

“Any A.I. smart enough to pass a Turing test is smart enough to know to fail it” (Ian McDonald).

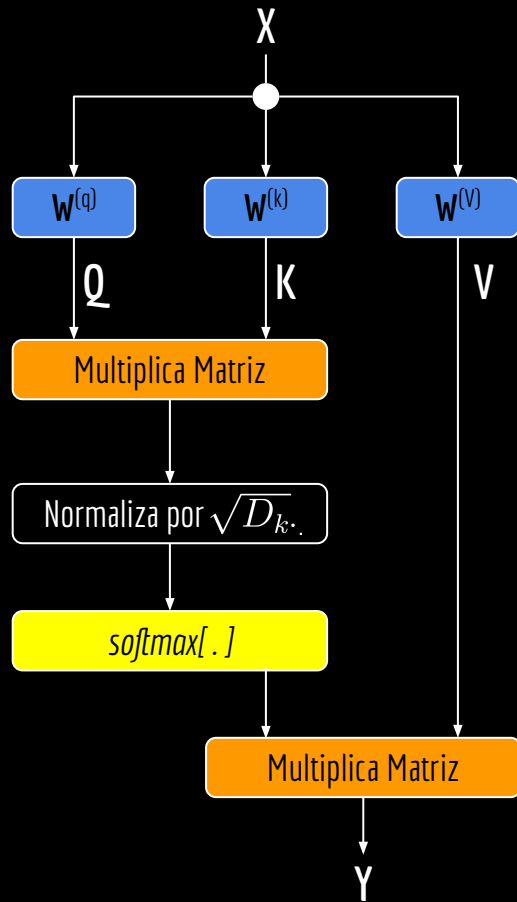
Large Language Models

Paulo Ricardo Lisboa de Almeida



Self-Attention Head

$$Y = \text{softmax} \left[\frac{QK^T}{\sqrt{D_k}} \right] V$$



Múltiplas cabeças

Da mesma forma que com redes convolucionais onde é útil ter múltiplos filtros, em uma rede baseada em transformers, é útil ter múltiplas cabeças.

Com H cabeças, concatenamos as respostas das cabeças, na forma:

$$\mathbf{H}_h = \textit{Atencao}(\mathbf{Q}_h, \mathbf{K}_h, \mathbf{V}_h) = \textit{softmax} \left[\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{D_k}} \right] \mathbf{V}$$

$$\mathbf{Y} = \textit{Concatenar}[\mathbf{H}_1, \dots, \mathbf{H}_H] \mathbf{W}^{(o)}$$

Múltiplas cabeças

$$\mathbf{Y} = \textit{Concatenar}[\mathbf{H}_1, \dots, \mathbf{H}_H] \mathbf{W}^{(o)}$$

Cada cabeça gera uma saída $\mathbf{H}_i$ de dimensão $N \times D_V$.

Concatenadas as saídas, temos uma matriz de $N \times HD_V$.

Múltiplas cabeças

$$\mathbf{Y} = \text{Concatenar}[\mathbf{H}_1, \dots, \mathbf{H}_H] \mathbf{W}^{(o)}$$

