Classificação de tumores cerebrais através de imagens de ressonância magnética

Carlos Augusto Porto Freitas
Engenharia Eletrônica
Universidade Federal de Santa Catarina
carlos.portof@hotmail.com

***Resumo*—O diagnóstico de tumores cerebrais é realizado, normalmente, por profissionais das área de oncologia e radiologia por meio de exames de imagens, como a ressonância magnética, especialmente para áreas sensíveis do corpo humano, como o cerébro. No entanto, tal abordagem está sujeita ao erro humano, por isso sempre buscou-se formas de auxiliar esse processo e, com o desenvolvimento de novas tecnologias como aprendizado de máquina e redes neurais, essa pesquisa tem tido avanços significativos. Este trabalho visa discutir a viabilidade do uso de arquiteturas famosas e de fácil acesso como VGG16, Resnet e Inception, bem como o uso de data augmentation para a tarefa de classificação de tumores cerebrais, específicamente através de imagens de ressonância magnética.**

***Palavras-chave*—Redes Neurais Convolucionais, Deep Learning, Tumor Cerebrais, Classificação de Imagens de Ressonância Magnética, Transfer Learning.**

I. Introdução

O câncer foi a causa mais comum de morte no mundo em 2020, segundo a *OMS* (Organização Mundial da Saúde) [1]. Portanto, tudo que tange prevenção, diagnóstico e tratamento desse doença é de extrema relevância para o quadro de saúde global, em especial o diagnóstico que, muitas vezes pode ser o fator determinante para a sobrevivência do paciente. Nesse contexto percebe-se a importância de um diagnóstico preciso e rápido, usualmente tal processo é realizado através de exames de imagem, normalmente é usado a ressonância magnética, cuja sigla em inglês é *MRI*, por ser menos agressiva, ser indolor e não necessitar de nenhum procedimento cirúrgico, se comparada à exames que não são de imagem como a biópsia, endoscopia, etc. Isso é particularmente verdade para áreas sensíveis, como o cérebro, o qual é o estudo desse projeto.

No entanto, há problemas inerentes na análise da imagem gerada pelo exame, pois, usualmente, o diagnóstico é feito por um profissional da área, um radiologista por exemplo, logo fica-se sujeito ao erro humano, seja a experiência do profissional, seja o cansaço do profissional no momento do diagnóstico, ou qualquer outro empecilho inerente humano. Por isso, a utilização de machine learning, em particular o deep learning, torna-se interessante, com topologias como *CNN* (Redes Neurais Convolucionais) é possível a extração de atributos das imagens provenientes do exames, de modo a treinar um modelo que prediz o tipo e grau do tumor.

Este trabalho tem como objetivo avaliar a possibilidade da utilização de técnicas de transfer learning, particularmente, o uso de modelos pré treinados de fácil acesso, como os da Keras Applications para resolver o problema da classificação de tumores cerebrais através de imagens de ressonância magnética. Dito isso, são utilizados os seguintes modelos da Keras Applications: VGG16, InceptionV3 e ResNet152V2. A escolha dos modelos foi feita de modo a se ter modelos com características diferentes quanto a profundidade das camadas ocultas, número de parâmetros, tempo de treinamento e memória necessária. Além disso, procura-se utilizar data augmentation afim de melhorar os datasets, portanto será testado algumas combinações diferentes de data augmentation, com enfâse na utilização do contraste randômico. Outro ponto importante da metodologia, será a análise do modelo em um dataset diferente do qual ele foi treinado, ou seja, será realizado um domain shift.

A. Trabalhos Relacionados

Atualmente, existem inúmeros artigos sobre implementações de *CNN's* e outras topologias de deep learning para diversos tipos de processamento em imagens de ressonância magnética, a exemplo de classificação de tumores, como no caso deste trabalho, segmentação de tecidos, etc. As principais referências do trabalho são Swati et al. [2] e Badža et al. [3] que propõem arquiteturas baseadas *CNN's* para classificação de tumores cerebrais, sendo que ambos utilizam o dataset apresentado por Jun Cheng [4], o qual também é a base deste trabalho. Os resultados obtidos por essas referências serão discutidos na Seção V.

