Projeto 3-1: Buscador de imagens Parte 2

Bruna Mayumi Kimura
Visão computacional

1. Introdução

Este projeto consiste em fazer um programa capaz de buscar imagens corresponde a uma determinada palavra. Ou seja, ao fazer uma busca por um certo objeto o programa irá retornar as 5 imagens com melhor resultado para aquela pesquisa. As imagens retornadas estão em um banco de dados previamente montado.

2. Modo de uso

Para testar o programa é necessário clonar este repositório. Em seguida fazer os dois passos abaixo:

1. Geração do banco de dados

A primeira etapa do projeto consiste em gerar o arquivo que possui os dados de todas as imagens do banco de dados classificada. Para tanto, é necessário rodar o arquivo ‘gera_dict.py’. Este programa irá gerar um outro arquivo chamado ‘dict.p’ que possui os dicionários relacionando as classes com as imagens do banco de dados.

Este arquivo ‘dict.p’ já foi previamente gerado utilizando as imagens da pasta ‘imagens’. Este processo é um pouco lento uma vez que ele itera sobre cada uma das imagens do banco de dados.

Como este arquivos já está contidos na pasta, pois foi anteriormente gerado, ele está pronto para uso, sem a necessidade de gerar novamente. Porém, caso queira refazer o treino, o arquivo será reescrito.

2. Busca das imagens semelhantes

Com o arquivo ‘dict.p’ devidamente treinado próximo passo é encontrar as imagens desejadas. Dessa forma, é necessário executar o arquivo ‘busca_imagens.py’. A execução deste programa é um pouco diferente, já que é necessário colocar um valor como argumento antes da execução como mostra a

figura 1. Neste exemplo foi passado o argumento ‘panda’, ou seja as imagens que serão buscadas são aquelas relacionadas à palavra ‘panda’.

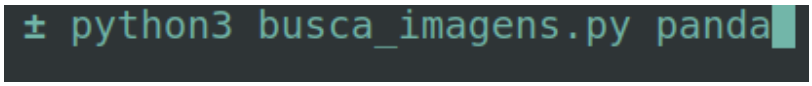

figura 1: como passar o argumento do busca_imagens.py

Há duas maneiras de se passar o argumento. A primeira é usando o nome da categoria inteiro. Ou seja, para pesquisar imagens com pandas é necessário saber o nome da categoria em que ele está inserido. Dessa forma é necessário passar como argumento 'giant panda, panda, panda bear, coon bear, Ailuropoda melanoleuca'. Caso você passe um termo que não corresponde a uma classe o seguinte log (figura 2) irá aparecer no console.

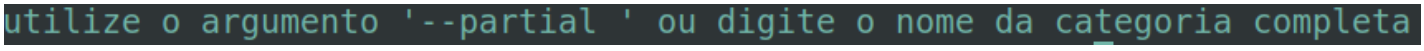

figura 2 - log de erro ao passar um termo que não é uma categoria

Como o próprio *log* acima diz, o outro modo de passar o argumento é colocando uma *flag* ‘--partial’ antes ou depois do termo a ser procurado. Neste caso o termo não precisa ser necessariamente o nome da categoria, mas parte dela. A forma de utilizar esta demonstrada na figura 3. O resultado desta busca é figura 5.

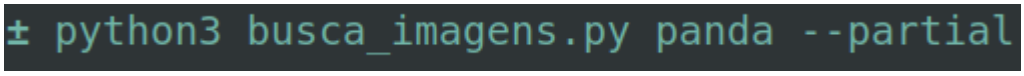

figura 3 - forma correta de passar apenas um termo para busca

Ao executar este comando o programa pode retornar duas respostas. A primeiro é caso não haja nenhuma imagem relacionada ao termo pesquisado. Neste caso o programa irá retornar uma mensagem de log no terminal dizendo que o termo não foi encontrado, como mostra a figura 4. Caso haja algum resultado sobre o termo pesquisado, irá abrir uma janela com no máximo 5 imagens mais próximas do termo escolhido. O título da janela será a categoria que o termo pesquisado esta inserido, já o título de cada imagem é a porcentagem de confiança de que aquela imagem realmente corresponde à categoria. A figura 5 representa esta situação.

