

PONTIFÍCIA UNIVERSIDADE CATÓLICA  
DO RIO DE JANEIRO



**Maurilio Gomes Donin de Souza**

**Detecção de Trincas a partir de imagens  
utilizando Aprendizado de Máquina**

**Projeto de Graduação**

Projeto de Graduação apresentado ao Departamento de Engenharia  
Mecânica da PUC-Rio

Orientador: Helon Vicente Hultmann Ayala / DEM  
Coorientador: Walisson Chaves Ferreira Pinto / DEM

Rio de Janeiro  
Novembro de 2023

## **Agradecimentos**

Agradeço ao meu orientador e ao meu coorientador por todo o conhecimento compartilhado. Agradeço a toda a minha família, pelo suporte durante os meus estudos, em especial aos meus pais, Miltolino e Marilene, e à minha avó, Virginia.

## Resumo

### Detecção de Trincas a partir de imagens utilizando Aprendizado de Máquina

A detecção de trincas representa um desafio comum no monitoramento de saúde de estruturas (SHM - de *Structural Health Monitoring*). Frequentemente conduzido manualmente, esse processo é suscetível a erros. A utilização de imagens oferece vantagens significativas, eliminando a necessidade de contato direto com a estrutura e proporcionando uma cobertura mais abrangente. Além disso, as técnicas de aprendizado de máquina têm demonstrado eficácia na classificação de imagens, identificando a presença ou ausência de danos. O presente trabalho implementa uma *pipeline* de aprendizado de máquina composta por pré-processamento, extração de características utilizando Análise das Componentes Principais, criação dos modelos Máquina de Vetor Suporte, Árvore de Decisão e K-ésimo Vizinho Mais Próximo, otimização de hiperparâmetros e análise dos resultados, na tarefa de classificação de imagens de concreto com trincas e sem trincas. A melhor performance obtida neste trabalho foi utilizando o modelo SVM obtendo uma acurácia de 98,18%, precisão de 98,70% *recall* de 97,60% e *F1-Score* de 98,15%.

### Palavras-Chave

Aprendizado de Máquina; Extração de Características; Monitoramento de Saúde de estruturas; Detecção de Trincas.

## **Abstract**

### **Crack Detection from Images using Machine Learning**

Crack detection poses a common challenge in structural health monitoring (SHM). Often conducted manually, this process is prone to errors. The use of images offers significant advantages by eliminating the need for direct contact with the structure and providing more comprehensive coverage. Furthermore, machine learning techniques have demonstrated effectiveness in image classification, identifying the presence or absence of damage. This study implements a machine learning pipeline consisting of preprocessing, feature extraction using Principal Component Analysis, model creation for Support Vector Machine, Decision Tree, and K-nearest neighbors, hyperparameter optimization, and results analysis in the task of classifying images of concrete with and without cracks. The best performance achieved in this study was using the SVM model, which exhibited an accuracy of 98.18%, precision of 98.70%, recall of 97.60%, and F1-Score of 98.15%.

## **Keywords**

Machine Learning; Feature Extraction; Structural Health Monitoring; Crack Detection.

## Sumário

1	INTRODUÇÃO .....	8
1.1	Objetivo Geral .....	9
1.2	Objetivo Específico.....	9
2	REVISÃO BIBLIOGRÁFICA.....	10
3	DESCRIÇÃO DO ESTUDO DE CASO .....	15
4	METODOLOGIA .....	16
4.1	Pré-processamento .....	16
4.2	Extração de Características.....	18
4.3	Máquina de Vetor Suporte.....	19
4.4	Árvore de Decisão .....	20
4.5	K-ésimo Vizinho Mais Próximo .....	21
4.6	Otimização de Hiperparâmetros .....	22
4.7	Avaliação dos resultados.....	24
5	RESULTADOS.....	26
5.1	Pré-processamento .....	26
5.2	Extração de Características.....	27
5.3	Criação dos Modelos e Calibração dos Hiperparâmetros .....	27
5.4	Validação dos Modelos .....	29
5.5	Avaliação dos Resultados .....	31
5.6	Resumo dos Resultados Obtidos .....	32
6	CONCLUSÃO .....	34
6.1	Trabalhos Futuros .....	34
	Referências bibliográficas .....	35

## Lista de figuras

Figura 1 – (A) (B) (C) Exemplo de imagens rotuladas como sem trincas (ÖZGENEL; SORGUÇ, 2018).....	15
Figura 2 - (D) (E) (F) Exemplo de imagens rotuladas como com trincas (ÖZGENEL; SORGUÇ,2018).....	15
Figura 3 - Representação do fluxo de processamento dos dados implementado.....	16
Figura 4 - Organização do conjunto de dados após o pré-processamento a serem utilizados no PCA. ....	18
Figura 5 - Representação das etapas do processo iterativo de validação cruzada com 5 partições. ....	23
Figura 6 - Exemplo de matriz de confusão.....	24
Figura 7 - Exemplos de resultados obtidos no pré-processamento.....	26
Figura 8 - Exemplos de resultado obtido na etapa de extração de características. ....	27
Figura 9 - Diagrama de Caixa com os resultados obtidos para a acurácia de cada modelo.....	30
Figura 10 - Matriz de confusão do modelo SVM (A), do modelo DTC (B) e do modelo KNN (C).....	31

## Lista de tabelas

Tabela 1 - Grade de hiperparâmetros testados.....	28
Tabela 2 - Resultados obtidos no processo de otimização de hiperparâmetros. ....	28
Tabela 3 - Melhores hiperparâmetros obtidos para cada modelo.....	29
Tabela 4 - Resultados obtidos na avaliação da acurácia dos modelos. ....	30
Tabela 5 - Resultados da classificação utilizado o conjunto de testes. ....	31

## Lista de abreviaturas

ACC	Acurácia do modelo
DTC	Decision Tree Classifier
KNN	K-nearest Neighbors
P	Precisão do modelo
PCA	Principal Component Analysis
R	<i>Recall</i> do modelo
RGB	Sistema de cor aditivo vermelho, verde e azul
SHM	Structural Health Monitoring
SVM	Support Vector Machine



# 1 INTRODUÇÃO

A detecção de trincas representa um desafio comum no monitoramento de saúde de estruturas (SHM - de *Structural Health Monitoring*), pois a sua presença pode comprometer a integridade, aumentar a suscetibilidade a falhas e diminuir a durabilidade da estrutura, resultando em riscos significativos para sua utilização (PHILIP et al, 2023). A identificação dessas trincas é, em geral, realizada por meio de inspeção visual, em que profissionais examinam minuciosamente a superfície da estrutura em busca dessas falhas. No entanto, esse processo manual enfrenta desafios consideráveis, especialmente ao distinguir trincas estruturais de elementos como texturas, irregularidades, marcas e detalhes da própria estrutura. Além disso, a inspeção visual pode ser ineficiente e propensa a erros quando aplicada em grandes áreas (ZOUBIR et al, 2022).

O aprendizado de máquina é uma área da ciência da computação que visa tornar computadores capazes de aprender a partir de dados previamente selecionados por meio da construção de modelos matemáticos (GERON, 2019). Essa técnica, de grande relevância no contexto atual, busca explorar a capacidade dos sistemas computacionais de analisar informações e identificar padrões, a fim de tomar decisões e realizar tarefas complexas de forma autônoma.

A aplicação de algoritmos de aprendizado de máquina supervisionado em problemas de classificação tem possibilitado o processamento de grandes volumes de dados (CHANDRA; BEDI, 2021). Existem diversos algoritmos de aprendizado supervisionado, entre eles podemos destacar a Máquina de Vetor Suporte (SVM – de *Support Vector Machine*), Árvore de Decisão (DTC – de *Decision Tree Classifier*) e K-ésimo Vizinho Mais Próximo (KNN – de *K-Nearest Neighbors*). Esses algoritmos têm sido utilizados em diversas tarefas, inclusive na classificação de imagens e em SHM (GRYLLIAS; ANTONIADIS, 2012), (AMARNATH et al., 2013), (TANG et al., 2023).

Além disso, para lidar com a complexidade dos dados de imagens de trincas, faz-se necessário a utilização de técnicas de redução de dimensionalidade. Essa abordagem consiste em transformar os dados em um espaço de menor dimensão, preservando as informações mais relevantes e descartando aquelas de menor importância. A redução de dimensionalidade permite simplificar a representação dos dados tornando o conjunto mais compacto e eficiente e, ainda, mantendo o máximo de variabilidade nos dados, facilitando o processo de classificação e análise (XIAO, 2010).

Nesse contexto, técnicas de redução de dimensionalidade e aprendizado de máquina supervisionado têm sido empregadas na classificação de imagens de trincas (PHILIP et al., 2023), (ZOUBIR et al., 2022). Tendo como o objetivo automatizar o processo de identificação e superar as limitações dos métodos tradicionais, os quais nem sempre dependem exclusivamente de uma avaliação quantitativa, tornando-se assim suscetíveis a erros.

### **1.1 Objetivo Geral**

O objetivo deste trabalho é realizar a redução da dimensionalidade de um conjunto de dados de imagens de concreto utilizando a técnica de Análise de Componentes Principais (PCA – de Principal Component Analysis) e classificar os dados de acordo com a presença ou ausência de trincas utilizando diferentes algoritmos de aprendizado de máquina supervisionado, são elas: SVM, DTC e KNN, a fim de realizar uma análise comparativa destes modelos.

### **1.2 Objetivo Específico**

Neste trabalho, será realizada primeiramente a construção de um conjunto de dados a partir das imagens originais. Este conjunto de dados será utilizado na aplicação do PCA com intuito de reduzir a dimensionalidade dos dados, preservando as informações relevantes para a classificação das trincas. Em seguida, os modelos de aprendizado de máquina serão calibrados para o problema e treinados para se obter um modelo capaz de distinguir as amostras com trincas das amostras sem trincas. Ao final, busca-se obter um sistema capaz de identificar com precisão a presença de trincas em um conjunto de dados, contribuindo para a detecção e manutenção preventiva de estruturas.

