

Método de reconhecimento e padronização de captação para imagens de chave em protocolos de manutenção de Estação Radio Base

Method for recognition and capture standardization for imagens of keys in maintenance protocols of Radio Base Stations

Bárbara Carvalho ¹  orcid.org/0000-0002-0137-9804

Bruno Fernandes ^{1,2}  orcid.org/0000-0002-6001-3925

¹ Escola Politécnica de Pernambuco, Universidade de Pernambuco, Recife, Brasil,

² Pós-graduação em Engenharia de Sistemas, Escola Politécnica de Pernambuco, Pernambuco, Brasil

E-mail do autor principal: Bárbara Carvalho bmbc@ecomp.poli.br

Resumo

O artigo apresenta um problema de verificação dinâmica de imagens de chave/não chave para um protocolo de manutenção de ERB e utiliza extratores de características e modelos de classificação de inteligência computacional a fim de desenvolver uma ferramenta para otimizar o trabalho humano envolvido. São testados extratores de momentos da imagem e uma arquitetura de rede profunda como extratores de características e posteriormente uma Random Forest Classifier e uma Support Vector Machine para determinar a classe das imagens analisadas. Os resultados obtidos são apresentados ao final performando no melhor caso descrito 97% de acurácia no problema.

Palavras-Chave: Imagem; Classificação; Extração;

Abstract

The paper presents a dynamic key/non-key image verification problem for a Radio Base Station maintenance protocol and utilizes feature extractors and computational intelligence classification models to develop a tool to optimize the human work involved. Image moment extractors and deep network architecture are experienced as feature extractors and then a Random Forest Classifier and Support Vector Machine to determine the class of the analyzed images. The results are presented at the end, performing in the best case described 97% of accuracy in the problem.

Key-words: Image; Classification; Extraction.

1 INTRODUÇÃO

É cada vez mais comum no dia-a-dia dos mais diversos trabalhos a utilização da tecnologia como uma forma de auxílio em diferentes processos. Muitas vezes o aparato tecnológico mais comum pode ser utilizado como aliado em verificações para trabalhos de campo que utilizam protocolos de segurança operacionais mais complexos - nesses casos, quanto mais acessível a ferramenta de verificação, melhor para o desempenho e padronização de tais protocolos.

Em muitos casos, verificação através de imagens são preferíveis dado que analisar imagens é um processo muito simples e intuitivo para o cérebro humano. No contexto de manutenção de ERB's (Estação Rádio Base) é comum a utilização e análise de imagens para garantir que os protocolos estão sendo seguidos; de modo que o agente em campo deve fotografar momentos específicos do processo, e tais imagens serão submetidas à avaliação de um terceiro para confirmar que todos os passos foram executados adequadamente. Nesses casos as imagens são capturadas através dos telefones celulares dos próprios agentes de campo, o que é positivo no quesito de acessibilidade, porém gera uma série de inconsistência e falta de padronização nos dados adquiridos.

Dado que o estágio de verificação das imagens é realizado por humanos é interessante que verificações mais triviais possam ser automatizadas para que o recurso humano seja usado em avaliações mais críticas ou mais complicadas, otimizando também o tempo do indivíduo que realiza tal tarefa.

Este trabalho foca na padronização e verificação das imagens de chaves capturadas nos processos de manutenção já referidos.

2 MODELO PROPOSTO

O problema trata-se de executar uma identificação e verificação de imagens nas quais se analisa a presença de um determinado objeto. Extratores de características foram utilizados visando extrair informações pertinentes a este contexto como, por exemplo, quais formas estão presentes na figura. Para este tipo de identificação, técnicas de extração de momentos da imagem mostram-se

eficazes, quando alinhados com um pré processamento de identificação de contorno [1].

Ainda no que se refere aos extratores de características, três técnicas foram utilizadas a fim de comparar os resultados; *Hu Moments* e *Zernike Moments*, os quais contém informações invariantes à rotação, escala e translação. O terceiro extrator de características utilizado foi uma modelagem de *Deep Learning*, mais especificamente a arquitetura de VGG (*Visual Geometry Group*) [2], muito utilizada para classificação de objetos dado que tem excelente performance na base de dados *ImageNet* [3]. As características retornadas pelos extratores citados foram passadas como *input* para dois diferentes modelos de classificação: *Support Vector Machine* e *Random Forrest*.

