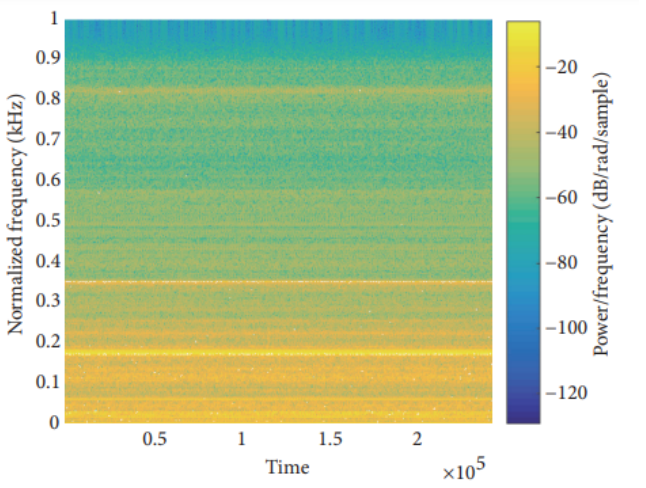

Figura 3. Espectograma STFT de sinais raw

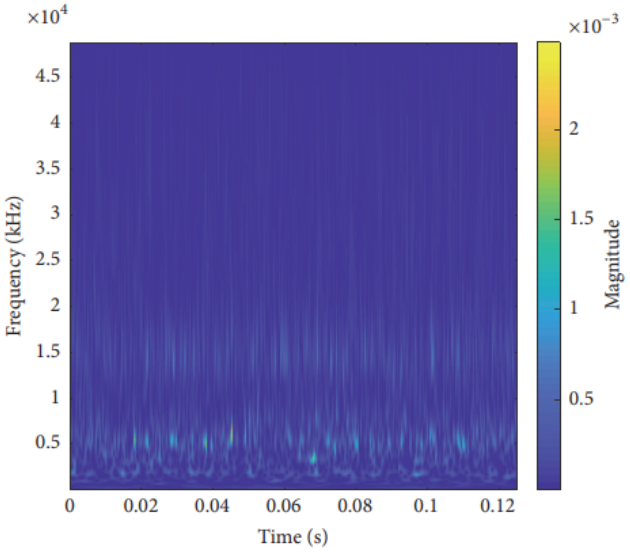

Figura 4. Escalograma CWT de sinais raw

Com base nas imagens construídas por meio das transformadas seja por meio de STFT ou WT, elas serão construídas numa etapa de construção do modelo, com base nisso, essas imagens servirão de input para uma modelo de deep learning, nesse caso, alguma arquitetura de CNN e por fim serão avaliadas por meio de técnicas tradicionais de análise de modelos, como podemos ver na imagem 4 [1].

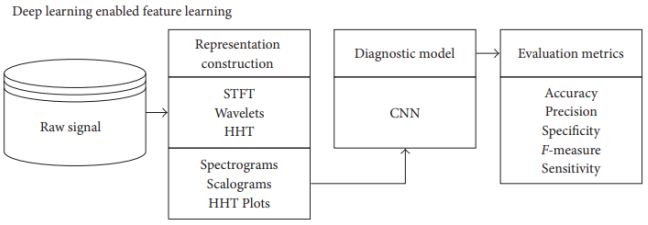

Figura 5. Esquema de utilização de DL usando espectogramas e escalogramas

D. Redes neurais convolucionais (CNN)

Em Deep Learning, a forma mais comum de aprendizado é a de aprendizado supervisionado, ou seja, temos dados que possuem *label* antes de serem utilizados como *input* do algoritmo. E dentro da comunidade de computação visual, temos um tipo de rede profunda preferida que consiste de camadas adjacentes completamente conectadas: a CNN. A arquitetura de uma CNN consiste de fases, cada uma cumprindo um diferente papel e cada papel sendo completado automaticamente com o algoritmo. Quatro propriedades constituem as arquiteturas de modelos de CNN: múltiplas camadas, *pooling/subsampling*, pesos compartilhados e conexões locais. A primeira fase da CNN consiste de dois tipos de camadas: camadas convolucionais que organizam as unidades em *feature maps* e camadas de *pooling* que unem *features* semelhantes em uma única *feature*. Na camada convolucional do mapeamento de *features*, cada unidade é conectada a um mapeamento de *features* de uma camada anterior através de um filtro que consiste basicamente de um conjunto de pesos e uma soma local correspondente de pesos, soma essa que utilizamos uma função não-linear como a unidade linear retificada (ReLU).

