CLASSIFICAÇÃO DE FALHAS EM MÁQUINAS ROTATIVAS UTILIZANDO MÉTODOS DE SIMILARIDADE E *RANDOM FOREST*

Matheus Araújo Marins

Projeto de Graduação apresentado ao Curso de Engenharia Eletrônica e de Computação da Escola Politécnica, Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Engenheiro.

Orientadores: Sérgio Lima Netto
Felipe Moreira Lopes Ribeiro

Rio de Janeiro
Setembro de 2016

CLASSIFICAÇÃO DE FALHAS EM MÁQUINAS ROTATIVAS UTILIZANDO MÉTODOS DE SIMILARIDADE E *RANDOM FOREST*

Matheus Araújo Marins

PROJETO DE GRADUAÇÃO SUBMETIDO AO CORPO DOCENTE DO CURSO DE ENGENHARIA ELETRÔNICA E DE COMPUTAÇÃO DA ESCOLA POLITÉCNICA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE ENGENHEIRO ELETRÔNICO E DE COMPUTAÇÃO.

Examinado por:

Prof. Sérgio Lima Netto, Ph.D.

Felipe Moreira Lopes Ribeiro, M.Sc.

Prof. Wallace Alves Martins, D.Sc.

Prof. Amaro Azevedo de Lima, D.Sc.

RIO DE JANEIRO, RJ – BRASIL
SETEMBRO DE 2016

Araújo Marins, Matheus

Classificação de Falhas em Máquinas Rotativas Utilizando Métodos de Similaridade e *Random Forest*/Matheus Araújo Marins. – Rio de Janeiro: UFRJ/ Escola Politécnica, 2016.

XV, 52 p.: il.; 29, 7cm.

Orientadores: Sérgio Lima Netto

Felipe Moreira Lopes Ribeiro

Projeto de Graduação – UFRJ/ Escola Politécnica/ Curso de Engenharia Eletrônica e de Computação, 2016.

Referências Bibliográficas: p. 50 – 52.

1. Similaridade. 2. Manutenção Preditiva. 3. *Random Forest*. 4. Máquinas Rotativas. I. Lima Netto, Sérgio *et al.* II. Universidade Federal do Rio de Janeiro, Escola Politécnica, Curso de Engenharia Eletrônica e de Computação. III. Título.

A minha família e aos meus amigos.

Agradecimentos

Certamente essa é a parte mais importante do trabalho a ser apresentado. Muitas pessoas fizeram parte da minha trajetória, e eu realmente acredito que sem qualquer uma delas o caminho teria sido muito mais complicado, ou até mesmo impossível de ser trilhado.

Agradeço à toda minha família, por todo carinho, atenção e suporte que me forneceram. Em especial à minha mãe, sempre muito solícita, generosa e disposta a me dar todo o conforto possível; ao meu pai, sempre fazendo todo o possível para me dar condições de estudar e vibrando com cada vitória minha; a Graça, por toda amizade e carinho; e aos meus tios Marconi e Marcioni, pela amzidade, e toda ajuda.

Agradeço também ao meu irmão, que desde quando eu tenho alguma memória eu lembro dele fazendo tudo por mim, mais por mim do que por ele. Foi ele que me apresentou o estudo e fez com que eu seguisse pela área das exatas.

Agradeço a todos os funcionários do Departamento de Eletrônica, em especial aos professores Teodósio, José Gabriel, Eduardo Silva, Wallace Martins, Luiz Wagner e Marcello Campos.

Agradeço ao professor José Carlos D'Ávila, que exerceu o papel de coordenador do curso de Engenharia Eletrônica e de Computação durante quase toda a minha graduação. Tendo cumprido sua função de forma magnífica, sendo não apenas um porto seguro, mas também um amigo, sempre livrando os alunos de assuntos alheios a engenharia.

Agradeço ao meu orientador e amigo Sérgio, por todos ensinamentos e conselhos. Obrigado pela confiança e pela paciência, e por ter me ajudado a crescer, tanto como pessoa quanto como profissional. Espero ser capaz de retribuir com bons resultados.

Agradeço fortemente ao amigo Felipe Curicica Ribeiro, que também fez parte da minha orientação e com quem pude aprender mais coisas do que eu imaginei que fosse capaz em tão pouco tempo.

Agradeço a todos os funcionários do Laboratório de Sinais, Multimídia e Telecomunicações (SMT), por terem me acolhido e por terem criado um ambiente diferenciado para o estudo e para a troca de ideias e conhecimento.

Agradeço a todos os *fakes* que me ajudaram a aguentar a faculdade, superando cada adversidade imposta a nós. Em especial, devo agradecer àqueles que definitiva-

mente fizeram possível eu me formar (ou ao menos me economizaram 1 ou 2 anos), e pelos quais guardo carinho especial: Felipe, Rafael, Roberto, Igor, Maria Zeneide, Luciane, Nasser e Vinícius.

Agradeço aos meus amigos do Severo, por toda amizade, companheirismo e confiança que vocês me proporcionaram. Espero poder contar com cada um de vocês ao meu lado por muito anos.

Por fim, agradeço aos professores Amaro Azevedo de Lima e Wallace Alves Martins, que aceitaram o convite para compor a banca avaliadora deste trabalho.

Resumo do Projeto de Graduação apresentado à Escola Politécnica/ UFRJ como parte dos requisitos necessários para a obtenção do grau de Engenheiro Eletrônico e de Computação.

Classificação de Falhas em Máquinas Rotativas Utilizando Métodos de Similaridade e *Random Forest*

Matheus Araújo Marins

Setembro/2016

Orientadores: Sérgio Lima Netto
Felipe Moreira Lopes Ribeiro

Curso: Engenharia Eletrônica e de Computação

Este trabalho trata da detecção automática de falhas em máquinas rotativas por meio de métodos baseados em similaridade e utilizando *Random Forest*. O sistema foi avaliado em uma base de dados criada com o auxílio de um *Rotorkit*, o qual foi submetido a diversas falhas, gerando dados suficientes para o treinamento do modelo e para a sua posterior validação. O sistema é completamente descrito, e buscou-se pelo melhor conjunto de parâmetros a fim de otimizar o sistema para o reconhecimento de falhas. Resultado próximo a 98.5% é alcançado, com ocorrência de erros principalmente em casos onde não ocorre falha, devido ao fato da escassez de amostras com a máquina em funcionamento normal.

Palavras-chave: Similaridade, Máquinas rotativas, Diagnóstico de falhas, Manutenção preditiva, *random forest*, Aprendizado de máquina.

Abstract of Undergraduate Project presented to POLI/UFRJ as a partial fulfillment of the requirements for the degree of Engineer.

FAULT DIAGNOSIS USING SIMILARITY BASED METHODS AND RANDOM FOREST

Matheus Araújo Marins

September/2016

Advisors: Sérgio Lima Netto
Felipe Moreira Lopes Ribeiro

Course: Electronic Engineering

This work deals with the automatic fault detection on rotating machines using similarity based methods and the well-known Random Forest. The system was validated on a database created utilizing a Rotorkit. The Rotorkit was submitted to a reasonable number of faults, generating enough data to train the model and for its subsequent validation. The system is fully detailed throughout this project, followed by a search for its best set of parameters parameters. The final configuration was able to achieve results up to 98.5%, with misclassification errors occurring basically when the machine is operating with no faults, due to the low quantity of samples from this operation mode.

Keywords: Similarity, Rotating machines, Fault diagnosis, Predictive maintenance, Random Forest, Machine learning

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Símbolos

R_f	Frequência de rotação estimada do rotor
$s(n)$	n-ésima amostra do sinal discreto no tempo
F_s	Frequência de amostragem
H	Entropia
M_X	Média da distribuição X
K	Curtose
$\mathbf{V}_i$	Saída do bloco de pré-processamento referente à amostra i
$\mathbf{D}_K$	Matriz de estados representativos do SBM referente à classe K
$\mathbf{G}$	Matriz de transformação do treinamento do SBM
$\mathbf{I}_d$	Matriz identidade de dimensões $d \times d$
$\mathbf{x}_{\text{obs}}$	Vetor de entrada do SBM
$\mathbf{x}_{\text{est}\|c}$	Estimativa para o vetor de entrada $\mathbf{x}_{\text{obs}}$ pada dada classe c
ℓ_p	Norma de ordem p
γ	Parâmetro das funções de similaridade
$\mathbf{w}$	Vetor de pesos utilizado para estimar $\mathbf{x}_{\text{est}\|c}$ a partir de $\mathbf{D}_c$
$\mathbf{r}_{\text{est}\|c}$	Resíduo referente à estimação do vetor $\mathbf{x}_{\text{obs}}$ para a classe c
$\mathcal{D}_K$	Conjunto formado exclusivamente por amostras da classe K
$\mathbf{v}_{\mathcal{M}}$	Mediana geométrica do conjunto $\mathcal{M}$
B	Total de árvores utilizadas no *Random Forest*
T_b	b-ésima árvore de decisão
$\hat{C}_b(\mathbf{x})$	Classificação do vetor $\mathbf{x}$ pela árvore T_b
$n_{features}$	Número de *features*
p	Número de *features* utilizadas em cada árvore do *Random Forest*
k	Número de *folds* utilizados na validação cruzada *k-fold*
τ	Limiar de similaridade para o método de treinamento do limiar para a matriz$\mathbf{D}$
m_k	Número amostras presentes em $\mathbf{D}_K$

Glossário

AAKR *Auto-Associative Kernel Regression*. 21, 24, 30, 31, 34, 35, 36

ABTV *Alignment/Balance Vibration Trainer*. xi, 5, 6, 7

ACC acurácia. 33

ANN *Artificial Neural Networks*. 45, 46, 48

CCK *Cauchy Kernel*. 20, 21, 34, 38

Defeito C Defeito na parte externa do mancal. 10, 11

Defeito B Defeito no elemento rolante do mancal. 10, 11

Defeito A Defeito na gaiola do mancal. 9, 11

DFT *Discrete Fourier Transform*. 13, 14

DH Falha de desalinhamento horizontal. 41

DSB Falha de desbalanceamento. 41

DV Falha de desalinhamento vertical. 41

EXP *Exponential Kernel*. 20, 21, 34

IES *Inverse Euclidean Similarity*. 20, 21, 34, 38, 48

IMK *Inverse Multiquadratic Kernel*. 20, 21, 34

LEDAV Laboratório de Ensaios Dinâmicos e Análise de Vibração. 1

MI Falha no mancal invertido. 41

MNI Falha no mancal não invertido. 41

Modelo 4 Função de similaridade IES, com $\gamma = 0.01$, utilizando norma ℓ_1 e método de treinamento do limiar com $\tau = 0.9$. 39

Modelo 3 Função de similaridade CCK, com $\gamma = 0.1$, utilizando norma ℓ_2 e método de treinamento do limiar com $\tau = 0.5$. 39

Modelo 2 Função de similaridade CCK, com $\gamma = 1$, utilizando norma ℓ_1 e método de treinamento original com amostragem = 7. 39

Modelo 1 Função de similaridade IES, com $\gamma = 0.01$, utilizando norma ℓ_1 e método de treinamento original com amostragem = 7. 39

NR Máquina atuando livre de falhas. 41

PDF Probability Density Function. 15

RBF *Radial Basis Function*. 20, 21, 34, 35

RPM rotações por minuto. 10

SBM *Similarity-Based Modeling*. xi, xii, 3, 4, 18, 20, 21, 22, 24, 29, 30, 31, 34, 35, 36, 37, 38, 42, 48

SMT Laboratório de Sinais, Multimídia e Telecomunicações. 1

SVM Support Vector Machine. 45, 48, 49

UFRJ Universidade Federal do Rio de Janeiro. 1

Capítulo 1

Introdução

1.1 Tema

Este trabalho analisa uma abordagem alternativa para a detecção de falhas em máquinas rotativas. O projeto foi realizado no Laboratório de Sinais, Multimídia e Telecomunicações (SMT) e o desenvolvimento da base de dados foi realizado em [1] no Laboratório de Ensaios Dinâmicos e Análise de Vibração (LEDAV), ambos localizados na Universidade Federal do Rio de Janeiro (UFRJ).

A base de dados trata-se de sinais coletados através de acelerômetros, microfone e um tacômetro em uma bancada experimental, a fim de simular as falhas mais recorrentes em máquinas rotativas, assim como a máquina operando em condições normais.

Após o processo de aquisição de dados são aplicadas algumas técnicas de pré-processamento com o objetivo de melhor descrever o sinal e, consequentemente, aumentar o desempenho na etapa de classificação. Feito isso, serão aplicadas técnicas de *machine learning* para classificar cada uma das amostras.

1.2 Delimitação

O crescimento da automatização ao longo dos anos trouxe um grande aumento do número de máquinas especializadas nas indústrias das mais variadas áreas de produção. Tais equipamentos são extremamente caros, e a cada vez que ocorre uma falha em um desses equipamentos, compromete-se a produção, o que pode acarretar em grande prejuízo.

Com isso, técnicas de monitoramento de falhas vêm tendo cada vez mais valor no mercado, atraindo a atenção dos setores de pesquisa e desenvolvimento de diversas empresas. No caso de máquinas rotativa, uma das formas de pesquisar e investigar o problema é através do uso de uma bancada experimental chamada de *Rotorkit*.

Nesta é possível simular diversas falhas de funcionamento que podem ocorrer devido o uso do equipamento.

Para criar um sistema que seja robusto a diversos defeitos, foram geradas falhas de desalinhamento, desbalanceamento e falhas nos mancais de rolamento. Assim, cobre-se a maior parte das falhas que usualmente ocorrem em máquinas rotativas [2].

1.3 Justificativa

Parar uma linha de produção por conta de falhas mecânicas causa reflexos diretamente no lucro da empresa. Esse problema pode ser muito mais crítico caso ocorra de forma inesperada, sem existir o suporte necessário à espera. Diagnosticar tais falhas de forma antecipada diminui o tempo de reparo e evita gastos desnecessários. Tais motivos fazem dessa abordagem algo valioso para qualquer empresa que faça uso de máquinas em suas linhas de produção.

Tratando-se de máquinas rotativas, este trabalho traz uma nova abordagem para o problema do diagnóstico automático de falhas. Fazer tais diagnósticos de forma automática e por meio de reconhecimento de padrões tem a grande vantagem de não exigir um ser humano para tomar essas decisões, tarefa que muitas vezes se mostra inviável. O modelo criado mostrou-se capaz de detectar as mais diversas falhas sobre as quais as máquinas rotativas estão sujeitas de forma extremamente eficaz.

O modelo a ser apresentado apresenta uma outra importante vantagem. Após a extração dos dados, todo o desenvolvimento do resto do sistema é feito de forma a não necessitar de um especialista na área no qual será aplicado, visto que o modelo é baseado em técnicas de aprendizado de máquina, sendo gerado de forma exclusivamente empírica.

As atuais indústrias fazem uso extensivo de máquinas rotativas e é esperado que a utilização desse tipo de maquinário continue crescendo. Então, por apresentar alta confiabilidade nos resultados e grande robustez, a técnica aqui apresentada torna-se uma boa alternativa para o diagnóstico automático de falhas, algo que certamente é buscado por empresas que querem se manter em uma posição de competitividade no mercado.

1.4 Objetivos

O objetivo desse trabalho é projetar um sistema que seja capaz de diagnosticar falhas em máquinas rotativas de forma automática. Por se tratar de algo crítico e que deve ser sempre diagnosticado, almeja-se um classificador com alta taxa de acerto, preferencialmente com baixa taxa de falso negativos (quando deixa-se de detectar

uma falha). Então, é necessário que o sistema receba informações relevantes dos sinais, com a menor redundância possível.

A abordagem escolhida é fazer uso de técnicas de similaridade para facilitar na classificação de cada amostra obtida, e para este papel foi escolhido o *Similarity-Based Modeling* (SBM). Essa abordagem busca mensurar quanto uma medida é similar a cada uma das falhas estudadas, auxiliando o processo de classificação posterior. Deseja-se também encontrar a melhor configuração para o SBM, dentre as diversas versões que o mesmo pode apresentar, assim como estudar a viabilidade do mesmo como classificador.