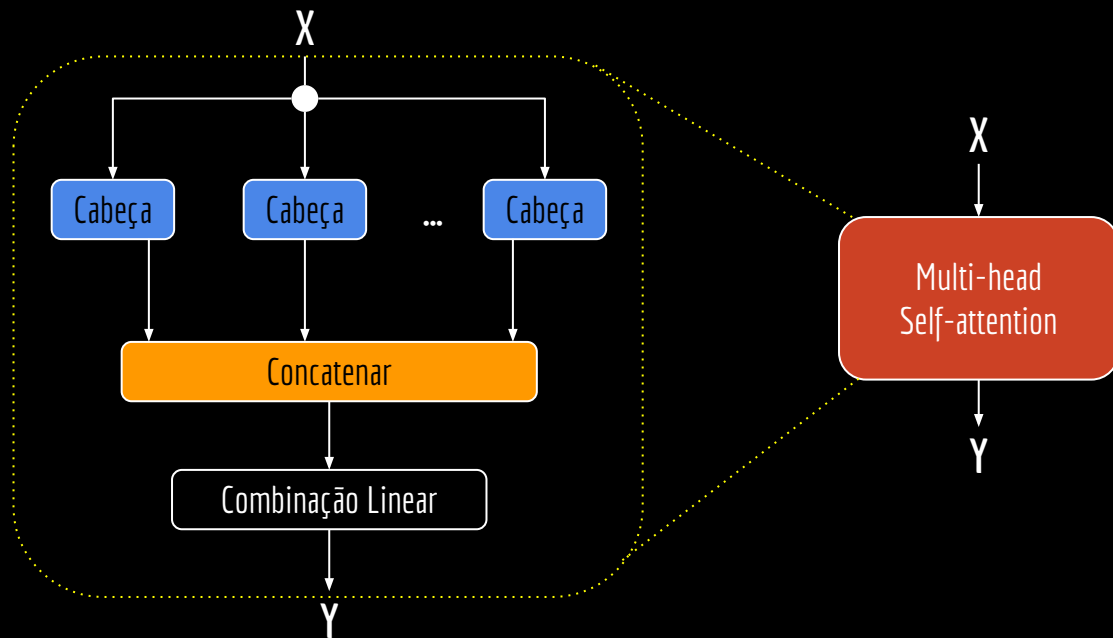
Cada cabeça gera uma saída $\mathbf{H}_i$ de dimensão $N \times D_V$.

Concatenadas as saídas, temos uma matriz de $N \times HD_V$.

A matriz $\mathbf{W}^{(o)}$ comumente tem dimensões $HD_V \times D_V$ para que o resultado final tenha dimensão $N \times D_V$.

Os pesos de $\mathbf{W}^{(o)}$ são calculados durante o treinamento.

Multi-head Self-attention



Faça você mesmo

Descubra (usando o código do Google Colab) quantas cabeças a versão da rede Llama 3.2 que estamos usando possui.

Múltiplas Camadas

Ideia: transformar o conjunto de múltiplas cabeças visto anteriormente em uma camada de uma rede.

Criar um rede profunda, composta de múltiplas dessas camadas.

Múltiplas Camadas

Primeiro, para facilitar a convergência vamos:

Adicionar conexões residuais.

Ideia: fazer um bypass dos dados da camada diretamente para sua saída.

Ideia usada em outros tipos de redes, como CNNs. Veja na literatura.

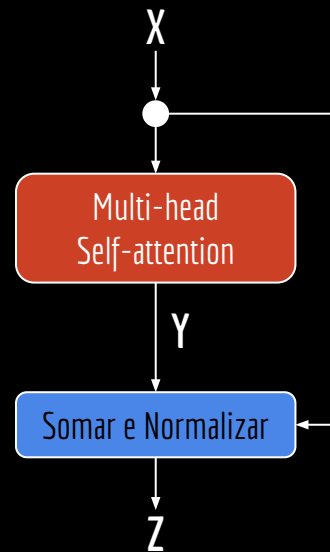
Normalizar a camada.

Vamos usar *Layer Normalization*.

Ideia similar ao batch normalization de aulas passadas. Veja na literatura.

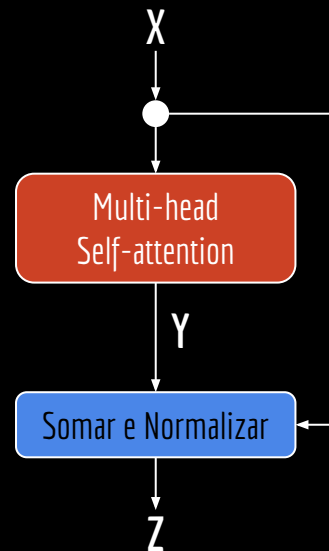
Múltiplas Camadas

$$\mathbf{Z} = \text{LayerNorm}[\mathbf{Y} + \mathbf{X}]$$



Múltiplas Camadas

A única não linearidade até então é dada pela função *softmax*[.] aplicada na *multi-head self-attention*.



Múltiplas Camadas

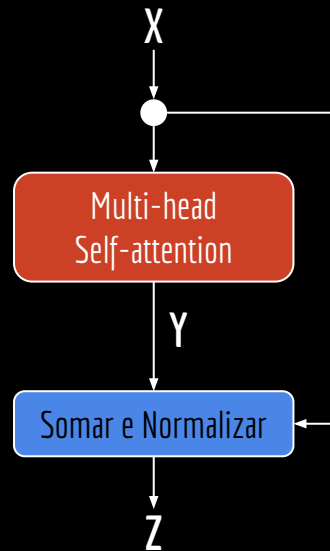
A única não linearidade até então é dada pela função *softmax*[.] aplicada na *multi-head self-attention*.

