MULTILAYER PERCEPTRON

LISTA DE EXERCÍCIOS

Questões de Múltipla Escolha.

I) Por que o *Perceptron* como proposto originalmente ($\hat{y}^{(i)} = \text{sign}(\mathbf{X}^{(i)}w)$) não pode ser organizado em uma rede para formar um *Multi Layer Perceptron*?

a) Pois a derivada da função de ativação em relação aos pesos ou é igual a zero ou indefinida.

b) Pois o *dot product* dos pesos com as *features* não pode ser derivado.

c) Pois existe uma descontinuidade na função de ativação $\text{sign}(x)$ quando $x = 0$ e isso impede o processo de otimização via descida de gradiente.

d) Pois o perceptron possui a limitação de modelar apenas problemas que são linearmente separáveis.

II) Em uma unidade (neurônio) de uma rede neural artificial, qual a operação realizada entre as entradas e os pesos durante o *forward pass*?

(a) Produto Escalar (*Dot Product*).

(b) Função de Ativação (*Activation Function*).

(c) Retropropagação de Erros (*Backpropagation*).

(d) Descida de Gradiente (*Gradient Descent*).

III) Em uma rede perceptron multicamadas (*Multi Layer Perceptron*). Qual das alternativas abaixo melhor descreve o *Forward Pass* e o *Backward Pass* respectivamente?

(a) Computar as saídas de cada uma das unidades e as entradas.

(b) Computar as saídas de cada uma das unidades e o gradiente em relação aos pesos.

(c) Computar as entradas de cada uma das unidades e as saídas.

(d) Realizar a Retropropagação de Erros (*Backpropagation*) e a Descida de Gradiente (*Gradient Descent*).

(e) Realizar a Descida de Gradiente (*Gradient Descent*) e a Retropropagação de Erros (*Backpropagation*).

IV) Dentre as funções de ativação abaixo, qual é a opção mais adequada para modelar um problema de classificação **multiclass** no qual você quer interpretar a saída da rede como as probabilidades da observação pertencer a uma dada classe?

(a) ReLU: $f(x_i) = max(0, x_i)$

(b) Sigmoid: $f(x_i) = \frac{1}{1+e^{-x_i}}$

(c) Tanh: $f(x_i) = \frac{2}{1+e^{-2x_i}} - 1$

(d) Softplus: $f(x_i) = \log_e(1 + e^{x_i})$

(e) Softmax $f(x_i) = \frac{e^{x_i}}{\sum_z e^{x_z}}$

(f) Linear $f(x_i) = x_i$

V) Dentre as funções de ativação abaixo, qual é a opção mais adequada para modelar um problema de classificação **multilabel** no qual você quer interpretar a saída da rede como as probabilidades da observação pertencer a uma dada classe?

(a) ReLU: $f(x_i) = max(0, x_i)$

(b) Sigmoid: $f(x_i) = \frac{1}{1+e^{-x_i}}$

(c) Tanh: $f(x_i) = \frac{2}{1+e^{-2x_i}} - 1$

(d) Softplus: $f(x_i) = \log_e(1 + e^{x_i})$

(e) Softmax $f(x_i) = \frac{e^{x_i}}{\sum_z e^{x_z}}$

(f) Linear $f(x_i) = x_i$

VI) "Para um problema de classificação **multilabel** onde a saída pode ter 10 classes diferentes (simultaneamente ou não) a função de ativação na camada de saída do MLP deve ser _____ e o treinamento deve buscar otimizar a função de perda _____". Quais opções abaixo melhor preenchem as lacunas da afirmação acima?

(a) ReLU: $f(x_i) = max(0, x_i)$

(b) Sigmoid: $f(x_i) = \frac{1}{1+e^{-x_i}}$

(c) Tanh: $f(x_i) = \frac{2}{1+e^{-2x_i}} - 1$

(d) Softplus: $f(x_i) = \log_e(1 + e^{x_i})$

(e) Softmax $f(x_i) = \frac{e^{x_i}}{\sum_z e^{x_z}}$

(f) Linear $f(x_i) = x_i$

(g) *Binary Cross Entropy Loss*: $L_{bce} = -(y\log(\hat{y}) + (1-y)\log(1-\hat{y}))$.