1.5 Metodologia

O sistema desenvolvido consiste basicamente em 4 partes: a aquisição dos sinais de vibração, a extração das *features* dos sinais obtidos, o diagnóstico das amostras por meio do SBM e a posterior classificação através do *Random Forest*.

O roteiro seguido pelo projeto a fim de alcançar os objetivos supracitados foi:

1. Estudo da base de dados a ser utilizada, assim como a busca na literatura por meios eficazes de abordar o problema em mão;
2. Desenvolvido de algoritmos para extrair as características relevantes dos sinais, a fim de evitar informações redundantes, assim como ruídos.
3. Estudo do SBM, incluindo as diferentes formas de se calcular a similaridade entre dois vetores, a escolha entre diferentes métodos para a realização da etapa de treino e também como passar as informações para o bloco seguinte;
4. Busca por um classificador eficaz que trabalhe de forma supervisionada. A decisão foi adotar o *Random Forest*;
5. Análise dos resultados obtidos, em busca da melhor configuração do modelo para o problema.

1.6 Descrição

A seguir, uma breve descrição de cada um dos capítulos do presente trabalho:

- Capítulo 2: será apresentada a base de dados utilizada para a validação do projeto a ser apresentado. Ela conta com 1951 experimentos, sendo alguns feitos em condições normais e os demais contando com a simulação de alguma falha.

- Capítulo 3: apresentação e descrição das características dos sinais que serão utilizadas no decorrer do projeto, assim como os motivos para utilizá-las.
- Capítulo 4: explicação do sistema, desde o bloco de pré-processamento, o SBM e o *Random Forest*. O bloco intermediário é detalhado, explicando as diversas formas de se treiná-lo e todas as possíveis variações.
- Capítulo 5: apresentação dos possíveis perfis de simulação, explorando cada um dos modelos para tratar do problema. Também são apresentadas as formas como serão avaliados os modelos. A seguir são apresentados os resultados na fase de treinamento e de testes.
- Capítulo 6: conclusões a respeito dos resultados obtidos e também alguns possíveis caminhos para continuar esse projeto.

Capítulo 2

Base de Dados - Máquinas Rotativas

As máquinas rotativas possuem vasta aplicabilidade na automatização da indústria. Suas aplicações variam desde o uso em sistemas de transformação de energia elétrica em mecânica até na utilização em eixos de veículos. Com isso, o estudo dessas máquinas torna-se então alvo de pesquisas e investimentos a fim de otimizar o seu funcionamento. A abordagem da manutenção preditiva chama a atenção uma vez que, ao se ter um sistema que alerta o operador sobre futuras falhas, o mesmo pode desligar a máquina e fazer os devidos reparos. Esse processo é essencial em indústrias de óleo e gás, por exemplo [3].

Portanto, evitar interrupções na linha de produção de uma fábrica, reduzir gastos com manutenções emergenciais e prolongar o tempo de vida útil da máquina simultaneamente soa como uma abordagem tentadora aos olhos dos investidores. Com o intuito de validar técnicas de detecção de falhas em máquinas são necessários dados para auxiliar no desenvolvimento e teste do sistema. Além disso, esses dados precisam ser tratados, validados e rotulados. A seguir será explicado o processo de aquisição dos dados utilizados neste trabalho.

Na Seção 2.1 será apresentada a bancada experimental e os demais equipamentos utilizados nesse processo. Nas Seções 2.2 e 2.3 serão discutidos os defeitos simulados e as medições de cada um deles, respectivamente.

2.1 Experimento Prático

A fim de se produzir a base de dados foi utilizada uma bancada experimental ABTV, comercializada pela *Spectra Quest Inc.* No decorrer desse capítulo serão apresentadas algumas figuras retiradas de [1] com a devida autorização do autor. Toda a obtenção dos dados foi feita em [1], onde detalhes mais técnicos da bancada podem ser encontrados. Entretanto, para maior compreensão dos dados, alguns aspectos serão discutidos.

O ABTV é uma plataforma desenvolvida para auxiliar no estudo de falhas em

máquinas, sendo extensamente utilizada no desenvolvimento de técnicas de manutenção preditiva e de aprendizado automático do comportamento das falhas. As falhas simuladas na bancada serão discutidas à frente. Na Figura 2.1 tem-se uma imagem do ABTV utilizado para a simulação dos dados.

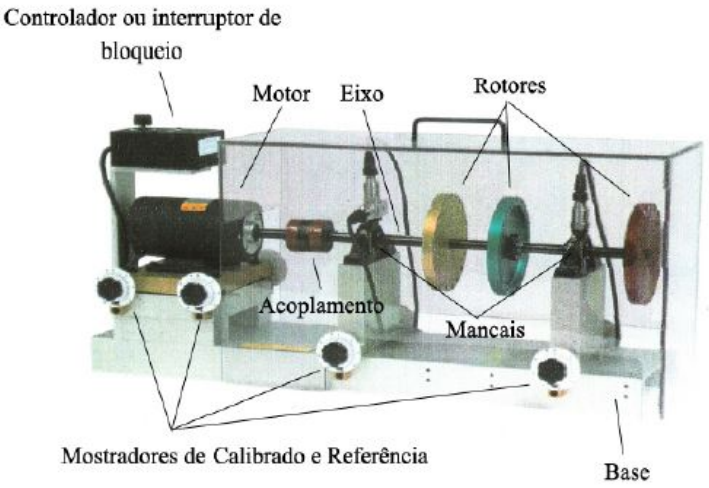


Figura 2.1: *Spectra Quest Inc.* ABTV [1].

Sobre o *Rotorkit* foram acoplados alguns equipamentos para realizar a aquisição dos dados. O sistema utilizado está representado na Figura 2.2.

Os dados foram adquiridos por meio de 6 componentes:

- 3 acelerômetros e 1 acelerômetro triaxial:

  Os acelerômetros são utilizados de modo a captar os sinais de vibração dos mancais. Para melhor entender a vibração do motor em casos de falhas é de grande valia utilizar acelerômetros nas direções radial, tangencial e axial. Em máquinas que estão sujeitas a erros no rolamento é aconselhável o uso de um conjunto de acelerômetros em cada mancal [4].

- 1 microfone:

  O microfone é utilizado para averiguar a relação entre o ruído da máquina e as falhas as quais ela vai ser submetida.

- 1 tacômetro:

  O tacômetro é utilizado para transformar um movimento rotatório em um sinal elétrico. Na aplicação em questão o tacômetro será utilizado para medir a frequência de rotação do motor. O processo de como essa medição é feita será explicado em detalhes na Seção 3.1.

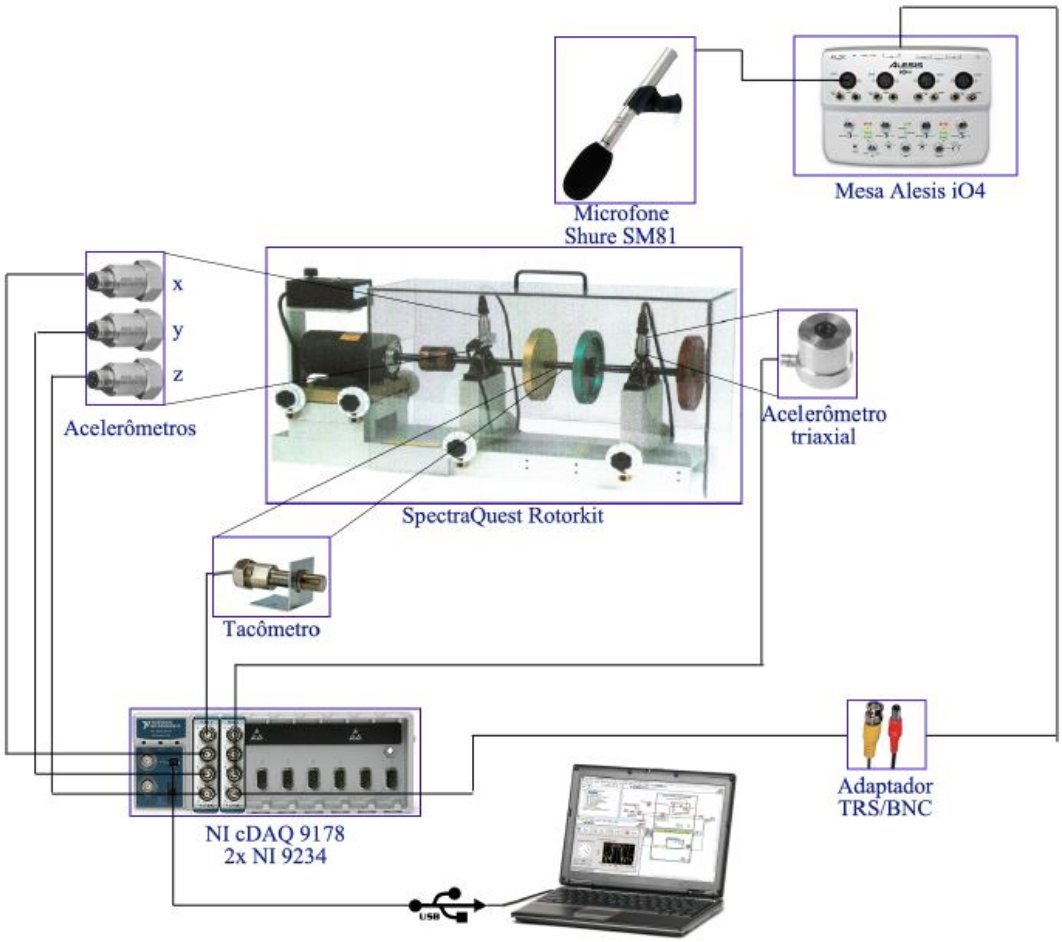


Figura 2.2: Diagrama do sistema experimental [1].

2.2 Defeitos

Com a necessidade de criar uma base de dados vasta abrangendo todos os possíveis erros, foram induzidas diversas condições adversas à bancada de testes. Alguns experimentos foram feitos com o sistema operando livre de falhas, servindo como referência. Foram simuladas falhas de desbalanceamento, desalinhamento dos eixos e mancais defeituosos. Esses defeitos serão descritos a seguir.

2.2.1 Desbalanceamento

O rotor é desenvolvido para que em seu estado normal ele possua simetria axial perfeita. Entretanto, com o eventual desgaste da máquina ou algum choque ocasional, essa simetria pode ser comprometida, causando desbalanceamento do rotor.

Com o objetivo de simular esse efeito, o ABTV oferece a possibilidade de se acoplar massas ao redor do rotor, causando o efeito de desbalanceamento. Na Figura 2.3 tem-se um exemplo da indução desse efeito por meio de uma massa, esta

destacada pelo círculo vermelho.

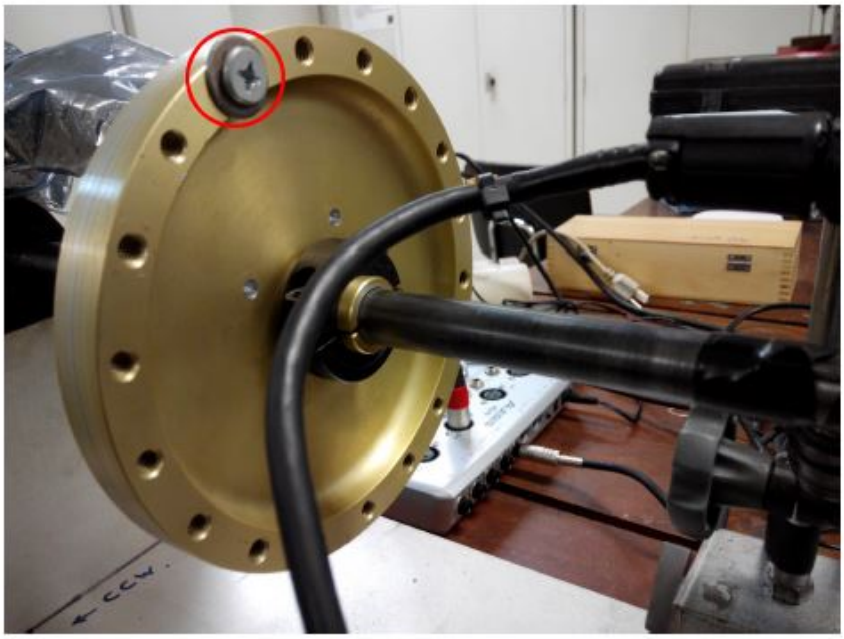


Figura 2.3: Exemplo da indução de erro de desbalanceamento [1].

2.2.2 Desalinhamento

Para o bom funcionamento do sistema é necessário que os eixos radiais da máquina motriz e da máquina motora estejam alinhados. Com essa condição satisfeita, o tempo médio entre falhas é maior, acarretando em um menor custo com a manutenção do sistema.

Em [1] foram provocados dois tipos de desalinhamento dos eixos do rotor: vertical e horizontal; ambos paralelos. Nas Figuras 2.4 e 2.5 tem-se duas ilustrações de como são esses desalinhamentos.

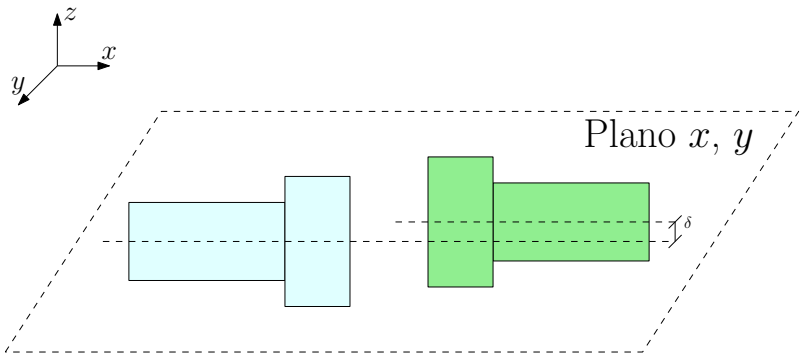


Figura 2.4: Desalinhamento Vertical.

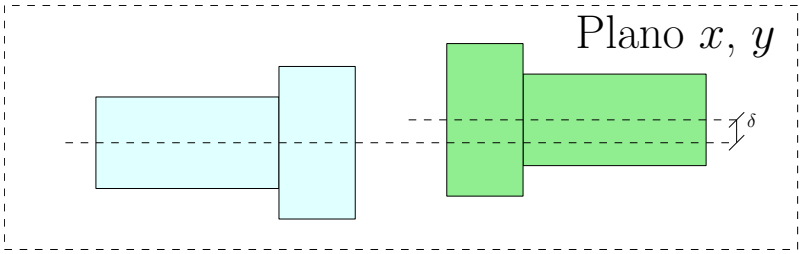


Figura 2.5: Desalinhamento Horizontal.

2.2.3 Mancais Defeituosos

Os últimos tipos de falhas induzidas foram sobre os mancais. Os mancais são peças responsáveis pela sustentação dos eixos das máquinas e, no caso, será estudado o efeito de falhas nos mancais de rolamento. Na Figura 2.6 tem-se o esquemático de um mancal de rolamento, assim como a vista de seu interior, onde pode-se notar certa complexidade da peça.

Figura 2.6: Ilustração de um Mancal de Rolamento.

Sendo uma das peças mais complexas da máquina, os mancais de rolamento também são as maiores causas de defeitos [2]. Cerca de 37% das falhas em máquinas elétricas são causadas por essas peças. Um comparativo das frequências de cada um dos tipos de falhas pode ser visto na Tabela 2.1.

Tabela 2.1: Relação entre falhas e suas incidências em máquinas elétricas [2].

Causas das Falhas	**Frequência** (%)
Rolamentos	37
Ventilação	33
Eixo/Acoplamento	6
Dispositivos Externos	5
Rotor	5
Buchas	3
Não Especificado	11

Foram utilizados diversos mancais para gerar diferentes defeitos. Esses defeitos são:

- Defeito na gaiola do mancal (Defeito A);
- Defeito no elemento rolante do mancal (Defeito B); e
- Defeito na parte externa do mancal (Defeito C)).

Por fim, tem-se ainda dois mancais no sistema encontrado na bancada experimental. O primeiro é o mancal interno, também chamado de não invertido, que encontra-se entre o rotor e o motor; o segundo é o externo, denominado durante o projeto como invertido, e encontra-se longe do motor. Separadamente, cada um deles será substituído por um mancal defeituoso com um dos tipos de falhas supracitadas.

