Trabalho de Computação Científica e Análise de Dados – Classificação de gatos e cachorros por meio de imagens

Jhayson de Brito Jales

Instituto de Computação
Universidade Federal de Rio de Janeiro (UFRJ) – Rio de Janeiro, RJ – Brazil

jhaysonbj@ic.ufrj.br

Resumo. *Este trabalho de Computação científica e Análise de Dados visa construir um modelo de machine learning para Classificação de gatos e cachorros por meio de imagens, nele usamos um dataset com imagens de gatos e cachorros para realizar o treinamento do nosso modelo.*

1. INTRODUÇÃO

Dentre as motivações para escolha da temática está o meu carinho e apreço pelos animais, na disciplina banco de dados tive a oportunidade de também fazer um trabalho que envolvia os animais, mais especificamente a fauna brasileira. Além disso, sempre tive curiosidade no tema de machine learning e considerei que poderia ser um trabalho desafiador e interessante para me introduzir à temática.

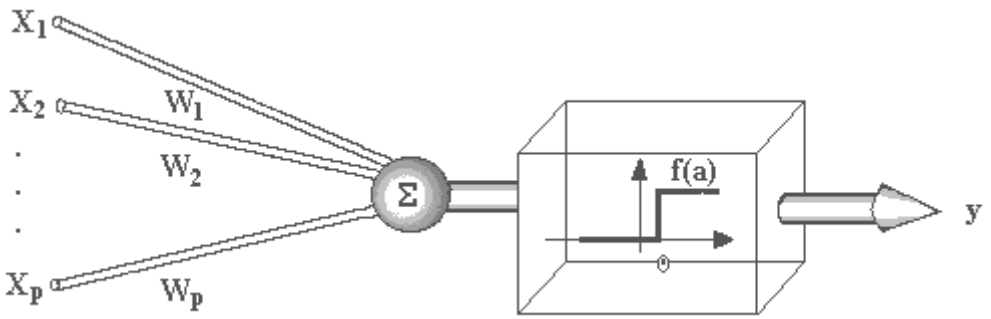
1.1 DEFINIÇÃO DE REDE NEURAL

De acordo com o site da Universidade de São Paulo (USP), podemos definir uma rede neural da seguinte forma:

“Uma rede neural artificial é composta por várias unidades de processamento, cujo funcionamento é bastante simples. Essas unidades, geralmente são conectadas por canais de comunicação que estão associados a determinado peso. As unidades fazem operações apenas sobre seus dados locais, que são entradas recebidas pelas suas conexões. O comportamento inteligente de uma Rede Neural Artificial vem das interações entre as unidades de processamento da rede.

A operação de uma unidade de processamento, proposta por McCullock e Pitts em 1943, pode ser resumida da seguinte maneira:

- Sinais são apresentados à entrada;
- Cada sinal é multiplicado por um número, ou peso, que indica a sua influência na saída da unidade;
- É feita a soma ponderada dos sinais que produz um nível de atividade;
- Se este nível de atividade exceder um certo limite (threshold) a unidade produz uma determinada resposta de saída.



” [5]

Informações extraidas de https://sites.icmc.usp.br/andre/research/neural/#hist

1.2 DEFINIÇÃO DE REDE NEURAL

O que? Especificação da entrada e saída

O modelo de rede neural proposto é desenvolvido para classificar imagens como sendo de gatos ou (exclusivo) cachorros. A rede neural recebe como entrada uma imagem e gera uma saída na forma de um número real (float), variando de 0 a 1, que indica a probabilidade da imagem retratar um gato ou um cachorro. Quanto mais próximo de 0, maior a probabilidade de ser um gato, e quanto mais próximo de 1, maior a probabilidade de ser um cachorro.

Ao utilizar o código fornecido, podemos treinar um modelo de rede neural capaz de classificar imagens de gatos e cachorros e utilizar esse modelo para fazer previsões em novas imagens. Isso nos permite automatizar o processo de classificação e obter resultados rápidos e precisos.

2. METODOLOGIA

Para resolver o problema de classificação de imagens de gatos e cachorros, utilizamos o código abaixo: https://colab.research.google.com/drive/1ezi66uP2KmR3PPoV1UMlhb6yInt8vmBm#scrollTo=3npLnI70AdIC

A implementação consiste em um conjunto de funções e etapas que nos permitem treinar um modelo de rede neural e utilizá-lo para fazer previsões sobre novas imagens. Para o treinamento do modelo neural foi utilizado um conjunto de dados com imagens de gatos e cachorros[3].