Um ponto importante a se ressaltar das camadas convolucionais para análise de imagens é que unidades de mesmo mapa de *features* compartilham o mesmo *filter bank*. Então para poder lidar com a possibilidade da localização do mapa de *feature* não ser o mesmo para toda imagem, diferentes mapas de *feature* utilizam diferentes *filter banks*. O segundo estágio da CNN consiste de uma camada de *pooling* responsável de unir *features* similares em uma só. Esse processo efetivamente reduz as dimensões da representação. Após múltiplas pilhas de camadas serem finalizadas, a saída desse processo pode ser alimentado na última fase do CNN, uma camada totalmente conectada de um Perceptron Multicamadas (MLP), que é uma rede neural de classificação *feedforward*. As saídas da camada final de *pooling* são utilizadas como entrada para mapear as *labels* providas para os dados. Portanto, a análise e predição das imagens de vibração são uma série de representações do sinal bruto [1].

Para o desenvolvimento do nosso projeto, foi considerada a arquitetura proposta por [1] visto que essa de acordo com o artigo apresentou boas métricas (acurácia, precisão e especificidade) para o diagnóstico de falhas por meio de escalas gramas e espectrogramas. Dessa forma, como *input* para a arquitetura temos os *datasets* de espectrogramas e escalogramas gerados anteriormente os quais foram divididos em 70% para treinamento e 30% para teste, sendo essas imagens possuindo uma resolução de 96x96. Em seguida, temos duas camadas convolucionais contendo 32 *features maps* cada geradas por meio de um *kernel* 3x3. Em seguida, foi realizado um *max pooling* e depois temos mais duas camadas convolucionais contendo cada uma 64 *features maps,* também com um *kernel* 3x3, logo em seguida, é realizado novamente um *max pooling*. Por fim, temos mais duas camadas convolucionais, com 128 *features maps*. Após isso, é realizado um *flatten* e um dropout (.5) é realizado, após esse processo duas camadas densas contendo 100 neurônios são conectadas a rede contendo entre elas dois dropouts (.5) e então temos 3 saídas para as redes que representam o diagnóstico do problema (falha no *inner race*, falha no *outer race* ou sem falha). A construção da rede CNN neste trabalho foi realizada por meio da biblioteca *tensorflow.keras*.

IV. RESULTADOS

A. Processamento dos dados em Features Tradicionais

Com a base de dados dividida em chunks com 512 dados brutos em cada um, foram extraídas as seguintes informações estatísticas: média, variância, amplitude, *peak to peak* e média quadrática. As informações foram tabelas e, além disso, adicionamos uma coluna representando a categoria, classe relacionada. Dessa forma, podemos treinar e ajustar esses dados para algum modelo de machine learning tradicional. Como podemos ver na tabela 1.

	average	variance	max_amplitude	rms	peak_to_peak	category
0	-0.069958	0.736731	2.760705	0.861176	5.721649	0
1	-0.134192	0.756117	2.312626	0.879843	5.039613	0
2	-0.109054	0.865251	2.475260	0.936560	5.238192	0
3	-0.082351	0.637332	2.456825	0.802567	4.825000	0
4	-0.132110	0.811191	2.346098	0.910299	5.490275	0
...	...	...	...	...	...	...
10863	-0.197145	0.948817	4.078002	0.993822	7.649424	2
10864	-0.217298	0.516480	2.701354	0.750799	5.545836	2
10865	-0.186529	1.039268	3.864078	1.036369	8.055035	2
10866	-0.151504	13.984358	19.836850	3.742634	39.299030	2
10867	-0.203518	0.625160	2.687350	0.816443	5.390180	2

10868 rows × 6 columns

Tabela 1. Base de dados com features tradicionais

B. Processamento dos dados em espectrogramas (STFT) e escalogramas (WT)

Dividindo a base de dados, também em *chunks* contendo 512 dados agrupados, construímos os espectrogramas para cada uma das classes, conforme podemos ver abaixo nas figuras 5,6,7.