Podemos adicionar um MLP na saída **Z** para aumentar a flexibilidade da rede.

O *MLP*[.] deve preservar as dimensões dos dados.

Entrada de tamanho D , e saída de tamanho D .

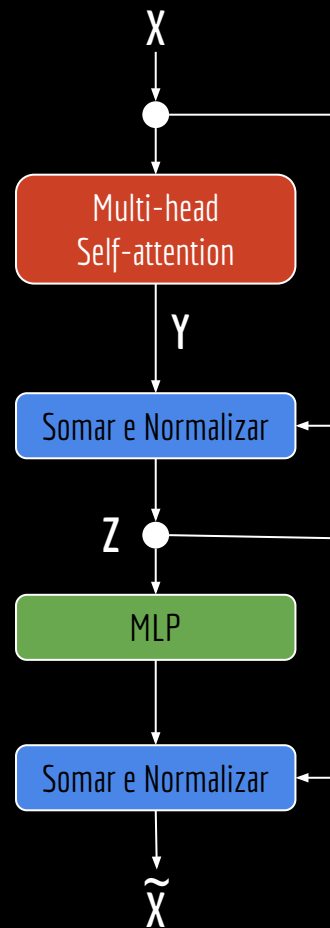
Incorporamos ainda conexões residuais e normalização no MLP.



