**ALUNO(S): Gilberto Luis De Conto Junior**

1. **INTRODUÇÃO**

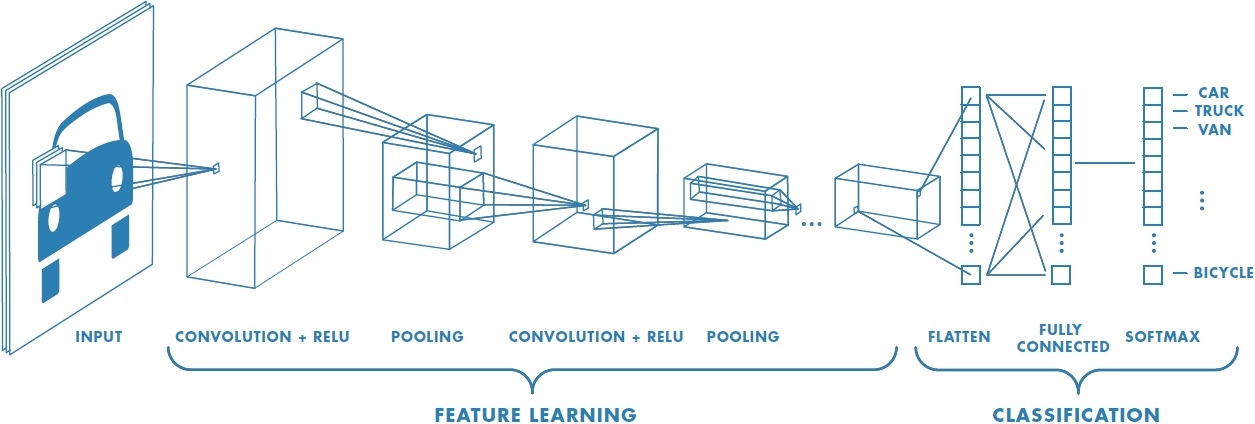
O intuito deste trabalho é utilizar uma rede neural junto com a técnica do *fine-tune* para classificar imagens de oito personagens da Marvel através de uma base de imagens com 2513 exemplos ao todo. A rede recebe um conjunto de dados para treino e gera um modelo que é aplicado para avaliar as imagens de teste.

Esta técnica permite que a rede aproveite os pesos gerados por outra rede que foi treinada em uma base de dados maior, para o exemplo o qual se quer resolver. Isso possibilita que uma rede treinada em uma base de dados menor consiga ter um melhor desempenho aproveitando o conhecimento já obtido pela rede com maior número de exemplos. Para este trabalho foram selecionados dois modelos pré-treinados para fins de comparação, o MobileNetV2 e o InceptionV3.

Ao final do trabalho serão expostos os resultados de cada modelo bem como a matriz de confusão gerada por cada um de modo a identificar qual performou da melhor maneira nesta base, com estes parâmetros.

1. **ALGORITMOS DE APRENDIZAGEM DE MÁQUINA**

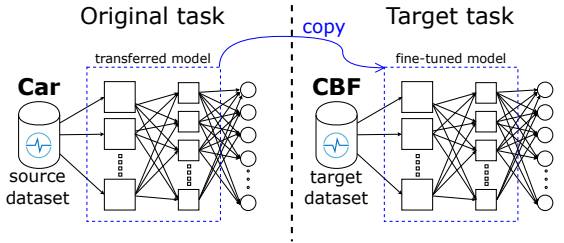
Neste trabalho foram empregadas técnicas de aprendizagem supervisionada através do uso de redes neurais artificiais. As CNN’s ou redes neurais convolucionais funcionam de acordo com a figura abaixo.



Fonte: MathWorks.com (entre 2000 e 2019)

Conforme a ilustração acima em uma CNN a imagem é ‘desmontada’ e cada um de seus pixels passam pelas camadas da rede, sejam elas convolucionais, de *pooling*, densas entre outras...

O *transfer learning* é uma técnica que de acordo com Fawaz et al (2018) é o processo de primeiro se treinar uma rede neural tendo em base uma fonte, e em seguida transferir as características aprendidas pela rede (chamados de ‘weights’, ou, pesos), para uma segunda rede que será treinada em outro dataset.



