Análise de duas implementações de CNNs para a classificação de tipos de doenças de rim e modelos de carros

João G. Sousa[1], **Mateus C. Marques**[1], **Yuri M. Neres**[1]

[1]Instituto Federal do Sul de Minas Gerais (IFSULDEMINAS)
{joao1.sousa,mateus1.marques,yuri.macedo}@alunos.ifsuldeminas.edu.br

Abstract. *This article presents the development and evaluation of two Convolutional Neural Networks (CNNs) for classifying renal tomography images and car models using two distinct databases. The main objective was to train the CNNs to categorize images of kidney diseases and vehicle models, respectively. Custom CNN architectures were proposed for each dataset and trained for testing on precision, F1-score, and confusion matrix metrics. The results revealed that the CNN designed for car classification achieved an accuracy of 94.4%, while the CNN for renal diseases achieved 91.5%. These findings underscore the potential of CNNs in accurately classifying different types of images, highlighting their applicability in medical and automotive scenarios.*

Resumo. *Este artigo apresenta o desenvolvimento e avaliação de duas Redes Neurais Convolucionais (CNNs) para a classificação de imagens de tomografia renal e modelos de carros utilizando duas bases de dados distintas. O objetivo central consistiu em treinar as CNNs para categorizar imagens em de doenças renais e modelos de veículos, respectivamente. Para isso, foram propostas arquiteturas de CNNs personalizadas para cada conjunto de dados e treinadas para serem testadas nas métricas de precisão, F1-score e matriz de confusão. Os resultados revelaram que a CNN destinada à classificação de carros alcançou uma precisão de 94,4%, enquanto a CNN para doenças renais obteve 91,5%. Esses achados evidenciam o potencial das CNNs na precisão da classificação de diferentes tipos de imagens, destacando sua aplicabilidade em cenários médicos e automotivos.*

1. Introdução

No cenário atual da revolução digital, a capacidade de interpretar e extrair informações valiosas a partir de imagens tornou-se um pilar fundamental em diversos domínios [Telles et al. 2020]. Setores como a medicina e a indústria automotiva têm testemunhado avanços significativos na aplicação da Inteligência Artificial (IA) para analisar e classificar imagens com precisão sem precedentes.

A identificação por tomografia computadorizada e a forma mais comum e padrão de diagnosticar diversas doenças renais [Poonia et al. 2022], a abordagem convencional e a análise de um médico especialista nas imagens geradas pela tomografia, sendo elas imagens transversais que mostram estruturas anatômicas em diferentes planos. Paralelamente, na indústria automotiva, a capacidade de discernir imagens de modelos de carros desempenha um papel fundamental no desenvolvimento de sistema de transporte inteligente, evoluindo os sistemas de assistência à condução [Liang et al. 2018], a demanda por sistemas de visão computacional capazes de identificar e interpretar com precisão diferentes modelos de veículos por imagens é muito importante para a segurança

e eficiência desses automóveis. Deste modo a classificação precisa de imagens auxilia tanto na identificação de imagens de tomografia quanto na identificação de imagens de veículos. A aplicação de Redes Neurais Convolucionais (CNNs) tornou-se uma âncora essencial nesse cenário.

Desta forma este trabalho propõe o desenvolvimento de duas Redes Neurais Convolucionais (CNNs). Estas CNNs serão treinadas para uma delas classificar imagens de tomografia de rins em 4 classes de diferentes doenças renais, além disso, o segundo modelo será focado na classificação de imagens de modelos de carros em 4 classes de marcas de carros, com isso essas CNNs serão testadas nas métricas de acurácia, precisão f1-score e matriz de confusão, para assim poder verificar a qualidade e eficácia desses modelos na classificação de imagens de tomografia de rins e modelos de carros.