A primeira etapa é definir a arquitetura do modelo de rede neural. Nesse caso, utilizamos a arquitetura VGG16, que é uma rede neural convolucional pré-treinada, reconhecida por sua eficácia em tarefas de classificação de imagens, utilizado no reconhecimento de câncer de pele por meio de imagens[10]. Adicionamos camadas classificadoras no topo do modelo VGG16 para adaptá-lo à nossa tarefa específica.

Em seguida, definimos uma função chamada define_model() que carrega a arquitetura do modelo, marca as camadas como não treináveis, adiciona camadas classificadoras personalizadas e compila o modelo com um otimizador SGD[11], uma taxa de aprendizagem de 0.001 e uma função de perda binária cruzada (binary_crossentropy), essa entropia é útil para modelos iguais ao nosso, de classificação binária iguais[13] . Essa função retorna o modelo compilado.

O modelo otimizador Gradient Descent and Stochastic (SGD) é uma variante do Gradient Descent (GD), ambos são adequados para treinar um modelo de regressão linear, pois ajustam os parâmetros do modelo minimizando a função sobre os dados de treino. Os dois modelos garantem encontrar o mínimo global (solução óptima) se houver tempo suficiente e se a taxa de aprendizagem não for muito elevada. Duas variantes importantes do Gradiente Descendente que são amplamente utilizadas na

regressão linear e nas redes neuronais são o Gradiente Descendente em blocos (BGD) e o Gradiente Descendente Estocástico (SGD)12.

Na função run_test_harness(), executamos o treinamento do modelo. Primeiro, criamos um gerador de dados chamado ImageDataGenerator, que realizará pré-processamentos nas imagens durante o treinamento, como a centralização dos valores de pixel com base nos valores médios do conjunto de treinamento. Em seguida, preparamos o iterador para o conjunto de treinamento usando o gerador de dados.

Utilizamos o método fit_generator() para treinar o modelo. Esse método ajusta o modelo aos dados de treinamento, utilizando o iterador criado anteriormente. Especificamos o número de etapas por época e o número de épocas de treinamento. Neste caso, definimos 10 épocas.

Após o treinamento, salvamos o modelo em um arquivo H5 para uso posterior. Esse arquivo contém os pesos e a arquitetura do modelo treinado.

Para utilizar o modelo treinado para fazer previsões em novas imagens, podemos carregar o modelo salvo usando a função load_model(). Em seguida, podemos utilizar a função predict() do modelo para obter a probabilidade de cada imagem ser um gato ou um cachorro.

2.1 DIFERENÇAS ENTRE OS MODELOS OTIMIZADORES

Batch Gradient Descent	Stochastic Gradient Descent
Algoritmo caro e lento	Mais rápido e menos custoso que o BGD
Não recomendado para treinamentos em larga escala	Pode ser usado para treinamentos de larga escala
Determinístico.	Estocástico.
Dá a solução ótima com tempo suficiente para convergir.	Dá uma boa solução, mas não é a melhor.
Não é necessário embaralhar aleatoriamente os pontos	A amostra de dados deve estar numa ordem aleatória, e é por isso que queremos baralhar o conjunto de treino para cada época.
Convergência é devagar.	Alcance a convergência mais rapidamente.

A taxa de aprendizado do modelo é fixa.	A taxa de aprendizado do modelo pode ser ajustada dinamicamente.

Dados da tabela acima traduzidos de:
https://www.geeksforgeeks.org/difference-between-batch-gradient-descent-and-stochastic-gradient-descent/

Note que , no nosso código, utilizamos a função flow_from_directory do ImageDataGenerator para criar um gerador de dados de treinamento. Essa função, por padrão, já embaralha os dados antes de cada época de treinamento, garantindo que a ordem das amostras seja aleatória.[19]

Podemos definir modelos estocásticos como "Um processo estocástico é uma variável que se comporta, durante o tempo, de uma maneira onde pelo menos parte é considerada randômica."[18]

Para mais informações sobre SGD e BGD, acesse:
https://www.geeksforgeeks.org/difference-between-batch-gradient-descent-and-stochastic-gradient-descent/

3. RESULTADOS

Para o modelo utilizado, todo o conjunto de saída está no intervalo de 0 a 1, incluindo os extremos. Para resultados próximos de zero a classificação é definida como gato, para resultados próximos de 1 a classificação é definida como cão.