A escolha dos modelos de classificação se baseou principalmente na velocidade do processamento, para que o modelo treinado pudesse ser embarcado em aparelhos mobile. O modelo deve ser portátil, leve e de resposta rápida. Além disso, algumas bibliotecas para *Python* já disponibilizam os modelos SMV ou Randon Forrest pré prontos tem portabilidade para mobile, além de extratores de características [4][5].

2.1 SVM

Support Vector Machine, proposta em 1995 por Vapnick [6] é um algoritmo de aprendizado de máquina supervisionado baseada fortemente em um modelo matemático que calcula um hiperplano ótimo em que as classes do problema sejam linearmente separáveis por ele.

É determinístico e uma vez encontrada a fronteira para classificação seu processamento é leve para o dispositivo no qual está sendo executado; porém sua complexidade computacional pode ser sensível à quantidade de dimensões que o problema possui.

2.2 Random Forest

Por se tratar de um agrupamento de várias árvores de decisão, o *Random Forest* se torna um classificador robusto e resiliente à ruído da informação. O classificador gera uma floresta de árvores de decisão e para definir o resultado de uma classificação é feita uma votação entre os resultados de cada árvore gerada. Outro ponto positivo é que o processo de criação de árvores presente no *Random*

Forest dificulta o surgimento de comportamento de *overfit* [7].

2.3 Momentos de uma imagem

Momentos são valores escalares usado para caracterizar uma função e capturar suas características mais significantes. Do ponto de vista matemático, momentos são projeções de uma função em uma base polinomial.

Momentos de uma imagem podem ser descritos como um grupo de características invariantes extraídas através de uma função matemática que é executada sobre valores dos *pixels*. Como toda invariante, tal técnica sofre pouca ou nenhuma influência de fatores como ruído, qualidade da imagem, rotação, translação ou espelhamento; sendo assim muito utilizada para identificar grupos de objetos semelhantes [8].

A maior parte dos extratores de momentos da imagem retornam um vetor com informações referentes à área da imagem, centroide e rotação; tal vetor tende a ter um tamanho fixo; facilitando também a modelagem da entrada do modelo que deverá avaliar as informações extraídas da imagem.

2.3.1 Hu Moments

Hu Moments, apresentados pelo pesquisador homônimo em 1962 [9] são um grupo de momentos invariantes calculados sob os valores de *pixel* de uma imagem que descrevem os formatos presentes na mesma de forma única e independente da rotação, translação e escala das formas que desejam ser identificadas.

2.3.2 Zernike Moments

Zernike Moments [10] apresenta uma abordagem mais robusta de que *Hu Moments* por não possuir informação redundante em seus valores. São cálculos da magnitude de momentos complexos ortogonais do centroide de uma imagem, permitindo variar o raio utilizado para o cálculo. Para que a extração seja mais eficiente, é preciso que a imagem seja pré processada a fim de destacar as bordas extraíndo os contornos e remover o fundo, para que o objeto que se quer identificar fique bem destacado.

2.4 VGG

Visual Geometry Group é uma arquitetura de rede profunda convolucional, proposta pelo grupo de

mesmo nome da universidade de Oxford. O modelo é muito utilizado para identificação de objetos tendo sido desenvolvida inicialmente para o desafio da base de dados *ImageNet*, onde conseguiu em 2014 o primeiro e segundo lugar nos desafios de localização e classificação respectivamente. [2]

O modelo da VGG escolhida para este trabalho é conhecida como VGG16 por possuir 16 camadas; uma visualização de sua arquitetura pode ser observada na **Figura 1** [11], sendo as 4 ultimas camadas responsáveis pela classificação dentro das classes do *ImageNet*. Para este trabalho, estas 4 ultimas camadas foram dispensadas.