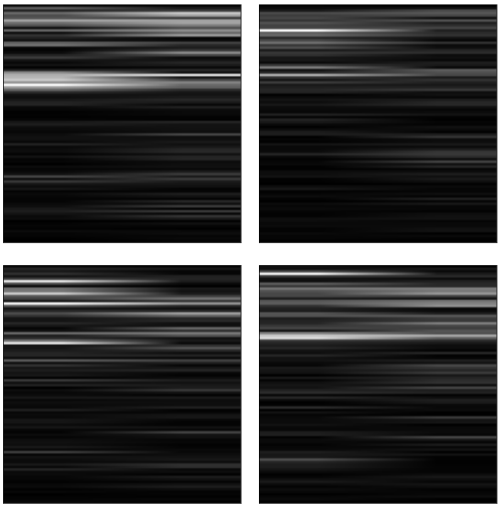

Figura 5. Espectrogramas em estado sem falha

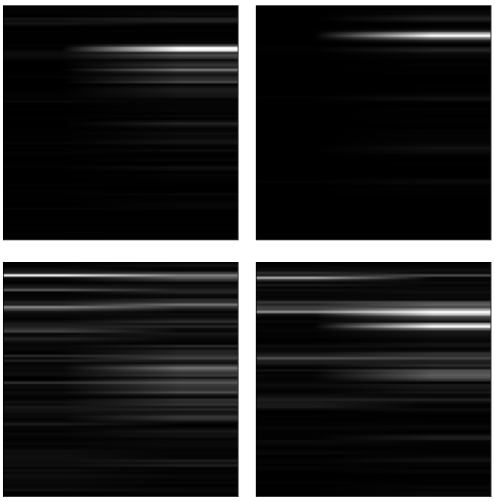

Figura 6. Espectrogramas com falha no *inner race*

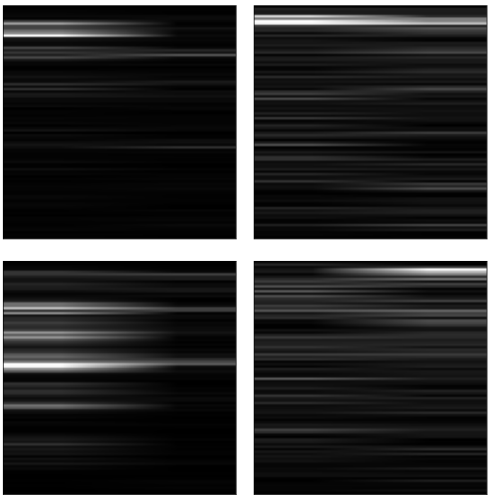

Figura 6. Espectrogramas com falha no *outer race*

Como podemos analisar de uma forma mais aguçada, podemos perceber que cada estado em que se encontra rolamento possui um padrão visual distinto nas figuras, como podemos observar. Note também que essas imagens encontram-se de uma forma um tanto diferente que as imagens mostradas anteriormente sobre espectrogramas, isso se deve ao fato da construção das imagens, escolha de parâmetros utilizados, mas ambos carregam a mesma informação gráfica.

Da mesma forma, construímos as escalogramas os quais foram construídas também utilizando chunks de 512 dados brutos da base de dados original. Como podemos observar nas imagens 7, 8 e 9. Note que também eles se apresentam de uma forma diferente que os explicitados anteriormente pela mesma razão dos espectrogramas.