2.3 Medições

Para se ter uma grande variedade na base de dados criada, diversas medições foram feitas e, para cada tipo de defeito, também foram aplicadas variações.

A frequência de amostragem das medições foi de 50 kHz, e cada sinal possui 250000 amostras, o que equivale a 5 segundos de gravação. A frequência de rotação da máquina também variou de 700 a 3600 rotações por minuto (RPM).

- **Funcionamento Normal**

  Para o sistema funcionando em condições normais foram feitas 49 medições. Nesses casos, apenas a frequência de rotação variou, de 737 a 3686 RPM.

- **Desalinhamento Horizontal**

  Foram feitas 197 medições para esse caso. As distâncias de desalinhamento foram: 0.5, 1.0, 1.5, 2.0 mm.

- **Desalinhamento Horizontal**

  Para este defeito foram feitas 301 medições, com os seguintes valores de desalinhamento: 0.51, 0.63, 1.27, 1.40, 1.78 e 1.90 mm.

- **Desbalanceamento**

  Nesse caso, o desbalanceamento foram utilizadas massas com os seguintes valores: 6, 10, 15, 20, 25, 30 e 35 g, gerando um total de 333 medições.

- **Defeitos nos Mancais**

  Os defeitos nos mancais são praticamente imperceptíveis caso não haja nenhum desbalanceamento. Por isso, além da configuração balanceada, foram aplicados 3 níveis de desbalanceamento: 6, 10 e 20 g, então, um total de 513 medições foram feitas com defeito no mancal invertido e mais 558 medições para o caso de defeito no mancal não invertido.

Na Tabela 2.2 têm-se todas as variações de defeitos e suas respectivas quantidades.

Após feito todo o processo de obtenção da base de dados, têm-se, para cada experimento, 8 sinais temporais (3 provenientes do acelerômetro triaxial e 1 de cada um dos outros 5 componentes de aquisição de dados). Cada um desses sinais possuem 250000 amostras, tornando inviável a criação de um sistema que trabalhe com tantos dados de dimensão tão elevada. Portanto, é necessário determinar características desses sinais que sejam capazes de os descrever de forma a otimizar o desempenho do sistema. O próximo capítulo trata da busca por essas características, descrevendo cada uma delas, assim como o processo de extração das mesmas.

Tabela 2.2: Medições para cada erro induzido sobre o *Rotorkit*.

Defeito	**Variação**		**Medições**	**Total de Medições**
Normal			49	49
Desbalanceamento	6 g		49	333
	10 g		48	
	15 g		48	
	20 g		49	
	25 g		47	
	30 g		47	
	35 g		45	
Desalinhamento Horizontal	0,5 mm		50	197
	1.0 mm		49	
	1.5 mm		49	
	2.0 mm		49	
Desalinhamento Vertical	0.51 mm		51	301
	0.63 mm		50	
	1.27 mm		50	
	1.40 mm		50	
	1.78 mm		50	
	1.90 mm		50	
Mancal Não Invertido	Defeito A	0 g	49	513
		6 g	49	
		20 g	49	
		35 g	41	
	Defeito B	0 g	49	
		6 g	49	
		20 g	49	
		35 g	41	
	Defeito C	0 g	49	
		6 g	43	
		20 g	25	
		35 g	20	
Mancal Invertido	Defeito A	0 g	49	558
		6 g	48	
		20 g	49	
		35 g	42	
	Defeito B	0 g	49	
		6 g	49	
		20 g	49	
		35 g	37	
	Defeito C	0 g	50	
		6 g	49	
		20 g	49	
		35 g	38	
Total de Experimentos				1951

Capítulo 3

Extração de Características e Pré-processamento

O Capítulo 2 apresentou como foi feita a obtenção dos dados que serão utilizados para avaliar o sistema classificador de falhas. No entanto, diante da quantidade de dados, é necessário reduzir o tamanho dos mesmos a um ponto em que seja viável processá-los. Essa redução é feita determinando as características, também chamadas de *features*, que sejam capazes de descrever cada uma das falhas a serem estudadas. Esse pré-processamento também é essencial no sentido de eliminar ruídos dos sinais, por isso as características escolhidas devem ser capazes de descrever o sinal suficientemente bem, desconsiderando informações danosas e redundantes.

Na Seção 3.1 será feita uma apresentação as características representativas, explicando o porquê de terem sido escolhidas e como é feito o processo de extração das mesmas. Na Seção 3.2 será apresentado o vetor de medidas usado pelo sistema e uma visão geral do bloco de pré-processamento.

3.1 Características Representativas

Conforme falhas vão ocorrendo, a primeira reação do sistema é a mudança na frequência de rotação do rotor. Com isso, leva-se a crer que esta seja de extrema relevância para a detecção de falhas, sendo bastante discriminativa. Ainda na análise espectral, têm-se as amplitudes dos sinais, que podem indicar a gravidade da falha, ajudando na sua detecção. Estudos foram feitos usando essas informações [1], [5] junto a diversos classificadores e, posteriormente, foi sugerido o uso de medidas estatísticas, como a curtose e a entropia [6].

3.1.1 Frequência de Rotação

A frequência de rotação é determinada através do sinal de vibração captado pelo tacômetro, $s(n)$. A partir dele é calculada a *Discrete Fourier Transform* (DFT), $S(k)$.

Entretanto, sinais defeituosos podem adicionar picos de energia em frequências que não correspondem à frequência fundamental da rotação encontrada em $s(n)$. Como a base de dados é composta basicamente por sinais defeituosos, é necessário ter cuidado ao determinar tal frequência. Em [5] foi proposto um método para resolver esse problema. Nele, primeiramente é calculada a DFT do sinal captado pelo tacômetro. Com $S(k)$ calculado, é determinado o índice que representa a frequência onde $S(k)$ possui maior magnitude, ou seja,

$$k_1 = \arg\max_k |S_t(k)|. \tag{3.1}$$

A partir de então é determinado o valor de f_1 de acordo com a Equação (3.2), e então todos os valores de $S_t(k)$ são zerados para $k \in [k_1 - 3, k_1 + 3]$:

$$f_1 = \frac{k_1 F_s}{N}, \tag{3.2}$$

onde F_s é a frequência de amostragem do sinal e N a quantidade de amostras do sinal $S(k)$.

Esse processo repete-se por mais 3 vezes, determinando assim os valores de f_2, f_3 e f_4. Por fim, a frequência de rotação R_f será dada por:

$$R_f = \min \{f_1, f_2, f_3, f_4\}. \tag{3.3}$$

3.1.2 Amplitudes Significativas

Ao analisar o espectro dos sinais provenientes do microfone e dos acelerômetros, é possível notar um comportamento interessante [7]. Em sinais obtidos a partir de experimentos com defeito de desbalanceamento, aparecem amplitudes relevantes nas frequências $2R_f$ e $3R_f$, apesar da magnitude na frequência R_f ser predominante. Já nos sinais provenientes de um experimento com falha de desalinhamento, nota-se que as magnitudes nas frequências do primeiro e segundo harmônicos são superiores a encontrada na frequência fundamental.

As magnitudes nessas frequências irão gerar as *features* em questão. Uma vez determinada a frequência de rotação através do sinal do tacômetro, serão determinadas as magnitudes na frequência fundamental, R_f, e nos dois primeiros harmônicos, $2R_f$ e $3R_f$, de cada um dos demais 7 sinais. Com isso, 21 novas características foram selecionadas.

3.1.3 Medidas Estatísticas

O domínio do tempo também pode contribuir com características discriminativas. Em [6] é calculada a entropia e a curtose para cada um dos 8 sinais. A abordagem atual também calcula a média de cada um desses sinais, gerando assim 24 novas características a serem consideradas.

- **Entropia**

  A entropia, H, é a medida de aleatoriedade e imprevisibilidade de um sinal [8]. Matematicamente, a entropia de um sinal discreto $s(n)$, com Probability Density Function (PDF) $P(s)$ pode ser escrita da seguinte forma:

  $$H = -\sum_{i} P(s_i) \log P(s_i)., \tag{3.4}$$

  Neste trabalho, $P(s)$ é calculada através do histograma do sinal, utilizando largura fixa para cada bin igual a 0.2.

- **Média**

  A média, M, é o momento de primeira ordem do sinal e pode ser determinada, para um sinal com N amostras, por

  $$M = \frac{\sum_{i=1}^{N} s(i)}{N}. \tag{3.5}$$

- **Curtose**

  A curtose, K, é o medida estatística de quarta ordem de um sinal e, matematicamente, pode ser escrita como:

  $$K = \frac{E\left[(X - M_X)^4\right]}{E\left[(X - M_X)^2\right]}, \tag{3.6}$$

  onde M_X é a média da variável aleatória X.

3.2 Vetor de Características e Bloco de Pré-processamento

3.2.1 Vetor de Características

A pretendida redução de dimensionalidade foi alcançada. No início existiam 8 sinais, cada um com 250000 amostras, e agora, após a escolha das *features* apropriadas,

tem-se apenas um único vetor com 46 dimensões. Na Tabela 3.1 consta uma breve descrição das características selecionadas.

Tabela 3.1: Características dos sinais e seus respectivos símbolos.

Característica	**Símbolo**
Frequência de Rotação	R_f
Magnitude da DFT do acelerômetro i no eixo x na frequência f	$S_x^{(i)}(f)$
Magnitude da DFT do acelerômetro i no eixo y na frequência f	$S_y^{(i)}(f)$
Magnitude da DFT do acelerômetro i no eixo z na frequência f	$S_z^{(i)}(f)$
Magnitude da DFT do microfone na frequência f	$S_M(f)$
Entropia, Curtose e Média do acelerômetro i no eixo x	$H_x^{(i)}, K_x^{(i)}$ e $M_x^{(i)}$
Entropia, Curtose e Média do acelerômetro i no eixo y	$H_y^{(i)}, K_y^{(i)}$ e $M_y^{(i)}$
Entropia, Curtose e Média do acelerômetro i no eixo z	$H_z^{(i)}, K_z^{(i)}$ e $M_z^{(i)}$
Entropia, Curtose e Média do microfone	H_M, K_M e M_M

Então, o vetor $\mathbf{V}_i$ a ser classificado irá possuir a estrutura representada por

$$\mathbf{V}_i = [\overbrace{R_f \;\; S_x^{(1)}(R_f) \;\; S_y^{(1)}(3R_f) \; S_x^{(2)}(R_f) \;\; S_x^{(2)}(3R_f) \;\cdots\; S_A^{(2)}(R_f)}^{\text{22 medidas frequenciais}} \underbrace{H_x^{(1)} \;\; H_y^{(1)} \;\; H_z^{(1)} \;\cdots\; M_A^{(2)}}_{\text{24 medidas estatsticas}}]. \tag{3.7}$$

O intuito de se utilizar uma bancada experimental para simular as falhas é de se criar um sistema que seja capaz de ser implantado em máquinas rotativas. Portanto, a fim de diminuir a dependência com a configuração utilizada nesse projeto, os sinais foram normalizados de modo a possuírem energia unitária. Dado um sinal $s(n)$, com N amostras, o mesmo sobre a transformação de acordo com

$$s(n) = \frac{s(n)}{\sqrt{\sum_{i=1}^{N} s(i)^2}}. \tag{3.8}$$

A matriz de dados agora tem dimensões 1951×46. Os experimentos serão divididos em dois conjuntos: um de treino e um para o teste. O conjunto de treino será usado para criar o modelo com as especificações desejadas e o conjunto de teste será usado para avaliar o modelo criado.

3.2.2 Pré-processamento

Ao longo do capítulo foram discutidas quais serão as *features* utilizadas no escopo desse projeto, assim como a forma de calculá-las. Na Figura 3.1 tem-se a representação do funcionamento desse bloco. O processo é, basicamente, receber os sinais dos dispositivos de aquisição de dados para cada experimento e, então, gerar as

características.

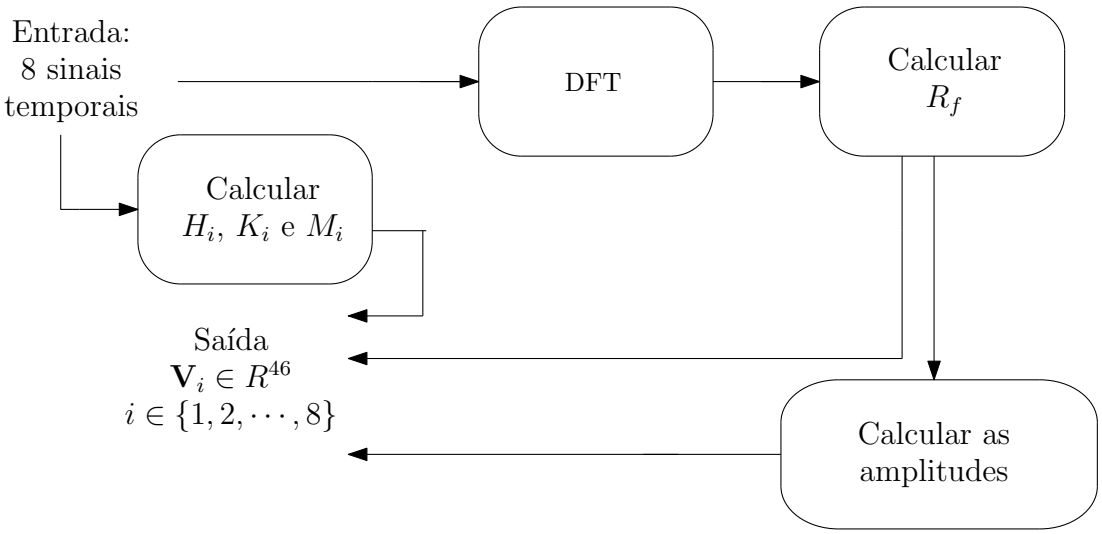


Figura 3.1: Esquemático do bloco de pré-processamento.

No Capítulo 5 serão explicitados os ganhos na eficiência do modelo ao variar o conjunto de características utilizadas como entrada do modelo. No próximo capítulo serão apresentados os diferentes sistemas utilizados para criar os modelos, assim como toda a base teórica para ajudar na compreensão dos mesmos.

Capítulo 4

Sistema

O sistema proposto como solução do problema será composto por três partes. A primeira será um bloco de pré-processamento, que tem como objetivo a extração das *features* a serem usadas; a segunda é o SBM, um método não paramétrico baseado em similaridade; e, por fim, temos o classificador, que será um *Random Forest*.

Na Seção 3.2 o bloco de pré-processamento foi apresentado. Nas Seções 4.1 e 4.2 serão discutidos os blocos referentes ao SBM e ao *Random Forest*, respectivamente. Serão explicados os mais importantes aspectos de cada um, e exemplos mais simples serão apresentados para ajudar no entendimento.

4.1 *Similarity-Based Modeling*

O SBM é um método não-paramétrico para modelagem de classes que faz uso de técnicas de similaridade [9]. A principal vantagem de se fazer uso de um método não paramétrico é não ter necessidade de um conhecimento *a priori* do sistema, uma vez que ele irá trabalhar através do reconhecimento de padrões, baseando-se na métrica usada pelo algoritmo implementado [10].

Esse método foi proposto em [11] com o objetivo de monitorar e detectar falhas em um dado sistema e visando a aplicação na indústria. Tais aplicações incluem a detecção de anomalias em estações de energia [12] e também a modelagem de rotas de vôos [13].

4.1.1 Base Teórica

No SBM tem-se, para cada possível classe, um conjunto de estados que são suficientes para descreverem o mesmo. Tendo feita a seleção na etapa de treinamento, dado um novo estado, a saída é estimada como uma ponderação dos estados que descrevem

aquela classe. Com isso, cada classe será representado por uma matriz $\mathbf{D}_K$, que contém esses estados, e será escrita como:

$$\mathbf{D}_K = \begin{bmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,m_K} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,m_K} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n,1} & x_{n,2} & \cdots & x_{n,m_K} \end{bmatrix}, \tag{4.1}$$

onde $\mathbf{x}_{i,j}$ é a i-ésima componente da amostra $\mathbf{x}_j$. Desse modo, cada amostra possui n componentes e a matriz referente à classe K, $\mathbf{D}_K$, possui m_K estados que são suficientes para descrever a classe.

