UNIVERSIDADE ESTADUAL PAULISTA "JÚLIO DE MESQUITA FILHO"
FACULDADE DE ENGENHARIA
CAMPUS DE ILHA SOLTEIRA

ANA PAULA ALVES GUIMARÃES

UTILIZAÇÃO DO ALGORITMO DE APRENDIZADO DE MÁQUINAS PARA MONITORAMENTO DE FALHAS EM ESTRUTURAS INTELIGENTES

Ilha Solteira
2016

ANA PAULA ALVES GUIMARÃES

UTILIZAÇÃO DO ALGORITMO DE APRENDIZADO DE MÁQUINAS PARA MONITORAMENTO DE FALHAS EM ESTRUTURAS INTELIGENTES

Dissertação apresentada à Faculdade de Engenharia do Campus de Ilha Solteira – UNESP como parte dos requisitos para obtenção do título de mestre em Engenharia Mecânica.
Área de Conhecimento: Mecânica dos sólidos.

Prof. Dr. Vicente Lopes Junior
Orientador

Ilha Solteira
2016

Ficha catalográfica

FICHA CATALOGRÁFICA
Desenvolvido pelo Serviço Técnico de Biblioteca e Documentação

G963u

Guimarães, Ana Paula Alves.
Utilização do algoritmo de aprendizado de máquinas para monitoramento de falhas em estruturas inteligentes / Ana Paula Alves Guimarães. -- Ilha Solteira: [s.n.], 2016
123 f. : il.

Dissertação (mestrado) - Universidade Estadual Paulista. Faculdade de Engenharia. Área de conhecimento: Mecânica dos Sólidos, 2016

Orientador: Vicente Lopes Junior
Inclui bibliografia

1. Monitoramento da condição estrutural. 2. Aprendizado de máquina. 3. Estruturas inteligentes. 4. Inteligência artificial. 5. Support vector machine.

UNIVERSIDADE ESTADUAL PAULISTA

Câmpus de Ilha Solteira

CERTIFICADO DE APROVAÇÃO

TÍTULO DA DISSERTAÇÃO: **UTILIZAÇÃO DO ALGORITMO DE APRENDIZADO DE MÁQUINAS PARA MONITORAMENTO DE FALHAS EM ESTRUTURAS INTELIGENTES**

AUTORA: ANA PAULA ALVES GUIMARÃES
ORIENTADOR: VICENTE LOPES JUNIOR

Aprovada como parte das exigências para obtenção do Título de Mestra em ENGENHARIA MECÂNICA, área: MECANICA DOS SÓLIDOS pela Comissão Examinadora:

Prof. Dr. VICENTE LOPES JUNIOR
Departamento de Engenharia Mecânica / Faculdade de Engenharia de Ilha Solteira

Prof. Dr. AMARILDO TABONE PASCHOALINI
Departamento de Engenharia Mecânica / Faculdade de Engenharia de Ilha Solteira

Prof. Dr. PAULO JOSÉ PAUPITZ GONÇALVES
Departamento de Engenharia Mecânica / Faculdade de Engenharia de Bauru

Ilha Solteira, 20 de dezembro de 2016

DEDICO

Aos meus pais **Hélio Guimarães Pereira** e **Maria Helena Alves Ferreira**, à minha irmã **Ana Claudia Alves Guimarães** e aos meus avôs **Geralda Alves Ferreira** e **Mozart Pinto Ferreira** (*in memorian*) pelo amor incondicional que sempre manifestaram.

AGRADECIMENTOS

Inicialmente agradeço ao Universo, na qual retiro minha inspiração, coragem e força para continuar minha jornada.

Agradeço carinhosamente meus familiares que sempre me apoiaram e me fortaleceram nos momentos difíceis e de incertezas.

Agradeço de forma afável meu querido Rodrigo André Bonfanti de Cól pelo auxílio, ternura e compreensão nas oscilações de humor (risos).

Ao meu orientador Vicente Lopes Junior pela paciência e gentileza de repassar seus ensinamentos. Pela oportunidade de aprofundar meus conhecimentos acadêmicos. Pela ajuda nos momentos de dúvida. E, por fim, pelo seu bom humor e pela positividade.

E, finalmente, aos que direta ou indiretamente contribuíram para a consolidação deste trabalho.

“Somos meros exploradores do infinito em busca da perfeição absoluta.”
(frase do filme O Homem Que Viu O Infinito)

RESUMO

O monitoramento da condição estrutural é uma área que vem sendo bastante estudada por permitir a construção de sistemas que possuem a capacidade de identificar um determinado dano em seu estágio inicial, podendo assim evitar sérios prejuízos futuros. O ideal seria que estes sistemas tivessem o mínimo de interferência humana. Sistemas que abordam o conceito de aprendizagem têm a capacidade de serem autômatos. Acredita-se que por possuírem estas propriedades, os algoritmos de aprendizagem de máquina sejam uma excelente opção para realizar as etapas de identificação, localização e avaliação de um dano, com capacidade de obter resultados extremamente precisos e com taxas mínimas de erros. Este trabalho tem como foco principal utilizar o algoritmo *support vector machine* no auxílio do monitoramento da condição de estruturas e, com isto, obter melhor exatidão na identificação da presença ou ausência do dano, diminuindo as taxas de erros através das abordagens da aprendizagem de máquina, possibilitando, assim, um monitoramento inteligente e eficiente. Foi utilizada a biblioteca LibSVM para análise e validação da proposta. Desta forma, foi possível realizar o treinamento e classificação dos dados promovendo a identificação dos danos e posteriormente, empregando as predições efetuadas pelo algoritmo, foi possível determinar a localização dos danos na estrutura. Os resultados de identificação e localização dos danos foram bastante satisfatórios.

Palavras-chave: Monitoramento da condição estrutural. Aprendizado de máquina. Estruturas inteligentes. Inteligência artificial. *Support vector machine*.

ABSTRACT

Structural health monitoring (SHM) is an area that has been extensively studied for allowing the construction of systems that have the ability to identify damages at an early stage, thus being able to avoid serious future losses. Ideally, these systems have the minimum of human interference. Systems that address the concept of learning have the ability to be autonomous. It is believed that by having these properties, the machine learning algorithms are an excellent choice to perform the steps of identifying, locating and assessing damage with ability to obtain highly accurate results with minimum error rates. This work is mainly focused on using support vector machine algorithm for monitoring structural condition and, thus, get better accuracy in identifying the presence or absence of damage, reducing error rates through the approaches of machine learning. It allows an intelligent and efficient monitoring system. LIBSVM library was used for analysing and validation of the proposed approach. Thus, it was feasible to conduct training and classification of data promoting the identification of damages. It was also possible to locate the damages in the structure. The results of identification and location of the damage was quite satisfactory.

Keywords: Structural health monitoring. Machine learning. Smart structures. Artificial intelligence. Support vector machine.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

AM	Aprendizado de Máquina
C - SVC	C – Support Vector Classification
CCDM	Correlation Coefficient Deviation Metric
FN	False Negative
FP	False Positive
GMSINT	Grupo de Materiais e Sistemas Inteligentes
GPU	Grapichs Processing Unit
GPS	General Problem Solver
IA	Inteligência Artificial
IDE	Integrated Development Kit
JDK	Java Kit Development
kHz	Quilohertz
KKT	Karush-Kunh-Tucker
KNN	K-Nearest Neighbors
LT	Logic Theorist
mHz	Megahertz
NU-SVC	NU- Support Vector Classification
PCA	Principal Component Analysis
PZT	Piezoeletric Materials Element
RBF	Radial Basis Function
RMS	Root Mean Square
RMSD	Root Mean Square Deviation
RNA	Rede Neural Artificial
SDK	Software Development Kit
SHM	Structural Heath Monitoring
SVM	Support Vector Machine
TAE	Teoria do Aprendizado Estatístico
TN	True Negative
TP	True Positive
VC	Vapnik-Chervonenkis

LISTA DE SÍMBOLOS

n	Quantidade limitada de elementos
i	Linha da matriz de treinamento
j	Coluna da matriz de treinamento
y	Saída ou rótulo de uma classe
y^*	Saída desejada
k	Parâmetro especificado pelo usuário no algoritmo KNN ou quantidade de classes em um conjunto de dados
x_n	Entradas do neurônio artificial
x_i	*i-ésima* entrada do neurônio artificial ou *i-ésimo* exemplo do vetor de entradas do conjunto de treinamento
w_n	Vetor peso do neurônio
w_i	*i-ésimo* elemento do vetor peso do neurônio
$P(x,y)$	Função de distribuição da probabilidade
R^2	Ambiente bidimensional
m	Dimensão em um determinado espaço
X	Matriz do conjunto de treinamento
w^T	Vetor peso transposto
x	Vetor de entrada
b	Bias
p	Margem de separação
w_o	Vetor peso ótimo
b_o	Bias ótimo
$g(x^{(s)})$	Hiperplano baseado nos vetores de suporte
$x^{(s)}$	Vetores de suporte
$y^{(s)}$	Rótulos dos vetores de suporte
r	Distância algébrica de um ponto x em relação ao hiperplano central
$\|\|w_o\|\|$	Norma euclidiana do vetor peso ótimo
$\phi(w)$	Função de custo
$J(w,b,\alpha)$	Função Lagrangiana baseada nos valores do vetor peso, bias e coeficiente de Lagrange
α_i	Multiplicadores de Lagrange

$\sum_{i=1}^{N}$	Somatório dos elementos compreendidos de 1 até o limite especificado
$Q(\alpha)$	Função objetivo
$\alpha_{o,i}$	Multiplicador de Langrange ótimo
$x^*(1)$	Vetores de suporte com rótulo 1
$x^*(-1)$	Vetores de suporte com rótulo -1
ε_i	Relaxamento das margens
C	Parâmetro de custo
$\phi(w, \varepsilon)$	Função de custo baseado no vetor peso e no relaxamento das margens
φ	Função real transformadora
$\varphi_j(x)$	Função responsável em mapear os vetores de entrada em uma nova dimensão
m_1	Dimensão maior
$\{w_j\}_{j=1}^{m_1}$	Simboliza o agrupamento dos vetores com o novo espaço de características
$\varphi^T(x)\varphi(x_i)$	Produto interno de dois vetores no espaço de características
$\sum_{j=0}^{m_1}$	Somatório dos elementos no espaço de características
$k(x, x_i)$	Função *kernel*
obj	Valor ótimo da função objetivo
rho	Termo bias na implementação do algoritmo SVM
nSV	Número de vetores de suporte livres
nBSV	Número de vetores de suporte limitados
P_{ij}	Caminho atuador i e sensor j
P_{ji}	Caminho atuador j e sensor i
nu	Parâmetro equivalente ao termo C usado no algoritmo nu-SVC
M_1	Métrica um
RMS_d	Valor médio eficaz do sinal atual (dano)
RMS_b	Valor médio eficaz do sinal íntegro
M_2	Métrica dois obtida
$S_d(i)$	Cada ponto do sinal atual (dano)
$S_b(i)$	Cada ponto do sinal íntegro
M_3	Métrica três

$S_b \otimes S_d$	Convolução entre os sinais (íntegro e dano)
$\sigma_b . \sigma_d$	Multiplicação dos desvios padrões entre os sinais (íntegro e dano)
M_4	Métrica quatro
M_5	Métrica cinco
$S_b . S_d$	Produto escalar entre os sinais (íntegro e dano)
$\|\|S_b\|\|$	Norma do sinal íntegro
$\|\|S_d\|\|$	Norma do sinal dano
M_6	Métrica seis
$H_{S_b}^2$	Norma H2 do sinal íntegro
$H_{S_d}^2$	Norma H2 do sinal com dano
M_7	Métrica sete
$H_{S_b}^\infty$	Norma Hinfinito do sinal íntegro
$H_{S_d}^\infty$	Norma Hinfinito do sinal com dano

SUMÁRIO

1 INTRODUÇÃO

Atualmente as empresas têm utilizado a manutenção preditiva como um processo efetivo em suas atividades, pois elas têm como foco a prevenção do surgimento de possíveis falhas em componentes. As falhas podem surgir ou se acumular devido ao uso contínuo do material ou aos esforços dinâmicos a que estes são submetidos. Conforme Marqui (2007) danos estruturais podem mudar de modo significativo às propriedades dinâmicas e / ou geométricas e afetar consideravelmente o desempenho do sistema.

O desenvolvimento tecnológico trouxe inúmeras inovações em sensores, atuadores e controladores contribuindo para um monitoramento inteligente (MARQUI, 2007). Esta análise pode ser feita em diversos níveis como identificação, localização, avaliação e prevenção (RYTTER, 1993). Portanto, quanto melhor a precisão dos processos realizados em cada fase, melhor serão os resultados obtidos.

Desta forma indaga-se: Existem técnicas disponíveis no mercado atual que possam oferecer meios que possibilite encontrar danos com as menores taxas de erros possíveis nos materiais que compõem uma estrutura? Infelizmente, mesmo tendo muitos trabalhos com alto grau de desenvolvimento e experimentação, os métodos ainda são específicos e não podem ser aplicados de maneira generalizada.

As três primeiras etapas da análise anterior são altamente relevantes para o sucesso do monitoramento, por isto, é fundamental usar metodologias que possibilitem criar sistemas que possam detectar o dano, principalmente em seu momento inicial. Este detalhe se torna essencialmente significativo em sistemas críticos, impedindo uma futura catástrofe.

Sistemas que abordam o conceito de aprendizagem têm a capacidade de aprenderem a partir de uma experiência adquirida na execução de suas tarefas e melhorar cada vez mais a execução das atividades. Atualmente, busca-se por sistemas de monitoramento autômatos. Acredita-se que os algoritmos de aprendizagem de máquina sejam uma excelente opção para realizar as etapas de identificação, localização e avaliação de uma falha, obtendo assim, resultados extremamente precisos e com taxas mínimas e às vezes ausentes de erros.

Portanto, o objetivo geral deste trabalho foi utilizar o algoritmo de aprendizagem, através de uma metodologia supervisionada, denominado *support vector machine* (SVM) na identificação de danos em estruturas. Esta técnica ofereceu boa exatidão na detecção de danos incipientes, diminuindo as taxas de erros. Através das classificações efetuadas pelo algoritmo e com as criações de algumas rotinas, foi possível promover a localização dos danos no

material. Por meio das técnicas disponíveis no aprendizado de máquina foi praticável proporcionar um monitoramento inteligente e com resultados satisfatórios. O projeto abordou os seguintes objetivos específicos:

- Empregar os métodos providos das etapas de análise e pré-processamento de dados com o intuito de entender e melhorar a qualidade dos dados obtidos pelos sensores/atuadores piezelétricos adicionados em um material;
- Elaborar um aplicativo que possa facilitar a entrada dos dados de treinamento e classificação. Permitir a definição do tipo do SVM e seus parâmetros. Executar o algoritmo de aprendizado. Elaborar uma matriz confusão e suas métricas. Promover a identificação e a localização de danos na estrutura;
- Avaliar o funcionamento do algoritmo das máquinas de vetores de suporte para que se possam obter resultados mais precisos no processo de classificação;
- Aprimorar o monitoramento em estruturas através de sistemas inteligentes.

O trabalho está modularizado em cinco capítulos, resumidamente apresentados a seguir:

Capítulo 2 – Conceitos usados na metodologia – Esta parte da dissertação mostra o estudo sobre o monitoramento de estruturas inteligentes, conceito de inteligência artificial, técnicas providas do aprendizado de máquina e as teorias de estatística e matemática que sustenta o funcionamento das máquinas de vetores de suporte.

Capítulo 3 – Metodologia – Apresenta a série de procedimentos adotados para a efetivação do projeto. Obtenção de dados oriundos do monitoramento efetuado por uma rede de sensores e atuadores piezelétricos em uma chapa de alumínio. A preparação das informações obtidas através de técnicas providas da análise e pré-processamento de dados, ficando aptos a serem submetidos ao algoritmo. Explanação mais abrangente sobre a biblioteca LibSVM, o uso do algoritmo, a configuração dos parâmetros, a escolha da função *kernel,* o processo de treinamento e classificação permitindo a detecção dos danos. Criação das rotinas usadas para promover a geração da matriz confusão e suas métricas que servem para estimar a confiabilidade do modelo gerado e finalmente, efetuar a etapa de localização dos danos no material.

Capítulo 4 – Resultados e Discussões – Neste ponto é apresentado os resultados obtidos e realizada as discussões pertinentes.

Capítulo 5 – Conclusões e Sugestões– Neste capítulo apresenta-se as conclusões do trabalho e as sugestões para trabalhos futuros.

2 CONCEITOS USADOS NA METODOLOGIA

Neste capítulo são apresentadas algumas explanações referentes aos conceitos de monitoramento da saúde da condição estrutural de estruturas e sobre inteligência artificial. Também engloba uma revisão sobre aprendizado de máquina, juntamente com as diversas técnicas que este campo oferece no estudo e tratamento dos dados. As formas de aprendizado que podem ser aplicadas aos algoritmos, possibilitando a resolução de problemas mais difíceis, porém essenciais para a construção de sistemas autônomos. São mostrados também, exemplos de algoritmos que usam a forma de aprendizado supervisionado. Conjuntamente, a explicação de conceitos mais complexos como a teoria do aprendizado estatístico, o funcionamento do algoritmo *support vector machine* e suas subclassificações.

2.1 MONITORAMENTO DA CONDIÇÃO ESTRUTURAL - SHM

A manutenção preditiva é um tipo de atividade que vêm recebendo uma grande aceitação na rotina das indústrias e grandes empresas. Trata-se de um processo realizado em ciclos nos maquinários, em estruturas como pontes, viadutos, na indústria aeronáutica, entre outros, com o intuito de adquirir informações que transmitem a situação atual daquele sistema.

A análise e a catalogação de algumas características presentes nos elementos são fundamentais para o sucesso de uma manutenção preditiva. Atributos como vibrações dos sistemas (um sistema ou parte dele não se encontra na sua posição de equilíbrio), a propagação de ondas mecânicas, geração de ruídos, oscilações de pressão, variações de temperatura, queda de desempenho entre outros, possibilitam identificar a condição estrutural dos objetos utilizados rotineiramente pelas pessoas. Mudanças nas condições geométricas e dinâmicas como: desgaste, perda de massa, aumento ou diminuição da rigidez, alterações no amortecimento podem mudar de modo significativo o desempenho do sistema.

Com o objetivo de evitar acidentes sérios que podem ocasionar perdas de vidas humanas e grandes perdas financeiras, pesquisadores vêm se dedicando ao estudo do monitoramento em estruturas. A idéia central desse procedimento é fazer um acompanhamento periódico nos equipamentos com o intuito de conhecer a atual situação destes. Por exemplo, se existe a presença de uma determinada falha, o perito através dos estudos e da investigação pode formular hipóteses de qual foi o agente causador deste dano,

qual sua localização e estimar a vida útil do componente, ou seja, por quanto tempo um aparelho pode continuar sendo utilizado e não ocasionar prejuízos ou até desastres.

O monitoramento da integridade estrutural é um tipo de manutenção preditiva que vem sendo amplamente estudada na engenharia. O ponto central deste tipo de manutenção é detectar o momento inicial quando o sistema apresenta variações em suas propriedades geométricas ou dinâmicas. Dependendo da estrutura e da técnica utilizada, estas variações só podem ser detectadas quando o dano já é muito elevado. As pesquisas atuais encaram o SHM como um problema de reconhecimento de padrões. De acordo com Farrar e Worden (2007), quatro etapas são englobadas neste paradigma envolvendo reconhecimento de padrões:

- Investigação operacional e ambiental: consiste no processo investigativo. Serão realizadas análises sobre as mudanças climáticas, ambientais e operacionais que atuam sobre a peça. Estes fatores estariam correlacionados com futuros problemas que podem acometer ao sistema? Qual seria a característica presente em uma falha para esta ser considerada nociva e possuir a capacidade de comprometer o funcionamento do sistema? Estes são exemplos de requisitos abordados nesta etapa.
- Obtenção da informação: etapa relacionada com obtenção e tratamento nos dados obtidos. No processo de obtenção das informações, a maneira de como o sensoriamento será empregada é fundamental. Em relação ao tratamento dos dados, procedimentos como normalização, filtragem, seleção de amostragem são muito indicados. O tempo do monitoramento é um ponto importante também a ser considerado.
- Seleção da característica e sintetização da informação: detecção das informações capazes de mostrar as diferenças entre as propriedades produzidas por um sinal sem danos com um sinal que apresenta falhas. No SHM às vezes é produzido, dependo da periodicidade do monitoramento, uma gama muito grande de dados, dessa forma, é interessante realizar uma redução das informações para facilitar o processo de detecção das características.
- Elaboração de um modelo estatístico para identificação da característica: fase que utiliza a introdução de algoritmos capazes de identificar, localizar e mensurar a situação do dano no material, através

das características dos dados submetidos a eles.

Segundo Rytter (1993), a investigação da situação de um determinado dano presente em uma estrutura pode ser realizada através do levantamento dos seguintes requisitos descritos a seguir:

- Identificação: oferece apenas a indicação da existência ou não do dano na estrutura, portanto esta fase é responsável pelo processo de detecção;
- Localização: provê referências sobre a eventual posição do dano, ou seja, identificar a localização da falha na estrutura;
- Avaliação: fornece um diagnóstico da dimensão do dano, além de identificar e localizar, será realizado uma quantificação do dano;
- Prognóstico: provê conhecimentos sobre a confiabilidade da estrutura e pode realizar uma previsão do restante de vida útil do componente.

Através do avanço tecnológico foi possível adicionar muitos elementos inteligentes nos sistemas possibilitando um monitoramento mais eficiente. Dessa forma, os sistemas promovem de forma mais independente e menos interrupta a etapa de investigação de danos estruturais. Assim, o processo de recuperação do componente somente é executado quando o problema é identificado (MARQUI, 2007).

O monitoramento contínuo consiste na característica fundamental do monitoramento da saúde das estruturas, dessa forma o processo da detecção de danos é uma das atividades mais importantes. A busca desta investigação de maneira contínua, sem interrupções, levou os pesquisadores a integrarem o monitoramento estrutural com a autonomia provida dos sistemas inteligentes, portanto, o objetivo principal é possibilitar um acompanhamento global de forma autônoma.

Todo material é formado por ligações atômicas e as forças dessas associações resultam nas características que este possui, por exemplo, elasticidade, plasticidade, resistência, limite à fadiga entre outras. Porém, com o passar do tempo de usabilidade ou até mesmo, devido aos excessos de esforços, o material tende a apresentar falhas.

A identificação e localização de falhas são processos bastante estudados em monitoramento e uma das técnicas abordadas com sucesso é o procedimento conhecido como impedância elétrica. Este é caracterizado como um procedimento que não ocasiona danos no material. Essa técnica usa sinais que apresentam comprimento de onda curta (TEBALDI; COELHO; LOPES JUNIOR, 2006). Nesta técnica são adicionados sensores e atuadores piezelétricos na estrutura, que podem ter as configurações *pitch-catch* ou *pulse-echo*. Os

atuadores geram ondas que percorrem grandes distâncias na estrutura. Caso exista uma falha no material, ocorrerão mudanças na propagação destas ondas, causando diferenças nos sinais, quando comparados ao sinal da estrutura sem dano (*baseline*). Por fim, estas diferenças entre um material sem dado e um que apresenta dano passam pela etapa de análise de identificação e localização (OLIVEIRA, 2013).

Os materiais piezelétricos (PZT) têm a propriedade de transformar tensão mecânica em tensão elétrica. Essa alteração pode ser feita no sentido direto ou indireto. Quando é direito é chamado de efeito sensor e o contrário, de efeito atuador. Portanto, a estes materiais foram adotados o termo de materiais inteligentes (LEO, 2007). As características encontradas nestes materiais servem para possibilitar o SHM através do procedimento da impedância elétrica. Desta forma, a presença de uma falha muda a estrutura do material e isto causa uma alteração na impedância mecânica. Portanto, o método consiste em acoplar os materiais piezelétricos em uma determinada estrutura e com isto, conseguir informações da impedância elétrica dos materiais piezelétricos. Logo, através da leitura destes dados fornecidos pode-se estimar a presença ou não de falhas na estrutura, visto que existe uma associação entre a impedância elétrica do material (PZT) com a impedância mecânica do material que está sendo investigado (PALOMINO; STEFFEN JUNIOR, 2007).

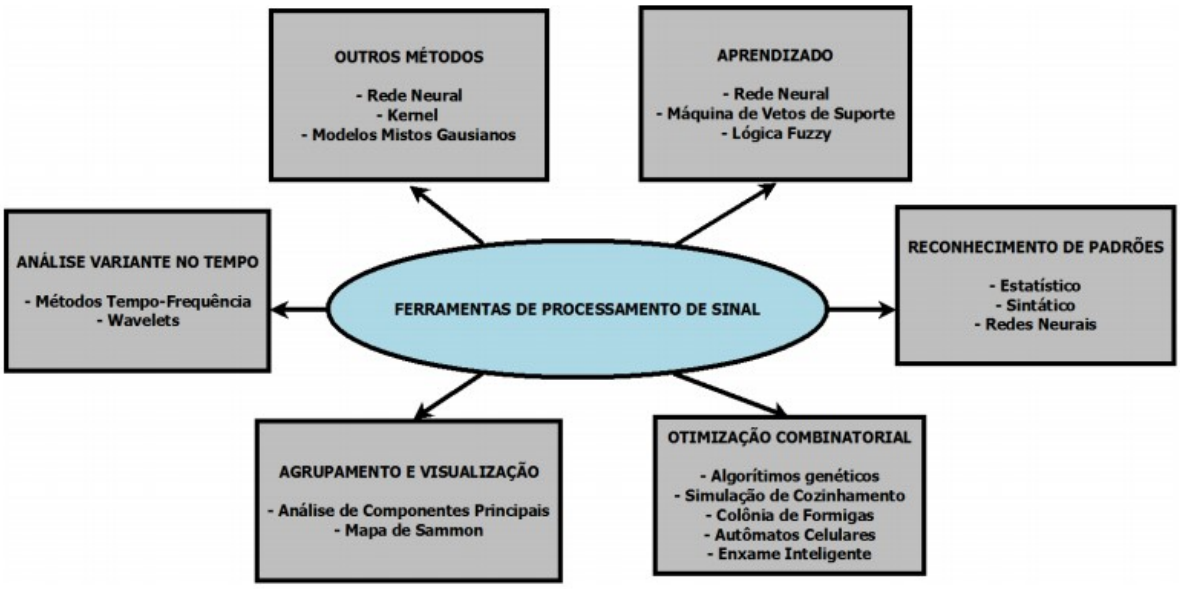
Outra técnica de monitoramento que pode ser utilizada não ocasionando prejuízos no material é a utilização de ondas guiadas, também conhecidas como ondas de Lamb. Uma das vantagens deste procedimento é o fato das ondas percorrerem grandes extensões do material monitorado. As ondas podem ser propagar no material através de dois modos: simétrico (sentido transversal) e antissimétrico (sentido longitudinal) (SU; YE; LU, 2006). Nessa abordagem sensores / atuadores piezelétricos também são adicionados no material. Os dispositivos piezelétricos são responsáveis em enviar e captar os sinais que são propagados. Assim, a constatação de fenômenos ocorridos como reflexão, atenuação e dispersão apresenta indícios de presença de danos no material principalmente em materiais que possuem propriedades bem definidas, cuja velocidade de propagação do sinal é regular em todas as direções (ROSA, 2016).

De acordo com Tebaldi, Coelho e Lopes Junior (2006), o SHM pode ser definido por um conjunto de técnicas que promovem a obtenção, confirmação e investigação das informações obtidas pelos sensores com o intuito de promover um diagnóstico em relação ao tempo que uma determinada peça ainda pode ser utilizada, porém uma das etapas mais difíceis é quantificar as falhas existentes em um material. Algumas técnicas utilizam conceitos de inteligência artificial, como algoritmos que abordam os paradigmas genéticos e de

otimização, para tentar solucionar esta questão.

O desenvolvimento da tecnologia SHM possibilita criar um monitoramento com menos interferência humana e resultados mais precisos no gerenciamento de falhas. A figura 1 mostra diversas técnicas computacionais usadas com frequência na detecção de danos (OLIVEIRA, 2013).

Figura 1- Esquematização de técnicas computacionais envolvidas na etapa de detecção de falhas.

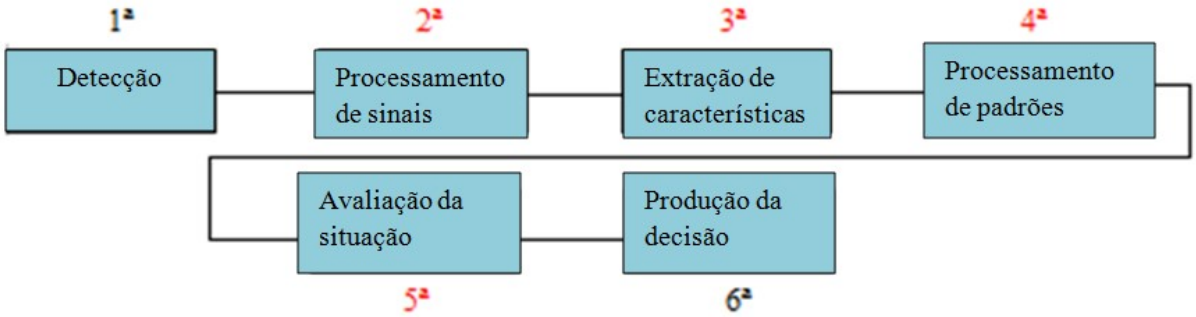


Fonte: Oliveira (2013).

No monitoramento, o diagnóstico pode ser feito comparando um sinal sem defeito com um sinal com falha. Qualquer alteração no sinal recebido pode ser interpretada com uma possível variação estrutural, ou alteração das condições operacionais ou ambientais. Cada sensor produz uma gama de dados muito grande, portanto, é necessário usar técnicas de análise e pré-processamento de dados para que se obtenha um resultado mais preciso da análise. Estes procedimentos podem ser realizados através da aprendizagem de máquina, uma subárea da inteligência artificial que promove diversos procedimentos que podem auxiliar nas fases de entendimento e melhoramento dos dados. Este campo de estudos também oferece algoritmos que possuem a capacidade de aprimorar o desempenho de suas tarefas através das experiências passadas. Esses algoritmos podem ser empregados no processo de descobrir a existência ou a ausência de falhas no material, na determinação do posicionamento (local do dano no material) e definir a extensão do tamanho das falhas nas estruturas.

As técnicas providas pela aprendizagem de máquina fornecem soluções para os problemas relacionados com a identificação, localização e avaliação de danos. O ponto essencial é organizar as informações para que os processos de compactação dos dados e extração do conhecimento sejam bem efetivos resultando em uma boa atuação dos algoritmos de aprendizagem (classificação ou regressão). Devido à flexibilidade do modelo de organização das informações, este pode ser usado dentro de um contexto específico. A figura 2 mostra as etapas abordadas no processo de organização das informações que podem ser introduzidas dentro do contexto de aprendizado de máquina (WORDEN; MANSON, 2007).

Figura 2 -Etapas empregadas no conceito de organização dos dados.



Fonte: Bedworth e O'Brien (2000). (Adaptado pela autora)

Dessa forma, como o modelo acima é adaptativo este poderia ser combinado para mostrar os passos envolvidos no monitoramento de estruturas usando uma abordagem de reconhecimento de padrões. Na etapa1 (um) e 2 (dois) compreenderiam técnicas usadas para obter e extrair informações da situação atual do material. Portanto, os sensores podem ser usados sendo responsáveis pela obtenção dos dados, que podem ser de variações de temperatura, oscilações de pressão, vibrações excessivas, ondas acústicas, etc. As 3 (três) e 4 (quatro) poderiam ser refinadas pelas técnicas disponíveis no aprendizado de máquina como a análise e pré-processamento de dados.

A análise de dados é realizada com a finalidade de compreender os dados, extrair informações e características. Desta forma, é possível encontrar padrões e tendência nos elementos. As técnicas usadas para adquirir a extração das informações consistem nos métodos estatísticos e nas observações. Esse processo de análise contribui para a obtenção de muitas informações importantes, auxiliando na seleção da técnica mais apropriada de pré-processamento dos dados e de aprendizado.

Retomando os passos contidos na figura 2, a etapa 5 (cinco) trata da solução do

monitoramento que inclui um algoritmo para a classificação dos dados. Este pode usar uma aprendizagem supervisionada, quando o conjunto de treinamento é formado por elementos de entrada e saída que são passados ao sistema para que este possa construir o classificador. O sistema também pode utilizar uma abordagem não supervisionada, quando o conjunto de treinamento é somente formado pela entrada dos dados, ou seja, não há rotulação neles.

Em algumas abordagens de monitoramento da integridade estrutural são utilizadas as duas metodologias fornecidas pela aprendizagem de máquina: supervisionada e não-supervisionada. Desta forma, ocorre uma combinação de técnicas para promover de forma eficiente o monitoramento inteligente. A adoção destas práticas foi empregada em um projeto desenvolvido por Nick et al. (2015), que utilizaram rotinas baseadas em uma abordagem não supervisionada com o intuito de promover as duas etapas iniciais do monitoramento. Adotaram algoritmos orientados a dados cuja finalidade era realizar a detecção e o posicionamento das falhas na estrutura; algoritmos com a capacidade de aprender através de elementos já rotulados visandoreconhecer a seriedade e as característicasdos danos no material e a análise de componentes principais (PCA) para a obtenção de informação qualitativa visando a dimuição dos atribuitos dos dados. O trabalho também mostra que o uso do PCA nos dados pode influenciar o processo de classificação e de predição. Em alguns algoritmos, o cálculo das métricas usando PCA melhora a rapidez de classificação e em outros casos, a presença da técnica nos dados leva a uma queda de precisão na classificação.

A etapa 6 (seis) da figura 2 inclui a tomada de decisão do monitoramento, ou seja, se existe um dano ou não na estrutura. O tamanho e a localização também podem ser determinados nesta fase, dependendo do algoritmo a ser utilizado. A última etapa do monitoramento relacionada com a estimativa de duração do material, ou seja, a confiabilidade de uso do componente, não pode ser realizada apenas com as metodologias oferecidas pelo aprendizado de máquina. Dessa forma, sua realização necessita de outros métodos, pois engloba outros aspectos físicos. É necessário envolver outros paradigmas na solução dessa etapa (WORDEN; MANSON, 2007).

Um dos campos mais estudados pelo monitoramento de estruturas é a detecção de características de dados que permite diferenciarquando a estrutura apresenta ou não um dano. Depois do processo de extração das características, estes dados são submetidos aos algoritmos de aprendizagem que deve ter a capacidade de realizar a avaliação de que existe ou não falhas na estrutura. Porém, um problema a ser considerado são as presenças de falsos positivos e falsos negativos que podem aparecer devido a alguma variação nas condições operacionais e ambientais. Talvez este seja uns dos dilemas que assolam o monitoramento de estruturas. Em

um trabalho desenvolvido por Figueiredo et al. (2011), é apresentado a utilização de quatro algoritmos de aprendizagem de máquina através de uma abordagem não superviosionadapara identificar a presença de danos ocasionados ou influenciados por mudanças operacionais e/ou ambientais. Essas oscilações ambientais e operacionais podem ocultar danos reais ou o contrário, apresentar falhas sem que estas realmente existam no material.

Atualmente existemuitos algoritmos que utilizam a abordagem supervisionada no processo de aprendizagem como: redes neurais artificiais, k-vizinhos mais próximos, máquinas de vetores de suporte entre outros.

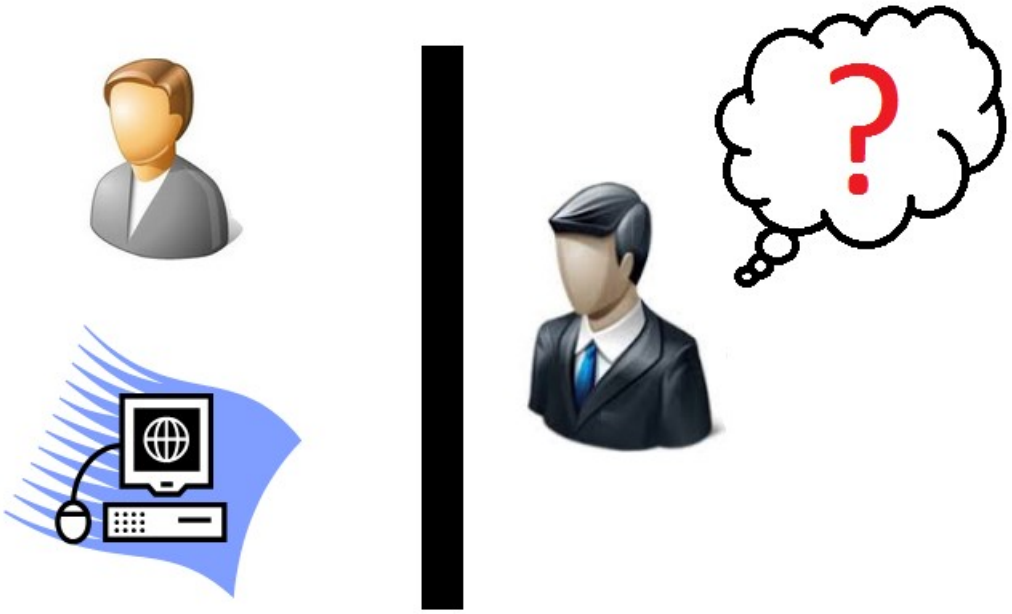
2.2 INTELIGÊNCIA ARTIFICIAL: UM BREVE HISTÓRICO

A inteligência artificial pode ser vista como método ou procedimento realizado por uma máquina numa tomada de decisão, que reflete as características de hábitos ou atitudes desempenhadas por um agente portador de inteligência. Esta envolve diversos conceitos como: otimização, reconhecimento de padrões, automatização, robótica entre outros. Geralmente esta ciência possui como finalidade criar sistemas com a capacidade de reproduzir um determinado tipo de comportamento, podendo este ser, humano, animal ou outros. Devido a sua vasta abrangência, fica difícil às vezes com poucas palavras defini-la, dessa forma, alguns pesquisadores se referem a esta como:

- Trata-se de uma ciência que investiga a possibilidade dos computadores executarem ações, que atualmente os indivíduos as fazem, através de um conjunto de funcionalidade mais ampla resultando em um desempenho superior (RICH, 1988).
- IA pode ser explicada como “sistemas que pensam e atuam racionalmente e sistemas que pensam e atuam como seres humanos” (RUSSEL; NORVIG, 2004, p. 5).

O considerado “pai da computação”, o matemático britânico Alan Turing, apresentou uma sugestão de teste modelado para expor um conceito básico de inteligência artificial. Este exame seria composto por um interrogador humano e mais dois participantes, um humano e uma máquina, que responderiam perguntas aleatórias. Portanto, caso o interrogador não conseguisse distinguir o autor da resposta, a máquina seria interpretada como inteligente. Este teste ficou imortalizado como o Teste de Turing. A figura 3 mostra uma representação de como seria o Teste de Turing.

Figura 3 - Esquematização do Teste de Alan Turing.



Fonte: própria autora.

O propósito fundamental desta ciência é o agente realizar suas interações com o meio e que estas possam passar imperceptíveis aos olhos de outros observadores. Desta forma, esta grande área de conhecimento pode ser considerada como uma ciência que procura representar o entendimento e o comportamento humano aos seus moldes programados, com a finalidade de possibilitar a criação de sistemas que apresentem características humanas.

A origem da IA começa em 1943 quando dois pesquisadores Warren McCulloch e Walter Pitts desenvolveram um projeto na qual um sistema de computador era interpretado como um cérebro humano. Os dois cientistas desenvolveram um projeto sobre neurônios artificiais, cada neurônio pode possuir dois estados: ligado e desligado, que muda seu *status* de acordo com os estímulos recebidos por outros neurônios. Este projeto se tornou o precursor das tradições lógicas e conexionistas de IA (RUSSEL; NORVIG, 2004).

