Mimicry: Um crawler de classificação automática de suspeitas de phishing utilizando análise de imagens

Gustavo Henrique Lopes de Souza
Centro de Informática - UFPE
ghls@cin.ufpe.br

Abstract

At most information security companies, the phishing detection is performed by a human who analyzes the visual identity of some artefact and check if it's trying to imitate a known brand and whether it's authentic or not. This process is slow and doesn't scale well. To deal with this situation we're proposing a tool which performs brand classification over visual identity of an artefact using instance based learning.

1. Introdução

O processo de classificação de phishing utilizado hoje na maior parte das empresas especializadas em segurança da informação é bastante lento e heurístico. No geral um ser humano analisa um artefato suspeito, verifica sua identidade visual afim de saber se o artefato está tentando imitar alguma marca conhecida ou não e, por fim, valida se o artefato é ou não legítimo.

Afim de semi-automatizar este procedimento, propomos uma ferramenta que já classifica a identidade visual do artefato, indicando se esta pertence ou não a alguma marca previamente conhecida.

Para isto montamos uma base que guarda imagens correspondentes às identidades visuais de algumas marcas, extraímos as features destas imagens através de um descritor HoG e, para cada artefato suspeito apresentado, comparamos suas features com as features da base através de técnicas de aprendizado baseado em instâncias, com o objetivo de classificá-lo.

Portanto, o método desenvolvido é composto pelas etapas de montagem da base de imagens, extração de features e descritores das imagens da base, extração de features e descritores das imagens suspeitas e classificação das imagens suspeitas.

2. Montagem da base

A montagem da base de dados pode ser feita de forma automática ou manual. A principal preocupação desta etapa deve ser capturar elementos que descrevam bem a identidade visual de uma dada marca, como por exemplo logotipos, esquemas de cores, produtos e serviços oferecidos, etc.

Para a montagem automatizada, desenvolvemos um procedimento que visa extrair os componentes visuais mais comuns em websites legítimos de uma dada marca. Para tal, pedimos como entrada uma lista de URLs legítimas de uma determinada marca, como por exemplo site oficial, página do Facebook e página do Twitter. Para cada URL informada, um browser é inciado para abrir a página, executar um código em JavaScript que remove textos e links do site, afim de deixá-lo mais limpo para a captura de tela que é feita logo em seguida. Na imagem da captura, são extraídos os contornos e a hierarquia dos contornos usando a técnica apresentada em [1]. A hierarquia indica se um contorno está contido ou não em outro. Dos contornos extraídos, mantemos apenas aqueles que não possuem nenhum outro contorno dentro de sua área e aqueles que mantém que são pais dos que não têm filhos.

Fig 1: Extração de contornos do isntagram do Bradesco

Como resultado dos procedimentos aplicados, teremos regiões simples como retângulos, círculos e quadrados, que na maioria das vezes detém o logotipo da marca em questão e outras regiões não tão úteis como caracteres. Para eliminar essas regiões inúteis, aplicamos uma máscara de erosão na região binarizada, utilizando um kernel 5x5. Se após a erosão, mais de 90% dos pixels da imagem resultante tiverem valor 0, a região é descartada.

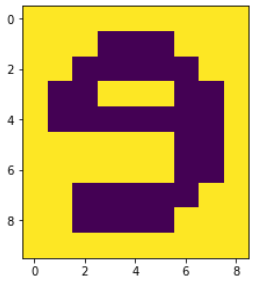

Fig 2: Caractere 9 extraído do instagram do bradesco.

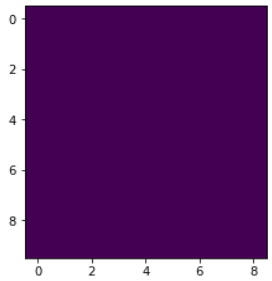

Fig 3: Caractere 9 após a erosão.

Após fazer esse processo para todos as URLs fornecidas na entrada, extraímos os vetores de features de cada uma das regiões obtidas SIFT [2], em seguida cada vetor é comparado com os demais através das distâncias entre eles, de modo que regiões que possuem features semelhantes são mantidas, enquanto regiões em que suas features não se assemelham com as de outra região são descartadas. Isso é feito para que elementos visuais destoantes do padrão sejam eliminados.

Por fim, as regiões remanescentes são adicionadas na base, todas rotuladas com o nome da marca em questão.

3. Extração de features

A extração de features é feita usando SIFT [2], com o foi mencionado na seção anterior. Nesta seção será detalhado como este processo é feito.

Primeiro, o método se preocupa em extrair os chamados keypoints da imagem, que são pontos onde características importantes da imagem podem ser encontradas, geralmente são bordas ou quinas. A região do keypoint então é extraída e redimensionada para uma janela 16x16.

A janela extraída é então dividida em 16 quadrados de 4x4 pixels de dimensão. Para cada quadrado, é calculado o gradiente de cada um dos seus pixels e um histograma é montado indicando a orientação dos pixels. Oito direções são consideradas: norte, sul, leste, oeste, nordeste, noroeste, sudeste e sudoeste.

Em seguida os 16 histogramas são concatenados, formando um vetor de features com 128 (16*8) dimensões. Esse vetor representa o keypoint em questão. Todos os vetores de todos os keypoints representam a imagem.

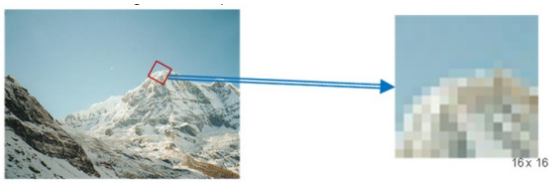

Fig 4: Extração de keypoint da imagem.