Na etapa de treinamento do modelo, são determinadas as amostras representativas para a formação da matriz $\mathbf{D}_K$ correspondente à classe K. Uma vez que o modelo esteja treinado e validado, ao ser feita mais uma observação, ele tentará estimar qual seria o vetor ($\mathbf{x}_{\text{est}|K}$), obtido através de uma combinação linear das colunas de $\mathbf{D}_K$, que teria o resíduo com menor norma ℓ_2, ou seja,

$$\mathbf{x}_{\text{est}|K} = \min_{\mathbf{x}} \|\mathbf{x} - \mathbf{x}_{\text{obs}}\|_2. \tag{4.2}$$

Escrevendo $\mathbf{x}_{\text{est}|K}$ como uma combinação de estados representativos da classe K tem-se

$$\mathbf{x}_{\text{est}|K} = \mathbf{D}_K \mathbf{w}. \tag{4.3}$$

A partir da Equação (4.2) define-se $f(\mathbf{x})$ como

$$f(\mathbf{x}) = \|\mathbf{D}\mathbf{w} - \mathbf{x}_{\text{obs}}\|_2^2. \tag{4.4}$$

Calculando gradiente de $f(x)$ em relação à $\mathbf{w}$ tem-se

$$\nabla_{\mathbf{w}} f(\mathbf{x}) = 2(\mathbf{D}^T\mathbf{D})\mathbf{w} - 2\mathbf{D}^T\mathbf{x}_{\text{obs}} \tag{4.5}$$

Visto que $f(\mathbf{x})$ é uma função convexa, ao igualar $\nabla_{\mathbf{w}} f(\mathbf{x})$ à 0 irá ser possível encontrar o valor de $\mathbf{w}$ que a minimiza. Portanto

$$\nabla_{\mathbf{w}} f(\mathbf{x}) = 0 \Rightarrow \mathbf{w} = (\mathbf{D}^T\mathbf{D})^{-1}\mathbf{D}^T\mathbf{x}_{\text{obs}} \tag{4.6}$$

A partir das Equações 4.3 e 4.6 é possível concluir que

$$\mathbf{x}_{\text{est}} = \mathbf{D}\mathbf{w} \Rightarrow \mathbf{x}_{\text{est}} = \mathbf{D}(\mathbf{D}^T\mathbf{D})^{-1}\mathbf{D}^T\mathbf{x}_{\text{obs}} \tag{4.7}$$

Essa abordagem é refém de diversos aspectos negativos, dentre eles a possibilidade da matriz $\mathbf{D}^T\mathbf{D}$ ser mal condicionada, fazendo com que a sua inversa demande

grande custo computacional para ser calculada, ou até mesmo que seja impossível. Outro aspecto a ser considerado é o tamanho da matriz $\mathbf{D}$; caso ela seja muito grande, demandará muito esforço computacional [11].

Com isso, na técnica SBM foi proposta uma nova abordagem, que pode ser encarada como uma generalização do método linear. Agora, ao invés de considerar o produto interno entre dois vetores, é considerada a similaridade entre os mesmos. Essa operação deve atender algumas restrições [14] :

1. A similaridade deverá ser um escalar, entre 0 e 1, inclusive;
2. A similaridade entre dois vetores idênticos deverá ser 1, e 0 no caso de serem totalmente diferentes; e
3. A similaridade deve aumentar conforme os vetores se aproximam.

Define-se $\mathbf{x} \otimes \mathbf{y}$ como sendo a operação de similaridade entre os vetores $\mathbf{x}$ e $\mathbf{y}$. Podemos então definir $\mathbf{C} = \mathbf{A}^T \otimes \mathbf{B}$ de modo que $c_{i,j} = \mathbf{a}_i \otimes \mathbf{b}_j$, onde $\mathbf{a}_i$ e $\mathbf{b}_j$ são a i-ésima e j-ésima colunas de $\mathbf{A}$ e $\mathbf{B}$, respectivamente. Mas, como visto, essa operação não é única, deve apenas obedecer as restrições já citadas. As opções a serem consideradas nesse trabalho são:

1. *Radial Basis Function* (RBF):

$$\mathbf{x} \otimes \mathbf{y} = e^{-\gamma \|\mathbf{x}-\mathbf{y}\|_p^2}; \tag{4.8}$$

2. *Inverse Euclidean Similarity* (IES):

$$\mathbf{x} \otimes \mathbf{y} = \frac{1}{1 + \gamma \|\mathbf{x} - \mathbf{y}\|_p}. \tag{4.9}$$

3. *Inverse Multiquadratic Kernel* (IMK):

$$\mathbf{x} \otimes \mathbf{y} = \frac{1}{\sqrt{1 + (\gamma \|\mathbf{x} - \mathbf{y}\|_p)^2}}. \tag{4.10}$$

4. *Exponential Kernel* (EXP):

$$\mathbf{x} \otimes \mathbf{y} = e^{-\gamma \|\mathbf{x}-\mathbf{y}\|_p}; \tag{4.11}$$

5. *Cauchy Kernel* (CCK):

$$\mathbf{x} \otimes \mathbf{y} = \frac{1}{1 + (\gamma \|\mathbf{x} - \mathbf{y}\|_p)^2}. \tag{4.12}$$

Foi utilizada a norma p para calcular a distância entre os dois vetores $\mathbf{x}$ e $\mathbf{y}$. Esta é definida de acordo com

$$\|\mathbf{x}-\mathbf{y}\|_p = \left(\sum_{i=1}^{q} |x_i|^p\right)^{1/p}, \tag{4.13}$$

onde q é o tamanho dos vetores $\mathbf{x}$ e $\mathbf{y}$.

A função IES é uma generalização da proposta em [14]. Já a função RBF foi utilizado por ser amplamente utilizada em problemas que tratam de classificação. Já IMK, EXP[15] e CCK[16] também satisfazem os critérios para serem utilizadas no SBM.

Portanto, utilizando essa nova abordagem, temos que a solução presente na Equação (4.7) se tornará

$$\mathbf{x}_{\text{est}} = \mathbf{D}\bar{\mathbf{w}}. \tag{4.14}$$

A Equação (4.14) mostra que o valor estimado será uma ponderação entre as amostras presentes em $\mathbf{D}$. Para isso, $\bar{\mathbf{w}}$ é a normalização (pela soma de seus elementos) do vetor $\mathbf{w}$, ou seja,

$$\bar{\mathbf{w}} = \frac{\mathbf{w}}{\sum_{i=1}^{n} w_i}, \tag{4.15}$$

onde

$$\mathbf{w} = \left(\mathbf{D}^T \otimes \mathbf{D}\right)^{-1} \mathbf{D}^T \otimes \mathbf{x}_{\text{obs}}. \tag{4.16}$$

Considerando $\mathbf{G} = \mathbf{D}^T \otimes \mathbf{D}$ e $\mathbf{a} = \mathbf{D}^T \otimes \mathbf{x}_{\text{obs}}$, temos:

$$\mathbf{w} = \mathbf{G}^{-1}\mathbf{a}. \tag{4.17}$$

Da forma como foi construído, o vetor $\mathbf{a}$ representa um vetor com as similaridades entre o vetor de entrada, $\mathbf{x}_{\text{obs}}$, e as amostras presentes na matriz $\mathbf{D}$. A partir daí, o vetor $\mathbf{a}$ é transformado, através de $\mathbf{G}^{-1}$, em um vetor de pesos, de acordo com as similaridades das amostras presentes em $\mathbf{D}$ [10].

Idealmente, as amostras pertencentes à $\mathbf{D}_K$ serão dissimilares, ou seja, teremos

$$\mathbf{x}_i \otimes \mathbf{x}_j = \begin{cases} 1, & \text{se } i = j \\ 0, & \text{caso contrário} \end{cases}, \forall i, j \in \{1, 2, \cdots, m_K\} \,|\mathbf{x}_i, \mathbf{x}_j \in \mathbf{D}_K, \tag{4.18}$$

ou seja, $\mathbf{G} = \mathbf{I}_{m_K}$. Mas, na prática, isso dificilmente vai acontecer. Um caso particular do SBM é feito forçando a matriz $\mathbf{G}$ a ser a matriz identidade. Essa particularização é conhecida por *Auto-Associative Kernel Regression* (AAKR) [17].

Na Figura 4.1 tem-se representado o modelo de avaliação para uma entrada em relação à uma dada classe c. Com o modelo já treinado, a matriz $\mathbf{D}_c$ já está definida e, consequentemente, $\mathbf{w}$ também está. Repare que, para a saída do SBM pode-se usar $\mathbf{r}_{\text{est|c}}$, que se trata do resíduo da estimativa da entrada para a classe c, ou $\varepsilon = \|\mathbf{r}_{\text{est|c}}\|_2$, que é a norma ℓ_2 do resíduo.

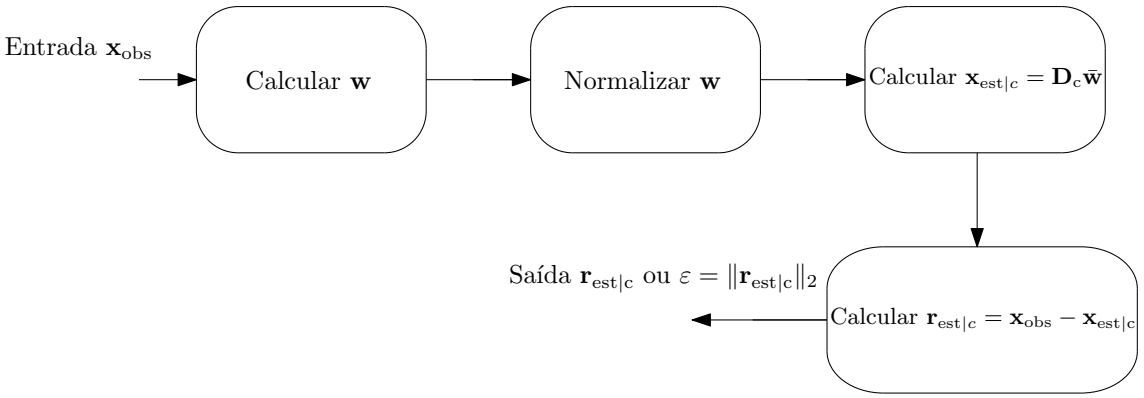


Figura 4.1: Modelo do SBM já treinado para dada classe c.

4.1.2 Treinamento

A grande tarefa ao se treinar um SBM é a escolha da matriz $\mathbf{D}$. Na Figura 4.2 uma ideia geral de como funciona o treinamento de um modelo é apresentada. O conjunto de treinamento conta com N pares $\{\mathbf{x}_1, d_1\}, \{\mathbf{x}_2, d_2\}, \cdots, \{\mathbf{x}_N, d_N\}$, visto que o treinamento é supervisionado. Estes pares estão divididos em classe, e para cada uma delas é necessário determinar seus estados representativos.

Com isso, para cada classe, começa a seleção para as amostras que irão compor a matriz $\mathbf{D}$. Define-se $\mathcal{D}_K$ como o conjunto com todas as amostras, e somente elas, pertencentes à classe K. A seguir, serão apresentadas algumas formas de se fazer essa seleção:

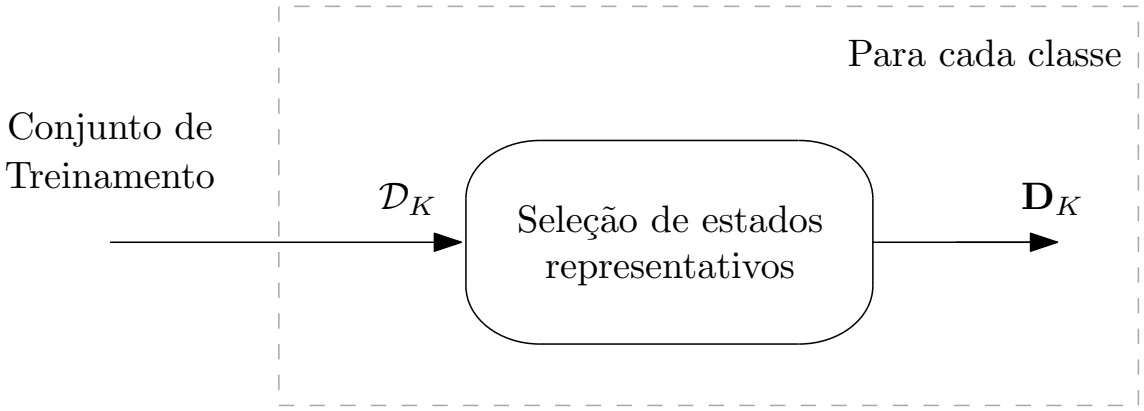


Figura 4.2: Modelo de Treinamento do SBM.

1. Todos fazem parte

Essa forma é a mais ingênua possível e menos complicada. Nela, não há escolha, todas as amostras que pertencerem a uma determinada classe irão, automaticamente, fazer parte da matriz $\mathbf{D}$ do respectivo estado.

2. Limiar de similaridade

Nessa abordagem escolhe-se a mediana geométrica do conjunto $\mathcal{D}_K$ e a adiciona à $\mathbf{D}_K$. A mediana geométrica de um conjunto $\mathcal{M} \in \mathbb{R}^n$ é o vetor $\mathbf{x}$ de modo que seja solução da Equação (4.19)

$$\mathbf{v}_{\mathcal{M}} = \arg\min_{\mathbf{x}} \sum_i \|\mathbf{x} - \mathbf{u}_i\|_2, \ \forall \ \mathbf{u}_i \in \mathcal{M}. \tag{4.19}$$

Ou seja, encontra-se o elemento de $\mathcal{M}$ que minimiza a soma das distâncias euclidianas a todos os demais pontos do conjunto. A mediana geométrica é uma abordagem melhor do que se escolher o centroide do conjunto, uma vez que esse último dificilmente será um elemento do conjunto, e isso é necessário, visto que apenas amostras podem fazer parte da matriz $\mathbf{D}$.

Feito isso, para cada outra amostra pertencente à $\mathcal{D}_K$, calcula-se a similaridade em relação a todos amostras já presentes em $\mathbf{D}_K$. Caso todos os valores sejam menores que um determinado valor, τ, essa amostra fará parte de $\mathbf{D}_K$. Ou seja, a condição para uma nova amostra, $\mathbf{x}_{\text{obs}}$ ser considerada significativa pode ser expressa por:

$$\mathbf{x}_{\text{obs}} \otimes \mathbf{D}_K \preceq \tau \tag{4.20}$$

O algoritmo a seguir traz o pseudocódigo dessa nova abordagem:

Algoritmo 4.1 Algoritmo para seleção de amostras para a matriz $\mathbf{D}_K$

função SELECIONAR_AMOSTRAS($\mathbf{X}_K$, τ),
  Seleciona-se, dentre as amostras presentes em $\mathcal{D}_K$, a mediana geométrica
  Adiciona-se as amostras selecionadas à $\mathbf{D}_K$
  para $\mathbf{x}_i \in \mathcal{D}_K$ **faça**
    se $\mathbf{x}_i \otimes \mathbf{x}_j \leq \tau \ \forall j \mid \mathbf{x}_j \in \mathcal{D}_K$ **então**
      Adicionar o vetor $\mathbf{x}_i$ à matriz $\mathbf{D}_K$

3. Método original

Esse método é não iterativo, o que o faz uma melhor aposta no que se trata de custo computacional. Nele, para cada classe, iremos escolher algumas amostras do conjunto de treinamento em duas etapas [18]:

1. Escolhem-se, para cada característica estudada, as amostras que possuírem os valores extremos; e

2. Ordenam-se as amostras de acordo com a norma ℓ_2, em ordem decrescente. Feito isso, seleciona-se amostras começando da primeira e escolhendo as demais com um determinada de amostragem.