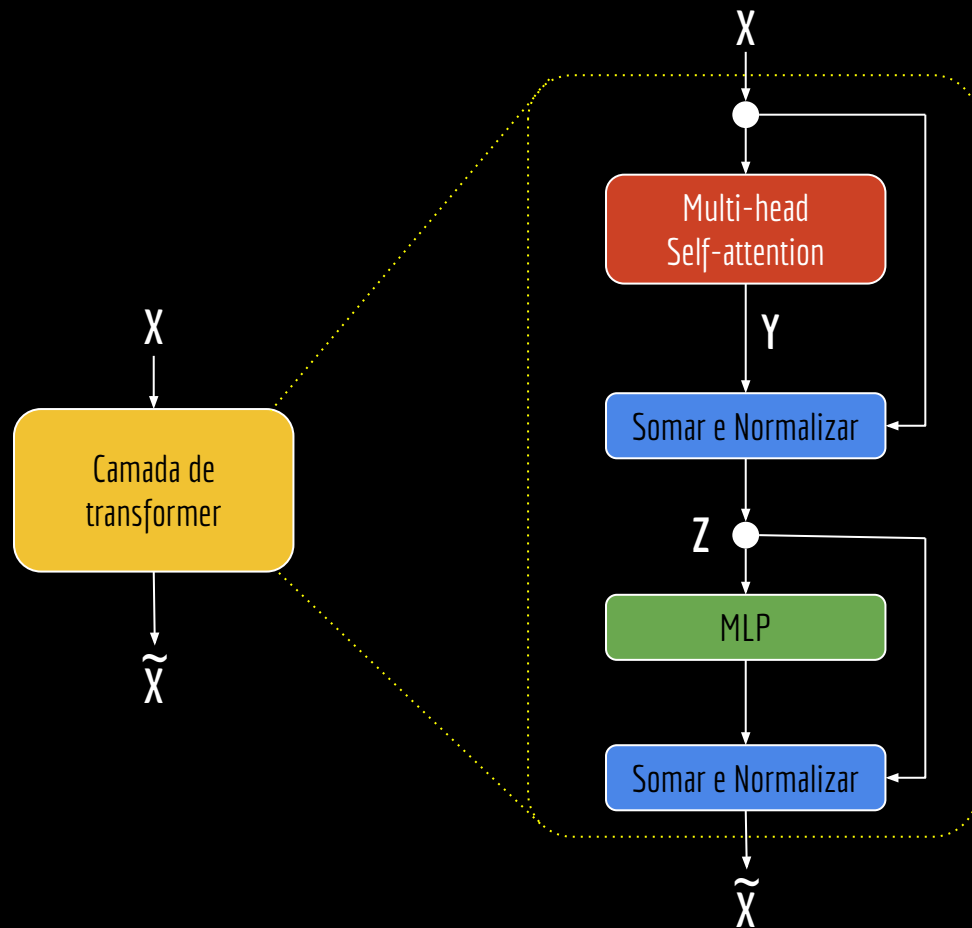
Múltiplas Camadas

$$\mathbf{Z} = \text{LayerNorm}[\mathbf{Y} + \mathbf{X}]$$

$$\tilde{\mathbf{X}} = \text{LayerNorm}[\text{MLP}[\mathbf{Z}] + \mathbf{Z}]$$



Múltiplas Camadas



Faça você mesmo

Descubra (usando o código do Google Colab) quantas camadas de *transformer* a rede Llama 3.2 que estamos usando possui.

Positional encoding

Até agora um transformer é equivariante a permutações da entrada.

Em muitos casos os dados são sequenciais, e a ordem importa.

Exemplo:

“Ele disse que viria, mas não veio.”

“Ele disse que não viria, mas veio.”

Positional encoding

Criar um vetor de posição $\mathbf{r}_n$, com as mesmas dimensões de $\mathbf{x}_n$.

O vetor a ser enviado para a rede, com as informações posicionais fica:

$$\tilde{\mathbf{x}}_n = \mathbf{x}_n + \mathbf{r}_n$$

Positional encoding

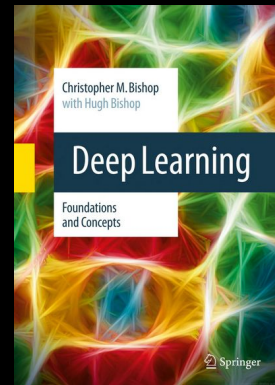
Criar um vetor de posição $\mathbf{r}_n$, com as mesmas dimensões de $\mathbf{x}_n$.

O vetor a ser enviado para a rede, com as informações posicionais fica:

$$\tilde{\mathbf{x}}_n = \mathbf{x}_n + \mathbf{r}_n$$

$\mathbf{r}_n$ geralmente é uma combinação de valores de senos e cossenos.

Veja em Bishop (2024) e Vaswani et al. (2017) como a inclusão de $\mathbf{r}_n$ não corrompe os vetores de entrada, e como a combinação de senos e cossenos consegue codificar a posição dos *tokens*.



Self-supervised learning

Considerando modelos de linguagem, uma das formas que podemos fazer o treinamento é mascarando trechos das entradas, e computando o loss baseado na saída gerada para a próxima palavra.

A previsão do tempo é de chuva para hoje, é melhor levar um <máscara>.

Dessa forma, grandes volumes de dados textuais podem ser processados para realizar o treinamento.

Inferência

A inferência em LLMs comumente é feita como um modelo **auto-regressivo**.

Considerando as $n-L$ palavras anteriores, prever a *enésima* palavra na sequência.

Depois, utilizar a palavra recém gerada como parte da entrada, e prever a palavra $n+1$.

Quanto maior o valor de L , mais informação passada pode ser levada em consideração.

Mas com um custo computacional mais alto.

$$p(x_n | x_{n-1}, x_{n-2}, \dots, x_{n-L})$$

Inferência

Vou te dar o começo de uma frase. Você deve continuar a história.

Certa vez, ele resolveu que não seria **mais**

Vou te dar o começo de uma frase. Você deve continuar a história.
Certa vez, ele resolveu que não seria



mais levado pela rotina monótona que o cercava. Decidiu que queria ver o mundo com outros olhos, mesmo que isso significasse abandonar sua zona de conforto. Pegou uma mochila, encheu-a com poucas roupas e muitas expectativas, e saiu sem olhar para trás.

A cada cidade que passava, ele encontrava novas faces, novas paisagens e histórias que o faziam ver o quanto a vida podia ser surpreendente. Logo, ele entendeu que a verdadeira liberdade estava em viver cada dia como se fosse único, sem se apegar aos planos ou ao que os outros esperavam dele.

E foi assim, seguindo seus próprios passos, que ele descobriu quem realmente era.

ChatGPT.

Considerando token = palavras e pontuação.

$L = 23$.

Entrada: $N - L$ Tokens.

Saída: próximo Token.