Durante as previsões, utilizamos os seguintes arquivos para os cachorros:

https://drive.google.com/drive/folders/1I6azOhx7nNjY_iZYA1NRGkIkdOPP190W?usp=drive_link

No link acima, possuímos oito imagens imagens, com os seguintes nomes:

- dog1.jpg
- dog2.jpg
- dog3.jpg
- dog4.jpg
- dog5.jpg
- dog6.jpg
- dog7.jpg
- dog8.jpg

Utilizamos o modelo para definir se as imagens em questão são cachorros ou gatos e tivemos o resultado abaixo:

Na previsão, definimos como “aproximadamente cão” toda imagem que o modelo não possuía resultado exatamente 1. Entretanto, para os testes realizados os resultados se concentravam na margem de 0.9 a 1 para cães.

De forma análoga, fizemos o mesmo para os gatos.

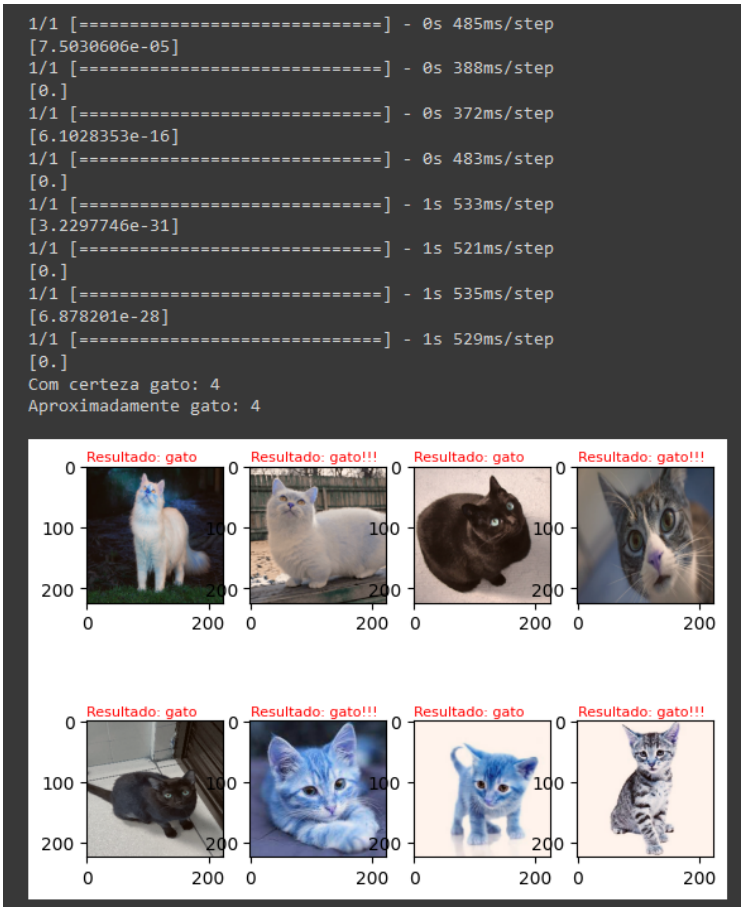
Durante as previsões, utilizamos os seguintes arquivos para os cachorros:

https://drive.google.com/drive/folders/1RkombacJRkEsNTH_VF2zObn_kFzuW5iz?usp=drive_link

No link acima, possuímos oito imagens imagens, com os seguintes nomes:

- cat1.jpg
- cat2.jpg
- cat3.jpg
- cat4.jpg
- cat5.jpg
- cat6.jpg
- cat7.jpg
- cat8.jpg

Utilizamos o modelo para definir se as imagens em questão são cachorros ou gatos e tivemos o resultado abaixo:



Na previsão, definimos como “aproximadamente gato” toda imagem que o modelo não possuía resultado exatamente 1. Entretanto, para os testes realizados os resultados se concentravam na margem de 0 a 0.1 para gatos.

Vale destacar que, para a amostragem de 16 imagens, podemos ver que foi mais “fácil” designar as imagens que são gatos, onde todos os valores são zero na prática. Além disso, o mesmo modelo definiu o “dog7.jpg” com valor 0.7124911.

Com isso, apesar da amostragem pequena para podermos afirmar, intuitivamente, podemos considerar que é mais difícil definir a classe gato do que a classe cão.

Para afirmarmos que o modelo tem maior facilidade na classificação de gatos deveríamos expandir a nossa amostragem e analisarmos o erro na predição de gatos e depois analisarmos o erro na predição de cães.