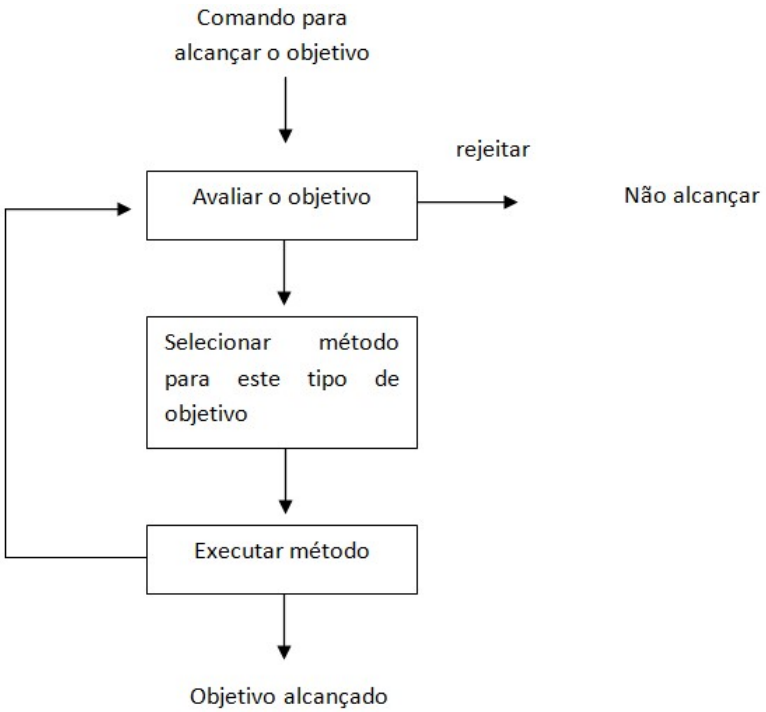
Pesquisadores fascinados pelos temas acerca de neurônios artificiais, conceito de autômatos e no aprendizado da inteligência se encontraram no seminário em Dartmouth College nos Estados Unidos em 1956. Nele foi apresentado um programa de computador denominado Logic Theorist (LT) que possuía a habilidade de resolver problemas de maneira não numérica. Foi neste encontro de “grandes mestres” que surgiu o nome inteligência artificial. Depois da realização deste *workshop*, a comunidade envolvida com IA na época ficou entusiasmada com os resultados obtidos. Assim, as primeiras fases da pesquisa foram bastante benéficas, pois as pessoas viam o computador como uma “calculadora gigante” cuja

finalidade inicial era apenas de realizar cálculos matemáticos, mas agora estes apresentavam habilidades de resolver problemas de forma não numérica. Alguns já vislumbravam a possibilidade de criar sistemas para solucionar problemas mais complexos (RUSSEL; NORVIG, 2004).

Seguindo este cenário de inovações, os cientistas Newell e Simon criaram um programa capaz de resolver problemas gerais utilizando maneiras humanas para tal. Este programa foi chamado de General Problem Solver (GPS) e foi considerado o primeiro *software* a adotar o conceito de raciocínio. Depois deste, foram aparecendo ao longo dos anos, uma série de programas que apresentavam características de raciocínio (RUSSEL; NORVIG, 2004).

A figura 4 mostra o fluxograma geral do funcionamento do GPS, desde o comando para atingir um objetivo até a execução do método para cumprir a meta. O programa feito na época para tentar resolver questões cotidianas usando protocolos humanos.

Figura 4 - Fluxograma Geral do GPS.



Fonte: Newell, Shaw e Simon (1960). (Adaptado pela autora)

Após vários resultados satisfatórios, o campo de pesquisas em IA começaram a apresentar alguns impasses. Os principais problemas eram o entendimento teórico limitado, a falta de sucesso na resolução de problemas, a insuficiência de *hardware* mais poderoso e a

complexidade de elaborar algoritmos para representar uma conduta inteligente (RUSSEL; NORVIG, 2004).

Com o melhoramento dos componentes de *hardware* e dos algoritmos,a IA começa a apresentar soluções mais práticas na resolução de dilemas. Começam a ser criados sistemas que possuem a habilidade de resolver problemas exprimindo o comportamento de um perito em uma determinada área. Estes sistemas ficaram conhecidos como sistemas especialistas. Segundo Mendes (1997), os elementos que formam os sistemas especialistas podem ser categorizados em:

- Interface: parte do sistema que permite a interação com o usuário. Possibilita que a pessoa introduza as perguntas no sistema;
- Motor de inferência: consiste na parte central, que realiza amanipulação das informações armazenadas nas estruturas que representam o conhecimento. O sistema vai utilizar as informações de entrada com o seu conhecimento base e este tem que possuir a capacidade de gerar conclusões;
- Base de conhecimento: o perito passa a sua experiência em determinada área usando uma representação do conhecimento. Essa estrutura é formada por regras de lógicas, associações e teorias.

Os sistemas especialistas podem ser usados na solução de problemas que necessitam de conhecimento peculiar em determinado assunto, sendo empregado em diversas áreas: no planejamento de construções civis (FORMOSO, 1993), auxílio na tomada de decisões como selecionar o espaço mais favorável para a implantação de uma empresa (CARNASCIALI; DELAZARI, 2011), diagnóstico de determinadas doenças (SOUZA; TALON, 2013) entre outras aplicabilidades.

À medida que os problemas se tornaram mais complexos, houve a necessidade de elaborar programas computacionais que tivessem menos interferência humana na execução de suas tarefas. Então, começam a surgir sistemas que possuem a habilidade de gerar modelos de forma independente se baseando nos seus conhecimentos pretéritos. Esta capacidade de se adequar a novos ambientes e superá-los é chamado de aprendizado de máquina, que apresenta diversas técnicas para que a máquina possa atingir tal objetivo (FACELI et al., 2011).

2.3 CONCEITOS ABORDADOS NO APRENDIZADO DE MÁQUINA

“O aprendizado permanece sendo uma área desafiadora para a IA. A importância do aprendizado, entretanto, é inquestionável, particularmente porque essa habilidade é um dos componentes mais importantes do comportamento inteligente” (LUGER, 2013, p.23).

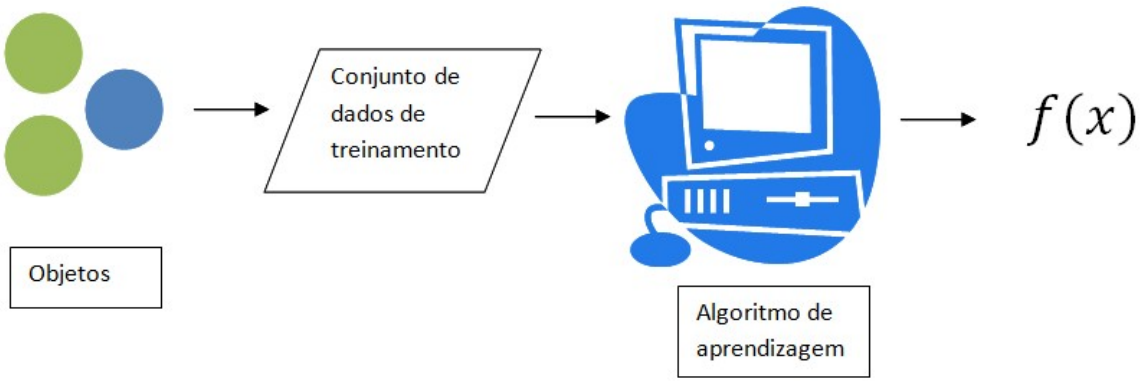
A aprendizagem de máquina (AM) trata-se um conceito que objetiva a criação de algoritmos que permitam que um sistema aprenda por meio das informações a ele disponibilizadas. A meta deste conceito é criar uma função matemática, através de um conjunto de elementos de treinamento, que possa classificar ou descrever novos dados que forem submetidos ao algoritmo.

O sistema recebe um conjunto de dados que representa objetos do mundo real. Dessa forma, através de um aprendizado indutivo, um algoritmo de AM tem que gerar modelo específico. Apesar da hipótese gerada ter sido obtida por um determinado conjunto de dados inseridos externamente, esta deve possuir a capacidade de resolver o problema quando aplicado em dados novos. Dessa forma, apresentando o conceito de generalização (FACELI et al., 2011).

Michell (1997) caracterizou AM como a habilidade de um sistema aumentar seu funcionamento na efetivação de uma atividade através do conhecimento adquirido em tentativas anteriores. Desta forma, um algoritmo que possui a capacidade de aprender realizaria na primeira tentativa uma determinada tarefa com desempenho X e, portanto, se fosse solicitado a este novamente a execução da mesma tarefa, o sistema cumpriria a função, porém apresentando taxas de desempenho melhor do que a anterior.

A figura 5 demonstra o esquema usado no processo de aprendizagem de máquina. A partir de um conjunto de dados, atua-se um indutor, um algoritmo, que gera um classificador que possui a capacidade de classificar novos valores.

Figura 5 - Etapas no processo de aprendizado de máquina.



Fonte: própria autora.

Simon (1983), citado por Scott (1983, p. 359) expõe sua ideia de aprendizado como: “qualquer mudança em um sistema que melhore o seu desempenho na segunda vez que ele repetir a mesma tarefa ou outra tarefa tirada da mesma população”.

Existem alguns métodos para se trabalhar com o problema de aprendizagem de máquina, são eles: o tratamento simbólico, que se utiliza da ideia de que a atuação essencial em sua forma de agir está baseada na representação nítida do seu conhecimento; a visão conexionista, o sistema é visto como um cérebro humano, ou seja, existem neurônios artificiais cujo aprendizado se dará de forma tácida através da comunicação destes; as concepções genéticas, cujo conhecimento progride como uma forma de se adequar ou se harmonizar ao novo ambiente que se modifica com o passar do tempo e o conceito dinâmico, que utiliza métodos estatísticos com a finalidade de fazer previsões (LUGER, 2013).

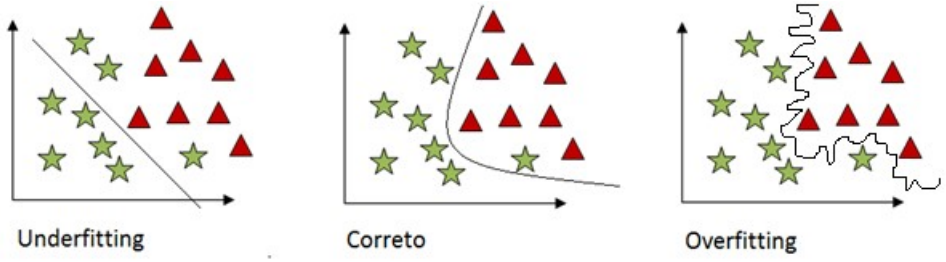
Frequentemente os dados apresentam características heterogêneas (tamanho, formato, escala e tipagem). Portanto, é necessário adotar procedimentos que deixem os dados mais aptos para poderem ser utilizados pelos algoritmos de aprendizagem. Atualmente são encontrados diversos procedimentos de análise e pré-processamento de dados que ajudam na eliminação ou minimização de dados redundantes, desbalanceados, valores inexistentes, ruidosos e incompletos. Estas práticas são benéficas porque como o algoritmo aprende com os dados, esse processo de tratamento no conjunto de dados influencia na performance do algoritmo (FACELI et al., 2011).

As etapas de análise e pré-processamento de dados são de extrema importância, pois contribuem para a descoberta de *outliers*, dados que contêm valores que não seguem os padrões. Estas práticas também favorecem o não aparecimento de problemas de classificação

conhecidos como *overfitting* e *underfitting*.

Na ocorrência de *overfitting*, o classificador gerado pelo treinamento simplesmente memoriza os padrões. Desta forma, a hipótese gerada é muito particular aos dados de treinamento, portanto, quando o classificador é apresentado a novos dados, este demonstra péssima taxa de classificação. No problema do *underfitting* o classificador fica muito genérico e o modelo inferido não consegue classificar os dados adequadamente, ou seja, apresenta baixa taxa de exatidão, tanto na fase de treinamento quanto na etapa de testes. Geralmente este processo ocorre quando são usados elementos mais quantitativos do que qualitativos na etapa de treinamento. Dessa forma, os dados não demonstram valores relevantes (MONARD; BARANAUSKAS, 2003). Outro fator que pode ocasionar a ocorrência desses fenômenos é o uso inadequado de parâmetros das funções que promovem a definição da fronteira de decisão. A utilização de valores exacerbados nos parâmetros dessas funções geralmente promove o *overfitting*. A situação contrária, ou seja, valores subestimados tendem a ocasionar o aparecimento de *underfitting*. A diferença entre estes dois fenômenos pode ser visualizada com mais detalhes na figura 6. Ambos representam falhas na classificação, porém com suas particularidades.

Figura 6 - Problemas de classificação que podem aparecer no treinamento: *overfitting* e *underfitting*.



Fonte: própria autora.

Na área de aprendizado de máquina são empregados diversos paradigmas na resolução de problemas complexos, além destas técnicas utilizadas, o processo de aprendizagem em um sistema pode ser particionado em três grandes grupos: abordagem supervisionada, quando existe um instrutor que ensina o caminho de aprendizado ao algoritmo; não supervisionada, neste caso, o sistema aprende sozinho e por fim, o aprendizado por reforço, cuja máquina aprende por tentativas e erros.

Muitos problemas complexos podem ser resolvidos através dos algoritmos de AM, devido a sua peculiaridade de melhorar seu desempenho através das experiências passadas. Atualmente estes sistemas são empregados em diversas áreas como: na análise sintática e processamento automático da língua portuguesa (ALENCAR, 2011), identificação de elementos através do processamento de imagens (ALMEIDA, 2014), avaliação da radiação solar ultravioleta em determinadas regiões (ALMEIDA, 2013) entre outras aplicabilidades.

Porém, antes de adentrar no aprofundamento dos conceitos brevemente relatados acima, serão explanadas mais detalhadamente as técnicas de análise de dados e pré-processamento de dados, pois estas duas etapas são de extrema relevância para o processo de aprendizado. Estas duas fases se tornam fundamentais porque o algoritmo utiliza um aprendizado indutivo, ou seja, seu processo de conhecimento é adquirido através de um conjunto de dados que são coletados. Portanto, se a etapa inicial não for bem refinada e tratada a fase de gerar o classificador pode ser totalmente comprometida levando a resultados indesejáveis como, por exemplo, péssima taxa de exatidão.

2.3.1 Análise de dados

Esta técnica tem como finalidade entender com mais detalhes os conjuntos de dados que servem como entrada para o algoritmo de aprendizado. A qualidade da aprendizagem, ou seja, a condição do resultado obtido no processo de indução depende diretamente dos dados que são disponibilizados à máquina de aprendizado. Portanto, o objetivo principal da análise é encontrar padrões e interpretá-los. Geralmente são usados métodos que possibilitam a exploração dos dados, mostram como estes estão distribuídos e formados, além de proporcionar o estudo da melhor forma de interpelar o enigma utilizando um algoritmo mais apropriado para cada cenário.

De acordo com Faceli et al. (2011), um conjunto de dados pode ser representado por diversos vetores formados por n objetos, portanto, trata-se de vetores multidimensionais. Cada linha deste vetor simboliza uma instância que representa uma entidade real, por exemplo, uma planta. Cada elemento que compõe este vetor, mapeado pela junção da linha i pela coluna j representa os atributos, ou seja, as características de cada objeto.

A tabela 1 mostra detalhes de um simples conjunto de dados, o *dataset Iris*, bastante conhecido na comunidade de aprendizado de máquina. Este conjunto encontra-se disponível no seguinte endereço http://archive.ics.uci.edu/ml/datasets/Iris. O conjunto original é formado

por cento e cinqüenta objetos, que neste caso, representam cento e cinquenta plantas, e por cinco atributos, sendo que quatro simbolizam os tamanhos e comprimentos de pétalas e sépalas e o quinto atributo é a classe a qual pertence cada planta. Esse conjunto de dados foi citado apenas para demonstrar como é composto um *dataset* típico.Neste exemplo são mostradas apenas algumas partes do conjunto.

Tabela 1 - Representação parcial do *dataset Iris*.

Classe	Comprimento da Sépala (cm)	Largura da Sépala (cm)	Comprimento da Pétala (cm)	Largura da Pétala (cm)
Iris-setosa	4,6	3,2	1,4	0,2
Iris-setosa	5,3	3,7	1,5	0,2
Iris-setosa	5,0	3,3	1,4	0,2
Iris-versicolor	7,0	3,2	4,7	1,4
Iris-versicolor	6,4	3,2	4,5	1,5
Iris-versicolor	6,9	3,1	4,9	1,5
Iris-virginica	7,1	3,0	5,9	2.1
Iris-virginica	6,3	2,9	5,6	1,8
Iris-virginica	6,5	3,0	5,8	2,2

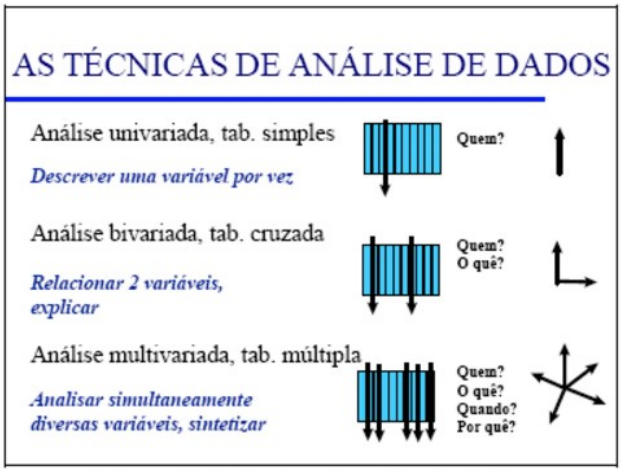
Fonte: University of California (2016). (Adaptado pela autora)

Dependendo da abordagem utilizada, algumas coleções de exemplos de preparação podem apresentar um atributo especial. É dito especial porque ele vai instruir o algoritmo no processo de aprendizagem. Esse atributo pode apresentar em seu conteúdo, valores inteiros ou reais. O atributo de interesse pode ser usado no dilema da categorização ou regressão dos dados. Neste caso, quando existe a presença deste atributo, a meta do algoritmo de AM é gerar uma função matemática que faça o mapeamento dos valores de entrada com o valor de saída.

Cada elemento no conjunto de dados que representa a abstração de um objeto no mundo real é definido por atributos. Cada atributo representa uma dimensão no espaço. Os atributos também rotulados como características são formados por tipos e escalas. O tipo está relacionado com os valores que o atributo pode possuir, dessa forma, podem ser classificados em qualitativo (categoria) ou quantitativo (discreto ou contínuo). Por exemplo: definir o nome de uma pessoa é um tipo de dado qualitativo; determinar o salário de uma pessoa seria um tipo quantitativo contínuo. Outro ponto referente aos atributos é a escala. Esse termo está relacionado com as manipulações que podem ser aplicadas aos valores dos atributos. Por exemplo, no atributo salário pode ser aplicada uma operação de soma. A escala pode ser classificada em nominal, ordinal, intervalar e racional. Cada escala abrange os procedimentos corretos que podem ser efetuados com os valores dos atributos (FACELI et al., 2011).

Outra forma muito usada para entender os dados, procurar padrões e tendências é produzir medidas com a finalidade de obter informações quantitativas em relação aos dados. Para usar essa abordagem, os dados devem ser fragmentados em dois grupos. O primeiro chamado de dados univariados, na qual uma determinada instância possui somente um atributo. O segundo é rotulado como dados multivariados são aqueles que possuem dois ou mais atributos de entrada. Nesses casos, geralmente são mostrados a relação linear existente entre dois atributos. O conjunto de dados *Iris*, por exemplo, trata-se de dados multivariados (FACELI et al., 2011). A figura 7 mostra com mais detalhes como seria os diferentes tipos de análise dados: univariada, bivariada e multivariada.

Figura 7- As diferentes técnicas de análise de dados.



Fonte: Freitas e Moscarola (2002).

Na análise univariada, cada atributo é analisado um após o outro, também conhecida como um processo de tabulação simples. Na análise bivariada, procura-se encontrar as relações existentes entre dois atributos, geralmente é rotulada como tabulação cruzada. Na análise multivariada compreende-se em estudar universalmente um conjunto de atributos com a finalidade de produzir uma síntese dos dados (FREITAS; MOSCAROLA, 2002).

Antes de usar os algoritmos de aprendizado é primordial que o conjunto de dados seja submetido aos procedimentos envolvidos na etapa de pré-processamento de dados. Existem diversas técnicas que podem ser empregadas aos dados possibilitando um melhoramento na qualidade destes.

2.3.2 Pré-processamento de dados

O objetivo principal do pré-processamento de dados é realizar um tratamento no conjunto de dados. Como os dados geralmente são obtidos através de dispositivos físicos, no processo de transmissão podem ocorrer algumas falhas ocasionando dados corrompidos. Dessa forma, o conceito é realizar a eliminação de dados incompletos, duplicados, ruidosos entre outros problemas que os dados podem demonstrar. Esse processo almeja melhorar a situação dos dados deixando-os aptos para os algoritmos de AM.

As técnicas abordadas na etapa de pré-processamento de dados podem diminuir o custo de recursos computacionais no processo de treinamento e classificação do algoritmo favorecendo o desempenho e eficácia deste. Segundo Zhang, Yand e Liu (2005) citado por Chino (2014) em mineração de dados, essa fase é vista como uma das mais essenciais, pois na técnica de mineração a forma de aprendizado geralmente abrange abordagens não supervisionadas. Portanto, a fase de tratamento nos dados pode representar grande parte de todo o projeto.

De acordo com (FACELI et al*.,* 2011), as técnicas utilizadas possuem a intenção de preparar o conjunto de dados para serem posteriormente usados no algoritmo. A tabela 2 mostra os conceitos que fundamentam cada uma.

Tabela 2 – Técnicas abordadas no pré-processamento de dados (continua).

Técnica	Finalidade
Eliminação manual de atributos	Consiste em retirar atributos que não são importantes para o atributo alvo. Por exemplo: o nome do pai de uma criança é insignificante para saber se este gosta ou não de sorvete.
Integração de dados	Quando os atributos estão disseminados em diversos conjuntos de dados, é imprescindível fazer a integração dos dados. Na técnica de *data warehouse,* os dados são concentrados em um único local.
Amostragem de dados	Geralmente não é compensatório trabalhar com volume muito grande de dados, é preciso retirar uma amostra de dados para favorecer o desempenho dos algoritmos, pois estes tendem a apresentar dificuldades na execução devido ao crescimento do custo computacional.
Dados desbalanceados	O ideal é que o conjunto de dados seja composto por classes que apresentam quantidades de objetos igualitários, ou aproximados. *Datasets* que apresentam classes majoritárias ou minoritárias tendem a prejudicar o processo de treinamento. Por exemplo, o conjunto de dados *Iris* apresenta a mesma quantidade de objetos para as três classes.
Limpeza de dados	Visa tratar os dados que se apresentam, por exemplo, não completos (um atributo que está sem valor); não consistentes (atributos que contrastam as informações); redundantes (elementos apresentam características com mesmos valores); e dados ruidosos (atributos que apresentam falhas ou seus conteúdos estão acima ou inferior ao padrão).

Tabela 3 – Técnicas abordadas no pré-processamento de dados (conclusão).

Técnica	Finalidade
Transformação de Dados	Alguns algoritmos de aprendizado só funcionam com determinado tipo de dado, desta forma é necessário fazer uma conversão.
Redução de Dimensionalidade	Quando os dados possuem grande dimensionalidade, os algoritmos tendem a apresentar péssimo desempenho, desta forma o ideal é usar procedimentos para fazer uma moderação. A diminuição de características aumenta a performance do classificador, diminui o gasto de recursos computacionais e apresenta conclusão mais satisfatória.

Fonte: Faceli et al. (2011). (Adaptado pela autora)

A redução da dimensionalidade pode ser realizada por técnicas de seleção de atributos e de agregação. Dessa forma, o conceito é retirar apenas as características significativas, ou seja, aquelas que realmente influenciam para a geração do atributo alvo. As demais não fazem sentido ser submetidas ao algoritmo, pois apenas tendem ao consumo maior de recursos computacionais, atrapalhando no desempenho dos sistemas de AM.

Depois das etapas de análise e pré-processamento de dados, o conjunto de dados esta apto a ser usado no processo de treinamento através do procedimento de aprendizagem. Existem alguns paradigmas adotados e formas abordadas no processo de aprendizado de máquina que serão explanados nos tópicos seguintes.

2.3.3 Tipos de paradigmas e de aprendizagem

De acordo com o dicionário de língua portuguesa (2016), a palavra paradigma pode ser definida como “algo que serve de exemplo geral ou de modelo; conjunto das formas que servem de modelo de derivação ou de flexão”.

Na programação, paradigmas se referem à forma de pensar do programador, como este procura respostas para um determinado problema. Existem diversos paradigmas de programação: estruturada, orientada a objetos, concorrentes, imperativa, declarativa e outras.

Segundo Monard e Baranauskas (2003), o AM também engloba alguns paradigmas com o intuito de resolver determinados problemas encontrados, são eles: representação simbólica (proveniente da lógica convencional), modelo conexionista (baseada nos modelos precursores de McCulloch e Pitts), arquétipo de memorização (treinamentos mais simples, porém demorados na classificação), modelo estatístico (busca minimizar ou maximizar a função que resolve o problema) e abordagem genética (baseada na Teoria da Evolução). A tabela 3 explana uma síntese de cada paradigma.

Tabela 4 - Alguns paradigmas empregados no aprendizado de máquina.

Tipo de Paradigma	Definição
Representação simbólica	Procurar-se adquirir conhecimento formando modelagens simbólicas através de observações de determinadas situações e o oposto delas. Elas são descritas na maneira de sentenças lógicas como, por exemplo: árvores de decisão.
Arquétipo de memorização	O programa precisa guardar os moldes na memória para poder classificar os recentes, distinguindo-se dos algoritmos gulosos que usam uma abordagem distinta. Devido a esta característica de armazenar sempre os exemplos, estes programas são chamados de preguiçosos. Geralmente essas rotinas tendem a apresentar um treinamento de fácil assimilação e implementação.
Modelo conexionista	O programa é visto como um cérebro, ou seja, como um emaranhado de neurônios conectados entre si. Abordagem bastante popular na área de aprendizado. Um dos maiores exemplos desse cenário são as redes neurais artificiais, que são apenas funções matemáticas baseadas na fisiologia de uma célula nervosa humana. São muito utilizadas em reconhecimento de padrões e de voz, classificações de dados, robótica, entre outros.
Abordagem genética	O modelo é formado por conjunto de classificadores que concorrem entre si para realizar as futuras classificações. Nesse conjunto de elementos, os que apresentam um mau comportamento são rejeitados, mas aqueles que apresentaram boas performances, sobrevivem e fabricam mutações provenientes deles próprios. Essas mutações consistem em uma combinação entre os classificadores.
Modelo estatístico	É usado técnicas estatísticas para buscar um bom classificador, geralmente usam parâmetros com o intuito de achar modelos mais otimizados. Um bom exemplo desta categoria são as redes bayesianas.

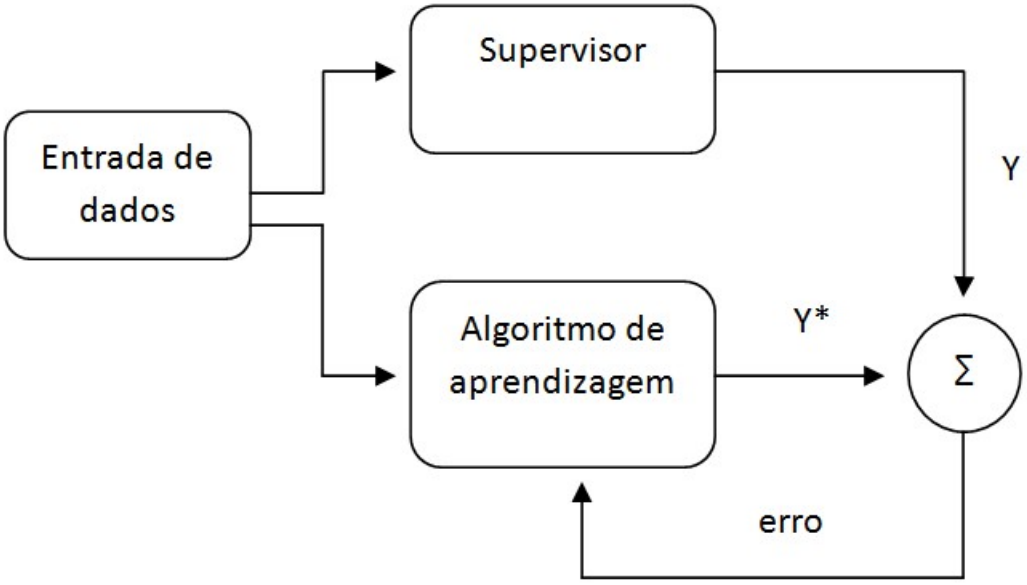
Fonte: Monard e Baranauskas (2003). (Adaptado pela autora)

Além dos tipos de paradigmas, o AM engloba também maneiras de aprendizado, ou seja, métodos na forma de possibilitar um algoritmo à capacidade de aprender. Dentro deste contexto, pode-se dividir, basicamente, em três grupos: aprendizado supervisionado, não-supervisionado e aprendizado por reforço. Segundo Coppin (2010) existem também outras subdivisões como, por exemplo, o aprendizado por hábito na qual o sistema categoriza somente as informações conhecidas e não tenta diminuir o erro da função gerada. De acordo com Luger (2013) o aprendizado por indução tem a capacidade de produz conhecimento por meio de observações realizadas por um conjunto de exemplos.

Na abordagem supervisionada a máquina de aprendizado recebe um conjunto de dados de treinamento que possuem dados de entrada (características) e saída (rótulos de cada classe). Este atributo alvo expõe o fator de interesse. Usando como referência o contexto de aprendizagem estatístico, existe um produtor de vetores aleatórios que são escolhidos através de uma função de distribuição de probabilidade não conhecida e precisa. Assim, o instrutor recebe esses vetores e produz uma saída para cada um. Dessa forma, o algoritmo deve receber os elementos contidos no conjunto de treinamento e gerar um valor que se aproxime o máximo possível do valor emitido pelo professor, assim ele acha a função matemática mais

coerente para o problema. A figura 8 mostra o processo de aprendizado supervisionado, observe que ocorre a comparação do *y* (supervisor) com o *y** produzido pelo algoritmo de AM (aprendiz) (SANTOS, 2002).

Figura 8 - Funcionamento do processo de aprendizado supervisionado.



Fonte: Haykin (2001). (Adaptado pela autora)

Depois de encontrada a solução mais exímia para resolver o problema proposto, o algoritmo deverá ser capaz de receber novos dados compostos apenas de entrada e realizar a classificação destes. Dessa forma, o classificador genérico que foi obtido através de dados rotulados, manipulará dados que não apresentam classificações, portanto esse processo de identificação torna-se a sua função.

Na abordagem de aprendizagem não-supervisionada a máquina recebe um conjunto de dados que é constituído apenas pelos valores de entrada e sem a respectiva saída, ou seja, não existe um instrutor, não há um atributo alvo que possa auxiliar o indutor no processo de aprendizagem. Segundo Domingues (2003), o algoritmo sistematiza de forma desassistida a que classe o objeto pertence através de métodos que analisam os pontos em comuns entre os objetos. Trata-se de um aprendizado orientado a dados, cujo algoritmo não conhece antecipadamente as classes que definem os elementos empregados na fase de treino (FACELI, et al., 2011). A metodologia não-supervisionada é muito usada na mineração de dados como o intuito de encontrar padrões, associações nos dados armazenados em um repositório de dados

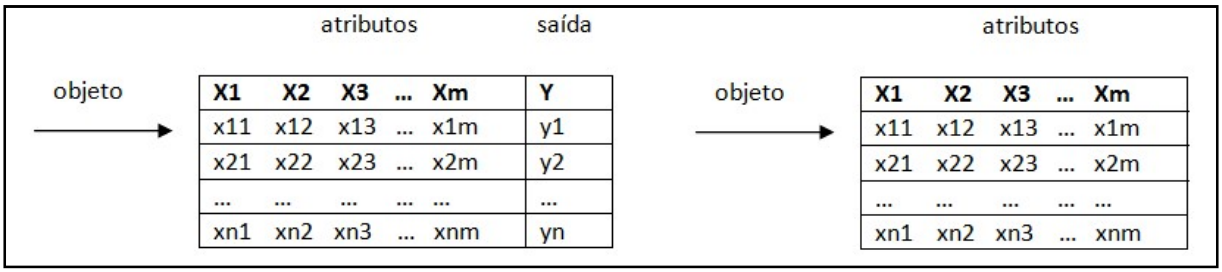
para melhorar a gestão das informações. Rezende, Marcacini e Moura (2011), apresentaram técnicas não supervisionadas para retirar e estruturar conhecimento através de informações textuais. A figura 9 mostra a abordagem usada no processo de aprendizagem não-supervisionada.

Figura 9 - Esquematização do método não-supervisionado.

Fonte: Haykin (2001).

A figura 10 mostra a diferença entre as abordagens supervisionada e não-supervisionada. Observe que no aprendizado supervisionado existe um atributo alvo, este parâmetro é que serve como "guia" ao algoritmo. Através deste, o algoritmo realiza o cálculo do valor encontrado com o valor esperado, assim ele vai se corrigindo até encontrar a função ideal. Já no aprendizado não-supervisionado, o conjunto de treinamento não possui esse atributo, portanto o algoritmo tem que aprender por si só, somente através das semelhanças entre os atributos das instâncias, a hipótese, ou seja, a função que permitirá as classificações futuras.

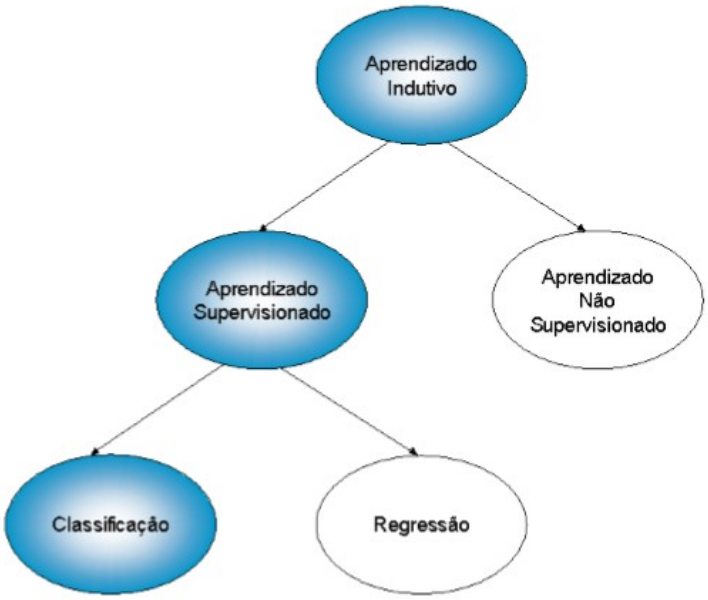
Figura 10 - Diferenças entre os conjuntos de dados de uma abordagem supervisionada e não-supervisionada.



Fonte: Monard e Baranauskas (2003). (Adaptado pela autora)

As atividades que abordam o conhecimento supervisionado podem ser divididas em classificação (discreto) e em regressão (contínuo). Entretanto, na abordagem não supervisionada os trabalhos podem ser classificados em: sumarização, procura-se um modelo que represente um mapeamento claro e resumido dos elementos; associação, relacionam-se as características dos dados procurando semelhanças e por fim, o agrupamento, cujos objetos são reunidos com base nas semelhanças que apresentarem, técnica muito usada em mineração de dados. O agrupamento é indicado quando não existe um conhecimento a *priori* dos dados. Os elementos são submetidos à rotina que analisa as semelhanças entre estes formando grupos conhecidos como *clusters* (FACELI et al., 2011). A figura 11 apresenta as subdivisões encontradas no aprendizado indutivo, ou seja, aquele que é realizado por intermédio de conjunto de dados adquiridos externamente.

Figura 11 - Subdivisões encontradas no aprendizado indutivo.



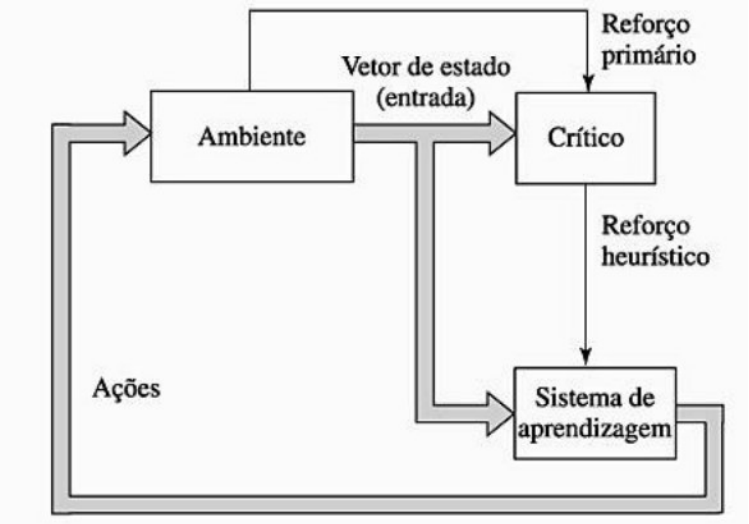
Fonte: Monard e Baranauskas (2003).

A aprendizagem por reforço se difere das metodologias anteriores, pois a extração do conhecimento é formulada através da interação do agente com o meio. Trata-se de uma abordagem mais ampla. O processo de aprendizagem é construído através dos resultados obtidos pela execução das tarefas efetuadas. Dessa forma, o agente inicialmente não tem ideia de qual tarefa realizar, então, através da investigação do ambiente ao este está interagindo, ele começa a desempenhar as atividades na qual ele recebe gratificações e tenta evitar as que resultam em prejuízos (LUGER, 2013).

No contexto de aprendizado por reforço a extração do conhecimento ocorre de forma desassistida, à máquina aprende através da interação deste com o ambiente na qual é submetido. Dessa forma, o sistema deve possuir a capacidade de formular um conceito mais geral por meio de suas experimentações. Neste paradigma, as ações e as conseqüências resultantes realizadas pelo agente são os responsáveis pelo processo de aprendizado.

De acordo com Luger (2013, p. 366), "o próprio agente de aprendizado, por tentativa e erro e realimentação, aprende uma política ótima para alcançar objetivos no seu ambiente." Dessa forma, esses dois procedimentos: a procura de tentativas e erros e reforço tardio são fundamentais no funcionamento e desempenho do agente. A figura 12 mostra o esquema de um sistema que usa o aprendizado por reforço.

Figura 12 - Fluxograma de um aprendizado por reforço.



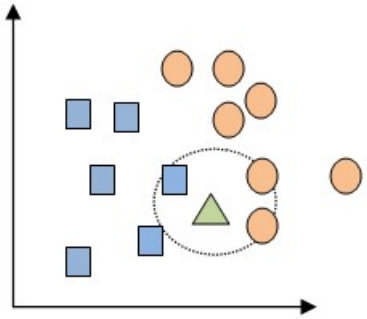
Fonte: Haykin (2001).

Embora existam diferentes abordagens de aprendizado, geralmente a mais usada é a abordagem supervisionada, devido à facilidade de construir algoritmos usando essa temática. Redes bayesianas, redes neurais artificiais, o método k-vizinhos mais próximos, árvores de decisão, *support vector machine* são exemplos de algoritmos que utilizam essa metodologia no processo de aprendizagem.

2.3.4 Algoritmos que utilizam a abordagem supervisionada

O algoritmo k-vizinhos mais próximos (*k-NearestNeighbors* ou KNN) utiliza um método cuja forma de aprender é baseada nas distâncias. No processo de treinamento é realizado o cálculo das distâncias de todos os atributos contidos na coleção de exemplos. Dessa forma, é necessário definir qual a métrica a ser utilizada, geralmente a mais usada é a distância euclidiana. Depois de estabelecida a métrica, o próximo passo é determinar o valor de *k*, que é feita pelo usuário, para definir quais elementos serão usados na comparação. Assim, cada vizinho fará uma votação no elemento novo a ser rotulado. A classe majoritária será a escolhida para determinar o rótulo da nova instância (FACELI et al., 2011). A figura 13 mostra a metodologia usada pelo método acima explanado.

Figura 13 - Metodologia do KNN usando o valor de K = 3.



Fonte: Faceli et al. (2011). (Adaptado pela autora)

O KNN é um algoritmo baseado nas distâncias acometidas entre as características presentes nos objetos. Como o processo de treinamento resulta dos cálculos efetuados dessas distâncias, as escalas dos atributos influem no desempenho. Dessa forma, é necessário usar a normalização dos dados antes de submetê-los ao algoritmo. Outro ponto que pode influenciar no desempenho de um algoritmo baseado nas distâncias dos atributos é o fator da dimensionalidade dos objetos. Assim, se os elementos do *dataset* de treinamento apresentar muitas características, a distância mais curta de um elemento em relação ao que vai ser categorizado acaba se igualando a distância obtida na instância mais distante (FACELI et al., 2011).