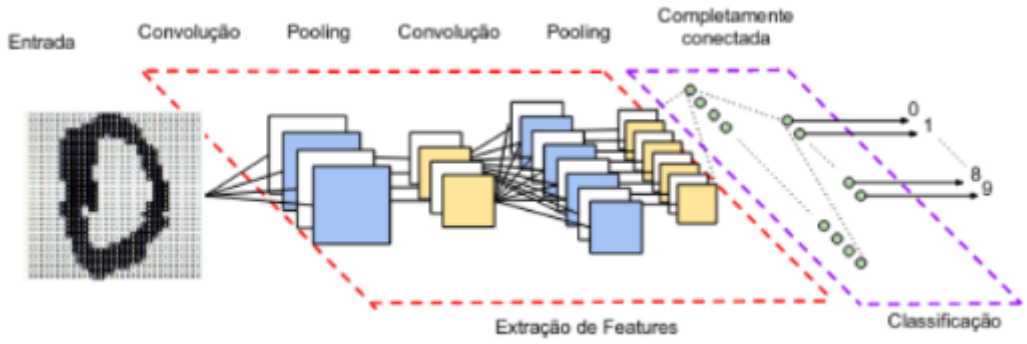
2. Fundamentação Teorica

Nesta seção, será apresentada uma base teórica fundamental para a compreensão e implementação de Redes Neurais Convolucionais (CNNs). Exploraremos a estrutura e o funcionamento das CNNs, detalhando suas camadas, sendo elas a camada convolucional, de pooling e totalmente conectada. Além disso, discutiremos o papel crucial das funções de ativação na introdução de não-linearidades. Abordaremos também o processo de flattening, essencial para a transição dos dados das camadas convolucionais para as camadas densas, juntamente com métricas de avaliação, como precisão, F1-score e matriz de confusão, utilizadas para analisar e avaliar o desempenho das CNNs, serão apresentados os datasets e classes escolhidas, bem como os frameworks e bibliotecas utilizados na implementação das CNNs para análise de imagens.

2.1. Redes Neurais Convolucionais

As Redes Neurais Convolucionais (CNNs) representam uma classe fundamental de arquiteturas de Deep Learn amplamente utilizadas em tarefas de visão computacional e processamento de [Gaba e outros 2022]. Inspiradas nos mecanismos visuais humanos, as CNNs se destacam por sua capacidade de reconhecimento de padrões complexos em dados de alta dimensionalidade, como imagens e vídeos, no reconhecimento de fala e outros campos.

Figura 1. Ilustração da arquitetura de uma LeNet e suas três principais camadas



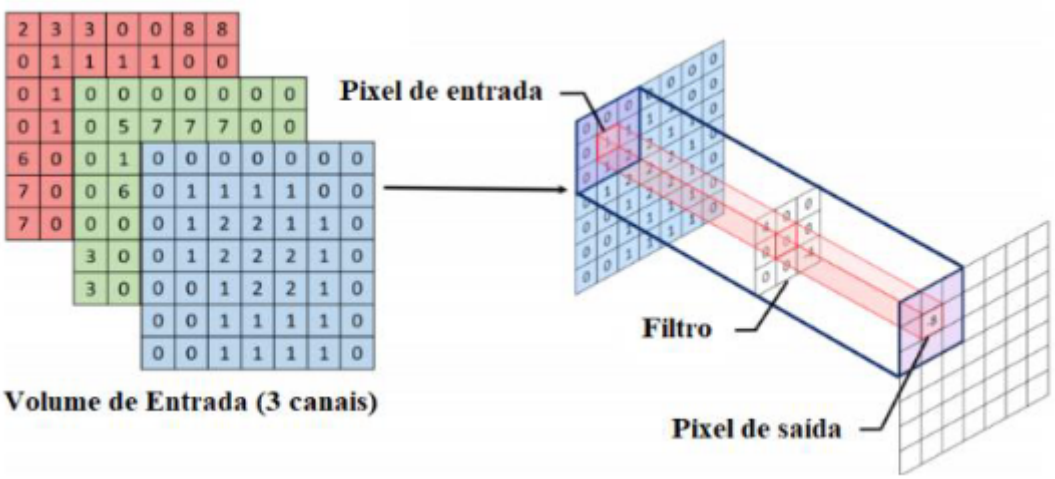
fonte: Goodfellow et al. 2016

As CNNs são formadas por uma sequência de camadas, e cada uma delas possui uma função específica, sendo as três principais camadas: convolucionais, pooling e a totalmente conectada como exemplo a figura 1 [Araújo et al. 2017].

2.2. Camada de convolução

Na camada de convolução é realizado o mapeamento das características mais relevantes da imagem. São utilizados filtros para realizar as convoluções, gerando os mapas de características. Cada filtro possui dimensão reduzida, porém, ele se estende por toda a profundidade do volume de entrada [Araújo et al. 2017]. Por exemplo, se a imagem for colorida, então ela possui três canais (R,G,B), e o filtro tamanho 3 logo o filtro da primeira camada convolucional terá o tamanho 3x3x3 (três de largura, três de altura e três de profundidade). Durante a fase de treinamento da CNN, os filtros são ajustados de acordo com o que está sendo treinado para extrair as características mais relevantes daquela imagem. Na Figura 2 encontra-se uma ilustração da operação de convolução citada anteriormente.