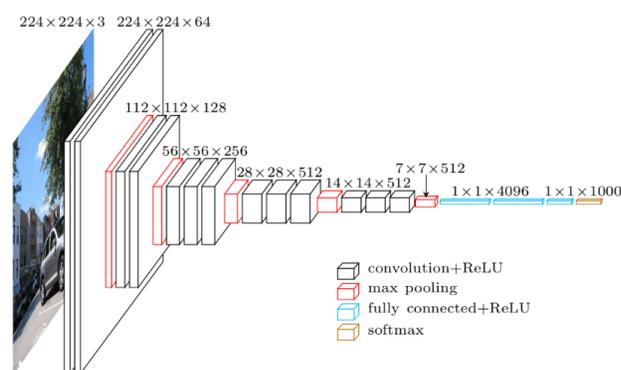


Figura 1: Visualização espacial da arquitetura VGG16.

3 ARRANJO EXPERIMENTAL

A base utilizada para o trabalho foi construída especificamente para este. Uma versão inicial possuía 100 imagens de chaves e 100 imagens de outros objetos distintos – porém confundíveis com chaves, como chaveiros, etiquetas e objetos com superfície/reflexo metálico. Nos exemplos negativos também constavam chaves de carro e chaves de estilo colonial que não devem ser aceitas no relatório. Após testes iniciais com a base de 200 amostras, foram coletadas mais amostras e o número final chegou a 340 imagens, sendo 170 de chaves aceitáveis e 170 outros objetos ou chaves não aceitadas, ou ainda imagens parciais de chaves.

Alguns exemplos cedidos pelo cliente, pertencentes à base original foram agregados à base como exemplos de chaves, porém apenas exemplos que seguem um padrão semelhante às demais imagens coletadas para base.

A Figura 2 é uma amostra de imagem aceita pelo modelo, e a Figura 3 um exemplo de imagem usada como exemplo negativo.



Figura 2: Exemplo de imagem utilizada como aceitável.



Figura 3: Exemplo negativo de treinamento da base.

É importante observar que a base utilizada trabalha encima de uma padronização do tipo de imagem que deve ser processada pelo modelo proposto, já que ao analisar exemplos cedidos pelo cliente, existiam imagens com mais de um objeto em destaque, ou ainda que a chave estava parcialmente coberta, ou até imagens com mais de uma chave. Como o objetivo do modelo é verificar se a chave se enquadra no padrão requerido para chaves de ERB's

é recomendado que as imagens sejam coletadas destacando o objeto a ser analisado.

4 ARQUITETURA DO SISTEMA PROPOSTO

O sistema tem uma arquitetura visando sua possível utilização em campo, o que implica em uma fácil portabilidade para aparelhos *mobile*. Além disso, neste contexto é esperado que o sistema entregue uma resposta em tempo real para o técnico responsável pela vistoria na ERB.

Dado que cada técnico coletará as amostras de imagem de seu próprio *smartphone*, não pode haver uma padronização em qualidade de imagem e tipo de câmera. O sistema deve ser resiliente a essa limitação adotando abordagens que independam dessas limitações. Outra saída é padronizar um pré-processamento igual para toda amostra de imagem recebida correndo o risco de perder alguma informação relevante.

As abordagens aqui testadas se dividem em dois modelos de classificadores que testam os mesmos 3 extratores de características como entrada, de modo que as arquiteturas podem diferir em algumas especificações que atendem as necessidades de cada tipo extrator.

4.1 Arquitetura e pré-processamento para os extratores de momentos

Após ser captada a imagem deverá ser convertida para tons de cinza, já que extratores de momento não utilizam informações sobre as cores presentes na imagem. Após a conversão para escalas de cinza, é utilizada uma função de *threshold* para discriminar apenas valores de pixels mais claros e mais escuros, e invertendo esta imagem, para que o objeto que se deseja identificar fique destacado na imagem.

Após isso, a imagem destacada é submetida a uma função de cálculo de contorno e este contorno então é desenhado separadamente em uma nova cópia denominada *outline*. Uma visualização mais intuitiva das transformações pode ser acompanhada na Figura 4. Todas as transformações descritas são feitas com funções disponíveis na biblioteca de OpenCV [12].

Esta ultima versão da imagem é submetida aos extratores de momento o que permite uma melhor detecção e calculo dos momentos da imagem.

A função de *Hu Moments* retorna 7 valores calculados através das equações de Hu que são transformados em um só vetor de tamanho 7 para cada imagem. As informações são então normalizadas em relação à sua magnitude através de uma transformação logarítmica.

