Lista de exercícios 1 - Visão Computacional

1. Considere uma imagem de 1 canal possuindo valores do tipo inteiro sem sinal (unsigned integer) de 8 bits (uint8). Um pixel dessa imagem pode possuir quantos valores? Se a imagem possuir 3 canais (RGB), quantos valores distintos um pixel pode ter?

2. Um conjunto de dados possui os seguintes valores para as variáveis x e y:

$x = [1, 2, 3, 4]$
$y = [2, 7, 6, 12]$

O modelo $f(x) = 2x + 1$ foi criado para representar esses dados. Calcule o erro quadrático médio do modelo.

3. Considere o seguinte modelo linear possuindo três variáveis de entrada:

$$f(x_1, x_2, x_3) = w_1 x_1 + w_2 x_2 + w_3 x_3 + b$$

Calcule o gradiente do modelo em relação aos seus parâmetros, isto é, a derivada da saída em relação aos valores w_1, w_2 e w_3.

4. O passo de atualização do método de gradiente descendente é dado por

$$w = w - lr\nabla_w L$$

onde w é um parâmetro do modelo, lr é a taxa de aprendizado e $\nabla_w L$ o gradiente da função de perda L (erro quadrático, entropia cruzada, etc). Explique em quais situações será mais adequado utilizar a equação

$$w = w + lr\nabla_w L$$

no método de gradiente descendente.

5. Explique em termos gerais o que é um *grafo de computação* para diferenciação automática e como ele pode ser utilizado em problemas de otimização.

6. Considere o seguinte código:

```
x1 = torch.tensor(5.)
x2 = torch.tensor(3.)
y = operacao(x1, x2)
soma = y[0] + y[1]
```

A função **operacao** realiza uma operação entre os tensores **x1** e **x2**. Você não possui acesso a essa função. Altere o código para calcular e imprimir o gradiente da variável **soma** em relação às variáveis **x1** e **x2** após a aplicação da função **operacao**.

7. Suponha que um modelo foi desenvolvido para classificar um conjunto de dados em 2 classes distintas. Para um conjunto de 5 itens o modelo associou os seguintes valores de probabilidades para cada item:

	Classe 1	**Classe 2**
Item 1	0.8	0.2
Item 2	0.9	0.1
Item 3	0.5	0.5
Item 4	0.3	0.7
Item 5	0.0	1.0

É conhecido que os itens 1, 2 e 3 devem ser classificados na classe 1, enquanto que os itens 4 e 5 devem ser classificados na classe 2. Qual o valor da entropia cruzada para esse conjunto de resultados? (veja o notebook sobre regressão logística)

8. Explique por que uma função de ativação não linear deve sempre ser inserida entre duas camadas lineares.

9. A camada linear de um modelo é definida por uma matriz possuindo 6 linhas e 15 colunas. Essa camada é aplicada em cada item $\boldsymbol{x_i}$ de um dataset. Quantos atributos cada item $\boldsymbol{x_i}$ possui? A camada irá gerar quantos novos atributos a partir de cada $\boldsymbol{x_i}$?

10. Um tensor de entrada de uma função do Pytorch possui tamanho [16, 3, 256, 256]. Descreva o significado de cada dimensão desse tensor.

11. Considere o seguinte código Pytorch:

```
x = torch.rand(16, 784)
linear = nn.Linear(784, 100)
y = linear(x)
```

Qual será o tamanho das dimensões (linhas e colunas) do atributo `.weight` da variável `linear`? Qual será o tamanho da variável y?

12. Calcule o resultado da convolução entre o sinal e filtros abaixo. Considere que o sinal é preenchido com zeros de forma que a saída tenha o mesmo tamanho que a entrada.

Sinal: [4, 0, 1, 2, 3, 1, 0]
Filtro: [1, 2, 3]

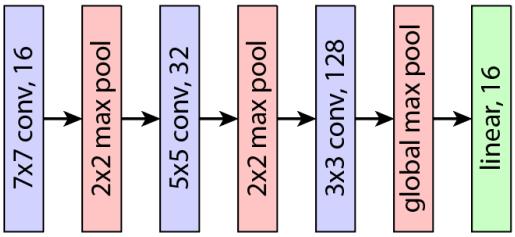
13. Descreva 2 vantagens de uma camada de convolução em relação a uma camada linear quando trabalhamos com imagens.

14. O atributo .weight de uma camada de convolução 2D do Pytorch possui tamanho [64, 16, 5, 5]. Qual deve ser o número de canais do tensor de entrada desta camada? Qual será o número de canais do tensor de saída? Qual o tamanho dos filtros da camada?

15. Como uma camada de convolução pode ser utilizada para transformar um tensor de tamanho 16x8x256x256 em um tensor de tamanho 16x64x128x128?

16. Explique a diferença entre uma função que possui invariância espacial e uma função que possui equivariância espacial.

17. Considere a rede neural abaixo. Liste os tamanhos dos tensores de saída de cada camada quando a imagem de entrada for um tensor de tamanho 3x224x224. Considere que as convoluções não alteram o tamanho espacial das ativações. Desconsidere a dimensão associada ao *batch*.



18. Descreva quatro abordagens que podem ser utilizadas para regularizar uma rede neural.

19. Descreva, utilizando exemplos, porque é importante que os dados de entrada de uma rede sejam normalizados, e que sejam adicionadas camadas de normalização como *batch normalization*.

20. Explique por que a rede neural abaixo possui 322 parâmetros. *Lembre-se que uma camada convolucional possui um valor de bias para cada filtro, e uma camada linear possui um bias para cada valor de saída.

```
class Model(nn.Module):

    def __init__(self):
        super().__init__()

        self.layers = nn.Sequential(
            nn.Conv2d(3, 3, kernel_size=3),
            nn.ReLU(),
            nn.Conv2d(3, 6, kernel_size=3),
            nn.ReLU(),
            nn.Conv2d(6, 1, kernel_size=3),
            nn.AdaptiveAvgPool2d(2),
            nn.Linear(4, 3)
        )
```

21. O que é um “atalho” em uma rede neural? Por que é comum dizer que um atalho facilita a propagação dos gradientes?

22. Porque dizemos que uma rede neural realiza um “aprendizado de representação”? Onde fica localizada tal representação?

23. Considere uma rede neural sendo treinada em uma GPU. Descreva uma situação na qual a GPU ficará ociosa durante o treinamento.

24. Uma rede neural foi treinada para classificar um conjunto de dados em 20 classes distintas. Explique o procedimento que pode ser utilizado para refinar essa rede em um novo conjunto de dados possuindo 4 classes.