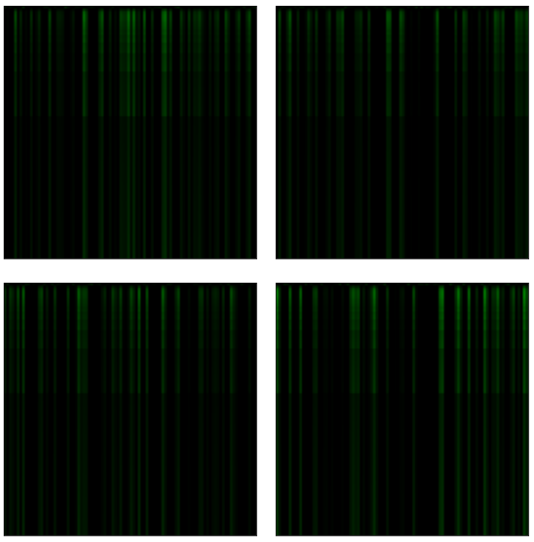

Figura 7. Escalogramas em estado sem falha

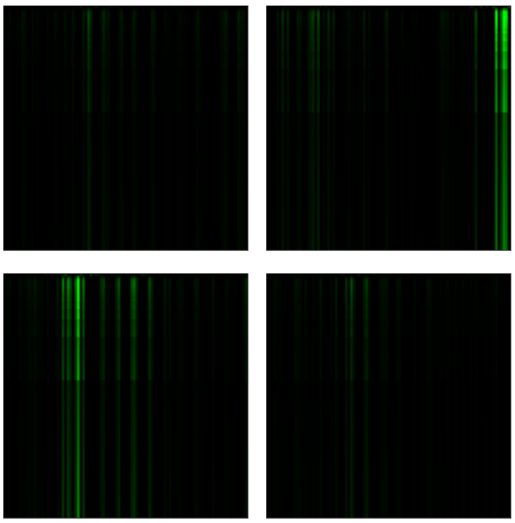

Figura 8. Escalogramas com falha no *inner race*

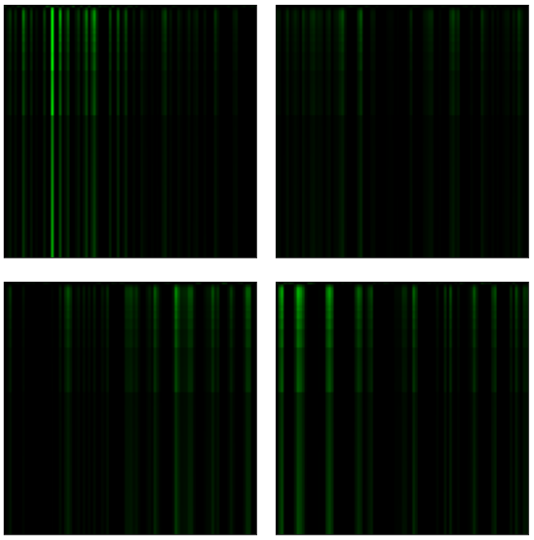

Figura 9. Espectrogramas com falha no *outer race*

C. Resultados por meio de modelos tradicionais de Machine Learning

Para essa análise foi utilizada a biblioteca Pycaret que nos retorna um compilado de modelos de classificação tradicionais com suas métricas avaliadas utilizando validação cruzada, por meio de *k-folds* (tendo como padrão k =10). Os 14 modelos avaliados estão presentes na tabela 2, na qual tiveram o *gradient boosting classifier* com melhor desempenho em termos de acurácia.