Uma estratégia semelhante é feita com os momentos extraídos utilizando a metodologia de *Zernike Moments*. Porém o tamanho do vetor retornado por essa metodologia é um total de 25 features para cada imagem.

Em ambos os casos separadamente o sistema então utiliza uma estrutura de *dataframe* reunindo todos os momentos extraídos de cada imagem para treinar cada um dos classificadores. Tal data frame possui, além dos momentos, os *labels* referentes às imagens de onde as informações foram extraídas; utiliza-se "1" para o *label* "key" e "0" para o *label* "nokey" -imagens de "chave" e "não chave", respectivamente.

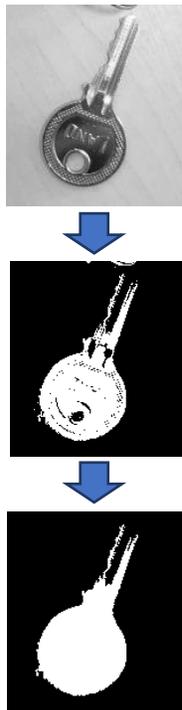


Figura 4: Exemplo das transformações das imagens.

4.2 Arquitetura utilizando VGG

No caso da VGG a imagem é passada diretamente para a rede inclusive com os canais RGB. Porém há um redimensionamento da imagem para uma escala

de 224 por 224 pixels combinado com uma transformação da imagem em vetor numérico. A arquitetura da rede em si segue o padrão disponível na biblioteca Keras[13] da VGG16, porém dispensando as 4 ultimas camadas como já explicado.

5 RESULTADOS

Os resultados aqui descritos serão separados por sua metodologia de classificador utilizado e dentro deste tópico separados por tipo de extratores.

Para ambos os casos os *dataframes* gerados a partir dos extratores de momentos foram passados como entrada do classificador SVM. Para o caso do extrator de Hu possuía 8 colunas que correspondem aos valores dos momentos calculados com a adição de uma coluna para a classe da amostra. No caso dos momentos de Zernike o extrator retorna 25 valores de momento por amostra, de modo que o *dataframe* utilizado como entrada na SVM possuía 26 colunas sendo a ultima responsável pelo identificador da classe. Os resultados para a SVM performaram uma acurácia de 54% para o kernel linear para o Hu Moments e um desempenho de 61% para os extratores de momentos de Zernike.

A extração de características da VGG foi utilizada como entrada na SVM no formato de um *dataframe* 25089 características como resultado do processo de achatamento da extração de características retornada pela rede *deep* que possuía dimensões (1x7x7x512) com a adição de uma coluna relativa ao *label*. A SVM foi utilizada com a configuração de kernel linear e apesar da grande quantidade de features apresentou um resultado de 97% de acurácia, desempenhando também 97% de taxa de acerto para os casos de positivo verdadeiro em ambas as classes.

Assim como no caso da SVM a modelagem utilizada para a entrada da Random Forest. Para o caso do extrator de Hu possuía 8 colunas que correspondem aos valores dos momentos calculados com a adição de uma coluna para a classe da amostra. No caso dos momentos de Zernike o extrator retorna 25 valores de momento por amostra, de modo que o *dataframe* utilizado como entrada na Random Forest possuía 26 colunas sendo a ultima responsável pelo identificador da classe. Os resultados para a Random Forest performaram uma

acurácia de 61% para o Hu Moments e 64% para Zernike.

A extração de características da VGG foi utilizada como entrada na Random Forest no formato de um *dataframe* 25089 características como resultado do processo de achatamento da extração de características retornada pela rede *deep* que possuía dimensões (1x7x7x512) com a adição de uma coluna relativa ao *label*. A Random Forest apresentou um resultado de 95% de acurácia total tendo 100% de verdadeiro positivo para a classe "no key" e 92% para "key".

Uma tabela com o resumo dos resultados obtidos pode ser visualizada na Tabela 1.

Tabela 1: Resumo dos resultados obtidos.