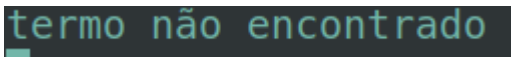

figura 4 - log do termo não encontrado no banco de dados

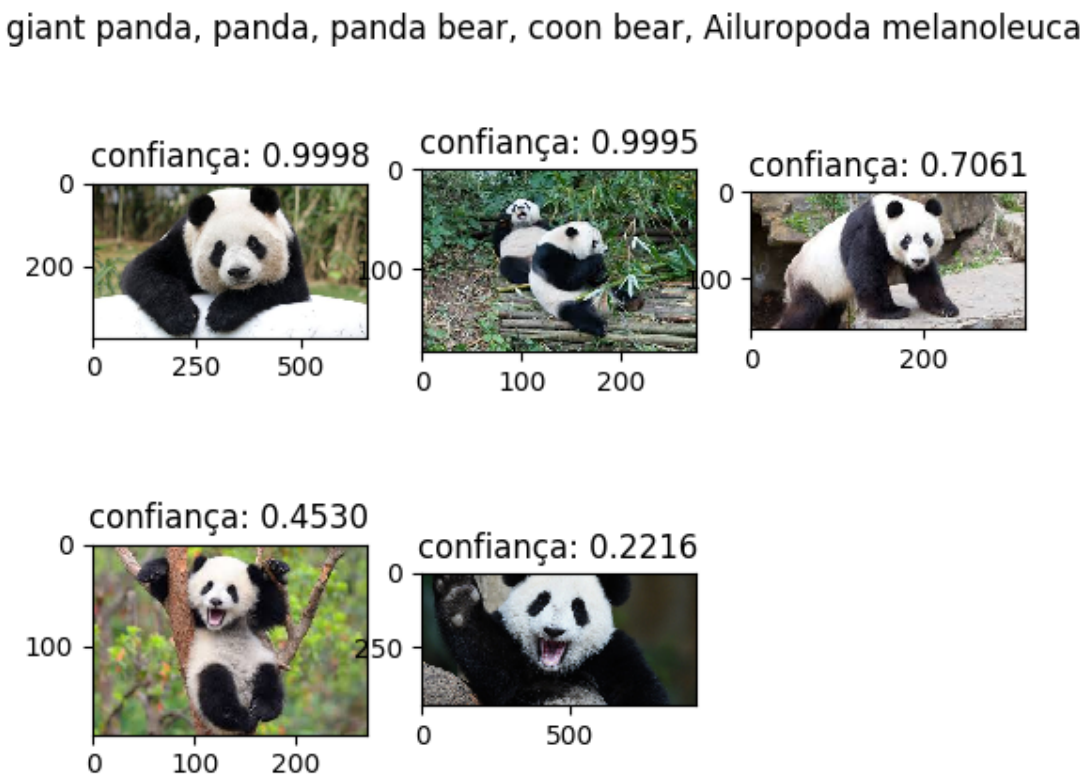

figura 5 - busca pelo termo ‘panda’. Resultado da pesquisa com as imagens, grau de confiança e título da classe a qual ela corresponde.

3. Código e técnicas utilizadas

1. Dicionario das classes (dicionario.py)

O arquivo ‘dicionário.py’ possui apenas um dicionário relacionando o id’s das classes com o seu nome. Ou seja, as chaves (keys) do dicionário são os identificadores numéricos (id’s) e o valor (values) é uma string com todas as possíveis palavras que estão relacionadas àquele id.

Este dicionário foi retirado no seguinte github: yrevar/imagenet1000_clsid_to_human.txt (Acesso pelo link: https://gist.github.com/yrevar/942d3a0ac09ec9e5eb3a) .

2. Banco de dados (gera_dict.py)

O arquivo ‘gera_dict.py’ é responsável por gerar um arquivo ‘dict.p’ que possuirá uma relação de id’s (das classes do arquivo descrito acima) com suas respectivas imagens do banco de dados. Assim, todas as imagens do banco de dados são classificadas para corresponder a um ou mais id’s identificados no arquivo descrito acima (dicionario.py).

Para classificar as imagens foi utilizado uma biblioteca do keras previamente treinada para 1000 categorias distintas, as mesmas que estão no arquivo dionario.py acima. Foram feitas duas funções, a primeira recebe o caminho de imagens e manda para a segunda que é responsável por aplicar o método ‘process_input()’ do keras. O retorno desta segunda função é utilizada na segunda para fazer os cálculos de confiança de acerto de cada imagem e dessa forma a primeira função retorna as 5 imagens com maior confiança de cada categoria e seu respectivo valor.

Por fim, o arquivo ‘dict.p’ é gerado sendo um dicionário que relaciona o id com as imagens e seus valores de confiança.

3. Busca da imagem (busca_imagens.py)

O arquivo ‘busca_imagens.py’ é responsável por receber um termo passado no momento da execução e encontra as imagens correspondentes dado o dicionado gerado acima (‘dict.p’).

A palavra passada pode encaminhar para duas funções distintas dependendo da utilização de uma *flag* como explicado no item 2.

Caso não seja passado nenhuma *flag* o programa irá comparar a palavra buscada com os nomes das categorias e só irá prosseguir caso a combinação seja exata, ou seja, a palavra passada é exatamente o nome da categoria por extenso.

Caso seja utilizado a *flag* ‘--pratical’ junto ao termo o programa irá verificar se esta palavra a ser buscada faz parte do nome das possíveis categoria do ‘dicionario.py’, sem ser exatamente a categoria por extenso. Todas as categorias que satisfazem ao termo são buscados no banco de dados gerado acima, o ‘dict.p’, caso ocorra correspondência entre a categoria do termo com as existentes no dicionário gerado, as imagens guardadas no dicionário dict serão mostradas. Caso não ocorra o termo no dicionário dict, aparecerá um log dizendo que o termo não foi encontrado como mostra a figura 2.

5. Conclusões e melhorias

De modo geral o programa retorna as imagens corretamente e o resultado é bastante satisfatório. O banco de dados é ainda muito pequeno e portanto acabam várias vezes aparecendo que o termo não foi encontrado, justamente por falta de materiais. Uma possível melhoria seria expandir o banco de dados.

Outro problema que ocorre é que algumas categorias possuem apenas uma imagem e com uma porcentagem de confiança bastante baixa. Para resolver isso, poderíamos

adicionar mais itens no banco de dados como sugerido acima Dessa forma, seria interessante colocar imagens dessas categorias que possuem poucas imagens e com pouca confiança, fazendo com que essas imagens “corretas” acabam se sobrepondo às atuais. Mas, talvez, uma forma mais eficiente a curto tempo e sem alterar o banco de dados seria tentar retirar as imagens com porcentagem menor que um determinado valor do dicionário gerado. Assim, essas imagens com pouca confiança nem seriam consideradas. Poupando os resultados ruins.

Uma outra melhoria seria otimizar a forma como as imagens são mostradas no caso da busca pelo termo. Neste caso se o termo buscado estiver em várias categorias diferentes e em todas essas tiverem amostra de imagens, irá abrir várias janelas e deve-se fechá-las uma a uma para visualização. Neste caso, as imagens poderiam ser mostradas em uma interface diferenciada para melhorar a visualização.