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
gbc	Gradient Boosting Classifier	0.8436	0.9438	0.8439	0.8451	0.8437	0.7464	0.7470	4.467
et	Extra Trees Classifier	0.8373	0.9389	0.8401	0.8389	0.8374	0.7368	0.7374	0.835
rf	Random Forest Classifier	0.8366	0.9376	0.8380	0.8378	0.8366	0.7355	0.7360	2.014
lightgbm	Light Gradient Boosting Machine	0.8359	0.9397	0.8391	0.8376	0.8360	0.7348	0.7355	0.347
ada	Ada Boost Classifier	0.7928	0.7849	0.8033	0.8009	0.7912	0.6704	0.6765	0.607
knn	K Neighbors Classifier	0.7862	0.8929	0.7731	0.7874	0.7857	0.6505	0.6513	0.199
dt	Decision Tree Classifier	0.7748	0.8086	0.7745	0.7753	0.7748	0.6335	0.6337	0.106
qda	Quadratic Discriminant Analysis	0.7388	0.9025	0.7212	0.7618	0.7310	0.5782	0.5931	0.043
lr	Logistic Regression	0.7327	0.8596	0.7082	0.7413	0.7320	0.5530	0.5568	1.829
nb	Naive Bayes	0.7158	0.8818	0.6918	0.7397	0.7055	0.5396	0.5553	0.033
lda	Linear Discriminant Analysis	0.6662	0.8100	0.6304	0.6824	0.6581	0.4261	0.4413	0.023
svm	SVM - Linear Kernel	0.6585	0.0000	0.6627	0.6877	0.6267	0.4384	0.4786	0.194
ridge	Ridge Classifier	0.6342	0.0000	0.5776	0.6613	0.6150	0.3553	0.3844	0.035
dummy	Dummy Classifier	0.5001	0.5000	0.3333	0.2501	0.3334	0.0000	0.0000	0.015

```
GradientBoostingClassifier(ccp_alpha=0.0, criterion='friedman_mse', init=None,
                           learning_rate=0.1, loss='deviance', max_depth=3,
                           max_features=None, max_leaf_nodes=None,
                           min_impurity_decrease=0.0, min_impurity_split=None,
                           min_samples_leaf=1, min_samples_split=2,
                           min_weight_fraction_leaf=0.0, n_estimators=100,
                           n_iter_no_change=None, presort='deprecated',
                           random_state=2294, subsample=1.0, tol=0.0001,
                           validation_fraction=0.1, verbose=0,
                           warm_start=False)
```

Tabela 2. Modelos tradicionais avaliados

Avaliando esse modelo, gradient boosting, para a base de dados de teste, nesse trabalho consideramos sempre a base de teste equivalente a 30% dos dados inclusive para as imagens, temos os seguintes resultados para o modelo.

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC
0	Gradient Boosting Classifier	0.8369	0.9441	0.8342	0.8373	0.8369	0.7348	0.735

	average	variance	max_amplitude	rms	peak_to_peak	category	Label	Score
0	-0.103877	0.449748	1.724398	0.678630	4.297882	1	1	0.9938
1	-0.145433	0.690559	3.642562	0.843629	7.207019	1	1	0.9359
2	-0.065268	0.447807	2.388922	0.672359	4.521589	1	1	0.9834
3	-0.089480	0.762825	2.309833	0.877970	5.229233	0	0	0.9182
4	-0.084164	0.819547	3.827431	0.909192	6.366201	1	1	0.6572
...	...	...	...	...	...	...	...	...
3256	-0.143703	0.745788	2.800440	0.875464	5.300877	0	0	0.7672
3257	-0.149462	1.177986	6.528404	1.095594	12.616691	1	1	0.8493
3258	-0.181893	2.938915	11.946370	1.723949	21.216528	1	2	0.6422
3259	-0.229052	0.632955	2.493349	0.827901	5.500342	2	1	0.6262
3260	-0.203702	0.719686	3.842961	0.872456	7.292519	1	1	0.5751

3261 rows × 8 columns

Tabela 3. Resultado do modelo para dos dados de teste

Note que a acurácia para a base de dados de teste e treino foi muito próxima com diferença de menos 1% o que indica que o modelo está generalizando bem. Contudo, uma acurácia por volta de 83,369% pode representar um sério problema ao ser utilizada na análise do diagnóstico de falha de rolamentos visto que indica que em média 1 em cada 5 diagnósticos realizados ocorre uma análise errada. Considerando o cenário em que o rolamento não está falha e o modelo de diagnóstico indica que há falha, ocorrerá a parada do equipamento (por exemplo, um compressor) para a troca do rolamento, o que pode afetar drasticamente a produtividade. Dessa forma, é necessário que o modelo seja mais assertivo e eficiente.