Arranjo de extratores e classificadores	Resultados (taxa de acurácia geral)
Hu Moments + SVM	54%
Zernike Moments + SVM	61%
VGG + SVM	97%
Hu Moments + Random Forest	61%
Zernike Moments + Random Forest	64%
VGG+ Random Forest	95%

6 CONCLUSÃO

O trabalho demonstra que metodologia para reconhecimento de imagens e extrator de características pode ajudar na identificação otimizada de imagens que seguem ou não determinados padrões requeridos. Atingindo resultados de 95% e 97% de acurácia com um extrator de características VGG em conjunto com classificadores Random Forest e SVM respectivamente. A aplicação aqui demonstrada utiliza modelos de chave, porém a mesma metodologia pode ser reproduzida para os mais diversos objetos (cadeados, fios, antenas) que estejam presentes no relatório.

7 TRABALHOS FUTUROS

Seria interessante uma otimização no tempo de consulta das imagens para que fosse possível gerar uma verificação portátil e em tempo real. Com um modelo leve e de verificação rápida, é possível e incentivado que seja criada uma API para validação das imagens como sendo aptas para compor o

relatório, ainda em campo, no momento da aquisição da imagem.

8 AGRADECIMENTO

Os autores gostariam de prestar seus agradecimentos à FITec/SECTI/CMA-Parqtel/UPE/FACEPE pelo conhecimento adquirido e pelas oportunidades oferecidas no decorrer do processo. Gostaria de agradecer também ao meu orientador pelo suporte nessa importante etapa final e aos meus amigos de turma pelo apoio durante todo o processo.

REFERÊNCIAS

- [1] ZHANG, L. et al.; *Application of improved HU moments in object recognition*, 2012 IEEE International Conference on Automation and Logistics, Zhengzhou, 2012, pp. 554-558.
- [2] SIMONYAN, K.; ZISSERMAN, A.; *Very Deep Convolutional Networks for Large-Scale Image Recognition*, Conference Paper at ICLR 2015.
- [3] Russakovsky, O. et al.; *ImageNet Large Scale Visual Recognition Challenge*, Int J Comput Vis, v. 115, pp. 211-252 (2015).
- [4] *Tensorflow Lite for Mobile & IoT*, 2019. Disponível em: <https://www.tensorflow.org/lite/?hl=pt_br>. Acesso em: 24 outubro 2019.
- [5] *OpenCV Platforms*, 2019. Disponível em: <<https://opencv.org/platforms/>>. Acesso em: 25 outubro 2019.
- [6] CORTES, C.; VAPNIK, V.; *Support-Vector Networks*. Kluwer Academic Publishers, Boston, 1995. Disponível em: <<https://link.springer.com/content/pdf/10.1007/BF00994018.pdf>>. Acesso em: 20 de outubro de 2019.
- [7] BREIMAN, L.; *Random Forests*, Statistics Department, University of California, 2001, Machine Learning. Disponível em: <<https://www.stat.berkeley.edu/~breiman/randomforest2001.pdf>>. Acesso em: 15 Outubro de 2019.
- [8] FLUSSER, J.; SUK, T.; ZITOVÁ, B.; *Chapter 1 Introduction to Moments*, 26 October 2009. Disponível em: <<https://pdfs.semanticscholar.org/314d/2579ec7>

[3c89e9c3aadf06f5ade899cf8f4b0.pdf](#)>, acesso em: 15 Outubro de 2019.

[9] HU, M.K.; **Visual Pattern Recognition by Moment Invariants**, IRE transactions on information theory, The University of Utah, 1962. Disponível em: <<http://www.sci.utah.edu/~gerig/CS7960-S2010/handouts/Hu.pdf>>, acesso em: 25 outubro de 2019.

[10] A. KHOTANZAD; Y. H. HONG; **Invariant image recognition by Zernike moments**, in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 5, pp. 489-497, May 1990.

[11] **VGG in TensorFlow**, 16 Jun 2016. Disponível em: <<http://www.cs.toronto.edu/~frossard/post/vgg16/>>. Acesso em: 25 outubro 2019.

[12] **OpenCV 2.4 13.7 Documentation**, Última atualização 18 Novembro 2019, disponível em: <<https://docs.opencv.org/2.4/index.html>>. Acesso em: 25 outubro 2019.

[13] **Keras Applications**. Disponível em: <<https://keras.io/applications/>>. Acesso em: 25 outubro 2019.