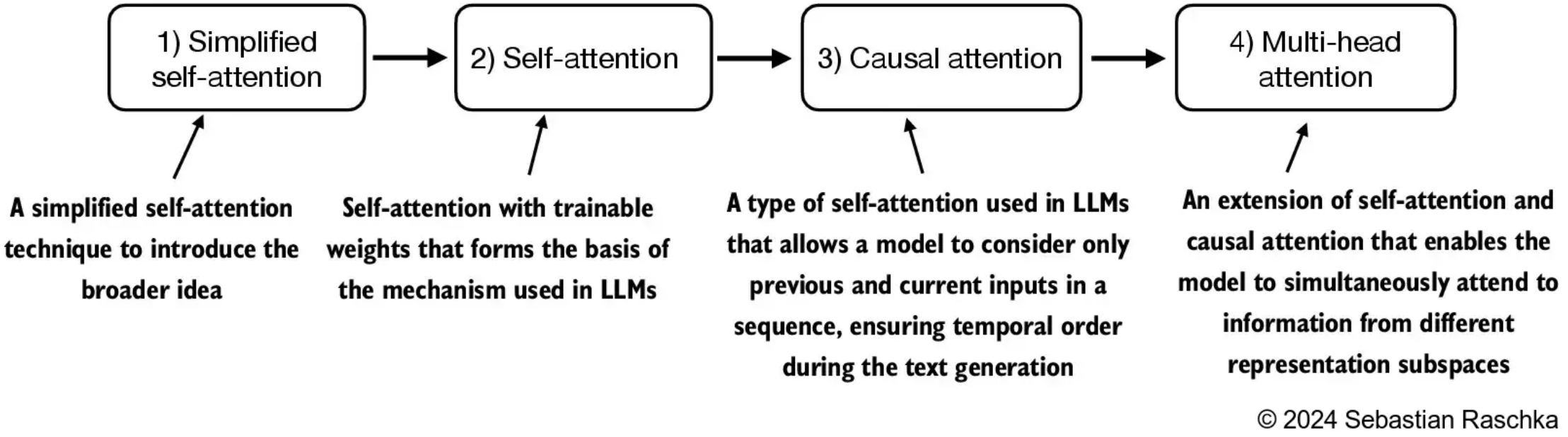
Self-Attention

Tópicos em Ciência de Dados

