



Árvore de Decisão Projeto

Computação Inteligente

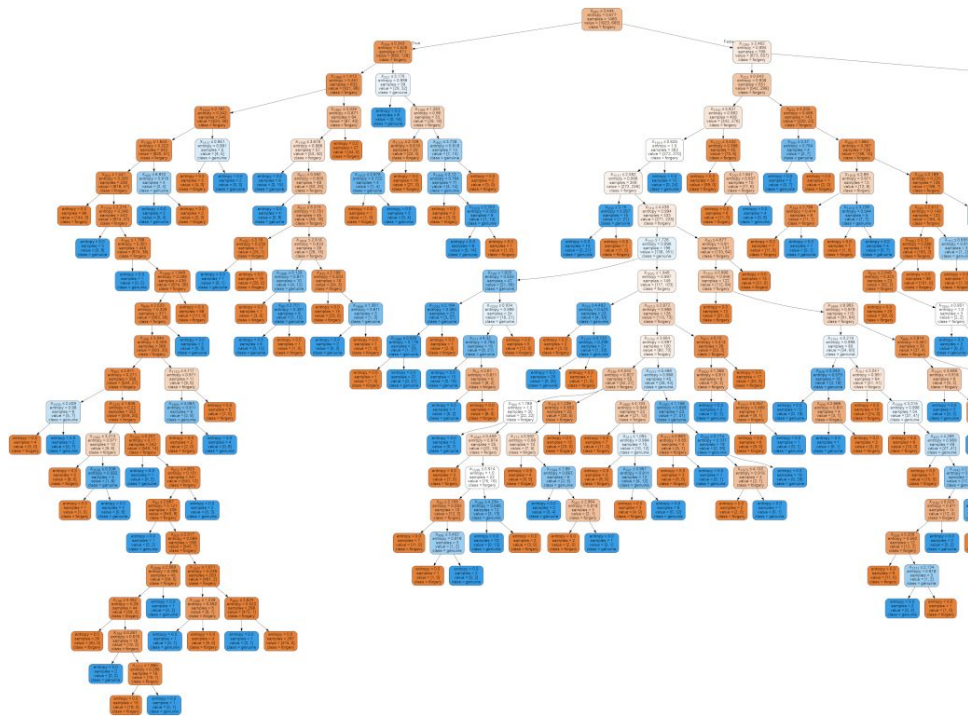
Arthur Flor
afsn@ecomp.poli.br

Conteúdo

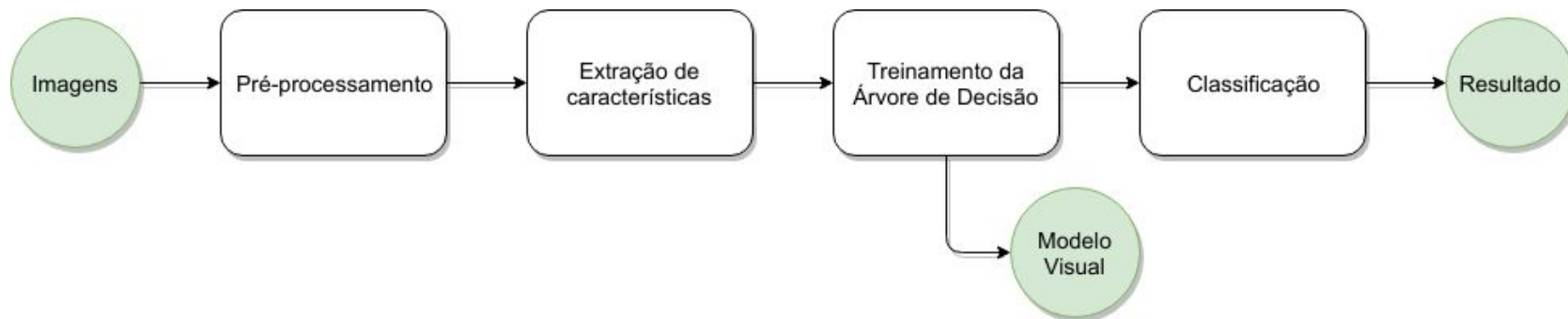
1. Introdução
2. Fluxograma do Projeto
3. Etapas do Projeto
4. Ferramentas
5. Experimento
6. Resultados
7. Conclusões
8. Referências

Introdução

No contexto de reconhecimento de escrita, o projeto utiliza a técnica Árvore de Decisão para classificar imagens de caracteres, buscando identificar qual caractere está associado à imagem.