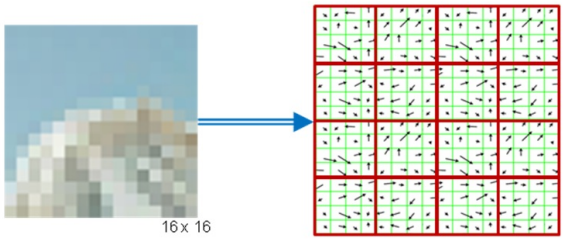

Fig 5: Divisão da região em 16 quadrados 4x4.

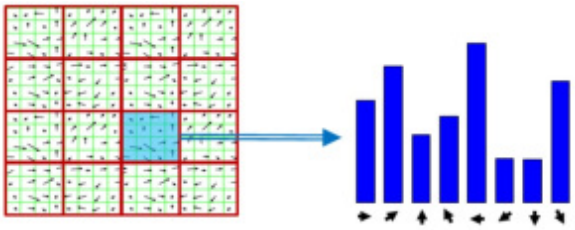

Fig 6: Montagem do histograma de direções de um dos quadrados

Fig 7: Concatenação dos 16 histogramas gerados resulta no vetor de features.

4. Classificação

O processo de classificação é feito usando SIFT e o método dos K-vizinhos mais próximos.

Dada uma captura de tela de uma suspeita de phishing, são extraídos os vetores de features dos keypoints desta imagem, conforme descrito na seção anterior. Esse vetor é então comparado por meio do kNN com todos os vetores de todas as imagens presentes no banco, a captura então recebe como rótulo a marca que possui os vetores de features que mais se assemelham com o vetor de feature atual.

Em cada imagem da base, os vetores de features mais próximos (semelhantes) dos vetores de features da captura que está sendo analizada recebem a

classificação de “goodMatch” e são contabilizados de forma que é possível saber ao fim do processo quantos goodMatches uma imagem do banco obteve quando comparada com a captura de tela em questão. O limiar de distância utilizado para determinar essa proximidade é de 70% da distância do vetor de feature da captura que está sendo analisado para a origem. Se o vetor de feature da base extrapola esse círculo, ele não é considerado uma “goodMatch”.

Por fim, cada marca recebe um score, que é a soma total dos goodMatches de todas suas imagens no banco, dividido pelo número de imagens do banco com rótulo pertencente a marca.

$$score_i = \frac{\sum_{j \in i} goodMatch_j}{images_i}$$

Eq 1: Cálculo do score da i-ésima marca da base de dados

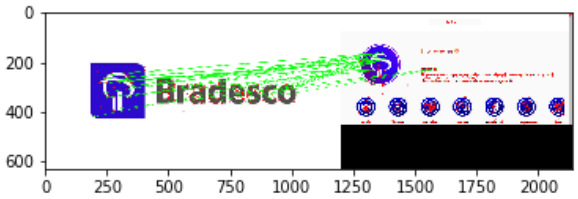

Fig 8: Processo de comparação de features feito entre o logotipo do Bradesco e o instagram do Bradesco. Os vetores de features mais próximos (isto é, semelhantes), foram demarcados com linhas verdes os interligando. Cada linha verde indica um goodMatch.

5. Validação dos resultados

Para validar os resultados, foi montada uma base de testes com capturas de tela de diversos phishings, totalizando um total de aproximadamente 30 imagens, pertencentes a 4 marcas distintas.

Foram utilizadas duas bases de identidades visuais: Uma contendo apenas imagens extraídas manualmente, de 5 marcas diferentes, com uma média de 4 imagens por marca e outra base contendo apenas imagens extraídas automaticamente a partir do processo descrito na seção 2, de 5 marcas diferentes com uma média de 6 imagens por marca.

Para cada imagem da base de phishing, foi testado se ela foi ou não classificada corretamente. Em caso negativo, foram exibidos os scores gerados.

Foram feitos dois testes, um utilizado a base de dados gerada manualmente e outro utilizando a base gerada automaticamente. Ambos obtiveram uma acurácia de acerto de 75%. Entretanto, em todos os casos de erro, a classificação correta estava com o segundo maior score. Portanto podemos dizer que o método aponta com 100% de acurácia as duas marcas mais prováveis para a captura de tela em questão.

7. Implementação

A implementação foi feita com auxílio das bibliotecas OpenCV, selenium, matplotlib e numpy da linguagem Python. Foi utilizado o descritor SIFT da biblioteca OpenCV, com os devidos parâmetros para fazer a classificação via kNN conforme foi descrito aqui. Um Jupyter notebook contendo o código está disponível no github.

8. Conclusão

Com este projeto foi possível verificar na prática a eficiência de métodos de aprendizagem baseado em instâncias quando as bases de treino são pequenas. Uma acurácia razoável foi obtida em troca de um baixo custo computacional.

Este método pode ser facilmente adicionado nos processos de cadastros de clientes novos em empresas de segurança de informação: Basta oferecer 3 ou 4 URLs de páginas pertencentes ao novo cliente, que uma base de dados de sua identidade visual é montada. A partir daí, toda imagem pode ser analisada automaticamente afim de verificar se está tentando ou não imitar algum dos clientes cadastrados.

9. Referências

[1] S. Suzuki et al, “Topological structural analysis of digitized binary images by border following”, *Computer Vision, Graphics and Image Processing*, 1985, pp. 32-46.

[2] Lowe, D. G., *“Distinctive Image Features from Scale-Invariant Keypoints”, International Journal of Computer Vision*, 2004, pp. 91-110.