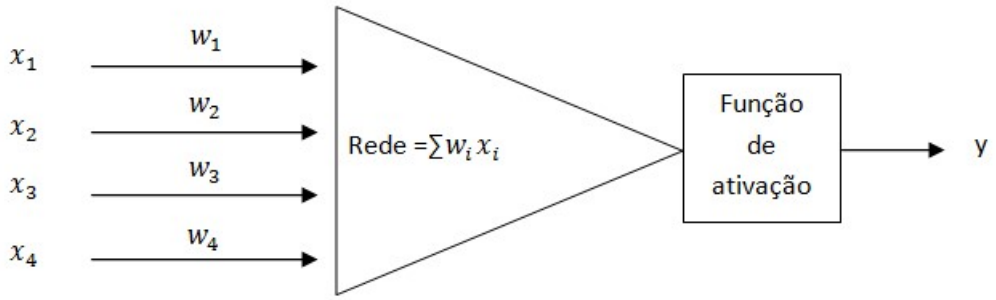
Devido a sua simplicidade de entendimento e implementação, o KNN é bastante usado pelos pesquisadores no desenvolvimento de seus projetos. Chen, Li e Teng (2013), usaram uma técnica chamada KNN *matting* baseada no algoritmo KNN na busca de vizinhos

não-locais. Seu o objetivo é extrair as múltiplas camadas de uma imagem, por exemplo, o primeiro plano com o fundo da imagem. Arefin et al. (2012) desenvolveram uma ferramenta de *software,* que usa o algoritmo KNN juntamente com uma Graphics Processing Unit (GPU) para melhorar o desempenho de classificação em larga escala.

A criação do neurônio artificial começou na década de 40, os pesquisadores da época elaboraram um paradigma conexionista. Para eles, um programa de computador poderia ser visto como um cérebro humano. Assim, as unidades individuais, seriam os neurônios que representam uma função matemática.

A figura 14 mostra com mais detalhes como seria a estrutura de um simples neurônio artificial. Cada elemento individual denominado neurônio é formado por uma quantidade de entradas (x_1,x_2, …, x_n) que são ponderados pelos vetores (w_1,w_2, …, w_n). Esses vetores representam os pesos da rede. Os vetores pesos são perpendiculares ao hiperplano, portanto, estes direcionam a função classificadora. Uma ligação no neurônio (sinapse) é formulada por x_i multiplicado por w_i. A próxima etapa consiste no somatório de todas as sinapses. Em seguida, a função de ativação limita o valor de saída do neurônio que podem ser valores discretos (0 ou 1) ou contínuos. (FACELI et al., 2011).

Figura 14 - Representação de um neurônio artificial.



Fonte: Luger (2013). (Adaptado pela autora)

Os neurônios podem estar conectados formando uma rede voltada para problemas de reconhecimento de padrões, classificação de dados, robótica, análise de voz e entre outras diversas aplicações. Trata-se de um algoritmo baseado em métodos de otimização em uma determinada função.

Uma rede neural artificial (RNA) é especificada pela arquitetura e aprendizado. A arquitetura está associada com os números de nós, quantidade de camadas e tipos de conexões. Já o aprendizado está voltado com configuração dos valores do vetor peso, pois sua

presença direciona o hiperplano determinando as saídas corretas ou não da rede. A escolha da topologia da rede é um fator bastante importante, porém não simples. Existe também outro tipo de conexão relacionada com a presença ou não de realimentação das camadas. As definições da arquitetura vão depender do tipo de problema a ser resolvido (FACELI et al., 2011).

Devido a sua popularidade, as RNAs são amplamente usadas em problemas de reconhecimento de padrões. Fiorin et al. (2011) aplicaram redes *perceptrons* multicamadas para realizar estimativas sobre a disponibilidade de energia solar no país, através de dados obtidos por estações da rede Sistema de Organização Nacional de Dados Ambientais. Debska e Guzowska-Swider (2011) utilizaram a capacidade de generalização das RNAs para classificar características presentes em cerveja. O intuito era analisar a qualidade das cervejas através de atributos presentes em sua composição. Araghinejad, Azmi e Kholghi (2011) combinaram classificadores para melhorar a previsão de vazões temporais de certos rios. Eles criaram redes neurais individuais e usaram o algoritmo KNN para associar as redes possibilitando a elaboração de um sistema com melhor capacidade de classificação. Dessa forma, é notório o poder de atuação dos algoritmos de aprendizagem de máquina na resolução de problemas complexos.

O algoritmo SVM é um algoritmo de aprendizado de máquina que pode ser utilizado adotando a abordagem supervisionada. Seu processo de treinamento é obtido através de métodos de otimização em uma determinada função. Um algoritmo de aprendizagem precisa ter meios de encontrar uma hipótese específica em uma família de possíveis formulações. Dessa forma, a maneira de encontrar essa solução é toda fundamentada na Teoria do Aprendizado Estatístico (TAE) cujo objetivo é encontrar um bom classificador genérico. A TAE apresenta vários procedimentos que devem ser mantidos para garantir a escolha de um exímio classificador (FACELI et al., 2011). Os dois tópicos seguintes abordam com mais detalhes a TAE e o funcionamento do algoritmo SVM.

2.4 TEORIA DO APRENDIZADO ESTATÍSTICO DAS MÁQUINAS DE VETORES DE SUPORTE

O algoritmo SVM possui sua fundamentação na Teoria do Aprendizado Estatístico. Nela são definidos vários passos que promovem o encontro de um modelo que apresenta um índice bom de generalização. Um modelo que possui tal característica é tido como ideal. Trata-se de um algoritmo que usa métodos de otimização para minimizar a função objetivo.

As máquinas de aprendizado *support vector machine* apresentam alguns pontos positivos em relação às redes neurais artificiais. Este algoritmo demonstra um bom desempenho em dimensões altas, tolerância a ruídos e um erro de mínimo global. Sua principal característica é ser um algoritmo determinístico, ou seja, a disposição da apresentação dos dados à rotina não altera a resposta obtida, diferente das redes neurais que apresentam conclusões aleatórias (FACELI et al., 2011).

O principal objetivo da TAE é a escolha de um exímio classificador. O algoritmo de AM precisa encontrar um classificador dentro de um conjunto possível de classificadores. Portanto, a TAE apresenta os critérios necessários para que o algoritmo possa encontrar a função desejada, ou seja, aquela que apresente erros mínimos na classificação dos conjuntos de treinamentos e de teste (FACELI et al., 2011).

Primeiramente, considera-se que os elementos são ajustados de maneira aleatória conforme a distribuição realizada pela atuação de uma função de probabilidade não conhecida. Dessa forma, o ambiente na qual o algoritmo é submetido possui uma característica de imutabilidade (HAYKIN, 2001). Dessa forma, quantificar as classificações exatas realizadas pelo classificador é uma maneira interessante de selecionar um bom classificador. Porém, a função que rotula os atributos de entrada com os de saída não é conhecida, portanto, o erro esperado não pode ser calculado de forma imediata. No algoritmo, apenas o conjunto de elementos do treinamento é conhecido, sendo assim, a solução para o dilema pode ser através do cálculo do risco empírico. Assim, entende-se que reduzindo o risco empírico, em consequência, depreende-se a redução do risco esperado (FACELI et al., 2011).

O risco empírico precisa apenas da coleção de instâncias de treinamento e isto é fornecido quando se trabalha com algoritmos que possuem a capacidade de aprender, portanto, pode ser usado sem dificuldades. O risco empírico corresponde à taxa de inexatidões realizadas pelo classificador no *dataset* de treinamento. Então, se chega à conclusão que o risco empírico não necessita de qualquer função de mapeamento probabilística e seu valor pode ser diminuído com as configurações realizadas através de parâmetros de AM (SANTOS, 2002).

Segundo Faceli et al. (2011) quando o conjunto de treinamento é formado por pouco objetos, o princípio da minimização do risco empírico pode não funcionar. Isto acontece porque às vezes quando se trabalha com dados escassos, estes se tornam insuficientemente relevantes e geram classificadores que tendem a memorizar os dados de treinamento. Neste tipo de cenário, o valor do risco empírico pode ser baixo, entretanto, o risco esperado tende a aumentar.

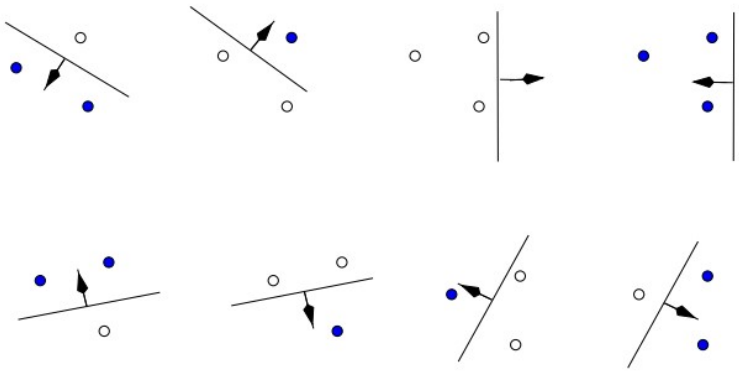
Uma hipótese selecionada pode pertencer a um conjunto de hipóteses. Portanto, uma alternativa para solucionar o impasse explanado anteriormente, é determinar algumas limitações no conjunto de hipóteses geradas. Uma destas restrições está relacionada com a complexidade da função elaborada, também denominada em algumas literaturas como capacidade. Modelos que possuem alta complexidade geralmente inclinam-se para a ocorrência de *overfitting* (FACELI et al., 2011).

Portanto, é realizada uma limitação no risco esperado, fazendo uma associação entre o risco funcional (esperado) com as taxas de erros obtidos nos dados de treinamento e um novo fator, que demonstra influência no processo de busca de um separador com a capacidade de representação mais genérica, definida como dimensão VC (Vapnik-Chervonenkis). Essa dimensão sugere abordagens empregadas em algoritmos de AM que visam aumentar o desempenho do modelo gerado na classificação dos dados (FACELI et al., 2011).

A dimensão de um algoritmo utilizado em AM se refere a uma característica que mostra a quantidade de elementos que compõem o conjunto de treinamento que podem ser divididos por uma determinada função. O fator VC trata-se de uma particularidade presente em uma coleção de funções, na qual o algoritmo busca uma função específica. Assim, em um problema de classificação binária, serão analisadas as funções contidas em um agrupamento de funções, em que se consigam dividir os elementos considerando todas as possibilidades binárias existentes (BURGES, 1998).

A figura 15 mostra um conjunto de funções lineares que podem ser formadas em um ambiente bidimensional considerando três pontos no espaço. Observe que é possível traçar oito hiperplanos que conseguem separar os dados em duas classes, portanto, o número VC em um ambiente bidimensional é três, pois esta é a maior quantidade possível de elementos que uma reta consegue particionar (LORENA; CARVALHO, 2003).

Figura 15 - Conjunto de funções lineares formadas através das oito possibilidades admissíveis em um ambiente R^2.



Fonte: Burges (1998).

Portanto, o valor da dimensão VC no espaço *n*é *n+1*, pois se deve levar em consideração a origem para que mantenham a característica de não serem dependentes dentro de um contexto linear (BURGES, 1998). Seu valor determina o nível de complexidade da função. Com o maior valor desta, maior complexidade do modelo, logo, classificadores com baixas taxas de exatidão nos dados de teste porque eles tendem a ficar superajustados aos dados de treinamento (FACELI et al. 2011).

O risco estrutural é obtido através da dimensão VC. Este estabelece uma relação entre uma função que apresenta poucos erros de classificação nos objetos de treinamento com o seu índice de complexidade. Procura-se por uma função mais genérica, portanto, com menor capacidade (FACELI et al. 2011). Dessa forma, este se depreende como uma função que visa diminuir as inexatidões observadas nos dados de treinamento e também minorar as falhas analisadas na etapa de classificação nos objetos novos a serem submetidos. A principal meta da máquina de aprendizado é achar uma função que sintetiza a compensação entre essas imperfeições evitando a possibilidade de ocorrer *overfitting* (BONESSO, 2013).

Comumente, obter o valor VC é algo complexo, em algumas situações é impossível descobrir seu valor. Então, outra opção é utilizar algumas características apresentadas pelas funções lineares que definem os classificadores. Uma destas particulares é relacionar o risco esperado com o conceito de margem. A longitude obtida entre o separador principal e o objeto pertencente ao conjunto de treinamento, resulta no conceito de margem. Dessa forma, ela pode ser entendida como fator de credibilidade do modelo. Portanto, a idéia seria maximizar a margem para obter uma menor medida de erro do classificador (FACELI et al., 2011).

Dessa maneira, tem-se conceituado o princípio do erro marginal. A idéia é que

quanto maior a distância do exemplo com o classificador, menor seria a complexidade da função gerada para separar os dados. Porém, é necessário que haja uma ponderação, pois o crescimento excessivo da margem pode violar as limitações em relação aos dados, porque todos estes devem apresentar a mesma distância ponderável em relação ao classificador. A diminuição da margem geralmente ocasiona uma redução do erro marginal, entretanto, eleva a complexidade da hipótese, podendo resultar em separadores dependentes dos dados de treinamento. Logo, o ideal é que haja um equilíbrio na definição desses termos. É necessário encontrar uma função que determina um classificador que possua uma margem de separação alta, promovendo um modelo mais genérico. Porém, esse separador precisa apresentar pequenos riscos marginais, reduzindo as taxas de inexatidões em relação aos objetos usados no treinamento e nos elementos de teste utilizados na classificação. Através do cumprimento sistemático destes conceitos é possível ao sistema de aprendizado obter o classificador desejado (FACELI et al., 2011).

2.5 ALGORITMO *SUPPORT VECTOR MACHINE*

Trata-se de um algoritmo que possui a capacidade de aprender de forma autônoma supervisionada, baseado na Teoria do Aprendizado Estatístico. Sua forma de aprendizado é derivada do processo de maximização das margens.

De acordo com o problema que o SVM tenta resolver, este pode sofrer variações, portanto, atualmente existem três denominações básicas do algoritmo. Estas categorizações são explanadas logo abaixo:

- SVM de margens rígidas, o hiperplano é traçado e dentro dos limites inferior e superior da reta tracejada (ambiente R^2), não apresenta dados nesta região.
- SVM de margens flexíveis, que manipula dados também considerados lineares, entretanto, admite a presença de alguns pontos dentro das margens delimitadas pelo hiperplano.
- SVM não lineares que manuseiam elementos com características de não linearidade, dessa forma, é necessário fazer uma alteração na dimensão de espaço dos dados através do uso das funções *kernels* para que a classificação possa ser realizada.

O algoritmo de aprendizagem SVM vem sendo empregadas em muitos trabalhos

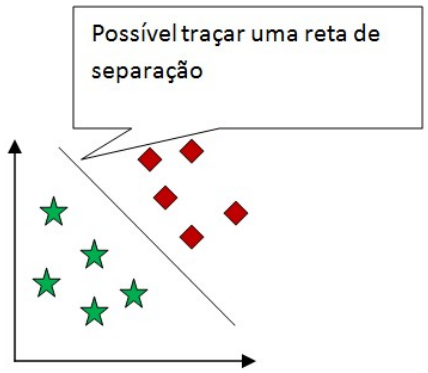
relacionados com reconhecimento de padrões. Yu, Yu e Zeng (2014) usaram o algoritmo para detectar estruturas de oligonucleótidos de cadeia simples. O reconhecimento consistia em uma classificação binária. Também foram usados algoritmos genéticos para promover a configuração dos melhores parâmetros a serem usados no algoritmo. Andreola e Haertel (2009) aplicaram o algoritmo SVM com um algoritmo de árvore binária para otimizar classificações multiclasses em imagens que possuíam alta dimensionalidade. A robustez do algoritmo SVM vem despertando o interesse de muitos pesquisadores na resolução de problemas de classificação.

2.5.1 SVM de margens rígidas

Este tipo de algoritmo será empregado em dados cuja característica é de apresentarem linearidade, ou seja, podem ser dividas por um hiperplano. Segundo Faceli et al. (2011) a partir deste modelo é que derivou-se as demais variações. De acordo com Santos (2002) devido à inflexibilidade das margens, este representante ficou conhecido como o Classificador de Margens Máximas.

Em problemas de classificação linearmente separáveis ocorre à existência de um hiperplano que separa os elementos contidos em um conjunto de dados em duas partes. Dessa forma, os elementos podem ser particionados em uma classe positiva ou negativa (BURGUES, 1998). A figura 16 mostra um conjunto de dados linearmente separáveis.

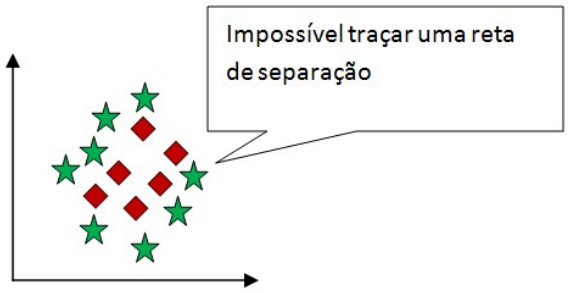
Figura 16–Objetos de classes distintas que podem ser separados por um hiperplano.



Fonte: própria autora.

Observe que na figura 16 o hiperplano pode ser traçado tranquilamente e há ausência de dados nos limites superior e inferior da reta traçada. Já a figura 17 apresenta um conjunto de dados inseparáveis linearmente.

Figura 17–Objetos de classes distintas que não podem ser separados por um hiperplano.



Fonte: própria autora.

Segundo Haykin (2001), a representação do hiperplano é definida pela seguinte equação:

$$f(x) = w^T x + b = 0 \quad (1)$$

O hiperplano pode ser definido como a universalização de um plano. Em um ambiente de uma dimensão, este será visto como um ponto; em um ambiente bidimensional, este terá a forma de uma reta e em um ambiente tridimensional, possui a característica de um plano. Este separador de classes pode ser obtido pela multiplicação escalar das variáveis *w* e *x*, sendo que *x* representa os elementos contidos na matriz *X* (conjunto de dados) e o *w* simboliza o vetor peso perpendicular ao classificador. A longitude do hiperplano associado à origem pode ser definida através de cálculos efetuados entre o vetor peso e o termo bias (*b*). Os valores de *w* e *b* são definidos na fase de aprendizado do modelo (FACELI et al., 2011).

De acordo com Haykin (2001), o problema de classificação envolvendo a separação de objetos em duas classes distintas pode ser resolvido adotando as seguintes formulações:

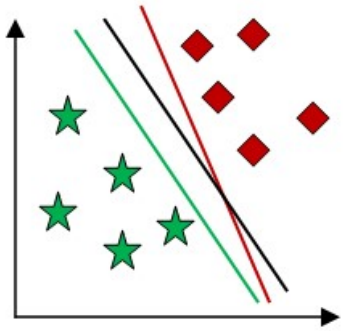
$$w^T x_i + b \geq 0 \, para \, y = +1 \quad (2)$$

$$w^T x_i + b < 0 \, para y = -1 \quad (3)$$

A distância mais próxima situada entre o classificador e um dado oriundo de um conjunto de dados é denominada margem de separação e é simbolizada por *p*. Quando *p* assume um valor positivo, é possível traçar diversos separadores, entretanto, estima-se encontrar o classificador que apresenta a maior longitude. O objetivo crucial na metodologia abordada pelo algoritmo é achar um classificador que oferece o valor de *p* mais elevado, por

fim, encontrando o hiperplano exímio (SEMOLINI, 2002). A figura 18 mostra como é possível traçar mais de um hiperplano na separação de duas classes.

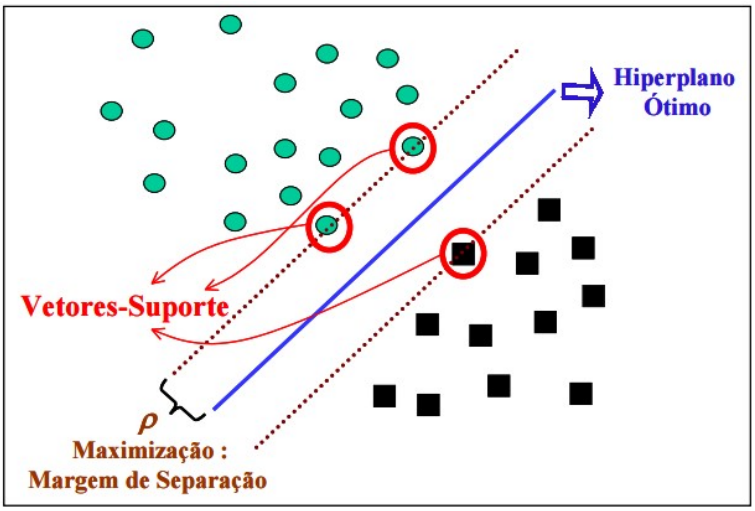
Figura 18 - Diferentes hiperplanos traçados na separação dos dados.



Fonte: própria autora.

Logo, máquinas de vetores de suporte é o método empregado em AM com a finalidade de conseguir o classificador que possui o maior índice de maximização das margens. Assim, os vetores de suporte são os elementos oriundos do conjunto de treinamento submetidos ao algoritmo localizados nas margens superior e inferior do separador (SEMOLINI, 2002). A figura 19 apresenta os vetores de suporte e o hiperplano exímio.

Figura 19 - Vetores de suporte e o hiperplano ótimo.



Fonte: Semolini (2002).

De acordo com Haykin (2001), o classificador desejado pode ser representado pela seguinte equação:

$$f(x) = w_o^T x + b_o = 0 \quad (4)$$

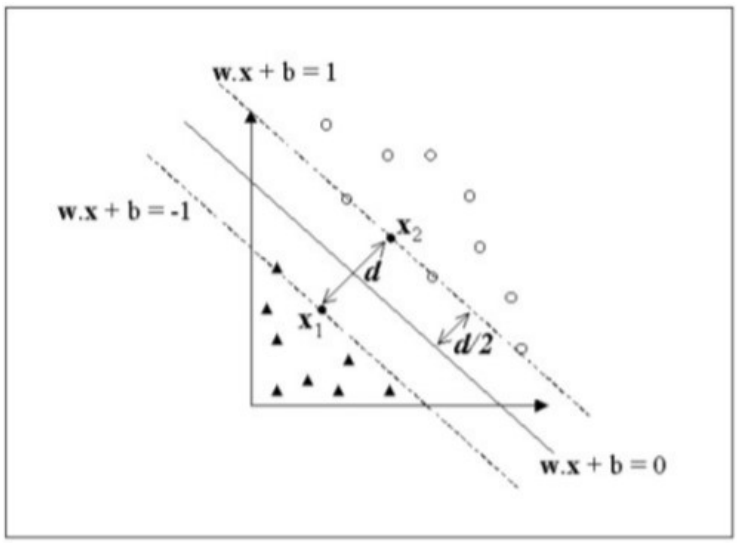
Assim, w_o simboliza o vetor ótimo e b_o o valor ideal do termo *bias.* A eq. (4) permite obter o valor da longitude de x até a linha divisora. Os valores do vetor peso e do termo *bias* ótimo devem ser adquiridos através dos dados de treinamento. Dessa forma, para encontrar os valores de w_o e de b_o pode-se utilizar as seguintes inequações (HAYKIN, 2001):

$$w_o^T x_i + b_o \geq 1 \, para \, y_i = +1 \quad (5)$$

$$w_o^T x_i + b_o \leq -1 \, para \, y_i = -1 \quad (6)$$

Portanto, os pontos contidos no conjunto de treinamento empregados nas eq. (5) e (6) cujo resultado é idêntico a +1 ou a -1 formam os conhecidos hiperplanos canônicos. Estes definem a margem superior e inferior do hiperplano principal. Assim, o próximo passo é o obter a distância dos vetores de suporte em relação ao hiperplano central (HAYKIN, 2001). A figura 20 ilustra a distância entre os vetores de suporte em relação à linha de separação central.

Figura 20-Obtenção da distância entre os hiperplanos canônicos.



Fonte: Lorena e Carvalho (2003).

Os vetores de suporte são os elementos mais importantes e mais penosos de serem encontrados. Portanto, para os hiperplanos canônicos pode ser feito a seguinte formulação:

$$g\left(x^{(s)}\right) = w_o^T x^{(s)} \mp b_o \mp 1 \; para y^{(s)} = \mp 1 \quad (7)$$

no qual $x^{(s)}$ representam os vetores de suporte e $y^{(s)}$ seus respectivos rótulos. Assim é possível obter a longitude algébrica de ponto $x^{(s)}$ em relação do hiperplano exímio usando as seguintes equações:

$$r = \frac{g(x^{(s)})}{||w_o||} \quad (8)$$

$$= \begin{cases} \frac{1}{||w_o||} \; se \; y^{(s)} = +1 \end{cases} \quad (9)$$

$$= \begin{cases} -\frac{1}{||w_o||} \; se \; y^{(s)} = -1 \end{cases} \quad (10)$$

cujo *r* representa a longitude dos vetores de suporte em relação ao separador desejado e w_o o vetor peso ótimo. Portanto para se obter margem máxima (*p*) pode ser efetuado 2 vezes *r*, ou seja, 2 / || w_o|| (HAYKIN, 2001).

Logo, para encontrar o hiperplano ideal deve-se minimizar a norma euclidiana do vetor *w*. Dessa forma, para prover a solução, chega-se a um dilema de otimização quadrática convexa que apresentam limitações que devem ser respeitadas no processo de busca de uma solução (HAYKIN, 2001):

$$y_i\left(w^T x_i + b\right) \geq 1 \; para \; i = 1,2, \ldots, N \quad (11)$$

A eq. (11) significa que devem ser achados os valores ótimos de *w* e *b* formadores do hiperplano ótimo. Esses valores devem ser encontrados através do conjunto de treinamento submetido ao algoritmo, entretanto com limitações. Segundo Faceli et al. (2011), a imposição desses limites possuem a finalidade de garantir que não ocorra a presença de elementos pertencentes ao conjunto de treinamento no espaço delimitado entre as margens. De acordo com Haykin (2001) o vetor peso *w* tem por regra diminuir a função de custo $\phi(w)$ definida logo abaixo:

$$\phi(w) = \frac{1}{2} w^T w \quad (12)$$

Existem soluções já estudadas para resolver o problema quadrático convexo. Para o caso explanado acima é utilizado o método de Lagrange (HAYKIN, 2001).

$$J(w,b\ ,\alpha) = \frac{1}{2}\, w^T w - \sum_{i=1}^{N} \alpha_i\, [y_i(w^T x_i + b) - 1] \tag{13}$$

na qual $J(w,b,\alpha)$ é a função lagrangiana. Os valores positivos simbolizados por α_i são conhecidos como os multiplicadores de Lagrange e simbolizam as restrições da eq. (11). Segundo Faceli et al. (2011) como se trata de um problema quadrático convexo é necessário reduzir a função de Lagrange, o que resulta em potencializar os multiplicadores e diminuir o vetor peso e o termo *bias* para solucionar o dilema. Segundo Haykin (2001) é necessário realizar as derivadas parciais da lagrangiana em relação aos dois valores que devem ser minimizados, dessa forma conseguindo o seguinte resultado:

$$w = \sum_{i=1}^{N} \alpha_i\, y_i x_i \tag{14}$$

$$\sum_{i=1}^{N} \alpha_i\, y_i = 0 \tag{15}$$

A eq. (14) mostra que o vetor peso pode ser obtido através dos dados de treinamento. O mesmo não acontece com a eq. (15). Os multiplicadores de Langrage aceitáveis nas restrições são os únicos com valores diferentes de zero. Isto pode ser definido através das condições de Karush-Kuhn-Tucker (KKT) que tem um papel fundamental na teoria e na prática de resolver problemas com restrições em otimização (HAYKIN, 2001). Segundo Burges (1998, p. 10), "essa técnica é assegurada para todos os SVMs, pois as condições são sempre lineares. Dessa forma, as condições KKT são necessárias e suficientes para que solução seja adquirida através de *w*, *b* e α" seguindo a seguinte imposição:

$$\alpha_i[y_i(w^T x_i + b) - 1] = 0 \quad para\ i = 1, 2, \ldots, N \tag{16}$$

Resolvendo o problema dual, resolve-se o problema primal. Assim, aplicam-se alguns procedimentos na langrangiana (13). Depois faz uma substituição nos termos resultantes utilizando as condições de otimização (14) e (15). Emprega-se a função objetivo, culminado na seguinte equação (HAYKIN, 2001).

$$Q(\alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \, \alpha_j y_i y_j x^T x_j \tag{17}$$

$$Com\ as\ restrições: \sum_{i=1}^{N} \alpha_i \, y_i = 0 \; e \; \alpha_i \geq 0 \; para \; i = 1, 2, \ldots, N \tag{18}$$

A forma dual é mais interessante, pois as restrições são mais fáceis de serem resolvidas e as soluções ficam apenas embasadas no conjunto de dados de treinamentos e seus respectivos atributos alvos. Na forma primal a solução fica apoiada nos valores de *w* e *b,* porém estes são desconhecidos. Dessa forma, calculam-se os multiplicadores de Lagrange para descobrir quais elementos dos conjuntos de treinamento que importam para a solução do problema, ou seja, os dados de treinamento que definem os vetores peso e o *bias* (FACELI et al., 2011).

De acordo com Haykin (2001) as restrições (18) expressam que dentro do conjunto de treinamento, devem-se achar os multiplicadores de Lagrange que potencializam a função objetivo. Com os valores já adquiridos é possível calcular o valor do vetor ótimo. A equação usada para essa finalidade é a seguinte:

$$w_o = \sum_{i=1}^{N} \alpha_{o,i} \, y_i x_i \tag{19}$$

cujos, $\alpha_{o,i}$ simbolizam os multiplicadores de Lagrange desejados e w_o representa o vetor de pesos exímio.

De acordo com Santos (2002), através do valor aproximando do vetor *w* é possível calcular o valor do termo aproximando de *b* (bias) por meio da seguinte formulação apresentada logo abaixo. Ressaltando que o termo $x^*(1)$ simboliza os elementos formadores de um dos hiperplanos canônicos referente à classe rotulada como +1 e o outro termo $x^*(-1)$ representa os que delimitam uma das margens pertinente à classe denominada como -1.

$$b_o = \frac{1}{2}[(w_o x^*(1)) + (w_o x^*(-1))] \tag{20}$$

Através dos valores calculados do vetor *w* e do *bias* ótimos juntamente com os vetores de suporte é possível obter o classificador ideal.Logo, em um problema de classificação binária, os valores contidos no conjunto de dados submetidos ao classificador

cujo resultado for positivo, significa que este elemento pertence a classe +1. Caso o valor seja negativo, isso expressa que este dado pertence à outra classe denominada -1.

2.5.2 SVM de margens suaves

No mundo real, os objetos podem ser serializados ou representados na forma de dados, porém, a maioria destes apresenta valores acima ou abaixo dos padrões, portanto, não possível usar a teoria das SVMs rígidas neste tipo de cenário. Desta forma, foi realizada uma alteração no SVM clássico para tratar de dilemas que possam apresentam erros de treinamento. Segundo Santos (2002) essa modificação resultou em acrescentar uma "folga" nas equações, para que pontos que se apresentassem dentro da margem superior ou inferior fossem admitidos. Essa nova técnica foi batizada de SVMs de margens suaves.

Então, se acrescentou um relaxamento nas margens de delimitação, representado por um conjunto de variáveis escalares positivas. Assim, alguns elementos podem ser encontrados entre as margens. A infração às restrições pode ser efetuada de duas formas: o elemento é classificado de maneira correta, porém apresenta-se situado entre as margens, dessa forma tem-se $0 \leq \mathcal{E}_i \leq 1$; outra possibilidade é quando o dado é classificado de forma errônea, logo, $\varepsilon_i > 1$. Portanto, a formulação resultante após a introdução dessa folga pode ser expressa assim (HAYKIN, 2001):

$$y_i(w^T x_i + b) \geq 1 - \varepsilon_i, \qquad i = 1, 2, \dots, N \tag{21}$$

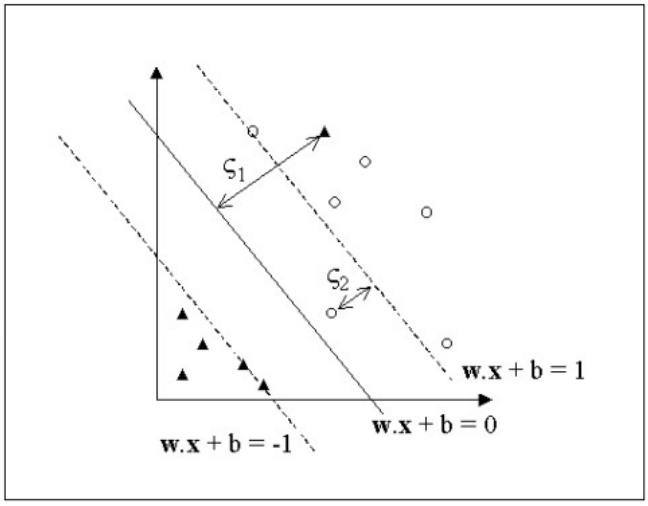
Dessa maneira, atribui-se um custo extra de erros chamado de C alterando a função objetivo. Neste caso, o usuário seleciona o valor do parâmetro C, quanto maior o valor de C mais elevado será a penalidade de erros, portanto, o parâmetro C passa a ser um item regularizador (BURGES, 1998). Portanto, para maximizar as margens, deve-se novamente minimizar a norma euclidiana de *w.* Segundo Haykin (2001) após a introdução das folgas e do parâmetro C,o resultado da função que define o custo resulta em:

$$\Phi(w, \varepsilon) = \frac{1}{2} w^T w + C \sum_{i=1}^{N} \varepsilon_i \tag{22}$$

A figura 21 mostra como fica a margem superior e inferior com a adição das variáveis de folga possibilitando a presença de alguns dados entre as delimitações das margens. Dessa

forma, é possível a ocorrência de alguns pontos dentro das delimitações das margens.

Figura 21 - Delimitação das margens com a presença das folgas.



Fonte: Lorena e Carvalho (2003).

Repetidamente, chega-se a um problema quadrado convexo que deve ser resolvido por Lagrange. Dessa maneira, são realizados os mesmos passos explanados nas SVMs que adotam o conceito de inflexibilidade nas margens para se chegar à fórmula final que estabeleça o classificador. A diferença neste cenário é que vão aparecer outros tipos de vetores de suporte, os livres e os limitados.

2.5.3 SVM não - linear

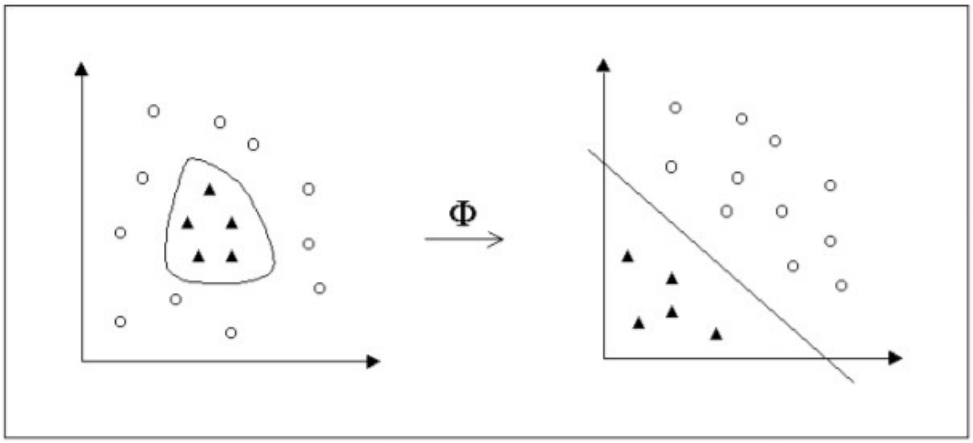
Como foi dito anteriormente, a probabilidade dos dados de apresentarem linearidade é muito baixa e para se trabalhar com as SVMs rígidas e flexíveis é imprescindível a separação do conjunto de dados em forma linear.

Para possibilitar que o SVM pudesse manipular elementos que não apresentam linearidade, foram inseridas funções reais que realizam o mapeamento de um espaço de entradas para uma nova dimensão. Este novo ambiente é chamado espaço de características (FACELI et al., 2011).

De acordo com Cover (1965), geralmente os dados que se apresentam não lineares em uma dimensão mais baixa, tendem a ser lineares quando estes são expostos a ambientes que possuem maior dimensionalidade. A figura 22 demonstra visualmente dados que

apresentam não linearidade em um ambiente de duas dimensões, porém quando mapeados para uma dimensão maior, exibem linearidade.

Figura 22 - Dados linearmente inseparáveis em R² (esquerda) e dados linearmente separáveis em uma nova dimensão maior (direita).



Fonte: Lorena e Carvalho (2003).

Para trabalhar com as SVMs não-lineares, primeiramente deve-se usar a função real φ nos dados de entrada, fazendo a transformação destes para um ambiente que possui alta dimensionalidade. Depois, utiliza as computações que realizam a elaboração do hiperplano ideal em cima destes dados transformados (FACELI et al., 2011). Portanto, o separador de classes pode ser definido assim:

$$\sum_{j=1}^{m_1} w_j \varphi_j(x) + b = 0 \tag{23}$$

na qual $\varphi_j(x)$ é a função transformadora, m_1 é a nova dimensão e $\{w_j\}_{j=1}^{m_1}$ simboliza o agrupamento dos vetores pesos com a nova dimensão. Considerando $\varphi_o(x) = 1$ para todo vetor de entrada e w_o simbolizando o termo *bias*, o separador de classes pode ser reescrito dessa maneira (HAYKIN, 2001):

$$w^T \varphi(x) = 0 \tag{24}$$

Adotando a eq. (19) neste novo cenário, obtêm a seguinte formulação do vetor peso:

$$w = \sum_{i=1}^{N} \alpha_i \, y_i \varphi(x_i) \tag{25}$$

sendo $\varphi(x_i)$ simbolizando os vetores de entrada x na nova dimensão. Realizando as permutas entre as equações (26) e (24) chega-se na seguinte formulação para o vetor peso em ambientes não lineares (HAYKIN, 2001):

$$w = \sum_{i=1}^{N} \alpha_i \, y_i \varphi^T(x_i)\varphi(x) = 0 \tag{26}$$

O mapeamento da entrada de dados para outras dimensões afeta o funcionamento do SVM, às vezes o cálculo da função real φ pode gerar problemas de difíceis soluções ou até intratáveis, devido ao aumento da dimensionalidade. Quando adiciona essa função no problema de otimização das SVMs lineares, é observado que a transformação de um objeto no ambiente original para o atual (dimensão maior) pode ser simplesmente obtido pelo produto interno entre os elementos no seu novo ambiente. Portanto, estes problemas podem ser revolvidos usando as funções *kernels* (FACELI et al., 2011).

2.6 FUNÇÕES *KERNELS* E OUTRAS TÉCNICAS

A função *kernel* é uma função simétrica cuja responsabilidade é mapear o vetor de entradas x em uma dimensão nova e maior. Dessa forma, o classificador desejado pode ser definido no espaço de características com um custo computacional menor:

$$K(x, x_i) = \varphi^T(x)\varphi(x_i) = \sum_{j=0}^{m_1} \varphi_j(x)\varphi_j(x_i) \; para \; i = 1, 2, \ldots N \tag{27}$$

no qual, $\varphi^T(x)\varphi(x_i)$ simboliza o produto escalar de dois vetores situados em uma maior dimensão, possibilitando uma superfície de separação (HAYKIN, 2001).

Segundo Lima (2002) ao se trabalhar com SVM na resolução de problemas é preciso ter atenção ao adotar as configurações corretas para se obter uma boa representação genérica dos elementos. Estas composições compreendem essencialmente na configuração do parâmetro C que representa o custo, ou seja, o quanto de relaxamento acometido nas restrições das margens; a função real *kernel* que oferece uma melhor condição de organização dos espaços na qual os objetos estão situados e, os seus parâmetros, nos quais representam os ajustes necessários para que o *kernel* consiga realizar bem seu propósito que é fornecer uma boa configuração de superfície dos conjuntos de dados possibilitando uma melhor atuação do hiperplano classificador. A tabela 4apresenta a formulação matemática dos quatro *kernels*

mais populares ou mais utilizados citados acima.

Tabela 5 - Formulação matemática dos *kernels* mais populares.

Tipo do *kernel*	Formulação matemática
Linear	$K(x_i, x_j) = x_i^T x_j$
Polynomial	$K(x_i, x_j) = (\gamma x_i^T x_j + r)^d, \gamma > 0$
Radial BasisFunction	$K(x_i, x_j) = \exp(-\gamma \|\|x_i - x_j\|\|)^2, \gamma > 0$
Sigmoid	$K(x_i, x_j) = \tanh(\gamma x_i^T x_j + r)$

Fonte: Chang e Lin (2011). (Adaptado pela autora)

O limite de separação do modelo está intimamente ligado à seleção e a configuração dos parâmetros do *kernel.* Portanto, o conhecimento e manipulação dos parâmetros são fundamentais para uma boa performance do modelo gerado (FACELI et al., 2011).