D. Resultados por meio da CNN

Por meio das imagens construídas na etapa de processamento e com arquitetura de [1] explicitada no tópico de CNN presente neste relatório foram treinados dois modelos utilizando, para esses treinamentos tivemos como parâmetros um *batch_size* de 256 e o modelo foi treinado durante 10 épocas. Nos gráficos 1,2,3 e 4 podemos ver o desempenho da acurácia e da loss ao longo das épocas tanto para o treinamento por meio dos espectrogramas quanto dos escalogramas para as base de treino e de teste, como podemos observar que não há overfitting e underfitting e o treinamento ocorre muito bem. Na tabela 3, estão apresentadas as acuracias relacionadas a cada um dos modelos. Diferentemente do trabalho de [1], o modelo com espectrograma performou melhor do que os resultados apresentados no artigo. Isso provavelmente se deve ao fato de que do pré-processamento dos dados ter diferenças, as quais não são explicadas no artigo de [1], como também devido a aleatoriedade da definição dos pesos de uma CNN. Contudo, ambas as acurácias finais foram acima de 98% o que indica que são modelos bem interessantes para serem utilizados na indústria pois apresentam uma alta assertividade ao contrário do modelo tradicional mais eficiente demonstrado anteriormente.

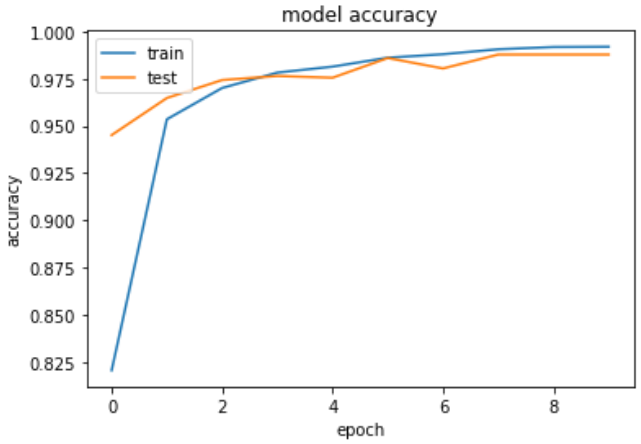

Gráfico 1. Acurácia ao longo das épocas para o modelo com espectrogramas

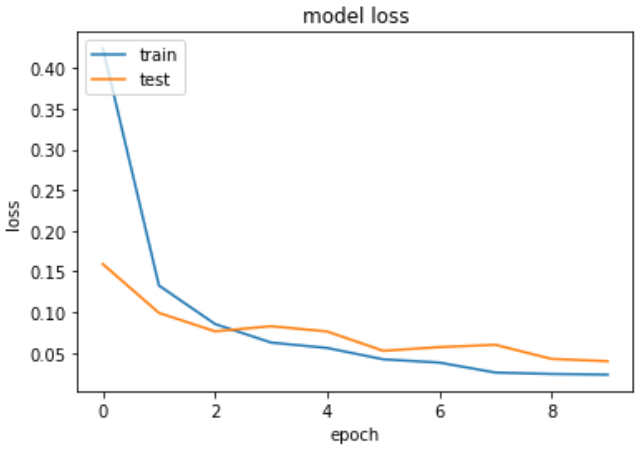

Gráfico 2. Loss ao longo das épocas para o modelo com espectrogramas

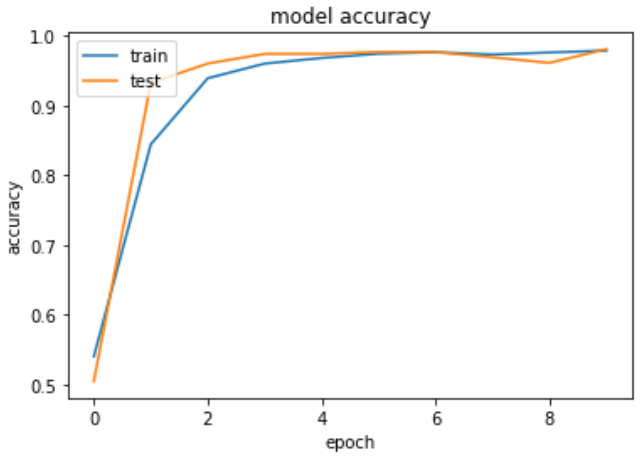

Gráfico 3. Acurácia ao longo das épocas para o modelo com escalogramas

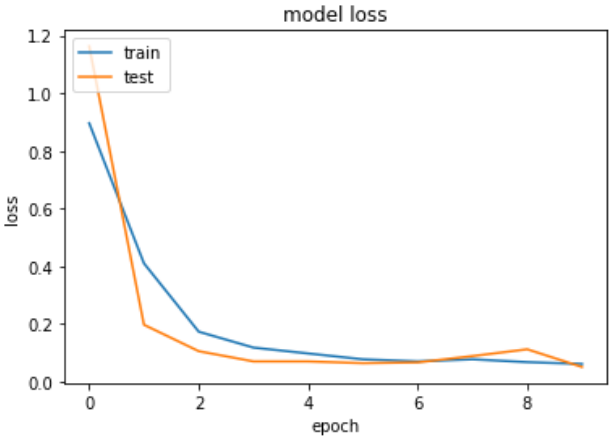

Gráfico 4. Loss ao longo das épocas para o modelo com escalogramas

Modelo	Acurácia
Espectrogramas	98,77%
Escalogramas	99,3%

Tabela 3. Resultados presentes na CNN