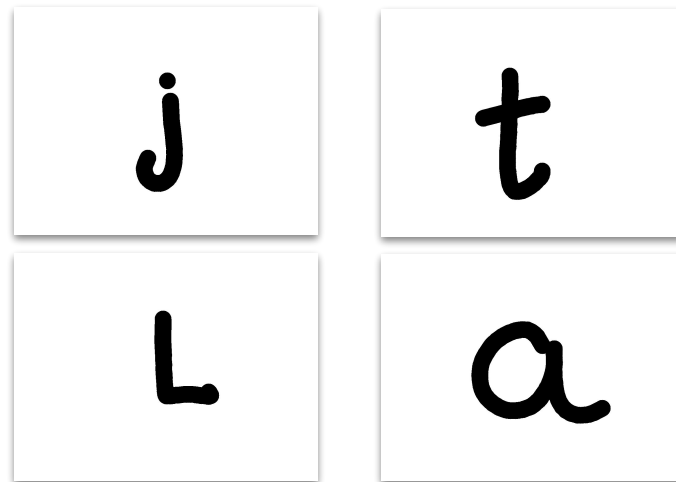


Fluxograma do Projeto



Dados Utilizados

- Chars74K image dataset - University of Surrey
- Última atualização em 2015
- 1980 imagens de caracteres (128x96 pixels)
- 36 classes (0-9, A-Z)



Exemplo das imagens do dataset

Pré-processamento

1. Binarização da imagem
2. Minimum Bounding Box no caractere
3. Redimensionamento da imagem para 512x512



Imagem 128x96



*Binarização e Minimum
Bounding Box*

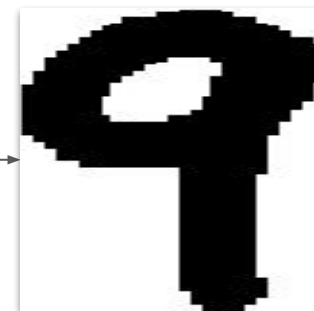
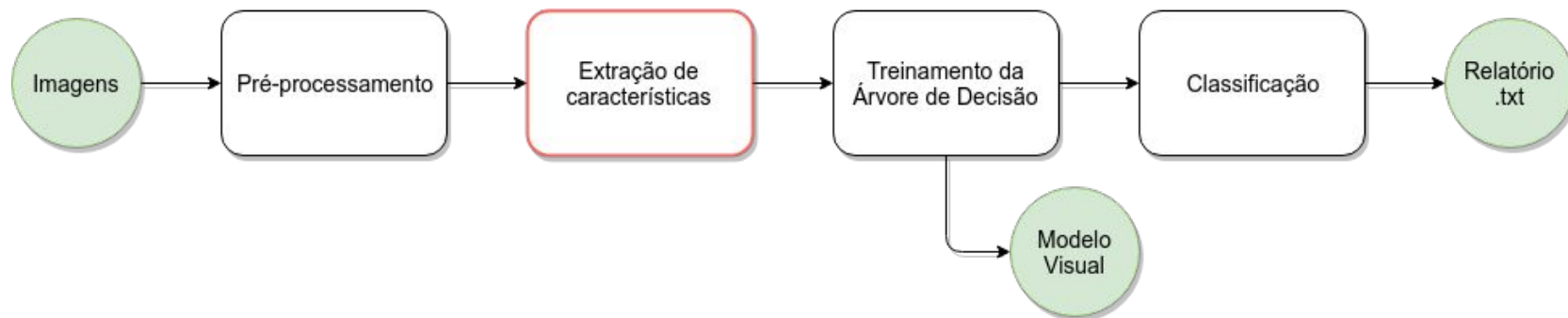