Na busca pela melhor exatidão do algoritmo, além dos parâmetros usados, tipos de *kernels* e classificadores, podem ser adotados também, outras técnicas. Métodos como *houldout* e *cross-validation* são muito usados para avaliar o desempenho do modelo gerado. Segundo Faceli et al. (2011) na técnica de *houldout* um terço do conjunto de dados são usados na parte de teste e os dois terços restantes são utilizados na fase de treinamento. Este procedimento é, geralmente, indicado quando possui uma vasta quantidade de elementos na coleção. Entretanto, no procedimento de *v-foldcross-validation* a coleção de objetos é segmentado em v partes, normalmente do mesmo tamanho. Por exemplo: se um conjunto de dados for dividido em v = 3 (partesbloco_1, bloco_2 e bloco_3), no primeiro ciclo serão usados os dadosbloco_1 e bloco_2 para treinamento e bloco_3 como teste; no segundo ciclo utilizar-se-ão os dados bloco_2 e bloco_3 para treinamento e bloco_1 para teste e, finalmente, no terceiro ciclo, os dados bloco_1 e bloco_3 para treinamento e bloco_2 para teste. Assim, a avaliação do modelo preditivo será obtida pela média calculada sobre os desempenhos de cada teste.

No trabalho de Piooznia e Deng (2006) são mostrados outros procedimentos que também podem ser aplicados na utilização do algoritmo SVM:

- *Shrinking:* trata-se de uma técnica cujo objetivo é promover a

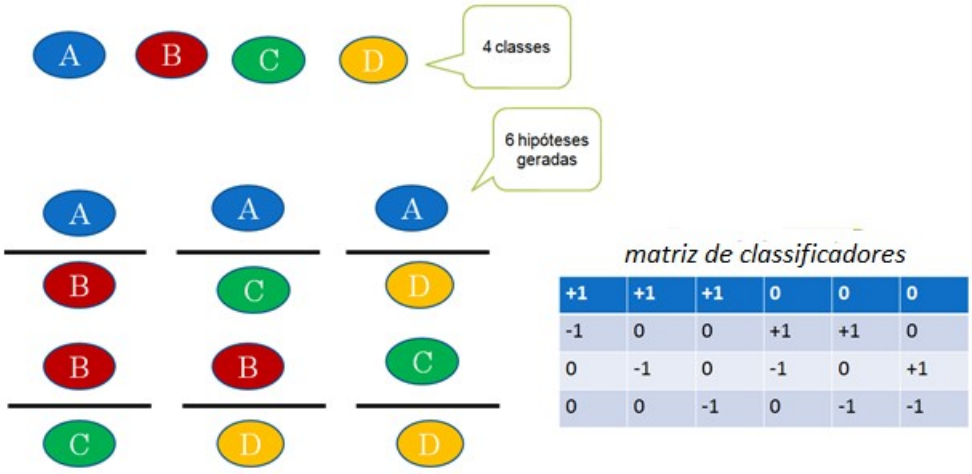
minimização do esforço do problema. Segundo Chang e Li (2011) o intuito é retirar alguns multiplicadores de Lagrange já definidos melhorando o tempo de treinamento. Glasmachers, Igel (2006) relata que técnicas de heurísticas no *shrinking* são empregadas para reduzir a quantidade de variáveis envolvida nos cálculos.

- *Caching:* consiste em outro método para tentar minimizar o tempo de computação gasto. De acordo com Chang e Li (2011) usualmente o tamanho da matriz usada no processo de treinamento é grande, portanto, uma forma de consumir menos recursos computacionais é obter os elementos da matriz apenas quando necessários e os armazenado em um lugar específico destinado à computação dos elementos da função *kernel*.

Inicialmente o algoritmo SVM foi formulado para tratar problemas de classificação envolvendo apenas duas classes. Portanto, um modelo gerado possuía a habilidade de separar objetos pertencentes a duas classes distintas. Depois foram adicionados outros métodos no algoritmo para que este oferecesse suporte a classificações multiclasses. De acordo com Faceli et al. (2011) as abordagens mais utilizadas para separar objetos de diversas classes são:

- Método um-contra-um: nessa metodologia são construídos $k\ (k\text{-}1)/\ 2$ modelos, sendo que k corresponde a quantidade de classes que rotulam cada objeto no conjunto de treinamento. Cada modelo gerado terá a capacidade de classificar apenas duas classes por vez. Apesar de apresentar um processo de aprendizado mais rápido, um ponto negativo é que alguns modelos sempre classificaram dados de forma errônea. A figura 23 mostra um exemplo para melhor entendimento.

Figura 23 – Abordagem um-contra-um em problemas de classificação multiclasse.

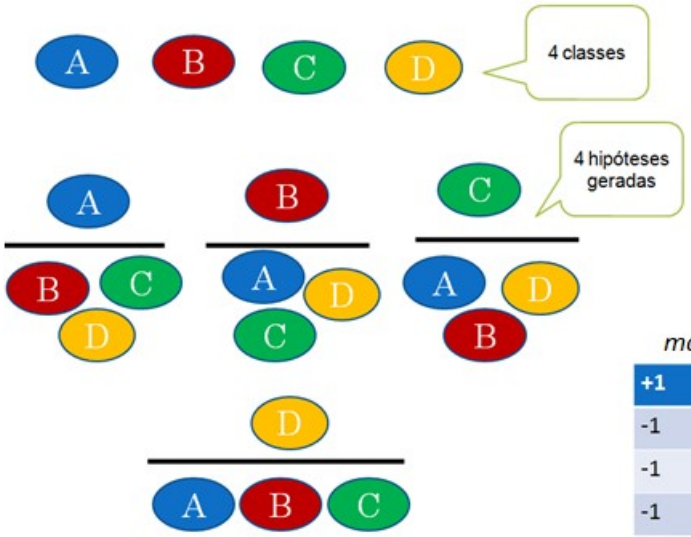


Fonte: própria autora.

Conforme pode ser visto na figura 23, se são usadas quatro classes no treinamento serão gerados seis classificadores. Cada modelo terá a responsabilidade de aprender a distinguir duas classes. Cada coluna da matriz de classificação define um classificador. Portanto, na primeira coluna tem-se um modelo que fará a separação dos objetos na classe A ou B, a segunda coluna realizará a classificação dos elementos na classe A ou C e assim por diante.

- Método um-contra-todos: nessa metodologia, os classificadores têm que aprender a separar uma classe em relação às outras. Se existe *k* classes têm-se *k* modelos. A desvantagem desse procedimento é que o tempo de treinamento é maior. A figura 24 mostra com mais detalhes o processo empregado.

Figura 24 – Abordagem um-contra-todos em problemas de classificação multiclasse.



Fonte: própria autora.

De acordo com a figura 24 para classificar quatro classes usando o método um-contra-todos devem ser gerados quatro classificadores. Cada coluna da matriz de classificação corresponde a um modelo gerado. Portanto, na coluna um tem-se uma hipótese que deve reconhecer a classe A das demais, na segunda coluna o modelo deve aprender a distinguir a classe B das demais e assim sucessivamente.

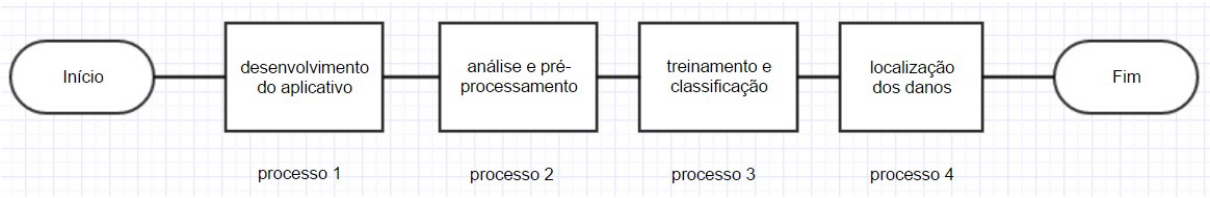
Conforme explanado neste tópico, o algoritmo SVM possui muitas técnicas e conceitos que contribuem efetivamente para uma boa classificação. Logo, é fundamental compreender o funcionamento do algoritmo e cada técnica empregada nos dados com o intuito de obter o melhor desempenho do modelo gerado.

3 METODOLOGIA

O objetivo deste trabalho é a proposição de um método de monitoramento da condição estrutural baseado em conceitos de inteligência artificial que utilizam um algoritmo de aprendizado de máquina. O algoritmo empregado é o *support vector machine* cuja tarefa é aprender, através de uma abordagem supervisionada, um modelo que servirá para classificar futuras informações. Esse classificador é obtido a partir dos dados gerados pelos sensores / atuadores piezelétricos colocados em uma placa de alumínio. Este monitoramento tem o intuito de obter dados que representam a ausência ou a presença de danos estruturais. Dessa forma, o trabalho propõe uma abordagem para promover as duas etapas relacionadas com o monitoramento da integridade estrutural: identificação e localização dos danos em uma estrutura. Essa metodologia estabelece o uso de algoritmos de aprendizado de máquinas através de uma abordagem supervisionada para a realização das duas fases.

Para uma melhor compreensão da metodologia empregada para elaborar este trabalho, foi criado um fluxograma relatando as fases usando um contexto mais genérico. A figura 25 mostra as três etapas que compõem este fluxograma geral.

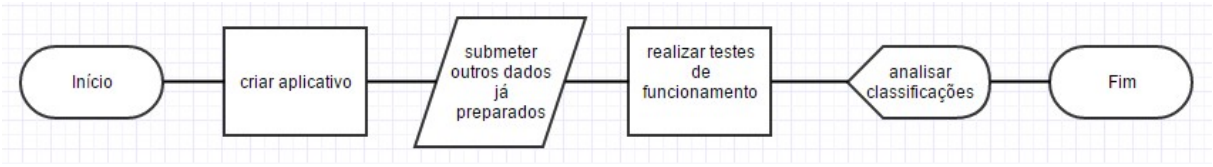
Figura 25 -Fluxograma geral.



Fonte: própria autora.

Observe que o fluxograma da figura 25 explana os três processos que basicamente compõem as etapas a serem desenvolvidas: o processo 1, consiste essencialmente na parte de criação do aplicativo, possibilitando a utilização do algoritmo; o processo 2, abrange as fases de manipulação dos dados a serem usados na máquina de aprendizado, o processo 3 contém as etapas necessárias para o treinamento e classificação dos dados e o processo 4 consiste em utilizar o arquivo de classificação gerado pelo algoritmo e possibilitar a localização dos danos na estrutura. A figura 26, mostra com mais detalhes as fases que compõe o processo 1.

Figura 26 -Sequência de passos para a realização do processo 1.



Fonte: própria autora.

De acordo com a figura 26, foi necessário inicialmente criar o aplicativo que possibilitou a execução do algoritmo SVM. Depois, o aplicativo foi alimentado com dados já pré-processados oriundos de repositórios voltados para aprendizado de máquinas. Estes dados, portanto, possibilitaram realizar testes para entender o funcionamento do algoritmo. E por último, foram observados os resultados verificando a taxa de exatidão. Os textos subsequentes relatam com mais detalhes o que foi preciso para concluir o processo 1.

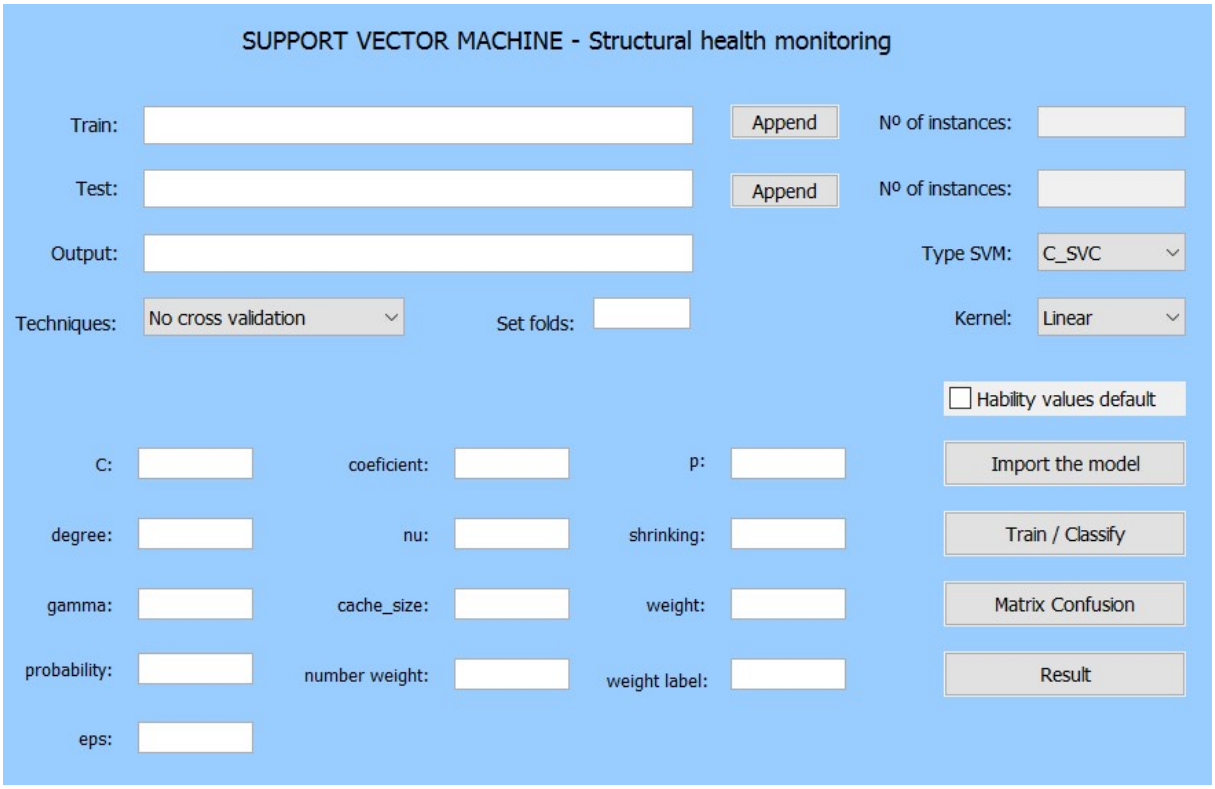
Através da linguagem de programação Java foi elaborado um aplicativo para facilitar ao usuário o manuseio com os dados e com o algoritmo. Este programa foi desenvolvido usando o ambiente de desenvolvimento integrado (Integrated Development Enviroment - IDE) chamado NetBeans versão 8.1, disponível no endereço https://netbeans.org/downloads/. Trata-se de uma ferramenta que auxilia na produção de *softwares*, pois esta, já disponibiliza uma série de recursos como um editor de código-fonte, um atalho para chamar o compilador e o executor, permite realizar depuração do código, oferece bibliotecas que permitem criar programas com interfaces gráficas ao usuário, enfim, uma coleção de funcionalidades práticas para o programador. Ao entrar no endereço citado anteriormente, basta baixar a versão mais leve que possui apenas os itens: Software Development Kit (SDK) do NetBeans e a plataforma Java 2 Standard Edition.

Após a obtenção da IDE, foi preciso baixar o Java Kit Development (JDK) do Java. Trata-se de um conjunto mínimo necessário para desenvolver aplicações voltadas para ambientes *desktops*. Foi usada a versão 1.8, para realizar o *download* do *kit*, basta acessar o seguinte endereço http://www.oracle.com/technetwork/pt/java/javase/downloads/jdk8-downloads-2133151.html. Depois de realizar o processo de instalação tanto do JDK e do SDK, foi possível iniciar o processo de criação do aplicativo.

Para continuar com o desenvolvimento do aplicativo, era primordial obter a biblioteca que permitisse a utilização do algoritmo, para tal, foi usado uma reescrita da LibSVM. Para maiores informações sobre os autores, acesso à biblioteca e licença de uso consulte o Apêndice A deste trabalho.

Após o adicionamento da biblioteca, foi possível terminar a parte do aplicativo relacionada com a execução do algoritmo. Conforme pode ser visto na figura 27, o aplicativo possui campos que permitem que usuário realize as predições de maneira bem amigável. Dessa forma, não é necessário conhecimento em linguagens de programação para utilizar os recursos oferecidos pela biblioteca. Assim, futuramente poderia ser desenvolvido um produto que possa ser usado na manutenção para identificação e localização de danos em estruturas.

Figura 27 - Interface gráfica do aplicativo.



Fonte: própria autora.

Inicialmente para saber como funciona o algoritmo SVM, foram feitos alguns testes usando o conjunto de dados *wine*. Esse conjunto pode ser baixado no site http://archive.ics.uci.edu/ml/datasets/Wine. Esse *dataset* é oriundo de resultados de análises químicas sobre vinhos de uma mesma região da Itália derivados de três cultivos diferentes. O conjunto é composto por cento e setenta e oito elementos sendo que cada instância possui treze características e um atributo alvo que corresponde a classe do vinho, que é classificado como 1, 2 ou 3. Na primeira parte de testes, foi usado o tipo SVM *C-SVC* (usado para classificação) e os *kernels*: *linear*, *polynomial*, *radial basis function* (RBF) e *sigmoid*. Para o teste com o *kernel linear* foram usados os valores padrões. A tabela 5 mostra os parâmetros

usados, sua finalidade ou significado e os seus valores padrões.

Tabela 6 - Parâmetros usados pelo algoritmo SVM e seus valores padrões.

Parâmetros	Valores Padrões	Finalidade / Significado
C	1	referente ao custo, ou seja, a penalidade de erros. Quanto maior o valor de C, menor será a maximização das margens, dessa forma, tem-se um classificador com menos erros de classificação, embora mais complexo. Porém, um valor menor de C favorece margens mais suaves.
degree	3	valor usado na função *kernel polynomial*.
gamma	0	parâmetro usado na função *kernel polynomial*, *RBF* e *sigmoid.*
probability	0	realizar estimativas de probabilidades (os valores podem ser 0 ou 1).
eps	0.001	critério de parada.
coeficient	0	parâmetro usado na função *kernel polynomial* e *sigmoid*.
nu	0.5	valor usado nos outros tipos de SVM como: *nu-SVC*, *one-class SVM* e *nu-SVR*. Equivale ao parâmetro C.
cache-size	100	configura o tamanho da memória a ser usada pelo *kernel.*
p	0.1	usado no SVM do tipo *epsilon-SVR*.
shrinking	1	serve para definir o uso da técnica de *shrinking* (os valores podem ser 0 ou 1).
weight	0	Esses três parâmetros são usados para conjuntos muito desbalanceados, caso o usuário deseje colocar pesos nas classes. Empregado somente no *C-SVC*.
weight label	0	
number weight	0	

Fonte: Chang e Lin (2011). (Adaptado pela autora)

O resultado do teste realizado utilizando o *dataset wine* foi bastante motivador, o algoritmo chegou a uma exatidão de aproximadamente 99,4%, ou seja, de 178 elementos o SVM acertou 177. A figura 28 mostra com mais detalhes o resultado.

Figura 28 -Resultado do teste com o SVM tipo C-SVC usando *kernel linear* e parâmetros padrões.

```
*
optimization finished, #iter = 37
nu = 0.10707336757934471
obj = -8.385472190906668, rho = -1.488708298689322
nSV = 17, nBSV = 9
Total nSV = 37
Accuracy = 99.43820224719101% (177/178) (classification)
```

Fonte: própria autora.

Na figura 28 pode ser observado o aparecimento de outras variáveis. A LibSVM sempre apresenta esses resultados após o treinamento. O símbolo *obj* representa o valor exímio da função objetivo que surge devido ao procedimento de maximização das margens. O termo *rho* representa uma constante na função de decisão. Parâmetro *nu*, cuja funcionalidade equivale ao termo *C*, porém, este é usado no SVM tipo *nu-SVC*. O resultado da biblioteca também apresenta a quantidade de vetores de suporte empregados na construção do hiperplano. O *nSV* refere-se a número de vetores de suporte livres (aqueles responsáveis em formar os hiperplanos canônicos). O *nBSV* refere-se ao vetores de suporte limitados que aparecem devido ao acréscimo dos relaxamentos nas margens (CHANG; LIN, 2011). Segundo Faceli et al. (2011) os vetores de suporte limitados podem ser aqueles objetos que aparecem dentro do espaço delimitado pelos hiperplanos canônicos e são classificados de forma correta. Outra variação de vetores de suporte limitados são aqueles rotulados de forma errônea. O trabalho realizado por Chang e Lin (2011) explana com detalhes toda a implementação do algoritmo SVM.

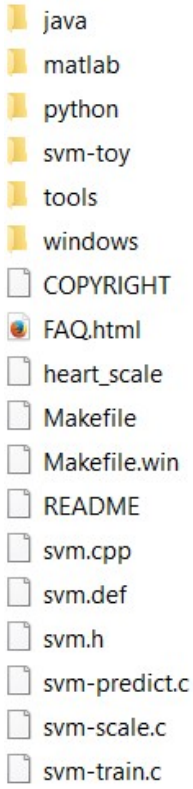
Utilizando a abordagem anterior, foram realizados mais testes usando os outros *kernels* restantes, porém os resultados não foram satisfatórios. Isto aconteceu por causa da configuração dos valores padrões dos parâmetros. Portanto, é necessário fazer alterações nos parâmetros para cada tipo de *kernel* a ser empregado.

O próximo passo para os testes foi estabelecer a configuração dos parâmetros. Para iniciantes, esta etapa é muito difícil, embora seja importante, pois esta influencia diretamente no processo de treinamento. Esta fase pode resultar em um bom ou mau desempenho do

algoritmo. Existem muitos artigos relacionados com essa etapa, porém os autores da biblioteca LibSVM, Chang e Lin (2011) criaram alguns arquivos para auxiliar as pessoas a utilizarem o SVM de uma maneira mais simples que serão apresentados logo a seguir. Este pacote pode ser baixado no seguinte endereço http://www.csie.ntu.edu.tw/~cjlin/libsvm.

Depois de realizado o *download* do pacote compactado, basta descompactá-lo em qualquer diretório. Dentro da pasta *libsvm 3.21* estão armazenadas várias outras pastas como pode ser visto na figura 29.

Figura 29 -Estrutura de diretórios da ferramenta LibSVM.

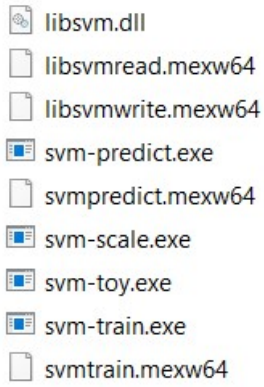


Fonte: própria autora.

A pasta *libsvm 3.21* oferece suporte a várias outras linguagens como pode ser visto na figura 29, possibilitando a utilização do algoritmo em diversas linguagens de programação de acordo com a preferência do programador.

Outras pastas interessantes, são as pastas *windows* e *tools*. Na *windows* são encontradas uma série de arquivos executáveis que permitem usar o algoritmo SVM de forma bem prática. A figura 30 apresenta estes arquivos.

Figura 30 - Estrutura da pasta *windows* da biblioteca LibSVM.



Fonte: própria autora.

Os arquivos responsáveis pelo treinamento e classificação do algoritmo são o *svm-train.exe* e *svm-predict.exe.* Para maiores detalhes de utilização basta ler o arquivo README contido dentro do pacote *libsvm 3.21*.

Como já foi citado anteriormente, para se trabalhar com os *kernels* é necessário configurar os valores de parâmetros. Com esse propósito na pasta *tools* situada dentro da pasta *libsvm 3.21* existe o arquivo *grid.py* que oferece um direcionamento na escolha dos melhores parâmetros na configuração do *kernel* RBF. A tabela 6 mostra alguns valores para os parâmetros C (*cost*) e g (*gamma*) em relação aos *kernels: linear, polynomial, RBF e sigmoid* para o conjunto de dados *wine.train*. Os demais parâmetros ficaram com os valores padrões.

Tabela 7 - Configuração dos parâmetros C (*cost*) e g (*gamma*) com *kernels* distintos para o *dataset wine.train.*

Kernel	**C (*cost*)**	**g (*gamma*)**	**Taxa de exatidão**
Linear	1,0	0	~99,4 %
Polynomial	0,5	0,5	~99,4%
RBF	0,5	0,5	~99,4%
Sigmoid	0,5	0,5	~96,6%

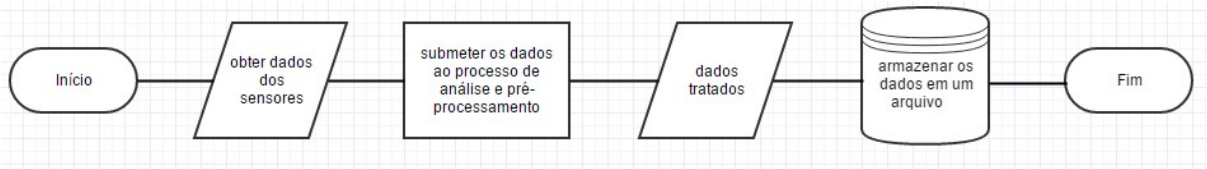
Fonte: própria autora.

Os testes realizados mostraram a importância da escolha dos parâmetros e foram importantes para adquirir conhecimento na forma de usar o algoritmo SVM. Portanto, pode-se seguir para a próxima etapa do projeto. Esta fase consiste em usar realmente os dados obtidos

pelos sensores e atuadores colocados na placa de alumínio.

A figura 31compreende os passos que compõem o processo 2 do fluxograma geral, que foi abordado na figura 25. Pode ser observado que as etapas descritas compreendem essencialmente a fase de análise e pré-processamento de dados. O intuito é preparar os dados, melhorando a qualidade destes, deixando-os aptos a serem usados pela máquina de aprendizado.

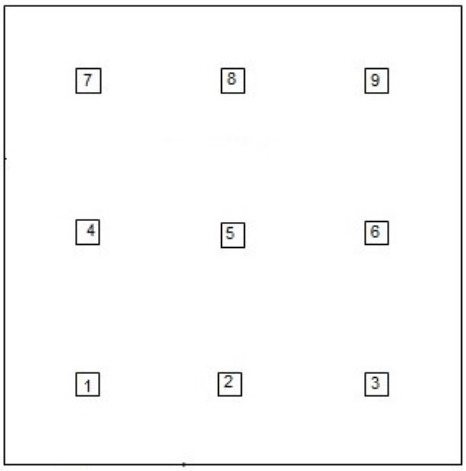
Figura 31-Sequência de passos do processo 2 do fluxograma geral.



Fonte: própria autora.

Os dados experimentais foram fornecidos por Rosa (2016), que também faz parte do grupo GMSINT (Grupo de Materiais e Sistemas Inteligentes). Diferentemente do atual trabalho, Rosa utiliza uma técnica de imagens para detecção e localização de danos. Os resultados obtidos por Rosa foram importantes para comparação e validação da proposta deste trabalho. Maiores informações sobre as medidas podem ser obtidas no referido trabalho, no entanto, para clareza deste trabalho, serão fornecidas informações básicas dos testes experimentais. Inicialmente foram colocados nove sensores / atuadores piezelétricos em uma placa de alumínio sem danos estruturais. Cada atuador envia um sinal para os outros sensores, portanto, um atuador gera oito caminhos para os demais sensores. Por exemplo: o caminho do atuador 1 para sensor 2 é chamado de P12, o caminho do atuador 1 para sensor 3 é chamado de P13 e assim sucessivamente.Dessa forma, tem-se um mapeamento de todas as medidas na placa. Os sensores / atuadores foram adicionados em uma placa de alumínio com dimensões: 500 milímetros (mm) de comprimento por 500 mm (de altura) por 2 mm(de largura). A figura 32 mostra a disposição dos sensores / atuadores PZT na placa de alumínio.

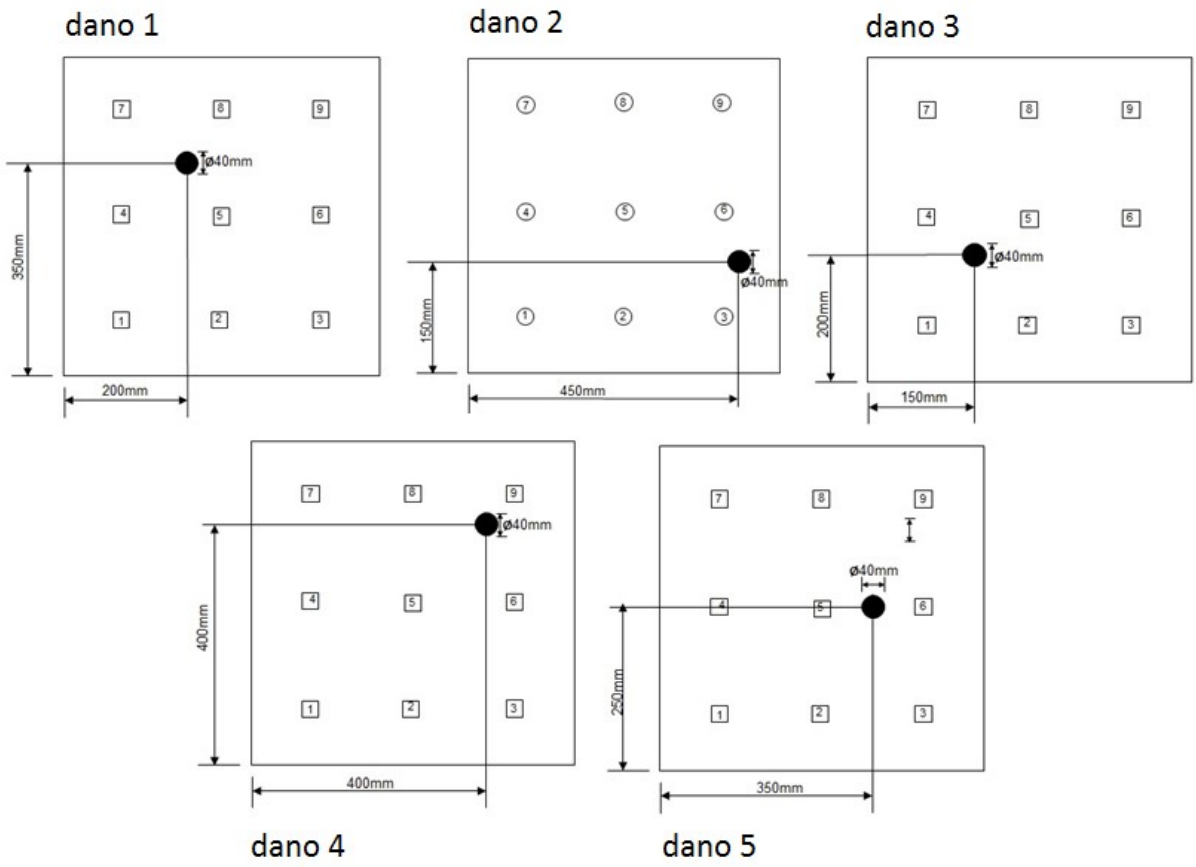
Figura 32 - Configuração dos sensores / atuadores piezelétricos na placa de alumínio.



Fonte: Rosa (2016)

Cada atuador gera sinais em oito caminhos, perfazendo um total de 72 caminhos. O mesmo procedimento deve ser realizado na placa que apresenta dano estrutural. Foram simulados cinco danos situados em posições diferentes na placa. Rosa (2016) realizou medidas em diferentes temperaturas, utilizando uma câmara com controle de umidade e temperatura, e em diferentes frequências: 200, 250, 300, 350 e 400 kHz. O objetivo foi, exatamente, criar um banco de dados que pudesse ser utilizado por outros membros do grupo. A figura 33 mostra a configuração da placa com os respectivos pontos onde foram implantadas as falhas estruturais. As falhas foram incorporadas na estrutura através da colagem de massas. Cada ponto que representa um dano na placa foi feito por 15 gramas de uma massa adesiva Hexcel.

Figura 33 -Configuração dos sensores / atuadores piezelétricos na placa de alumínio simulando danos estruturais em posições distintas.

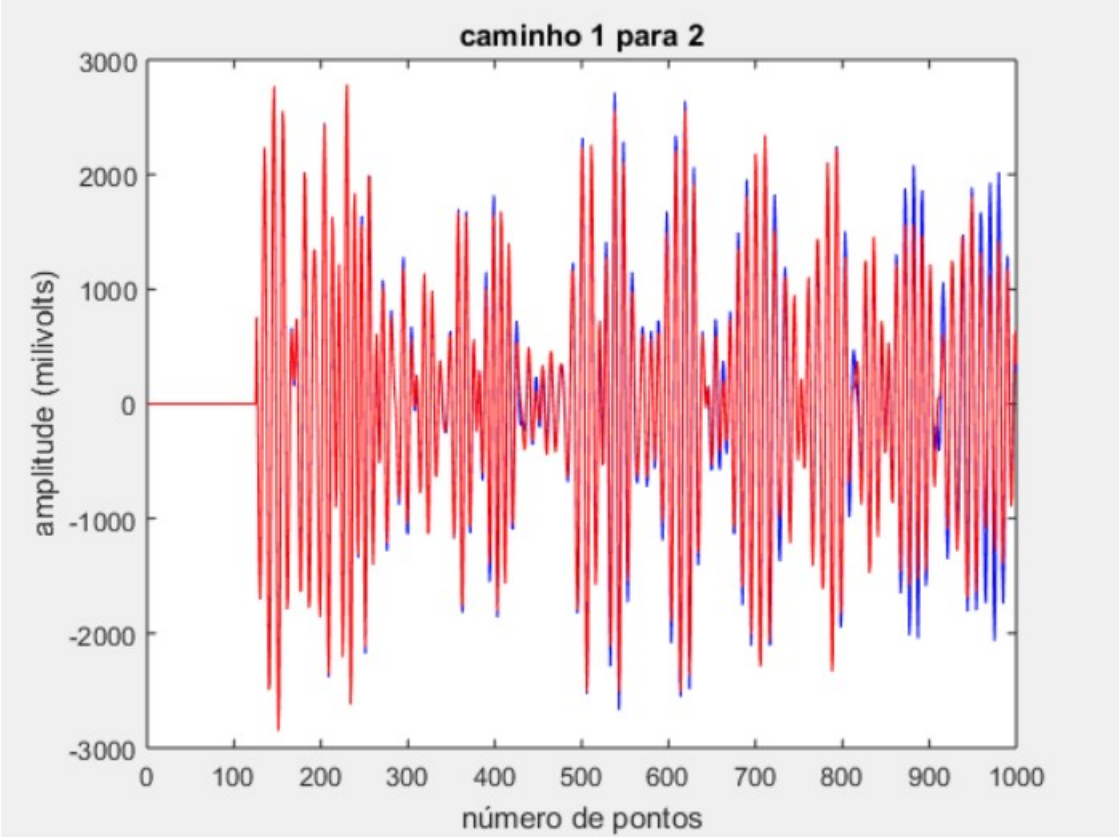


Fonte: Rosa (2016). (Adaptado pela autora)

Cada sinal gerado pelo caminho atuador-sensor é constituído por 1000 pontos. Depois de obter estes sinais, estes foram plotados em gráficos para uma melhor compreensão. Em um mesmo gráfico foi plotado o sinal representando uma estrutura sem danos (*baseline*) juntamente com o sinal obtido com um dano estrutural. Assim, fica mais fácil compreender como o sinal se modifica quando existe uma falha estrutural. Para facilitar o entendimento, foi usada a seguinte nomenclatura para definir um caminho: Pij (atuador i e sensor j). Assim, o caminho do atuador 1 para o sensor 2, é definido P12. A figura 34 apresenta a plotagem de um sinal sem defeito (azul) e um sinal com defeito (vermelho) em um gráfico. O sinal representa o caminho do atuador 1 para o sensor 2 referente ao dano 1, para a frequência de 300 kHz. Embora tenha sido realizado medidas em diferentes frequências, neste trabalho foi apenas utilizado a excitação *burst*5com frequência de 300 kHz. Para esse tipo de monitoramento (ondas de Lamb) esse tipo de sinal é o mais usado. Geralmente utilizam-se sinais de fácil

análise. O *burst* 5 é um sinal com cinco ciclos oriundo da multiplicação de dois sinais: uma senóide com amplitude de 35 milivolts janelado com um filtro de *hamming*. Foi usado uma frequência de amostragem de 3mHz e 1000 pontos para representar cada sinal.Para clareza do texto e figuras, nos próximos gráficos não será especificada a excitação e nem a frequência.

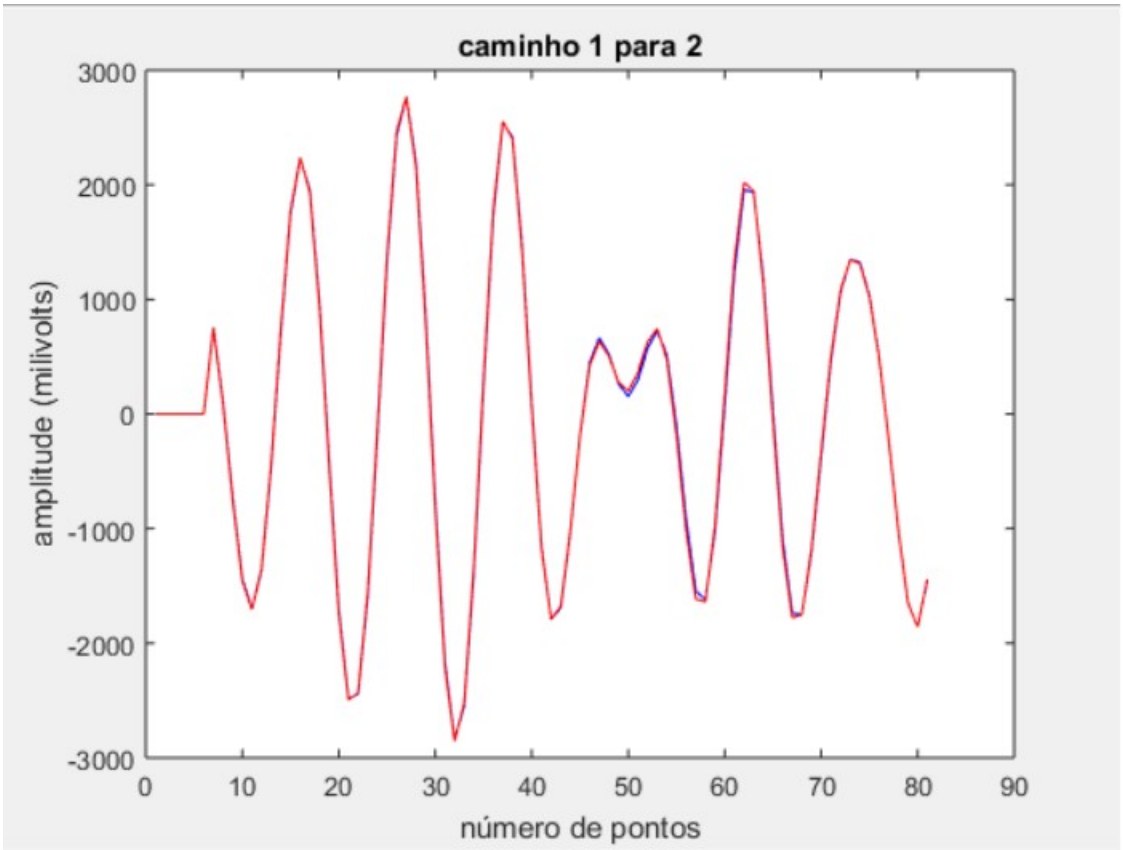
Figura 34 - Sinal completo (1000 pontos) obtido no caminho P12, referente ao dano 1, para excitação *burst* 5 com 300 kHz.



Fonte: própria autora.

Como se observa na figura 34, a variação do sinal é muito pequena no início, pois o dano esta longe deste caminho. Porém, quando o sinal é emitido de um atuador ocorre retorno e reflexões das ondas. Dessa forma, trabalhar com os 1000 pontos não é algo compensatório, visto que aumenta a complexidade do sinal, devido às ondas refletidas e aparição de modos de ondas mais elevados. Para realizar o monitoramento estrutural utilizando este tipo de sinal, a maioria dos pesquisadores utiliza apenas o pacote inicial do sinal, que são os primeiros picos do sinal. Portanto, foi realizada uma segmentação no sinal gerado (em torno de 80 pontos iniciais). A figura 35 mostra o gráfico gerado pelos primeiros pontos do sinal referente ao caminho 1 para 2 referente ao dano 1.

