Proposta de um sistema de autenticação por reconhecimento facial em linguagem de programação Python e banco de dados MySQL

Guilherme de Freitas Anacleto & Eduardo Alves Moraes

Faculdade de Tecnologia de Ourinhos - FATECOU - Av. Vitalina Marcurso, 1400, Campus Universitário

guilherme.anacleto@fatec.sp.gov.br & eduardo.moraes@fatec.sp.gov.br

Resumo

O reconhecimento facial está em evolução a cada dia. Reconhecer um rosto ou identificar quem seja apenas pela face, é uma tarefa simples para qualquer individuo, que se adaptou ao longo de milhões de anos de existência. A visão é uma das características fundamentais para a existência da espécie humana. Com todos os avanços tecnológicos, conseguiu-se programar uma visão computacional que simula a visão humana. No entanto reproduzir tal característica tão complexa em um sistema computacional, não seria tão simples sem o avanço continuo e rápido da tecnologia. Por meio de um sistema criado a partir de uma linguagem de programação de alto nível chamada Python, as complexidades encontradas no desenvolvimento deste tipo de funcionalidade foram reduzidas em apenas algumas linhas de programação, Sendo possível identificar um individuo por meio de sua face utilizando uma simples *webcam*.

1. Introducão

O campo da biometria ganhou a maior atenção e fez seu lugar como a opção mais confiável de reconhecimento durante o passado, devido à disponibilidade de tecnologia viável após extensa pesquisa neste campo e lacunas em outros sistemas de identificação [1].

Esse software desenvolvido em Python, tem como finalidade, analisar o rosto de um ser humano, sendo capaz de determinar, se esse usuário já esta na base de dados cadastrado ou não, ao identificar o usuário, o nome do individuo será apresentado logo abaixo de seu rosto, caso não seja possível identifica-lo, o sistema exibirá que a entidade é desconhecida. Utilizando a linguagem Python, se abriu um leque de possibilidades, com uma facilidade sem igual, com suas bibliotecas que trabalham diretamente com aplicações visuais e com um vasto suporte de detecção facial podendo posteriormente se transformar em uma autenticação.

2. Referências Bibliográficas

Com o propósito de contribuir para a ampliação do conhecimento da área, o presente artigo científico tem como objetivo analisar os resultados obtidos por meio da autenticação facial baseada na linguagem de programação Python, utilizando algumas de suas bibliotecas voltadas para visão computacional.

Este estudo trás como revisão bibliográfica a temática de autenticação facial, utilizando linguagem de programação Python, a definição do tema, as questões de pesquisa levantadas, objetivos e sua implementação são meios de se obter resultados a respeito de sua acurácia. Para tal, realizou-se a utilização de diversos artigos científicos, com seus temas voltados a autenticação facial e seus métodos, reconhecimento facial e sua história, linguagem de programação Python.

A busca foi realizada de maneira remota, por meio de pesquisa de artigos científicos, monografias, sendo periódicos disponíveis no Google Acadêmico.

Sendo selecionado como analise, todos os artigos que mencionassem ”Autenticação”, ”Reconhecimento Facial”, ”Visão Computacional”. Priorizou-se por incluir as palavras chaves como ”Autenticação Facial”, já que é a definição mais clara do que será executado neste artigo. Com o material obtido, foi possível o desenvolvimento deste documento e suas demais citações, bem como seus devidos testes.

3. Metodologia

No quesito modo de pesquisa, foi aderido obter resultados, informações e aprendizado existentes em artigos, vídeo aulas, e sites como GitHub, para fins de teste com o reconhecimento facial, utilizando informações relevantes a respeito da biblioteca *face recognition*, que se dedicavam explicitamente em apresentar sobre os seguintes temas, autenticação facial, linguagem de programação Python e suas bibliotecas.

Para efetuar a programação foi utilizado o Visual Studio Code. É o editor de código-fonte da Microsoft para Windows, Linux e macOS. Ele inclui suporte para depuração, controle de versão Git integrado, realce de sintaxe, conclusão de código inteligente, *snippets* e refatoração de código.

