Aplicação de algoritmos de classificação utilizando visão computacional para análise e identificação de índices de cáries em dentes

Lucas Ferrari de Oliveira
Setor de Educação Profissional e Tecnológica
Universidade Federal do Paraná
Curitiba, Brasil
lferrarioliveira@gmail.com

Vinícius Trainotti
Setor de Educação Profissional e Tecnológica
Universidade Federal do Paraná
Curitiba, Brasil
viniciustrainotti@gmail.com

***Resumo*—Este documento visa apresentar o Trabalho de Conclusão da matéria de Visão Computacional do Curso de Especialização em Inteligência Artificial Aplicada da UFPR. O estudo apresentado realiza a classificação de diferentes níveis de cáries em dentes, a partir das características de *shape* e textura. A análise foi realizada utilizando-se *Random Forest, Support Vector Machine* e *K-Nearest Neighbors* sobre a base disponibilizada pela UFPR. A linguagem Python foi utilizada como base de desenvolvimento com auxilio das bibliotecas *OpenCV* e *SKImage*. Os melhores resultados de predição foram gerados pelo modelo de *Random Forest*. Será apresentado os passos de desenvolvimento e fluxo de utilização do script desenvolvido.**

***Palavras-Chave*—visão computacional, skimage, sklearn, opencv, random forest, support vector machine, k-nearest neighbors**

***Abstract*—This document to present the conclusion work of the Computational Vision class of the specialization course in Applied Artificial Intelligence, from UFPR. The present study performs the classification of different levels of caries in teeth, based on the characteristics of shape and texture. The analysis was performed using Random Forest, Support Vector Machine and K-Nearest Neighbors on the base provided by UFPR. The Python language used as the development base with support OpenCV and SKImage libraries. The best prediction results were generated by the Random Forest model. The development steps and usage flow of the developed script will be presented.**

***Index Terms*—computational vision, skimage, sklearn, opencv, random forest, support vector machine, k-nearest neighbors**

I. DESENVOLVIMENTO

Em análise ao estudo proposto de desenvolvimento de um algoritmo, que realize a classificação das imagens apresentando métricas como acurácia, F1 scores, etc. Foi possível escolher os classificadores *Random Forest*, *Support Vector Machine* e *K-Nearest Neighbors* já trabalhados em matérias anteriores do curso.

A. Treinamento à escolha de melhor modelo

Avaliando a disposição e separação das imagens disponibilizadas, foi possível desenvolver o algoritmo para execução do script de visão computacional e treinamento para escolha de melhor modelo executado. Acompanhando o fluxo da Figura 1, é realizado o *discovery* das imagens, respeitando as suas separações por diretório de índice de classificação (c2, c3, c4, c5, c6 e hg), logo após, é realizado a extração de recursos das imagens (*shape* e *texture*) para separação entre treinamento e teste, em uma porcentagem de respectivamente 80% e 20% dentre utilização dos classificadores. Após finalização, é salvo o melhor modelo para testes futuros e apresentado os resultados.

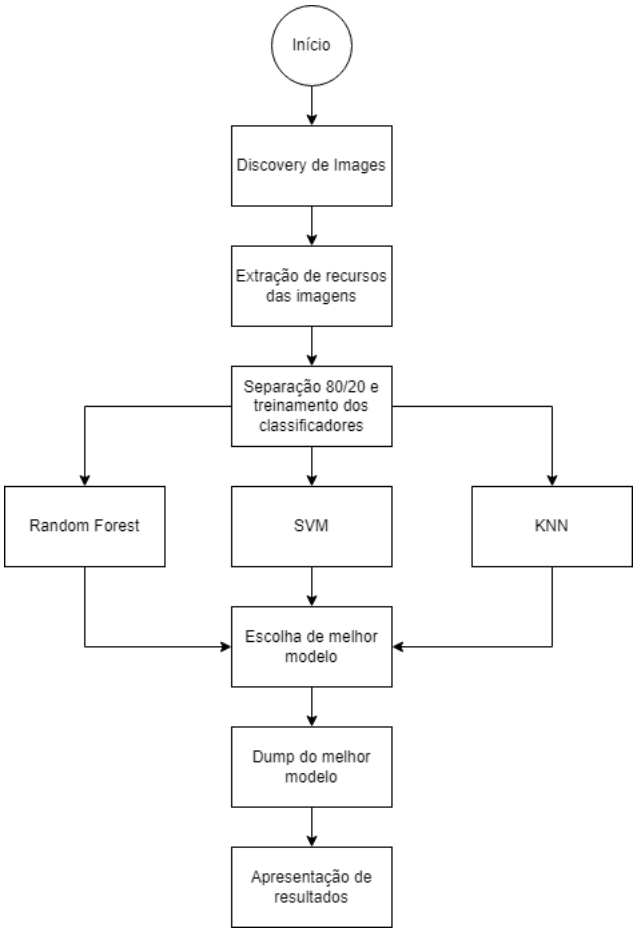

Figure 1. Fluxo para treinamento e dump de melhor modelo.