(h) *Categorical Cross Entropy Loss*: $L_{cce} = -\sum_{i=1}^{c} y_i \log \hat{y}_i$

(i) *Sum of Squared Residuals*: $L_{ssr} = (y - \hat{y})^2$.

(j) *Absolute Error Loss*: $L_{abs} = |y - \hat{y}|$.

(k) *Zero-One Loss*: $L_{01} = I(y \neq \hat{y})$.

VII) Durante o treinamento de uma rede neural você percebe que os gradientes diminuem rapidamente até que se aproximam de zero, fazendo que os pesos da rede permaneçam praticamente sem mudança. Você rapidamente identifica que se trata de um caso de *vanishing gradient*, marque dentre as opções abaixo possíveis causas para esse problema.

(a) As cadamadas ocultas usam função de ativação *Tanh*.

(b) As cadamadas ocultas usam função de ativação *ReLU*.

(c) As cadamadas ocultas usam função de ativação *Sigmoid*.

(d) A rede usa *batch normalization*.

VIII) O valor da função *loss* da sua rede neural está oscilando ao invés de decrescer constantemente conforme esperado durante o treinamento. Supondo que você esteja utilizando *Mini-Batch Gradient Descent*, quais das opções abaixo são boas modificações que você pode introduzir para tentar resolver esse problema?

(a) Aparentemente isso é um problema dos dados, obter uma quantidade maior de dados para efetuar o treinamento é uma alternativa razoável.

(b) Aumentar o valor hiperparâmetro *learning rate* pode ajudar pois o processo de otimização vai chegar a um mínimo mais rápido.

(c) Diminuir o valor hiperparâmetro *learning rate* pode ajudar pois o processo de otimização para garantir que um mínimo local seja encontrado e não "pulado".

(d) Aumentar o hiperparâmetro *batch size* pode ajudar pois vai introduzir maior variabilidade em cada *mini-batch*. Assim as chances de cada um dos passos da otimização gerar uma diminuição no loss aumenta.

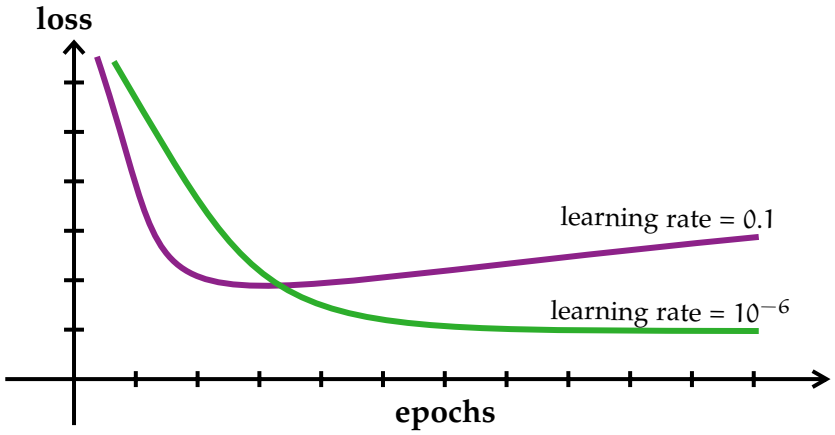
IX) François Chollet, o criador do *Keras*, comentou em 2022 que você não deve tentar aproveitar um *scheduler* de *learning rate* que funcionou bem em outro conjunto de dados para treinar em um novo conjunto de dados. A afirmação de Chollet vale mesmo nos casos nos quais a arquitetura da rede se mantém igual. Dentre as opções abaixo, quais são racionais razoáveis para essa afirmação?