Ao longo da elaboração do artigo existiu alguns obstáculos para se obter o conteúdo necessário, sendo um deles, instalar todas as bibliotecas necessárias, levando a problemas na execução do código, sendo o principal detalhe, a versão do *Python*, onde a biblioteca do *face recognition* falhou na versão mais atual até o momento da publicação deste artigo, sendo ela *Python 3.9.7*.

Foi executado a aplicação com êxito, com a versão do *Python 3.8.3*, onde não existiu falhas significativas. Porém a biblioteca utilizada exige uma quantidade considerável de processamento, onde o video apresenta uma quantidade baixa de quadros por segundo.

4. Resultados e Discussão

Todos os testes foram realizados em um computador de mesa, com as seguintes configurações, processador Intel i5 750, 8 GB de memória RAM, placa de video GTX 750 ti, sendo usado uma *webcam*, com a capacidade máxima de gravar vídeos em 480 pixeis.

Com a ferramenta Visual Studio Code, foi possível executar o código em Python, onde foi possível realizar todos os testes desejados.

Seguindo os processos que o programa executa, temos, carregar a foto, obter apenas o rosto da foto, transformar o rosto em um vetor, cadastrar esse vetor no banco, por fim comparar o rosto visualizado na *webcam* com os dados no banco e retornar se esse rosto existe na base dados ou não.

Ao realizar um teste utilizando a imagem apresentado na Figura 1, inicialmente o programa vai carregar a imagem, depois localizar apenas o rosto nessa imagem.

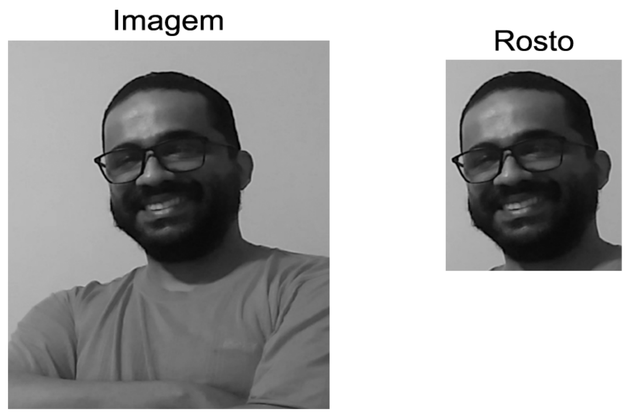

Figure 1: *A biblioteca face recognition irá localizar o rosto na imagem, para processar apenas a face.*

A partir do rosto encontrado na Figura 1, é gerado um vetor estruturado e apresentado na Figura 2.

```
[-9.31140333e-02  1.36347309e-01  5.76222986e-02 -3.95874828e-02
 8.10826314e-04 -7.97739625e-03  3.56786996e-02 -4.16545803e-03
 1.73047379e-01 -4.78014462e-02  1.90434322e-01 -5.37134334e-02
-1.87315032e-01 -9.71490815e-02 -1.75816361e-02  1.08819373e-01
-1.83032200e-01 -1.10453986e-01 -2.01985501e-02 -5.74079342e-02
 1.65647827e-02 -5.67717571e-03  1.87942735e-03  5.61386943e-02
-9.84536484e-02 -4.08220977e-01 -8.91670734e-02 -1.94804385e-01
 5.63235059e-02 -1.59697443e-01 -2.04914100e-02  5.33060580e-02
-1.62762374e-01 -2.23314390e-02 -6.25901446e-02  3.67623344e-02
-7.90311210e-03 -4.73111942e-02  1.38298973e-01  8.62961262e-02
-6.87812269e-02  1.11290412e-02 -1.12692509e-02  3.72567505e-01
 2.08722264e-01  2.72146259e-02 -3.11672688e-04  3.97700891e-02
 6.31182492e-02 -2.21053973e-01  2.25325581e-02  1.05703309e-01
 8.50314423e-02  8.33195075e-02  5.31794652e-02 -1.14681661e-01
 2.23344024e-02 -2.32209004e-02 -2.22390860e-01  6.08249567e-02
 2.17610933e-02 -2.61916276e-02 -5.59190288e-02  1.18112676e-02
 2.97494650e-01  9.49992910e-02 -1.15631178e-01 -4.00296822e-02
 1.73038945e-01 -1.56199053e-01  3.83101888e-02 -3.96796837e-02
-1.43434525e-01 -1.13570675e-01 -1.96788773e-01  1.53188497e-01
 3.51463079e-01  1.66165248e-01 -1.22764722e-01  4.38887775e-02
-1.52755916e-01 -1.61146279e-04  1.97935523e-03  5.96992187e-02
-1.46991387e-01  3.20443958e-02 -6.91718757e-02  7.82248229e-02
 8.86496753e-02  2.18483154e-02 -3.27690616e-02  1.97523877e-01
-5.96015938e-02  3.41963954e-03  3.45214717e-02 -7.30164871e-02
-8.55271593e-02 -1.67789571e-02 -1.11325935e-01 -2.49817930e-02
 4.13780138e-02 -1.57796279e-01 -1.34396367e-02  1.44075021e-01
-2.15163201e-01  8.87601003e-02  2.36421395e-02 -4.41100076e-02
 2.99421623e-02  1.04323268e-01 -1.33155674e-01 -7.17625171e-02
 1.36491194e-01 -2.37803206e-01  1.76193103e-01  1.98602721e-01
 7.50521123e-02  1.18576325e-01  4.24303859e-02  5.71078658e-02
-1.64657626e-02 -7.37689435e-04 -1.09145984e-01 -1.09757613e-02
 4.75674495e-02  2.66618654e-02 -9.81043559e-03  1.52333416e-02]
```