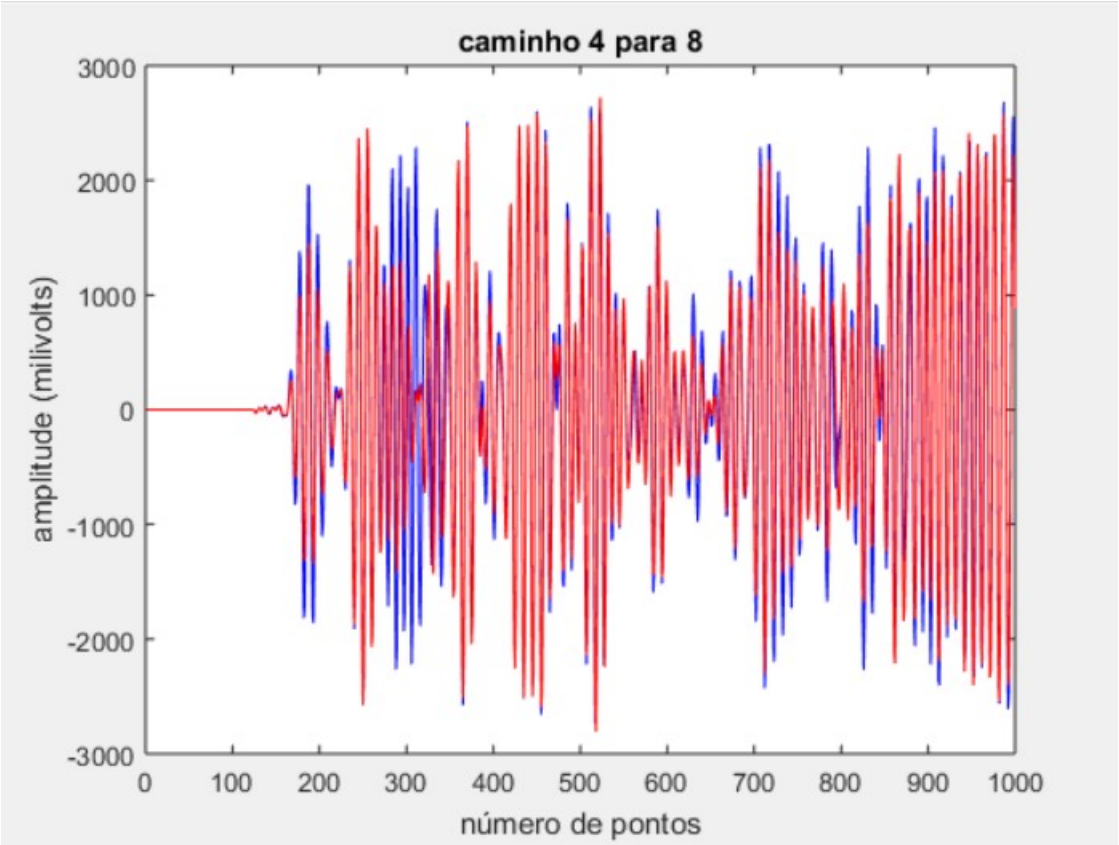
Figura 35- Sinal do caminho P12 nos 80 pontos iniciais.



Fonte: própria autora.

Para promover o monitoramento inteligente é necessário comparar um sinal com defeito com um sem defeito. A diferença entre estes sinais vai resultar nas futuras métricas que servirão para determinar um modelo de classificação no algoritmo de AM. Ainda sem usar IA, apenas usando a observação, pode-se constatar que a comparação da figura 34 representa uma situação sem defeito. A figura 36 mostra o sinal obtido no caminho P48, isto é, atuador 4 e sensor 8, nas mesmas condições da figura anterior. Note que a diferença entre os sinais é mais acentuada, pois a medida está próxima ao dano.

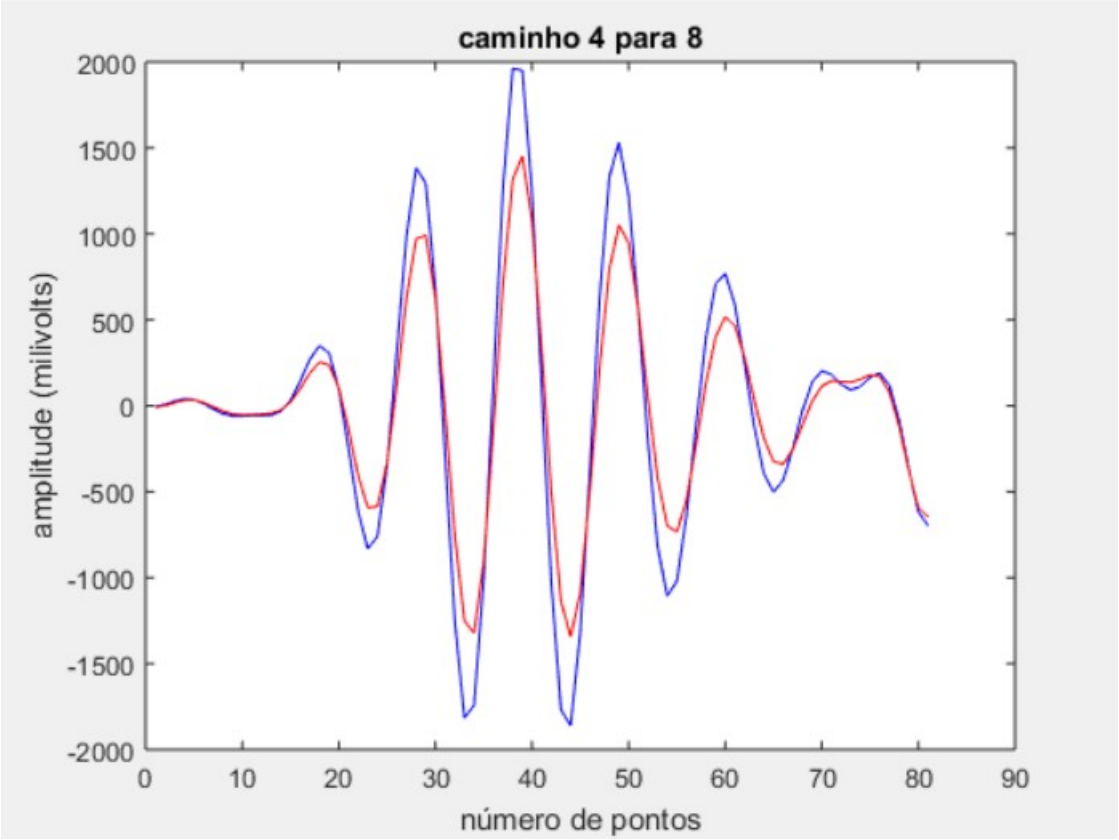
Figura 36- Sinal completo (1000 pontos) obtido no caminho P48, referente ao dano 1.



Fonte: própria autora.

Foi realizado o mesmo processo de segmentação do sinal nos primeiros oitenta pontos iniciais. A figura 37 apresenta o resultado dessa segmentação. Esta imagem demonstra características bem distintas em relação à figura 35. Nesta, ocorre um distanciamento bastante visível entre os sinais. A altura dos picos apresentados na linha azul são maiores que as demonstradas na linha vermelha. Através de uma comparação visual entre a figura 35 e a 37 é nítida a percepção que se tratam de sinais que representam situações adversas.

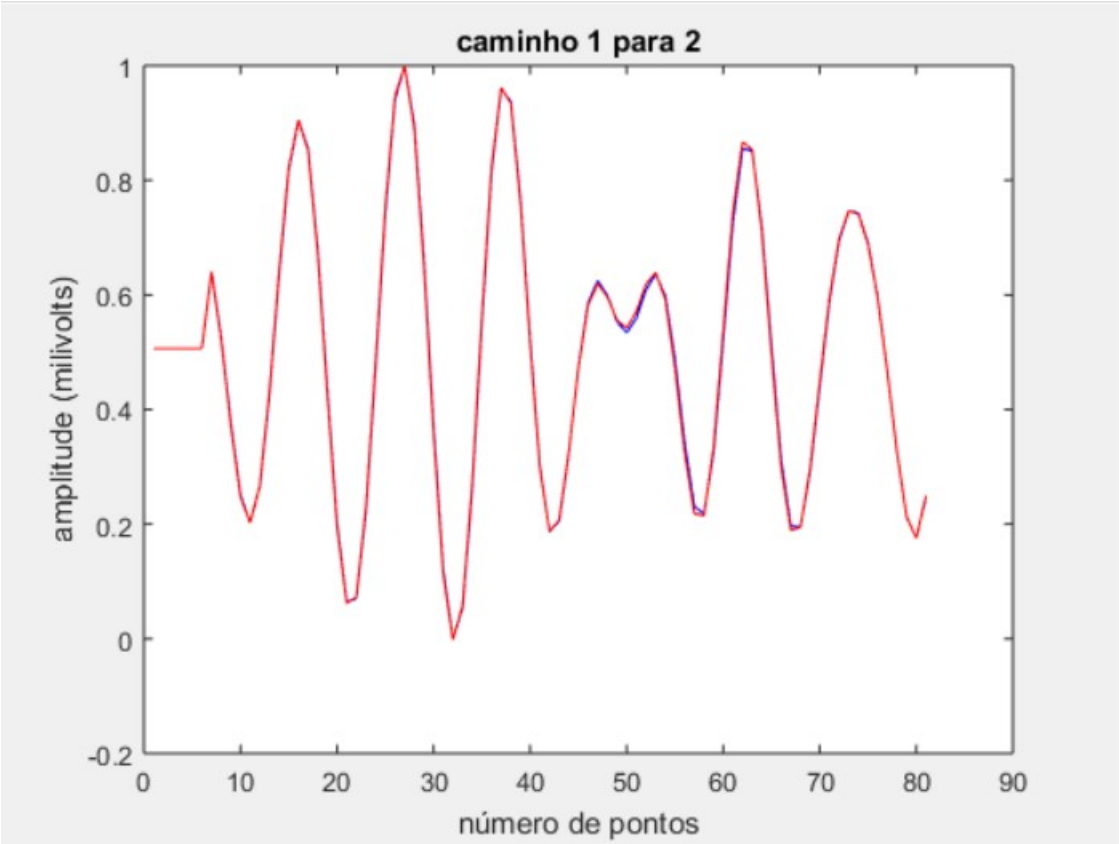
Figura 37 - Sinal do caminho P48, referente ao dano 1, nos 80 pontos iniciais.



Fonte: própria autora.

Este processo de segmentação foi feito com todos os sinais obtidos de todos os nove atuadores (72 caminhos). Após este processo, a fase seguinte foi realizar a normalização dos dados numa escala compreendida entre 0 e 1. Como neste processo basta analisar a diferença entre os sinais, foi feita a normalização acima para que os dados utilizados em um teste fossem aproveitados em outras situações de falhas. Portanto, o programa ficou geral e não é necessário o treinamento para todas as situações de danos. A figura 38 apresenta o mesmo segmento do caminho P12 normalizado.

Figura 38- Segmentação do sinal obtido no caminho P12 nos 80 pontos iniciais.



Fonte: própria autora.

Todo o processo de segmentação e normalização foi realizado para todas as medidas e em todas as situações de danos. O processo foi feito usando o *software* MATLAB. Foi preferível usar a normalização para melhor compreensão dos dados e obter métricas em uma escala mais favorável para a futura utilização no algoritmo. Como já foi dito, o processo de identificação de falhas deve ser realizada comparando os dois sinais (com defeito e sem defeito). Esta equiparação foi feita através dos cálculos de alguns índices já muito usados no SHM. Desta forma, o conjunto de métricas vai definir um objeto, representando uma falha ou não. Esses índices são utilizados, pois para o algoritmo de AM é muito mais significativo na geração do modelo, o uso de atributos qualitativos do que quantitativos dos dados.

O próximo passo foi construir as rotinas para calcular as métricas. Foram calculadas sete métricas: a diferença obtida pela comparação do cálculo do valor médio eficaz (RMS), uma variação do Root Mean Square Deviation (RMSD), o Correlation Coefficient Deviation Metric (CCDM), a distância euclidiana, a similaridade de cossenos, o índice de variação das normas H2 e a Hinfinito. A métrica relacionada com o RMS foi obtida da seguinte forma:

$$M_1 = RMS_d - RMS_b \tag{28}$$

sendo M_1a métrica provida pela subtração do valor médio eficaz de cada sinal, RMS_d o valor médio eficaz do sinal com defeito (ou atual) e RMS_b o valor médio eficaz do sinalíntegro.

A segunda métrica refere-se a uma variação do cálculo do desvio do valor médio eficaz, foi calculada da seguinte maneira:

$$M_2 = \left(\frac{\sum_{i=1}^{n}(S_d(i)-S_b(i))^2}{n}\right)^{\frac{1}{2}} \tag{29}$$

na qual M_2 é uma variação do desvio do valor médio eficaz, $S_d(i)$ representa cada ponto que compõe o sinal com defeito, $S_b(i)$ representa cada ponto que compõe o sinal íntegro e n onúmero de pontos.Desta forma, o resultado da métrica da variação do RMSD pode ser calculado inicialmente pela somatória da subtração de cada ponto do sinal que apresenta falha com o sinal íntegro elevado ao quadrado. Depois é realizada uma divisão entre a somatória obtida com o número total de pontos e finalmente é efetuada a raiz quadrada.

A terceira métrica consiste no cálculo do CCDM cujo valor pode ser definido pela seguinte equação:

$$M_3 = 1 - \left(\frac{S_d \otimes S_b}{\sigma_d . \sigma_b}\right) \tag{30}$$

cuja variável M_3 representa o índice de correlação entre os sinais, S_d simboliza o sinal com defeito, S_b simboliza o sinal íntegro, σ_d representa o desvio padrão do sinal com defeito e σ_b representa o desvio padrão do sinal íntegro. A correlação entre os sinais indica o grau de similaridade presente e seu valor varia dentro do intervalo [0,1]. Portanto, quanto mais próximo de um, mais equivalente é um sinal do outro. Nesta métrica, optou-se pela subtração do escalar um, para que todas as métricas tivessem a mesma tendência de variação para as estruturas com danos. Neste caso, o resultado obtido pela divisão da convolução entre os sinais pela multiplicação de seus desvios padrões é subtraído de um. Como isto, se o valor do resultado for próximo de zero significa sem dano, caso contrário, significa com dano.

O outro cálculo a ser efetuado é a distância euclidiana, índice muito usado para determinar o grau de dissimilaridade entre sinais. Este valor pode ser obtido pela raiz quadrada da somatória efetuada pela elevação do resultado da subtração de cada ponto dos sinais, conforme mostrado na equação abaixo:

$$M_4 = (\sum_{i=1}^{n}(S_d(i) - S_b(i))^2)^{\frac{1}{2}} \quad (31)$$

sendo M_4 a métrica que contém o valor da distância Euclidiana dos sinais, $S_d(i)$ cada ponto do sinal com defeito, $S_b(i)$ cada ponto do sinal sem defeito. Quanto maior o valor da distância, mais heterogêneos são os sinais.

Outro atributo também utilizado foi a similaridade de cossenos. Este valor pode ser obtido pelo produto interno entre cada ponto que constitui os sinais divididos pela norma de cada sinal. Sua formulação é definida como:

$$M_5 = 1 - \left(\sum_{i=1}^{n} \frac{S_d.S_b}{\|S_d\| - \|S_b\|}\right) \quad (32)$$

na qual M_5 é a métrica que obtém o resultado do índice de similaridade, $S_d.S_b$ representa o produto interno entre os sinais com defeito e íntegro, $\|S_d\|$ a norma do sinal com defeito e $\|S_b\|$ a norma do sinal íntegro. Para seguir o mesmo raciocínio usado no cálculo do CCDM, o resultado obtido foi subtraído da unidade, pois assim cada valor que estiver próximo de zero, significa que são mais equivalentes, o contrário representa o quão distantes estão os sinais entre si.

A sexta métrica consiste no cálculo da norma H2 que representa o valor da área sob a curva do sinal (tanto a parte positiva como a negativa). A equação abaixo representa este procedimento:

$$M_6 = (H_{S_d}^2 - H_{S_b}^2)/H_{S_b}^2 \quad (33)$$

cujo valor de M_6 é o resultado obtido da subtração entre as normas dos sinais dividido com a norma do sinal íntegro, $H_{S_d}^2$ representa a norma do sinal com dano e $H_{S_b}^2$ representa a norma do sinal íntegro.

Por fim, a obtenção da última métrica consiste no cálculo da norma Hinfinito que representa o valor de pico do sinal. Foi realizado o mesmo processo que foi efetuado no cálculo da norma H2. Calculou-se a norma Hinfinito de cada sinal e subseqüente realizou-se a subtração. Como mostrado na equação abaixo:

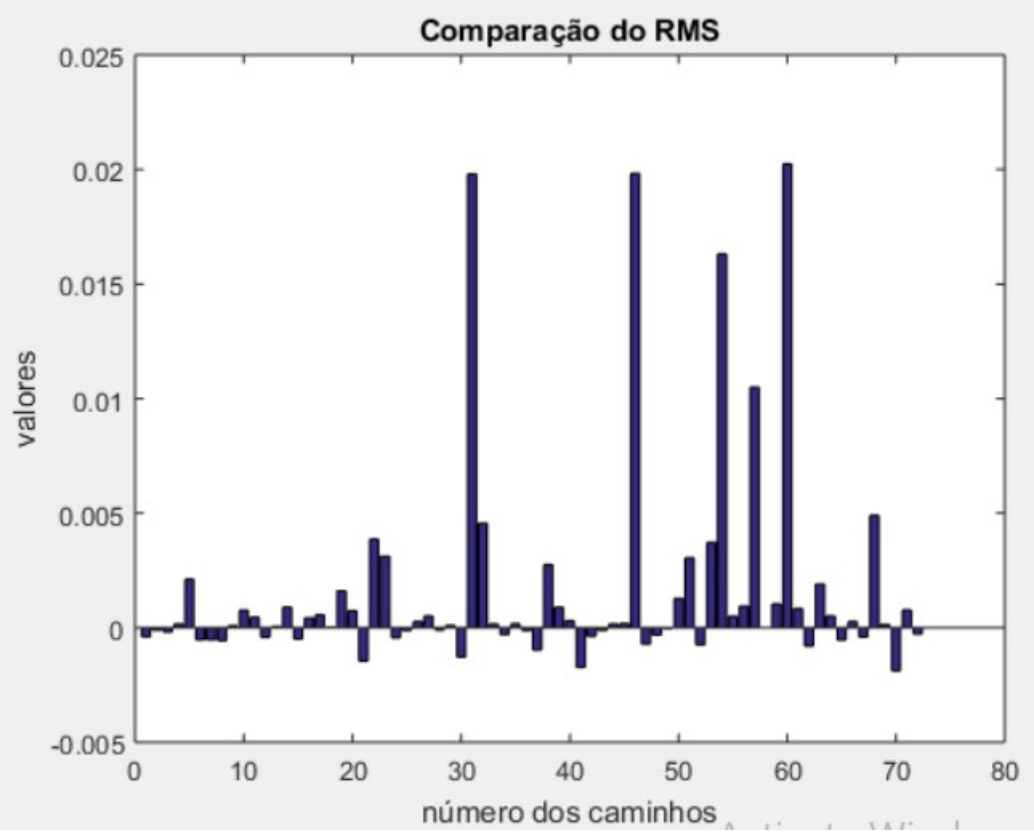
$$M_7 = (H_{s_d}^{\infty} - H_{S_b}^{\infty})/H_{S_b}^{\infty} \quad (34)$$

sendo que a variável M_7 corresponde ao valor obtido da subtração entre as normas dos sinais dividido com a norma do sinal íntegro, H_{sd}^{∞} representa a norma Hinfinito do sinal com defeito e $H_{S_b}^{\infty}$ simboliza a norma Hinfinito do sinal íntegro.

Portanto, considera-se a utilização de sete métricas para definir o estado da estrutura. É importante destacar que não foi feito uma análise das melhores métricas, pois este não é o objetivo do presente trabalho.

Todo processo de obtenção das métricas foi realizado também nas outras situações de danos. Portanto, para cada monitoramento realizado na placa de alumínio simulando danos em posições distintas, obteve-se uma tabela de dados com 72 objetos, cada um representando um caminho de propagação entre um atuador e um sensor. Desta forma, foram obtidas cinco tabelas para a representação dos danos 1, 2, 3, 4 e 5. No apêndice B é possível analisar a tabela completa das sete métricas obtidas referente à configuração do dano 1. Com o intuito de obter uma melhor visualização dos dados, cada coluna da tabela foi plotada em um gráfico de barras, como mostra a figura 39.

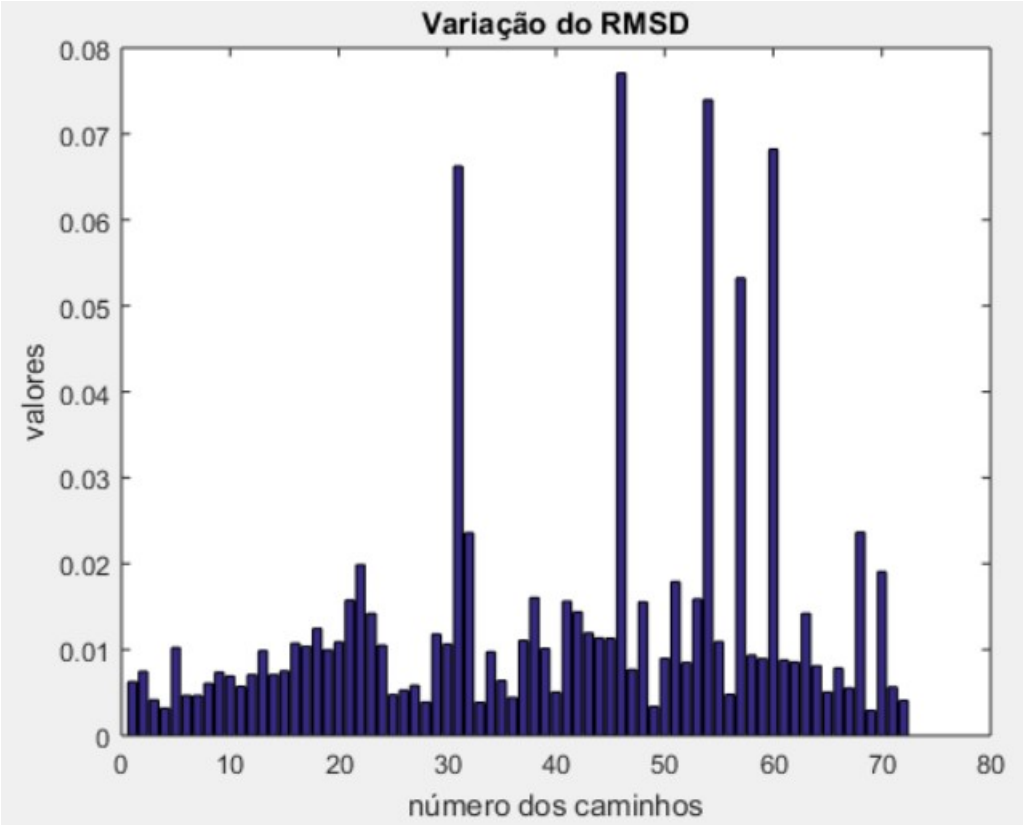
Figura 39-Plotagem da métrica comparação do RMS para o dano 1.



Fonte: própria autora.

Conforme pode ser visto na figura 39, ocorre variações nos valores das métricas. As maiores variações estão nos elementos 31, 46, 54, 57 e 60. Estes elementos correspondem aos caminhos P4-8, P6-7, P7-6, P8-1 e P8-4. Constata-se através da figura 38, que estes caminhos são os que cruzam a posição do dano 1. A figura 40 mostra o mesmo tipo de gráfico para a métrica RMSD.

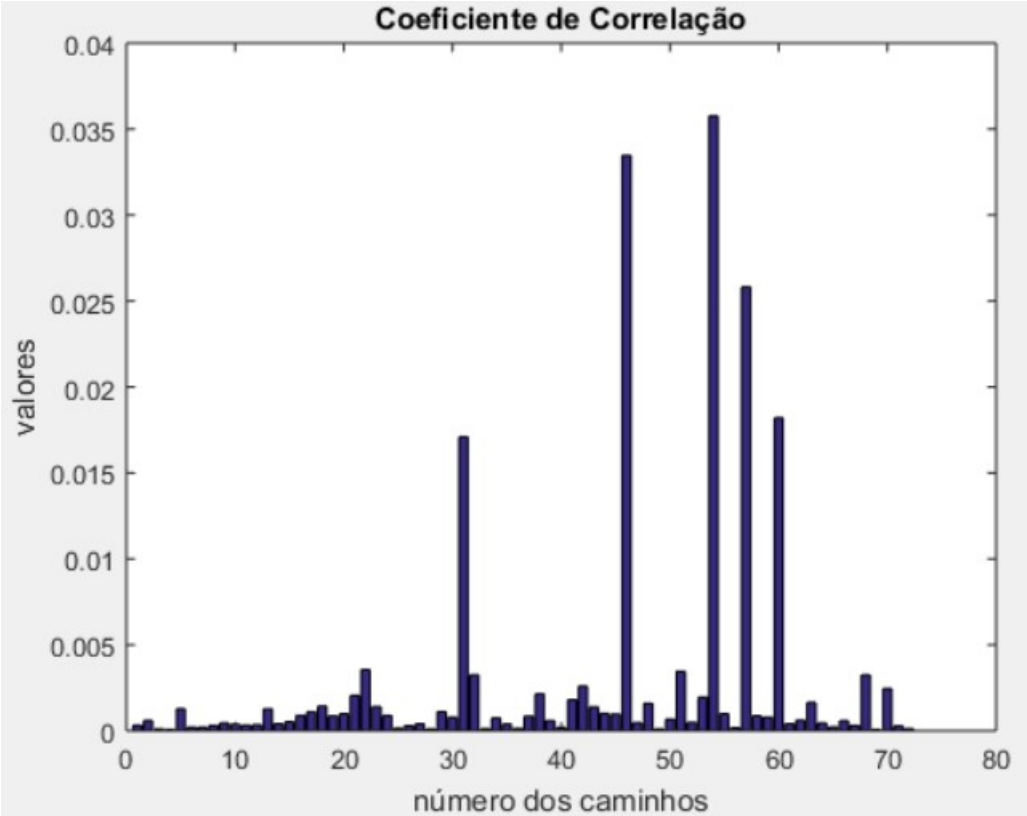
Figura 40 -Plotagem obtida através da comparação do RMSD dos dados obtidos no monitoramento referente ao dano 1.



Fonte: própria autora.

Analisando a figura 40 é observado que os mesmos objetos destacados no gráfico anterior também apresentam o mesmo distanciamento em relação aos demais. A figura 41 o grafico de barras para a métrica do coeficiente de correlação.

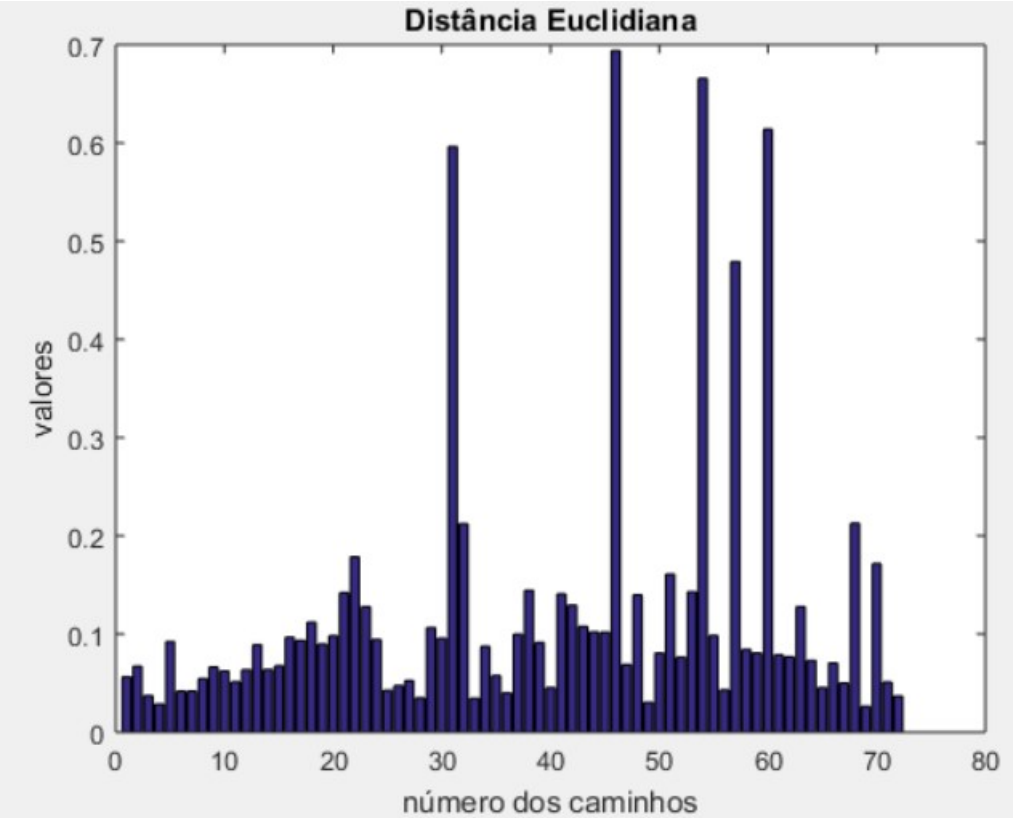
Figura 41-Plotagem da métrica obtida do cálculo CCDM dos dados obtidos no monitoramento referente ao dano 1.



Fonte: própria autora.

Verificando a figura 41 é nítido a dispersão apresentada pelos objetos 31, 46, 54, 57 e 60 em relação aos demais. A figura 42 os valores da métrica distância euclidiana.

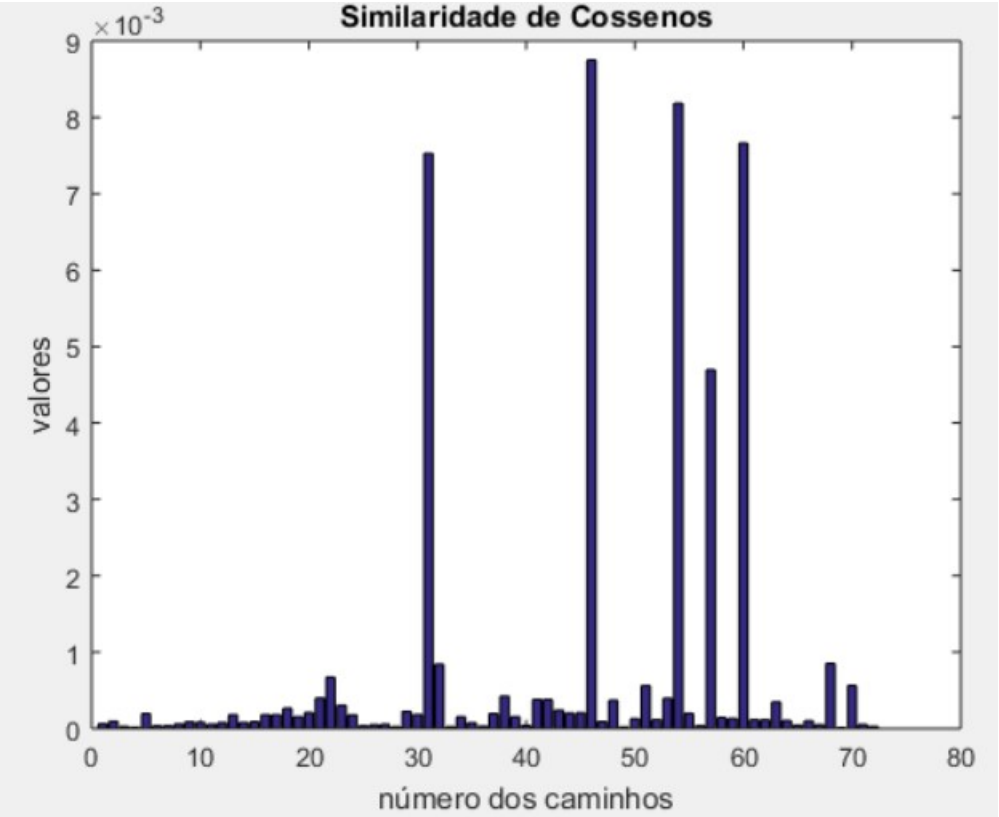
Figura 42 -Plotagem da métrica obtida do cálculo da distância euclidiana dos dados obtidos no monitoramento referente ao dano 1.



Fonte: própria autora.

A figura 42 confirma as variações presentes entre os dados. A figura 43 mostra a métrica baseada na similaridade dos sinais em relação ao cosseno.

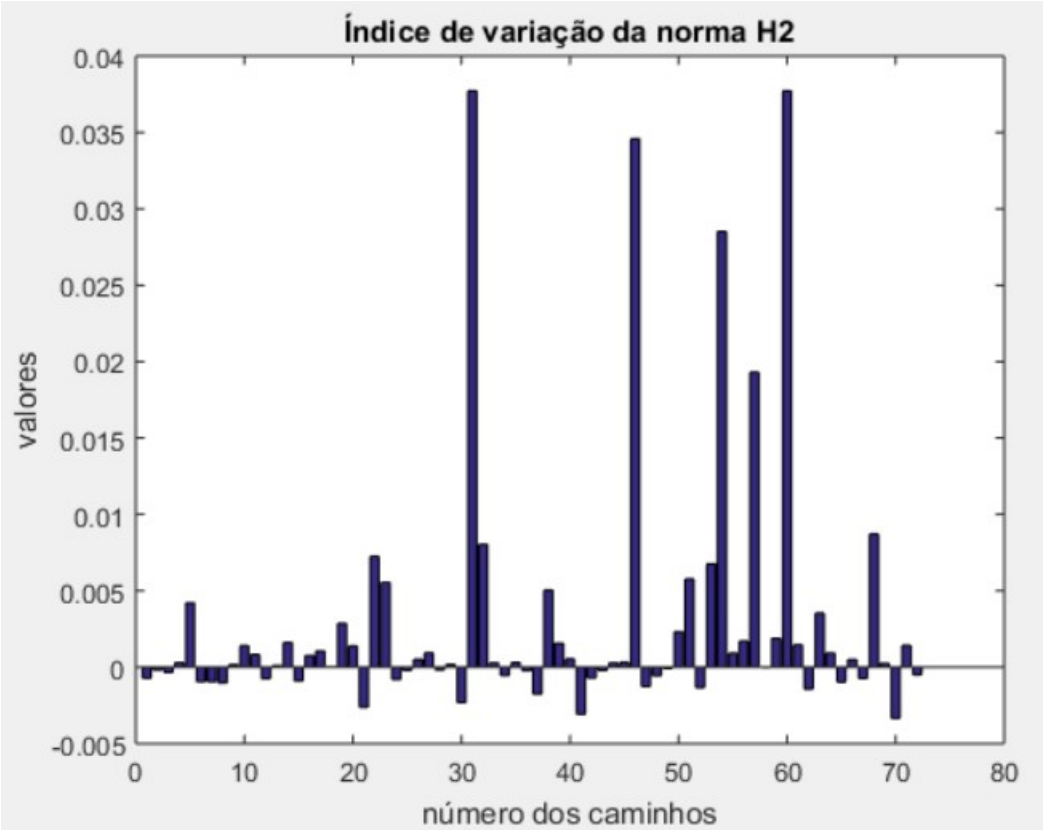
Figura 43 -Plotagem da métrica obtida do cálculo da similaridade de cossenos dos dados obtidos no monitoramento referente ao dano 1.



Fonte: própria autora.

A figura 44 mostra a disposição dos dados em relação ao cálculo da norma H2, que representa a área do sinal sob a curva. Novamente, os dados 31, 46, 54, 57 e 60 apresentam os maiores índices.

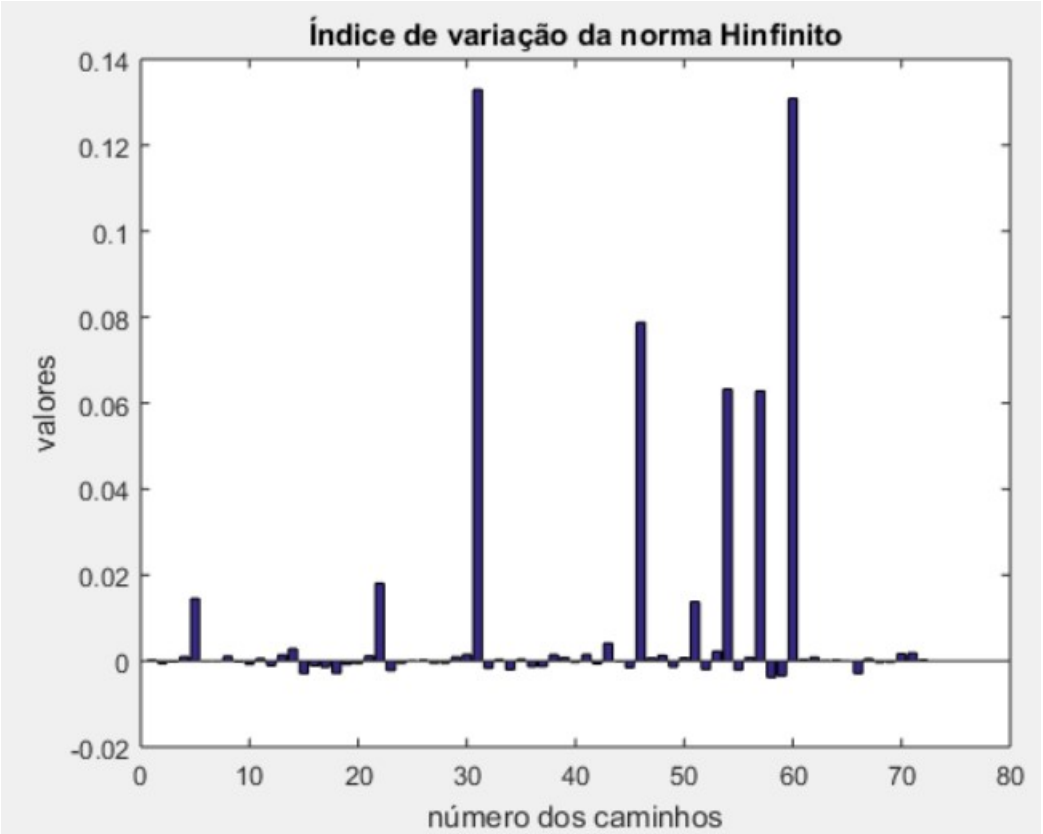
Figura 44 -Plotagem da métrica obtida através do cálculo das normas H2 dos dados obtidos no monitoramento referente ao dano 1.



Fonte: própria autora.

A figura 45 apresenta a distribuição dos elementos em relação ao cálculo da norma Hinfinito mostrando as variações entre os picos do sinal. Os dados 31, 46, 54, 57 e 60 apresentam os maiores índices.

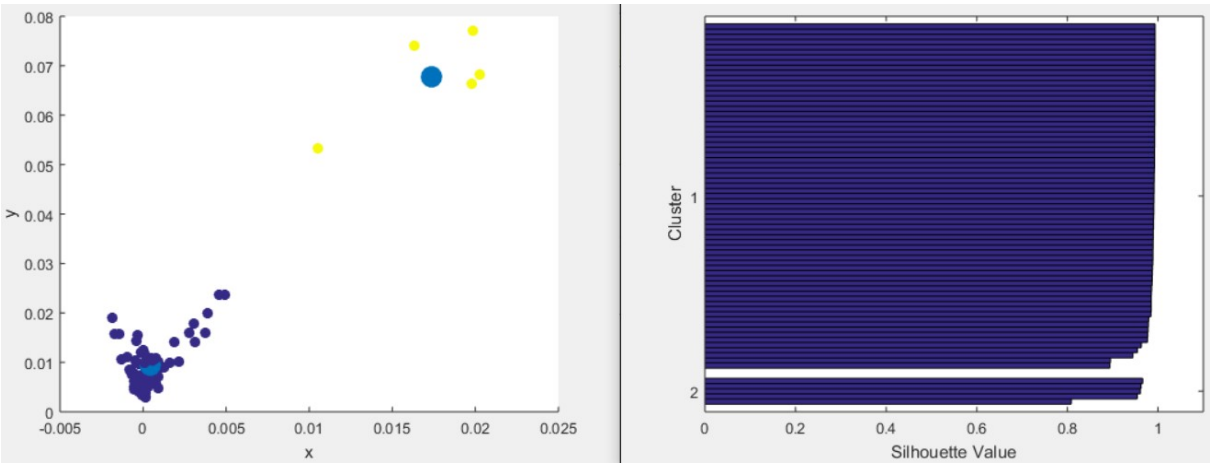
Figura 45 -Plotagem da métrica obtida através do cálculo das normas Hinfinito dos dados obtidos no monitoramento referente ao dano 1.