(a) *Learning rate schedulle* deve variar conforme o número de observações no conjunto de dados. Dessa forma se os dados mudarem precisamos mudar o *scheduller*.

(b) O *learning rate schedulle* deve mudar sempre que a distribuição alvo mudar. Nos casos que ela se mantém igual não é necessário realizar modificação.

(c) Um novo conjunto de dados modifica o *tradeoff* entre otimização e regularização. Dessa forma, o *learning rate scheduller* é específico para um conjunto de dados.

X) Você está experimentando com diferentes *learning rates* ao treinar uma rede neural específica. A evolução do valor da função *loss* no decorrer das épocas usando *learning rates* 0.1 e 10^{-6} estão estão retratados na figura abaixo. Com base nesses resultados, qual seria um bom valor de *learning rate* para testar a seguir?



(a) 0.2

(b) 10^{-4}

(c) 10^{-8}

XI) *Label Smoothing* é uma técnica de regularização que visa diminuir a tendência de *overfitting* do modelo combatendo principalmente inferências com *overconfidence*. Essa técnica redefine os valores alvo da seguinte forma:

$$\mathbf{y}_{ls}^{(i)} = (1-\alpha)\mathbf{y}_{oh}^{(i)} + \frac{\alpha}{C}$$

onde $y_{oh}^{(i)}$ é a variável alva codificada em *one-hot encoding*, α é a constante de *smothing* e C é o número de classes do problema. Sobre *Label Smoothing*, marque as alternativas verdadeiras:

(a) Quando $\alpha = 1$ a variável alvo fica distribuída uniformemente entre as classes.

(b) Quando $\alpha = 0$ a variável alvo permanece inalterada em formato *one-hot encoding*.

(c) Para encontrar o valor ideal de $\alpha = 1$ devemos considerar a quantidade de observações no conjunto de dados.

(d) Quando $\alpha = 0$ temos a maior intensidade de *smoothing*.

XII) A função de ativação *Rectified Linear Unit (ReLU)* é uma das funções de ativação mais utilizadas em redes neurais. A *ReLU* é definida como $f(x) = \max(0, x)$. Sobre essa função de ativação, é correto afirmar que:

(a) Minimiza o problema de *vanishing gradient* em comparação com a função de ativação sigmoid.

(b) É continua e diferenciável em todo seu domínio.

(c) Não é contínua e não é diferenciável em todo seu domínio.

(d) É contínua e não é diferenciável em todo seu domínio.

XIII) Quando usamos a seguinte expressão $w_{(t+1)} = w_{(t)} - \eta\nabla\mathcal{L}$ para a atualização dos pesos de uma rede neural do tipo *multilayer perceptron*, considerando que o gradiente está sendo estimado com base em uma única observação, temos qual variação do algoritmo de descida de gradiente?

(a) *Batch Gradient Descent*.

(b) *Mini Batch Gradient Descent*.

(c) *Stochastic Gradient Descent*.

(d) *Deterministic Gradient Descent*.

XIV) Dentre as alternativas abaixo, qual melhor descreve o processo de *hyperparameter tuning*?

(a) É um processo de otimização de parâmetros da rede neural que é feito via decida de gradiente.

(b) É um processo de busca cujo objetivo é encontrar valores ideais para parâmetros que não são aprendidos durante o processo de treino.

(c) É o processo de busca pela hipótese que melhor descreve a relação entre um conjunto de atributos e a variável alvo conforme uma função loss.

(d) É o processo equivalente a regularização, que visa limitar a complexidade do modelo para melhorar a sua capacidade de generalizar.

XV) Você está utilizando uma rede convolucional para fazer classificação de imagens. Depois de realizar o *deploy* do seu modelo, você percebe que a acurácia em produção é bem menor do que sua acurácia estimada em teste. Após horas de depuração você percebe que as imagens que os usuários mandam para classificação no seu sistema estão levemente rotacionadas e você não utilizou imagens com rotação para treinar seu modelo. Dentre as opções abaixo, quais seriam opções adequadas para melhorar a acurácia do seu modelo em produção?