Figure 2: *Vetor criado através do rosto obtido da imagem.*

O vetor informa ao programa que esse conjunto de dados representam uma face de um individuo, após essa etapa o programa já consegue identificar individuo por meio de seus vetores.

Na Figura 3 é exibido o resultado obtido após o teste de autenticação, o sistema identifica o rosto da pessoa comparando o vetor com os vetores que estão no banco de dados, caso o sistema consiga achar o vetor correspondente ao rosto do individuo, significa que ele está cadastrado, dessa forma o sistema retorna o nome do usuário, caso o sistema não consiga identificar o indivíduo, ele exibe a mensagem ”Não identificado”

Figure 3: *Rosto detectado após gravar os dados do vetor no banco de dados.*

Segundo a documentação da biblioteca *face recognition*, ela foi construída usando o reconhecimento de rosto de última geração da dlib, que foi construído com aprendizado profundo. O modelo tem precisão de 99,38 no referencial *Faces in the Wild* [2]

Os resultados obtidos foram satisfatórios, pois ao testar com uma parcela de pessoas, o programa não teve dificuldade em identificar elas, assim alcançando seu objetivo de identificar e autenticar faces humanas.

5. Conclusão

Analisando os tipos de autenticação, senhas ou palavras-passe, que geram ao usuário a dependência da memorização ou que faça ele gravar essa informação em algum lugar, são métodos obsoletos, já que se o usuário não for capaz de lembrar, terá perdido o acesso ou se anotar, fará essa senha perder o seu propósito, pois outras pessoas podem conseguir ela através de outros meios.

Pensando em *tokens* e em cartões de autenticação, podem ser comparados a uma chave, que permite que o usuário possa abrir uma porta, onde terá o acesso autenticado, sendo uma de suas desvantagens o objeto autenticador.

Considerando a autenticação biométrica, ela não cria dependência alguma com a memória ou a necessidade de se ter um objeto autenticador, pois já seria autenticado pelas suas características biométricas, tais como as íris dos olhos, seus dedos, palmas das mãos, voz ou rosto.

Tudo isso só foi possível por conta da evolução exponencial da tecnologia, junto á capacidade do ser humano aprender e se adaptar, permitindo que a raça humana, criasse algoritmos aparentemente simples, porém ao mesmo tempo com um grau complexidade.

References

[1] SHARIF, M. Face Recognitio: Survey. *[s.n.], 2017. Disponível em: https://pdfs.semanticscholar.org/bb86/bed5f8b98c65a4f882858523bb8ee12ad6ba.pdf.* Acesso em: 30 ago 2021.

[2] GEITKEY, A. Reconhecimento facial. *[s.n.], 2017. Disponível em: https://face-recognition.readthedocs.io/en/latest/readme.html.* Acesso em: 04 dec 2021.