Figura 2. Ilustração de uma convolução de um filtro 3x3



fonte: Goodfellow et al. 2016

2.2.1. Função de ativação

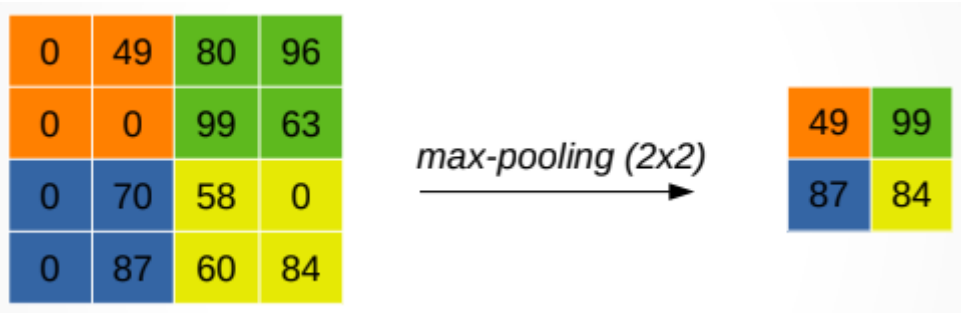
A função de ativação é responsável por introduzir não-linearidade nas saídas das camadas, permitindo que a rede aprenda e represente relações complexas nos dados[Dubey et al. 2022]. Ela determina se um neurônio deve ser ativado ou não com base em uma determinada função. Por exemplo, a função de ativação ReLU (Rectified Linear Activation) retorna zero para valores negativos e o próprio valor para valores positivos, e a sigmoid, que mapeia valores para um intervalo entre 0 e 1, ela são comumente aplicadas depois de uma convolução e depois de uma camada densa.

2.3. Camada de Pooling

Após a camada convolucional, geralmente e usado uma camada de pooling. O objetivo dessa camada é reduzir progressivamente a dimensão espacial do volume de entrada, consequentemente a redução diminui o custo computacional da rede [Araújo et al. 2017]. A

forma mais comum de pooling consiste em substituir os valores de uma região pelo valor máximo, essa operação é conhecida como max-pooling (Figura 3). Essa operação é útil para eliminar valores desprezíveis, reduzindo a dimensão da representação dos dados e acelerando a computação necessária para as próximas camadas, além de criar uma invariância apequenas mudanças e distorções locais.

Figura 3. Ilustração um max-pooling em uma matriz 4x4



fonte: Goodfellow et al. 2016

2.4. Flattening

A camada de Flattening, também chamada de achatamento, é uma etapa específica onde a estrutura de dados tridimensional é transformada em um vetor unidimensional, podendo ser após uma camada de convolução ou uma camada de pooling [Wang et al. 2020]. Essa transformação é crucial para se conectar a camada totalmente conectada, permitindo a transição de características extraídas de imagens ou outros dados com estrutura espacial para um vetor, com isso a próxima camada obrigatoriamente só poderá ser a totalmente conectada.

2.5. Camada totalmente conectada

A partir dos mapas de características da imagem, gerados pelas camadas anteriores, e depois do achatamento(flattening), a camada totalmente conectada é responsável por traçar um caminho de decisão a qual dará a classificação das amostras com o uso da função Softmax na saída(para problemas multi-classes) [Wang et al. 2020]. Ela converte um vetor de números reais em probabilidades, sendo a soma de todas as saídas é igual a 1, ela é aplicada a um vetor de valores reais e transforma esses valores em probabilidades normalizadas, cada valor original é transformado em um valor entre 0 e 1, representando a probabilidade de pertencer a uma determinada classe.

2.6. Datasets

Para o treinamento das CNNs é necessário um conjunto de imagens, por ser um treinamento supervisionado os datasets devem estar organizados em uma pasta para treino, teste e validação, sendo que em cada pasta deve conter as pastas das respectivas classes com as imagens.

Os datasets selecionados foram o Car make, model, and generation [University 2023] que e um dataset de imagens de carros que possui 24 diferentes marcas como Audi, Toyota, Hyundai e etc, e 90 diferentes modelos como Audi Q7, Toyota Camry, Hyundai Accent e etc, e 196 diferentes gerações como o ano do modelo. Com

isso para a implementação da primeira CNN foi definido 4 classes: o modelo Audi a7 com todas suas gerações, o modelo BMW series 7 com todas suas gerações, o modelo Dodge charger com todas suas gerações e o modelo porsche 911 com todas suas gerações.