4

4.1.3 Exemplo Ilustrativo

Para melhor ilustrar o SBM será apresentado um exemplo 2D. Foram criadas duas distribuições gaussianas de variância unitária, uma delas com média 2 e outra com média -2, ambas pertencentes à mesma classe. A partir daí o sistema foi treinado a fim de determinar os estados representativos da classe. Com o modelo treinado, foram geradas mais amostras seguindo o mesmo procedimento anterior, incluindo algumas amostras *outliers*, para executar os testes e verificar o comportamento do modelo. Na Figura 4.3, as amostras pertencentes à classe estão representadas na cor azul e os *outliers* na cor vermelha. No eixo z tem-se a norma do resíduo entre a amostra e sua estimativa, feita pela matriz D da classe. Repare que as amostras em azul apresentam erro bem próximo de 0, enquanto a maior parte dos *outliers* possuem erro maior que 0, acarretando uma clara separação entre as amostras. Esse exemplo foi feito utilizando o SBM (não o AAKR), com o método da similaridade máxima para treinamento, e utilizando como métrica de similaridade a métrica IES.

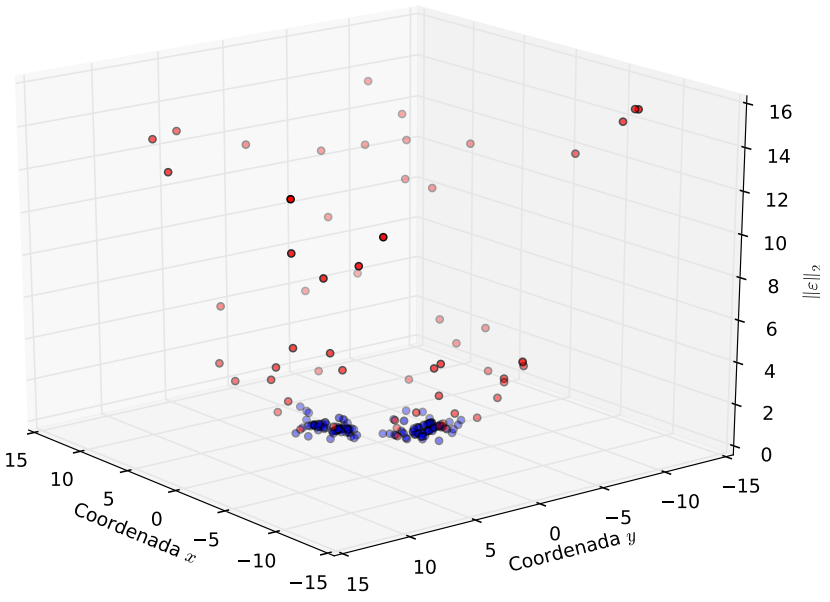


Figura 4.3: Exemplo 2D do SBM.

4.2 Classificador *Random Forest*

O último bloco do sistema proposto é um classificador *Random Forest*. Este, por sua vez, irá tomar a decisão levando em consideração a amostra preprocessada e as saídas do bloco SBM. O *Random Forest* é baseado em árvores de decisões e em uma técnica chamada *bagging* [19] que se assemelha a escolha por votação. Esta, por sua vez, é usualmente aplicada a sistemas que possuem alta variância, o que a faz uma excelente escolha quando se trata de árvores de decisão. A Figura 4.4 traz um exemplo para ilustrar como se constrói uma dessas estruturas.

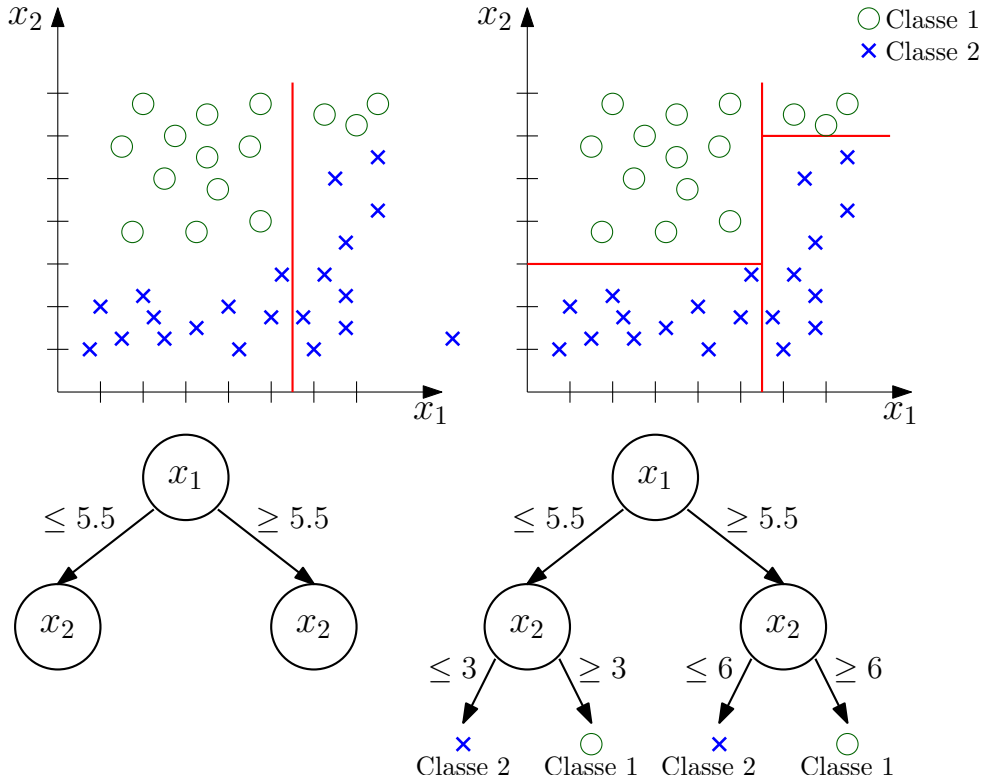


Figura 4.4: Exemplo de árvore de decisão com dois parâmetros.

Nesse exemplo, têm-se dados pertencentes a duas classes que estão distribuídos no plano $\mathbb{R}^2$. Deseja-se então separá-los de acordo com as respectivas classes. O processo começa buscando no eixo x_1 o ponto através do qual seja possível criar uma primeira separação entre as duas classes. Escolhe-se então o ponto $x_1 = 5.5$, criando-se um nó a partir do mesmo. Continuando o processo, abrem-se duas buscas independentes e que ocorrem em paralelo. A primeira busca na região do plano onde $x_1 \geq 5.5$ e a segunda nos pontos com $x_1 \leq 5.5$. Repare que para cada um dos grupos, dois novos limiar são estabelecidos, para o primeiro grupo tem-se $x_2 = 3$ e para o segundo, $x_2 = 6$.

No exemplo anterior, foi relativamente fácil determinar os pontos para se traçar os cortes, ou seja, os nós das árvores. Em problemas onde os dados possuem dimensões muito grandes, essa tarefa fica inviável de ser feita visualmente. Existem

diversos critérios que podem ser adotados para escolher os pontos de separação, mas para problemas de classificação as mais comuns são [20]:

- Através do ganho de informação [8]:
- Através do índice Gini [21].

Ao utilizar o *bagging* o conjunto de treino é dividido em diversos subconjuntos, podendo ter sobreposições. Feito isso, é treinado um modelo para cada um desses subconjuntos. Então, ao se estimar uma entrada, cada árvore irá, de forma independente das demais, classificar o subconjunto, ou seja, irá votar. Apurando-se todos os votos, a classe com o maior número de votos será a classificação para dada entrada. Uma importante característica é a possibilidade de paralelizar o processo, visto que os processos de apuração de voto de cada árvore são independentes entre si, o que é relevante no caso de modelos muito grandes.

O *Random Forest* por sua vez adiciona mais uma etapa ao *bagging* [22]. Novamente, o conjunto de treino será dividido em diversos subconjuntos, e a partir deles diferentes árvores serão criadas. No entanto, para cada subconjunto, algumas *features* do conjunto de dados serão selecionadas, e partir delas será criada a árvore de decisão. No Algoritmo 4.2 é apresentado o pseudocódigo do treinamento do *Random Forest*.

Algoritmo 4.2 Algoritmo *Random Forest*.

B = quantidade de árvores
para $b = 1$ até B **faça**
    Seleciona-se, de forma aleatória, um subconjunto do conjunto de treino com tamanho N.
    Seleciona-se, de forma aleatória, p *features* do total.
    Cria-se uma árvore de decisão T_b

Com o modelo treinado, para classificar uma nova amostra $\mathbf{x}$ deve-se analisar os votos de cada árvore presente no modelo criado no treinamento. Matematicamente, o *Random Forest* é regido de acordo com as Equação (4.21). Define-se $\hat{C}_b(\mathbf{x})$ como a predição da b-ésima árvore, ou seja, o seu voto; e $\mathcal{C}_1^B = \left\{\hat{C}_1(\mathbf{x}), \hat{C}_2(\mathbf{x}), \cdots, \hat{C}_B(\mathbf{x})\right\}$. Portanto, tem-se que

$$\hat{C}(\mathbf{x}) = \arg\max_c \sum_{i=1}^{B} \delta_{c,\hat{C}_i(\mathbf{x})} \qquad (4.21)$$
$$\text{sujeito à } c \in \{1, 2, \cdots m\},$$

onde $\delta_{i,j}$ é o delta de Kronecker, dado por:

$$\delta_{i,j} = \begin{cases} 1, & \text{se } i = j, \\ 0, & \text{caso contrário.} \end{cases} \tag{4.22}$$

Como já ressaltado, árvores de decisão possuem alta variância. Com isso, tenta-se sempre diminuir a variância do modelo através do ajuste de parâmetros. Uma maneira de melhorar o resultado nesse quesito é através do ajuste do número de *features* selecionadas para cada árvores (parâmetro p no Algoritmo 4.2). Ao aumentar esse parâmetro, menos sujeito à variância do conjunto de dados o modelo estará. No entanto, o aumento desse valor acarreta no aumento do *bias* do sistema. Para problemas de classificação, usualmente usa-se $p = \sqrt{n_{features}}$; já para problemas de regressão, usa-se $p = n_{features}$. Entretanto, essa não é a melhor maneira de diminuir a variância do modelo, visto que o aumento de *features* também acarreta no aumento do *bias*.

Quanto à quantidade de árvores utilizadas no *Random Forest*, esta pode variar de acordo com quanto processamento computacional pode ser desprendido. Ao aumentar o número de árvores, mais preciso e menos variante será o modelo. Entretanto, esse ganho nos resultados satura, chega a certo ponto que não adianta aumentar o número de árvores do modelo, ele não se tornará mais eficiente. A partir desse ponto, o contínuo aumento causa um efeito parecido com o *overfitting*, diminuindo a capacidade de generalização do modelo.

O ajuste de todos esses parâmetros deve levar em consideração a possibilidade de ocorrer *overfitting*. Este é quando o modelo é treinado muito especificamente para os dados do conjunto de treino, e com isso, apresenta resultados piores no conjunto de testes, a Figura 4.5 apresenta uma ilustração desse fenômeno.

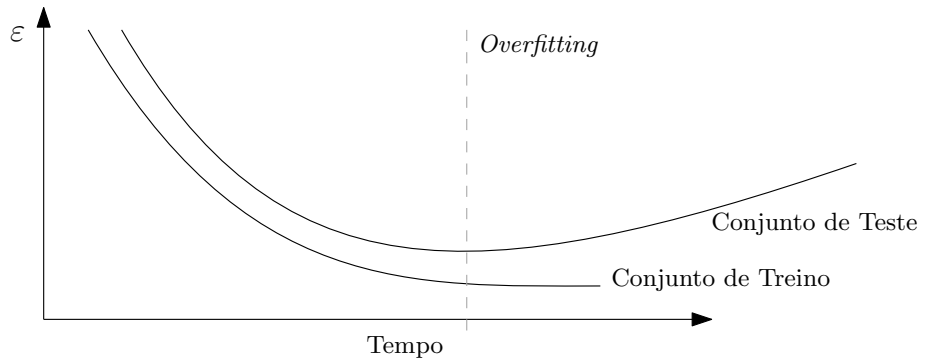


Figura 4.5: Exemplo de *overfitting*.

Em árvores de decisão, normalmente combate-se o *overfitting* limitando o número de camadas da árvore. Esse procedimento é feito de forma imediata, limitando a profundidade a um valor pré-determinado ou então através da poda da árvores.

Essa última é feita crescendo a árvore de forma ilimitada, e então juntar nós filhos ao respectivo pai caso o erro acumulado diminua. Entretanto, normalmente tais técnicas não são adotadas no *Random Forest*.

O *Random Forest* explora a característica das árvores de decisão de possuírem alta variância, uma vez que a utilização do *bagging* é mais eficaz. Procura-se então usar um modelo que apresente baixo *bias* e alta variância, e através do *bagging* e da seleção de um grupo de *features* para cada árvore, diminui-se a variância do modelo. Dessa forma não somente se reduz a variância do modelo, mas também diminui a chance de ocorrer *overfit*.

Todos os modelos a serem testados foram elucidados. No capítulo a seguir serão apresentados os perfis de simulação utilizados, assim como as características de cada um deles. A seguir os resultados serão apresentados e discutidos.

Capítulo 5

Simulações e Análise de Resultados

Como visto na Seção 4.1, existem diferentes formas de se treinar o SBM, e consequentemente, todo o sistema. Neste capítulo, os desempenhos dessas diferentes abordagens serão analisados, de forma a encontrar o classificador com melhores resultados.

Na Seção 5.1 serão apresentados os diversos perfis de simulação testados. Após isso, a Seção 5.2 traz as métricas usadas para analisar os resultados obtidos pelos perfis de simulação propostos. A Seção 5.3 traz o roteiro seguido na etapa de treinamento, cujos resultados são apresentados na Seção 5.4. Os resultados obtidos na etapa de teste serão apresentadas na Seção 5.5. Por fim, na Seção 5.6 será feita uma comparação dos resultados obtidos com os encontrados em outros trabalhos que utilizaram a mesma base de dados.

5.1 Simulações

O sistema apresentado contava com três grandes módulos: o responsável pelo pré-processamento, o SBM e o *Random Forest*. No entanto, existe ainda a possibilidade de tratar o SBM como um classificador por si só. O modelo ilustrado na Figura 5.1 explora essas possibilidades.

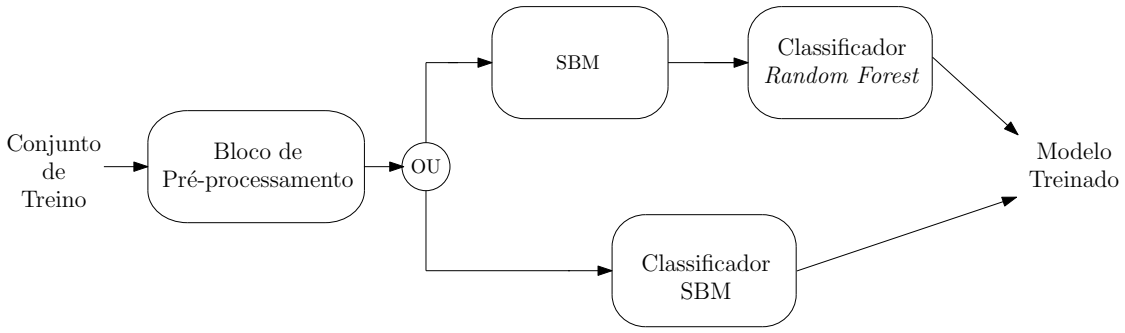


Figura 5.1: Diagrama de blocos do sistema, considerando as duas possibilidades para o classificador.

5.1.1 SBM Como Classificador

As saídas do SBM podem ser os resíduos em relação a cada uma das classes envolvidas no problema ou podem ser a similaridade da amostra para cada uma dessas classes. Para tratar o SBM como classificador foi utilizada a medida de similaridade para tomar a decisão.

Nesta topologia, para classificar uma nova amostra, serão geradas as similaridades com cada uma das classes determinadas na etapa de treino. A classe escolhida será a que possuir maior similaridade. Esse processo está descrito a seguir:

$$K_i = \arg\max_{c} \{\mathbf{x}_{\text{obs}|i} \otimes \mathbf{x}_{\text{est}|c}\} \quad \text{sujeito à} \quad c \in \{1, 2, \cdots, m\}, \tag{5.1}$$

onde m é a quantidade de classes, K_i é a classificação encontrada para a i-ésima amostra observada, $\mathbf{x}_{\text{obs}|i}$.

Ainda assim, existem diversas possibilidades para construir o classificador. A operação de similaridade foi feita através de uma das cinco funções de similaridades apresentadas na Seção 4.1.1, e para cada uma delas o parâmetro γ, apresentado nas Equações 4.8 a 4.12, foi variado para estudar as diversas variações de cada operação. Foram ainda testados dois valores para norma p, Equação 4.13, no cálculo das distâncias.