Fonte: FAWAZ et al (2018)

Na figura acima percebe-se um exemplo desta técnica na qual o primeiro modelo é treinado com o *dataset* ‘*Car’* e os pesos são transferidos para o segundo exemplo treinado em outro *dataset* aproveitando o conhecimento obtido pela primeira rede. Alguns exemplos de modelos utilizados nesta técnica são: InceptionV3, MobileNetV2, ResNet50, VGG 16, VGG 19, entre outros.

1. **METODOLOGIA**

Nesta seção serão abordados os temas pertinentes à metodologia aplicada neste trabalho.

* 1. Base de Dados

A base de dados inicial contava com 3.055 imagens divididas entre os oito personagens, porém, percebeu-se inconsistência nas mesmas tendo em vista que algumas imagens não eram condizentes com sua classe, ou seja, nem todas as imagens de cada classe eram de fato daquele personagem, ou a imagem era de outro personagem ou possuía vários personagens juntos sem que o personagem a ser representado fosse claramente o principal da foto, como por exemplo, na classe do Hulk, imagem de um poster de um filme no qual o Hulk não aparece, ou aparece com menos destaque do que outro personagem.

Para resolver este problema foi realizado uma vistoria manual em toda a base de dados de modo a garantir que as imagens de fato representem o personagem da classe. A tabela abaixo representa a distribuição final de exemplos para cada classe.

|  |  |
| --- | --- |
| Personagem | Número de Imagens |
| Black Widow | 296 |
| Captain America | 327 |
| Doctor Strange | 316 |
| Hulk | 292 |
| Ironman | 321 |
| Loki | 307 |
| Spider-Man | 371 |
| Thanos | 283 |

Ao final do refino de imagens, como pode-se perceber na tabela acima, resultaram 2.513 imagens ao todo.

* 1. Pré-Processamento

Para fazer a leitura das imagens da pasta para o sistema, foi utilizado um *for* para percorrer os subdiretórios e imagens, e armazenar o caminho e a classe de cada uma delas em um *DataFrame*, estrutura pela qual facilitará a obtenção dos arquivos. Com o *DataFrame* devidamente alimentados foi utilizado a função do *Keras* chamada *train\_test\_split* que divide o *DataFrame* em dois, um para treino e um para teste, o primeiro com 70% dos dados e o último com os outros 30%. A base de treino é a responsável pelo treinamento da rede, é através dela que o algoritmo vai ‘aprender’ seus padrões e peculiaridades. A base de teste serve para, após obtido o modelo treinado, avaliar a acurácia do modelo bem como outras métricas de desempenho que serão mostradas adiante.

Após isso foi utilizado biblioteca *ImageDataGenerator* aplicando a função de pré-processamento do modelo e também a divisão de 30% da base de treino para destinar à validação. Os resultados dessas variáveis foram usados como parâmetro para outra função chamada *flow\_from\_dataframe* que é responsável por percorrer os diretórios do *DataFrame* e ler as imagens tanto de treino e validação como de teste e atribui-las à variáveis que serão inseridas na rede.

* 1. Execução do Algoritmo

Para a realização do trabalho foi utilizado o Python como linguagem e o Spyder como interpretador, o Keras para a montagem da rede, e duas redes neurais pré-treinadas, os MobileNetV2 e o InceptionV3 para fins de comparação. Ambos os modelos foram testados com diferentes mudanças em suas estruturas como redução de *learning rate*, aumento de camadas e troca de funções otimizadoras, porém em ambos os casos o melhor resultado se deu com a mesma configuração que pode ser observada abaixo.

|  |  |
| --- | --- |
| **Camadas Incluídas** | 3 Camadas Densas, sendo duas com 128 neurônios cada e função *sigmoid* como ativação, e uma com oito neurônios representando as oito possíveis classes de saída com ativação *softmax* |
| **Otimizador do modelo** | Adam com taxa de aprendizado padrão |
| **Função de erro** | *categorical\_crossentropy* |

A rede padrão do MobileNetV2 conta com 53 camadas de profundidade, é treinada com o *dataset imagenet* que possui mais de um milhão de imagens. Esta rede recebe imagens com tamanho de 224x224. Já a rede InceptionV3 tem 48 camadas também é treinada com o *imagenet* e pode receber imagens com diferentes tamanhos de 80x80 até 299x299, neste trabalho foi utilizado a mesma dimensão do MobileNetV2, 224x224 (MATHWORKS, 2021).