Além das referências principais este trabalho se utilizou de outros artigos, entre os quais destacam-se Nayak et al. [5], no qual uma parte das arquiteturas propostas neste trabalho foram baseadas, Cheng et al. [6], sendo o artigo que originalmente apresentou o dataset [4] e, por fim, Ul Haq et al. [7], no qual o domain shift foi baseado.

II. Metodologia

Como dito na Seção I o objetivo do trabalho é estimar a viabilidade do uso de transfer learning, na forma de utilizar os modelos conhecidos disponíveis na Keras Applications, bem como avaliar o data augmentation de contraste randômico, uma vez que o contraste da imagem de *MRI* é fundamental para o diagnóstico. Para isso utilizou-se como base os modelos VGG16, ResNet152V2 e InceptionV3 da Keras Applications, além de data augmentation, que serão comentados em detalhe nas Seções II-B e II-D, respectivamente. O trabalho foi realizado na linguagem de programação python, no formato de

um *Jupyter Notebook*, sendo que a plataforma Google Colab foi escolhida para, de fato, executar o script, essa escolha foi feita devido ao acesso a *GPU's* através da platorma. Além disso, o código se beneficia das *API's* do tensorflow e numpy que facilitam grandemente o desenvolvimento e utilização dos modelos, inclusive o tensorflow da acesso direto as *API's* do Keras.

Outro ponto importante para validação de um modelo para diagnóstico médico, baseado em imagens *MRI*, é a generalização do modelo quanto aos tipos de imagens que ele recebe e diagnostica corretamente, isto é, o modelo deve manter resultados próximos aos obtidos no teste mesmo quando recebe como entrada dados de datasets diferentes, caso contrário não há sentido de se utilizar o modelo em produção. Levando em conta essa premissa, utilizou-se um dataset para treinamento que é composto por imagens *MRI* do cerébro feitas em diferentes planos de visão, além disso foi realizado um estudo de um domain shift, isto é, utilizar um dataset diferente para verificar a performance dos modelos com outro dataset. Os datasets usados para treinamento e domain shift serão discutidos em detalhe nas Seções II-A e II-E, respectivamente.

A. Dataset

O dataset usado para treinamento, nomeado doravante de figshare por simplicidade, foi montado por Jun Cheng [4] e é composto por 3064 imagens de *MRI* do tipo T1, o qual basicamente se refere ao realce e luminosidade de diferentes de tipos de tecido na imagem, esse tipo é frequentemente usado para análise de estruturas anatômicas, pois consegue prover bom contraste entre os tecidos moles do corpo humano. As imagens foram adquiridas de 233 pacientes diferentes, sendo coletadas em hospitais chineses entre os anos de 2005 e 2010, sendo que a última atualização do dataset foi em 2017.

Esse conjunto de dados é dividido entre três tipos de tumores cerebrais, sendo 708 imagens de meningiomas, 1426 imagens de gliomas e 930 imagens de tumores pituitários. Além disso, o dataset também provê diferentes planos do cérebro dos pacientes, sendo eles o sagital, coronal e axial.

Um exemplo das imagens disponibilizadas no figshare está na Fig 1 que mostra os três tipos de tumores, bem como os planos de visão das imagens. Outra característica desse dataset é o formato em que ele foi disponibilizado, o *.mat*, que representa o *matlab data format*, portanto será necessário um etapa extra de pré processamento para converter o arquivo em estruturas do python. Além disso, as imagens em preto e branco, contidas dentro do arquivo *.mat* são disponibilizadas no formato de um *array* de dimensão 512x512, no qual os valores de cada pixel são representados no intervalo de 0 à 500, esse fato também adiciona uma etapa extra de pré processamento.

Outro ponto relevante é o desbalanço das classes nesse dataset, bem como o número de imagens disponíveis para cada paciente, visto que o desbalanço de classes pode prejudicar a generalização do modelo, já o desbalanço das imagens por paciente pode impactar a divisão dos conjuntos de treinamento, validação e teste. O primeiro pode ser visto na Fig 2 e o segundo pode ser visto na Fig 3.