Fonte: própria autora.

Todo o processo de plotagem de gráficos foi realizado com os demais conjuntos de dados obtidos, ou seja, foram realizados também no conjunto de dados que compõem os danos 2, 3, 4 e 5. Após este processo de análise nos dados, o passo seguinte foi definir quais grupos representariam os dados com defeito e ausentes de defeitos. Apesar dos gráficos apresentarem um direcionamento visual da dispersão dos elementos, foi usado um algoritmo de clusterização (recurso disponível no MATLAB) para auxiliar neste processo de separação dos dados em dois grupos. Um algoritmo de *cluster* é comumente usado quando se possui um conjunto de dados na qual ainda não se sabe a qual rótulo este pertence. Portanto, sua finalidade é analisar a semelhança entre os dados e agrupá-los de acordo com as equivalências encontradas. Desta maneira, formando grupos distintos, chamados habitualmente de *clusters*. A figura 46 apresenta os *clusters* criados usando como referência o conjunto de dados do dano 1.

Figura 46-*Clusters* criados a partir das métricas calculadas através dos dados obtidos no monitoramento referente ao dano 1.



Fonte: própria autora.

A figura 46 reafirma o que foi observado nos gráficos. O *cluster* separou os dados em dois grupos: *cluster*1 (íntegros) e *cluster* 2 (com danos). Os dois pontos maiores em azul claro na figura representam o centro de cada *cluster*. Apenas cinco elementos representam os danos (em amarelo), são os elementos: 31, 46, 54, 57 e 60. Desta forma, os pontos em azul escuro, representam os dados sem danos. O mesmo procedimento de clusterização foi realizado nos demais conjuntos de dados (dano 2, dano 3, dano 4 e dano 5). Com a rotulação efetuada pelo algoritmo de clusterização, a etapa seguinte foi adequar o rótulo e as métricas obtidas no formato que o algoritmo possa compreender. Esse processo foi realizado manualmente. Tabela 7 apresenta uma parte do arquivo formatado. O arquivo completo pode ser visualizado no Apêndice C.

Tabela 8 - Arquivo no formato especificado pela biblioteca LibSVM (continua).

1	1:0.000183375670573471	2:0.0113077432631623	3:0.000960616353512189
2	1:0.0198302218097122	2:0.0770959209433578	3:0.0334702584663725
1	1:-0.000713525091795453	2:0.00766127559770839	3:0.000465658198318786
1	1:-0.000320776027660785	2:0.0155435616141405	3:0.00158935989132836
1	1:-0.0000412041226014459	2:0.00337525193605097	3:0.000105505152250385
1	1:0.00127942286165095	2:0.00896974914640573	3:0.000663904266136828
1	1:0.00304930807292014	2:0.0178975489956353	3:0.00346096599723433
1	1:-0.000747863218412248	2:0.00848575434392716	3:0.000478285643952248
1	1:0.00372682204076702	2:0.0158827533660089	3:0.00195269876561677
2	1:0.0163214732947837	2:0.0739710207112087	3:0.0357771421038767

Fonte: própria autora.

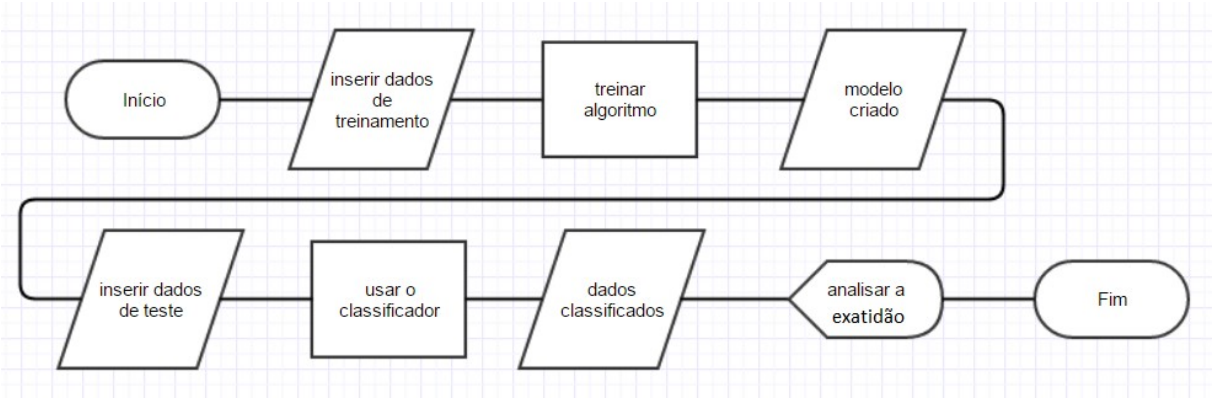
Tabela 9 - Arquivo no formato especificado pela biblioteca LibSVM (conclusão).

1	1:0.000500998179507439	2:0.010929528907388	3:0.000985749311080975
1	1:0.000932320139524934	2:0.00478791319001943	3:0.000168100556943496
2	1:0.0104975956035542	2:0.0532416833786955	3:0.0258177253130112
1	1:-0.00000667654836539189	2:0.00935585396500935	3:0.000870377486030249
1	1:0.00103161851196543	2:0.00894370037813598	3:0.00078420795713352
2	1:0.0202388920120772	2:0.0682174992024282	3:0.0182084736637049
1	1:0.000842457292377974	2:0.00876624470080958	3:0.000415654742085492

Fonte: própria autora.

Este procedimento de formatação foi realizado também nos outros conjuntos de dados (dano 2, dano 3, dano 4 e dano 5). Cada conjunto de dados, é formando por 72 linhas e por 8 colunas. Cada linha representa um objeto. Este objeto é caracterizado por 8 atributos. O primeiro atributo é o rótulo, ou seja, este objeto simboliza um dano (valor 2) ou a ausência de dano (valor 1). As demais métricas ou atributos são as características que formam o objeto.Cada atributo recebe um índice de identificação. Neste exemplo, as características são: o valor da comparação do RMS, a variação do RMSD, o índice CCDM, a distância euclidiana, o índice de similaridade de cossenos, a diferença da norma H2 e Hinfinito. Portanto, a segunda etapa do projeto encontra-se concluída. Dessa forma, pode-se seguir a próxima fase do trabalho. A figura 47 apresenta as etapas necessárias para realizar o processo 3 do fluxograma geral da figura 25. Estas fases englobam deste a inserção dos dados já pré-processados até a análise de classificação do algoritmo.

Figura 47-Sequência de passos abordados pelo processo 3 do fluxograma geral.



Fonte: própria autora.

Inicialmente foi criado um arquivo denominado *dataset_01.txt* para ser usado no treinamento. Esse arquivo foi formado através de uma seleção de objetos que representam os danos nos conjuntos de dados 1, 2, 3, 4 e 5. Depois foram selecionados aleatoriamente os dados que representam a estrutura íntegra. No final, este arquivo possui 76 dados, sendo 26 que representam os danos e 50 que representam a estrutura íntegra. Foi usado o *kernel* RBF com os valores de *C = 2048 e gamma = 8*. Estes valores foram empregados após executar o arquivo *grid.py* conforme já foi explanado sua finalidade (consultar página 72). Depois de criado o modelo, o classificador gerado foi testado nos demais conjuntos de testes, que consistem nos demais *datasets*: dano 1, 2, 3, 4 e 5. A tabela 8 mostra a configuração usada na geração do modelo e os resultados da classificação obtidos para cada conjunto de dado.

Tabela 10 - Configuração e resultado do treinamento e classificação do *dataset_01.txt.*

DATASET: *dataset_01.txt*	
SVM: C_SVC	kernel: RBF
PARÂMETROS	
C: 2048	degree: 3
gamma: 8.0	probability: 0
eps: 0.001	coefficient: 0
nu: 0.5	cache-size: 100
number weight: 0	p: 0.1
shrinking: 1	weight: 0
weight label: 0	
SAÍDA DO TREINAMENTO	
nu:0.032909094902080306	obj: -3149.062453882274
rho: -0.21185432130360995	nSV: 5
nBSV: 1	Total nSV: 5
CLASSIFICAÇÃO	**TAXA DE EXATIDÃO**
Conjunto de dados (dano 1)	100%
Conjunto de dados (dano 2)	100%
Conjunto de dados (dano 3)	100%
Conjunto de dados (dano 4)	100%
Conjunto de dados (dano 5)	100%

Fonte: própria autora.

Conforme pode ser visto na tabela 8, o modelo gerado a partir do conjunto de dados *dataset_01.txt* conseguiu classificar 100% todos os conjuntos de dados de testes. O algoritmo classificou corretamente as 72 instâncias de cada conjunto de testes. Portanto, tratou-se de um modelo consistente e eficaz.

Foi realizado um segundo teste, um arquivo de treinamento formado com os dados contidos nos conjuntos de dados 1, 2 e 3. Portanto, totalizou-se um *dataset* com 12 danos e 48 sem danos. Este conjunto de dados foi treinado usando o *kernel* RBF com o termo *C = 512* e *gamma = 0.5.* Em seguida, este modelo foi testado nos conjuntos de dados: 1, 2, 3, 4 e 5. A tabela 9mostra os resultados obtidos da classificação.

Tabela 11 - Resultados obtidos usando um modelo construído a partir dos conjuntos de dados 1, 2 e 3.

DATASET	TAXA DE ACURÁCIA
Conjunto de dados (dano 1)	100%
Conjunto de dados (dano 2)	100%
Conjunto de dados (dano 3)	100%
Conjunto de dados (dano 4)	~98.6%
Conjunto de dados (dano 5)	~98.6%

Fonte: própria autora.

Depois foi realizado um terceiro teste. Neste o arquivo de treinamento foi formado com os dados contidos nos conjuntos de dados 1,2 e 4. Portanto, totalizou-se um *dataset* com 16 danos e 48 sem danos. Este conjunto de dados foi treinado usando o *kernel* RBF com o termo *C = 8* e *gamma = 0.5.* Em seguida, este modelo foi testado nos conjuntos de dados: 1, 2, 3, 4 e 5. A tabela 10 mostra os resultados obtidos da classificação.

Tabela 12 - Resultados obtidos usando um modelo construído a partir dos conjuntos de dados 1, 2 e 4 (continua).

DATASET	TAXA DE EXATIDÃO
Conjunto de dados (dano 1)	100%
Conjunto de dados (dano 2)	~97.2%
Conjunto de dados (dano 3)	100%

Fonte: própria autora.

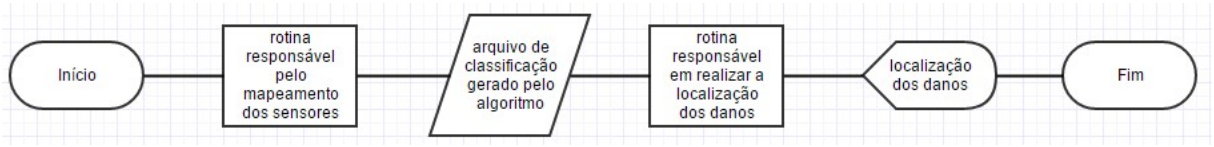
Tabela 13 - Resultados obtidos usando um modelo construído a partir dos conjuntos de dados 1, 2 e 4 (conclusão).

DATASET	TAXA DE EXATIDÃO
Conjunto de dados (dano 4)	100%
Conjunto de dados (dano 5)	100%

Fonte: própria autora.

A etapa seguinte consiste em utilizar estas classificações efetuadas pelo algoritmo para localizar os danos na placa de alumínio. A figura 48 mostra as fases do processo 4 retratado na figura 25 do fluxograma geral.

Figura 48-Fluxograma detalhado do processo 4.



Fonte: própria autora.

Para realizar a etapa de localização de danos, foi criado um procedimento (método). Este método tem a finalidade de criar uma matriz de String, cuja cada célula contém o mapeamento do caminho do atuador/sensor. Como o procedimento de medidas é o *pitch-catch*, isto é o atuador envia e o sensor capta o sinal, os valores correspondentes a diagonal principal da matriz são vazios. O próximo passo foi elaborar um método que recebe dois parâmetros: a matriz de mapeamento e a classificação efetuada pelo algoritmo. Então, todo objeto que foi classificado como um dano presente, foi armazenada sua posição (de acordo com a disposição dos elementos contidos no conjunto de teste), assim foi possível encontrá-lo na matriz de mapeamento.

O capítulo seguinte mostra os resultados alcançados com a metodologia proposta, juntamente com as discussões.

4 RESULTADOS E DISCUSSÕES

As classificações realizadas pelos modelos gerados através dos arquivos de treino foram muito satisfatórias. Foi escolhido o terceiro teste, pois este apresentou os melhores resultados. Para obter uma avaliação mais consistente dos resultados foi construído as matrizes confusão da hipótese gerada. Esta técnica é interessante, pois mostra a quantidade de classificações corretas contra as incorretas. Então, para cada classificação será apresentada a matriz confusão e as métricas derivadas desta. A tabela 11 apresenta a matriz confusão juntamente com as métricas calculadas usando o conjunto de dados do dano 1 como teste.

Tabela 14- Matriz confusão e métricas derivadas para o teste usando o conjunto de dados dano 1.

Matriz Confusão	**Classe1 (sem danos)**	**Classe2 (com danos)**
Classe1 (sem danos)	TP: 67	FN: 0
Classe2 (com danos)	FP: 0	TN: 5
Outras métricas		
Taxa de erro – classe 1	0.0	
Taxa de erro – classe2	0.0	
Erro total	0.0	
Acurácia total	1.0	
Precisão	1.0	
Sensibilidade	1.0	
Especificidade	1.0	
Medida harmônica	1.0	
Confiabilidade positiva	1.0	
Confiabilidade negativa	1.0	
Suporte	~0.93	
Cobertura	~0.93	
Kappa	1.0	

Fonte: própria

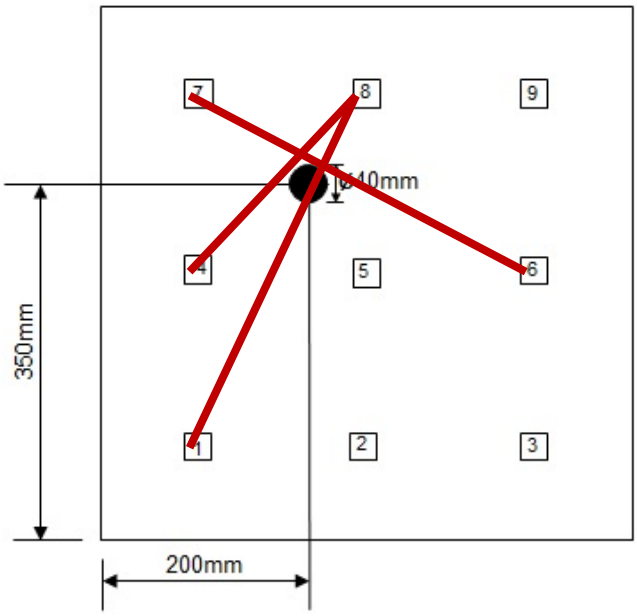
Os termos TP, TN, FP e FN significam verdadeiro positivo, verdadeiro negativo, falso positivo e falso negativo.

Na tabela 11é possível através da análise da diagonal principal da matriz notar que de

72 objetos, 67 foram classificados precisamente como pertencentes a classe 1 (sem dano) e 5 rotulados como classe 2 (com dano). Não houve nenhuma ocorrência de falsos positivos e nem falsos negativos. Outro ponto importante foi os valores da confiabilidade positiva, da sensibilidade e da média harmônica serem 1.0, correspondendo a 100 % de exatidão. A confiabilidade positiva corresponde ao número de verdadeiros positivos dividido pela somatória dos verdadeiros positivos com os falsos positivos. A sensibilidade consiste em dividir o número de verdadeiros positivos com a soma obtida entre os verdadeiros positivos com os falsos negativos. Para conseguir a média harmônica devem-se realizar três passos: primeiramente multiplica-se a confiabilidade positiva com a sensibilidade e depois por dois; a segunda etapa consiste em somar a confiabilidade positiva com a sensibilidade e por fim, dividi-se o resultado da etapa um com a dois.

Apesar da exatidão ter sido excelente, o fator relevante foi a localização dos danos na estrutura. Para o teste envolvendo o conjunto de dados 1 foram diagnosticados cinco caminhos: 4-8, 6-7,7-6, 8-1 e 8-4. A figura 49 mostra os caminhos identificados com dano na estrutura. O cruzamento destes caminhos determina a provável localização do dano.

Figura 49-Localização do dano na estrutura referente ao conjunto de dados com dano 1.

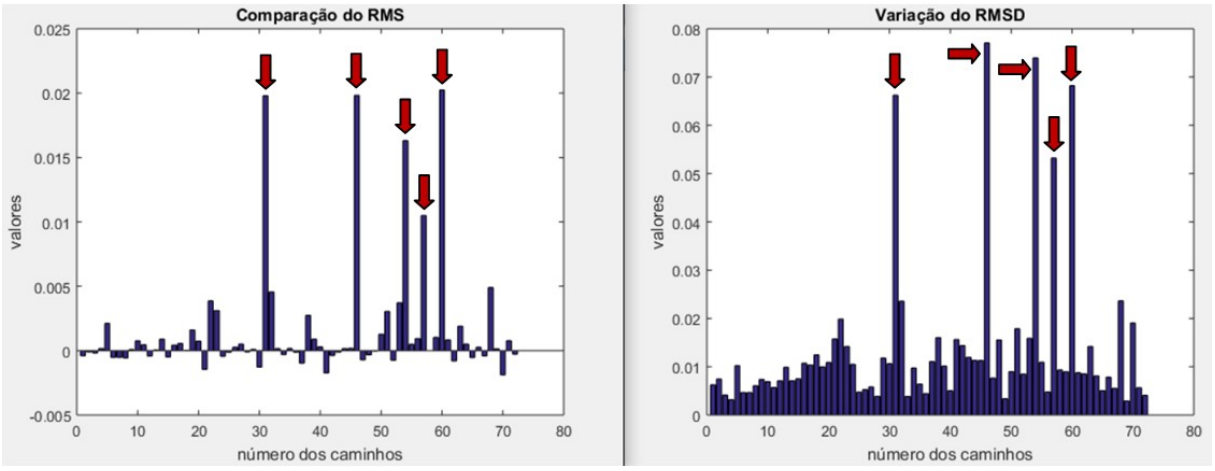


Fonte: própria autora.

Após a identificação das posições do dano na placa, é possível fazer uma comparação das localizações com os gráficos gerados a partir das métricas. Na figura 50, usando como referência as métricas comparação do RMS e variação do RMSD, através das

setas vermelhas são mostrados quais são os números de caminhos (objetos) que representam os caminhos diagnosticados pelas rotinas criadas para promover a localização dos danos. Os caminhos 4-8, 6-7, 7-6, 8-1 e 8-4 representam respectivamente os elementos: 31, 46, 54, 57 e 60.

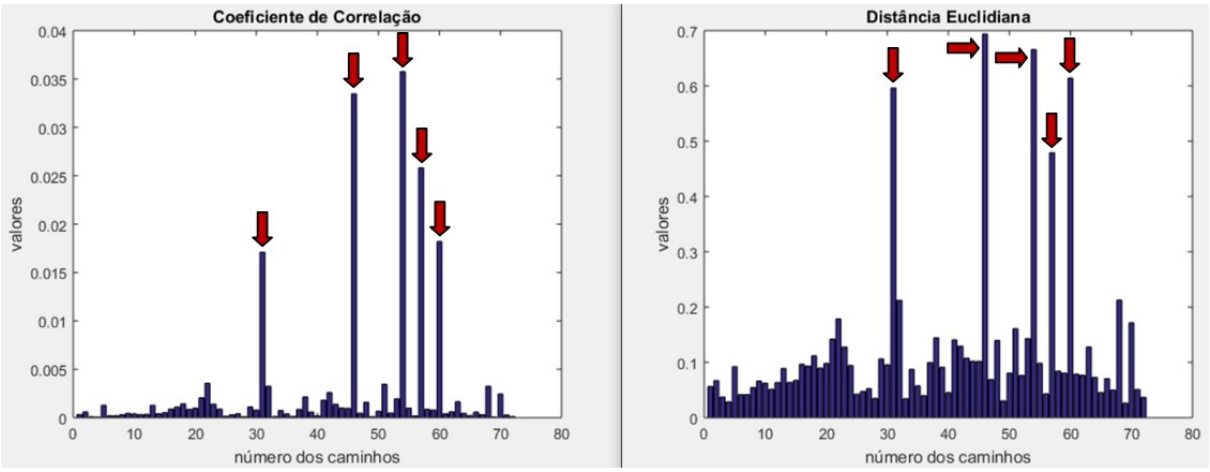
Figura 50 - Relação da localização dos danos com o número do caminho nas métricas 1 e 2.



Fonte: própria autora.

O mesmo pode ser constato na figura 51, porém usando como referência as métricas relacionadas com o coeficiente de correlação e a distância euclidiana.

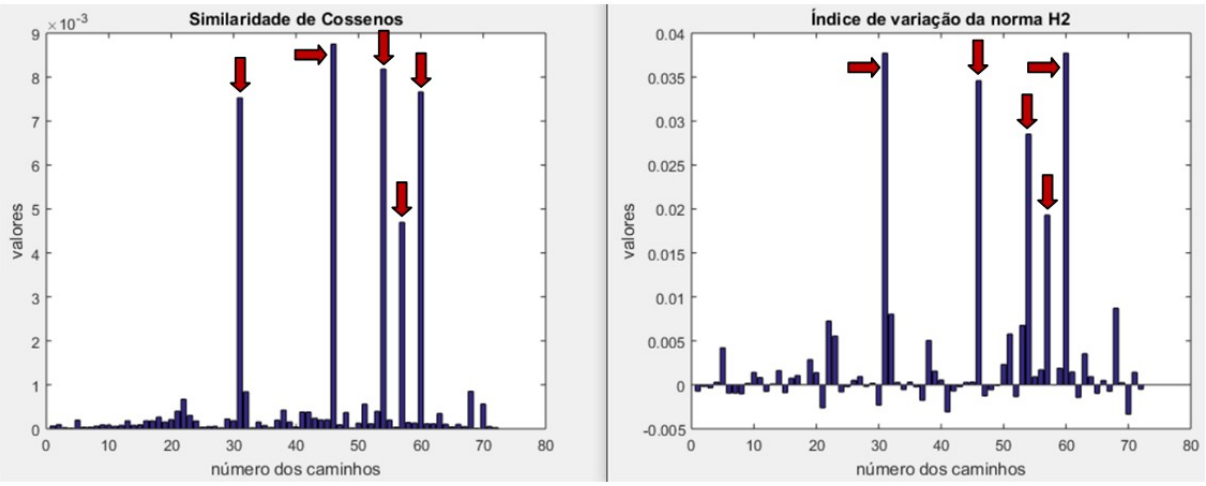
Figura 51- Relação da localização dos danos com o número do caminho nas métricas 3 e 4.



Fonte: própria autora.

Na figura 52 é mostrada a comparação usando como referência as métricas: similaridade de cossenos e norma H2. Os mesmos pontos são apresentados.

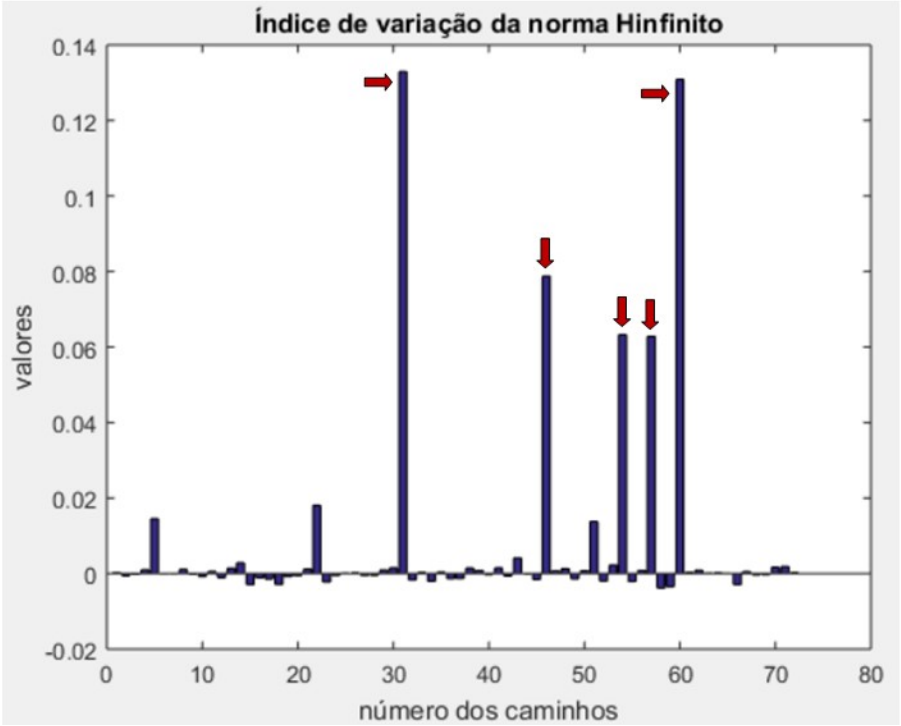
Figura 52-Relação da localização dos danos com o número do caminho nas métricas 5 e 6.



Fonte: própria autora.

E finalmente, na figura 53, tem-se a apresentação dos pontos: 31, 46, 54, 57 e 60 em relação à métrica Hinfinito.

Figura 53-Relação da localização dos danos com o número do caminho na métrica7.



Fonte: própria autora.

No teste realizado com o conjunto de dados do dano 2, os resultados de classificação não foram tão precisos. O modelo classificou 68 elementos como verdadeiros positivos (sem

dano), 0 como falsos negativos, 2 como falsos positivos e 2 como verdadeiros negativos (com dano). Isto pode ser visto na tabela 12.

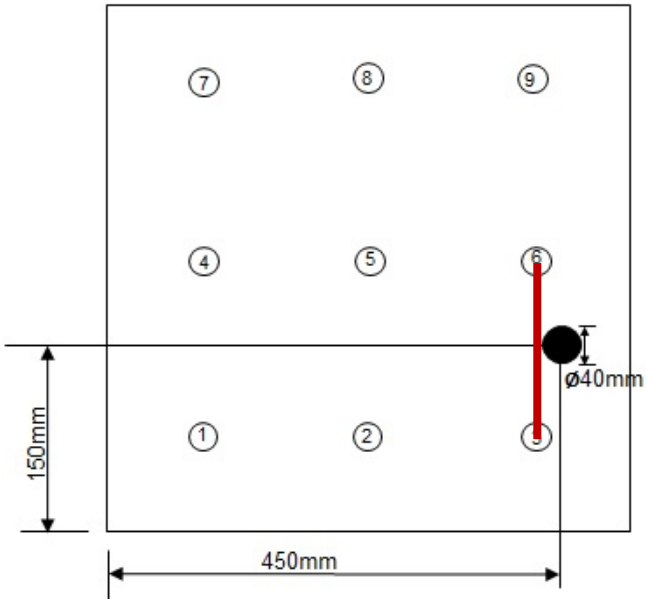
Tabela 15 - Matriz confusão e métricas derivadas para o teste usando o conjunto de dados com dano 2.

Matriz Confusão	**Classe1 (sem dano)**	**Classe2 (com dano)**
Classe1	TP: 68	FN: 0
Classe2	FP: 2	TN: 2
Outras métricas		
Taxa de erro – classe1	0.0	
Taxa de erro – classe 2	0.5	
Erro total	~0.02	
Acurácia total	~0.97	
Precisão	~0.97	
Sensibilidade	1.0	
Especificidade	0.5	
Média harmônica	~0.98	
Confiabilidade positiva	~0.97	
Confiabilidade negativa	1.0	
Suporte	~0.94	
Cobertura	~0.97	
Kappa	~0.65	

Fonte: própria autora.

Apesar das classificações não terem sido tão precisas, a localização do dano na chapa de alumínio foi satisfatória. O algoritmo conseguiu identificar dois caminhos: 3-6 e 6-3. A Figura 54 mostra com detalhes o diagnóstico realizado. Deve-se destacar que os sinais em caminhos opostos, isto é, Pij (atuador i e sensor j) e Pji (atuador j e sensor i) são, geralmente, similares. Em alguns casos, onde o dano está mais próximo de i ou de j, os sinais e, portanto, as métricas apresentam pequenas diferenças.

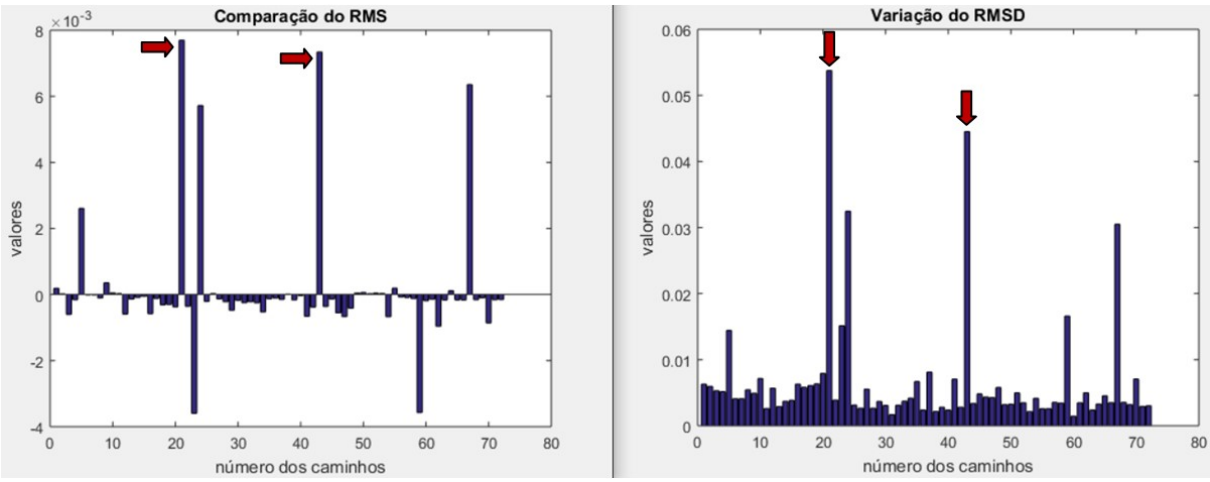
Figura 54 -Localização do dano na estrutura referente ao conjunto de dados dano 2.



Fonte: própria autora.

Neste caso, as rotinas conseguiram identificar apenas dois caminhos mais prováveis para a localização do dano. Portanto, os caminhos 3-6 e 6-3 correspondem respectivamente aos elementos 21 e 43. As setas vermelhas na figura 55 mostram os números dos caminhos detectados.

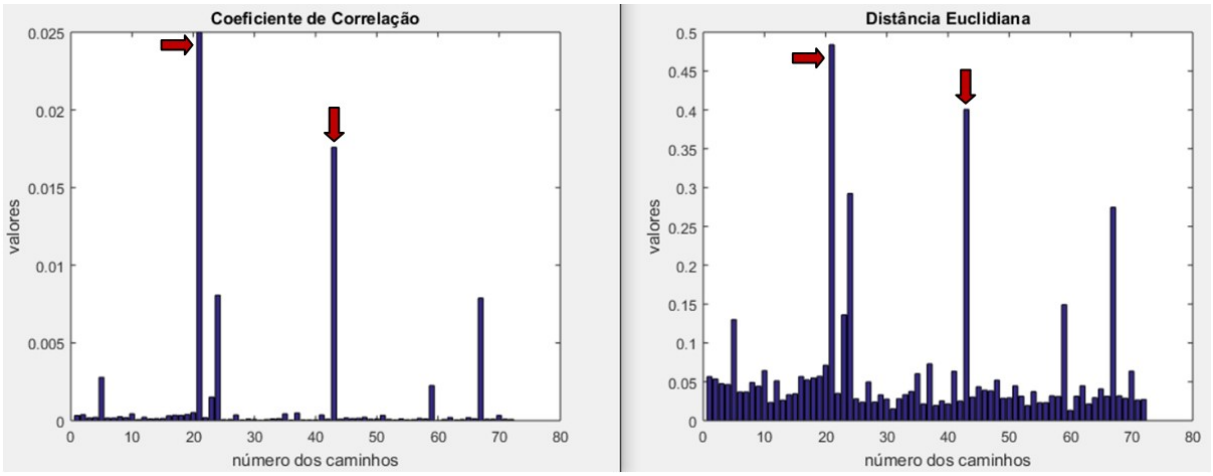
Figura 55-Relação da localização dos danos com o número do caminho nas métricas 1 e 2.



Fonte: própria autora.

O mesmo pode ser constato na figura 56, porém usando como referência as métricas relacionadas com o coeficiente de correlação e a distância euclidiana.

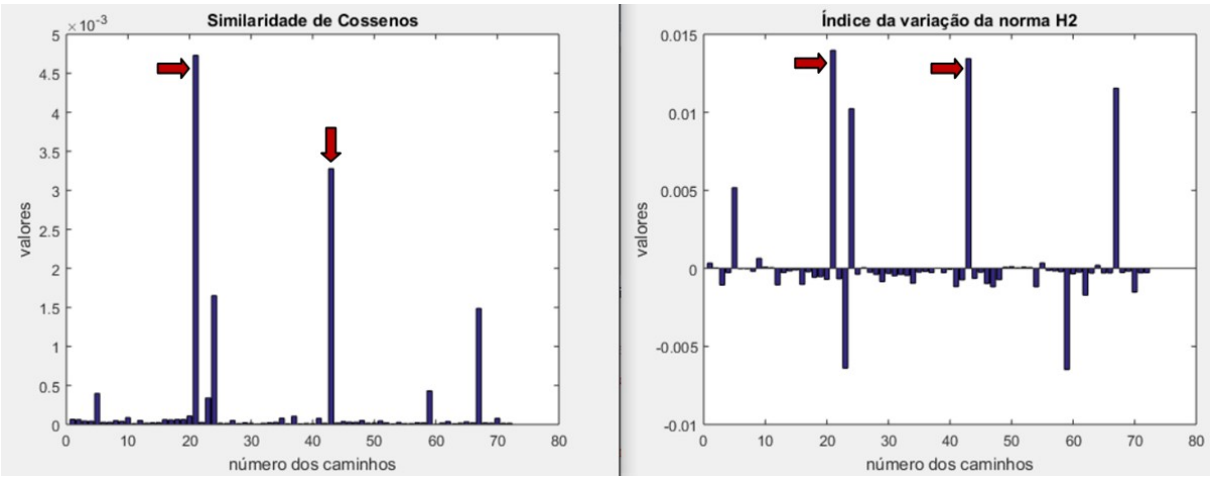
Figura 56-Relação da localização dos danos com o número do caminho nas métricas 3 e 4.



Fonte: própria autora.

Na figura 57 é mostrada a comparação usando como referência as métricas: similaridade de cossenos e norma H2. Os mesmos pontos são apresentados.

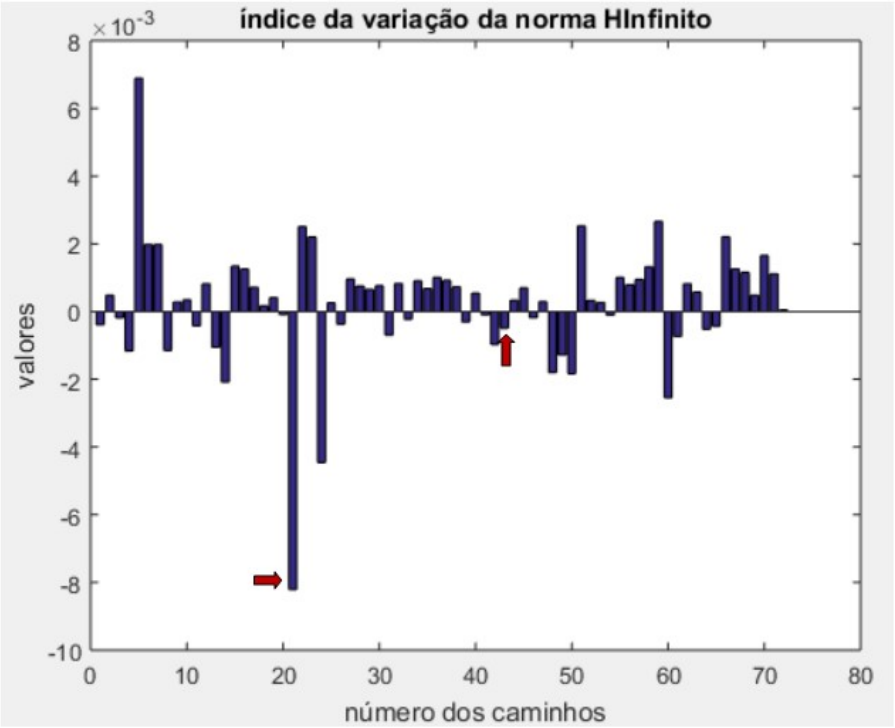
Figura 57-Relação da localização dos danos com o número do caminho nas métricas 5 e 6.



Fonte: própria autora.

E finalmente, na figura 58, tem-se a apresentação dos pontos: 21 e 43.

Figura 58-Relação da localização dos danos com o número do caminho na métrica7.



Fonte: própria autora.

No teste usando o conjunto de dados referente ao dano 3, o classificador também teve um bom desempenho. Como pode ser visto na tabela 13, não teve nenhuma ocorrência de falsos positivos e nem falsos negativos. Dos 72 caminhos analisados, 69 foram devidamente classificados como sem dano e três com a presença de danos.

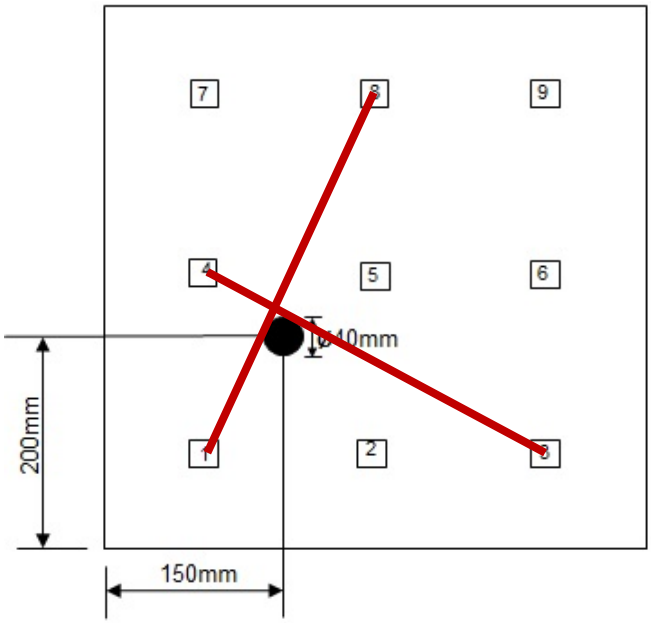
Tabela 16 – Matriz confusão e métricas calculadas para o teste usando o conjunto de dados com dano 3.

Matriz Confusão	**Classe1 (sem dano)**	**Classe2 (com dano)**
Classe 1	TP: 69	FN: 0
Classe 2	FP: 0	TN: 3
Outras métricas		
Taxa de erro – classe 1	0.0	
Taxa de erro – classe 2	0.0	
Erro total	1.0	
Acurácia total	1.0	
Precisão	1.0	
Sensibilidade	1.0	
Especificidade	1.0	
Média harmônica	1.0	
Confiabilidade positiva	1.0	
Confiabilidade negativa	1.0	
Suporte	~0.95	
Cobertura	~0.95	
Kappa	~1.0	

Fonte: própria autora.

Para o teste envolvendo o conjunto de dados com o dano 3 foram encontrados três caminhos com dano: 3-4, 4-3 e 8-1. A figura 59 apresenta com melhor clareza os caminhos encontrados que apresentam a localização do dano na estrutura.