## 2 REVISÃO BIBLIOGRÁFICA

O monitoramento da saúde de estruturas vem sendo aplicado a partir de diferentes técnicas de inspeção que visam determinar o estado de uma estrutura. Essa área de pesquisa pode ser considerada como multidisciplinar devido as diferentes abordagens para a identificação de danos na estrutura em diversos materiais. Técnicas utilizando sensores piezoelétricos e visão computacional e imagens vem sendo estudadas (YEUM; DYKE, 2015), (TANG et al., 2023).

Os sistemas de monitoramento de estruturas baseados em imagens apresentam diversas vantagens, como: o fato de não ser necessário o contato com a estrutura, distanciamento da estrutura e área de cobertura grande (YEUM; DYKE, 2015). Trabalhos recentes vêm aplicando essa técnica em diferentes análises.

Yeum e Dyke, propuseram um sistema de inspeção baseado em imagens de trincas capaz de analisar uma grande quantidade de imagens sem controle de ângulo, posicionamento e sem necessidade de calibração capaz de identificar trincas próximas de parafusos em pontes, este sistema utilizou técnicas de detecção de objetos e alcançou uma taxa de acerto de 98,7% (YEUM; DYKE, 2015). Ho et al introduziram um sistema que utilizou um algoritmo que combinou técnicas de aprimoramento de imagens e redução de ruídos com PCA para identificar danos na superfície de cabos, os resultados alcançaram uma taxa de acerto de 93,3% utilizando imagens produzidas durante um experimento (HO et al., 2013).

Além disso, técnicas baseadas em dados também foram utilizadas, Yang et al utilizaram um algoritmo de rede totalmente convolucional para classificar trincas em estruturas de concreto e pavimentos de asfaltos a partir de imagens, o algoritmo foi capaz de identificar a presença de trincas, assim como estimar o comprimento e largura das trincas, na identificação de trincas a estratégia apresentou uma acurácia de 97,96% (YANG et al., 2018). Kim et al fizeram uso de técnicas de extração de características de imagens, combinadas com algoritmos de SVM e redes neurais convolucionais a fim de classificar, a partir de imagens, a presença, ou não, de trincas em superfícies de concreto, como resultado foi verificado que a estratégia utilizando redes neurais convolucionais apresentou o melhor resultado, alcançando uma acurácia de 98% (KIM et al., 2019).

Como evidenciado, técnicas baseadas em dados vêm sendo utilizadas no monitoramento da saúde de estruturas, diferentes algoritmos de aprendizado podem ser utilizados nesta tarefa.

O algoritmo de aprendizado de máquina supervisionado SVM tem sido amplamente aplicado e estudado ao longo das últimas décadas, apresentando resultados promissores em uma variedade de domínios. Os primeiros trabalhos nesse campo foram realizados por Vapnik em 1963, introduzindo a noção de Hiperplano de margem máxima e um algoritmo de treinamento para calcular esse hiperplano (VAPNIK, 1963). Esses estudos pioneiros estabeleceram os princípios fundamentais do SVM, abrindo caminho para avanços posteriores.

Uma das principais extensões do SVM ocorreu em 1995, quando Vapnik e Cortes propuseram o conceito de "*kernel trick*", permitindo que o modelo lidasse com dados não lineares (CORTES; VAPNIK, 1995). Essa abordagem inovadora possibilitou a aplicação do SVM em problemas de classificação mais complexos e ampliou seu alcance em diversas áreas.

Ao longo dos últimos anos, diversos trabalhos têm aplicado com sucesso algoritmos de SVM mostrando sua eficiência em problemas de classificação em uma variedade de domínios (CHANDRA; BEDI, 2021), (PHILLIPS, 1998).

No campo da engenharia, aplicações desse algoritmo em classificação de sinais são empregadas frequentemente. Gryllias e Antoniadis utilizaram uma classificação em dois estágios utilizando SVM para, primeiramente, separar os sinais entre normais e falhos e posteriormente classificar o tipo de falhas em rolamentos em um ambiente industrial a partir dos sinais de vibração da máquina. Os resultados obtidos mostraram ótima capacidade de classificação chegando a 100% de taxa de sucesso, utilizando um universo de 37 dados experimentais e obtidos em ambiente industrial (GRYLLIAS; ANTONIADIS, 2012).

O algoritmo de aprendizado supervisionado SVM tem sido aplicado em trabalhos relacionados à classificação de imagens. P.Philips explorou com sucesso a aplicação de algoritmos baseados em SVM em um problema multiclasse de identificação facial, alcançando uma taxa de acerto de 78%. O trabalho mostrou que o valor encontrado superou os resultados de estratégias não baseadas em aprendizado de máquina (PHILLIPS, 1998). Priya et al propuseram um método que utilizou processamento de imagens e PCA para extração de características na tarefa de classificação de imagem de plantas por meio do reconhecimento das folhas, foram utilizados modelos de SVM e KNN para classificar as imagens, os resultados mostraram que o modelo SVM performou melhor nesta tarefa de classificação com uma acurácia de 94,5% (PRIYA et al., 2012).

A aplicação de algoritmos de aprendizado de máquina supervisionados no monitoramento da saúde de estruturas tem ajudado na identificação de defeitos com

mais precisão. Jac Fredo et al, utilizaram diferentes características extraídas a partir de imagens de compósitos com entalhes como *inputs* para o modelo SVM para classificar o dano presente, resultando em uma acurácia de 96,6% (JAC FREDO et al., 2019). Coelho et al desenvolveram uma metodologia para classificar a extensão de trincas em diferentes categorias de tamanho em juntas de encaixe utilizando SVM a partir de sinais de ondas guiadas (COELHO et al., 2008).

Da mesma forma, Liu et al. desenvolveram uma metodologia para detecção e classificação de trincas em superfícies de túneis a partir de imagens, combinando a análise de intensidade da trinca com SVM, classificando entre com trincas, sem trincas e um tipo intermediário, os resultados obtidos alcançaram uma acurácia de 86,3% (LIU et al, 2002).

Outro exemplo de algoritmo de aprendizado supervisionado é a árvore de decisão. Este algoritmo vem sendo aplicado em diversas tarefas, especialmente na tarefa de classificação. Os trabalhos iniciais de Breiman et al em 1987 implementaram uma estratégia recursiva de particionamento dos dados em busca da melhor divisão dos dados na árvore de decisão (BREIMAN et al., 1987). Quinlan em 1986 introduz um algoritmo para selecionar os melhores atributos para dividir o conjunto de dados utilizando o conceito de ganho de informação (QUINLAN, 1986). Esses trabalhos iniciais estabeleceram a base para desenvolvimentos e implementações futuras.

No campo da engenharia diversos trabalhos vêm aplicando técnicas baseadas em árvore de decisão para a classificação de dados. Jimenez et al classificaram a presença de delaminações em pás de turbinas eólicas a partir de sinais de ondas guiadas utilizando diferentes algoritmos de aprendizado de máquina na classificação dos dados. Foram considerados 6 cenários de delaminação, relacionados a extensão dos danos, foram utilizados diferentes modelos, redes neurais, KNN e árvore de decisão, neste trabalho o modelo de árvore de decisão alcançou acima de 80% de taxa de acerto (ARCOS JIMÉNEZ et al., 2018). Amarnath utilizou uma abordagem de aprendizado de máquina para detectar a presença de falhas em rolamentos, a análise foi feita a partir de sinais de som e a classificação foi feita utilizando o algoritmo de árvore de decisão, os resultados obtidos mostraram que este modelo resultou em uma acurácia de 95% (AMARNATH et al., 2013).

Além disso, Firdaus utilizou diferentes algoritmos de aprendizado de máquina, SVM, KNN e árvore de decisão, em uma tarefa de classificação multiclasse relacionada a segmentação de regiões utilizando imagens de alta resolução obtidas por satélite, neste trabalho os resultados mostraram que o algoritmo de árvore de decisão resultou

em uma acurácia de 68,4%, sendo esta a menor em relação aos outros modelos (FIRDAUS, 2018)

Além dos algoritmos citados, o algoritmo K-ésimo Vizinho Mais Próximo, também vem sendo aplicado em trabalhos envolvendo aplicações de monitoramento da saúde de estruturas e engenharia. Os trabalhos iniciais envolvendo esse algoritmo foram desenvolvidos inicialmente por Evelyn Fix e Joseph Hodges em 1951 (FIX; HODGES, 1989), posteriormente diversos trabalhos utilizaram esse algoritmo na tarefa de classificação.

Tang utilizou o algoritmo KNN combinado com estratégias de extração de características para detectar a presença de danos iniciais em pás de turbinas eólicas a partir de sinais de ruído de operação das turbinas. Os resultados mostraram boa capacidade de classificação do modelo KNN, alcançando uma acurácia de 98,9% (TANG et al., 2023). Glowacz utilizou o algoritmo KNN em um sistema de monitoramento de falhas iniciais em motores DC utilizando sinais acústicos, a estratégia resultou em uma taxa de acerto acima de 80% (GLOWACZ, 2015).

Além disso, a utilização da análise de componentes principais como método para a extração de característica de imagens também é amplamente utilizada em problemas de classificação. Ao reduzir a dimensionalidade dos dados, o PCA facilita a identificação de padrões e a separação de regiões com características distintas. Um estudo seminal realizado por Turk e Pentland em 1991 introduziu o conceito de "*eigenfaces*" como uma aplicação prática do PCA na área de reconhecimento facial (TURK; PENTLAND, 1991). Os autores demonstraram que é possível representar imagens de faces como combinações lineares de componentes principais, permitindo a identificação e reconstrução de rostos com base nesses componentes, abordagem que abriu caminho para que outros sistemas de reconhecimento facial fossem desenvolvidos.