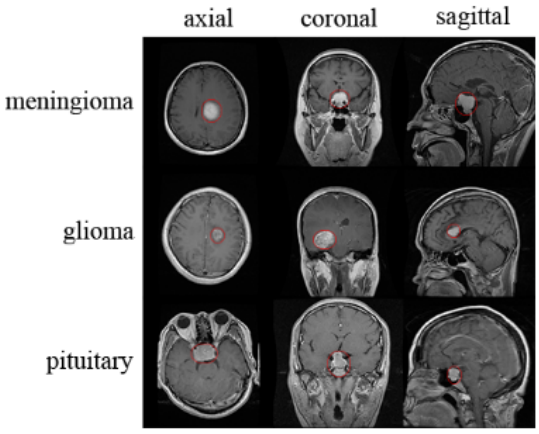

Fig. 1. Exemplo de imagens do dataset fishare, com o tumor circulado em vermelho. Retirada de Badža et al. [3].

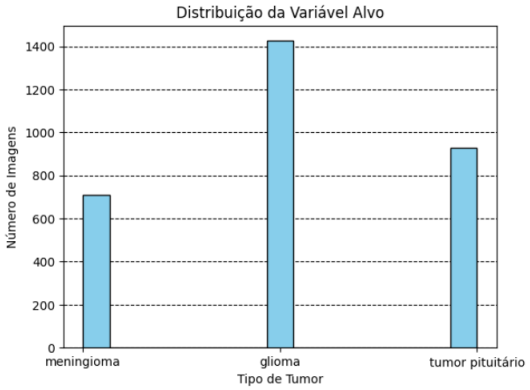

Fig. 2. Distribuição das classes no dataset figshare.

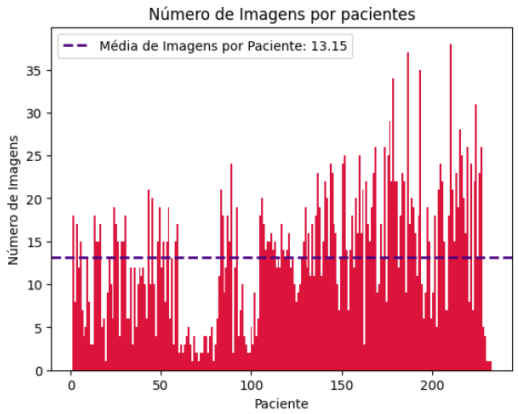

Fig. 3. Distribuição das imagens por paciente no dataset figshare.

B. Modelos

Neste trabalho foram desenvolvidos três modelos baseados em arquiteturas famosas da Keras Applications. esses modelos

foram baseados nas seguintes arquiteturas: VGG16, InceptionV3 e ResNet152V2. A escolha dessas arquiteturas foi feita de modo a se ter modelos com características diferentes quanto a profundidade das camadas ocultas, número de parâmetros, tempo de treinamento e memória necessária.

Os modelos propostos utilizam as arquiteturas do Keras Applications para extrair os atributos das imagens e, ao contrário das arquiteturas originais, eles não terminam em sequência com a camada de classificação, pois foram adicionadas camadas densas, data augmentation e, para evitar overfitting, camadas de dropout. A arquitetura das camadas densas foi baseada no artigo de Nayak et al. [5], onde o fator do dropout foi alterado devido a especificidades dos modelos e os resultados obtidos de etapa de escolha de hiperparâmetros. Por fim, tem-se a camada de classificação multi-classe com ativação softmax. As estruturas completa dos modelos podem ser vistas nas figuras 4, 5 e 6.