Em ambas as redes, foi passado o parâmetro *include\_top* como *false* indicando que as ultimas camadas do modelo original não serão utilizadas neste caso pois estas são responsáveis pela classificação específica do modelo original e no caso deste trabalho, esta classificação será dentre as oito classes dos personagens do *dataset*. Para substituir as camadas do modelo original, foram incluídas três camadas densas conforme a tabela acima explicita.

Outra técnica utilizada foi o *EarlyStopping* que é responsável por monitorar as iterações entre as épocas da rede e, em caso de aumento do erro por três épocas seguidas, o algoritmo para de iterar para evitar um resultado indesejável. Juntamente com os dados de treino, validação e o número escolhido de épocas (25 neste caso) o *EarlyStopping* foi adicionado ao método *fit* do *Keras* que é responsável por fazer o treinamento da rede.

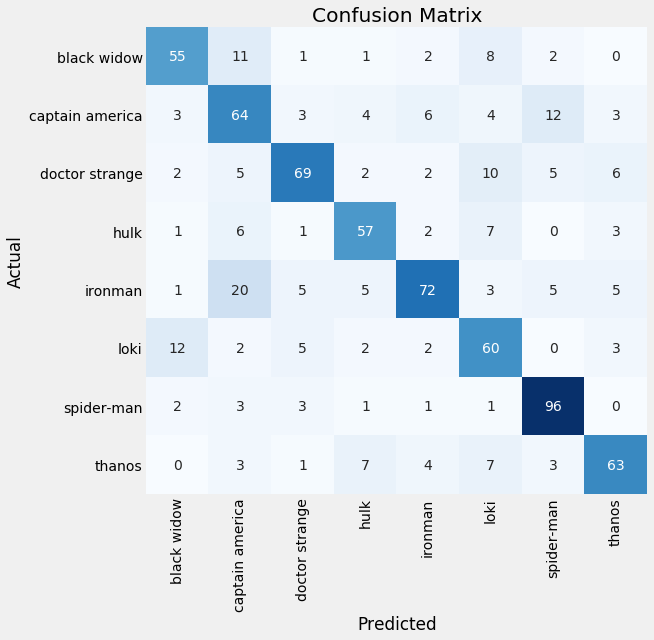
Com o modelo treinado, ele foi aplicado na base de teste e salvo os resultados para *plotagem* na matriz de confusão ilustrada na seção de resultados.

1. **RESULTADOS OBTIDOS**

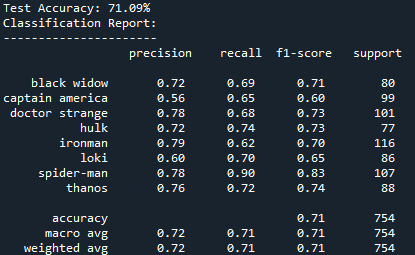
Nesta seção serão apresentados os resultados obtidos pelo modelo na base de teste.

4.1 MobileNetV2

Após treinado o modelo, obteve-se a seguinte matriz de confusão:



Como pode ser observado na imagem acima, não houve um erro crasso em alguma classe e personagens muito específicos como o Hulk que é verde e gigante e o Spider-Man que possui uma roupa e máscara características não apresentaram muitos erros. Em compensação o Capitain America que possuía exemplos com máscara e uniforme e outros apenas com o uniforme, foi confundido com o ironman. Tendo os resultados em vista, a acurácia no teste foi de 71.09%, abaixo será exibido uma imagem com cálculos de desempenho para melhor exibir.



