

Lista 2

Lista preparatória para a segunda prova de Redes Neurais Artificiais.

Questões

- 1. Explique as diferenças entre Redes convolucionais e Redes profundas.**

Redes convolucionais: são redes que se caracterizam pelas camadas que visam extrair características das imagens dadas como entrada (camadas de convolução). Uma característica única no conceito de redes neurais. A partir dessas camadas, é possível realizar o aprendizado e classificar as imagens como for desejado.

Redes profundas: são redes que possuem por conceito duas camadas principais: entrada e saída. Entre essa camadas temos as camadas ocultas (*hidden layers*), que executarão o processo de aprendizado.

- 2. Dado [10,5,3,1,8] como saída de uma rede neural para um problema de cinco classes ([*ford, volks, gm, bmw, honda*]) onde a classe verdadeira do exemplo é *ford*. Calcule a entropia cruzada deste exemplo.**

2	10	Fold 5
	5	gm
	3	bmw
	1	honda
	8	

1. Aplicar softmax

$$S(y_i) = \frac{\exp(y_i)}{\sum_{j=1}^n \exp(y_j)}$$

$$\sum_{j=1}^n \exp(y_j) = 25179.641$$

$$S(10) = 22026.466 / 25179.641 = 0.875$$

$$S(5) = 149.413 / 25179.641 = 0.006$$

$$S(3) = 20.096 / 25179.641 = 0.001$$

$$S(1) = 2.718 / 25179.641 = 0.0$$

$$S(8) = 2970.956 / 25179.641 = 0.118$$

0.875
0.006
0.001
0.0
0.118

2. Aplicar a Cross Entropy

$$CE = -\sum_{j=0}^n y_j \log(p(c_j|x))$$

↑ número de classes
softmax do y_j

1	$\cdot \log(0.875)$	$= -0.13375$
0	$\cdot \log(0.006)$	$= 0$
0	$\cdot \log(0.001)$	$= 0$
0	$\cdot \log(0.0)$	$= 0$
0	$\cdot \log(0.118)$	$= 0$

$$CE = -1 \cdot (-0.13375) = 0.13375$$

$$-0.13375 = \sum_{j=0}^n y_j \log(p(c_j|x))$$

3. Explique como transformar validação cruzada em leave-one-out.

Leave-one-out é um tipo de validação cruzada em que a quantidade de *folds* é igual a quantidade de exemplos. Sendo assim, o conjunto de teste é composto apenas por um único exemplo do dataset (daí a origem do nome).

4. O que reflete melhor o desempenho do algoritmo em produção o erro no conjunto de validação ou no conjunto de teste?

O conjunto de validação é apresentado ao modelo com o fim de a cada iteração de época, durante o treinamento, a loss ser atualizada. Dessa maneira, ocorre a atualização dos pesos em direção a convergência.

Uma vez que o conjunto de validação é apresentado ao modelo (de acordo com a quantidade de iterações de

épocas), o desempenho calculado a partir desse conjunto se torna enviesado, visto que, a loss a corrigida a partir dele.

Portanto, o desempenho calculado em cima do conjunto de testes é o que melhor reflete a realidade do modelo, dado que será um conjunto de exemplos inéditos sendo apresentados ao modelo.

5. Quando o pré-processamento deve ser realizado depois da separação de treino e teste?

O pré-processamento é realizado após a separação do conjunto de treino e teste, quando o pré-processamento envolver a execução de processos (e.g., discretização) que podem acabar vazando informações do conjunto de teste para dentro do conjunto de treino. O que poderia prejudicar a confiabilidade da performance do modelo.

6. Qual é o perigo de utilizarmos o resultado da validação como resultado do modelo?

O conjunto de validação é apresentado ao modelo durante o período de treinamento. Logo, o erro calculado em cima desse conjunto pode estar enviesado. Uma vez que, existe a possibilidade de um overfitting a esses exemplos. O correto seria utilizarmos o resultado de um conjunto de teste, contendo exemplos inéditos nunca apresentados ao modelo.

7. Explique como fazer validação cruzada interna (nested cross-validation).

Assim como na validação cruzada comum, são realizadas as divisões de k *folds*, em que uma porcentagem (maior) será o conjunto de treino e a o restante o conjunto de teste.