Ademais, outras características relevantes são a função perda usada nos modelos, bem como o otimizador utilizado, no caso dos modelos deste trabalho tem-se que a função perda é sparse_categorical_crossentropy do tensorflow, sua equação correspondente é a equação (1), quanto ao otimizador escolheu-se o otimizador Adam.

$$\text{loss} = -\frac{1}{N}\sum_{i=1}^{N}\log\left(\frac{e^{f_{y_i}}}{\sum_j e^{f_j}}\right) \tag{1}$$

onde:

N é o número total de amostras

f_j é a pontuação (logit) associada à classe j

y_i é a classe verdadeira da i-ésima amostra

Layer (type)	Output Shape	Param #
input_2 (InputLayer)	[(None, 224, 224, 3)]	0
tf.__operators__.getitem (SlicingOpLambda)	(None, 224, 224, 3)	0
tf.nn.bias_add (TFOpLambda)	(None, 224, 224, 3)	0
data_augmentation (Sequent ial)	(None, 224, 224, 3)	0
vgg16 (Functional)	(None, 512)	14714688
flatten (Flatten)	(None, 512)	0
dense (Dense)	(None, 720)	369360
dropout (Dropout)	(None, 720)	0
dense_1 (Dense)	(None, 360)	259560
dropout_1 (Dropout)	(None, 360)	0
dense_2 (Dense)	(None, 360)	129960
dropout_2 (Dropout)	(None, 360)	0
dense_3 (Dense)	(None, 180)	64980
dense_4 (Dense)	(None, 3)	543

Total params: 15539091 (59.28 MB)
Trainable params: 15539091 (59.28 MB)
Non-trainable params: 0 (0.00 Byte)

Fig. 4. Sumário do modelo baseado no VGG16.

Layer (type)	Output Shape	Param #
input_6 (InputLayer)	[(None, 224, 224, 3)]	0
tf.math.truediv_1 (TFOpLam bda)	(None, 224, 224, 3)	0
tf.math.subtract_1 (TFOpLa mbda)	(None, 224, 224, 3)	0
data_augmentation (Sequent ial)	(None, 224, 224, 3)	0
inception_v3 (Functional)	(None, 2048)	21802784
flatten_2 (Flatten)	(None, 2048)	0
dense_10 (Dense)	(None, 720)	1475280
dropout_6 (Dropout)	(None, 720)	0
dense_11 (Dense)	(None, 360)	259560
dropout_7 (Dropout)	(None, 360)	0
dense_12 (Dense)	(None, 360)	129960
dropout_8 (Dropout)	(None, 360)	0
dense_13 (Dense)	(None, 180)	64980
dense_14 (Dense)	(None, 3)	543

Total params: 23733107 (90.53 MB)
Trainable params: 23698675 (90.40 MB)
Non-trainable params: 34432 (134.50 KB)

Fig. 5. Sumário do modelo baseado no InceptionV3.

Layer (type)	Output Shape	Param #
input_4 (InputLayer)	[(None, 224, 224, 3)]	0
tf.math.truediv (TFOpLambd a)	(None, 224, 224, 3)	0
tf.math.subtract (TFOpLamb da)	(None, 224, 224, 3)	0
data_augmentation (Sequent ial)	(None, 224, 224, 3)	0
resnet152v2 (Functional)	(None, 2048)	58331648
flatten_1 (Flatten)	(None, 2048)	0
dense_5 (Dense)	(None, 720)	1475280
dropout_3 (Dropout)	(None, 720)	0
dense_6 (Dense)	(None, 360)	259560
dropout_4 (Dropout)	(None, 360)	0
dense_7 (Dense)	(None, 360)	129960
dropout_5 (Dropout)	(None, 360)	0
dense_8 (Dense)	(None, 180)	64980
dense_9 (Dense)	(None, 3)	543

Total params: 60261971 (229.88 MB)
Trainable params: 60118227 (229.33 MB)
Non-trainable params: 143744 (561.50 KB)

Fig. 6. Sumário do modelo baseado na ResNet152V2.