Diversos trabalhos no campo da engenharia têm incluído o PCA em *pipelines* de aprendizado de máquina supervisionado. Jiménez, empregou PCA como uma das técnicas de extração de características em uma *pipeline* de aprendizado de máquina que tinha como objetivo detectar a presença de lama e classificar em 5 diferentes níveis em pás de turbinas eólicas a partir de sinais de ondas guiadas. Utilizando o modelo KNN combinado com PCA resultou nos melhores resultados, atingindo uma acurácia de 100% (JIMÉNEZ, 2019).

No trabalho de Viana, PCA foi utilizado em uma *pipeline* de aprendizado de máquina como técnica de extração de características e redução de dimensionalidade, em conjunto com o modelo KNN para classificar imagens de mãos (VIANA, 2020). Ho e Kim utilizaram PCA no processamento de imagens em uma estratégia que aplicou

aprendizado de máquina na classificação danos na superfície de cabos (HO et al., 2013). Priya et al, utilizaram PCA para reduzir a dimensionalidade das características obtidas a partir das imagens de plantas a serem utilizadas em um algoritmo de aprendizado supervisionado, com a finalidade de balancear a complexidade computacional e acurácia da metodologia (PRIYA et al., 2012).

Estes exemplos destacam de maneira significativa a relevância e o vasto potencial dos algoritmos de aprendizado supervisionado na execução de tarefas complexas. Os estudos previamente mencionados não apenas exemplificam a notável capacidade desses algoritmos, mas também evidenciam a amplitude de suas aplicações no contexto da engenharia. Adicionalmente, os estudos que empregaram o PCA evidenciam sua eficiência como técnica de redução de dimensionalidade em problemas diversos, especialmente aqueles relacionados ao processamento de imagens.

### 3 DESCRIÇÃO DO ESTUDO DE CASO

Como visto, a avaliação da condição de componentes estruturais é essencial, já que estes estão sujeitos a defeitos e falhas. Porém, identificar defeitos em estruturas, pode ser uma tarefa complexa se considerarmos que estruturas podem possuir grandes formatos, difícil acesso e outros fatores que podem dificultar uma inspeção manual (PHILIP et al., 2023), (ZOUBIR et al., 2022).

Dessa forma, o presente trabalho utiliza os dados disponibilizados por Özgenel (ÖZGENEL; SORGUÇ, 2018). Este conjunto de dados é composto por 40.000 imagens de superfícies de concreto, das quais 20.000 apresentam trincas e o restante são imagens de superfícies livres de trinca, as imagens possuem dimensão de 227 x 227 *pixels*. Na Figura 1 pode ser visto exemplos de imagens sem trincas já na Figura 2 há exemplos de imagens com trincas. Este conjunto de dados foi gerado a partir de 458 imagens de alta resolução utilizando o método proposto por Zhang (ZHANG et al., 2016), nenhuma estratégia de aumento de dados como, rotação ou espelhamento foi aplicada.

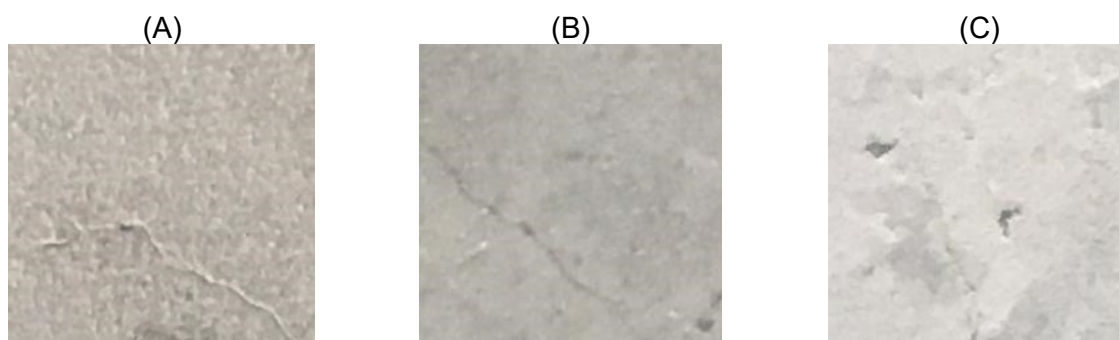


Figura 1 – (A) (B) (C) Exemplo de imagens rotuladas como sem trincas (ÖZGENEL; SORGUÇ, 2018).

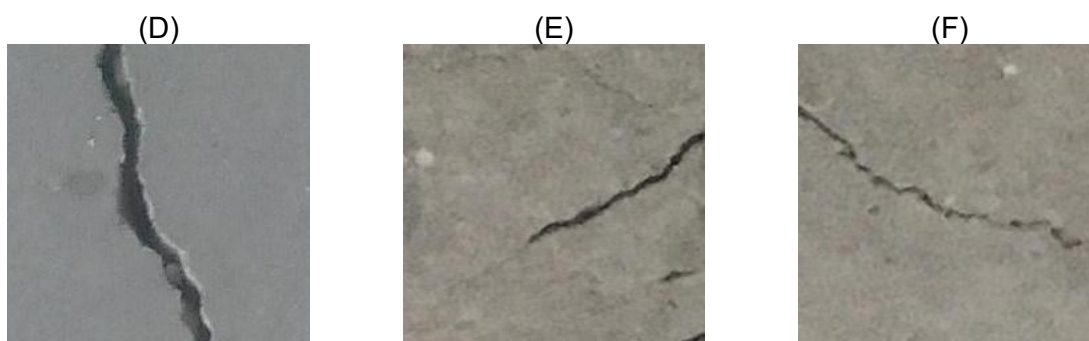


Figura 2 - (D) (E) (F) Exemplo de imagens rotuladas como com trincas (ÖZGENEL; SORGUÇ, 2018).



## 4 METODOLOGIA

Neste trabalho será utilizado um *pipeline* de aprendizado de máquina visando abordar o problema previamente exposto. Este *pipeline* foi dividido em: Pré-processamento, extração de características, criação de modelos e calibração de hiperparâmetros, validação dos modelos e avaliação dos resultados. A fim de se atingir o objetivo de ter um modelo capaz de distinguir as amostras com trincas das amostras sem trincas com boa precisão. Na Figura 3 temos uma representação do fluxo de processamento dos dados a partir das imagens originais até chegar aos resultados da classificação.

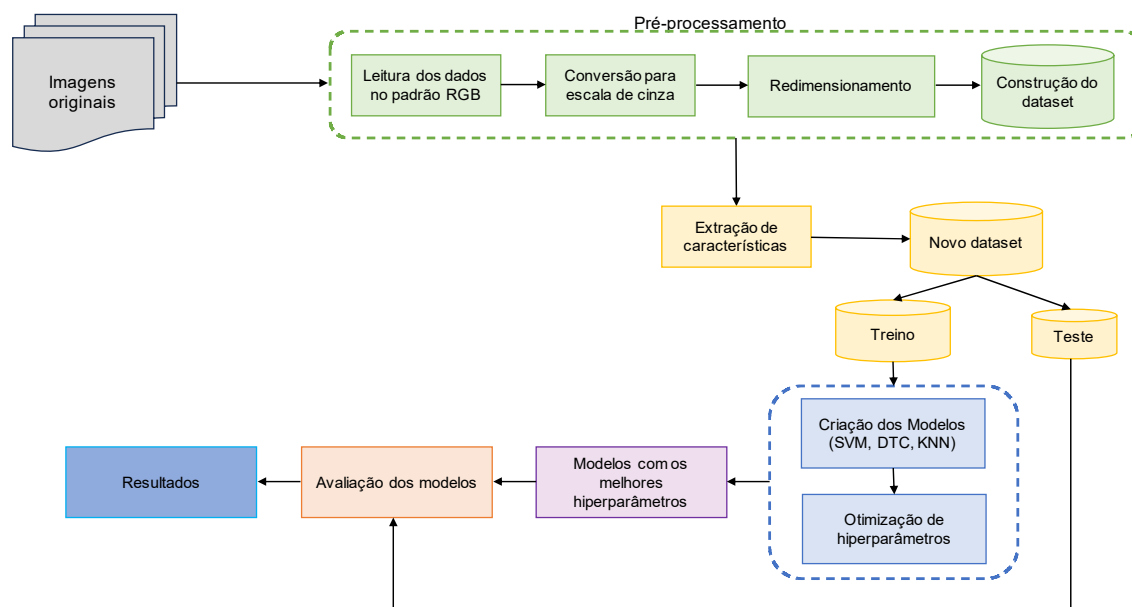


Figura 3 - Representação do fluxo de processamento dos dados implementado

### 4.1 Pré-processamento

Na etapa de pré-processamento serão feitos tratamentos nas imagens disponibilizadas para que a utilização seja mais eficiente computacionalmente, como conversão para escala de cinza e normalização, conforme pode ser visto na Figura 3 esta etapa é composta por 4 processos.

A representação de uma imagem será feita, inicialmente, como uma matriz de tamanho  $M \times N$  *pixels* dentro do padrão RGB. Neste padrão, há uma matriz  $X$  ( $M \times N$ ) com a intensidade, dada em uma escala de 0 a 255, para as cores vermelho, verde e azul, respectivamente. Sendo assim, para cada *pixel* em uma determinada posição, existem 3 valores de intensidade de cor a serem considerados.