O segundo dataset é o Multicâncer [Zaman 2023] , composto por imagens médicas de diversas doenças em vários órgãos, como glioma cerebral, tumor renal, etc, abrangendo um total de 22 classes. Nesse contexto, foram selecionadas quatro classes específicas para a implementação da segunda CNN: cisto no rim, rim normal, pedra no rim e tumor no rim, para focar em somente um órgão, facilitando o aprendizado da CNN e os resultados obtidos.

2.7. Métricas

Após todo o processo de treinamento o nosso modelo deverá passar por algumas métricas para comprovar sua qualidade sendo uma dessas métricas a precisão, sendo ela uma medida que avalia a habilidade do modelo em fazer previsões corretas dentre todas as previsões positivas que fez, a precisão indica a proporção de instâncias corretamente previstas como positivas em relação a todas as instâncias previstas como positivas, independentemente de estarem corretas ou incorretas.

Outra métricas que os modelos serão testados e o f1-score sendo ela uma medida que combina precisão e recall em um único valor, fornecendo uma avaliação mais equilibrada do desempenho do modelo de classificação. Ela considera tanto a proporção de instâncias corretamente previstas positivas (precisão) quanto a capacidade do modelo em identificar todas as instâncias positivas (recall).

Por fim a última métrica será a matriz de confusão sendo ela uma matriz que mostra o número de verdadeiros positivos (amostras corretamente previstas como positivas), falsos positivos (amostras erroneamente previstas como positivas), verdadeiros negativos (amostras corretamente previstas como negativas) e falsos negativos (amostras erroneamente previstas como negativas) [Filho 2023].

2.8. Framework

O ambiente utilizado na implantação das CNNs foi o Google Colab, que é uma plataforma de código aberto fornecida pelo Google, que oferece um ambiente de desenvolvimento baseado na nuvem. Seu principal benefício é que ele permite que os usuários escrevam e executem códigos diretamente no navegador, além de disponibilizar GPUs e TPUs, gratuitas para executar os códigos.

A linguagem de programação utilizada foi o python que é a suportada pelo Colab e a implementação das CNNs foi simplificada pela biblioteca do tensorflow, que é uma biblioteca também desenvolvida pelo Google, utilizada para aprendizado de máquina e desenvolvimento de modelos de inteligência artificial.

3. Trabalhos Relacionados

Na literatura, encontram-se vários trabalhos que abordam a utilização de Redes Neurais Convolucionais para as mais diversas coisas como a classificação binária se um carro possui danos ou não, [Pachón-Suescún et al. 2019] em que os autores elaboraram a arquitetura da sua CNN de classificação e utilizaram técnicas como duas camadas de convolução

seguidas para capturar e aprender características complexas e abstratas dos dados de entrada, além do uso da técnica de dropout aplicada na camada totalmente conectada com o objetivo de evitar que o overfitting.

Paralelamente, também se encontram muitos trabalhos acerca da utilização de Redes Neurais Convolucionais, como para classificação binária se o rim possui ou não tumor, [Alzu'bi et al. 2022] no qual propõe 3 modelos de CNN para classificação com arquiteturas pré treinadas, sendo elas CNN-6, ResNet50, VGG16, que são modelos de redes neurais que foram treinados em grandes conjuntos de dados e depois disponibilizados para uso geral, e utilizaram de um pré-processamento de imagens em que as imagens de tomografia foram convertidas para escala de cinza para reduzir as camadas de dados da imagem e também utilizaram de um um processo de data augmentation que tem como objetivo aumentar e diversificar conjuntos de dados de treinamento aplicando transformações controladas e variadas, como rotação, espelhamento, zoom nas imagens já possuídas, como conclusão um de seus modelos o VGG16 obteve uma precisão baixa de 59% , já o ResNet50 obteve 97% de precisão e por último o CNN-6 obteve 95% de precisão.

4. Material e Métodos

Nesta seção, será apresentado os métodos e materiais para a implementação das CNNs propostas. Exploraremos o pré-processamento, que detalha as etapas de preparação dos conjuntos de dados além da definição das funções de camadas, descrevendo como foi implementado para a estruturação das redes convolucionais, abordaremos também a arquitetura, delineando as configurações específicas adotadas para cada CNN. Por fim, o treinamento adotado para ajustar os pesos das redes, incluindo as configurações de épocas e taxa de aprendizado para cada modelo.