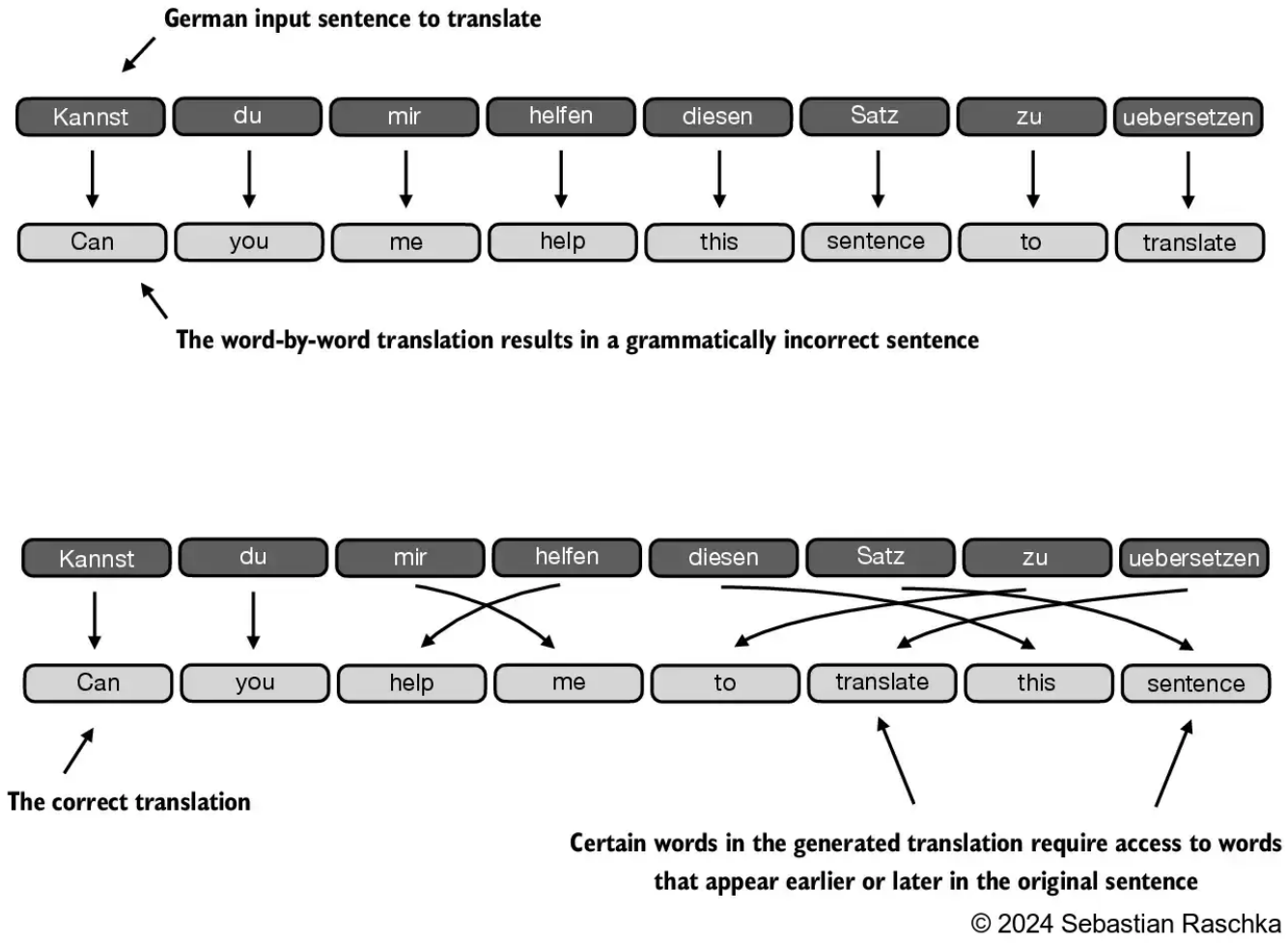
Pontifícia Universidade Católica de Campinas