Para facilitar a manipulação, podemos converter a imagem em cor para uma versão em escala de cinza, por meio de uma média simples, dessa forma o valor de intensidade média de cor  $V(m, n)$  em uma determinada posição é dado por:

$$V(m, n) = \frac{\sum_{c=1}^3 I_c(n, m)}{3} \quad (1)$$

onde,  $I_c$  é o valor da intensidade de cada cor do padrão RGB para uma dada posição descrita por  $(n, m)$ . Desse modo, teremos uma única matriz de tamanho  $(M \times N)$  para representar a imagem, em que cada elemento, representa a intensidade média das cores que compunham cada pixel em uma escala de 0 a 255.

Considerando o tamanho original das imagens disponibilizadas de 227 x 227 *pixels*, teremos uma matriz com 51.529 elementos. A fim de se obter uma versão reduzida, representativa da imagem original, foi aplicado um processo de redimensionamento, utilizando a técnica de interpolação de Lanczos, este método faz uma interpolação multivariada, utilizando o produto da função seno cardinal (*sinc*) normalizada e uma constante  $a$ , chamada de janela de Lanczos. Desta forma, é gerado uma janela de convolução  $K(i, j)$  tal que (TURKOWSKI, 1990), (BURGER; BURGE, 2009):

$$k(i, j) = P(i)P(j),$$

$$P(u) = \begin{cases} \text{sinc}(u) \cdot \text{sinc}\left(\frac{u}{a}\right) & -a < u < a \\ 0 & \text{caso contrário} \end{cases} \quad (2)$$

onde,  $i$  e  $j$  são as coordenadas dos pixels a serem interpolados e  $a$  é janela de Lanczos e  $P(u)$  representa a implementação para uma coordenada  $u$ .

Ainda na etapa de processamento, foi possível, também, representar a matriz da imagem com um vetor  $x^T$   $n$ -dimensional, onde  $n = m.n$ , representando a matriz  $X$  concatenada. Dessa forma, a imagem pode ser descrita como um vetor no espaço  $R^n$ . Então, um espaço de  $\mathbf{Zx}$  é um espaço multidimensional que contém todas as imagens do conjunto de dados (KITANI; 2006).

Desse modo, foi possível construir um *dataset* em forma de matriz, em que cada linha representa uma imagem de forma planificada, este *dataset* será utilizado para as etapas seguintes.

## 4.2 Extração de Características

Nesta etapa, a extração de características será feita utilizando a Análise das Componentes Principais com a finalidade de reduzir a dimensionalidade dos dados mantendo o máximo de variabilidade. Então, teremos um conjunto de dados reduzido representativo dos dados originais de forma a tornar o treinamento dos modelos mais eficientes.

Considerando a matriz obtida no procedimento anterior para a elaboração do PCA teremos,  $P$  objetos e  $X$  variáveis que representam os dados de intensidade de cor em cada posição da imagem. Uma representação dessa organização pode ser vista na Figura 4.

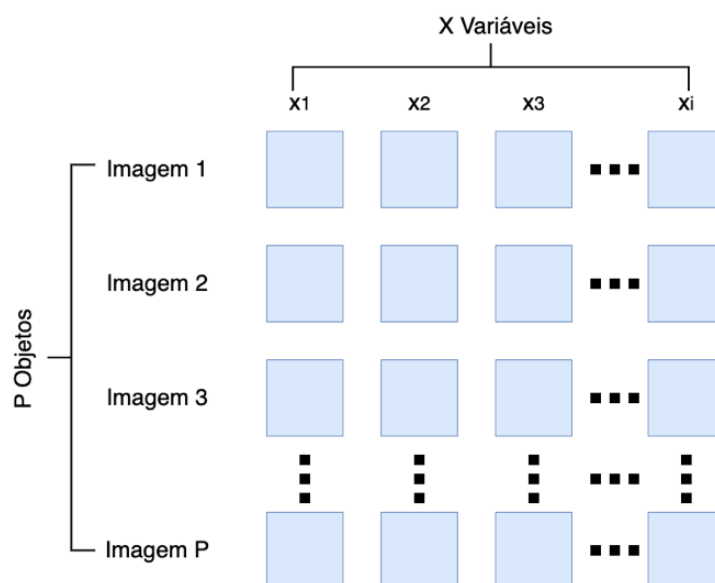


Figura 4 - Organização do conjunto de dados após o pré-processamento a serem utilizados no PCA.

As componentes principais são definidas como combinação linear das variáveis onde cada eixo está na direção de uma determinada variância dos dados, ou seja, as componentes principais são os eixos de um sistema de coordenadas obtidos através da rotação específica onde os eixos estão na direção de variância máxima (KITANI; 2006).

Devido ao tamanho do *dataset* a abordagem tradicional do PCA torna-se custosa computacionalmente, já que é necessário armazenar todos os dados na memória para a execução do método. Diante disso, foi utilizada uma metodologia de PCA incremental

(LIPPI; CECCARELLI, 2019), (ARTAC et al., 2002), que processa determinado número de linhas por vez. Por fim, o conjunto de dados original será representado por um conjunto de dados reduzido.

### 4.3 Máquina de Vetor Suporte

Nesta etapa é definido o modelo SVM a ser utilizado no problema de classificação binária. O modelo SVM é uma técnica de aprendizado de máquina supervisionado que busca encontrar um hiperplano que melhor separa as duas classes em um *dataset*. Dentro deste problema, o modelo pretende maximizar a margem entre as classes enquanto minimiza a normal do vetor peso (VAPNIK, 1963).

Considere um conjunto de dados de treinamento com  $m$  amostras, cada uma pertencendo a uma das duas classes, representadas como  $(x_i, y_i)$ , onde  $x_i$  é o vetor características da  $i$ -ésima amostra e  $y_i$  é o rótulo da classe correspondente ( $y_i \in \{-1, 1\}$ ).

O objetivo do SVM é encontrar um hiperplano representado pela Equação (3) (DU; HE; SUN, 2016):

$$w \cdot x + b = 0 \quad (3)$$

onde,  $w$  é o vetor de pesos e  $b$  é o termo de viés.

A função de decisão para uma entrada  $x$  é dada pela Equação (4):

$$f(x) = w \cdot x + b \quad (4)$$

Então, o problema de otimização do modelo SVM envolve encontrar  $W$  e  $B$  que maximizem a margem entre as duas classes, a margem  $M$  é definida na Equação (5):

$$M = \frac{2}{\|w\|} \quad (5)$$

A otimização está sujeita a restrição de que  $y_i(w \cdot x_i + b) \geq 1$  para todo  $i = 1, \dots, m$  [36].

Sendo assim, o problema de otimização pode ser formulado como (NOVA FRANCO, 2019):

$$\text{minimizar } \frac{1}{2} |w|^2 \quad \text{sujeito a } y_i(w \cdot x_i + b) \geq 1 \quad \text{para todos } i = 1, \dots, m \quad (6)$$

Este é um problema de otimização quadrática, e vários métodos, como o algoritmo *Sequential Minimal Optimization* (SMO), podem ser usados para encontrar os valores ótimos de  $W$  e  $B$ .

Além do caso linear discutido acima, pode-se estender a capacidade do classificador SVM utilizando as funções de *Kernel*. Essas funções permitem que o SVM lide com dados que não são linearmente separáveis no espaço original. O chamado *Kernel Trick*, substitui o produto escalar  $w \cdot x$  por  $K(x_i, x)$ , em que  $K$  é a função de *Kernel*. Existem diferentes funções de *Kernel*, alguns exemplos são (LORENA; CARVALHO, 2003):

- Linear:

$$K(x_i, x) = x_i \cdot x \quad (7)$$

- Polinomial

$$K(x_i, x) = (x_i \cdot x + c)^d \quad (8)$$

- Função de Base Radial:

$$K(x_i, x) = \exp\left(-\frac{|x_i - x|^2}{2\sigma^2}\right) \quad (9)$$

onde  $c, d, \sigma$  são parâmetros que influenciam o comportamento do kernel.

O uso de funções de *kernel* amplia a capacidade do SVM para lidar com relações não lineares nos dados, permitindo a classificação de conjuntos de dados mais complexos.

#### 4.4 Árvore de Decisão

O modelo DTC é baseado em um modelo de árvore de decisão, um modelo hierárquico, não paramétrico, para aprendizado supervisionado. O modelo se baseia em uma estrutura de árvore hierárquica composta por um nó raiz, ramos, nós internos e nós folhas.

Considerando o problema de classificação binária onde a variável alvo é ( $Y \in \{0,1\}$ ). O objetivo é aprender uma árvore  $T$  que mapeia o vetor características  $x$  para um rótulo de classe previsto  $\hat{y}_i$ . Para isso, a árvore é construída recursivamente particionando o espaço de características.

Seja:

$E_c = \{X_1, X_2, \dots, X_n\}$  o espaço de características, e  $x_i$  o vetor de características

$D = (x_1, y_1), (x_2, y_2), \dots, (x_m, y_m)$  o conjunto de dados, onde  $x_i$  é o vetor de características e  $y_i$  é o rótulo correspondente.

A Árvore de Decisão é construída recursivamente selecionando características e limiares para dividir o conjunto de dados em cada nó. As decisões de divisão são determinadas otimizando um critério, como Ganho de Informação,  $GA(D, A)$ , ou Impureza de Gini,  $IG(D, A)$ .

Estes critérios são definidos como (BREWKA, 1996):

$$GA(D, A) = E(D) - \sum_{v \in \text{valores}(A)} \frac{|D_v|}{|D|} \cdot E(D_v) \quad (10)$$

Onde,  $D$  é o conjunto de dados,  $A$  é uma característica candidata para dividir,  $D_v$  é o subconjunto de  $D$  para o qual a característica  $A$  tem valor  $v$ , e  $E(D)$  é a entropia do conjunto de dados, definida por (SOMVANSI et al., 2016):

$$E(D) = - \sum_{c \in \text{classes}} P(c) \cdot \log_2(P(c)) \quad (11)$$

onde,  $P(c)$  é a proporção de instâncias na classe  $c$  no conjunto de dados.

