

Nome Completo: Daniel Ferraz de Campos Filho / Matrícula: A57635769

Mini Paper

O projeto desenvolvido pelo grupo foi uma extensão do tema explorado na disciplina Desafios e Requisitos dos Projetos Analíticos.

Instruções

O trabalho poderá ser feito em até 5 pessoas.

O objetivo do trabalho deve contemplar assuntos da aula (deep neural network, convolutional neural network, recurrent neural network) e deve ser escolhido por cada grupo. Na nossa última aula, dia 15/06, os alunos devem apresentar o grupo e qual o tema e o objetivo específico do trabalho. Ex.: 'O grupo, composto pelos alunos, terá como trabalho a estimativa dos casos de covid no Brasil utilizando arquiteturas de redes recorrentes, como LSTM e GRU.' Os grupos que tiverem dificuldade de escolher projetos, podem entrar em contato comigo. Algumas sugestões/ inspirações podem ser obtidas em sites de competição (ex Kaggle), mas a solução final deve conter elementos originais.

A entrega final deve ser feita individualmente por cada aluno até o dia 31/07 e deverá conter:

Os notebooks/ scripts com o trabalho feito e as soluções produzidas pelo grupo (comum a todos os membros do grupo);

Documentação individual, explicando o que foi feito no trabalho, problemas encontrados e possibilidades de melhoria da solução, em formato de 'mini papers' (2-5 páginas). Cada um deve ter a sua submissão de mini-paper.

31/julho: Entrega do notebook / scripts com o trabalho e o mini paper individual, no EClass.

Introdução

Atualmente presenciamos situações sem precedentes no contexto da saúde ao que nos referimos de pandemia Nossa sociedade enfrenta o desafio de combate e prevenção contra o crescente alastramento da COVID-19 em uma escala global. Deste desafiador contexto, emergem efeitos colaterais inéditos na era moderna, como o distanciamento social, dentre diversas mudanças de hábitos. Com base nas evidências atuais, o vírus COVID-19 é transmitido entre pessoas através de contato próximo e gotículas. Calcula-se que uma pessoa com infecção o transmita para de duas a quatro pessoas.

Recentemente, em 3 de Julho de 2020 também foi sancionado pelo presidente da república, a Lei de número 14.019 de 02/07/2020 que determina o uso obrigatório de máscaras de proteção individual para circulação em espaços públicos e privados acessíveis ao público, em vias públicas e em transportes públicos durante a vigência das medidas para enfrentamento da emergência de saúde pública de importância internacional decorrente da pandemia da Covid-19.

Sob este contexto decidimos aplicar algumas técnicas de deep learning para o desenvolvimento de um modelo capaz de detectar a utilização de máscara de proteção individual.

Desenvolvimento e Metodologia

Para este trabalho, foi utilizada uma base de dados de imagem pública contendo um total de 90.000 imagens de faces sem máscaras e 2.203 imagens de faces com máscaras faciais e outra base pseudo artificial, produzida por Prajna Bhandary, com 1.376 imagens das quais 690 imagens estão classificadas como faces com máscara e 686 para a classe complementar. O conceito de pseudo-artificialidade mencionado é uma liberdade poética dos autores para expressar que, apesar de serem imagens reais de pessoas, as máscaras foram artificialmente criadas e posicionadas, via algoritmo, para o específico propósito de treinamento de modelos como o modelo que se pretende criar.

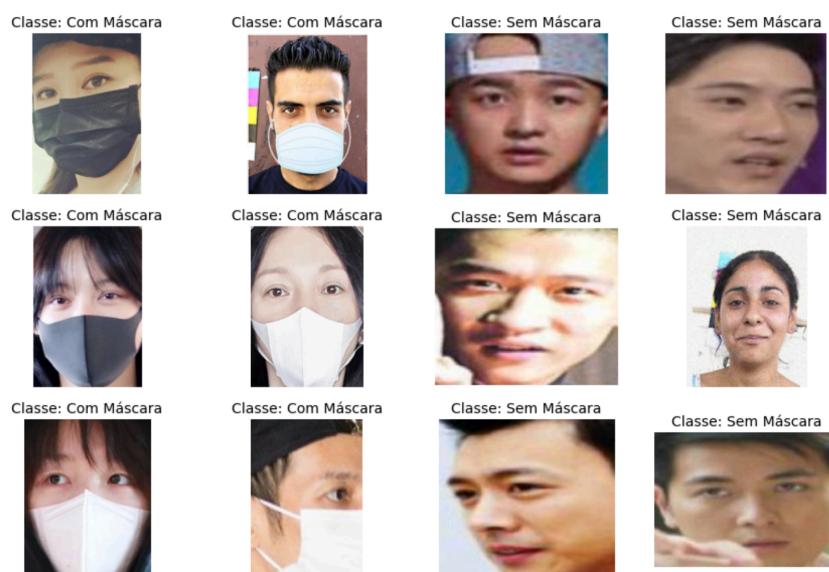


Figura 1: Exemplo da base de dados usada

Sabemos o quanto importante é termos um banco de dados robusto e grande o suficiente para habilitar a generalização do nosso modelo.