4.1. Pre-processamento

O pré-processamento de imagens é uma etapa fundamental para preparar os dados antes de prosseguir com as etapas da criação da CNN pois a qualidade e sentido das imagens é fundamental para a qualidade dos resultados finais.

o dataset multicâncer [Zaman 2023], as classes escolhidas foram rim normal, rim com cisto, pedra no rim, tumor no rim, cada uma delas possui 4000 arquivos para treino, 200 a 700 arquivos no teste, 300 a 700 na validação, a maioria das imagens consiste em uma tomografia do rim porém há algumas imagens de tomografia do tórax onde não é possível identificar a que classe a imagem deveria pertencer, porém é perceptível que as imagens do tórax possuía muito mais pixels brancos que as do rim como pode ver na figura 4, por isso foi feito um processo automatizado em que se somou todos os pixels das imagens e pode constar que as imagens do tórax resultava em valores entre 20 milhões e 25 milhões, já as imagens somente dos rins resulta valores entre 17 a 22 milhões, com isso todas as imagens com mais de 22 milhões foram excluídas, por serem do tórax resultando em 2 mil imagens excluídas ao todo, porém ainda restaram algumas imagens do tórax. Além disso as imagens restantes foram convertidas de RGB para escala de cinza e foram reduzidas de 500x500 para 250x250, para diminuir o processamento. Após esse processo foi feito uma regularização nas pastas de treino teste e validação em que foi feito um data augmentation que é uma técnica para expandir um conjunto de dados existente, para que todas as classes possuísse 4000 imagens no treino 500 no teste e 500 na validação.

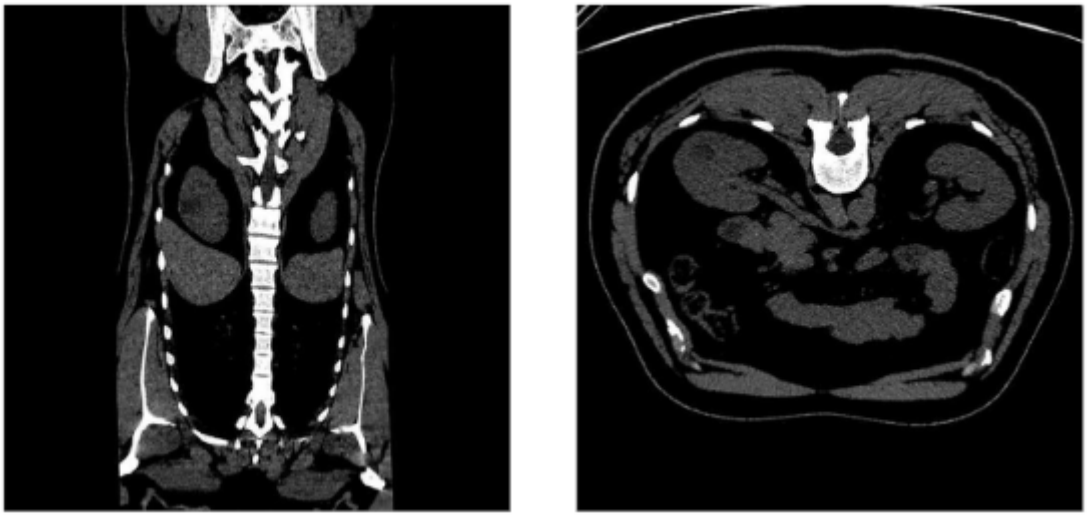


Figura 4. Comparação da imagem de um tórax e de um rim

Já o dataset Car make, model, and generation [University 2023], as classes escolhidas foram Audi a7, BMW series 7, Dodge charger, Porsche 911, com 635, 426, 237, 438 imagens respectivamente, como cada imagem possuía um tamanho, foi padronizado para 500x500 todas. Após esse processo foi feito a separação das pastas de treino teste e validação em que foi feito um data augmentation, para que todas as classes possuíssem 1000 imagens no treino, 250 no teste e 250 na validação.

4.2. Definição das funções de camadas

Para elaborar arquitetura das CNNs foi definido funções que representará cada uma das camadas, sendo elas convolucional, pooling, e totalmente conectada, para facilitar o entendimento e o processo de elaborar a arquitetura.