A impureza de Gini é definida por (SOMVANSI et al., 2016):

$$IG(D, A) = 1 - \sum_{v \in \text{valores}(A)} \left( \frac{|D_v|}{|D|} \right)^2 \quad (12)$$

Similar ao Ganho de Informação,  $D_v$  representa o subconjunto de  $D$  para o qual a característica  $A$  tem valor  $v$ .

Sendo assim, a árvore pode ser representada como um conjunto de regras (DU et al., 2016).

#### 4.5 K-ésimo Vizinho Mais Próximo

O modelo KNN é um modelo não paramétrico de aprendizado supervisionado. O modelo se baseia na ideia de que características similares tendem a ter rótulos similares, uma representação simplificada de um modelo KNN pode ser visto na Figura 7. No processo de treinamento, o modelo armazena todo o conjunto de dados de treino como referência e ao realizar classificações o modelo calcula a distância entre o input e todos os exemplos dos dados de treino, usando alguma forma de medida de distância (FIX; HODGES, 1989).

Seja  $D$  o conjunto de dados, onde cada ponto é representado por  $(x_i, y_i)$  e  $y_i$  pertencente ao conjunto  $\{1, -1\}$ .

A distância entre dois pontos  $x_i$  e  $x_j$  é definida como  $Dist(x_i, x_j)$ .

Para realizar a classificação a partir de um  $x_{teste}$  o modelo procede da seguinte forma:

Cálculo das distâncias entre  $x_{teste}$  e cada ponto  $x_i$  no conjunto de dados de treinamento usando a métrica de distância escolhida, como por exemplo a distância euclidiana, definida por (BREWKA, 1996):

$$Dist_{Euclidiana}(x_{test}, x_i) = \sqrt{\sum_{j=1}^c (x_{test,j} - x_{i,j})^2} \quad (13)$$

onde,  $c$  é o número de características em cada ponto do conjunto de dados.

O passo seguinte consiste em ordenar as distâncias calculadas em ordem crescente e retorna os índices  $c$  que representam as menores distâncias. Dessa forma,  $N_k$  é o conjunto de índices que possui os  $c$  índices de menor distância:

Pode-se, ainda, atribuir pesos para os vizinhos com base em suas distâncias. Vizinhos mais próximos podem ter uma influência maior. Uma função peso  $w(Dist)$  atribui um peso a cada vizinho baseado em sua distância do ponto de teste. Uma função de peso usual é função de ponderação inversa da distância, dada por:

$$w(Dist) = \frac{1}{dist} \quad (14)$$

Dessa forma, a classe majoritária é definida por uma soma ponderada dos rótulos da classe (BREWKA, 1996):

$$\hat{y}_i = \text{sign} \left( \sum_{i \in N_k} w(Dist_{Euclidiana}(x_{test}, x_i)) \cdot y_i \right) \quad (15)$$

Por fim, o modelo é parametrizado por  $K$ , o número de vizinhos a ser considerado e a função peso a ser utilizada.

## 4.6 Otimização de Hiperparâmetros

Essa fase do processamento visa aprimorar o desempenho dos modelos apresentados nos dados do problema. Para se obter resultados eficientes na utilização de modelos de aprendizado de máquina é necessário ajustar os hiperparâmetros de cada modelo. Hiperparâmetros são variáveis de configuração dos modelos que

dependem do conjunto de dados em que serão utilizados. Dessa forma, é necessário realizar um procedimento com o objetivo de se obter a melhor conjunto de hiperparâmetros para cada modelo.

A otimização dos hiperparâmetros de cada modelo é realizada por meio de um procedimento conhecido como Busca em Grade (*Grid Search*). Essa abordagem envolve a exploração sistemática de todas as combinações de hiperparâmetros fornecidas em uma grade predefinida, identificando a configuração mais eficaz para cada modelo (BELETE; HUCHAIAH, 2022).

Adicionalmente, em conjunto com a Busca em Grade, a avaliação do desempenho dos modelos incorpora uma estratégia de validação cruzada com  $K$  partições (*folds*), neste caso,  $K = 5$ . Cada combinação de hiperparâmetros de cada modelo será treinada utilizando  $K-1$  partições e a partição restante será utilizada para a avaliação (SANTOS, 2022). Uma representação esquemática desse processo pode ser vista na Figura 5.

Conjunto de Dados					
	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5
Etapa 1	Teste	Treino	Treino	Treino	Treino
Etapa 2	Treino	Teste	Treino	Treino	Treino
Etapa 3	Treino	Treino	Teste	Treino	Treino
Etapa 4	Treino	Treino	Treino	Teste	Treino
Etapa 5	Treino	Treino	Treino	Treino	Teste

Figura 5 - Representação das etapas do processo iterativo de validação cruzada com 5 partições.

Dessa forma, o processo de otimização de hiperparâmetros ocorre seguindo as etapas abaixo:

- 1- Definição da grade de hiperparâmetros a ser utilizada para cada modelo;
- 2- Divisão do conjunto de dados em  $K = 5$  partições para a validação cruzada;
- 3- Treinar e avaliar a performance de cada modelo para cada combinação de hiperparâmetros utilizando a estratégia de validação cruzada.



- 4- Calcular a média das métricas de avaliação de cada de cada partição para cada combinação de hiperparâmetros. Selecionar a combinação que gerou o melhor resultado.

#### 4.7 Avaliação dos resultados

A partir das predições fornecidas pelos modelos, foram determinadas métricas de avaliação dos resultados de forma a avaliar a performance de cada modelo.

A matriz de confusão é uma forma de avaliar as predições de um determinado modelo. Nesta matriz as linhas representam as classes verdadeiras do rótulo que está tentando prever, e as colunas representam a classe predita pelo classificador. Uma representação de matriz de confusão para classificação binária pode ser vista na Figura 6.

		Valor Predito	
		Sim	Não
Real	Sim	Verdadeiro Positivo (TP)	Falso Negativo (FN)
	Não	Falso Positivo (FP)	Verdadeiro Negativo (TN)

Figura 6 - Exemplo de matriz de confusão.

Utilizando a matriz de confusão é possível classificar os dados em:

- Verdadeiros Positivos (TP – de True Positive): Casos em que o modelo previu corretamente a classe positiva.
- Falsos Positivos (FP – de False Positive): Casos em que o modelo previu incorretamente a classe positiva
- Verdadeiros Negativos (TN – de True Negative): Casos em que o modelo previu corretamente a classe negativa.
- Falsos Negativos (FN – de False Negative): Casos em que o modelo previu incorretamente a classe negativa

A partir dessas classificações é possível definir outras métricas quantitativas da performance de um modelo. São elas:

$$Acc = \frac{TP+TN}{TP+TN+FP+FN} \quad (16)$$

$$P = \frac{TP}{TP+FP} \quad (17)$$

$$R = \frac{TP}{TP+FN} \quad (18)$$

$$F1 - score = 2 \times \frac{Precisão \times Recall}{Precisão + Recall} \quad (19)$$

onde, *ACC* é a acurácia do modelo, *P* é a precisão do modelo, *R* é o *recall*.

Sendo assim, a avaliação dos resultados gerais será feita por meio do cálculo da Precisão, que determina a proporção das amostras detectadas corretamente pelo modelo, Acurácia, que avalia a proporção de previsões positivas corretas em relação a todas as previsões positivas feitas pelo modelo, *Recall*, que mede a proporção de previsões positivas corretas em relação a todos os casos positivos presentes nos dados e, por fim, o *F1-Score*, uma média harmônica entre Precisão e *Recall*.

## 5 RESULTADOS

Após implementar a metodologia descrita acima foi possível analisar e avaliar o desempenho dos modelos utilizados. Os resultados obtidos em cada etapa da metodologia serão analisados abaixo.

### 5.1 Pré-processamento

Inicialmente na etapa de pré-processamento foram avaliados os resultados obtidos na etapa de pré-processamento, considerando o tamanho original das imagens originais disponibilizadas de 227 x 227 *pixels*. Após a realização do pré-processamento as imagens foram convertidas para escala de cinza e redimensionadas para um tamanho de 150 x 150 *pixels*, uma redução de 56,34% no tamanho da imagem. A Figura 7 mostra o resultado obtido nesta etapa, por meio de dois exemplos, uma imagem com e outra sem trincas. Ainda nesta etapa, todas as imagens foram agrupadas formando um *dataset* de dimensão 40.000 x 22.500.

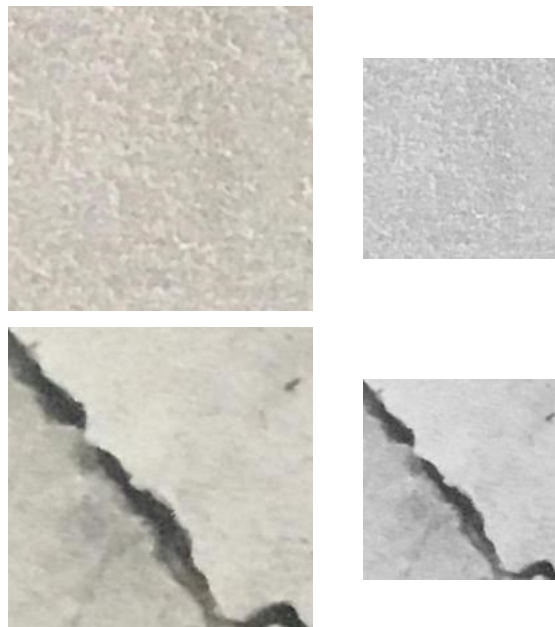


Figura 7 - Exemplos de resultados obtidos no pré-processamento.