Na validação cruzada interna, essa divisão será o que chamamos de camada externa (*Outer loop*). A diferença é que teremos mais um divisão similar a essa ocorrendo, que será realizada dentro do conjunto de treino. Assim teremos:

- Camada externa (*Outer loop*): divisão do dataset em k *folds* em treino e teste ($|treino| > |teste|$).
- Camada interna (*Inner Loops*): divisão dos conjuntos de teste em treino e validação ($|treino| > |validação|$).

Dessa forma a camada externa poderá estimar a qualidade do melhor modelo estimado pela camada interna. O nested cross-validation é importante para evitar o vazamento de informação do conjunto de treino para o conjunto de teste. A utilização do nested cross-validation ainda implica em evitar o overfitting dos modelos.

8. O que é o bias correction utilizado nos otimizadores?

O *bias correction* é responsável por "potencializar" os valores iniciais de um conjunto. Dessa forma ele permite "encontrar" (aproximadamente) os valores iniciais de uma série quando não se possui essa informação.

9. Qual é a relação entre Adam, momentum e RMSProp? Explique apresentando as equações do Adam.

Uma característica em comum destes três otimizadores (entre a maioria dos otimizadores), é a utilização do conceito de média móvel exponencial (MME).

- **Momentum**: representa a utilização o MME dentro do conceito de gradiente, permitindo uma convergência mais suave dos valores.

$$V = \beta \times V_{t-1} + (1 - \beta) \frac{\partial Loss}{\partial W_t}$$

Equação de atualização:

$$W_{t-1} = w_t - \alpha \times v_t$$

- **RMSProp**: é uma evolução de um outro otimizador nomeado **Ada Grad**. o **Ada Grad** permite uma atualização do hiperparâmetro learning rate. A evolução que o **RMSProp** proporciona é adicionar a ideia de MME ao **Ada Grad**.

$$S_t = \beta \times S_{t-1} + (1 - \beta) * (Loss)^2$$

Equação de atualização:

$$W_{t-1} = w_t - \frac{\alpha}{\sqrt{S_t + \xi}} \times \frac{\partial Loss}{\partial W_t}$$

- **Adam**: é a combinação entre o otimizador **Momentum** e o **RMSProp**.

The diagram illustrates the Adam optimization update rule as a combination of three components: Momentum, RMSProp, and a learning rate adjustment. It shows the Adam update rule:

$$W_{t+1} = W_t - \frac{\alpha}{\sqrt{S_t + \epsilon}} \cdot v_t$$

where $v_t = \beta_1 v_{t-1} + (1 - \beta_1) \nabla Loss$ and $S_t = \beta_2 S_{t-1} + (1 - \beta_2) [\nabla Loss]^2$.

Below the equations, arrows point from each component to the corresponding label: a downward arrow from the first term to 'ADAM', another downward arrow from the second term to 'MOMENTUM', and a third downward arrow from the third term to 'RMS PROP'.

10. Qual é o efeito do Weight decay na atualização de pesos dos algoritmos?

O **Weight Decay** possui como função regularizar a complexidade do modelo, de forma a limitar o crescimento exponencial da complexidade. Isso ocorre adicionando uma nova variável nomeada termo de regularização no cálculo da loss. O termo de regularização é composto por um λ e pela norma L2 do vetor de pesos ao quadrado:

$$\text{termo_regularizador} = \lambda * \|W\|_2^2$$

Onde o λ , é um hiperparâmetro, que controla a complexidade do modelo, ou seja, o valor da norma L2.

11. Qual é o problema de utilizar função logística como função de ativação?

A função logística (sigmoide) satura-se muito rapidamente, não permitindo a diferenciação dos valores que encontram-se em seus extremos (0 e 1). A logística deve ser aplicada a situação em que precisa-se de uma função "liga/desliga" uma vez que a sigmoide se aproxima, por exemplo, da função degrau.

12. Explique porque não é necessário aplicar softmax na última camada da rede antes de calcular cross-entropy.

O cálculo da cross-entropy já implica no cálculo da softmax, perdendo então a necessidade de aplicação na normalização softmax.

13. Considere um problema de análise de sentimento com as classes positivo, neutro e negativo.

(a) Como deve ser o setup da última camada da rede ao utilizar cross-entropy como função de perda.

A última camada da rede deve conter como saída três neurônios que irão representar cada uma das classes [positivo, neutro, negativo].