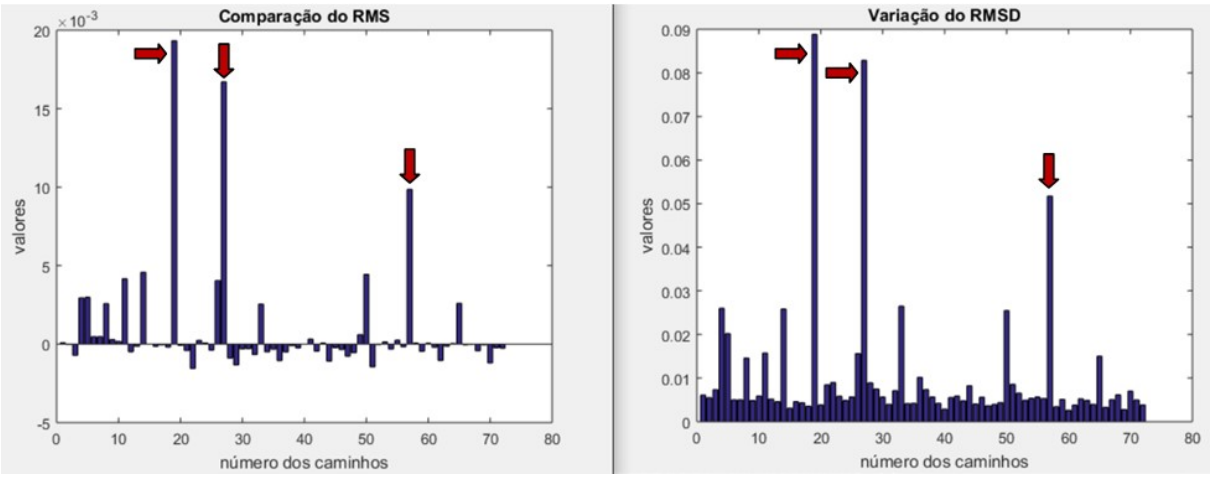
Figura 59-Localização do dano na estrutura referente ao conjunto de dados com dano 3.



Fonte: própria autora.

O procedimento conseguiu identificar os caminhos: 3-4, 4-3 e 8-1 como o lugar mais provável de onde o dano esteja. Esses caminhos correspondem respectivamente aos elementos 19, 27 e 57. A figura 60 mostra os números dos caminhos detectados.

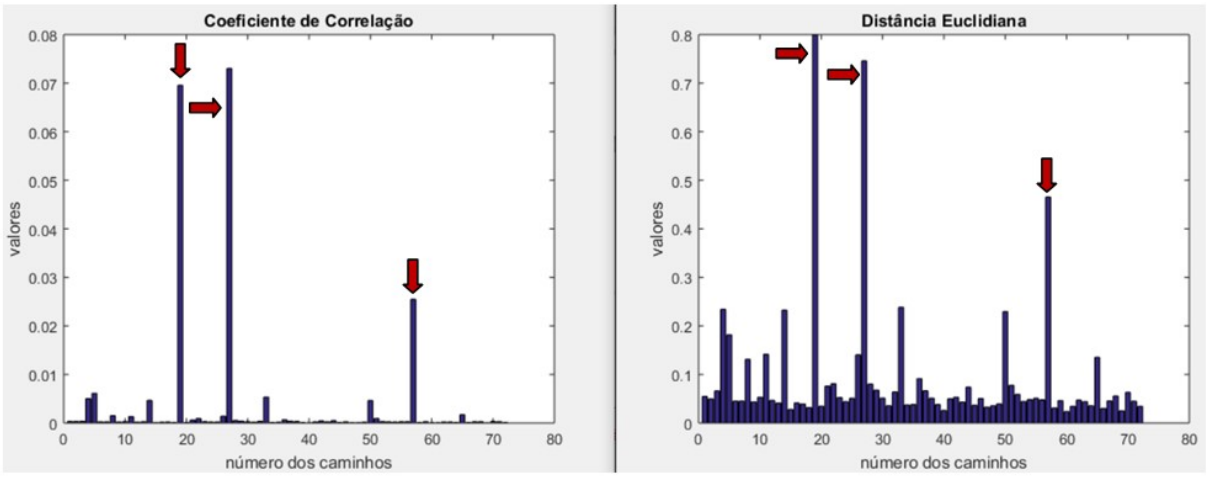
Figura 60 -Relação da localização dos danos com o número do caminho nas métricas 1 e 2.



Fonte: própria autora.

O mesmo pode ser constato na figura 61, porém usando como referência as métricas relacionadas com o coeficiente de correlação e a distância euclidiana.

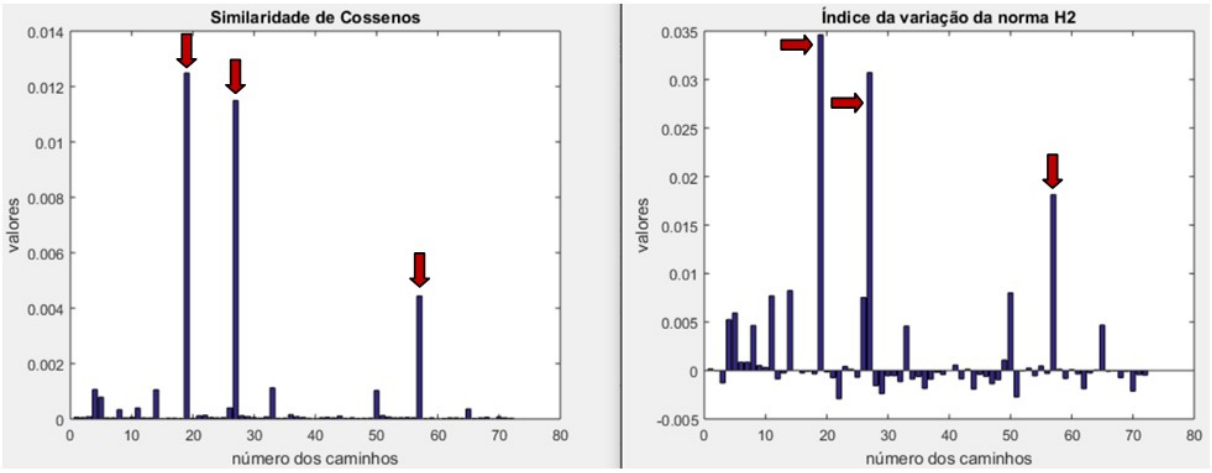
Figura 61-Relação da localização dos danos com o número do caminho nas métricas 3 e 4.



Fonte: própria autora.

Na figura 62 é mostrada a comparação usando como referência as métricas: similaridade de cossenos e norma H2. Os mesmos pontos são apresentados.

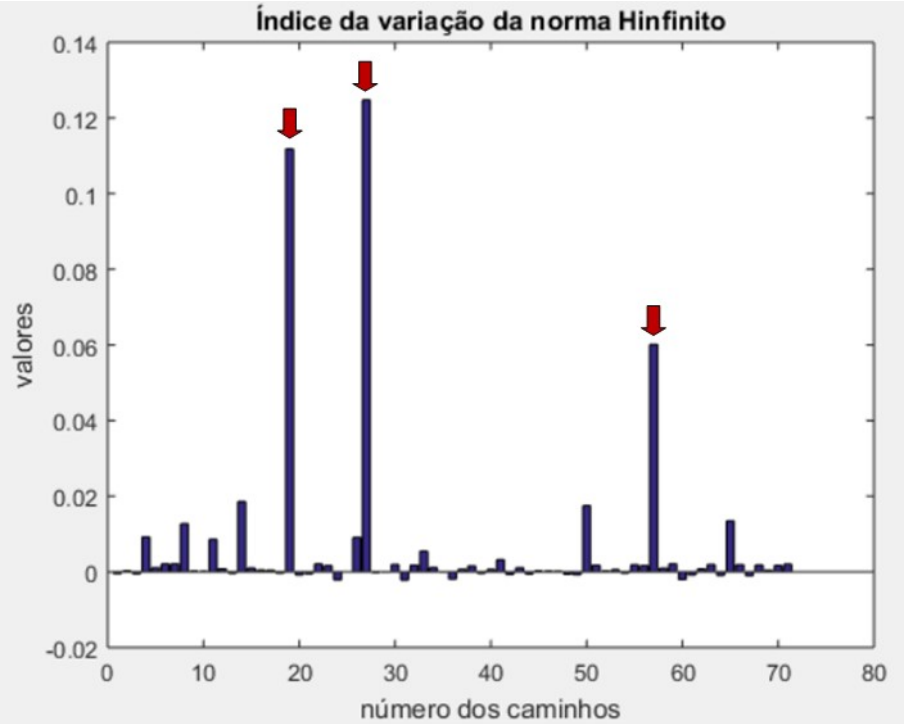
Figura 62-Relação da localização dos danos com o número do caminho nas métricas 5 e 6.



Fonte: própria autora.

E posteriormente, na figura 63, tem-se a apresentação dos pontos: 19, 27 e 57.

Figura 63-Relação da localização dos danos com o número do caminho na métrica 7.



Fonte: própria autora.

Os testes realizados com os conjuntos de dados referentes aos danos 4 e 5 apresentaram a mesma classificação. Nenhuma presença de falsos positivos ou falsos negativos. Dos 72 caminhos analisados, 65 foram precisamente classificados como sem danos e 7 apresentaram a ocorrência de danos conforme pode ser visto na tabela 14.

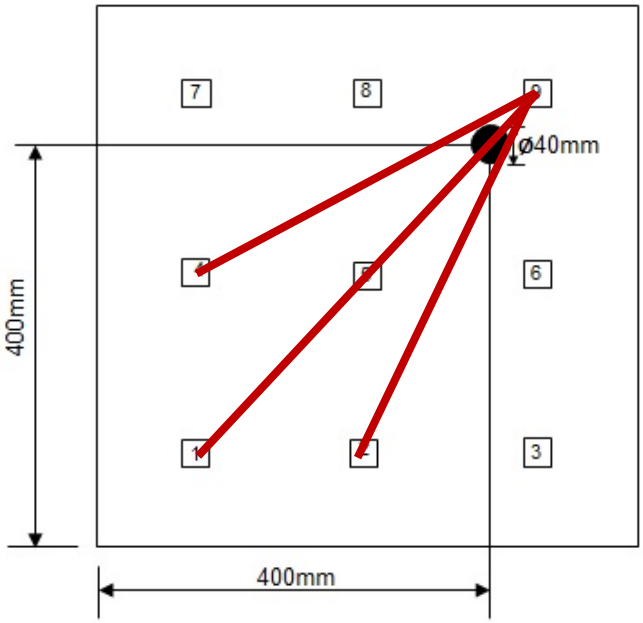
Tabela 17- Matriz confusão e métricas derivadas para o teste usando o conjunto de dados dano 4 e 5.

Matriz Confusão	**Classe1 (sem dano)**	**Classe2 (com dano)**
Classe 1	TP: 65	FN: 0
Classe 2	FP: 0	TN: 7
Outras métricas		
Taxa de erro – classe 1	0.0	
Taxa de erro – classe 2	0.0	
Erro total	1.0	
Acurácia total	1.0	
Precisão	1.0	
Sensibilidade	1.0	
Especificidade	1.0	
Média harmônica	1.0	
Confiabilidade positiva	1.0	
Confiabilidade negativa	1.0	
Suporte	~0.90	
Cobertura	~0.90	
Kappa	~1.0	

Fonte: própria autora.

Para o teste envolvendo o conjunto de dados com dano 4 foram diagnosticados sete caminhos: 1-9, 4-9, 5-9, 9-1,9-2, 9-4 e 9-5. A figura 64 apresenta com melhor clareza os caminhos encontrados que apresentam a localização do dano na estrutura.

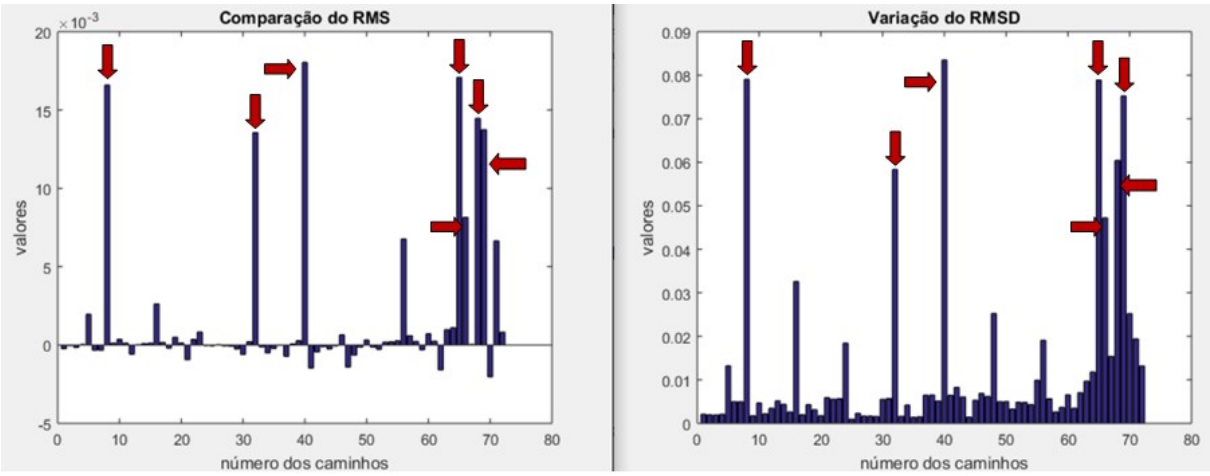
Figura 64-Localização do dano na estrutura referente ao conjunto de dados dano 4.



Fonte: própria autora.

A figura 65apresenta quais são os elementos relacionados com os seguintes caminhos que mostram a localização do dano identificado pelas rotinas. Os pontos detectados foram: 8, 32, 40, 65, 66, 68 e 69.

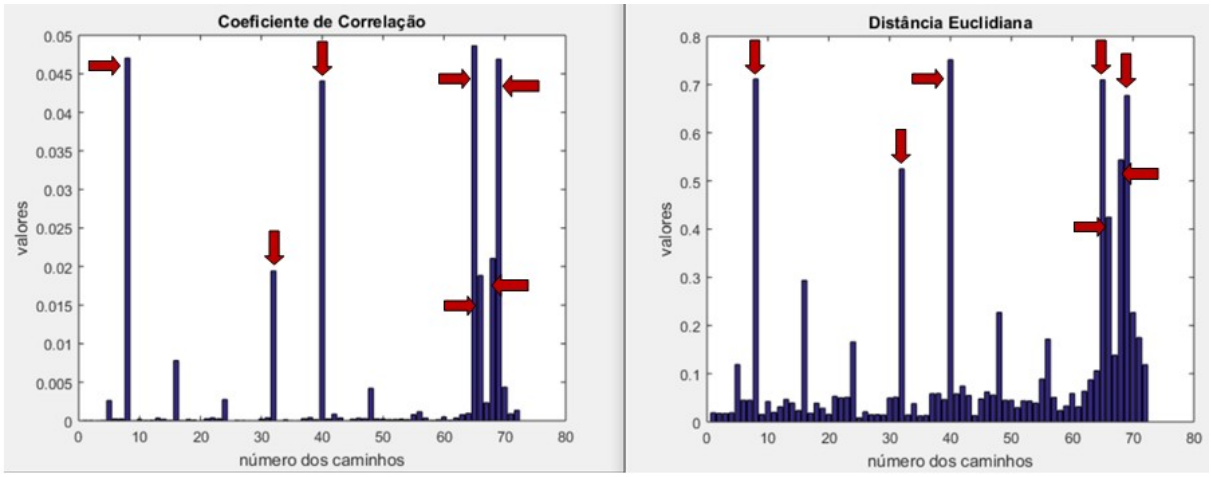
Figura 65-Relação da localização dos danos com o número do caminho nas métricas 1 e2.



Fonte: própria autora.

O mesmo pode ser constato na figura 66, porém usando como referência as métricas relacionadas com o coeficiente de correlação e a distância euclidiana.

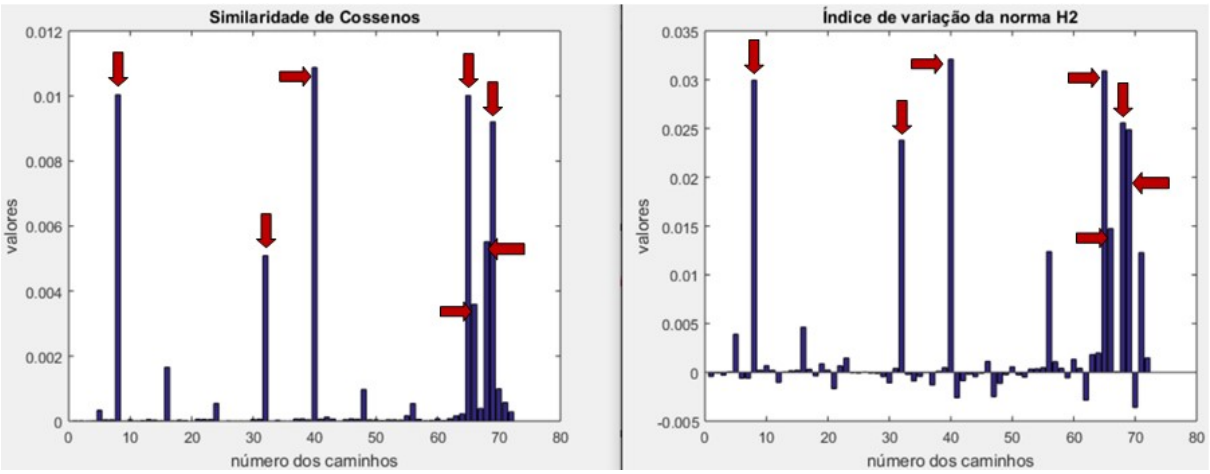
Figura 66-Relação da localização dos danos com o número do caminho nas métricas 3 e 4.



Fonte: própria autora.

Na figura 67 é mostrada a comparação usando como referência as métricas: similaridade de cossenos e norma H2. Os mesmos pontos são apresentados.

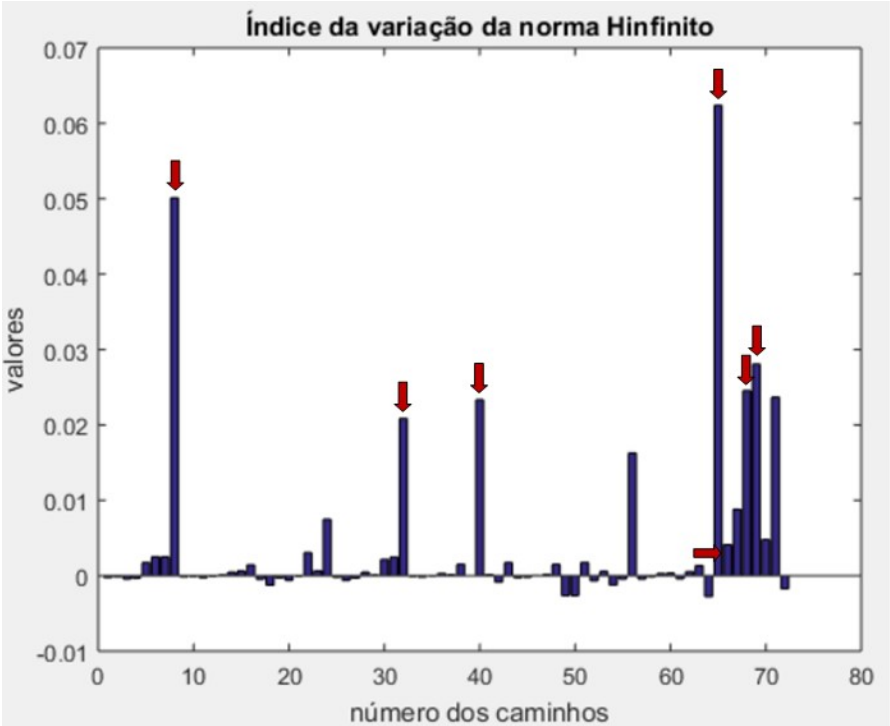
Figura 67-Relação da localização dos danos com o número do caminho nas métricas 5 e 6.



Fonte: própria autora.

E posteriormente, na figura 68, tem-se a apresentação dos pontos: 8, 32, 40, 65, 66, 68 e 69 usando como referência a métrica obtida através do cálculo da norma Hinfinito.

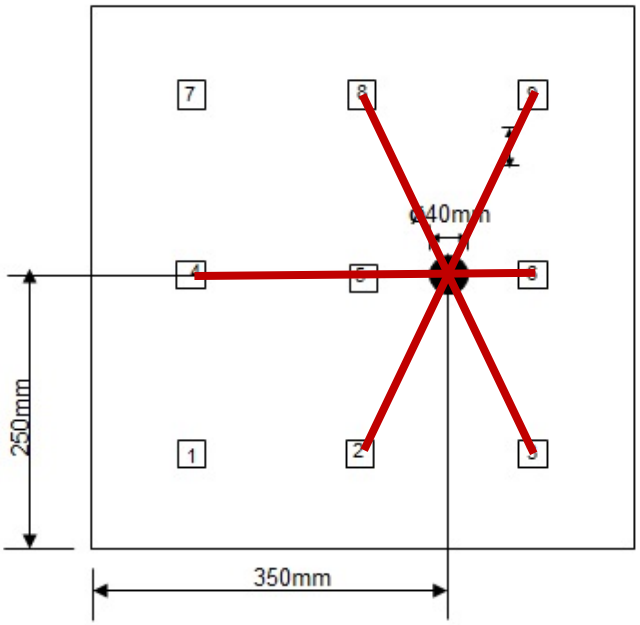
Figura 68-Relação da localização dos danos com o número do caminho na métrica 7.



Fonte: própria autora.

Para o teste envolvendo o conjunto de dados 5 foram diagnosticados sete caminhos: 3-8, 4-6, 5-6, 6-4, 6-5, 8-3 e 9-2. A figura 69 apresenta com melhor clareza os caminhos encontrados que apresentam a localização do dano na estrutura.

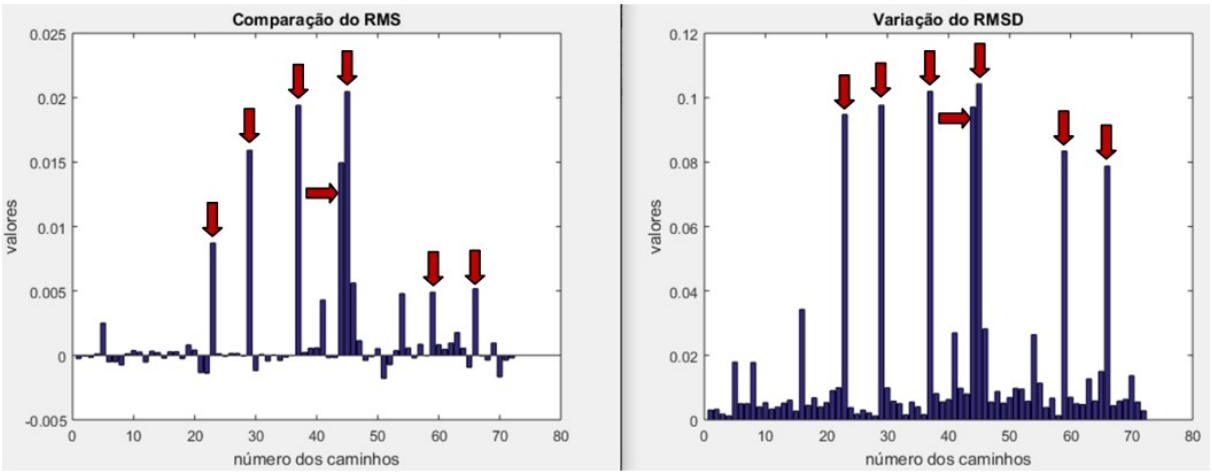
Figura 69-Localização do dano na estrutura referente ao conjunto de dados dano 5.



Fonte: própria autora.

A figura 70 apresenta quais são os elementos relacionados com os seguintes caminhos que mostram a localização do dano identificado pelas rotinas. Os pontos detectados foram: 23, 29, 37, 44, 45, 59 e 66.

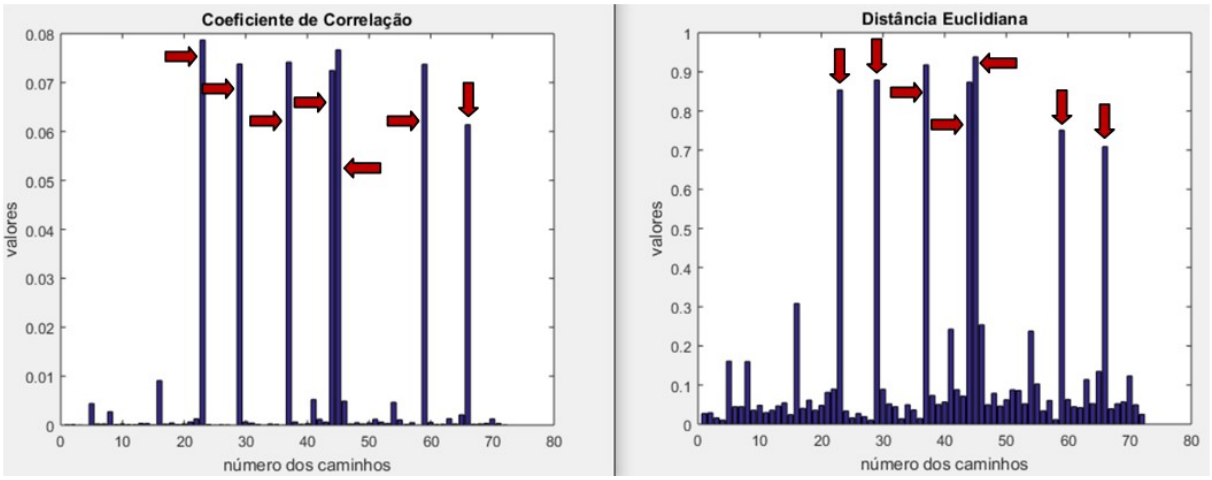
Figura 70-Relação da localização dos danos com o número do caminho nas métricas 1 e 2.



Fonte: própria autora.

O mesmo pode ser constato na figura 71, porém usando como referência as métricas relacionadas com o coeficiente de correlação e a distância euclidiana.

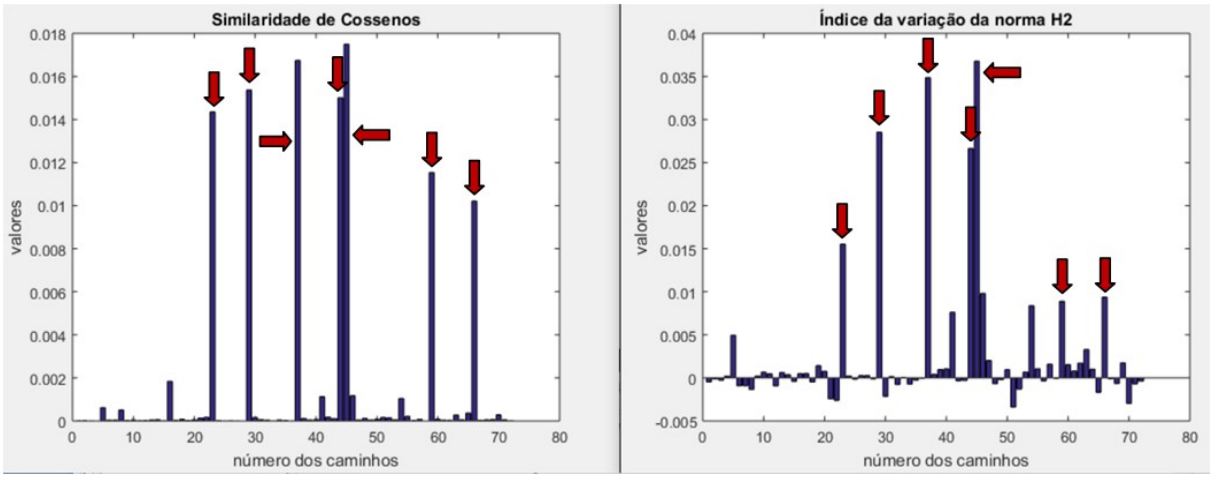
Figura 71-Relação da localização dos danos com o número do caminho nas métricas 3 e 4.



Fonte: própria autora.

Na figura 72 é mostrada a comparação usando como referência as métricas: similaridade de cossenos e norma H2. Os mesmos pontos são apresentados.

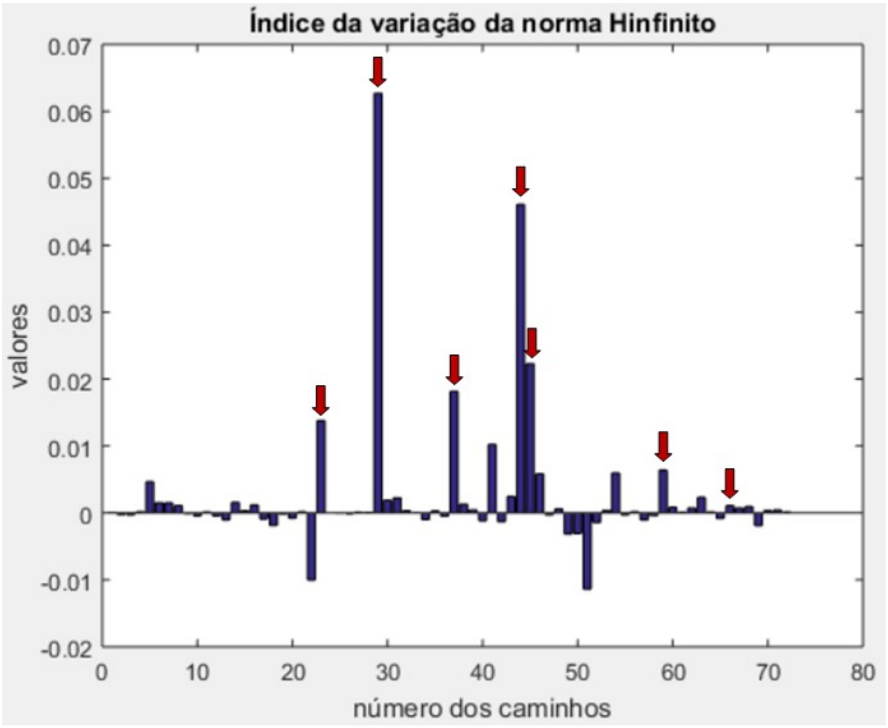
Figura 72- Relação da localização dos danos com o número do caminho nas métricas 5 e 6.



Fonte: própria autora.

E posteriormente, na figura 73, tem-se a apresentação dos pontos: 23, 29, 37, 44, 45, 59 e 66 usando como referência a métrica obtida através do cálculo da norma Hinfinito.

Figura 73-Relação da localização dos danos com o número do caminho na métrica 7.



Fonte: própria autora.

Os resultados mostrados neste tópico foram obtidos com o modelo gerado utilizando apenas os dados dos danos 1, 2 e 4. Após o treinamento e obtenção do modelo, foi feita a

análise de todos os cinco conjuntos de dados: dano 1, 2, 3, 4 e 5. Os resultados foram excelentes, demonstrando uma boa capacidade de generalização, isto é, não é necessário treinar o modelo com todos os possíveis danos. Os danos 3 e 5 foram corretamente identificados sem que os dados destes danos fossem utilizados no treinamento.

5 CONCLUSÕES E SUGESTÕES

O processo de aprendizagem é uma etapa muito importante para a construção de sistemas autônomos. No entanto, ainda, é um campo que apresenta desafios a serem transpassados pelos pesquisadores atuantes na área de inteligência artificial. Apesar das dificuldades, os algoritmos disponíveis atualmente já demonstram resultados relevantes no processo de reconhecimento de padrões, análise de sinais, robótica e outros. Atualmente estes programas estão sendo usados para solucionar problemas em diversas áreas como: na Biologia, através de procedimentos de localização de proteínas nas células, na Medicina por intermédio de análise de imagens no diagnóstico de doenças, na Engenharia, através do monitoramento contínuo de estruturas, entre outras.

O algoritmo SVM é um método baseado na otimização de uma função objetivo que pode ser treinado usando uma abordagem supervisionada. Sua meta é encontrar um hiperplano que maximize as margens de classificação em relação aos dados. Devido a algumas características como: ser tolerante a dados ruidosos, ser determinístico, robustez quando submetidos a dados que possuem muitas características, este vem se destacando na resolução de problemas.

Um dos pontos forte do SVM é que este pode ser aplicado em dados não lineares apresentando bons resultados de desempenho. O uso das funções *kernels* possibilitou o uso do algoritmo neste tipo de contexto. Porém, isto pode ser desafiador para um usuário iniciante, pois a busca pelos melhores parâmetros pode ser um processo difícil para pessoas que não possuem maturidade e experiência neste tipo de algoritmo.

Apesar da complexidade de entender o funcionamento do algoritmo, foi possível utilizá-lo para promover o monitoramento de estruturas. Uma das etapas que pode influenciar todo o projeto é a análise e pré-processamento dos dados. É um passo muito relevante, pois é necessário ter uma compreensão na forma em que estes dados foram gerados e nos seus significados. Para tal, foram efetuadas as métricas, porque com estas, tem-se um conhecimento qualitativo dos dados. Através destes índices, foi possível construir a base de dados para treinar o algoritmo.

Com os dados adequadamente trabalhados, o algoritmo promoveu boas classificações, concretizando o processo de identificação. Este foi capaz de discernir quando havia a presença ou a ausência de dados na estrutura. Outro fator crucial foi o processo de localização dos danos na placa de alumínio. As rotinas criadas mostraram resultados satisfatórios, pois conseguiram emitir os caminhos de onde o dano estava presente no

material. Portanto, pode-se concluir que foi possível realizar o monitoramento de falhas em estruturas inteligentes através da utilização do algoritmo SVM com uma abordagem supervisionada.

Uma das principais características desta metodologia é a possibilidade de se determinar danos em posições que não foram utilizadas no treinamento. Esta questão tem sido um dos limitantes de técnicas supervisionadas, pois, geralmente, é necessário o conhecimento prévio de todos os danos.

5.1 SUGESTÕES PARA TRABALHOS FUTUROS

Neste trabalho não se avaliou quais seriam as melhores métricas. Este pode ser um tópico interessante de pesquisa, pois como se comentou anteriormente, o desempenho do modelo é fortemente dependente das informações recebidas.

Outro ponto que pode ser melhor abordado é a escolha dos *kernels* e a configuração dos seus parâmetros, pois falhas estruturais são processos altamente não lineares. O processo de configuração dos parâmetros não é trivial. Utilizar valores altos ou baixos muda de modo significativo a construção do classificador. Um fato interessante é que a escolha dos parâmetros está relacionada com os dados de treinamento. Desta forma, poderiam ser explorados testes mais elaborados na busca dos melhores parâmetros, como a utilização de algoritmos específicos para este tipo de tarefa.

O método proposto tem capacidade de generalização para o mesmo tipo de dano. Os danos foram simulados colando massas em diferentes posições da estrutura. Esta capacidade poderia ser testada para diferentes tipos de danos, por exemplo, em materiais compósitos, será que a variação nos sinais causados por delaminações é similar a variação causada por furos?

REFERÊNCIAS

ALENCAR, L. F. de. Utilização de informações lexicais extraídas automaticamente de corpora na análise sintática computacional do português. **Revista de Estudos da Linguagem**, Belo Horizonte, v. 19, n. 1, p.7-85, 2011.

ALMEIDA, O. C. P. de. **Classificação de tábuas de madeira usando processamento de imagens digitais e aprendizado de máquina**. 2014. 107 f. Tese (Doutorado em Agronomia) – Faculdade de Ciências Agronômicas, Universidade Estadual Paulista, Botucatu, 2014.

ALMEIDA, T. do N. S. de. **Estimativa da radiação solar ultravioleta em Botucatu/SP/Brasil utilizando técnicas de aprendizado de máquina**. 2013. 57 f. Dissertação (Mestrado em Agronomia) - Faculdade Ciências Agronômicas, Universidade Estadual Paulista, Botucatu, 2013.

ANDREOLA, R.; HAERTEL, V. Support vector machines na classificação de imagens hiperespectrais. In: XIV SIMPÓSIO BRASILEIRO DE SENSORIAMENTO REMOTO, 14.,2009, Rio Grande do Norte. **Anais...** Natal:INPE, 2009.p. 6757-6764.

ARAGHINEJAD, S.; AZMI, M.; KHOLGHI, M. Application of artificial neural network ensembles in probabilistic hydrological forecasting. **Journal of Hydrology,** Amsterdam, v. 407, n. 1-4, p.94-104, 2011.

AREFIN, A. S.; RIVEROS, C.; BERETTA, R. MOSCATO, P. GPU-FS-KNN: a software tool for fast and scalable KNN computation using GPUs. **Public Library of Science**,San Francisco, v.7, n. 8, p.e44000, 2012.

BONESSO, D. **Estimação dos parâmetros do kernel em um classificador SVM de imagens hiperespectrais em uma abordagem multicalsse**. 2013. 108 f. Dissertação (Mestrado em Sensoriamento Remoto) – Centro Estadual de Pesquisas em Sensoriamento Remoto e Meteorologia, Universidade Federal do Rio Grande do Sul,Porto Alegre, 2013.

BEDWORTH, M.; O'BRIEN, J. The Omnibus model: a new model of data fusion? **IEEE Aerospace and Electronic Systems Magazine**, Piscataway, v. 15, n. 4, p.30-36, 2000.

BURGES, C. J. C. A tutorial on Support Vector Machines for pattern recognition.**Data Mining and Knowledge Discovery**, New York, v. 2, n. 2, p.121-167, 1998.

CARNASCIALI, A. M. dos S.; DELAZARI, L. S. A localização geográfica como recurso organizacional: utilização de sistemas especialistas para subsidiar a tomada de decisão locacional do setor bancário. **RAC**., Curitiba, v. 15, n. 1, p.103-125, 2011.

CHANG, C. C.;LIN, C. J. LIBSVM: a library for support vector machines. ACM Transactions on Intelligent Systems and Technology, New York, v. 2, n. 3, p.27, 2011.

CHENG, Q.; LI, D.; TANG, C. KNN Matting.**IEEE Transactions on pattern analysis and machine intelligence,** Piscataway, v. 35, n. 9, p.2175-2188, 2013.

CHINO, D. Y. T. **Mineração de padrões frequentes em séries temporais para apoio à**

tomada de decisão em agrometerologia. 2014. 78 f. Dissertação (Mestrado em Ciência da Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, São Carlos, 2014.

COPPIN, B. **Inteligência artificial**. Rio de Janeiro: LTC, 2010. 668 p.

COVER, T. M. Geometrical and Statistical Properties of Systems of Linear Inequalities with Applications in Pattern Recognition. **IEEE Transactions on Electron**, Piscataway, v. EC-14, n. 3, p.326-334, 1965.

DEBSKA, B.; GUZOWSKA-SWIDER, B. Application do artificial neural network in food classification. **Analytica Chimica Acta**, Amsterdam, v. 705, n. 1, p.283-291, 2011.

DOMINGUES, M. L. C. S. **Mineração de dados utilizando aprendizado não-supervisionado: um estudo de caso para banco de dados da saúde**. 2003. 127 f. Dissertação (Mestrado em Ciência da Computação) – Instituto de Informática, Universidade Federal do Rio Grande do Sul, Porto Alegre, 2003.

FACELI, K.; LORENA, A. C.; GAMA, J.; CARVALHO, A. C. P. L. F**. Inteligência Artificial: uma abordagem de aprendizado de máquina**. Rio de Janeiro: LTC, 2011. 378 p.

FARRAR, C.; WORDEN, K.An introduction to structural heath monitoring.**Royal Society of London Transactions Series A,** London,v. 365, n. 1851, p.303-315, 2007.

FIGUEIREDO, E.; PARK, G.; WORDEN, K.; FARRAR, C.; FIGUEIRAS, J. Machine Learning Algorithms for Damage Detection under Operational and Environmental Variability. **Journal Structural Health Monitoring,**London, v. 10, n. 6, p.559-572,2011.

FIORIN, D. V.; MARTINS, F. R.; SCHUCH, N. J.;PEREIRA, E. B. Aplicações de redes neurais e previsões de disponibilidade de recursos energéticos solares. **Revista Brasileira de Ensino de Física,** São Paulo, v. 33, n. 1, p.1309-1309, 2011.

FORMOSO, C. T. Metodologias para o desenvolvimento de sistemas especialistas para planejamento em construção. **Production**, São Paulo, v. 3, n. 1, p.5-13, 1993.

FREITAS, H.; MOSCAROLA, J. Da observação à decisão: métodos de pesquisa e análise quantitativa e qualitativa de dados. **RAE electron**, São Paulo, v. 1, n. 1, jan/jun.2002.

GLASMACHERS, T.; IGEL, C. Maximum-Gain Working Set Selection for SVMs. **Journal of Machine Learning Research,** Cambridge, v.7, p.1437-1466, 2006**.**

HAYKIN, S. **Redes neurais princípios e práticas**. 2.ed. Porto Alegre: Bookman, 2001.902 p.

LEO, D. J.**Engineering analysis of smart material systems**. New Jersey: Wiley, 2007.576 p.