Com isto em mente, uma prática comum aplicada à imagens é o processo de *data augmentation* e será o processo que usaremos para fazer o enriquecimento de nosso banco de imagens.

Este processo basicamente toma as imagens originais e aplica pequenas modificações nela tais como **rotação, translação, cisalhamento, ampliação e espelhamento** a fim de gerar novas imagens para o treinamento dos modelos. O conceito por trás desse processo é assumirmos que uma imagem continua representando a mesma classe mesmo com estas pequenas modificações, permitindo com que tenhamos outras óticas sobre a mesma imagem.

A seguir exibimos alguns exemplos desse processo. Adicionalmente, aplicamos o mesmo procedimento que será usado para treinamento dos modelos tal qual uso de **escalas de cinza ou esquema de cores e redimensionamento de imagens** para dimensões quadradas.

Esse contexto pode ser considerado como o equivalente à *feature engineering* em bases de dados tradicionais (não-imagens). A seguir os métodos usados para este processo:

- **rotation_range** é um valor em graus (0-180), um intervalo no qual rotacionaremos aleatoriamente as imagens.
- **width_shift** e **height_shift** são intervalos (como uma fração entre 0 e 1 da largura ou altura total) dentro dos quais é possível transladar aleatoriamente imagens na vertical ou na horizontal.
- **rescale** é um valor pelo qual multiplicaremos os dados antes de qualquer outro processamento. Nossas imagens originais consistem em coeficientes RGB no intervalo de 0 a 255, mas é recomendável a normalização dos dados para o processamento dos nossos modelos (dada uma taxa de aprendizado típica); portanto, transformaremos em valores entre 0 e 1 pela multiplicação do fator 1/255.
- **shear_range** é o intervalo o qual aplicaremos, aleatoriamente, transformações de cisalhamento.
- **zoom_range** é o intervalo destinado à transformações aleatórias de ampliação.
- **horizontal_flip** argumento do tipo booleano indicando que transformações aleatórias de espelhamento horizontal são permitidas - relevante quando não há suposições de assimetria horizontal (e.g. imagens reais).
- **fill_mode** é a estratégia usada para preencher pixels recém-criados, que podem aparecer após alguma das transformações previamente mencionadas.

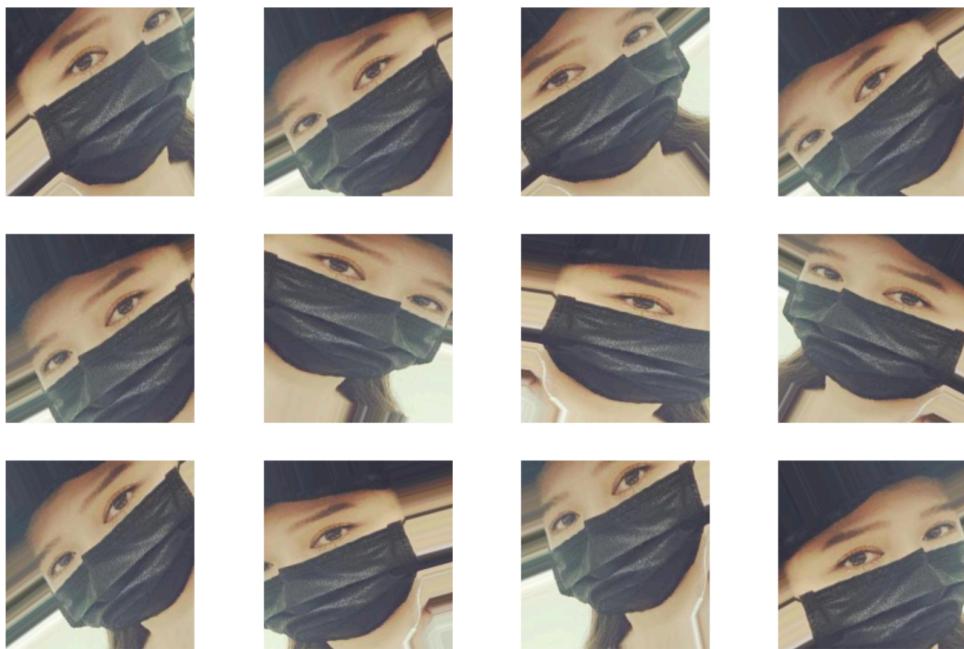


Figura 2: Exemplo da aplicação dos métodos de feature engineering.

Todas as arquiteturas de modelos foram treinadas com o mesmo conjunto de treino e teste para garantirmos a justiça entre elas.