AMOSTRAGEM DE 1007 IMAGENS

Considerando o parágrafo anterior, elaboramos duas células de código, a célula 1

recebe como parâmetro 1007 imagens de gatos, já a célula 2 recebe como parâmetro 1007 imagens de cachorros.
Selecionadas arbitrariamente as imagens de outro dataset que não o usado para treinar o modelo https://www.kaggle.com/c/dogs-vs-cats

Para a célula 1, temos as seguintes condições:

Reservamos uma pasta com imagens de apenas gatos e contamos os resultados calculados calculados pelo nosso modelo.

```
Gatos: 1006
Inconclusivo: 0
Cão 1
Lista das imagens previstas erroneamente
Cão [661]
Inconclusivo []
```

A partir do resultado, plotamos a imagem “cat.661.jpg” que foi classificada como cão (erroneamente) pelo nosso modelo.

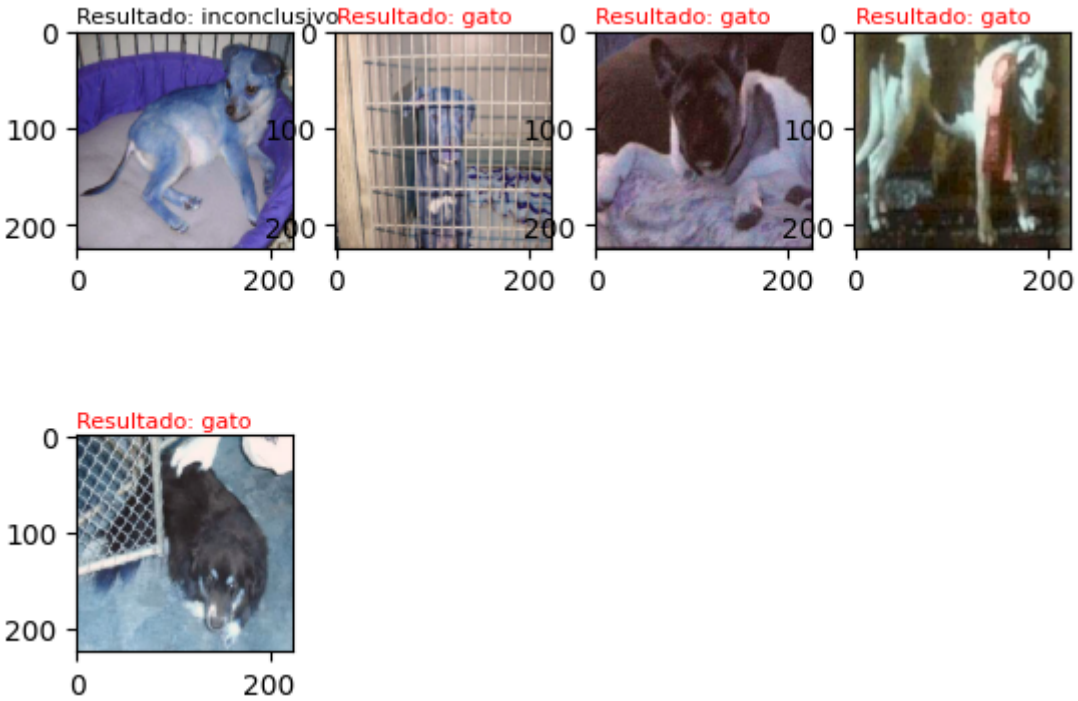
```
1/1 [==============================] - 1s 666ms/step
[0.90105784]
```

Para a célula 2, temos as seguintes condições:

Reservamos uma pasta com imagens de apenas cães e contamos os resultados calculados pelo nosso modelo.

```
Gatos: 4
Inconclusivo: 1
Cão 1002
Lista das imagens previstas erroneamente
Gatos [595, 919, 976, 1007]
Inconclusivo [356]
```

A partir do resultado, plotamos as imagens “dog.356.jpg”, “dog.595.jpg”,“dog.919.jpg”,“dog.979.jpg” e “dog.1007.jpg” que foi classificadas como gato/inconclusivo (erroneamente) pelo nosso modelo.



A partir dos resultados obtidos da amostragem de 1007 (para cada classe), é possível concluir que existe maior facilidade para classificação de gatos pelo nosso modelo que obteve 9.930486593843098e-4 de erro na classificação de gatos, enquanto o mesmo modelo obteve o erro 0.0049652432969215. Para fins de comparação, o ideal era aumentarmos ainda mais a amostragem, mas a velocidade de execução ficaria muito prejudicada, cada célula levou em média 20 minutos para ser executada, por isso utilizamos apenas 1007 imagens.