(b) Considere um exemplo da classe positiva, e uma que rede produz como saída os valores [0.5,0,1]. Qual é o valor da cross-entropy neste caso?

T3

$\hookrightarrow \log(p(c|x))$

b) Classe: positiva

classes = [positiva, neutra, negativa]

Saída = [0.5, 0, 1]

1. Softmax
$$S(y_i) = \frac{e^{p(y_i)}}{\sum_{j=1}^n e^{p(y_j)}}$$

$$\sum e^{p(y_j)} = 5.367$$

$$S(0.5) = 1.649 / 5.367 = 0.307$$

$$S(0) = 1 / 5.367 = 0.186$$

$$S(1) = 2.718 / 5.367 = 0.506$$

2. Cross-entropy
$$CE(y_i) = -\sum_{i=0}^n y_i \log(p(c|x))$$

$$\begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \cdot \log \begin{bmatrix} 0.307 \\ 0.186 \\ 0.506 \end{bmatrix} = -1.1902$$
$$+ -1.1902 = \sum_{i=0}^n \log(p(c|x))$$

$$CE = -1 \cdot (-1.1902) = 1.1902$$

14. Explique como uma skip connection presente nas ResNets podem fazer a rede ignorar algumas camadas da rede neural.

O conceito de **skip connection** foi introduzido a fim de tratar o problema de degradação. Esse problema consiste no comportamento de redes profundas que acabam por obter problemas de performances devido a profundidade de suas camadas.

A ideia é basicamente somar o valor da entrada (identidade) com o valor de saída da convolução de uma camada. Logo, o resultado de um **skip connection** será: $f(x) + x$.

15. A skip connection faz a soma da identidade com o fluxo da rede em diferentes pontos da rede. O que fazer com a identidade quanto o seu tamanho é incompatível com a região de junção na rede?

Quando a identidade não possui o tamanho compatível com a saída a ser somada é realizado o **downsample**. O **downsample**

consiste em aplicar convolução na identidade para tornar a dimensão da entrada compatível para a soma com a saída.

16. Considere o BPE e a lista de merges abaixo e faça a tokenização das palavras "inteligência" e "computação".

16

```
1{('c', 'i'): 'ci',
 2('ci', 'a'): 'cia',
 3('c', 'o'): 'co',
 4('co', 'm'): 'com',
 5('com', 'p'): 'comp',
 6('comp', 'u'): 'compu'}
```

• Tokenização BPE : INTELIGÊNCIA

- 1 [i, N, T, E, L, I, G, È, N, C, I, A] Resultado:
2 [i, N, T, E, L, I, G, È, N, C, I, A] [i, N, T, E, L, I, G, È, N, CIA]
3, 4, 5, 6 [i, N, T, E, L, I, G, È, N, CIA] X

• Tokenização BPE: COMPUTAÇÃO

- 1 [C O M P U T A Ç Ã O] X
2 [C O M P U T A Ç Ã O] X
3 [C O M P U T A Ç Ã O] Resultado:
4 [CO, M, P, U, T, A, Ç, Ä, O]
5 [COM, P, U, T, A, Ç, Ä, O]
6 [COMP, U, T, A, Ç, Ä, O]

17. Considere o WordPiece e o vocabulário abaixo faça a tokenização das palavras "facom" e "computacao".

17

computa
fa
##c
##a
##o
##m

Tokenização WordPiece: FACOM

FACOM X

FACOM X

FACOM X

FACOM → FA ##COM

##COM X

##COM X

##COM → FA ##C ##OM

##OM X

##OM → FA ##C ##O ##M

##M → FA ##C ##O ##M

Resultado:

[FA, ##C, ##O, ##M]

Tokenização WordPiece: COMPUTAÇÃO

COMPUTAÇÃO X

COMPUTAÇÃO X

COMPUTAÇÃO X

COMPUTAÇÃO → COMPUTA ##CAO Resultado:

##CAO X

##CAO X

##CAO → COMPUTA ##C ##AO

##AO

##AO → COMPUTA ##C ##A ##O

##O → COMPUTA ##C ##A ##O

18. O que acontece com palavras pouco frequentes no tokenizador WordPiece?

As palavras pouco frequentes tendem a estar "quebradas" no vocabulário do tokenizador.

19. O que acontece com palavras muito frequentes no tokenizador WordPiece?

As palavras mais frequentes tendem a estar de forma "completa" no vocabulário do tokenizador.