Prof. Dr. Denis M. L. Martins

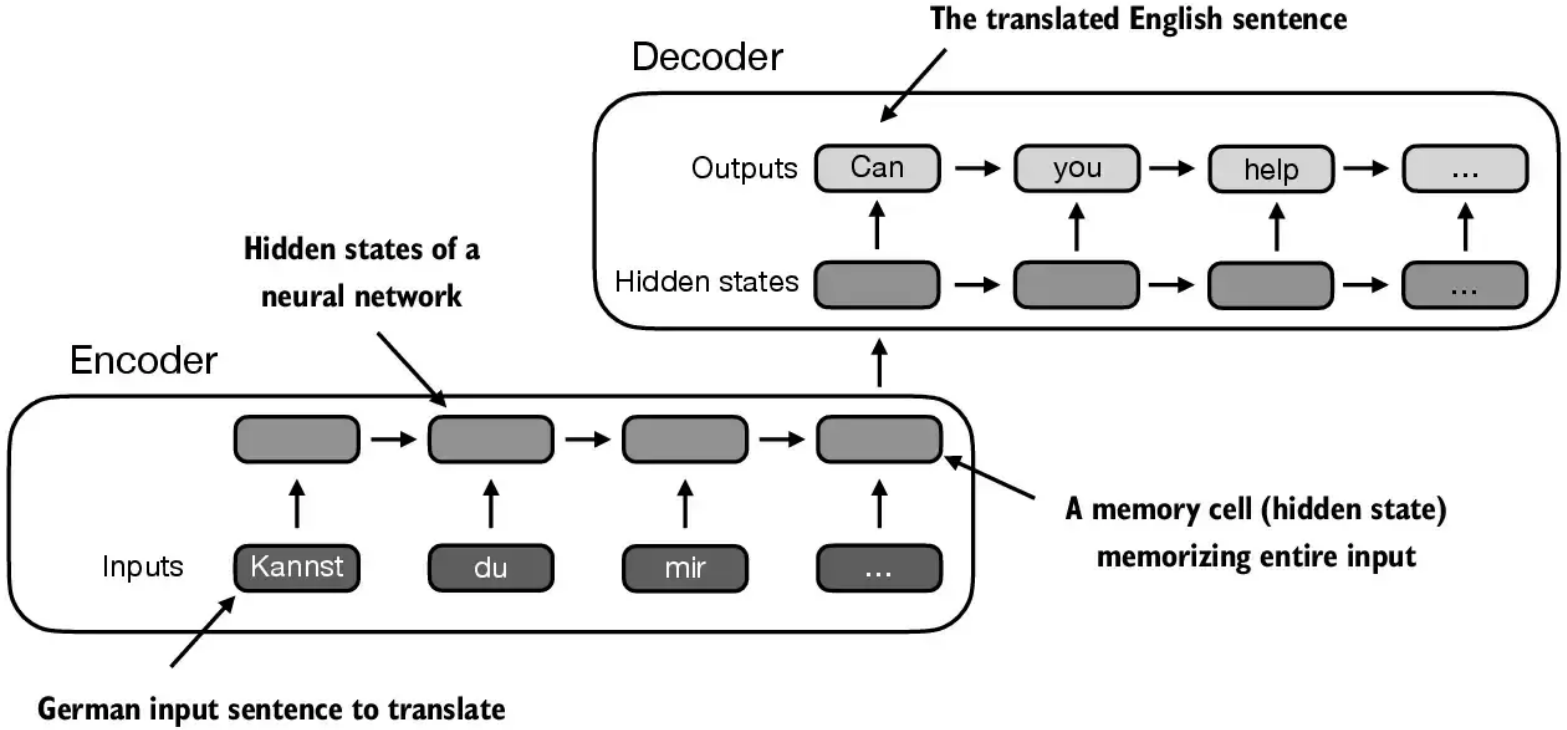
Diferentes Mecanismos de Atenção



Problema com longas sequências de texto



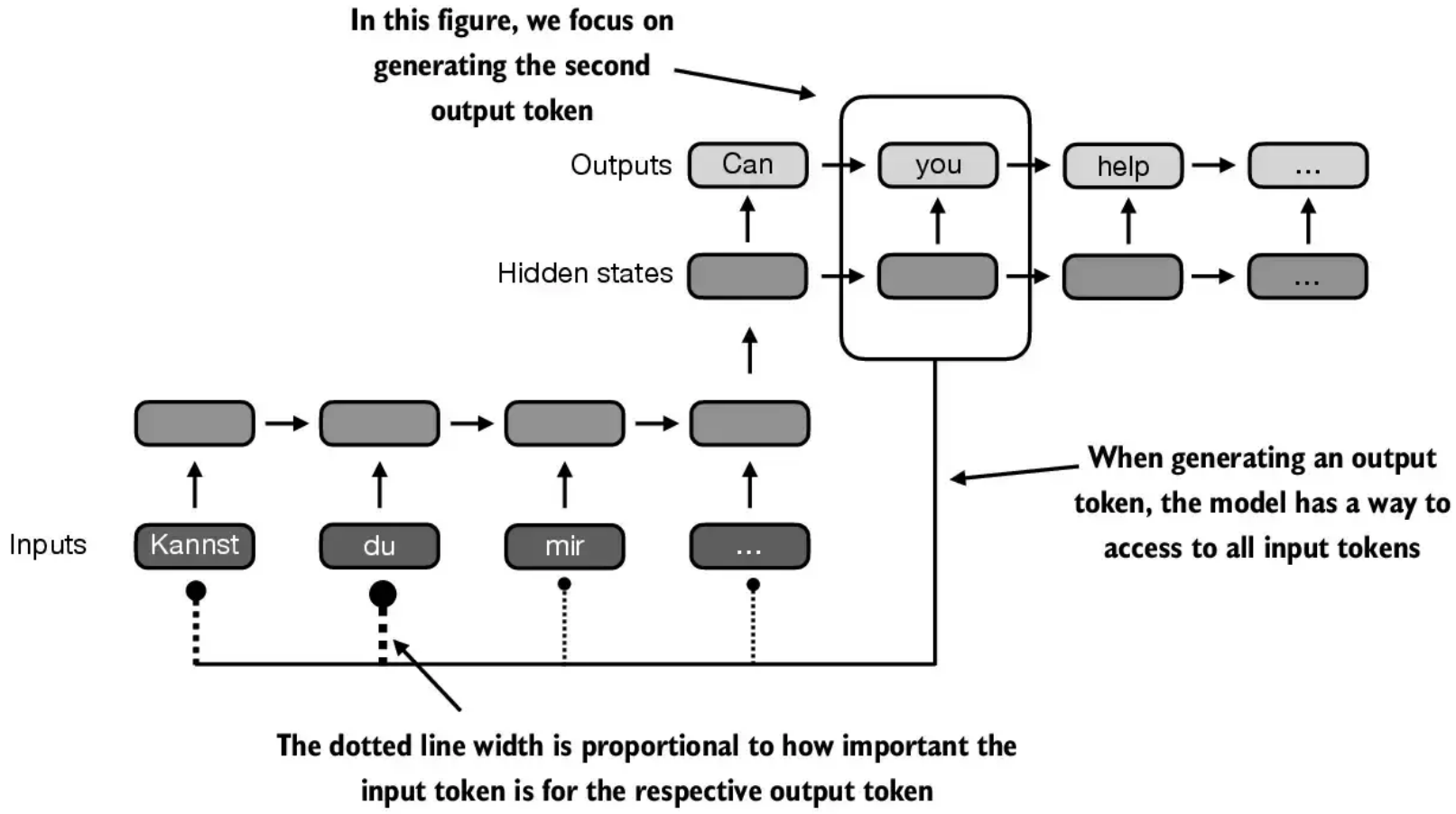
RNN perdem referência em longas sequências



RNNs/GRUs apresentam dificuldade de capturar dependências longas.