## 5.2 Extração de Características

Seguindo para a etapa de extração de características, foi utilizada uma metodologia de PCA incremental, processando 2.000 linhas por vez. Então, foi possível reduzir a dimensionalidade do problema de 22.500 características para 300, mantendo a variância acumulada explicada pelos componentes principais maior que 90%.

Sendo assim, o conjunto de dados original será representado por um conjunto de dados reduzido de dimensão 40.000 x 300, este valor representa uma redução de dimensionalidade do problema de 98,67%.

Analisando o resultado obtido nas duas etapas iniciais, pré-processamento e redução de dimensionalidade, foi possível reduzir a quantidade de dados, diretamente relacionada ao uso de memória, em 99,42%. Isto permitiu que as próximas etapas do desenvolvimento fossem realizadas de forma mais eficiente.

Para observar os efeitos do PCA nos dados, pode-se realizar a transformação inversa, onde parte-se de uma imagem reduzida e utilizando as componentes principais obtidas pelo PCA pode-se transformar a imagem de volta ao espaço original, e dessa forma, obter uma representação gráfica da perda de informação ocorrida, conforme mostra a Figura 8.

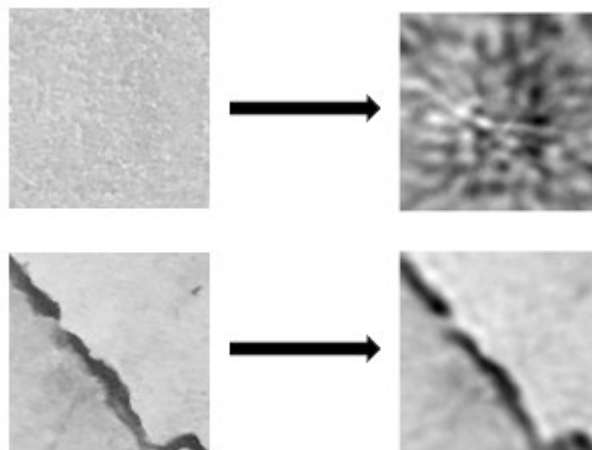


Figura 8 - Exemplos de resultado obtido na etapa de extração de características.

## 5.3 Criação dos Modelos e Calibração dos Hiperparâmetros

Nesta etapa, foram criados os modelos de aprendizado de máquina SVM, DTC e KNN. Após a criação estes modelos tiveram seus hiperparâmetros otimizados, contudo, devido ao grande número de amostras no conjunto de dados e ao

procedimento de validação cruzada aplicado na busca de grade, foi realizado a busca dos melhores hiperparâmetros em uma amostra reduzida do conjunto de dados devido a restrições computacionais. De forma a avaliar o tempo de processamento e o ganho na acurácia do modelo foi utilizado uma estratégia com 4 subconjuntos de dados de treino. Os dados de treino são compostos por 32.000 imagens, os subconjuntos utilizados contêm: 320 amostras (1%), 640 amostras (2%), 1.600 amostras (5%) e, por fim, 3.200 amostras (10%). A grade de hiperparâmetros utilizada para cada modelo pode ser vista na Tabela 1. Além disso, o tempo de processamento para cada subconjunto e a acurácia média obtida pelo modelo foram avaliadas e escolhida a combinação de hiperparâmetros que resultou na melhor acurácia média, os resultados dessa etapa podem ser vistos na Tabela 2.

Tabela 1 - Grade de hiperparâmetros testados.

<b>Modelo</b>	<b>Hiperparâmetros</b>	<b>Min</b>	<b>Max</b>	<b>Incremento</b>	<b>Escala</b>
	Parâmetro C	0,1	1000	10	Log
SVM	Parâmetro gamma do Kernel: "Scale" ou "Auto"	-	-	-	-
	Kernel: RBF ou Linear	-	-	-	-
DTC	Critério de divisão: Entropia ou Gini	-	-	-	-
	Máxima profundidade	2	16	2	Linear
	N-vizinhos	5	11	2	Linear
KNN	Função Peso: inverso da distância ou uniforme	-	-	-	-

Tabela 2 - Resultados obtidos no processo de otimização de hiperparâmetros.

<b>Modelo</b>	<b>Nº de amostras</b>	<b>Tempo [s]</b>	<b>Acurácia</b>
SVM	320	0,57	95,62%
	640	1,59	95,93%
	1.600	14,04	96,32%
	3.200	32,65	97,13%
DTC	320	1,28	85,93%
	640	2,93	90,31%
	1.600	8,05	89,43%
	3.200	15,29	91,25%
KNN	320	0,74	59,69%
	640	1,18	66,09%
	1.600	3,02	69,56%
	3.200	3,90	75,03%

Tabela 3 - Melhores hiperparâmetros obtidos para cada modelo.

<b>Modelo</b>	<b>Melhores Hiperparâmetros</b>
SVM	Parâmetro C: 10 Parâmetro gamma do Kernel: 10 Kernel: RBF
DTC	Critério de divisão: Entropia Máxima profundidade: 16
KNN	N-vizinhos: 5 Função Peso: inverso da distância

Observando a Tabela 2 vemos que há um grande acréscimo de tempo de processamento relacionado ao aumento do número de amostras em todos os modelos. No modelo de classificação SVM pode-se observar uma tendência de melhora marginal da acurácia com o aumento do número de amostras, visto que, para todos os subconjuntos o modelo resultou em uma acurácia maior que 95% e da menor amostra para a maior, foi obtido apenas uma melhora de 1,50% na acurácia. Para este modelo, a busca retornou o mesmo conjunto de hiperparâmetros independentemente do número de amostras. Para o modelo DTC, observa-se uma melhora gradativa da acurácia do modelo com o aumento do número de amostras, comparando a menor amostra com a maior vemos um acréscimo de 5,32% na acurácia, chegando a um valor máximo de 91,25%. Neste processo o conjunto de melhores hiperparâmetros sofreu variações com os diferentes subconjuntos, sendo assim, foi escolhido a combinação que produziu a melhor acurácia. Já para o modelo KNN, assim como no modelo DTC, observa-se uma melhora gradativa da acurácia do modelo com o aumento do número de amostras, resultando em um acréscimo de 15,34% na acurácia, contudo, o melhor conjunto de hiperparâmetros se manteve igual para todos os subconjuntos testados. O valor máximo da acurácia que o modelo resultou utilizando o subconjunto de 3.200 imagens foi de 75,03%, valor consideravelmente mais baixos que os encontrados para os modelos SVM e DTC.

Dessa forma, forma foi escolhido os melhores conjuntos de hiperparâmetros para serem utilizados nas etapas seguintes, os conjuntos podem ser vistos na Tabela 3.

#### **5.4 Validação dos Modelos**

Para a realização da validação dos modelos foi utilizado o melhor conjunto de hiperparâmetros obtidos para cada modelo. Os modelos tiveram sua acurácia avaliada

fazendo uso de uma estratégia de validação cruzada *K-Fold* com 10 *Folds* aplicada apenas nos dados de treino. Sendo assim, os modelos foram treinados e testados 10 vezes, para a realizar a avaliação da acurácia. Na Figura 9 podem ser encontrados os resultados obtidos para cada modelo representados em forma de diagrama de caixa e na Tabela 4 encontram-se os valores para a acurácia média, desvio padrão e intervalo de resultados de cada modelo utilizando os melhores hiperparâmetros, obtidos na etapa anterior.

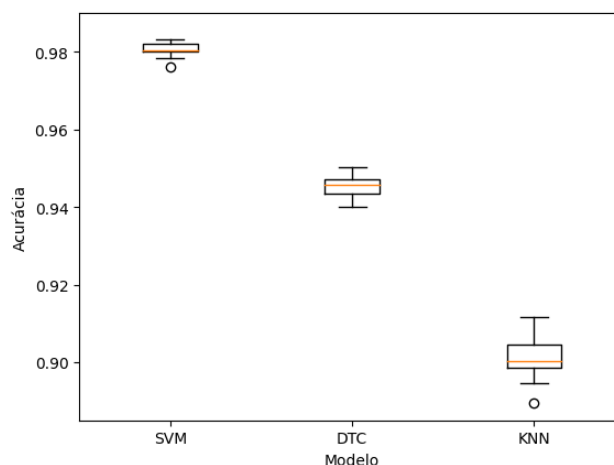


Figura 9 - Diagrama de Caixa com os resultados obtidos para a acurácia de cada modelo.

Tabela 4 - Resultados obtidos na avaliação da acurácia dos modelos.

<b>Modelo</b>	<b>Acurácia Média</b>	<b>Desvio Padrão</b>	<b>Range</b>
SVM	98,05%	0,20%	0,69%
DTC	94,54%	0,29%	1,03%
KNN	90,03%	0,59%	2,22%

A análise da Figura 9 e da Tabela 4 evidencia que o modelo SVM exibiu uma acurácia média superior, atingindo 98,05%, em contraste com o modelo DTC, que alcançou uma acurácia média de 94,54% e o modelo KNN que apresentou uma acurácia média de 90,03%. Ao avaliar a dispersão dos dados, nota-se que o modelo SVM apresenta um desvio padrão de 0,20%, com todos os resultados contidos em uma faixa de 0,69%. Em comparação, o modelo DTC exibe um desvio padrão de 0,29%, com os dados compreendidos em uma faixa de 1,03%, revelando uma dispersão ligeiramente mais ampla nos resultados. Já o modelo KNN apresentou uma dispersão mais elevada que os demais, apresentando um desvio padrão de 0,59% e uma de faixa de resultados ainda maior que o DTC de 2,22%.

Em termos gerais, os resultados indicam uma boa capacidade de classificação para os modelos. Entretanto, destaca-se que o modelo SVM demonstrou um desempenho superior.