A proporção definida foi de **75%** para o **treino** e **25%** para o **teste**. Para garantirmos que as classes estejam equilibradas em ambas as amostras, estratificamos pela coluna de classes. O framework utilizado foi o *train_test_split* do **scikitlearn**.

A metodologia usada baseia-se no *framework* do **Keras** de *ImageDataGenerator* (já mencionado e devidamente explicado anteriormente) associado ao *flow_from_dataframe* que cria um fluxo direto do diretório de imagens enquanto a definição de classes fica à cargo de um *dataframe* com o nome do arquivo (imagem) e sua respectiva classe.

Para o desenvolvimento deste trabalho, foram utilizados dois tipos de redes neurais: ANN e CNN. As redes criadas em cada caso estão demonstradas a seguir.

Uma Rede Neural Artificial (do inglês, Artificial Neural Network ou ANN) são sistemas de computação inspirados nas redes neurais biológicas que constituem cérebros de animais. Esses sistemas são uma coleção de unidades ou nós conectados chamados de neurônios artificiais. Cada conexão, como as sinapses no cérebro biológico, pode transmitir um sinal para outros neurônios. Nas ANNs pode-se ter várias camadas de neurônios, aumentando a complexidade da rede.

Treinamos, portanto, dois modelos, um com a etapa convolucional e outro apenas com a etapa densa por 150 épocas, abaixo podemos visualizar o comportamento da curva de aprendizado de ambos os modelos em função do resultado de nossa função de perda (binary crossentropy) e da acurácia dos mesmos nas bases de treino e testes, conseguimos verificar que o processo de aprendizado da rede CNN apesar de ser mais errático na base de treino acaba convergindo rapidamente na base de testes chegando a alcançar uma acurácia em testes de 99.66% frente a 91.5% da rede neural com apenas as camadas densas.