20. Contexto similar ou significado similar, qual dessas similaridades melhor representa word embeddings produzidos pelo Skip-gram? Explique por quê?

Contexto similar. Isso se deve a natureza da ideia por trás do Skip-gram, que tem como objetivo encontrar os termos ao redor do termo dado como input. Ou seja, ele encontra o contexto dado um termo central.

21. Explique como os embeddings dos tokens são treinados no transformer?

Internamente, o transformer, irá treinar as embeddings relativas a cada token pertencente ao seu vocabulário. Ou seja, primeiro é necessário a aplicação de um algoritmo de tokenização. No BERT esse algoritmo é o **Word Piece** e no GPT temos o **BPE**. Extraído o vocabulário, as embeddings de cada token serão inicializado como um vetor aleatório e o aprendizado ocorre por backpropagation em conjunto com o conceito de **positional encoding**.

Positional encoding é o conceito responsável por adicionar as embeddings a representação de proximidade entre as palavras do corpus.

22. Porque na equação do positional encoding tem seno e cosseno?

Com o conceito de **Positional Encoding** (PE) deseja-se, a partir da soma de um vetor V_2 com a embedding, gerar um

vetor V_3 que conterá em sua representação espacial, a proximidade entre as embeddings. Para isso, o PE necessita criar o V_2 . O V_2 será a projeção de um dado ponto no espaço em que as coordenadas são $(\cos\theta, \sin\theta)$. Dessa forma a inclinação de V_2 é dada pelo valor de θ .

23. Dois embeddings próximos devem ter positional encodings similares. Como isso é feito?

Temos que o **Positional encoding** depende o vetor V_2 para definir a relação de proximidade das embeddings. Sabemos também que, V_2 depende do valor de θ para obter sua projeção. Portanto, temos que embeddings próximas devem ser somadas com seus respectivos vetores V_2 de modo que θ não varie muito. E embeddings distantes possuem uma diferença maior entre os θ s que geraram seus respectivos vetores V_2 . A equação formal para definir o vetor a ser somado é:

$$PE_{(pos,2i)} = \sin(pos/10000^{2i/d_{\text{model}}})$$

$$PE_{(pos,2i+1)} = \cos(pos/10000^{2i/d_{\text{model}}})$$

24. Porque a operação escolhida para junção do embedding e do positional encoding é a soma?

O fator crucial para que o **Positional Encoding** funcione é o valor de θ que irá gerar o vetor a ser somado na embedding. É necessário que essa soma preserve a embedding original e as distâncias entre elas. Para que ocorra essa preservação é necessário preservar o ângulo das embeddings e a operação que permite a conservar esses ângulos é a operação de soma.

25. Qual é a função da multiplicação da Query vezes Key na self-attention do transformer?

Query e **Key** são projeções das embeddings. A multiplicação entre a **Query** e **Key** ($Q * K^T$) é um produto escalar que irá ter como função representar a relação entre as diferentes projeções das embeddings, ou seja, em alto nível irá representar a relação entre as palavras do corpus. Sendo que, as projeções similares acontecem quando $Q \approx K$. Uma vez que o produto escalar é representado por:

$$Q * K = \cos(\theta) * \|q\| * \|k\|$$

A matriz resultante dessa multiplicação, quando aplicada a softmax, teremos uma matriz de ativação contendo os valores mais altos representando as embeddings que possuem similaridade entre si.

26. O que é **Value** na self-attention do transformer?

Value, assim como a **Query** e a **Key**, também é uma projeção das embeddings do modelo. O resultado da multiplicação entre a **Query** e a **Key** irá gerar uma matriz de ativação. Portanto, a multiplicação dessa matriz de ativação com a **Value** é o resultado que será passado a frente dentro da arquitetura do transformers.

27. O que representa a resultante da equação de self-attention do transformer?

A resultante da equação de self-attention representa as palavras (embeddings) com forte relação entre si.

28. Considere os tokens *[Joao, gosta, de, Chocolate]*. A execução de softmax($Q.K^T$) resulta na seguinte matriz:

$$\text{matrix}([[0, 0, 0, 1], [1, 0, 0, 0], [0, 1, 1, 0], [1, 0, 0, 0]])$$

Quais são as palavras que essa matriz relaciona em cada linha?