Imagem 512x512

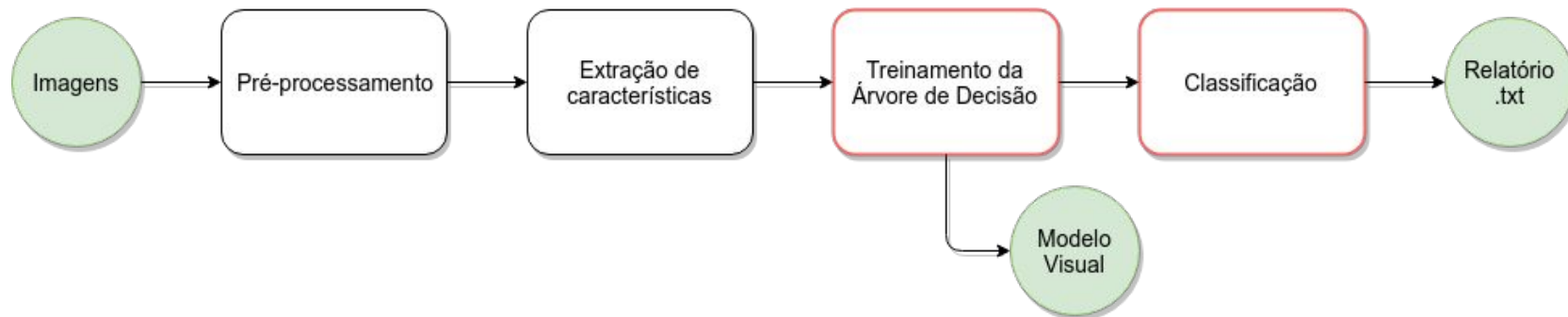
Extração de Características

- Momentos Invariantes de Hu (vetor com 7 características)
- Convolutional Neural Network - CNN (vetor com 2048 características)



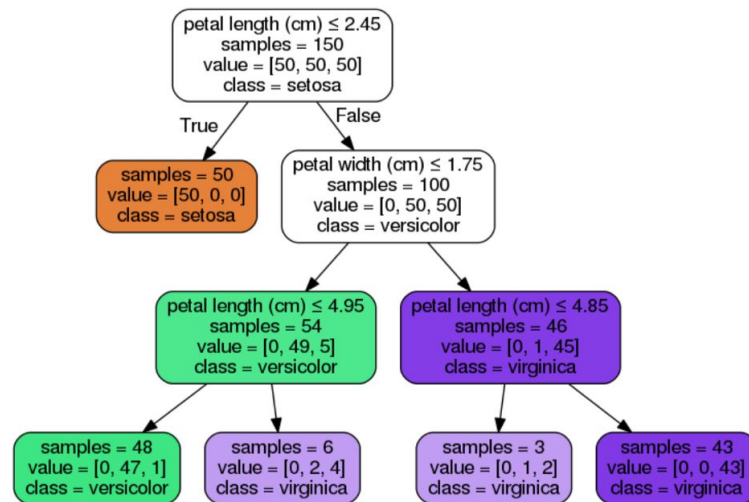
Algoritmos para Treinamento/Classificação

- Implementação manual: C4.5
- Implementação com sklearn: CART e Random Forest



Ferramentas

- Python
- Scikit-learn
- Theano/Lasagne
- Graphviz



Exemplo do modelo visual gerado pela biblioteca Graphviz

Experimento

- Executar todas as combinações das técnicas de extração de características e de árvore de decisão (6 combinações);
- Dividir as imagens, de modo aleatório, em 66,67% para treinamento e 33,33% para teste;
- Realizar 30 vezes cada combinação, armazenando o tempo de execução e a taxa de acerto.