(a) Não é necessário realizar nenhuma modificação pois redes convolucionais são invariantes a translação, rotação e escala.

(b) É possível adicionar ao *pipeline* de pré-processamento uma etapa de alinhamento das imagens para evitar que na camada de entrada do modelo existam imagens rotacionadas.

(c) É possível refazer o treinamento do modelo aplicando *data augmentation* com rotação das imagens de entrada.

(d) É possível refazer o treinamento por mais épocas aumentando o número de camadas convolucionais, de modo que o modelo treinado seja mais robusto a mudanças nos dados.

(e) É possível refazer o treinamento da rede aplicando regularização l2 e *dropout* para melhorar a generalização.

XVI) Uma das modificações mais comuns no algoritmo de descida de gradiente envolve alterar a atualização dos pesos para que considere dois termos: um dos termos é o gradiente da função *loss* em relação aos pesos considerando *batch* atual; enquanto o segundo termo é um valor definido recursivamente que leva em consideração os gradientes anteriores. Essa modificação obedece a seguinte equação:

$$W_{(t+1)} = W_{(t)} - \eta \mathcal{V}_{(t)}$$
$$\mathcal{V}_{(t)} = \beta \mathcal{V}_{(t-1)} + (1-\beta)\nabla\mathcal{L}$$

Sendo que por definição $\mathcal{V}_{(0)} = 0$.

Qual nome se dá para essa modificação na descida de gradiente?

(a) *Batch Gradient Descent*

(b) *Momentum*

(c) *L2 Regularization*

(d) *Stochastic Gradient Descent*

(e) *Batch Normalization*

XVII) Dentre as opções abaixo, quais adicionam não linearidades na rede neural.

(a) Adicionar camadas de convolução.

(b) Diminuir a taxa de aprendizado.

(c) Embaralhar o conjunto de treinamento.

(d) Usar função de ativação ReLU.

(e) Adicionar regularização.

XVIII) Você se depara com uma situação na qual o seu time de pesquisa está usando o maior *batch size* possível, respeitando apenas a limitação da memória disponível na GPU. Você entende existem argumentos a favor de um *batch size* grande, contudo você quer trazer para a discussão argumentos contrários para incentivar o uso de *batch sizes* menores. Dentre as alternativas abaixo, quais seriam bons argumentos para diminuir o *batch size*?

(a) Um *batch size* menor torna o processo de treino mais ruidoso, auxiliando na diminuição de *overfitting*.

(b) Um *batch size* menor pode tornar a paralelização mais otimizada e portanto o treinamento mais rápido.

(c) Um *batch size* menor pode levar a uma maior generalização da rede, evitando mínimos locais.

(d) Um *batch size* menor reduz o ruído no treinamento, melhorando a generalização do modelo treinado.

(e) O hiperparâmetro *batch size* apenas diz respeito a limitações de *hardware* e não influencia a solução que será otimizada.

Gabarito - Objetivas

I) a

II) a

III) b

IV) e

V) b

VI) b, g

VII) a, c

VIII) c, d

IX) c

X) b

XI) a, b

XII) a, d

XIII) c

XIV) b

XV) b, c

XVI) b

XVII) d

XVIII) a, c

Questões Dissertativas.

I) O preço de um determinado veículo i pode ser modelado satisfatoriamente pela seguinte função $f(\mathbf{X}^{(i)}) = 10000X_1^{(i)} + 50000X_2^{(i)} + 3500$. Responda as questões abaixo supondo que você está utilizando um *MultiLayer Perceptron* para aproximar o preço do veículo e justifique sua resposta.

   (a) Quantos neurônios/unidades são necessários no mínimo para modelar o preço dos veículos?

   (b) Qual função de ativação você deveria utilizar?

II) Cite e explique duas características desejáveis de uma função de ativação em redes neurais.

III) Por que é usual trabalhar com funções de ativação não-lineares em redes neurais?