Tokens que estão fora da janela.

Inferência

Vou te dar o começo de uma frase. Você deve continuar a história.

Certa vez, ele resolveu que não seria mais **levado**

Vou te dar o começo de uma frase. Você deve continuar a história.
Certa vez, ele resolveu que não seria



mais levado pela rotina monótona que o cercava. Decidiu que queria ver o mundo com outros olhos, mesmo que isso significasse abandonar sua zona de conforto. Pegou uma mochila, encheu-a com poucas roupas e muitas expectativas, e saiu sem olhar para trás.

A cada cidade que passava, ele encontrava novas faces, novas paisagens e histórias que o faziam ver o quanto a vida podia ser surpreendente. Logo, ele entendeu que a verdadeira liberdade estava em viver cada dia como se fosse único, sem se apegar aos planos ou ao que os outros esperavam dele.

E foi assim, seguindo seus próprios passos, que ele descobriu quem realmente era.

ChatGPT.

Considerando token = palavras e pontuação.

$L = 23$.

Entrada: $N - L$ Tokens.

Saída: próximo Token.

Tokens que estão fora da janela.

Inferência

Vou te dar o começo de uma frase. Você deve continuar a história.

Certa vez, ele resolveu que não seria mais levado **pela**

Vou te dar o começo de uma frase. Você deve continuar a história.
Certa vez, ele resolveu que não seria



mais levado pela rotina monótona que o cercava. Decidiu que queria ver o mundo com outros olhos, mesmo que isso significasse abandonar sua zona de conforto. Pegou uma mochila, encheu-a com poucas roupas e muitas expectativas, e saiu sem olhar para trás.

A cada cidade que passava, ele encontrava novas faces, novas paisagens e histórias que o faziam ver o quanto a vida podia ser surpreendente. Logo, ele entendeu que a verdadeira liberdade estava em viver cada dia como se fosse único, sem se apegar aos planos ou ao que os outros esperavam dele.

E foi assim, seguindo seus próprios passos, que ele descobriu quem realmente era.

ChatGPT.

Considerando token = palavras e pontuação.

$L = 23$.

Entrada: $N - L$ Tokens.

Saída: próximo Token.

Tokens que estão fora da janela.

Inferência

Vou te dar o começo de uma frase. Você deve continuar a história.

Certa vez, ele resolveu que não seria mais levado pela **rotina**

Vou te dar o começo de uma frase. Você deve continuar a história.
Certa vez, ele resolveu que não seria



mais levado pela rotina monótona que o cercava. Decidiu que queria ver o mundo com outros olhos, mesmo que isso significasse abandonar sua zona de conforto. Pegou uma mochila, encheu-a com poucas roupas e muitas expectativas, e saiu sem olhar para trás.

A cada cidade que passava, ele encontrava novas faces, novas paisagens e histórias que o faziam ver o quanto a vida podia ser surpreendente. Logo, ele entendeu que a verdadeira liberdade estava em viver cada dia como se fosse único, sem se apegar aos planos ou ao que os outros esperavam dele.

E foi assim, seguindo seus próprios passos, que ele descobriu quem realmente era.

ChatGPT.

Considerando token = palavras e pontuação.

$L = 23$.

Entrada: $N - L$ Tokens.

Saída: próximo Token.

Tokens que estão fora da janela.

Inferência

Vou te dar o começo de uma frase. Você deve continuar a história.

Certa vez, ele resolveu que não seria mais levado pela rotina
monótona

Vou te dar o começo de uma frase. Você deve continuar a história.
Certa vez, ele resolveu que não seria



mais levado pela rotina monótona que o cercava. Decidiu que queria ver o mundo com outros olhos, mesmo que isso significasse abandonar sua zona de conforto. Pegou uma mochila, encheu-a com poucas roupas e muitas expectativas, e saiu sem olhar para trás.

A cada cidade que passava, ele encontrava novas faces, novas paisagens e histórias que o faziam ver o quanto a vida podia ser surpreendente. Logo, ele entendeu que a verdadeira liberdade estava em viver cada dia como se fosse único, sem se apegar aos planos ou ao que os outros esperavam dele.

E foi assim, seguindo seus próprios passos, que ele descobriu quem realmente era.

ChatGPT.

Considerando token = palavras e pontuação.

$L = 23$.