## 5.5 Avaliação dos Resultados

Avaliando os resultados obtidos das classificações feitas por cada modelo após terem sido definidos os melhores hiperparâmetros foi utilizado todo o conjunto de dados de treino para a realização do treinamento dos modelos e avaliação de cada modelo no conjunto de testes, foi possível obter as métricas gerais do problema de classificação, assim como, a matriz de confusão. Os modelos foram treinados utilizando 80% (32.000 imagens) do conjunto de dados para treino e 20% (8.000 imagens) do conjunto de dados para teste.

Tabela 5 - Resultados da classificação utilizando o conjunto de testes.

Métrica	SVM	DTC	KNN
Precisão	98,70%	95,20%	99,96%
Acurácia	98,18%	94,74%	90,90%
Recall	97,60%	94,12%	81,64%
F1-Score	98,15%	94,65%	89,88%

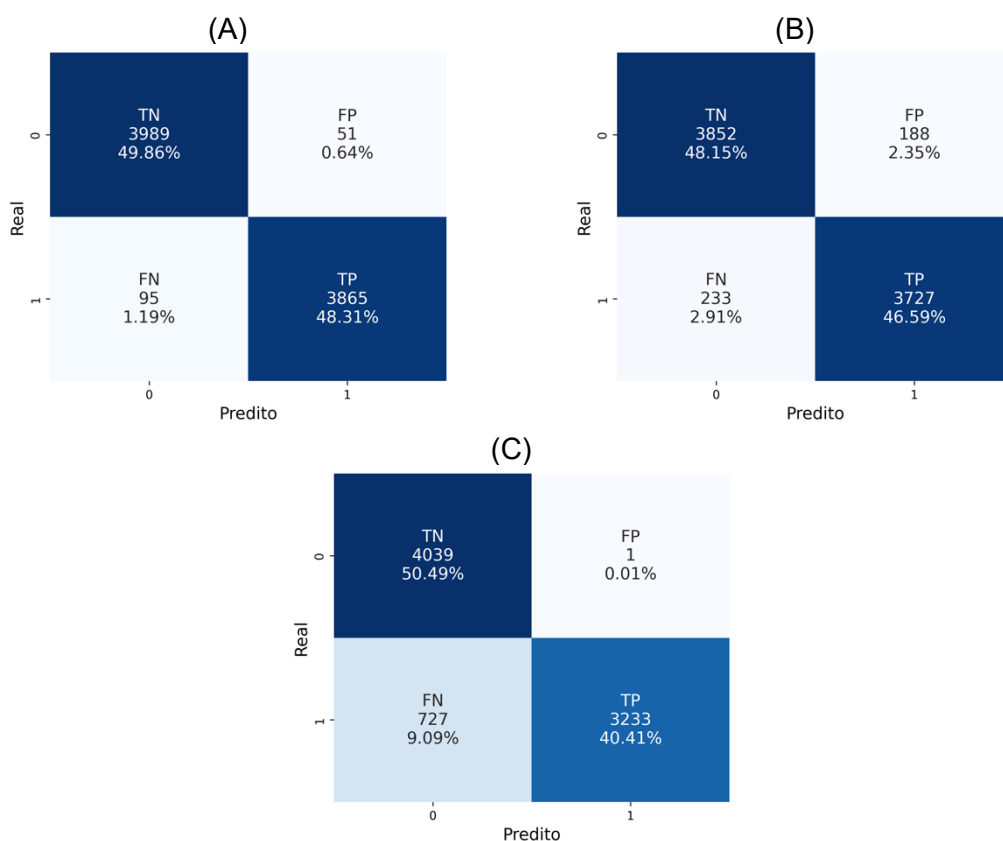


Figura 10 - Matriz de confusão do modelo SVM (A), do modelo DTC (B) e do modelo KNN (C).



Conforme apresentado na Tabela 5, o modelo SVM apresentou uma precisão de 98,69%, o que significa que o modelo classificou corretamente 98,69% das amostras. A acurácia, resultou em 98,18%. Já o *Recall* indicou que o modelo identificou corretamente 97,60% dos casos positivos. Por fim, o *F1-Score* de 98,15% indica um bom equilíbrio entre precisão e *Recall*. O modelo DTC, apresentou uma precisão de 95,20%, acurácia de 94,74%, *Recall* de 94,12% e, por fim, *F1-Score* de 94,65%. Já o modelo KNN resultou em uma precisão de 99,96%, e acurácia de 90,90%, isto quer dizer que o modelo consegue distinguir bem imagens sem trincas, contudo, o modelo apresenta uma discrepância nos valores da precisão e acurácia, indicando que o modelo não consegue identificar algumas imagens com trincas, penalizando a acurácia. Isto, também pode ser observado nos resultados do *Recall* de 81,64% e *F1-Score* de 89,88%.

Analisando as matrizes de confusão obtidas para cada modelo na Figura 10, pode-se observar que o modelo SVM é o que apresenta o menor número de falso positivo e falso negativo. O modelo DTC apresenta um resultado equilibrado, os resultados falsos positivos e falsos negativos representam 5,26% das amostras. Já o modelo KNN, identificou melhor imagens sem trinca pois apresentou um alto número de falso negativo, imagens com trincas que não foram classificadas corretamente.

Comparando o resultado dos dois modelos vemos que o modelo SVM se destaca frente aos modelos DTC e KNN por apresentar melhores métricas e pela análise das matrizes de confusão. Em geral, vemos que todos os modelos possuem alto número de verdadeiros positivos e verdadeiros negativos e baixos números de falsos positivos e falsos negativos, ainda que, no modelo KNN tenha-se uma quantidade de falso negativo de 5,29%. Os resultados também mostraram que os modelos foram melhores classificando imagens sem trincas corretamente, pois todos apresentaram uma quantidade maior de verdadeiro negativo que verdadeiro positivo.

Portanto, os resultados destacam a boa capacidade dos modelos de distinguir efetivamente entre as duas classes, destacando-se o modelo SVM por apresentar uma quantidade menor de falso positivo e falso negativo.

## 5.6 Resumo dos Resultados Obtidos

Em síntese, os resultados obtidos em cada etapa indicam que a *pipeline* implementada obteve êxito na produção de resultados satisfatórios. A dimensionalidade dos dados originais foi reduzida significativamente em 99,42%, empregando a estratégia apresentada durante a etapa de pré-processamento e extração de características por meio do PCA. Ao aplicar esses dados reduzidos aos algoritmos de aprendizado de

máquina, ficou evidente que ainda continham informações necessárias para realizar a tarefa de classificação entre imagens com trincas e sem trincas. Avaliando os resultados obtidos na classificação por cada modelo utilizando os dados de treino, destacou-se os resultados do modelo SVM, este modelo apresentou uma acurácia média de 98,05% e desvio padrão de 0,20%, superando significativamente os demais modelos. Além disso, ao classificar o conjunto de dados de testes, o modelo SVM demonstrou uma menor quantidade de classificações incorretas, apresentando precisão de 98,70%, recall de 97,60%, F1-Score de 98,15% e acurácia de 98,18%. Portanto, é possível afirmar que, nesta tarefa específica, o modelo SVM apresentou um desempenho superior em comparação aos demais.

## 6 CONCLUSÃO

Neste trabalho foi apresentado o conceito de monitoramento de saúde de estruturas no contexto da identificação de trincas, tarefa que quando realizada manualmente por meio de inspeção visual está propensa a diversas falhas (PHILIP et al., 2023), (ZOUBIR et al., 2022). Diante disso, foi demonstrado o uso de algoritmos de aprendizado de máquinas supervisionados no contexto do monitoramento da saúde de estruturas e identificação de defeitos (COELHO et al., 2008), (LIU et al., 2002), (ARCOS JIMÉNEZ et al., 2018). A partir disso, foi implementado uma *pipeline* de aprendizado de máquina para classificar um conjunto de imagens de concreto que apresentavam, ou não, trincas. Essa *pipeline* capaz de lidar com as imagens disponibilizadas foi composta pelas etapas de pré-processamento, extração de características utilizando PCA, criação dos modelos SVM, DTC e KNN e realização da calibração dos hiperparâmetros e análise dos resultados.

Através da metodologia descrita no trabalho, foi possível analisar diferentes métricas de cada modelo e determinar o mais eficaz no problema de classificação das trincas nas imagens disponibilizadas. As análises então, demonstraram que o modelo SVM destacou-se em relação aos demais, apresentando acurácia de 98,18% e precisão de 98,70% superiores, além de um equilíbrio mais favorável entre classificações falso positivo e falso negativo.

### 6.1 Trabalhos Futuros

Os resultados obtidos neste trabalho são importantes para a aplicação em trabalhos futuros. Poderá ser estudado a aplicação de técnicas de *data augmentation* com o objetivo de aprimorar o desempenho na classificação dos modelos de aprendizado de máquina (SHORTEN; KHOSHGOFTAAR, 2019). Além disso, poderá ser implementado redes neurais profundas utilizando modelos pré-treinados (ÖZGENEL; SORGUÇ, 2018), como por exemplo: AlexNet, VGG16, VGG19. Ao empregar esses modelos, seria pertinente investigar não apenas a presença ou ausência de trincas, mas também a localização dessas trincas na imagem. Essas abordagens promissoras podem contribuir de forma significativa para a realização da tarefa de identificação de danos a partir de imagens.

## Referências bibliográficas

AMARNATH, M.; SUGUMARAN, V.; KUMAR, Hemantha. **Exploiting Sound Signals for Fault Diagnosis of Bearings Using Decision Tree.**

Measurement, v. 46, n. 3, p. 1250–1256, 2013.

ARCOS JIMÉNEZ, Alfredo; GÓMEZ MUÑOZ, Carlos Quiterio; GARCÍA MÁRQUEZ, Fausto Pedro. **Machine Learning for Wind Turbine Blades Maintenance Management.** Energies, v. 11, n. 1, p. 13, 2018.