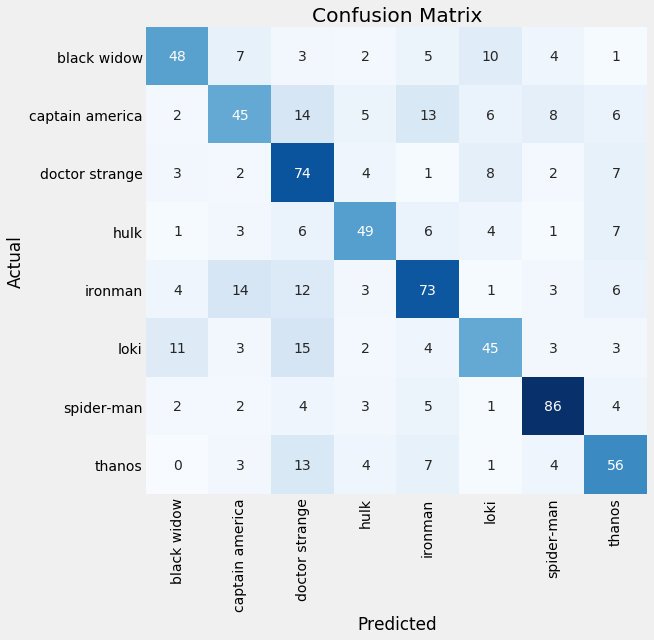
Percebe-se uma uniformidade de aproximadamente 71% de média em métricas como a precisão que calcula dos objetos classificados como ‘Loki’ por exemplo, quantos efetivamente eram ‘Loki’, o *recall* que calcula dos objetos que são da classe ‘Loki’, quantos foram classificados como tal, e o F-Score que calcula a média ponderada das outras duas métricas.



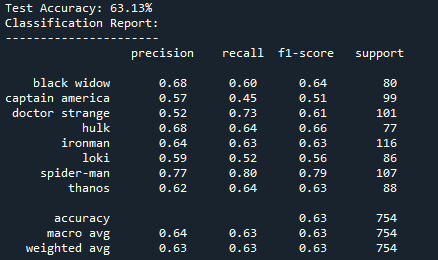
A imagem acima apresenta um exemplo de teste gerado pelo modelo. Em cima destaca-se o nome real do personagem, no meio a foto do mesmo e abaixo em vermelho a predição feita pela rede.

* 1. InceptionV3

Com o modelo InceptionV3 aplicado ao conjunto de teste obteve-se a matriz de confusão a seguir:



A matriz de confusão acima mostra um erro maior que no exemplo anterior principalmente do personagem ‘Loki’. Na próxima figura serão apresentadas as métricas obtidas com o InceptionV3.



Constata-se uma acurácia de 7.96% menor do que o MobileNetV2, apresentando 63.13% na base de testes. Outras métricas como precisão, *recall*, e F-Score também demonstram um menor desempenho do modelo.

1. **CONCLUSÃO**

A proposta do trabalho era utilizar uma rede neural com o auxílio de um modelo pré treinado para melhor classificar as imagens de oito personagens do universo Marvel. O *dataset* inicial contava com 500 imagens a mais do que foi utilizado devido a inconsistências nestas imagens conforme explicado no decorrer do relatório.

Durante o desenvolvimento algumas alternativas foram testadas como redução do *Learning Rate*, alteração do otimizador para outras opções como SGD e RMSProp, aumento e diminuição de camadas e neurônios bem como a inclusão da camada *Dropout* porém sem sucesso. A rede permaneceu no estado o qual foi obtido a maior acurácia dentre o que foi testado.

Conforme os resultados demonstram, a conclusão é que dentre os dois modelos testados nestas condições, o modelo MobileNetV2 obtém uma melhor convergência quando comparado ao InceptionV3 com 71% de acurácia na base destinada à teste.

1. **REFERÊNCIAS**

FAWAZ, H. I.; FORESTIER, G.; WEBER J.; IDOUMGHAR, L.; MULLER, P.A. **Transfer learning for time series classification.** IRIMAS, Universite de Haute-Alsace, Mulhouse, France, arXiv:1811.01533v1 [cs.LG] 5 Nov. 2018;

MathWorks.com. **Convolutional Neural Network**. Disponível em: <https://www.mathworks.com/solutions/deep-learning/convolutional-neural-network.html>. Acesso em: 7 de maio de 2021.

MathWorks.com. **inceptionv3**. Disponível em: <https://www.mathworks.com/help/deeplearning/ref/inceptionv3.html#:~:text=Inception%2Dv3%20is%20a%20convolutional,%2C%20pencil%2C%20and%20many%20animals.>. Acesso em: 7 de maio de 2021.

MathWorks.com. **mobilenetv2**. Disponível em: <https://www.mathworks.com/help/deeplearning/ref/mobilenetv2.html#:~:text=MobileNet%2Dv2%20is%20a%20convolutional,%2C%20pencil%2C%20and%20many%20animals.>. Acesso em: 7 de maio de 2021.