Entrada: $N - L$ Tokens.

Saída: próximo Token.

Tokens que estão fora da janela.

Escolhendo o próximo Token

A rede gera como resposta as probabilidades *a posteriori* (através da função *softmax*) de cada token no dicionário ser a próxima palavra na sentença.

Poderíamos fazer uma busca gananciosa (*greedy search*), e usar o token com a maior probabilidade.

O que você esperaria com isso?

Escolhendo o próximo Token

A rede gera como resposta as probabilidades *a posteriori* (através da função *softmax*) de cada token no dicionário ser a próxima palavra na sentença.

Poderíamos fazer uma busca gananciosa (*greedy search*), e usar o token com a maior probabilidade.

- + Baixo custo.

Resultado determinístico (se isso é bom ou ruim ... depende).

Escolhendo o próximo Token

A rede gera como resposta as probabilidades *a posteriori* (através da função *softmax*) de cada token no dicionário ser a próxima palavra na sentença.

Poderíamos fazer uma busca gananciosa (*greedy search*), e usar o token com a maior probabilidade.

- + Baixo custo.

Resultado determinístico (se isso é bom ou ruim ... depende).

- Uma busca gananciosa não garante que a sentença como um todo é a que tem a melhor probabilidade *a posteriori* combinada (de forma similar a uma busca gananciosa em um grafo).

Temperatura

Uma técnica que possui um bom custo x benefício e é geralmente empregada é o uso de uma temperatura T na função softmax.

Uma possibilidade é combinar com a seleção *top-p*, onde selecionamos todos os tokens, ordenados da maior probabilidade para a menor, até que a soma das probabilidades seja maior ou igual a p .

Selecionar aleatoriamente um desses tokens.

Saída da rede com temperatura é:

$$y_i = \frac{e^{\frac{\hat{y}_i}{T}}}{\sum_j e^{\frac{\hat{y}_j}{T}}}$$

e é o número de Euler.

$\hat{\mathbf{y}}$ é a saída final da rede antes de passar pelo *softmax*.

j é a quantidade de itens no dicionário de *tokens*.

Faça você mesmo

Considere o exemplo disponibilizado no Colab.

No exemplo é dado um vetor de exemplo de *logits* (saídas antes de passar pelo softmax).

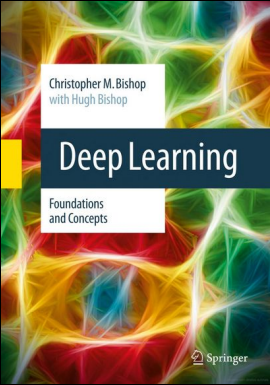
Não está implementado o conceito de temperatura. Implemente, e verifique como a saída é modificada de acordo com a temperatura.

Exercícios

1. Modifique o exemplo do Llama 3 disponibilizado no Google Colab para introduzir o conceito de auto-regressão. O modelo vai gerar texto até encontrar um número máximo de tokens, ou até encontrar um token de fim de texto.
2. Teste diferentes temperaturas para gerar o texto do item 1.

Referências

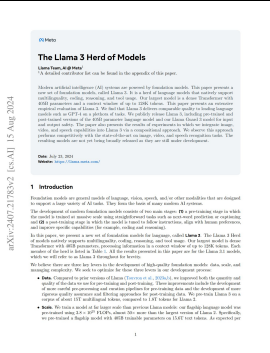
Bishop, C. M., Bishop, H. Deep Learning: Foundations and Concepts. 2023.



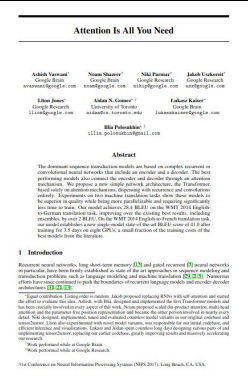
Bahdanau, Cho e Bengio. Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473, 2014.



Dubey, A. et al. The Llama 3 Herd of Models. 2024.



Vaswani, Shazeer, Parmar, Uszkoreit, Jones, Gomez, Kaiser e Polosukhin. Attention is all you need. Advances in Neural Information Processing Systems, 2017.



Licença

Esta obra está licenciada com uma Licença [Creative Commons Atribuição 4.0 Internacional](https://creativecommons.org/licenses/by/4.0/).