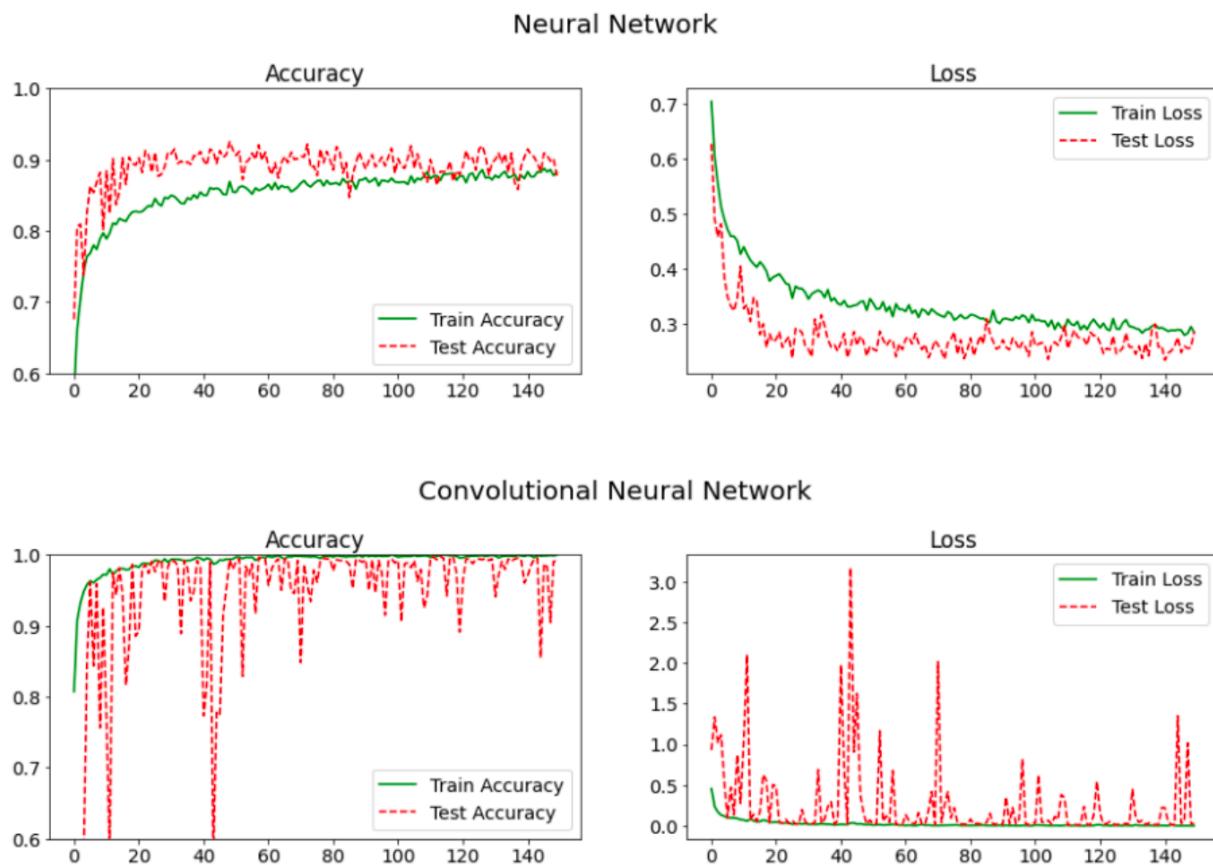


Figura 3: Resultado dos modelos

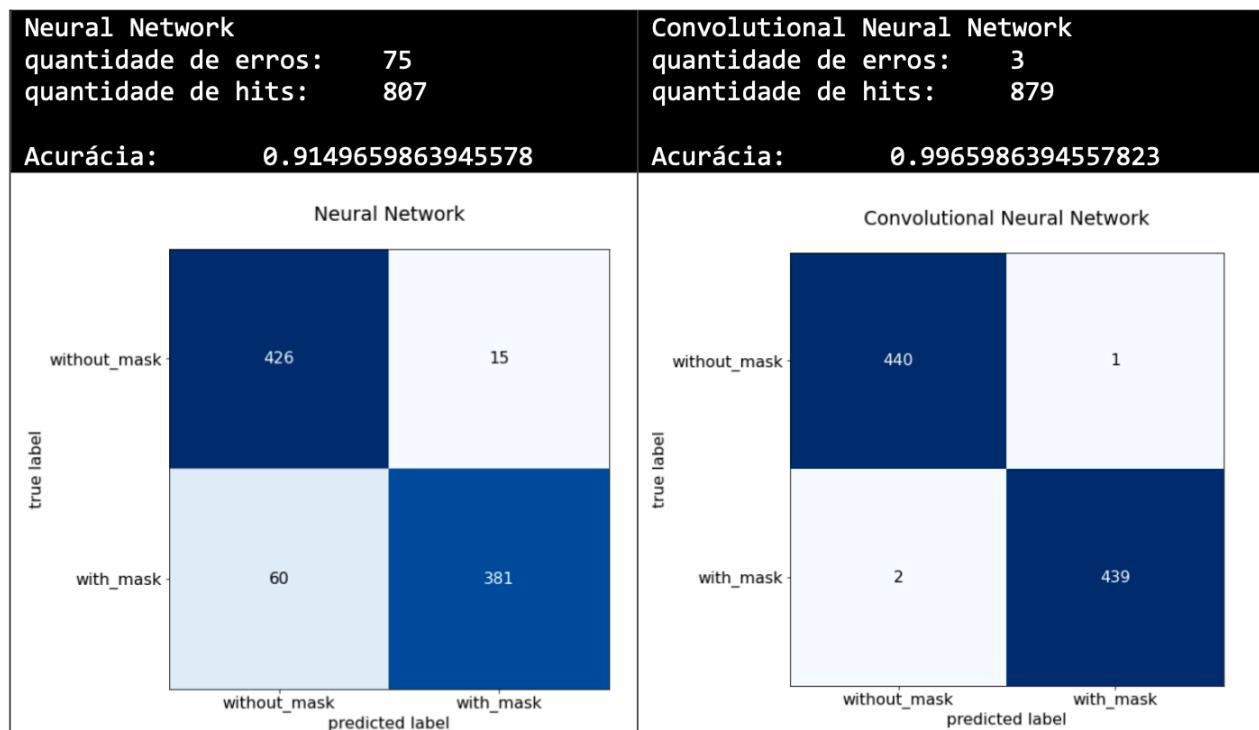


Figura 4: Acurácia dos modelos treinados

Podemos observar que ambos os modelos atingiram resultados surpreendentes na classificação das imagens. Porém, como esperado (baseado em literatura), os resultados do modelo com etapas convolutivas (CNN) foram bastante superiores ao modelo ANN.

Conclusão

De maneira geral, o projeto atingiu com sucesso o objetivo esperado. Assim, é possível aplicar este framework em implementações de sistemas de controle de entrada em espaços públicos, estabelecimentos que executem atividades essenciais, repartições públicas estaduais, transporte por aplicativo para um público alvo de consumidores, fornecedores, clientes, empregados, colaboradores, agentes públicos e prestadores de serviço.

Por fim, concluímos que esta solução é uma possibilidade viável para a ajuda no combate ao COVID-19.

Referências

CHOLLET, F. Deep Learning with Python. Manning. ISBN 9781617294433. NY, USA. 2018.

Keras API Reference. Disponível em <https://keras.io/api>. Acesso em: 28 de Julho de 2020

Scikit Learn Documentation. Disponível em <https://scikit-learn.org/stable/>. Acesso em 28 de julho de 2020.

OpenCV Python Documentation. Disponível em <https://pypi.org/project/opencv-python/>. Acesso em 28 de julho de 2020.