C. *Métricas*

Um fator determinante na availiação de um modelo é a métrica usada visto que, dependendo da aplicação, a escolha errada pode fazer com que o resultado obtido não tenha sentido para aplicação em questão, um exemplo seria utilizar a métrica *RMSE* (Root Mean Squared Error) para uma classificação multi-classe. Este trabalho utiliza como métrica principal a acurácia, pois é a métrica normalmente usada pelos artigos para avaliação da perfomance dos modelos de classificação de

tumores cerebrais, portanto, a fim de facilitar a comparação entre modelos escolheu-se tal métrica. Além disso, métricas secundárias como *precision*,*recall* e *f1-score* também foram calculadas para os modelos. As equações para cada uma dessas métricas são (2), (3), (4) e (5), respectivamente.

$$\text{Acurácia} = \frac{\text{Número de predições corretas}}{\text{Número total de amostras}} \tag{2}$$

$$\text{Precision} = \frac{\text{Verdadeiros Positivos}}{\text{Verdadeiros Positivos} + \text{Falsos Positivos}} \tag{3}$$

$$\text{Recall} = \frac{\text{Verdadeiros Positivos}}{\text{Verdadeiros Positivos} + \text{Falsos Negativos}} \tag{4}$$

$$\text{F1-score} = 2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \tag{5}$$

D. Pré Processamento

Como dito anteriormente na Seção II-A faz-se necessário existir uma etapa de pré processamento, isso deve-se, principalmente, a limitação do formato da entrada dos modelos da Keras Applications, os quais delimitam que a entrada seja uma imagem com três canais e que os valores de cada pixel estejam no intervalo de 0 à 255. Como o dataset figshare disponibiliza imagens de um só canal, com um intervalo de valores para os pixel's de 0 à 500 deve-se adequá-las à necessidade dos modelos, para isso começa-se com a *min-max normalization* utilizada também por Swati et al. [2], contudo, ainda é preciso multiplicar cada pixel por 255 para os valores finais dos pixel's estarem no intervalo desejado. Logo, a expressão final torna-se é a seguinte:

$$y_i = 255 * \frac{x_i - \min(x)}{\max(x) - \min(x)} \tag{6}$$

onde:

x_i é o valor do pixel i da imagem

y_i é o valor normalizado e reescalado do pixel i

$\min(x)$ é o valor máximo de um pixel ao longo da imagem

$\max(x)$ é o valor mínimo de um pixel ao longo da imagem

A próxima operação é reescalar o tamanho da imagem para 224x224, pois os modelos da Keras Applications foram desenvolvidos para esse tamanho de imagem, além de que o tamanho original das imagens do dataset tornariam o treinamento muito longo. Após reescalar a imagem, replica-se a imagem duas vezes para formar a imagem final com três canais.

Essas imagens são carregadas no treinamento dos modelos e então aplica-se o data augmentation, isso é feito devido ao número relativamente pequeno de imagens disponíveis, considerando a complexidade e profundidade dos modelos. O data augmentation presente em todos os treinamentos realiza as seguintes transformações:

- Inversão(horizontal e vertical)
- Rotação
- *Zoom*
- Translação

Há também a transformação de contraste randômico, a qual faz parte dos objetivos do trabalho, essa transformação, em específico, aplica um fator de contraste, que controla quanto contraste será aplicado, e então gera novas imagens transformadas, nesse trabalho foram verificados os seguintes valores para esse fator:

- Fator de Contraste de 0%, ou seja, não utilizar essa transformação.
- Fator de Contraste de 0.1%.
- Fator de Contraste de 5%.
- Fator de Contraste de 10%.
- Fator de Contraste de 25%.

E. Domain Shift

O domain shift foi realizado com o dataset montado por Bhuvaji et al. [8], sendo doravante chamado por brain dataset, que se encontra disponível gratuitamente no kaggle e github. O brain dataset é composto por 3264 imagens incluindo as seguintes classes, glioma(926 imagens), meningioma(937 imagens), tumor de pituitária(901 imagens) e sem tumores(500 imagens). Um exemplo de imagem deste dataset pode ser visto na Fig 7.