Para a função da camada de convolução, passado o tamanho da imagem e a quantidade de filtros ela passará pela (layers.Conv2D), esta função cria uma camada de convolução 2D com os parâmetros especificados, incluindo o número de filtros a serem aplicados, o tamanho do kernel para convolução (tamanho da imagem e quantidade de camadas), realizando a convolução das entradas inputs com a quantidade de filtros definidos, e produz uma saída x, que será passada agora para um (layers.BatchNormalization) que é uma função de normalização em lote é aplicada à saída da camada de convolução, a normalização em lote normaliza os valores de ativação de cada camada, ajudando na estabilidade do treinamento e acelerando a convergência do modelo, resultando em uma saída x que por fim passará por (layers.Activation(”relu”)) que aplica uma função de ativação ReLU (Rectified Linear Unit) na saída da camada de normalização em lote resultando em uma saída x que será o input para as próximas camadas, a função de ativação ReLU é a mais utilizada na literatura, e foi a que apresentou um melhor desempenho nas métricas.

Para a função da camada de pooling, passado a variável da última função, ela passará primeiro por uma função (layers.MaxPooling2D), que cria uma camada de pooling máximo 2D, onde é aplicada uma operação de redução de dimensionalidade pela seleção dos valores máximos regiões retangulares de tamanho 2x2 na entrada. Esta camada reduz a dimensionalidade da entrada, preservando características relevantes e diminuindo a

quantidade de parâmetros na rede, gerando uma saída que será passada a próxima função, (layers.BatchNormalization). que é uma função de normalização em lote é aplicada à saída da camada de pooling, por fim, uma camada de dropout (layers.Dropout) é aplicada à saída da camada de normalização em lote, o dropout é uma técnica de regularização que desativa aleatoriamente um determinado percentual de unidades neuronais durante o treinamento, definido para 50% serem desativados, ajudando a prevenir o overfitting ao reduzir a dependência excessiva de determinados neurônios ou características, por fim e retornado a saída deste processo.

Para a função da camada totalmente conectada ela recebe os dados de entrada e primeiramente passa pela (layers.Dense(out)), que cria uma camada densa com um número passado pela entrada de neurônios de saída. Esta camada conecta todos os neurônios de entrada aos neurônios de saída, realizando operações de multiplicação de matrizes e aplicação de um viés para produzir a saída, em seguida, uma camada de normalização em lote (layers.BatchNormalization), é aplicada à saída da camada densa, após a normalização em lote, uma função de ativação ReLU (layers.Activation("relu")) é aplicada à saída da normalização em lote, por fim, uma camada de dropout (layers.Dropout) é aplicada à saída da função de ativação definida em 30% para serem desativados, por fim e retornado a saída deste processo.

4.3. Arquitetura

Na CNN para classificação de carros a arquitetura foi definida de acordo com a figura 5, composta pela entrada inicial com o tamanho padrão das imagens sendo 500x500x3 (500 de largura, 500 de comprimento e 3 camadas de cores), depois duas camadas seguidas de convolução com 64 filtros, uma camada de pooling que reduziu os dados pela metade, seguiu esse padrão mais três vezes até a última camada de polling que deixou os dados 31x31x3, com os dados nesse tamanho foi necessário apenas uma última camada de convolução, depois passado pelo flatten e na camada totalmente conectada (dense) com 128 neurônios e por fim a saída com o resultado sendo passado no softmax para a classificação.