ARTAC, M.; JOGAN, M.; LEONARDIS, A. **Incremental PCA for On-Line Visual Learning and Recognition.** 2002 International Conference on Pattern Recognition. **Anais...**ago. 2002.

BELETE, Daniel Mesafint; HUCHAIAH, Manjaiah D. **Grid Search in Hyperparameter Optimization of Machine Learning Models for Prediction of HIV/AIDS Test Results.** International Journal of Computers and Applications, v. 44, n. 9, p. 875–886, 2022.

BREIMAN, Leo; FRIEDMAN, Jerome H.; OLSHEN, Richard A.; *et al.* **Classification And Regression Trees.** 1. ed. [s.l.]: Routledge, 2017.

BREWKA, Gerd. **Artificial Intelligence—a Modern Approach by Stuart Russell and Peter Norvig, Prentice Hall.** Series in Artificial Intelligence, Englewood Cliffs, NJ. The Knowledge Engineering Review, v. 11, n. 1, p. 78–79, 1996.

BURGER, Wilhelm; BURGE, Mark. **Principles of Digital Image Processing: Core Algorithms.** London: Springer, 2009.

CHANDRA, M. A.; BEDI, S. S. **Survey on SVM and their application in image classification**. International Journal of Information Technology, v. 13, n. 5, p. 1–11, out. 2021.

COELHO, C. K.; DAS, S.; CHATTOPADHYAY, A. **Binary Tree SVM Based Framework for Mining Fatigue Induced Damage Attributes in Complex Lug Joints**. (D. K. Lindner, Ed.)San Diego, California: 27 mar. 2008.

CORTES, C.; VAPNIK, V. **Support-vector networks**. Machine Learning, v. 20, n. 3, p. 273–297, set. 1995.

DU, C.-J.; HE, H.-J.; SUN, D.-W. **Object Classification Methods**. em: Computer Vision Technology for Food Quality Evaluation. [s.l.]: Elsevier, 2016, p. 87–110.

E. C. KITANI ; C. E. Thomaz. **Um Tutorial sobre Análise de Componentes Principais para o Reconhecimento Automático de Faces**. Relatório Técnico 01/2006, Departamento de Engenharia Elétrica, FEI, São Bernardo do Campo, São Paulo, Brasil, Maio 2006.

FIRDAUS, Muhammad Irsyadi. **Satellite Image Classification using Decision Tree, SVM and k-Nearest Neighbor**. Disponível em: <[https://www.academia.edu/37022326/Satellite\\_Image\\_Classification\\_using\\_Decision\\_Tree\\_SVM\\_and\\_k\\_Nearest\\_Neighbor](https://www.academia.edu/37022326/Satellite_Image_Classification_using_Decision_Tree_SVM_and_k_Nearest_Neighbor)>. Acesso em: 19 nov. 2023.

FIX, Evelyn; HODGES, J. L. **Discriminatory Analysis. Nonparametric Discrimination: Consistency Properties**. International Statistical Review / Revue Internationale de Statistique, v. 57, n. 3, p. 238, 1989.

GERON, Aurelie. **Hands-On Machine Learning with Scikit-Learn and TensorFlow**. 2.ed. Sebastopol: O'Reilly Media, 2019. 1150p.

GLOWACZ, Adam. **DC Motor Fault Analysis with the Use of Acoustic Signals, Coiflet Wavelet Transform, and K-Nearest Neighbor Classifier.** Archives of Acoustics; 2015; vol. 40; No 3; 321-327, 2015.

GRYLLIAS, K. C.; ANTONIADIS, I. A. **A Support Vector Machine approach based on physical model training for rolling element bearing fault detection in industrial environments.** Engipeneering Applications of Artificial Intelligence, v. 25, n. 2, p. 326–344, mar. 2012.

HO, Hoai-Nam; KIM, Ki-Deok; PARK, Young-Soo; *et al.* **An Efficient Image-Based Damage Detection for Cable Surface in Cable-Stayed Bridges.** NDT & E International, v. 58, p. 18–23, 2013.

JAC FREDO, A. R. *et al.* **Classification of Damages in Composite Images Using Zernike Moments and Support Vector Machines.** Composites Part B: Engineering, v. 168, p. 77–86, jul. 2019.

JIMÉNEZ, A. A. *et al.* **Dirt and Mud Detection and Diagnosis on a Wind Turbine Blade Employing Guided Waves and Supervised Learning Classifiers.** Reliability Engineering & System Safety, v.189, pag 2-12, 2019

KIM, Hyunjun; AHN, Eunjong; SHIN, Myoungsu; *et al.* **Crack and Noncrack Classification from Concrete Surface Images Using Machine Learning.** Structural Health Monitoring, v. 18, n. 3, p. 725–738, 2019.

LIPPI, V.; CECCARELLI, G. **Incremental Principal Component Analysis Exact Implementation and Continuity Corrections.** Proceedings of the 16th International Conference on Informatics in Control, Automation and Robotics. Anais...2019.

LIU, Z. *et al.* **Tunnel Crack Detection and Classification System Based on Image Processing.** (M. A. Hunt, Ed.)San Jose, CA: 14 mar. 2002.

LORENA, Ana Carolina e CARVALHO, André Carlos Ponce de Leon Ferreira. **Introdução às máquinas de vetores suporte (support vector machines)**. São Carlos: ICMC-USP.

NOVA FRANCO, Athur. **Otimização de Parâmetros de uma Máquina de Vetores de Suporte para Localização Móvel em Redes Celulares.**

Orientadora: Daniel Carvalho da Cunha. 2019. 47 f. TCC (Graduação) - Curso de Engenharia da Computação, Universidade Federal de Pernambuco, Recife, 2019.

ÖZGENEL, Ç. F.; SORGUÇ, A. G. **Performance Comparison of Pretrained Convolutional Neural Networks on Crack Detection in Buildings**. Taipei, Taiwan: 22 jul. 2018. ISARC Proceedings, p. 693–700, 2018.

PHILIP, R. E. et al. **A Comparative study on srack detection in concrete walls using transfer learning techniques**. Journal of Composites Science, v. 7, n. 4, p. 169, 18 abr. 2023.

PHILLIPS, J. **Support Vector Machines Applied to Face Recognition**. In: Proceedings of the 11th International Conference on Neural Information Processing Systems MIT Press, 1998.

PRIYA, C. A.; BALASARAVANAN, T.; THANAMANI, A. S. **An Efficient Leaf Recognition Algorithm for Plant Classification Using Support Vector Machine**. International Conference on Pattern Recognition, Informatics and Medical Engineering (PRIME-2012). Anais...mar. 2012.

QUINLAN, J. R. **Induction of decision trees**. Machine Learning, v. 1, n. 1, p. 81–106, 1986.

SANTOS, Matheus Augusto de Castro. **Desenvolvimento de um Método Automatizado para Classificação de Pneumonia em Imagens de Raio-X via Aprendizagem Profunda**. Orientadora: Dra. Liliam Berton. 2022. 90 f. TCC

(Graduação) - Curso de Engenharia da Computação, Universidade Federal de São Paulo, São José dos Campos, 2022.

SHORTEN, Connor; KHOSHGOFTAAR, Taghi M. **A survey on Image Data Augmentation for Deep Learning**. *Journal of Big Data*, v. 6, n. 1, p. 60, 2019.

SOMVANSHI, Madan; CHAVAN, Pranjali; TAMBABDE, Shital; *et al.* **A review of machine learning techniques using decision tree and support vector machine**. *In: 2016 International Conference on Computing Communication Control and automation (ICCUBEA)*. Pune: IEEE, 2016, p. 1–7.

TANG, Yaochi; CHANG, Yunchi; LI, Kuohao. **Applications of K-Nearest Neighbor Algorithm in Intelligent Diagnosis of Wind Turbine Blades Damage**. *Renewable Energy*, v. 212, p. 855–864, 2023.

TURK, M.; PENTLAND, A. **Eigenfaces for Recognition**. *Journal of Cognitive Neuroscience*, v. 3, n. 1, p. 71–86, 1 jan. 1991.

TURKOWSKI, Ken. **Filters for Common Resampling Tasks**. *In: Graphics Gems*. [s.l.]: Elsevier, 1990, p. 147–165.

VAPNIK, V.; LERNER, A. **Pattern Recognition Using Generalized Portrait Method**. *Automation and Remote Control*, v. 24, p. 774-780, 1963.

VIANA, Wandersom Moura. **Classificação de Imagens Multiespectrais da Palma da Mão Utilizando Análise de Componentes Principais**. 2020.

XIAO, B. **Principal component analysis for feature extraction of image sequence**. *2010 International Conference on Computer and Communication Technologies in Agriculture Engineering*. Anais...jun. 2010.

YANG, Xincong; LI, Heng; YU, Yantao; *et al.* **Automatic Pixel-Level Crack Detection and Measurement Using Fully Convolutional Network**.



Computer-Aided Civil and Infrastructure Engineering, v. 33, n. 12, p. 1090–1109, 2018.

YEUM, Chul Min; DYKE, Shirley J. **Vision-Based Automated Crack Detection for Bridge Inspection**. Computer-Aided Civil and Infrastructure Engineering, v. 30, n. 10, p. 759–770, 2015.

ZHANG, L. et al. **Road Crack Detection Using Deep Convolutional Neural Network**. 2016 IEEE International Conference on Image Processing (ICIP). **Anais...**Phoenix, AZ, USA: IEEE, set. 2016.

ZOUBIR, H. et al. **Concrete bridge crack image classification using histograms of oriented gradients, uniform local binary patterns, and kernel Principal Component Analysis**. Electronics, v. 11, n. 20, p. 3357, 18 out. 2022.