4. CONCLUSÃO

Neste trabalho, desenvolvemos um código para treinar e avaliar um modelo de rede neural que tem como objetivo classificar imagens como sendo de gatos ou cachorros.

Apesar da temática descontraída, o modelo pode ser generalizado para temas mais relevantes, como identificação do câncer de pele por meio de imagens[10], agilizando o processo de identificação de potenciais doentes, corroborando para efetividade dos tratamentos.

Entretanto, ao abordar esse problema, é importante considerar as implicações éticas e a discussão em torno da inteligência artificial (IA), um fator agravante é o uso de modelos de aprendizado de máquina para superar sistemas de segurança, como os CAPTCHAs[14].

Em conclusão, o código implementado permitiu treinar um modelo de rede neural capaz de classificar imagens como gatos ou cachorros com base em sua probabilidade de pertencer às classes “gato” e "cachorro". Na implementação, definimos arbitrariamente um valor resultante para considerarmos que a imagem possui um cão, utilizamos o

intervalo de 0.7 a 1, incluindo os extremos. De forma análoga, definimos um valor resultante para considerarmos que a imagem possui um gato, utilizamos o intervalo de 0 a 0.3, incluindo os extremos. Para o intervalo entre 0.3 e 0.7, excluindo os extremos, definimos que o resultado é inconclusivo mas para a amostragem de 16 imagens, nenhum resultado foi definido como inconclusivo.

O nosso modelo pode ser útil em diversas aplicações que envolvam a classificação de imagens de animais de estimação. É importante ressaltar que, para obter um desempenho ainda melhor, seria recomendável explorar técnicas de ajuste de hiperparâmetros e aumento de dados, além de realizar uma avaliação mais abrangente utilizando conjuntos de validação e teste[16].

5. REFERÊNCIAS

1. Estudo inicial do problema
   https://machinelearningmastery.com/how-to-develop-a-convolutional-neural-network-to-classify-photos-of-dogs-and-cats/
2. Estudo sobre as bibliotecas utilizadas na referência1
   https://keras.io/api/data_loading/image/
3. Dataset com imagens de cães e gatos
   https://www.kaggle.com/c/dogs-vs-cats/data
4. Estudo sobre redes neurais

   https://www.jeremyjordan.me/nn-learning-rate/

5. Definição de rede neural

   https://sites.icmc.usp.br/andre/research/neural/#hist

6. Colab onde testei outros modelos de menor precisão
   https://colab.research.google.com/drive/1E1NDudafb_xBgt9sebn2cpxUwKpLgtUS

7. Artigo sobre a publicação do VGG em classificação de imagem
https://arxiv.org/pdf/1409.1556.pdf
8. Guia de como usar o VGG
https://towardsdatascience.com/vgg-neural-networks-the-next-step-after-alexnet-3f91fa9ffe2c
9. Colab do projeto, adaptado da referência (1)
https://colab.research.google.com/drive/1ezi66uP2KmR3PPoV1UMlhb6yInt8vmBm#scrollTo=3npLnI70AdIC
10. Utilidade do VGG16 para identificação de câncer de pele
https://ieeexplore.ieee.org/document/9783556
11. Stochastic Gradient Descent (SGD)
https://scikit-learn.org/stable/modules/sgd.html
12. Diferença entre SGD e BGD
https://www.geeksforgeeks.org/difference-between-batch-gradient-descent-and-stochastic-gradient-descent/
13. Classificação binária (binary_crossentropy)
https://keras.io/api/losses/probabilistic_losses/#binary_crossentropy-function
14. Ataque ao captcha com uso machine learning
https://ietresearch.onlinelibrary.wiley.com/doi/10.1049/ise2.12047
15. Link para o projeto projeto
16. https://drive.google.com/drive/folders/19PCstO5iPgRcH-x97f_OE_zvt0eb3Bqw
17. Otimização do modelo de machine learning por meio de hiperparâmetros
https://dev.to/mage_ai/10-steps-to-build-and-optimize-a-ml-model-4a3h
18. Definição de modelos estocásticos
https://www.maxwell.vrac.puc-rio.br/10083/10083_5.PDF
19. Documentação do tensorflow
https://www.tensorflow.org/api_docs/python/tf/keras/preprocessing/image/ImageDataGenerator
20. Link para o dataset usado na amostragem de 1007 imagens
https://www.kaggle.com/c/dogs-vs-cats