B. Construção de Testes

Após apresentação de resultados e identificação de melhor modelo, será necessário realizar testes com outras imagens para executar o script para testes de classificação conforme definição do índice de cárie. Definido o fluxo na Figura 2, conteplando a definição de um arquivo separado por virgula com o caminho da imagem e índice de cárie para execução do script com o melhor modelo e apresentar a classificação.

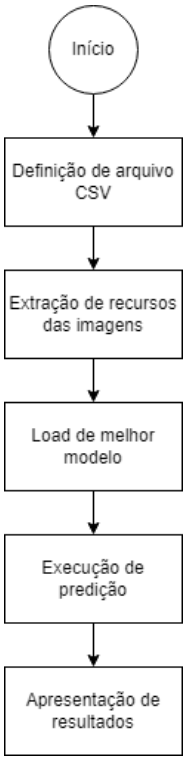

Figure 2. Fluxo para utilização do melhor modelo.

C. Script Python

Em análise em trabalhos e artigos semelhantes, para desenvolvimento do script, foi escolhido a linguagem Python versão 3.10.3, com auxílio das bibliotecas base OpenCV e SKImage para tratamento das imagens e SKLearn para utilização dos classificadores. O código completo para execução do treinamento, tanto para utilização do melhor modelo, encontram-se no repositório do Github no link abaixo.

https://github.com/viniciustrainotti/python-ai-visao-computacional/tree/master

Por padrão, o script esta de forma de uso para utilização de testes pelo melhor modelo, com inputs do arquivo "new_images.csv", seguindo o padrão de "path" e "class" nas suas colunas.

Para utilização do modo de treinamento e escolha de melhor modelo, será necessário comentar as linhas 95 à 100, e descomentar as linhas 79 à 93. Deste modo, o script já esta pronto para realização do processo de treinamento e escolha de melhor classificador.

Em relação ao desenvolvimento da função "extract_features_opencv" para extrair os recursos (*shape* e *texture*), foi convertido a imagem para escala de cinza, em seguida, aplicado o *threshold* para segmentar os dentes das cavidades. Em seguida, é extraído as características de *shape* e *texture* da imagem *thresholded* usando momento de Hu e matrizes de co-ocorrência. Por fim, é concatenado as características de *shape* e *texture* em um vetor único de características.

Para a função "train_and_evaluate", foi definido os clasificadores e por meio de um laço iterando-os, realizado a classificação dos dados e apresentação dos resultados por classificador.

II. RESULTADOS

Após a executação de treinamento e testes no script, a escolha de melhor modelo foi o *Random Forest*, conforme dados apresentados de saída na Tabela 1.

Table I
RESULTADO CLASSIFICADORES

Classificador	**Dados**			
	Accuracy	***Precision***	***Recall***	***F1 Score***
Random Forest	0.275	0.272	0.273	0.269
Support Vector Machine	0.191	0.183	0.203	0.145
K-Nearest Neighbors	0.200	0.202	0.196	0.196

Apesar do resultado não ter uma acurácia alta, em testes realizados com modelo com as informações do arquivo "new_images.csv" conforme Tabela 2, apresentaram resultados corretos na saída do script.

Table II
RESULTADO TESTES MELHOR MODELO

Dados		
path	***class***	***resultado***
banco_dentes_IAA/c2/0000000015_c2.jpg	c2	c2
banco_dentes_IAA/c2/0000000017_c2.jpg	c2	c2
banco_dentes_IAA/hg/0000000006_hg.jpg	hg	hg

REFERÊNCIAS

[1] A. Gopinathan, "Detecting tooth decay and cavities using Tensorflow Object Detection API" [Online]. Disponível em: https://github.com/atul-g/object_detection_on_cavities

[2] scikit-learn developers, "sklearn.ensemble.RandomForestClassifier" [Online]. Disponível em: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

[3] Grace F. Olseen, Susan S. Brillant, David Primeaux and Kayuan Najarian, "An image processing enabled dental caries detection system", IEEE International Conference on Multimedia and Expo, 2009.

[4] Tutorialspoint, "How to compute Hu-Moments of an image in OpenCV Python" [Online]. Disponível em: https://www.tutorialspoint.com/how-to-compute-hu-moments-of-an-image-in-opencv-python

[5] Notas de aula.