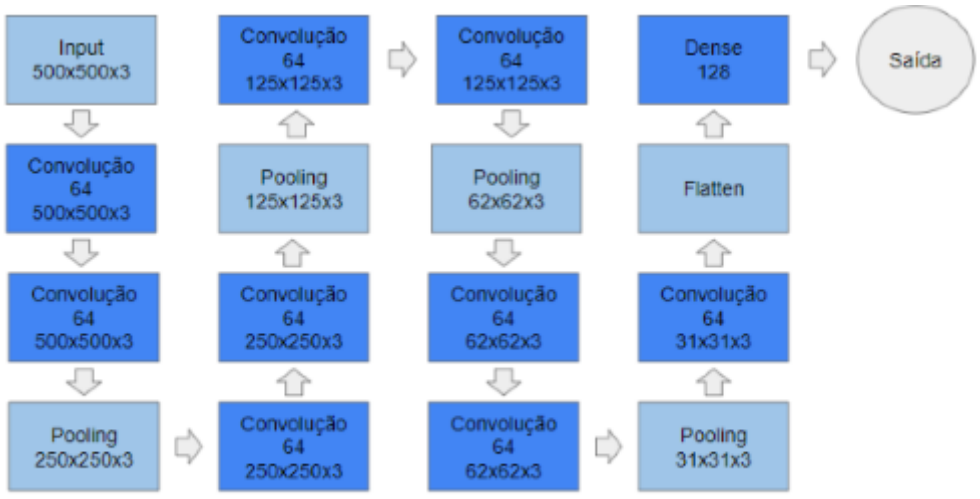


Figura 5. Arquitetura da CNN de carro

Foi usado duas camadas convolucionais consecutivas na maior parte da rede, pois isso permitiu que o modelo aprendesse características complexas e abstratas da entrada, exceto na última convolução que pelos dados já estarem muito reduzidos não foi necessário procurar por mais características, a quantidade de filtros foi definido através de testes, e ter mantido um padrão em todos foi o que melhor se desempenhou, além de que por o dataset não ter muitas imagens foi necessário fazer mais camadas de polling para abstrair a maior quantidade de características possíveis.

Na CNN para classificação de doenças nos rins a arquitetura foi definida de acordo com a figura 6, composta pela entrada inicial com o tamanho padrão das imagens sendo 250x250x1, por ela estar na escala de cinza possui apenas um canal de cor, depois duas camadas seguidas de convolução com 128 filtros, uma camada de pooling que reduziu os dados pela metade, seguiu esse padrão mais duas vezes porém com 64 filtros, até a última camada de polling que deixou os dados 31x31x1, com os dados nesse tamanho foi necessário apenas uma última camada de convolução, depois passado pelo flatten e na camada totalmente conectada (dense) com 96 neurônios e por fim a saída com o resultado sendo passado no softmax para a classificação.

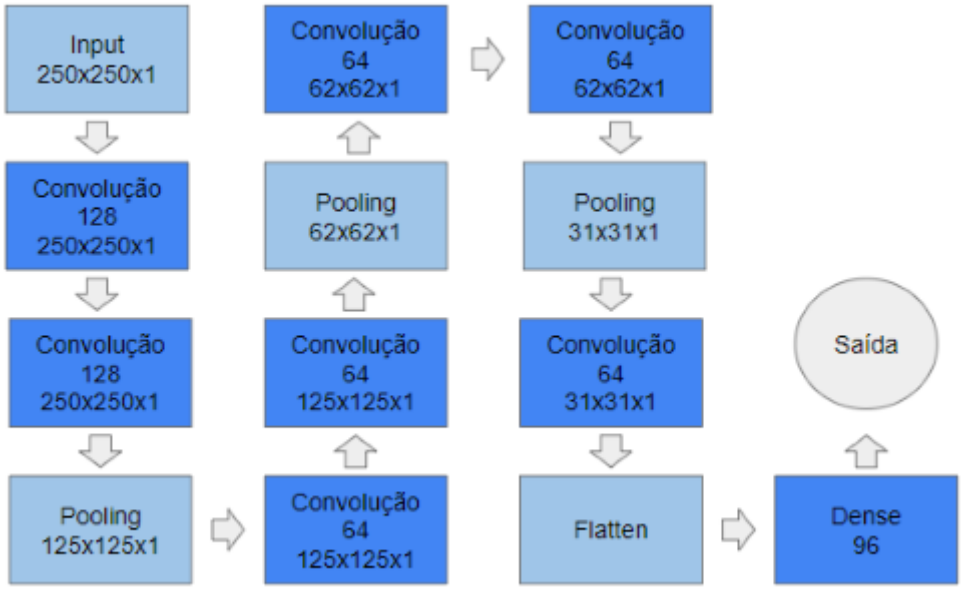


Figura 6. Arquitetura da CNN de doenças renais

As duas camadas convolucionais consecutivas foram usadas para também extrair dados mais complexos, e as duas primeiras convoluções tiveram mais filtros pois inicialmente as imagens apresentavam mais características, porém a quantidade de camadas de polling foi menor devido ao tamanho da imagem, não sendo preciso reduzir mais ainda, e a camada totalmente conectada teve 96 neurônios, já que devido a grande quantidade de imagens aumentar essa quantidade aumentava muito o tempo de treinamento.

4.4. Treinamento

O treinamento de uma CNN é o processo no qual os pesos das camadas da rede são ajustados iterativamente para minimizar a diferença entre as previsões passadas pela arquitetura

e os rótulos reais dos dados de treinamento, durante o treinamento, os dados são apresentados à rede em lotes, e o processo de ajuste dos pesos ocorre repetidamente por várias épocas.

com isso o treinamento da CNN de classificação de carros e da CNN de classificação de doenças renais foi usado 20 épocas e uma taxa de aprendizado de 1%, sendo a taxa o que determina o tamanho dos passos que o algoritmo de otimização dá ao ajustar os pesos da rede, nos dados de treino e validação, demorando 3 horas e 20 minutos e 4 horas e 12 minutos para terminar o treinamento respectivamente resultando em 95% de acurácia a classificação de carros e 94% de acurácia a classificação de doenças renais

5. Resultados e Discussões

Após o modelo treinado é usado a pasta de teste para verificar as métricas, utilizando uma biblioteca chamada sklearn é possível calcular todas as métricas carregando o modelo treinado e passando a pasta de teste.