Outra variação testada foi o uso de um tipo específico do SBM, o AAKR. Essa versão considera que os estados representativos de cada classe são descorrelacionados, implicando que a matriz de pesos $\mathbf{G}$ do SBM seja a identidade.

Por fim, foram testadas as três diferentes maneiras de fazer a seleção dos estados representativos para a formação das matrizes $\mathbf{D}$. No método original foram consideradas diferentes taxas de amostragem no passo em que estados são escolhidos de acordo com as suas respectivas notas. Já no método da similaridade máxima, diferentes valores para o parâmetro τ, encontrado na Equação 4.20, foram testados, a fim de determinar qual o limiar melhor representa dois estados similares.

Todos os códigos foram implementados em Python, fazendo uso da biblioteca sklearn [23].

5.1.2 SBM Com o *Random Forest* Como Classificador

O SBM também pode ser usado como um estimador de estados e não como classificador. Assim, as saídas do SBM irão alimentar a entrada do classificador, que no contexto é o *Random Forest*, conforme descrito na Seção 4.2. Essa passagem de informação é feita através de dois métodos de transformação.

O primeiro é utilizando os resíduos entre a entrada observada e a estimada para

cada uma das classes. Caso todas as *features* sejam utilizadas, o resíduo de cada classe contará com 46 dimensões. Conforme descrito na Seção 2.2, a base de dados conta com cinco diferentes falhas, ou seja, serão seis classes, contando com a máquina operando livre de falhas, gerando um total de $6 \times 46 = 276$ parâmetros passados para o classificador, sem contar com as características originais. Utilizando a passagem apenas das similaridades para o classificador, têm-se apenas 6 parâmetros, referentes a norma ℓ_2 do resíduo. A Tabela 5.1 mostra a dimensão dos dados de entrada para o classificador nos dois casos, contando que o sinal de entrado do SBM também é entrada do *Random Forest*.

Tabela 5.1: Comparativo entre a quantidade de dados utilizados no *Random Forest*.

Passagem de dados para o Random Forest	**Quantidade de dimensões dos dados de entrada**
Passagem dos resíduos	$322(n_{\text{resíduos}})$
Passagem das similaridades	$52(n_{\text{similaridade}})$

Ainda no caso de se utilizar o *Random Forest*, é possível fazer as mesmas variações que foram feitas no caso onde o SBM é tratado como classificador. Foram variados os modos de se escolher os estados representativos de cada classe, assim como os parâmetros inerentes de cada um destes. Mais uma vez, cada um dos modelos foi testado para as diferentes funções de similaridades e para diferentes valores do parâmetro γ, tanto para o SBM propriamente dito quanto para sua particularização, o AAKR.

No *Random Forest* foram utilizadas 150 árvores para cada modelo, e os demais parâmetros foram os padrões do sklearn [23]. Estes são:

- p: Número de *features* por árvore

$$p_{\text{similaridade}} = \lfloor \sqrt{n_{\text{similaridade}}} \rfloor = \lfloor \sqrt{52} \rfloor = 7$$
$$p_{\text{resíduos}} = \lfloor \sqrt{n_{\text{resíduos}}} \rfloor = \lfloor \sqrt{322} \rfloor = 17;$$

- Critério de separação: índice Gini;
- Uso do número máximo de núcleos, paralelizando o processo: $\mathrm{n}_{jobs} = -1$
- A profundidade máxima é ilimitada, sendo expandida até chegar a nós com apenas uma classificação.

5.2 Métricas Para Avaliação de Desempenho

Nessa seção serão apresentadas as métricas utilizadas para avaliar o desempenho dos modelos criados. Algumas dessas métricas são aplicadas ainda no conjunto de

treino, a fim de se evitar o *overfitting* e para escolher o parâmetros do modelo, outras são aplicadas apenas no conjunto de teste, para avaliar o modelo.

5.2.1 Validação Cruzada

A validação cruzada é um método de estimação de desempenho do modelo. Este método permite avaliar o comportamento do sistema quando o mesmo receber dados que ainda não são conhecidos pelo mesmo. Com isto, mede-se a capacidade de generalização do sistema. Esse tipo de procedimento ocorre com frequência em sistemas que possuem muito parâmetros para serem ajustados. Visto que o conjunto de testes tem como função apenas avaliar o modelo já pronto, este conjunto não pode ser usado para ajustar tais parâmetros.

Entretanto, criar um novo subconjunto para fazer os devidos ajustes pode ser inviável em casos com um reduzido número de amostras. Então, ao invés de criar o conjunto de validação, é feita a validação cruzada. Esta pode ser feita de diversas formas. Uma das maneiras mais conhecidas é a *k-fold*. Na validação cruzada *k-fold* o conjunto de treino é divido em k subconjuntos de iguais tamanhos e um desses conjuntos é designado a ser o conjunto de validação. Todos os demais farão parte do conjunto de treino que irá gerar o modelo. Com isso, o modelo é avaliado com o dados da validação. Esse processo é repetido k vezes, gerando k diferentes resultados, visto que a divisão é feita de forma aleatória. Finalmente, o resultado é dado pela média dos resultados parciais, assim como o desvio padrão.

Fazer o treinamento utilizando a validação cruzada é extremamente custoso, visto que ao invés de treinar o modelo apenas uma vez para cada perfil de simulação, o mesmo será treinado k vezes. Apesar desse esforço computacional bem superior, essa abordagem permite uma etapa de treinamento mais eficiente para situações onde a quantidade de dados é bem reduzida. Outra vantagem de proceder com o treinamento dessa maneira é o fato de que cada amostra do conjunto de treino será usada para validação do modelo apenas uma vez, e $k - 1$ vezes ela será usada para criá-lo, com isso diminuindo a ocorrência de *overfitting*. Neste trabalho, foram utilizados 10 *folds*, valor esse normalmente utilizado na literatura.

5.2.2 Matriz de Confusão

A matriz de confusão é uma maneira de apresentar os resultados de classificação obtidos no conjunto de testes, utilizada com o intuito de analisar o desempenho do modelo. Esta também se mostra eficiente no estudo do comportamento de cada classe. Nela, mostra-se, para cada classe, a quantidade de amostras que faziam parte daquela classe e como elas foram classificadas. Para ilustrar, na Tabela 5.2 tem um exemplo de uma matriz de confusão.

Tabela 5.2: Exemplo de matriz de confusão para 3 classes.

		Classificação		
	Classes	Classe 1	Classe 2	Classe 3
Classe Alvo	Classe 1	23	5	12
	Classe 2	0	45	3
	Classe 3	0	3	37

Cada elemento, $C_{i,j}$, representa a quantidade de amostras que faziam parte da classe i e foram classificadas como integrantes da classe j. Idealmente, almeja-se uma matriz apenas com elementos em sua diagonal, representando um acerto de 100% no conjunto de testes. No exemplo dado, o classificador apresenta uma taxa de acerto igual à 82.03%, resultado relativamente bom em determinadas aplicações. Entretanto, fica claro que o sistema não apresenta bons resultados quando se trata de classificar corretamente a classe 1, uma vez que apresenta taxa de acerto igual à 57.5%. Esse exemplo demonstra o quão esclarecedora a matriz de confusão pode ser na análise da classificação de cada classe.

5.2.3 Acurácia

Outra medida utilizada sobre os resultados do conjunto de testes é a acurácia. Esta medida fornece a taxa de acerto do sistema, checando a capacidade de generalização do sistema, visto que esse conjunto de dados é desconhecido pelo sistema até então. A acurácia (ACC) é expressa por:

$$\text{ACC} = \frac{\sum_{i=1}^{m} C_{i,i}}{\sum_{i=1}^{m} \sum_{j=1}^{m} C_{i,j}}, \tag{5.2}$$

onde m é o número de classes e $C_{i,j}$ é o elemento na linha i e coluna j da matriz de confusão.

Apesar de ser uma medida que dá uma noção geral do desempenho do sistema, a acurácia pode causar erros de interpretação em sistemas em que as quantidades de amostras em cada classe estão desbalanceadas. Por exemplo, se no caso anterior a classe 1 tivesse 5 amostras e todas classificadas de forma incorreta, a acurácia do sistema seria 0.88, que é um valor razoavelmente bom. Entretanto, este não é um modelo adequado, visto que o mesmo não conseguiria classificar corretamente a classe 1.

5.3 Roteiro de Experimentos

Na Tabela 5.3 encontra-se um resumo de todas as possíveis variações de parâmetros do modelo que serão analisadas na próxima seção. Nota-se que existe uma vasta

quantidade de perfis de simulação, totalizando 7.500 possibilidades. Apesar de existirem modelos que são praticamente equivalentes (por exemplo, o método de seleção com limiar de similaridade utilizando $\tau = 0.95$ é bem próximo ao método onde todos os estados são representativos), todas as possibilidades foram exploradas.

Tabela 5.3: Possíveis variações do modelo do sistema.

Variações		**Possibilidades**	**Total**
Modelo do SBM		SBM e AAKR	2
Método de seleção de estados representativos	Todos fazem parte	-	1
	Limiar de similaridade	$\tau \in \{0.05, 0.10, 0.15, \cdots, 0.95\}$	19
	Método original	amostragem $= 2, 3, 5, 7$ e 11	5
Funções de similaridade		RBF, IES, IMK, EXP e CCK	5
Parâmetro γ		$\gamma = 0.01, 0.1, 0.5, 1$ e 10	5
Classificador	SBM	-	1
	Random Forest	Transformações: residual e similaridade	2
Norma-p		$p = 1$ e 2	2

Na Seção 5.4 será apresentada uma série de resultados seguindo a linha cronológica na qual os modelos foram aumentando em complexidade, buscando sempre aumentar a sua eficiência. A cada passo foi tomada uma decisão no sentido de qual dos caminhos foi levado para as simulações seguintes, por isso a metodologia utilizada para comparar os modelos entre si será a validação cruzada, visto que ela analisa melhor a capacidade de generalização do sistema.

Os quatro experimentos feitos, em ordem cronológica, foram:

- Experimento 1:

  Esse experimento buscou determinar a influência das *features* extraídas, a fim de confirmar que o uso das 47 características gera os melhores resultados;

- Experimento 2:

  O intuito dessa etapa é confirmar o ganho de eficiência no modelo para o uso da versão do SBM geral em relação a particularização feita pelo AAKR;

- Experimento 3:

  Essa etapa buscou avaliar os classificadores, explorando a viabilidade do uso do próprio SBM para cumprir essa função. Assim como a melhor forma de passagem de informação entre o SBM e o *Random Forest*; e

- Experimento 4:

  Por fim, foram feitos os últimos ajustes de parâmetros para cada um dos métodos de treinamento do SBM.

5.4 Resultados do Treinamento

Os experimentos foi realizados na ordem que serão apresentados. Essa ordem foi feita aumentando-se a complexidade do sistema, procurando pelas configurações que otimizem o uso do SBM. Para a tomada de decisões serão usados, na maioria das vezes, os resultados obtidos na etapa de treino do modelo a partir da validação cruzada.

5.4.1 Experimento 1

O primeiro passo foi explorar a influência das *features* frequenciais (amplitudes significativas) e das estatísticas (curtose, entropia e média) junto à frequência de rotação. Foram utilizados três conjuntos de características:

- Utilizando a frequência de rotação e as amplitudes significativas;
- Utilizando a frequência de rotação e as medidas estatísticas; e
- Utilizando todas as *features*.

Para esse teste foram utilizados alguns perfis básicos de simulação. Todos utilizam o SBM como classificador e com a particularização feita pelo AAKR, função de similaridade RBF com norma ℓ_2 (utilizando $\gamma = 0.1$, 0.5 e 1.0), e para o método para escolha de estados significativos foram usados o método original com amostragem igual a 5, o método do limiar com $\tau = 0.6$ e o método onde todas as amostras são inclusas. Na Tabela 5.4 serão apresentados os resultados obtidos na validação cruzada para cada um dos casos explorados nesse primeiro experimento.

Tabela 5.4: Resultados para o SBM, utilizando a particularização AAKR, como classificador e função de similaridade RBF com norma ℓ_2.

Método de Treinamento de D	γ	R_f + **medidas frequenciais**	R_f + **medidas estatísticas**	**Todas as medidas**
Método original (amostragem = 5)	0.1	0,3785	0.6384	0.6728
	0.5	0.4410	0.7452	0.6836
	1.0	0.4473	0.7752	0.6495
Método do limiar ($\tau = 0.6$)	0.1	0.3998	0.5186	0.6802
	0.5	0.4961	0.7283	0.7890
	1.0	0.5331	0.7662	0,7909
Método com todas as amostras	0.1	0.4036	0.6556	0,7114
	0.5	0.4916	0,7654	0.7964
	1.0	0.5246	0.7801	0.7897

Pode se ver que separadamente as medidas estatísticas são mais discriminativas que as frequenciais, ao menos no caso do problema aqui tratado. Entretanto, ganhos significativos são obtidos ao se trabalhar com todas as 46 *features*.

A partir desse ponto, todas as simulações levarão em consideração todo o conjunto de características extraídas durante a etapa de pré-processamento. O experimento a seguir visa averiguar os ganhos ao se abrir mão da particularização do SBM.

5.4.2 Experimento 2

A Seção 4.1.1 apresentou a possibilidade de, durante o treinamento do SBM, utilizar uma particularização para a matriz $\mathbf{G}$ onde considera-se todas os estados representativos sendo dissimilares, acarretando que $\mathbf{G}$ seja igual à matriz identidade. Esse experimento analisa o quanto os resultados são afetados pelo uso do AAKR.

Nesse caso, os perfis de simulação usados contaram com o SBM como classificador, utilizando norma ℓ_2 e função de similaridade RBF (utilizando $\gamma = 0.1$, 0.5 e 1.0), e para o método de escolha de estados significativos foram usados o método original com amostragem igual a 5, o método do limiar com $\tau = 0.6$ e o método onde todas as amostras são inclusas. A tabela a seguir traz os resultados obtidos pelos modelos descritos.

Tabela 5.5: Resultados para o SBM e para o AAKR utilizando todo o conjunto de *features* e função de similaridade RBF com norma ℓ_2.

Método de Treinamento de D	γ	**SBM**	**AAKR**
Método original (amostragem = 5)	0.1	0.7458	0.6728
	0.5	0.6784	0.6836
	1.0	0.6404	0.6495
Método do limiar ($\tau = 0.6$)	0.1	0.7244	0.6802
	0.5	0.7962	0.7890
	1.0	0.7887	0,7909
Método com todas as amostras	0.1	0.8317	0,7114
	0.5	0.8072	0.7964
	1.0	0.7997	0.7897

Em poucos casos o AAKR se mostrou mais eficaz que o SBM, e nesses casos a vantagem é bem pequena. Também vale notar que as maiores taxas de acerto são provenientes de modelos que fazem uso do SBM. Portanto, o resultado é bastante esclarecedor em relação as perdas ao se utilizar o AAKR. Então, decidiu-se que as simulações seguintes contariam com a versão do SBM que calcula a matriz $\mathbf{G}$. A próxima variação a ser explorada é em relação ao classificador, onde deseja-se avaliar

a contribuição dos blocos SBM e do *Random Forest*.

5.4.3 Experimento 3

Na Seção 5.1 foram apresentados as possibilidades para o classificador utilizado pelo modelo. A fim de estudar a contribuição tanto do SBM quanto do *Random Forest*, quatro modos foram utilizados:

- SBM como classificador;
- *Random Forest* como classificador, sem o uso do SBM;
- *Random Forest* como classificador, utilizando o SBM transmitindo as similaridades; e
- *Random Forest* como classificador, utilizando o SBM transmitindo os resíduos.

Mais uma vez foram utilizados 9 perfis de simulação para comparar as três possibilidades, sendo estes os mesmos usados nos casos anteriores. A Tabela 5.6 traz os resultados obtidos, onde RF significa *Random Forest*. No caso do RF atuando sozinho, só um perfil de simulação foi utilizado (visto que todas as variações trabalham em cima dos parâmetros do SBM), e este obteve média de acerto de 0.9010.

Tabela 5.6: Resultados para o SBM utilizando todo o conjunto de *features* e função de similaridade RBF com norma ℓ_2.