LIMA, A. R. G. **Máquinas de vetores suporte na classificação de impressões digitais**. 2002, 81 f. Dissertação (Mestrado em Computação)- Departamento de Computação, Universidade Federal do Ceará. Fortaleza, 2002.

LORENA, A. C.; CARVALHO, A. C. P. L. F. **Uma introdução às máquinas de vetores de suporte**. São Carlos: Universidade de São Paulo, 2003. 66 p. (Relatórios Técnicos do ICMC).

LUGER, G. F. **Inteligência artificial**. São Paulo: Pearson, 2014.614 p.

MARQUI, C. R**. Modelagem de estruturas piezelétricas para aplicação em localização de falhas**. 2007. 237 f. Dissertação (Mestrado em Engenharia Mecânica) – Faculdade de Engenharia, Universidade Estadual Paulista, Ilha Solteira, 2007.

MENDES, R. D. Inteligência artificial: sistemas especialistas no gerenciamento da informação. **Ci. Inf.,** Brasília, DF, v. 26, n. 1, jan/apr. 1997. Disponível em: <http://www.scielo.br/scielo.php?script=sci_arttext&pid=S0100-19651997000100006> . Acessado em: 20 ago. 2016.

MITCHELL, T. M. **Machine learning**. 1. ed.New York: McGraw-Hill, 1997. 414 p.

MONARD, M. C.; BARANAUSKAS, J. A. Conceitos de aprendizado de máquina. In: REZENDE, S.O. **Sistemas inteligentes:** fundamentos e aplicações. Barueri: Manole, 2003. cap.4, p.89-114.

NEWELL, A.; SHAW, J. C.; SIMON, H. A. Report on a General Problem-Solving Program**.** In: INFORMATION PROCESSING PROCEEDINGS OF THE INTERNATIONAL CONFERENCE ON INFORMATION PROCESSING,19.,60, Paris. **Proceeding…**Paris: UNESCO, 1960. p. 256-264.

NICK, W.; ASAMENE, K.;BULLOCK, G.; ESTERLINE, A.;SUNDARESAN, M.A study of machine learning tecnhiques for detecting and classifying structural damage. **International Journal of Machine Learning and Computing,** Singapore, v. 5, n. 4. p.131-318, 2015.

OLIVEIRA, M. A. de. **Monitoramento de integridade estrutural baseada em sensores piezelétricos e análise de sinais no domínio do tempo**. 2013. 129 f. Tese (Doutorado em automação) - Faculdade de Engenharia, Universidade Estadual Paulista, Ilha Solteira, 2013.

PALOMINO, L. V.; STEFFEN JUNIOR, V. Estudo do método de impedância eletromecânica para detecção de danos incipientes em uma viga de alumínio. In: SIMPÓSIO DE PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA, 17., 2007, Uberlândia. **Anais...** Uberlândia:FEMEC/UFU, 2007.

PARADIGMA. In: DICIONÁRIO da língua portuguesa. Lisboa: Priberam Informática, 2013. Disponível em: <https://www.priberam.pt/DLPO/paradigma />. Acessado em: 16 mai. 2016.

PIOOZNIA M., DENG Y. SVM Classifier – a comprehensive java interface for support vector machine classification of microarray data. **BMC Bioinformatics,** London, v.7, n. 4, p. 1,2006.

REZENDE, S. O.; Marcacini, R. M.; MOURA, M. F. O uso da mineração de textos para extração e organização não supervisionada de conhecimento. **Revista de Sistemas de Informação da FSMA,**Visconde de Araújo, n. 7, p.7-21, 2011.

RICH, E. **Inteligência artificial**. São Paulo: McGraw-Hill, 1988.502 p.

ROSA, V. A. M. **Localização de danos em estruturas anisotrópicas com a utilização de ondas guiadas.**2016. 88 f. Dissertação (Mestrado em Engenharia Elétrica)- Faculdade de

Engenharia, Universidade Estadual Paulista, Ilha Solteira, 2016.

RUSSEL, S. J.; NORVIG, P. **Inteligência artificial**. 2.ed. Rio de Janeiro: Campus, 2004.1021 p.

RYTTER, A. **Vibration based inspection of civil engineering structures**. 1993. 193 f. Thesis (Ph. D.)- Department of Building Technology and Structural Engineering, University of Aalborg, Denmark, 1993.

SANTOS, E. M. **Teoria e aplicação de *support vector machines* à aprendizagem e reconhecimento de objetos baseado na aparência**. 2002. 111 f. Dissertação (Mestrado em Informática)- Faculdade de Informática, Universidade Federal da Paraíba, Campina Grande, 2002.

SEMOLINI, R. ***Support vector machines*, inferência transdutiva e o problema de classificação**. 2002. 128 f. Dissertação (Mestrado em Engenharia Elétrica)- Faculdade de Engenharia Elétrica e de Computação. Universidade Estadual de Campinas, Campinas, 2002.

SCOTT, P. D. Learning: the construction of a posteriori knowledge structures**.** Proceedings of the Eighth**.**In: INTERNATIONAL JOINT CONFERENCE ON ARTIFICIAL INTELLIGENCE, 8.,1983, Washington. **Proceeding…**Washington: AAAI,1983.p.359-363.

SOUZA, A.; TALON, A. Integração entre banco de dados e redes de bayes no suporte à medicina. **Revista FATEC**, Garça, v. 3, n. 2, art. 26, 2013.

SU, Z.; YE, L.; LU, Y. Guided Lamb waves for identification of damage in composite structures: A review. **Journal of soundand vibration**, Amsterdam,v. 295, n. 3-5, p.753-780, 2006.

TEBALDI, A.; COELHO, L.S.; LOPES JUNIOR., V. Detecção de falhas em estruturas inteligentes usando otimização por nuvens de partículas: fundamentos e estudos de casos. **Revista Controle&Automação**, Campinas, v. 17, n. 3, p.312-330, 2006.

UNIVERSITY OF CALIFORNIA. **Center for machine learning and intelligent systems**: iris data set. Irvine: UCI Machine Learning Repository, 2016. Disponível em: < http://archive.ics.uci.edu/ml/datasets/Iris>. Acessado em: 15 maio 2016.

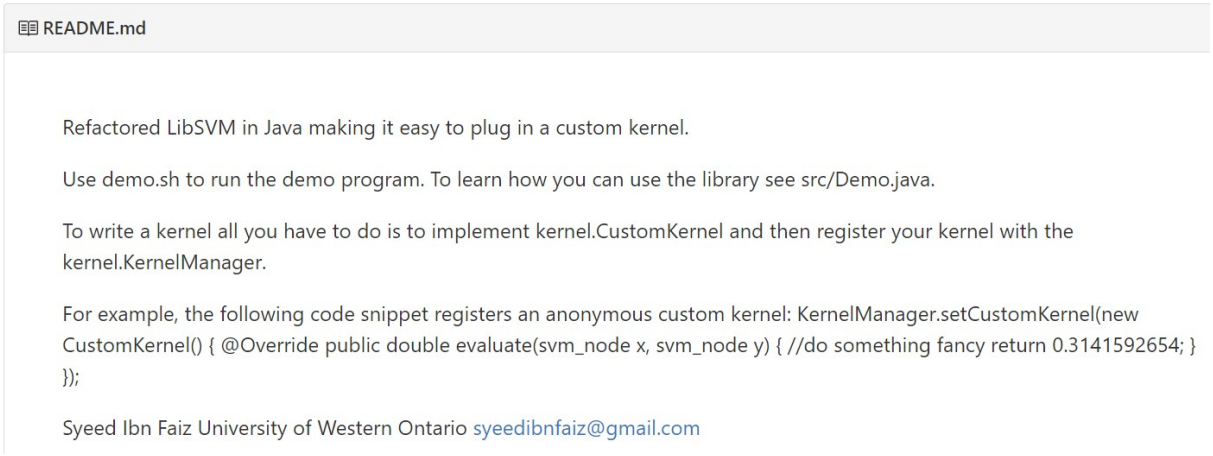
WORDEN, K.; MANSON, G.The application of machine learning to structural health monitoring**.Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences**, London, v. 365, n. 1851, p.515-537, 2007.

YU, X.; YU, Y.; ZENG, Q. Support vector machine classification of streptavidin-binding aptamers.**PLoS ONE,** San Francisco,v. 9, n. 6, p.e99964, 2014.

Apêndice A –Maiores informações sobre a biblioteca LibSVM (autoria, acesso e licença)

Antes de criar a tela do aplicativo, era primordial obter a biblioteca que permitisse a utilização do algoritmo, para tal, foi usado um conjunto de classes codificadas pela linguagem de programação Java. Essas foram desenvolvidas pelo engenheiro de *software* Syeed Ibn Faiz que trabalha (até a presente data) na empresa Google, seu perfil pode ser visto na rede social LinkedIn através do endereço: https://www.linkedin.com/in/syeed-ibn-faiz-52610b40. Ele fez uma revisão/atualização da biblioteca LibSVM, originalmente criada pelos pesquisadores Chang e Lin (2011), conforme pode ser visto na figura 74. Ele implementou a biblioteca usando a linguagem de programação Java e fez melhorias para facilitar a customatização dos *kernels,* ou seja, sua biblioteca oferece uma forma mais desacoplada de adicionar um novo *kernel* no algoritmo SVM. Seu projeto pode ser encontrado no seguinte endereço https://github.com/syeedibnfaiz/libsvm-java-kernel. O site GitHub é serviço web na qual o usuário pode armazenar seus projetos para compartilhar, geralmente estes são *open source*. Assim outras pessoas podem seguir o desenvolvimento do projeto e contribuir com correções, informar erros, criar novas versões, entre outros.

Figura 74- Reescrita da biblioteca LibSVM com customização dos *kernels*.


README.md

Refactored LibSVM in Java making it easy to plug in a custom kernel.

Use demo.sh to run the demo program. To learn how you can use the library see src/Demo.java.

To write a kernel all you have to do is to implement kernel.CustomKernel and then register your kernel with the kernel.KernelManager.

For example, the following code snippet registers an anonymous custom kernel: KernelManager.setCustomKernel(new CustomKernel() { @Override public double evaluate(svm_node x, svm_node y) { //do something fancy return 0.3141592654; } });

Syeed Ibn Faiz University of Western Ontario syeedibnfaiz@gmail.com

Fonte: https://github.com/syeedibnfaiz/libsvm-java-kernel/.

Depois de realizado o *download* das classes, estas foram adicionadas no projeto NetBeans. O autor da biblioteca disponibiliza uma classe chamada *Demo.java* na qual explana a maneira de usar algoritmo SVM. Porém, para um melhor aprendizado de manipulação com a biblioteca foi criada uma classe *Teste.java.* A figura 75 mostra o código-fonte da classe.

Figura 75-Implementação da classe Teste.java.

```
importca.uwo.csd.ai.nlp.kernel.KernelManager;
importca.uwo.csd.ai.nlp.kernel.RBFKernel;
importca.uwo.csd.ai.nlp.libsvm.ex.Instance;
importca.uwo.csd.ai.nlp.libsvm.ex.SVMPredictor;
importca.uwo.csd.ai.nlp.libsvm.ex.SVMTrainer;
importca.uwo.csd.ai.nlp.libsvm.svm_model;
importca.uwo.csd.ai.nlp.libsvm.svm_parameter;
importjava.io.IOException;
importjava.util.logging.Level;
importjava.util.logging.Logger;
importutils.DataFileReader;

public class Teste {
public static void main(String[] args) {
try {
        String arquivos[]=new String[2];
arquivos[0]="C:\\data\\iris\\iris.train";  // inserindo dados de treinamento
arquivos[1]="C:\\data\\iris\\iris.test";  //inserindo dados de teste

//lendo os dados de treinamento
Instance[] treino = DataFileReader.readDataFile(arquivos[0]);

//configuração de parâmetros do algoritmo e do kernel
svm_parameterparametros = new svm_parameter();
parametros.svm_type=0;
parametros.C=32;
parametros.gamma=0.5;

//configurando o kernel RBF
KernelManager.setCustomKernel(new RBFKernel(parametros));

        //criando um modelo
svm_model modelo = SVMTrainer.train(treino, parametros);

//lendo os dados de teste
Instance[] teste = DataFileReader.readDataFile(arquivos[1]);

//realizando a classificação
double[] resultado = SVMPredictor.predict(teste, modelo, true);

intcont = 0;
//imprimindo as predições
for (double r : resultado {
cont++;
System.out.printf("objeto [%d] %.1f\n",cont, r);
}

    } catch (IOException ex) {
Logger.getLogger(Teste.class.getName()).log(Level.SEVERE, null, ex);
    }

  }
}//main
```

Fonte: https://github.com/syeedibnfaiz/libsvm-java-kernel/blob/master/src. (Adaptado pela autora)

De uma forma resumida, a seguinte classe *Teste.java* recebe dois argumentos: o *dataset* de treino, de teste. Depois, o arquivo de treinamento é convertido em um vetor do tipo *Instance*, formada por um tipo primitivo *double* denominado *label* e por um objeto rotulado como *data*, portanto, cada posição deste vetor possui um objeto *Instance*. Por exemplo, uma linha do *dataset Iris* (conjunto de dados já pré-processado muito conhecido em AM) é formada por 1 1:-0.555556 2:0.25 3:-0.864407 4:-0.916667, o *label* terá valor 1 e o objeto *data* será formado por 1:-0.555556 2:0.25 3:-0.864407 4:-0.916667. Em seguida, é feito a configuração do *kernel* desejado, neste exemplo foi usado o *kernel RBF.* Na sequência procedeu-se a configuração dos parâmetros. Posteriormente, é realizado o treinamento dos dados; a conversão dos dados de teste em um vetor do tipo *Instance*; o processo de classificação e o resultado da predição é impresso na tela.

Apêndice B – Valores das sete métricas para a situação do dano 1

Comparação do RMS	Variação do RMSD	CCDM	Distância Euclidiana	Similaridade de Cossenos	Norma H2	Norma H infinito
-0.0004021924568620 16	0.00625580793073277	0.0003186645913891 84	0.0563022713765949	0.0000637334331762274	-0.0007276231641948 48	0.0002561953433798 39
-0.0000992936495159968	0.00745668508279506	0.0005987619754977 97	0.0671101657451556	0.0000965232455057974	-0.0001850483912294 81	-0.0005825750236385 61
-0.0001909473611372 28	0.00413042747489725	0.0001082106857700 8	0.0371738472740752	0.0000268053126907253	-0.0003389083370760 44	-0.0001812536401471 36
0.0001653344909340 33	0.00317645099075481	0.0000077525562287 1285	0.0285880589167933	0.0000159605557139564	0.0002944323035487 95	0.0009786492645689 2
0.0021198170290279 7	0.010225651357878	0.0012868688131679 7	0.0920308622209017	0.0001973884808030 85	0.0042015289163578	0.0145360488994984
-0.0005279285887119 79	0.00465075574743161	0.0002052126660457 32	0.0418568017268845	0.0000350664885153673	-0.0009572751921722 03	-0.0000591757800107 175
-0.0005279285887119 79	0.00465075574743161	0.0002053126660457 32	0.0418568017268845	0.0000350664885153673	-0.0009572751921722 03	-0.0000591757800107 175
-0.0005738642013809 47	0.00607398619932526	0.0003100637070787 56	0.0546658757939273	0.0000595285041450344	-0.0010360594922776	0.0011007828472167 1
0.0000874105779054002	0.00736882166937764	0.0004579773304880 24	0.0663193950243988	0.0000891784091171521	0.0001584188869826 26	-0.0002002656878544 42
0.0007642340829551 18	0.00690870820179615	0.0003864229918085 77	0.0621783738161653	0.0000809058511972438	0.0014148180734064 2	-0.0007842689295609 7
0.0004511484304945 41	0.00570373265268973	0.0003217598839135 86	0.0513335938742076	0.0000551770565313525	0.0008331752690097 11	0.0005573687642504 99
-0.0004224652927824 48	0.00707075061726063	0.0003453015292626 85	0.0636367555553457	0.0000790503655551111	-0.0007528916030085 01	-0.0011062937250534 6
0.0000525189024368 49	0.0098955176777721	0.0012753172891213 4	0.0890596590999489	0.0001788997371381 38	0.0001003878812183 18	0.0014208981760748 3
0.0008934094824368 8	0.00707621008310185	0.0004131096273604 53	0.0636858907479167	0.0000800323448096885	0.0016089303883058 3	0.0028811685483600 7
-0.0005010418150341 87	0.00751641981095877	0.0005204685138816 56	0.0676477782986289	0.0000919537560514216	-0.0009064136969341 55	-0.0029213178040406 6
0.0004262795871006 2	0.0107350061594268	0.0008954274801523 67	0.0966150554348408	0.0001803180904015 32	0.0007544069506154 53	-0.0011339643318370 6
0.0005676502490511 44	0.0103644555935997	0.0010975255490501 8	0.0932801003423969	0.0001820155465804 83	0.0010459927297180 3	-0.0014768399853450 8
0.0000129591005932905	0.0124421960224686	0.0014283183481421 7	0.111979764202217	0.0002655202186347 96	0.0000240014037028176	-0.0028799376104989 5
0.0016005236611920 1	0.00996442084210768	0.0008550318924097 63	0.0896797875789692	0.0001558058503947 63	0.0028685938017627 8	-0.0007802587633243 1
0.0007428111201365 22	0.0109118895043186	0.0009855971755566 85	0.098207005538867	0.0002089173271033 4	0.0013937549120883 2	-0.0005180151076352 42
-0.0014596135218895	0.0157758694543058	0.0020494965741624 2	0.141982825088752	0.0003994564723011 07	-0.0026298529602419 2	0.0011383002510244 6
0.0038625136731484 9	0.019846388001342	0.0035584688071931 3	0.178617492012078	0.0006720676053457 72	0.0072479495083750 3	0.0180927065489399
0.0031175429430551 22	0.0141886670168451	0.0013974702038490 6	0.127698003151606	0.0003056704773720 66	0.0055532613214902 8	-0.0022413894011089
-0.0004461651233501 62	0.0104861413390389	0.0008880574311498 36	0.0943752720513497	0.0001765674888725 14	-0.0008060357515597 2	-0.0004829994319175 57
-0.0001214606715800 31	0.0047451966005008 56	0.0001525818092157 35	0.042706769445077	0.0000359838989341 474	-0.0002172401311841 41	0.0001205087167444 05
0.0002714874487271 81	0.0052645842352 91	0.0002975773039377 57	0.0473812585411762	0.0000478432360705439	0.0005049861824978 72	0.0002463063225044 42
0.0005100229655677 91	0.0058399885809738 2	0.0004013389686743 36	0.0525598972287644	0.0000573080476643506	0.0009382118889627 84	-0.0004961174354234 2
-0.0001028026356753 03	0.0038918190318794 9	0.0001061439668350 59	0.0350263712869154	0.0000236954339919437	-0.0001819240067924 07	-0.0004732826396294 37
0.0001002819304258 83	0.0118177137297634	0.0011077965718231 4	0.106359423567871	0.0002246481362234 71	0.0001798590127403 17	0.0009230556749125 76
-0.0012785321085494 5	0.0106383802169971	0.0007806537626201 28	0.0957454219529736	0.0001823155344661 26	-0.0023143240595832 4	0.0015189817901727 4
0.019811609385398 7	0.0662428859091907	0.017100970057843 1	0.596185973182716	0.0075290178614296 8	0.037727741538275 5	0.132920631684711
0.0045506544868498 8	0.0235917375072863	0.0032570504861592 1	0.212325637565577	0.0008432412340555 25	0.0080405436060284 5	-0.0017167609569025 1
0.0001599894177691 8	0.0038354610053025 2	0.0001149734768037 81	0.0345191490477227	0.0000236848989425553	0.0002870303089269 92	0.0003579261033981 32
-0.0002991917504812 09	0.0097347570021085 5	0.0007367956495097 92	0.087612813018977	0.0001532581949850903	-0.0005384848186763 73	-0.0020624977564398 2
0.0001683181944571 73	0.0064032459796390 9	0.0003983327135056 45	0.0576292138167518	0.0000741474241942042	0.0003201654553099 13	0.0003857472527014 63
-0.0001256096180195 51	0.0044375829964718 5	0.0001316255261780 7	0.0399382469682466	0.0000303565126241745	-0.0002206679741687 8	-0.0013264469584394 2
-0.0009794528257677 24	0.0110816596027293	0.0008458512659751 39	0.099734936424564	0.0001956895823829 19	-0.0017569594935593 3	-0.0012962111331418 7
0.0027531232550542 8	0.0160377600755035	0.0021422417914256 4	0.144339840679531	0.0004233174944312 52	0.0050573654898798 7	0.0014495314680630 4
0.0008893082829677 38	0.010134394615833	0.0005800011118981 54	0.0912095515424973	0.0001556764898098 53	0.0015531810520226 3	0.0008122821441708 39
0.0003016229895755 9	0.0050379769645008 8	0.0001853720718804 7	0.0453417926805079	0.0000400803401426897	0.0005364839329995 16	-0.0003030430409541 81
-0.0017309541330575 7	0.0156411698104222	0.0018080930362940 3	0.1407705282938	0.0003808074465919256	-0.0030747383164938 9	0.0014895850957705 7
-0.0003707971173142 48	0.0143653205797346	0.0025991818563824 2	0.129287885217611	0.0003795975441872 6	-0.0007111699948937 315	-0.0006476860187212 88
-0.0001123288792520 55	0.0119568996573338	0.0013809269201549	0.107612096901604	0.0002380099535878 03	-0.0002049977405871 19	0.0041318357754670 5
0.0001551693264397 75	0.0113463365256355	0.0010060530579548 7	0.10211702873072	0.0002043219224543 42	0.0002764415214696 64	-0.0001030114728727 4
0.0001833757605734 71	0.0113077432631623	0.0009606163535121 89	0.101769689368461	0.0002058170956648 06	0.0003290085279571 73	-0.0015662249934951 6
0.0198302218097122	0.0770959209433578	0.0334702584663725	0.69386328849022	0.0087492044785397 7	0.0345950423730455	0.0787689627711304
-0.0007135250917954 53	0.0076612755977083 9	0.0004656581983187 86	0.0689514803793755	0.0000918256318676036	-0.0012684464240469 6	0.0007272614015346 823
-0.0003207760276607 85	0.0155435616141405	0.0015893598913283 6	0.139892054527265	0.0003707224604547 75	-0.0005622188012143 33	0.0012777680273080 6
-0.0000412041226014459	0.0033752519360509 7	0.0001055051525038 5	0.0303772674244587	0.0000185309264494338	-0.0000743269894561208	-0.0014102413410689 3
0.0012794228615609 5	0.0089679491640573	0.0006663904266136828	0.0807277423176515	0.0001284319646857012	0.0023069987367174 6	0.0007320558298931 08
0.0030493080729201 4	0.0178975489956353	0.0034609659972343 3	0.161077940960718	0.0005617715584119 72	0.0057788409096546 9	0.0138398755937057
-0.0007478632184122 48	0.0084857543439271 6	0.0004782856439522 48	0.0763717890953445	0.0001158592184696 83	-0.0013477138011368 1	-0.0019786643559684 2
0.0037268220407670 2	0.0158827533660089	0.0019526978656167 7	0.14294478029408	0.0003954554453700 08	0.0067655033983140 5	0.0022657520722584 3
0.0163214732947837	0.0739710207112087	0.0357771421038767	0.665739186400879	0.0081842318644411 7	0.028526922877808	0.0632920257228605
0.0005009981795074 39	0.0109295289073888	0.0009857493110809 75	0.0983657601664919	0.0002005419103559	0.0009185641105383 32	-0.0021232153139005 3
0.0009323201395249 34	0.0047879131900194 3	0.0001681005569434 96	0.0430912187101748	0.0000370071579792741	0.0017064736693598	0.0008351653970166 68
0.0104975956035542	0.0532416833786955	0.0258177253130112	0.47917515040826	0.0046979402128738 4	0.0193056230428155	0.0627866097592582
-0.0000066765483653918 9	0.0093558539650093 5	0.0008703774860302 49	0.0842026856850842	0.0001446119208852 18	-0.0000121363573708654	-0.0038355288723474
0.0010316185119654 3	0.0089437003781359 8	0.0007842079571335 2	0.0804933034032238	0.0001314086517482 1	0.0018807390517019 2	-0.0034815064370010 9
0.0202388920120772	0.0682174992024282	0.0182084736637049	0.613957492821854	0.0076607311610329 4	0.0377223392893938	0.1309124290707 97
0.0008424572923779 74	0.0087662447008095 8	0.0004156547420854 92	0.0788962023072862	0.0001151422536637 14	0.0014640750243288 8	0.0003393716394344 27
-0.0008051849887243 95	0.0085443306479873 1	0.0006078713714338 326	0.0768989758318858	0.0001161280614867 53	-0.0014436151803135 5	0.0008611198623602 6
0.0018978269644252	0.0142048846200874	0.0016449811875066 7	0.127843961580786	0.0003478984134740 46	0.0035497654080806 03	0.0001169770912288 31
0.0005125955854332 62	0.0081180393474853	0.0004570907797192 27	0.0730782354127368	0.0001051834478429 38	0.0009170350101684 61	0.0002209728390197 84
-0.0005405935511792 24	0.0050430113234429 7	0.0002244068168471	0.0453871019109867	0.0000411207449494411	-0.0009789061408538 15	0.0000678502025625027
0.0002745586100237 9	0.0078309473790241 4	0.0005862725590894 96	0.0704785264112172	0.0001010306004596 28	0.0004985641321778 77	-0.0029331343168651
-0.0004046415955111 901	0.0055529837135810 3	0.0003009405328450 18	0.0499768534222292	0.0000505769999986638	-0.0007351083073341 26	0.0005046985048329 23
0.0049086264119335 7	0.0236476080119023	0.0032438341719717 2	0.212828472107121	0.0008537616573383 03	0.0087300315626430 1	-0.0003694763667836 83
0.0001359116298260 69	0.0029301337470158 9	0.0000796209728870 512	0.026371203723143	0.0000140565240809298	0.0002461712748904 52	-0.0002732368425625 7
-0.0018855158859999 4	0.0190765367322565	0.0024503064966868 7	0.171688830590308	0.0005656200263089 99	-0.0033463037885815	0.0017140100969409 4
0.0007746774929116 02	0.0056590086875192	0.0002753415228068 38	0.0509310782077673	0.0000535494143218962	0.0014291251336923 1	0.0018533579572229 7
-0.0002754294042039 97	0.0040854423187176 4	0.0001237803616961 35	0.0367689808684587	0.0000274372539477152	-0.0005006733033136 84	0.0002711273977601 4

Fonte: própria autora.

Apêndice C - Arquivo completo formatado nas configurações da LibSVM referente ao dano 1

1	1:-0.000402192456862016	2:0.00625580793073277	3:0.000318664591389184	4:0.0563022713765949	5:0.0000637334331762274	6:-0.000727623164194848	7:0.000256195343379839
1	1:-0.0000992936495159968	2:0.00745668508279506	3:0.00059876197497797	4:0.067110165745155 6	5:0.0000965232455057974	6:-0.000185048391229481	7:-0.000582575023638561
1	1:-0.000190947361137228	2:0.00413042747489725	3:0.000108621068577008	4:0.0371738472740752	5:0.000026805312690725 3	6:-0.000338908337076044	7:-0.000181253640147136
1	1:0.000165334490934033	2:0.00317645099075481	3:0.000077525562287128 5	4:0.0285880589167933	5:0.000015960555713956 4	6:0.000294432303548795	7:0.00097864926456892
1	1:0.00211981702902797	2:0.010225651357878	3:0.00128686881316797	4:0.0920308622209017	5:0.000197388480803085	6:0.0042015289163578	7:0.0145360488994984
1	1:-0.000527928588711979	2:0.00465075574743161	3:0.000205312666045732	4:0.0418568017268845	5:0.000035064885153673	6:-0.000957275192172203	7:-0.000059175780010717 5
1	1:-0.000527928588711979	2:0.00465075574743161	3:0.000205312666045732	4:0.0418568017268845	5:0.000035064885153673	6:-0.000957275192172203	7:-0.000059175780010717 5
1	1:-0.000573864201380947	2:0.00607398619932526	3:0.000310063707087 56	4:0.0546658757939273	5:0.000059528504145 0344	6:-0.00103605949922776	7:0.00110078284721671
1	1:0.000087410577905400 2	2:0.00736882166937764	3:0.000457977330488024	4:0.0663193950243988	5:0.000089178409117521	6:0.000158418886982626	7:-0.000200265687854442
1	1:0.000764234082955118	2:0.00690870820179615	3:0.000386422991808577	4:0.0621783738161653	5:0.000080905851197243 8	6:0.00141481807340642	7:-0.00078426892956097
1	1:0.000451148430494541	2:0.00570373265268973	3:0.000321759883913586	4:0.0513335938742076	5:0.000055177056531352 5	6:0.000833175269009711	7:0.000557368764250499
1	1:-0.000422465292782448	2:0.00707075061726063	3:0.000345031529262685	4:0.0636367555553457	5:0.000079050365551111	6:-0.000752891603008501	7:-0.00110629372505346
1	1:0.000052518902436849	2:0.0098955176777721	3:0.00127531728912134	4:0.0890596590999489	5:0.000178899737138138	6:0.000100387881218318	7:0.00142089817607483
1	1:0.00089340948243688	2:0.00707621008310185	3:0.000413109627360853	4:0.0636858907479167	5:0.000080032344809 6885	6:0.00160893038830583	7:0.00288116854836007
1	1:-0.000501041815034187	2:0.00751641981095877	3:0.000520468513881656	4:0.0676477782986289	5:0.000091953756051421 6	6:-0.000906413696934155	7:-0.00292131780404066
1	1:0.00042627958710062	2:0.0107350061594268	3:0.000895427480152367	4:0.0966150554348408	5:0.000180318090401532	6:0.000754406950615453	7:-0.00113396433183706
1	1:0.000567650249051144	2:0.0103644555935997	3:0.00109752554905018	4:0.0932801003423969	5:0.000182015546580483	6:0.00104599272971803	7:-0.00147683998534508
1	1:0.000012959100593290 5	2:0.0124421960224686	3:0.00142831834814217	4:0.111979764202217	5:0.000265520218634796	6:0.000024001403702817 6	7:-0.00287993761049895
1	1:0.00160052366119201	2:0.00996442084210768	3:0.000855031892409763	4:0.0896797875789692	5:0.000155805850394763	6:0.00286859380176278	7:-0.00078025876332431
1	1:0.000742811120136522	2:0.0109118895043186	3:0.000985597175556685	4:0.098207005538867	5:0.000208917332710334	6:0.00139375491208832	7:-0.000518015107635242
1	1:-0.00145961352188 95	2:0.0157758694543058	3:0.00204949657416242	4:0.141982825088752	5:0.000399456472301107	6:-0.00262985296024192	7:0.00113830025102446
1	1:0.00386251367314849	2:0.019846388001342	3:0.00355846880719313	4:0.178617492012078	5:0.000672067605345772	6:0.00724794950837503	7:0.0180927065489399
1	1:0.00311754294305122	2:0.0141886670168451	3:0.00139747020384906	4:0.127698003151606	5:0.000305670477372066	6:0.00555326132149028	7:-0.0022413894011089
1	1:-0.000446165123350162	2:0.0104861413390389	3:0.000888057431149836	4:0.0943752720513497	5:0.000176567488872514	6:-0.000800603575155972	7:-0.000482999431917557
1	1:-0.000121460671580031	2:0.00474519660500856	3:0.000152581809215735	4:0.042706769445077	5:0.000035983898341474	6:-0.000217240131184141	7:0.000120508716744405
1	1:0.000271487448727181	2:0.00526458428235291	3:0.000297577303937757	4:0.0473812585411762	5:0.000047843236070 5439	6:0.000504986182497872	7:0.000246306322504442
1	1:0.000510022965567791	2:0.00583998858097382	3:0.000401338968674336	4:0.0525598972287644	5:0.000057308047664 3506	6:0.000938121888962784	7:-0.000496111743542342
1	1:-0.000102802635675303	2:0.00389181903187949	3:0.000106143966835059	4:0.0350263712869154	5:0.000023695433991 9437	6:-0.000181924006792407	7:-0.000473282639629437
1	1:0.000100281930425883	2:0.0118177137297634	3:0.00110779657182314	4:0.106359423567871	5:0.000224648136223471	6:0.000179859012740317	7:0.000923055674912576
1	1:-0.00127853210854945	2:0.0106383802169971	3:0.000780653726260128	4:0.0957454219529736	5:0.000182315534466126	6:-0.00231432405983124	7:0.00151898179017274
2	1:0.0198116093853987	2:0.0662428859091907	3:0.0171009700578431	4:0.596185973182716	5:0.00752901786142968	6:0.0377277415382755	7:0.132920631684711
1	1:0.00455065448684988	2:0.0235917375072863	3:0.00325705048615921	4:0.212325637565577	5:0.000843241234055325	6:0.00804054360602845	7:-0.00171676095690251
1	1:0.00015998941776918	2:0.00383546100530252	3:0.000114973476803781	4:0.0345191490477227	5:0.000023684898425553	6:0.000287303038926992	7:0.000357926103398132
1	1:-0.000299191750481209	2:0.00973475700210855	3:0.000736795649509792	4:0.087612813018977	5:0.000153258194850903	6:-0.000538484818876373	7:-0.00206249775643982
1	1:0.000168318194457173	2:0.00640324597963909	3:0.000398332713505645	4:0.0576292138167518	5:0.000074147424194204 2	6:0.000320165455309913	7:0.000385747252701463
1	1:-0.000125609618019551	2:0.00443758299647185	3:0.000131625261780 7	4:0.0399382469682466	5:0.000030356512624 1745	6:-0.00022066797416878	7:-0.00132644695843942
1	1:-0.000979452825767724	2:0.0110816596027293	3:0.000845851265975139	4:0.099734936424564	5:0.000195689582382919	6:-0.00175695949355933	7:-0.00129621113314187
1	1:0.00275312325505428	2:0.0160377600755035	3:0.00214224179142564	4:0.144339840679531	5:0.000423317494431252	6:0.00505736548987987	7:0.00144953146806304
1	1:0.000889308282967738	2:0.010134394615833	3:0.000580001111898154	4:0.0912095515424973	5:0.000155676489809853	6:0.00155318105202263	7:0.000812282144170839
1	1:0.00030162298957559	2:0.00503797696450088	3:0.000185372027188047	4:0.0453417926805079	5:0.000040080340142689 7	6:0.000536848932999516	7:-0.000303043040954181
1	1:-0.00173095413305757	2:0.0156411698104222	3:0.00180809303629403	4:0.1407705282938	5:0.000380074465919256	6:-0.00307473831649389	7:0.00148958509577057
1	1:-0.000370797117314248	2:0.0143653205797346	3:0.00259918185638242	4:0.129287885217611	5:0.000379597544187 26	6:-0.000711699948937315	7:-0.000647686018721288
1	1:-0.000112328879252055	2:0.0119568996557338	3:0.00138092692015 49	4:0.107612096901604	5:0.000238009953587803	6:-0.000204997740587119	7:0.00413183577546705
1	1:0.000155169326439775	2:0.0113463365256355	3:0.00100605305795487	4:0.10211702873072	5:0.000204321922454342	6:0.000276441521469364	7:-0.00010301147287274
1	1:0.000183375670573471	2:0.0113077432631623	3:0.000960616353512189	4:0.101769689368461	5:0.000205817095664806	6:0.000329008527957173	7:-0.00156622499349516
2	1:0.0198302218097122	2:0.0770959209433578	3:0.0334702584663725	4:0.69386328849022	5:0.00874920447853977	6:0.0345950423730455	7:0.0787689627711304
1	1:-0.000713525091795453	2:0.00766127559770839	3:0.000465658198318786	4:0.0689514803793755	5:0.000091825631867 6036	6:-0.00126844642404696	7:0.000727614015346823
1	1:-0.000320776027660785	2:0.0155435616141405	3:0.00158935989132836	4:0.139892054527265	5:0.000370722460454775	6:-0.000562218801214333	7:0.00127776802730806
1	1:-0.000041204122601445 9	2:0.00337525193605097	3:0.000105505152250385	4:0.0303772674244587	5:0.000018530926449433 8	6:-0.000074326989456126 8	7:-0.00141024134106893
1	1:0.00127942286165095	2:0.00896974914640573	3:0.000663904266136828	4:0.0807277423176515	5:0.000128431964857012	6:0.00230699873671746	7:0.000732055829893108
1	1:0.00304930807292014	2:0.0178975489956353	3:0.00346096599723433	4:0.161077940960718	5:0.000561771558411972	6:0.00577884090965469	7:0.0138398755937057
1	1:-0.000747863218412248	2:0.00848575434392716	3:0.000478285643952248	4:0.0763717890953445	5:0.000011585921846968 3	6:-0.00134771380113681	7:-0.00197866435596842
1	1:0.00372682204076702	2:0.0158827533660089	3:0.00195269876561677	4:0.142944780294 08	5:0.000395455445370008	6:0.00676550339831405	7:0.00226575207225843
2	1:0.0163214732947837	2:0.0739710207112087	3:0.0357771421038767	4:0.665739186400879	5:0.00818423186444117	6:0.028526922877808	7:0.0632920257228605
1	1:0.000500998179507439	2:0.010929528907388	3:0.000985749311080975	4:0.0983657601664919	5:0.0002005419103559	6:0.000918564110538332	7:-0.00212321531390053
1	1:0.000932320139524934	2:0.00478791319001943	3:0.000168100556943496	4:0.0430912187101748	5:0.000037007157979274 1	6:0.0017064736693598	7:0.000835165397016668
2	1:0.0104975956035542	2:0.0532416833786955	3:0.0258177253130112	4:0.47917515040826	5:0.00469794021287384	6:0.0193056230428155	7:0.0627866097592582
1	1:-0.000006676548365391 89	2:0.00935585396500935	3:0.000870377486030249	4:0.0842026856850842	5:0.000144611920885218	6:-0.000012136357370865 4	7:-0.0038355288723474
1	1:0.00103161851196543	2:0.00894370037813598	3:0.00078420795713352	4:0.0804933034032238	5:0.000131408651748 21	6:0.00188073905170192	7:-0.00348150643700109
2	1:0.0202388920120772	2:0.0682174992024282	3:0.0182084736637049	4:0.613957492821854	5:0.00766073116103294	6:0.0377223392893938	7:0.130912429070797
1	1:0.000842457292377974	2:0.00876624470080958	3:0.000415654742085492	4:0.0788962023072862	5:0.000115142253663714	6:0.00146407502432888	7:0.000339371639434427
1	1:-0.000805184988724395	2:0.00854433064798731	3:0.000607913714338326	4:0.0768989758318858	5:0.000116128061486753	6:-0.00144361518031355	7:0.00086111986236026
1	1:0.0018978296944252	2:0.0142048846200874	3:0.00164498118750667	4:0.127843961580786	5:0.000347898413474046	6:0.00354976540808603	7:0.000116977091228931
1	1:0.000512595585433262	2:0.00811980393474853	3:0.000457090779719227	4:0.0730782354127368	5:0.000105183447842938	6:0.000917035010168461	7:0.000220972839019784
1	1:-0.000540953551179224	2:0.00504301132344297	3:0.00022440681684 71	4:0.0453871019109867	5:0.000041120744949441 1	6:-0.000978906140853815	7:0.000067850202562502 7
1	1:0.00027455861002379	2:0.00783094737902414	3:0.000586272559089496	4:0.0704785264112172	5:0.000101030600459628	6:0.000498564132177877	7:-0.00293311343168651
1	1:-0.000404641595511901	2:0.00555298371358103	3:0.000300940532845018	4:0.0499768534222292	5:0.000050576999986638	6:-0.000735108307334126	7:0.000504698504832923
1	1:0.00490862641193357	2:0.0236476080119023	3:0.00324383417197172	4:0.212828472107121	5:0.000853761657338303	6:0.00873003156264301	7:-0.000369476366783668
1	1:0.000135911629826069	2:0.00293013374701589	3:0.000079620972887051 2	4:0.026371203723143	5:0.000014056524080929 8	6:0.000246171274890452	7:-0.000273423684256257
1	1:-0.00188551588599994	2:0.0190765367322565	3:0.00245030649668687	4:0.171688830590308	5:0.000565620026308999	6:-0.00334603307885815	7:0.00171401009694094
1	1:0.000774677492911602	2:0.00565900868975192	3:0.000275341522806838	4:0.0509310782077673	5:0.000053549414321896 2	6:0.00142912513369231	7:0.00185335795722297
1	1:-0.000275429404203997	2:0.00408544231871764	3:0.000123780361696135	4:0.0367689808684587	5:0.000027437253947715 2	6:-0.000500673303313684	7:0.000271127397760 14

Fonte: própria autora.