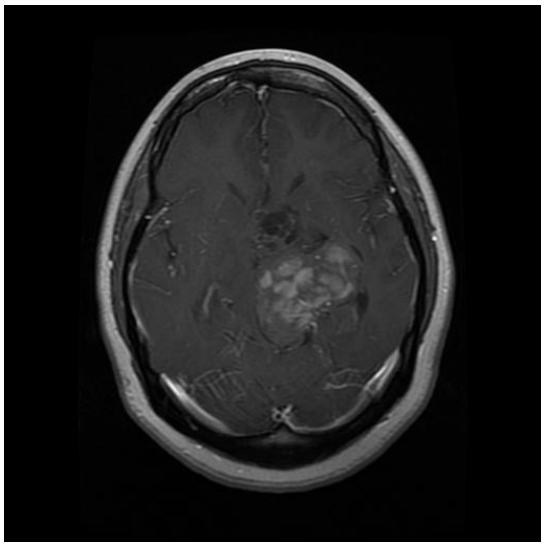

Fig. 7. Exemplo de imagem do brain dataset.

Para o domain shift utilizou-se apenas as imagens no diretório "Training" do brain dataset, o que totaliza 2475 imagens, sendo elas 826 imagens de gliomas, 822 imagens de meningiomas e 827 imagens de tumores pituitários, gerando uma distribuição consideravelmente balanceada, o que ajuda na identificação de problemas devido aos dados de treinamento serem desbalanceados. As imagens estão disponíveis no formato *jpg* e, devido ao propósito do domain shift, não se realiza nenhum pré processamento especial nas imagens, apenas se carrega os *jpg's*, decodifica-se eles e então, por fim,

deve-se reescalar as imagens, agora convertidas em tensores do tensorflow, para a dimensão de 224x224x3.

III. Treinamento

A. Seleção de Hiperparâmetros

B. Data Augmentation de Contraste

IV. Resultados

A. Figshare Dataset

B. Domain Shift

C. Discussão

V. Conclusão

Referências

[1] World Health Organization, "Cancer", Disponível online: https://www.who.int/news-room/fact-sheets/detail/cancer.(acessado em 1 de Dezembro de 2023).

[2] Zar Nawab Khan Swati, Qinghua Zhao, Muhammad Kabir, Farman Ali, Zakir Ali, Saeed Ahmed, Jianfeng Lu, "Brain tumor classification for MR images using transfer learning and fine-tuning, Computerized Medical Imaging and Graphics", Volume 75, 2019, Pages 34-46, ISSN 0895-6111.

[3] Milica M. Badža and Marko ˇC. Barjak-tarovi'c. "Classification of brain tumors from mri images using a convolutional neural network.", 2020, Applied Sciences MDPI.

[4] Jun Cheng, 2017. "brain tumor dataset". figshare. Dataset. https://doi.org/10.6084/m9.figshare.1512427.v5

[5] Dillip Ranjan Nayak, Neelamadhab Padhy, Pradeep Kumar Mallick, Mikhail Zymbler, and Sachin Kumar, "Brain Tumor Classification Using Dense Efficient-Net", 2022, Axioms 11, no. 1: 34. https://doi.org/10.3390/axioms11010034.

[6] Jun Cheng, Wei Huang, Shuangliang Cao, Ru Yang, Wei Yang, Zhaoqiang Yun, Zhijian Wang, Qianjin Feng,"Correction: Enhanced Performance of Brain Tumor Classification via Tumor Region Augmentation and Partition", 2015, PLOS ONE 10(12): e0144479. https://doi.org/10.1371/journal.pone.0144479.

[7] Ejaz Ul Haq, Huang Jianjun, Xu Huarong, Kang Li, and Lifen Weng, "A hybrid approach based on deep cnn and machine learning classifiers for the tumor segmentation and classification in brain mri.", 2022, Computational and Mathematical Methods in Medicine, pages 1–18.

[8] Sartaj Bhuvaji, Ankita Kadam, Prajakta Bhumkar, Sameer Dedge, & Swati Kanchan, "Brain Tumor Classification (MRI)[Data set].", 2020, Kaggle. https://doi.org/10.34740/KAGGLE/DSV/1183165