A CNN de classificação de carros obteve uma precisão de 94,4% um F1-Score de 94% e a matriz de confusão na imagem 7.

	Audi a7	BMW series 7	Dodge charger	Porsche 911
Audi a7	246	0	1	3
BMW series 7	12	237	0	1
Dodge charger	17	4	227	2
Porsche 911	7	4	5	234

Figura 7. Matriz de confusão da CNN de doenças renais

já a CNN de classificação de doenças renais obteve uma precisão de 91,5% e um F1-Score de 91,3% e a matriz de confusão na imagem 8

	Cisto no rim	Rim normal	Pedra no rim	Tumor no rim
Cisto no rim	480	0	20	0
Rim normal	2	465	17	16
Pedra no rim	23	14	423	40
Tumor no rim	14	21	2	463

Figura 8. Matriz de confusão da CNN de carros

6. Considerações Finais e Trabalhos Futuros

Neste trabalho foi elaborado e testado duas CNNs de classificação multiclasse de temas bem diferentes e pode se concluir que com arquiteturas até um pouco semelhantes foram obtidos bons resultados, não muito dentro do esperado já que a segunda CNN de classificação de doenças renais obteve um resultado mediano justamente pelo fato de que não foi possível excluir todas a imagens de tórax que com certeza influenciou negativamente no resultado final

Como trabalhos futuros pode ser feita essa mesma análise e classificação de doenças renais abrangendo mais doenças e utilizando uma técnica de segmentação, selecionar apenas o rim de uma imagem de um tórax e ela poder se tornar útil em uma CNN de classificação.

Referências

Alzu'bi, D. et al. (2022). Kidney tumor detection and classification based on deep learning approaches: A new dataset in ct scans. *Journal of Healthcare Engineering*, 2022:1–22.

Araújo, F. H. et al. (2017). Redes neurais convolucionais com tensorflow: Teoria e prática. In *SOCIEDADE BRASILEIRA DE COMPUTAÇÃO. III Escola Regional de Informática do Piauí. Livro Anais-Artigos e Minicursos 1*, pages 382–406.

Dubey, S. R., Singh, S. K., and Chaudhuri, B. B. (2022). Activation functions in deep learning: A comprehensive survey and benchmark. *Neurocomputing*.

Filho, M. (2023). Precisão, recall e f1-score em machine learning. `https://mariofilho.com/precisao-recall-e-f1-score-em-machine-learning/`.

Gaba, S. and outros (2022). A federated calibration scheme for convolutional neural networks: Models, applications and challenges. *Computer Communications*, 192:144–162.

Liang, J. et al. (2018). Car detection and classification using cascade model. *IET Intelligent Transport Systems*, 12(10):1201–1209.

Pachón-Suescún, C. G., Pinzón-Arenas, J. O., and Jiménez-Moreno, R. (2019). Detection of scratches on cars by means of cnn and r-cnn. *Int. J. Adv. Sci. Eng. Inf. Technol.*, 9(3):745–752.

Poonia, R. C. et al. (2022). Intelligent diagnostic prediction and classification models for detection of kidney disease. *Healthcare*, 10(2).

Telles, E. S., Barone, D. A. C., and da Silva, A. M. (2020). Inteligência artificial no contexto da indústria 4.0. In *Anais do I Workshop sobre as Implicações da Computação na Sociedade*. SBC.

University, P. S. (2023). Car make, model, and generation. `https://www.kaggle.com/datasets/riotulab/car-make-model-and-generation`.

Wang, Z. J. et al. (2020). Cnn explainer: Learning convolutional neural networks with interactive visualization. *IEEE Transactions on Visualization and Computer Graphics*, 27(2):1396–1406.

Zaman, M. u. (2023). Multicancer7 cancer 23 classes combined. `https://www.kaggle.com/datasets/mahibuzzaman/multicancer7-cancer-23-classes-combined-512x512?select=Multicancer+all+classes+%287+cancer_+23+classes+combined%29+512x512`.