Resultados (Precisão)

Treinamento: 1320 imagens

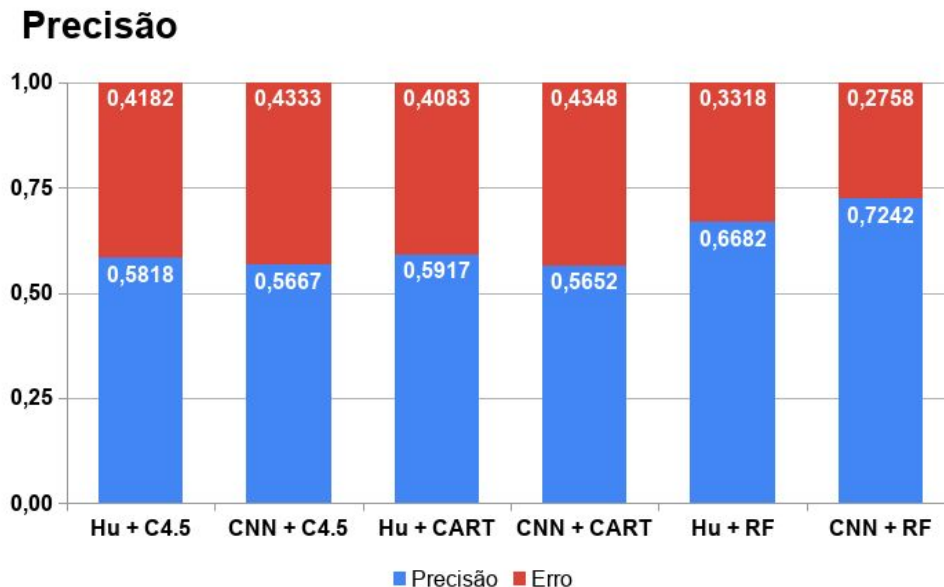
Teste: 660 imagens

Máx:

CNN + RF (72,42%): 478/660

Mín:

CNN + CART (56,52%): 373/660



Resultados (Tempo)

Treinamento: 1320 imagens

Teste: 660 imagens

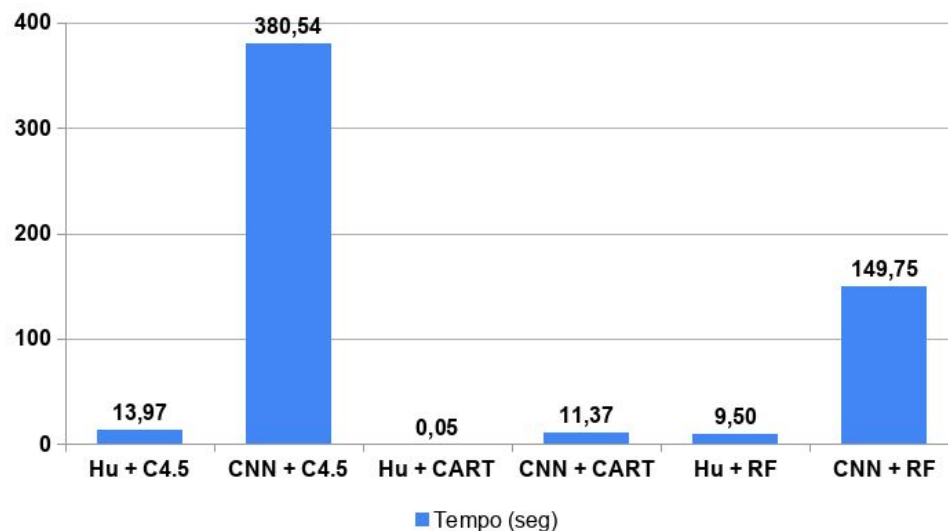
Máx:

CNN + C4.5: 380,54s

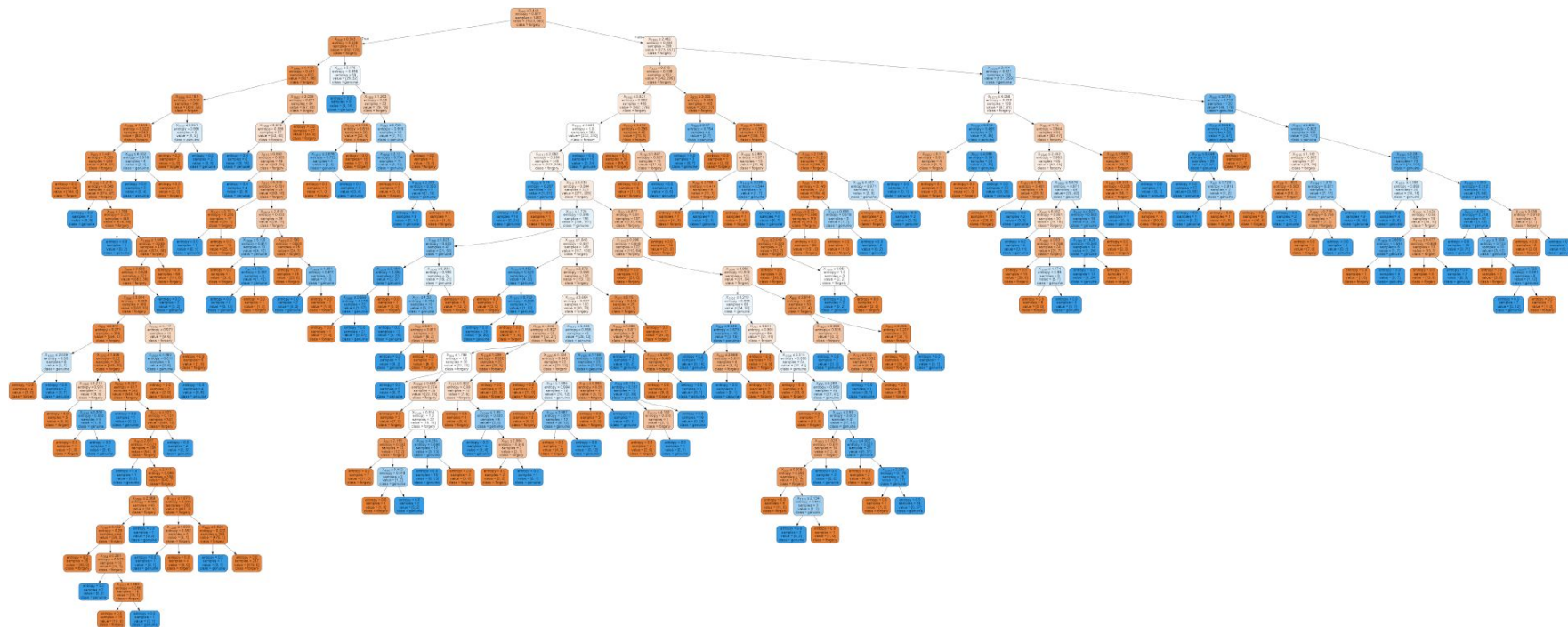
Mín:

Hu + CART: 0,05s

Tempo de execução



Visualização



Representação visual de uma árvore da Random Forest

Conclusões

- A taxa de acerto melhora conforme o pré-processamento;
- A CNN obteve um melhor resultado, comparado aos Momentos Invariantes de Hu, quando combinado a Random Forest;
- O código implementado manualmente (C4.5) não está otimizado. Isso foi observado no tempo de execução quando combinado a CNN (input de 2048 características).

Referências

- <http://www.ee.surrey.ac.uk/CVSSP/demos/chars74k/>
- <https://machinelearningmastery.com/implement-decision-tree-algorithm-scratch-python/>
- <http://scikit-learn.org/stable/modules/tree.html>
- <https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html>
- <https://medium.com/@williamkoehrsen/random-forest-simple-explanation-377895a60d2d>
- <https://towardsdatascience.com/random-forest-in-python-24d0893d51c0>