(28)

[joao, gosta, de, chocolate]

Q

[Joao]
[Gosta]
[de]
[Chocolate]

 4×4 K^T

$$4 \times 4 \quad \begin{bmatrix} J & G & d & C \\ o & s & e & h \\ a & t & c & o \\ o & a & a & t \end{bmatrix}$$

$$4 \times 4 \quad \begin{bmatrix} Jg & JG & Jd & JC \\ 0 & 0 & 0 & 1 \\ 6J & 6G & 6d & 6C \\ 1 & 0 & 0 & 0 \\ Jg & JG & Jd & JC \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

$$4 \times 4 \quad \begin{bmatrix} Jg & JG & Jd & JC \\ 0 & 0 & 0 & 1 \\ 6J & 6G & 6d & 6C \\ 1 & 0 & 0 & 0 \\ Jg & JG & Jd & JC \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

$J = joão$ $G = gosta$
 $d = de$ $C = chocolate$

$$\begin{bmatrix} J \times J & J \times G & J \times d & J \times C \\ G \times J & G \times G & G \times d & G \times C \\ d \times J & d \times G & d \times d & d \times C \\ C \times J & C \times G & C \times d & C \times C \end{bmatrix}$$

29. Multiplique a matriz anterior pela matriz de Value:

$$array([[0, 1, 2], [3, 4, 5], [6, 7, 8], [9, 10, 11]])$$

O que a resultado desta multiplicação representa.

(29)

$$QK^T = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}_{4 \times 4} \cdot V = \begin{bmatrix} 0 & 1 & 2 \\ 3 & 4 & 5 \\ 6 & 7 & 8 \\ 9 & 10 & 11 \end{bmatrix}_{4 \times 3} = \begin{bmatrix} 9 & 10 & 11 \\ 0 & 1 & 2 \\ 9 & 11 & 13 \\ 0 & 1 & 2 \end{bmatrix}_{4 \times 3}$$

Dada a matriz de ativação, representada por $Q * K^T$, temos que o resultado desta multiplicação implica na ativação das embeddings que geraram os maiores valores (possuem relação forte). Pode-se ainda dizer que, $\text{softmax}(Q * K^T) * V$ é a média ponderada das projeções da value. Esse é o valor a ser passado à frente na arquitetura.

30. Qual é a vantagem de se utilizar multi-head self attention frente a apenas uma self attention?

Multi-head attention é a aplicação de um conjunto de múltiplas instâncias de self-attention. Sua principal vantagem é capacidade do modelo conseguir focar e aprender diferentes relações e fenômenos do dado que foi concedido como entrada.

31. Porque em problemas de tradução faz sentido utilizar a cross-attention?

O conceito de cross-attention significa que, na etapa de multi-head attention do decoder, a matriz de ativação ($Q * K^T$) utilizada será fornecida pelo encoder (contendo embeddings e relações do idioma 1) e a matriz de value utilizada é produto da etapa de masked multi-head attention (contendo embeddings e relações do idioma 2). A combinação entre as matrizes query e key advindas do encoder com a matriz de value proveniente do decoder então permitirá o aprendizado da tarefa de tradução.

32. Dado um token t_i no meio de uma frase, explique como o masked-attention consegue dar atenção apenas para as palavras anteriores a t_i ?

O masked-attention é responsável por transformar a matriz resultante de $Q * K^T$ em uma matriz diagonal. Dessa forma o modelo consegue decodificar a sentença de forma sequencial, dado que a matriz de ativação conterá os valores a serem ativados de forma sequencial como mostrado a figura a seguir:

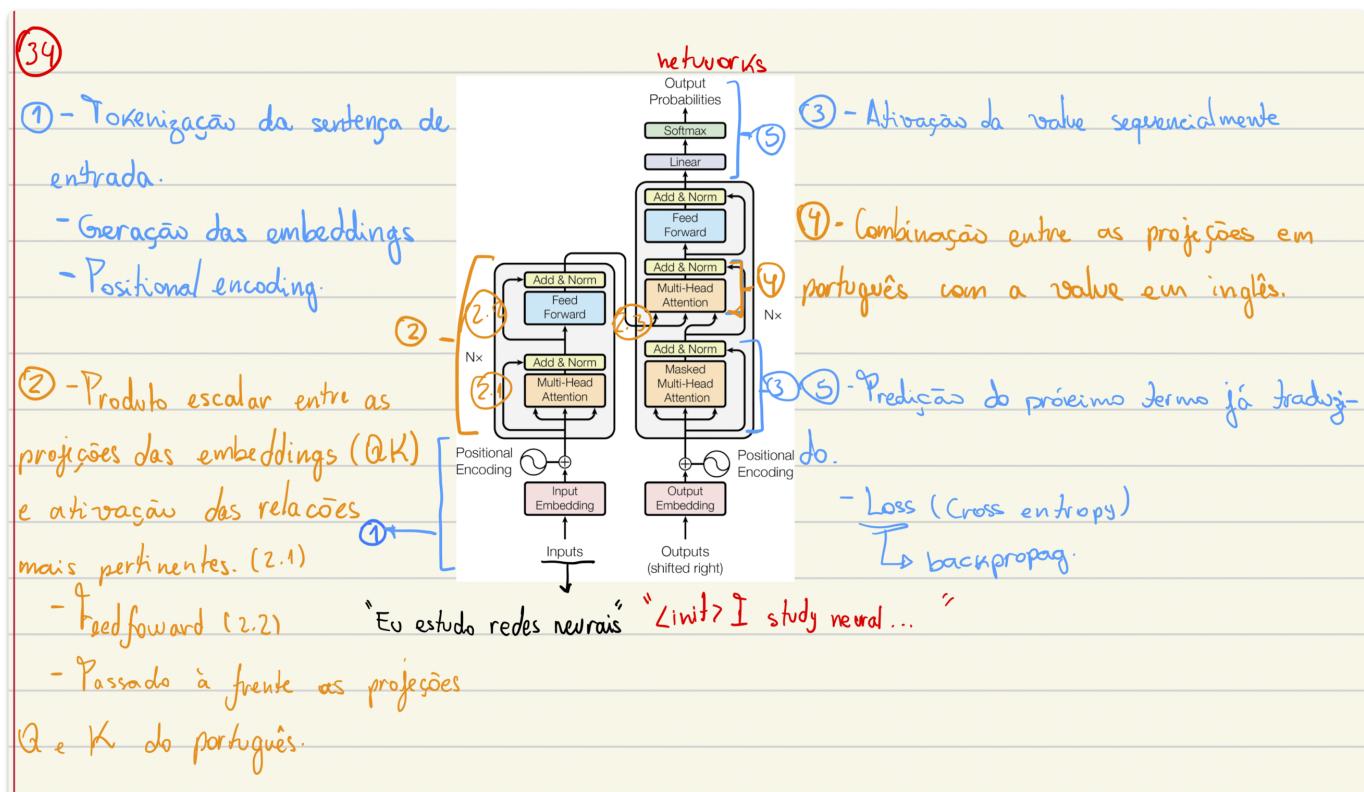
32

$$\begin{array}{c}
 Q \\
 \text{4x4} \\
 \left[\begin{array}{l} \text{emb. pal. 1 } p_1 \\ \text{emb. pal. 2 } p_2 \\ \text{emb. pal. 3 } p_3 \\ \text{emb. pal. 4 } p_4 \end{array} \right]
 \end{array}
 \quad
 \begin{array}{c}
 K^T \\
 \text{n x 4} \\
 \left[\begin{array}{cccc} e & e & e & e \\ m & m & m & m \\ b & b & b & b \\ p & p & p & p \\ a & a & a & a \\ i & i & i & i \\ 1 & 2 & 3 & 4 \\ p_1 & p_2 & p_3 & p_4 \end{array} \right]
 \end{array}
 \quad
 \begin{array}{c}
 = \\
 4 \times 4 \\
 \left[\begin{array}{cccc} p_1 p_1 & p_1 p_2 & p_1 p_3 & p_1 p_4 \\ p_2 p_1 & p_2 p_2 & p_2 p_3 & p_2 p_4 \\ p_3 p_1 & p_3 p_2 & p_3 p_3 & p_3 p_4 \\ p_4 p_1 & p_4 p_2 & p_4 p_3 & p_4 p_4 \end{array} \right] \\
 \xrightarrow{\text{diagonal}} \\
 \begin{array}{l}
 \rightarrow \text{Palavra 1} \\
 \rightarrow \text{Palavras 1 e 2} \\
 \rightarrow \text{Palavras 1, 2 e 3} \\
 \rightarrow \text{Palavras 1, 2, 3 e 4}
 \end{array}
 \end{array}$$

33. No treinamento do transformer na tarefa de tradução, o que é utilizado como classe? Qual é a função de perda (loss function) utilizada neste caso?