		Classificador		
Método de Treinamento de D	γ	**SBM**	**RF + SBM com resíduos**	**RF + SBM com similaridades**
Método original (amostragem = 5)	0.1	0.7458	0.9735	0.9539
	0.5	0.6784	0.9706	0.9447
	1.0	0.6404	0.9661	0.9426
Método do limiar ($\tau = 0.6$)	0.1	0.7244	0.9401	0.9520
	0.5	0.7962	0.9214	0.9401
	1.0	0.7887	0.7479	0.9328
Método com todas as amostras	0.1	0.8317	0.3906	0.9581
	0.5	0.8072	0.3833	0.9298
	1.0	0.7997	0.3769	0.9275

Os resultados mostram que o *Random Forest* é mais discriminador do que os perfis utilizados para o SBM. Entretanto, utilizar os dois juntos melhora consideravelmente os resultados.

Conforme explicitado pelos resultados, na maioria dos perfis de simulação apresentados, o caso de em que se passa apenas as similaridades de cada classe estimadas

pelo SBM para o *Random Forest* possui resultado superior ao caso em que é transmitido todo o resíduo. Entretanto, os três melhores resultados foram obtidos utilizando a passagem dos resíduos como um todo. Comparando o melhor resultado de cada um dos modos, tem-se uma redução no erro de estimação de 4.19% para 2.65%.

Ainda analisando a Tabela 5.6 nota-se que o método em que todos as amostras fazem parte da matriz **D** começa a perder para os outros métodos de treinamento ao fazer uso do *Random Forest*. Então, para o experimento a seguir esse método não será considerado.

Tendo concluído este experimento, o modelo escolhido para as próximas simulações conta com o SBM completo, passando os resíduos para o *Random Forest* atuar como classificador, e fazendo uso de todas as características. As escolhas dos demais parâmetros serão feitas no próximo experimento, assim como a análise da escolha do método de treinamento da matriz **D**.

5.4.4 Experimento 4

Este último experimento visa otimizar cada método de treinamento do SBM e depois compará-los. Para isto, para cada função de similaridade e para cada método de treinamento, serão apresentados os melhores valores para a validação cruzada. A Tabela 5.7 traz os resultado desse experimento, onde a última coluna da tabela mostra os parâmetros que geraram o melhor modelo para cada uma das 10 configurações.

Tabela 5.7: Resultado da validação cruzada para diferentes modelos, utilizando todas as *features*, *Random Forest*, com passagem dos resíduos pelo SBM.

Método de Treinamento	**Função de Similaridade**	**Média de Acerto**	**Desvio Padrão**	**Parâmetros Utilizados**		
				Norma	γ	s **ou** τ
Método Original	IES	0.9891	0.0058	1	0.01	7
	RBF	0.9802	0.0082	2	0.1	11
	IMK	0,9862	0.0090	2	0.1	5
	CCK	0.9856	0.0079	1	1	7
	EXP	0.9857	0.0093	1	0.1	11
Método do Limiar	IES	0.9868	0.0089	1	0.01	0.9
	RBF	0.9666	0.0117	2	0.01	0.55
	IMK	0.9856	0.0137	2	0.1	0.8
	CCK	0.9872	0.0078	2	0.1	0.5
	EXP	0.9833	0.0114	1	0.1	0.5

Vale notar, que todos os 10 modelos apresentam resultados superiores aos encontrados em outros trabalhos [1], [6], reforçando a viabilidade do uso do SBM para o problema de detecção automática de falhas.

Pode-se observar que em ambos os métodos de treinamento as funções de simila-

ridade IES e CCK possuem resultados superiores às demais, uma vez que possuem as maiores médias e menores desvios padrão. Entretanto, fica difícil a distinção entre os 4 modelos, uma vez que todos possuem faixas de erro bem próximas. Estes modelos são:

- Função de similaridade IES, com $\gamma = 0.01$, utilizando norma ℓ_1 e método de treinamento original com amostragem = 7 (Modelo 1);
- Função de similaridade CCK, com $\gamma = 1$, utilizando norma ℓ_1 e método de treinamento original com amostragem = 7 (Modelo 2);
- Função de similaridade CCK, com $\gamma = 0.1$, utilizando norma ℓ_2 e método de treinamento do limiar com $\tau = 0.5$ (Modelo 3); e
- Função de similaridade IES, com $\gamma = 0.01$, utilizando norma ℓ_1 e método de treinamento do limiar com $\tau = 0.9$ (Modelo 4).

Alguns pontos desses quatro modelos serão discutidos a seguir.

Parâmetro γ

É interessante notar que dentre os quatro modelos, aparecem três valores diferentes para o γ ótimo. A seguir, será avaliada a robustez de cada modelo em relação a escolha do valor do γ, dentre os cinco que foram testados: $\gamma \in \{0.01, 0.1, 0.5, 1, 10\}$.

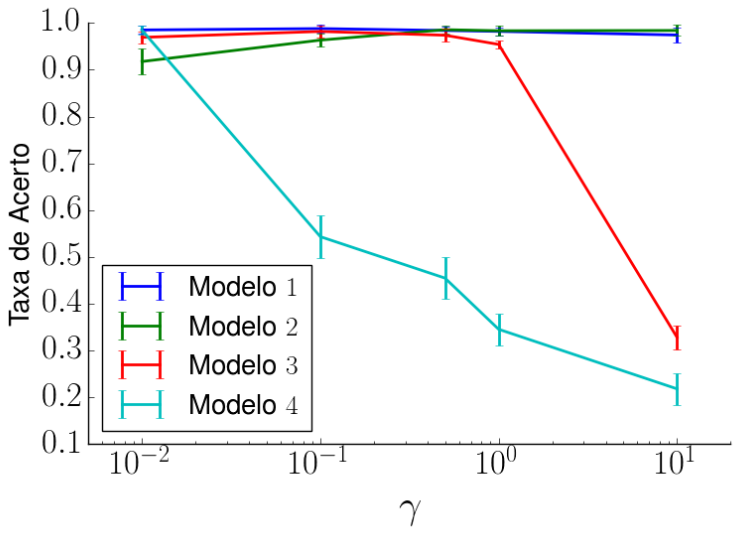


Figura 5.2: Resultado da taxa de acerto da validação cruzada para os modelos estudados.

Na Figura 5.3 têm-se os gráficos da taxa de acerto da validação cruzada para os quatro modelo, junto ao desvio padrão, conforme varia-se γ. Pode-se notar que

os Modelos 3 e 4 são extremamente sensíveis à variação do γ. A seguir, o mesmo gráfico será apresentado, só que com *zoom* na região com taxa de acerto entre 0.8 e 1.0:

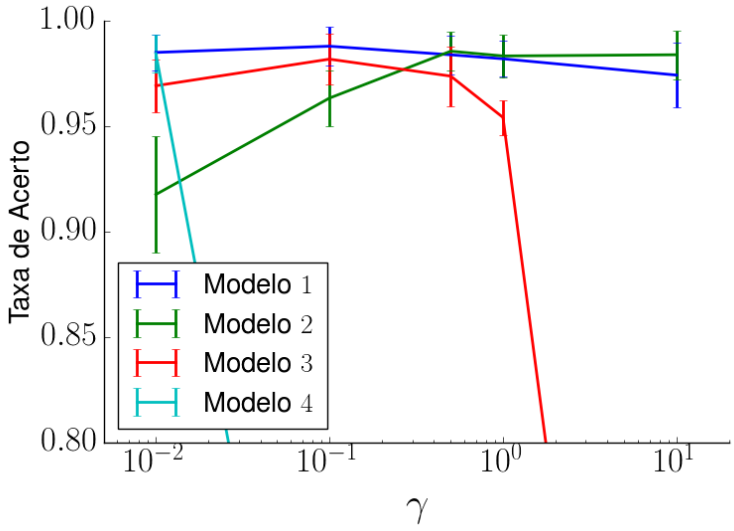


Figura 5.3: *Zoom* na faixa de taxa de acerto entre 0.8 a 1.0 para todos os modelos.

Nos Modelos 2 e 3 é usada a função de similaridade CCK. No Modelo 2 os resultados melhoram um pouco conforme o valor de γ cresce, só que a partir de certo ponto o modelo estabiliza. Já o Modelo 3 também começa melhorando, e depois acaba perdendo eficiência, até que em $\gamma = 1$ ocorre uma queda brusca nos resultados.

A Figura 5.3 mostra a tendência para o comportamento da taxa de acerto do Modelo 1. Apesar de cair de forma bem suave, o resultado diminui conforme aumenta-se γ. Este comportamento é apresentado também pelo Modelo 4, só que de forma mais abrupta. Ambos os modelo fazem uso da função de similaridade IES, o que leva a crer que o γ ótimo para essa função é de fato 0.01.

Esse estudo ao redor dos valores de γ mostra a robustez do sistema em relação a este parâmetro. O fato de existir um candidato ao valor ótimo de γ e resultados bem inferiores para demais valores indica que o sistema é dependente deste parâmetro. Já a existência de valores próximos ao ótimo faz com que o sistema não necessite de uma busca fina pelo valor ótimo de γ, uma vez que existem valores para este parâmetro que também produzem modelos com altíssimo desempenho, ou seja, o modelo é robusto a pequenas variações de γ.

Custo Computacional

Em relação à diferença de custo computacional entre os dois métodos de construção da matriz $\mathbf{D}$, vale ressaltar que o método do limiar demanda muito mais na fase de

treinamento do SBM. Entretanto, ele busca pela matriz de estados representativos menos redundante, acarretando na eventual redução de amostras na matriz $\mathbf{D}$. A Tabela 5.8 traz essa comparação para os 4 modelos que apresentaram os melhores resultados. As seis classes do sistema em questão são:

- Máquina atuando livre de falhas (NR);
- Falha de desbalanceamento (DSB);
- Falha de desalinhamento horizontal (DH);
- Falha de desalinhamento vertical (DV);
- Falha no mancal não invertido (MNI); e
- Falha no mancal invertido (MI).

A Tabela 5.8 traz o comparativo entre as dimensões das matriz $\mathbf{D}$ de cada classe e de cada modelo. O resultado em questão é a média das dimensões das respectivas matrizes geradas na etapa de treinamento, onde é feita a validação cruzada *k-fold*, calculando cada matriz $\mathbf{D}$ k vezes. Como já mencionado, foi utilizado $k = 10$.

Tabela 5.8: Média das dimensões das matrizes de estados significativos do SBM para cada classe e cada um dos modelos escolhidos geradas na etapa de treinamento.

Configuração	**NR**	**DSB**	**DH**	**DV**	**MNI**	**MI**
Modelo 1	33.1	87.6	65.7	80.1	131.2	118.
Modelo 2	33.3	76.2	60	68.4	110.2	98.8
Modelo 3	5	73.5	4.9	5	74.5	6
Modelo 4	5.9	78.8	8	8	94.5	29.8

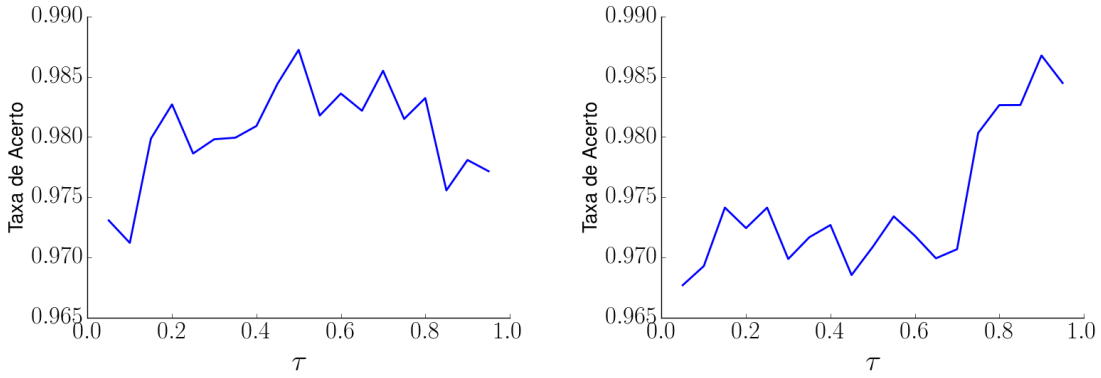
É notável a redução na dimensão das matrizes de estados significativos. Apesar de uma etapa de treinamento muito mais demorada, existe a vantagem de uma etapa de testes mais rápida, visto que a matriz $\mathbf{D}$ já está estruturada. Então, se o modelo for usado em uma aplicação *online*, o método do limiar é mais atraente.

Nas classes referentes a falhas de desbalanceamento e a falhas no mancal não invertido a redução nas dimensões pelo método do limiar é bem menor que nas outras classes. Essas falhas são mais custosas de serem definidas, visto que para gerar falhas nos mancais é necessário criar um desbalanceamento. Com isso, mais amostras serão necessárias na matriz $\mathbf{D}$ de cada uma das classes.

Comparando os resultados entre os Modelos 3 e 4, nota-se uma considerável redução nas dimensões das matrizes de estados representativos. Isso se dá pelo fato do Modelo 3 utilizar $\tau = 0.5$, sendo mais seletivo no que se trata da inclusão de novos estados na matriz $\mathbf{D}$ em relação ao outro modelo, que faz uso de $\tau = 0.9$.

Parâmetro τ

Os Modelos 3 e 4 fazem uso do método de treinamento do limiar e, consequentemente, dependem do parâmetro τ. A seguir será apresentado para os dois modelos o comportamento dos resultados ao se variar τ no intervalo de 0.5 à 0.95 com passo de 0.5:



(a) Modelo 3.

(b) Modelo 4.

Figura 5.4: Resultados obtidos na validação cruzada para todos os diferentes valores de τ.

O aumento do valor de τ acarreta na adição de mais amostras à matriz de estados significativos, uma vez que o critério de seleção de estados fica menos seletivo. Com isso, aumenta-se a complexidade do problema. Pelos gráficos da Figura 5.4 têm-se que esse aumento eventualmente causa queda no desempenho do sistema. Essa queda na eficiência do classificador já era de se esperar, uma vez que a partir de certo ponto amostras redundantes e contaminadas com ruído começarão a serem adicionadas ao conjunto de estados representativos.

Amostragem

Já os Modelos 1 e 2 fazem uso da técnica original para montar a matriz $\mathbf{D}$ de cada classe do SBM. Portanto, ambos dependem do fato de subamostragem escolhido no segundo passo do treinamento do algoritmo, como explicado na Seção 4.1.2. Na Figura 5.5 têm-se o gráfico comparativo entre os dois modelos.

Conforme o fator de subamostragem aumenta, menos amostras serão selecionadas como significativas e irão compor a matriz $\mathbf{D}$ da respectiva classe. O resultado reforça o que foi constatado ao analisar o parâmetro τ: o grande aumento na dimensão de $\mathbf{D}$ eventualmente acarreta em perda de desempenho do sistema.

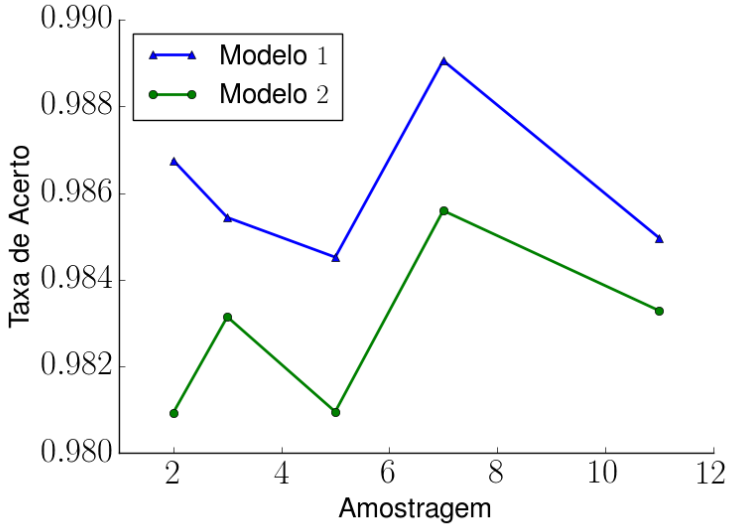


Figura 5.5

5.5 Resultado da Etapa de Testes

Nessa seção os quatro modelos previamente selecionados serão avaliados com os respectivos resultados na etapa de testes. No caso, todos os modelos irão classificar o mesmo conjunto de teste, o que não ocorre durante a etapa de treinamento devido o uso da validação cruzada *k-fold*. Serão apresentados e analisados os resultados utilizando a acurácia e a matriz de confusão.