V. CONCLUSÕES E SUGESTÕES PARA TRABALHOS FUTUROS

O diagnóstico em falhas de rolamentos é um problema bem significativo na indústria. E a detecção dessas falhas é essencial para ter uma economia nos gastos. Aplicações anteriores de deep learning nesse tipo de problema tinham certos problemas quanto à sensibilidade dos dados, então a CNN proposta e aplicada em nosso estudo foi aplicada à representações de séries temporais de frequência de sinais, sendo eles espectrograma e escalograma e os comparamos. Foi verificado qual era mais efetivo. A arquitetura proposta se mostrou bem robusta e efetiva na resolução do problema. Ela nos trás acurácia alta nos dois métodos, tendo melhor desempenho quando a análise é realizada com imagens de espectrograma, porém com escalograma bem próximo. Pode ser interessante como estudos futuros utilizar modelos sequenciais e fazer experimentos, visto que esse problema de falha de rolamentos são séries temporais, podendo trazer bons resultados. Diagnóstico de falhas de rolamentos é uma área que vem evoluindo constantemente e é essencial que a indústria adote medidas para se beneficiar desde o começo desse desenvolvimento que só tende a avançar, tendo assim uma iniciativa inovadora dentro da empresa.

REFERENCES

[1] David Verstraete, Andrés Ferrada, Enrique López Droguett, Viviana Meruane, Mohammad Modarres, "Deep Learning Enabled Fault Diagnosis Using Time-Frequency Image Analysis of Rolling Element Bearings", *Shock and Vibration*, vol. 2017, Article ID 5067651, 17 pages, 2017. https://doi.org/10.1155/2017/5067651

[2] Haining Zheng, Antonio R. Paiva, Chris S. Gurciullo, “Advancing from Predictive Maintenance to Intelligent Maintenance with AI and IoT”, Arxiv, arXiv:2009.00351

[3] Bechhoefer, E., A Quick Introduction to Bearing Envelope Analysis, MFPT Data, http://www.mfpt.org/FaultData/FaultData.htm.Set, 2016.