É utilizado como classe a próxima palavra a ser traduzida pelo modelo. Dado um input o modelo irá gerar uma saída para a sentença a ser traduzida. Quando a saída for gerada o modelo irá aplicar a como loss function a cross entropy de maneira a verificar a corretude do que foi predito, o resultado da loss então é retropropagado por toda a arquitetura do transformer. Note que, será necessário conceder ao modelo um corpus contendo as traduções do idiomas.

34. (!!!) Explique por meio de um exemplo como é feito treinamento do transformer para a tarefa de tradução de português para o inglês.



35. Qual é a vantagem do BERT frente aos modelos sequence to sequence existentes anteriormente?

O BERT, devido sua natureza encoder-only, tem acesso a toda a sentença dada como entrada, ao contrário dos modelos sequence to sequence. Ter acesso a toda sentença é uma vantagem em diferentes tarefas, como por exemplo análise de sentimento, onde seria necessário avaliar a sentença como um todo para definir a classificação da frase.

36. Explique como funcionado o masked LM para treinamento do BERT. Para este treinamento é necessário rotular dados?

O processo de MLM consiste na escolha aleatória de $n\%$ (sendo n um hiperparâmetro) dos tokens do input para que esse sejam substituídos por um token especial [MASK]. A partir dos termos que foram ocultos, o modelo irá calcular a cross entropy com base na predição feita para o termo que foi "escondido". Não é necessário rotular o dado pois já temos o dado real que deve ser predito pelo modelo apenas o "escondemos".

37. O que é preciso fazer para realizar ajuste-fine (fine-tunning) do BERT para uma tarefa de classificação?

Para utilizar o BERT em tarefas de classificação, primeiro será preciso modificar as últimas camadas do modelo adicionando as camadas lineares que forem desejadas, sendo que a última terá como saída a quantidade de classes da tarefa.

38. Explique como é treinado o GPT?

O GPT consiste em diversas camadas decoder empilhadas em sua arquitetura. Seu treinamento se dá pela predição do próximo token, onde na última camada, após o cálculo da loss, por backpropagation o valor é passado para toda arquitetura do GPT. É oferecido ao modelo um corpus (geralmente na casa de bilhões a trilhões de tokens) para

que seja possível aprender o padrões de linguagem. É possível ainda adicionar camadas de ajuste fino para as mais diversas tarefas como por exemplo: tornar o texto mais "humano", criar um sistema para que o que for predito não seja nocivo para a sociedade.

39. Considere um problema de análise de sentimento e modelos de mesmo tamanho, o que é mais provável, o BERT ser melhor que o GPT ou vice versa? Explique por quê.

BERT terá uma performance melhor que o GPT, uma vez que o BERT é composto por múltiplas camadas encoder do transformers. Temos que o encoder é capaz de analisar toda a sentença dado como input, isso permite que a análise de sentimento seja possível, uma vez que para determinar o sentimento é necessário ter conhecimento de todo o contexto presente.

40. Quando é recomendado utilizar BERT ao invés de GPT e vice versa?

BERT terá performances melhores que o GPT em tarefas em que o contexto completo da sentença de entrada deve ser considerada, ou seja, análise de sentimento, classificação de tema do texto, entre outros. Já o GPT, dada a sua arquitetura decoder only, irá sobressair em tarefas de geração e predição de texto dado uma sentença de entrada.

41. Como fazer In Context Learning (ICL) com o GPT?

ICL permite que o modelo consiga gerar um texto sobre algo que ele não possui treinamento para. Isso pode ser feito dando ao modelo um contexto sobre o que será pedido, o texto de entrada funciona como "ensinamento" e baseado nele é possível obter uma resposta muitas vezes precisa do modelo. O texto dado como contexto deve ser

aquele que possuir a embedding mais próxima da embedding da pergunta feita ao modelo.

42. Faz sentido fazer ICL no BERT? Explique por quê.

Dado apenas o conceito de ICL temos que não há muito sentido em aplicá-lo em modelos baseados na arquitetura do BERT. Entretanto, não há claras evidências e explicações na literatura que possam corroborar com a justificativa, ou com a utilidade em se aplicar ICL no BERT.