Na tabela a seguir têm-se os resultados de etapa de teste para os 4 modelos:

Tabela 5.9: Acurácia no conjunto de testes para os 4 modelos escolhidos.

Configuração	**Acurácia**
Modelo 1	0.9849
Modelo 2	0.9798
Modelo 3	0.9747
Modelo 4	0.9849

O Modelo 3 apresenta resultados abaixo do esperado de acordo com o treinamento do mesmo, levando a crer que o mesmo não possua boa capacidade de generalização. Os demais modelos apresentaram resultados coerentes com os obtidos durante o treinamento.

Conforme apresentado na seção anterior, a escolha para o modelo depende da aplicação. O Modelo 1 apresenta resultados ligeiramente mais precisos que os obtidos pelo Modelo 4. Entretanto, se o modelo for feito para uma aplicação que exija alto tempo de resposta durante etapas *online*, o Modelo 4 é uma escolha mais apropriada.

Por fim, foi feita uma investigação a respeito dos erros de classificação durante a etapa de testes. A seguir, as tabela 5.10, 5.11, 5.12 e 5.13 trazem as matrizes de

confusão para cada um dos resultados.

Tabela 5.10: Matriz de confusão para o Modelo 1.

Classe	NR	DSB	DH	DV	MNI	MI
NR	4	0	1	0	0	0
DSB	0	34	0	0	0	0
DH	0	0	20	0	0	0
DV	0	0	0	31	0	0
MNI	0	1	0	0	55	0
MI	0	1	0	0	0	51

No primeiro modelo notam-se dois erros de classificação de falhas nos mancais (um em cada mancal), sendo ambas classificadas como falha de desbalanceamento. Existe também um falso positivo, sendo o estado classificado como falha de desalinhamento horizontal.

Tabela 5.11: Matriz de confusão para o Modelo 2.

Classe	NR	DSB	DH	DV	MNI	MI
NR	4	0	1	0	0	0
DSB	0	34	0	0	0	0
DH	0	0	20	0	0	0
DV	0	0	0	31	0	0
MNI	0	1	0	0	54	1
MI	0	1	0	0	0	51

No Modelo 2 ocorrem os mesmos tipos de erro, com um adicional, onde a falha no mancal não invertido é classificada como falha no mancal invertido.

Tabela 5.12: Matriz de confusão para o Modelo 3.

Classe	NR	DSB	DH	DV	MNI	MI
NR	4	0	1	0	0	0
DSB	0	34	0	0	0	0
DH	0	0	20	0	0	0
DV	0	0	0	31	0	0
MNI	0	2	0	0	53	1
MI	0	1	0	0	0	51

No Modelo 3, tem-se um erro a mais de classificação em relação aos encontrados no caso anterior. Este erro também é uma falha em um dos mancais sendo classificada como falha de desbalanceamento.

Por fim, no Modelo 4, cuja matriz de confusão encontra-se na Tabela 5.13, surge um novo erro, uma falha de desalinhamento horizontal sendo classificada como falha de desalinhamento vertical. Em contra-partida, esse modelo foi capaz de classificar todas as classes normais corretamente.

Tabela 5.13: Matriz de confusão para o Modelo 4.

Classe	NR	DSB	DH	DV	MNI	MI
NR	5	0	0	0	0	0
DSB	0	34	0	0	0	0
DH	0	0	19	1	0	0
DV	0	0	0	31	0	0
MNI	0	1	0	0	54	1
MI	0	0	0	0	0	52

O resultado mostra que o sistema é capaz de generalizar para demais conjuntos. Entretanto, apresenta dificuldade ao classificar classes normais. As prováveis causas disso são a pequena quantidade de experimentos feitos com a máquina operando livre de falhas, apenas 49, ou então uma falha durante a etapa de aquisição de dados. Apesar de não ser ideal, detectar uma falha erroneamente não é algo tão crítico quanto deixar de detectar uma falha, uma vez que, para efeitos práticos, a perda de alvo provavelmente irá gerar altos prejuízos a fábrica.

Outros erros recorrentes ocorreram durante a classificação de experimentos com falhas nos mancais, sendo essas classificadas como falhas de desbalanceamento. Isso se dá devido à necessidade de se desbalancear o sistema a fim de simular falhas nos mancais. Então, em casos específicos, esse desbalanceamento pode ser mais relevante que as falhas induzidas nos mancais.

5.6 Comparação Com Outros Trabalhos

Como já mencionado no decorrer deste projeto, diversos outros trabalhos já abordaram o mesmo problema, utilizando a mesma base de dados ou uma versão reduzida da mesma. Em [1] foi criada a base de dados, e também foram testadas diversos classificadores. O problema foi abordado utilizando apenas 19 *features*, sendo estas as medidas frequenciais e a frequência de rotação do rotor. Neste trabalho tentou-se separar apenas entre três classes: funcionamento normal, desbalanceamento e desalinhamento, incluindo os dois tipos, vertical e horizontal.

Foram utilizadas quatro maneiras de classificar os dados: *Artificial Neural Networks* (ANN), *Random Forest*, regressão logística e Support Vector Machine (SVM). A Tabela 5.14 a seguir apresenta os resultados obtidos.

Os resultados são promissores. Entretanto, o problema resolvido é consideravelmente mais simples que o proposto pelo presente trabalho. Foi feito também uma extensão da base de dados onde não ocorre falha nenhuma. Esse processo foi feito adicionando ruído branco aos sinais simulados.

Outra maneira de tratar o problema foi proposta em [24]. Neste trabalho foram

Tabela 5.14: Resultados obtidos em [1]

Classificador	**Taxa de Acerto (%)**
Random Forest	100
SVM	99.5
ANN	99.1
Regressão logísitica	90.2

usados dois meios para classificar os dados: ANN e *Random Forest*. Três experimentos foram feitos:

- Experimento 1: Utilizando apenas as características adquiridas pelos sinais de um dos acelerômetros, contando com 19 *features*. Neste caso procura-se classificar entre as mesmas seis classes do presente trabalho;
- Experimento 2: Passa-se a usar as características provenientes do outro acelerômetro, utilizando 31 *features* no total. Nesse experimento também procura-se classificar dentre seis classes; e
- Experimento 3: Utiliza-se o mesmo vetor de característica do Experimento 2, mas dessa vez tenta-se classificar os dados entre nove classes, tentando diferenciar entre as diferentes falhas nos mancais.

A seguir, os resultados obtidos por essa abordagem:

Tabela 5.15: Taxa de acerto obtida em cada experimento por cada um dos classificadores utilizados em [24].

Experimento	**Classificador**	
	Random Forest	**ANN**
1	94.3	91.7
2	97.0	95.9
3	97.5	95.5

Repare que qualquer um dos quatro modelos escolhidos para o SBM apresentam resultados superiores aos apresentados na Tabela 5.15.

Em [6] essa base de dados também é estudada. Nele a classificação é feita por meio de redes neurais *perceptron* com múltiplas camadas, e a quantidade de *features* utilizadas foi variada. Em primeiro momento tentou-se classificar entre apenas três classes: funcionamento normal, falha de desbalanceamento e de desalinhamento. Os resultados mostraram que o uso das medidas estatísticas junto às medidas frequenciais geram melhores resultados, tendo obtido 99.1% de acerto.

Entretanto, o mesmo sistema é colocado para classificar o sistemas dentre as seis classes, tentando encontrar as falhas nos mancais assim como distinguir desalinha-

mento vertical do horizontal. Nessa nova formulação, o sistema alcança resultados abaixo do anterior, atingindo taxa de acerto de 95.8%.

Então, o sistema aqui proposto se mostrou superior às técnicas que já tentaram resolver o mesmo problema. Os ganhos nos resultados podem ser atribuídos ao uso do SBM assim como a expansão das *features* utilizadas, uma vez que passa-se a usar os sinais provenientes do microfone assim como as médias dos sinais extraídos.

Capítulo 6

Conclusão e Trabalhos Futuros

6.1 Conclusão

Foi proposto uma abordagem alternativa para classificar falhas em máquinas rotativas de forma automática, de forma a evitar falhas humanas devido a natureza complexa do problema. Durante o estudo dos sinais de vibração dos equipamentos de aquisição de dados procurou-se pelas características dos mesmos que representassem as informações relevantes no problema de classificação. Foi utilizado o *Similarity-Based Modeling* para monitorar as falhas, a fim de ajudar na classificação das mesmas. O SBM é capaz de procurar pelo menor número de amostras representativas que sejam suficientemente eficazes para descrever cada classe, fazendo isso de forma não-paramétrica.

Feito o monitoramento de cada amostra, foi feita a sua classificação através do *Random Forest*. Este foi escolhido por ser um classificador conhecido na bibliografia por tratar bem de modelos que possuam muita variância, que é o caso estudado. Foram testados diversos perfis de simulação de modo a encontrar os parâmetros do modelo capazes de otimizar os resultados.

Ao final da busca pelo modelo mais adequado à base de dados, chegou-se a uma taxa de acerto de 98.5% na etapa de testes. Este resultado é superior aos demais trabalhos desenvolvidos sobre a mesma base de dados, que contaram com o uso de SVM, ANN, e do próprio *Random Forest* [1], [5], [6]. O SBM se mostrou uma ferramenta eficiente na detecção de falhas, até mesmo como classificador. Entretanto, o *Random Forest* como classificador possui resultados superiores, e, aliado ao SBM, alcança acurácia muito alta.

A respeito da otimização do modelo do SBM pôde-se concluir que a norma ℓ_1 é mais descriminante nessa aplicação, e que a função de similaridade IES, com $\gamma = 0.01$, utilizando o treinamento do SBM pelo método original com amostragem igual à 7, produz melhores resultados. Entretanto, caso busque-se por um modelo

que necessite de um tempo de resposta muito alto durante etapas *online*, a melhor configuração é utilizando o método do limiar com $\tau = 0.9$, função de similaridade IES usando norma ℓ_1 e $\gamma = 0.01$.

O sistema requer um tempo relativamento baixo na etapa de treinamento, e mostra alta eficiência em etapas *online*. Fato este que permite a instalação em equipamentos a fim de monitorá-los em tempo real.

6.2 Trabalhos Futuros

A pesquisa na área irá continuar em busca de sistemas aplicáveis em demais bases de dados. As futuras aplicações e mudanças estão a seguir:

- Fazer um busca refinada pelos valores ótimos de γ em cada função de similaridade;
- Existe ainda a possibilidade de testar outros classificadores no lugar do *Random Forest*, como por exemplo o SVM, *Gradient Boosting*. Sendo o último o protagonista nas atuais pesquisas na área de classificação supervisionada;
- Ainda explorando a similaridade, pode-se usar a distância de Mahalanobis para substituir a norma-p. Essa abordagem é promissora pois leva em conta a correlação entre os vetores e no caso de serem descorrelacionados, essa métrica se resume a distância euclidiana [25];
- Para a base de dados do *Rotorkit* mais experimentos podem ser feitos, tanto para criar mais amostras da máquina funcionando livre de falhas, quanto para a inserção de novos defeitos; e
- Aplicação em novas bases de dados, podendo ser bases temporais, a fim de evitar que o defeito de fato ocorra. Nesses casos, é necessário o módulo de prognóstico.

Referências Bibliográficas

[1] LOPEZ, R. Z., *Classificação automática de defeitos em máquinas rotativas*. Projeto de graduação, Universidade Federal do Rio de Janeiro, 2014.

[2] FUJYMOTO, R. Y., *Diagnóstico automático de defeitos em rolamentos baseados em lógica fuzzy*. Ph.D. dissertation, Escola Politécnica da Universidade de São Paulo, 2005.

[3] HERZOG, J., "Making decisions about the best technology to implement", 2011.

[4] WHITE, G. D., *Introduction to machine vibration*. Reliabilityweb.com Press, 2008.

[5] DE LIMA, A. A., PREGO, T. D. M., NETTO, S. L., *et al.*, "On fault classification in rotating machines using fourier domain features and neural networks", *IEEE Fourth Latin American Symposium on Circuits and Systems (LASCAS),*, pp. 1–4, 2013.

[6] VIANA, D. P., LOPES, R. Z., LIMA, A. A. D., *et al.*, "The influence of feature vector on the classification of mechanical faults using neural networks", 2016.

[7] PRUFTECHNIK, "Machinery fault diagnosis", 2011.

[8] SHANNON, C. E., "A mathematical theory of communication", *The Bell System Technical Journal*, v. 27, n. July 1928, pp. 379–423, 1948.

[9] WEGERICH, S. W., "Similarity-based modeling of vibration features for fault detection and identification", *Sensor Review*, v. 25, n. 2, pp. 114–122, 2005.

[10] WEGERICH, S. W., "Similarity based modeling of time synchronous averaged vibration signals for machinery health monitoring", *IEEE Aerospace Conference Proceedings*, v. 6, pp. 3654–3662, 2004.

[11] SINGER, R., GROSS, K., HERZOG, J., "Model-based nuclear power plant monitoring and fault detection: Theoretical foundations", *Proceedings of*

9th International Conference on Inteligent System Application to Power Systems, v. Seoul, Kor, n. July 6-10, 1997.

[12] TOBAR, F. A., YACHER, L., PAREDES, R., *et al.*, "Anomaly detection in power generation plants using similarity-based modeling and multivariate analysis", *Proceedings of the American Control Conference (ACC)*, pp. 1940–1945, 2011.

[13] MOTT, J., PIPKE, M., "Similarity-based modeling of aircraft flight paths", *IEEE Aerospace Conference Proceedings*, v. 3, pp. 1580–1592, 2004.

[14] WEGERICH, S. W., WILKS, A. D., PIPKE, R. M., "Nonparametric modeling of vibration signal features for equipment health monitoring", *IEEE Aerospace Conference Proceedings*, v. 7, pp. 3113–3121, 2003.

[15] MICCHELLI, C. A., "Interpolation of scattered data: Distance matrices and conditionally positive definite functions", *Constructive Approximation*, v. 2, n. 1, pp. 11–22, 1986.

[16] BASAK, J., "A least square kernel machine with box constraints". In: *International Conference on Pattern Recognition*, dec 2008.

[17] GARVEY, J., GARVEY, D., SEIBERT, R., *et al.*, "Validation of on-line monitoring techniques to nuclear plant data", *Nuclear Engineering and Technology*, v. 39, n. 2, pp. 133–142, 2007.

[18] HERZOG, J. P., WEGERICH, S. W., GROSS, K. C., *et al.*, "MSET modeling of crystal river-3 Venturi flow meters", *6th International Conference on Nuclear Engineering*, , n. May 10-15, 1998.

[19] HASTIE, T., TIBSHIRANI, R., FRIEDMAN, J., *The elements of statistical learning*, v. 1. 2 ed. New York, Springer New York Inc., 2009.

[20] LOUPPE, G., *Understanding Random Forests: From theory to practice*. Ph.D. dissertation, University of Liège, 2014.

[21] CERIANI, L., VERME, P., "The origins of the Gini index: extracts from Variabilità e Mutabilità (1912) by Corrado Gini", *The Journal of Economic Inequality*, v. 10, n. 3, pp. 421–443, 2012.

[22] BREIMAN, L., "Random forests", *Machine Learning*, v. 45, n. 1, pp. 5–32, 2001.

[23] PEDREGOSA, F., VAROQUAUX, G., GRAMFORT, A., *et al.*, "Scikit-learn: Machine learning in Python", *Journal of Machine Learning Research 12*, v. 12, pp. 2825–2830, 2012.

[24] MARTINS, D. H. C. D. S. S., PESTANA-VIANA, D., PREGO, T. D. M., *et al.*, "Diagnóstico de falhas em máquinas rotativas utilizando Random Forest", *Anais do XXXIV Simpósio Brasileiro de Telecomunicações*, pp. 916–920, 2016.

[25] LINDEN, R., "Técnicas de agrupamento", *Revista de Sistemas de Informação da FSMA*, v. 4, pp. 18–36, 2009.