Custo computacional linear no comprimento da sequência.

ver tbm: https://learnopencv.com/attention-mechanism-in-transformer-neural-networks/

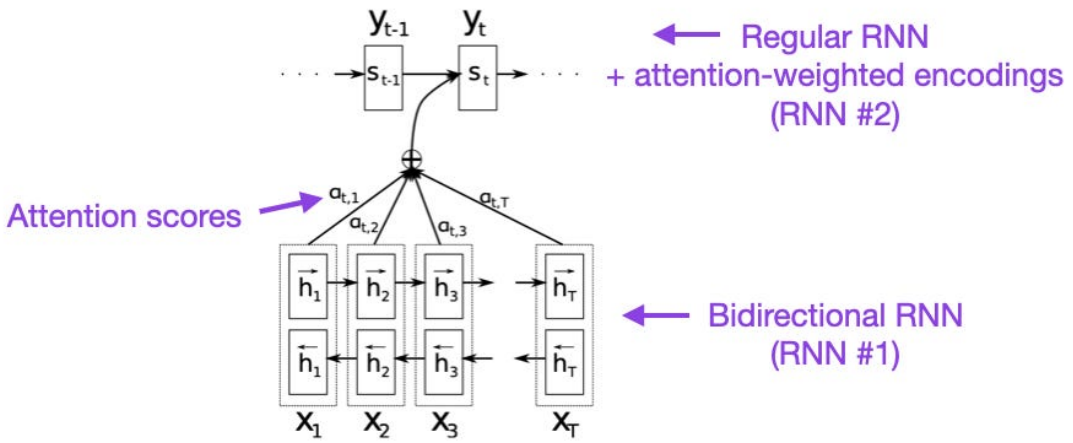
Resolvendo o problema através de atenção



Breve Histórico

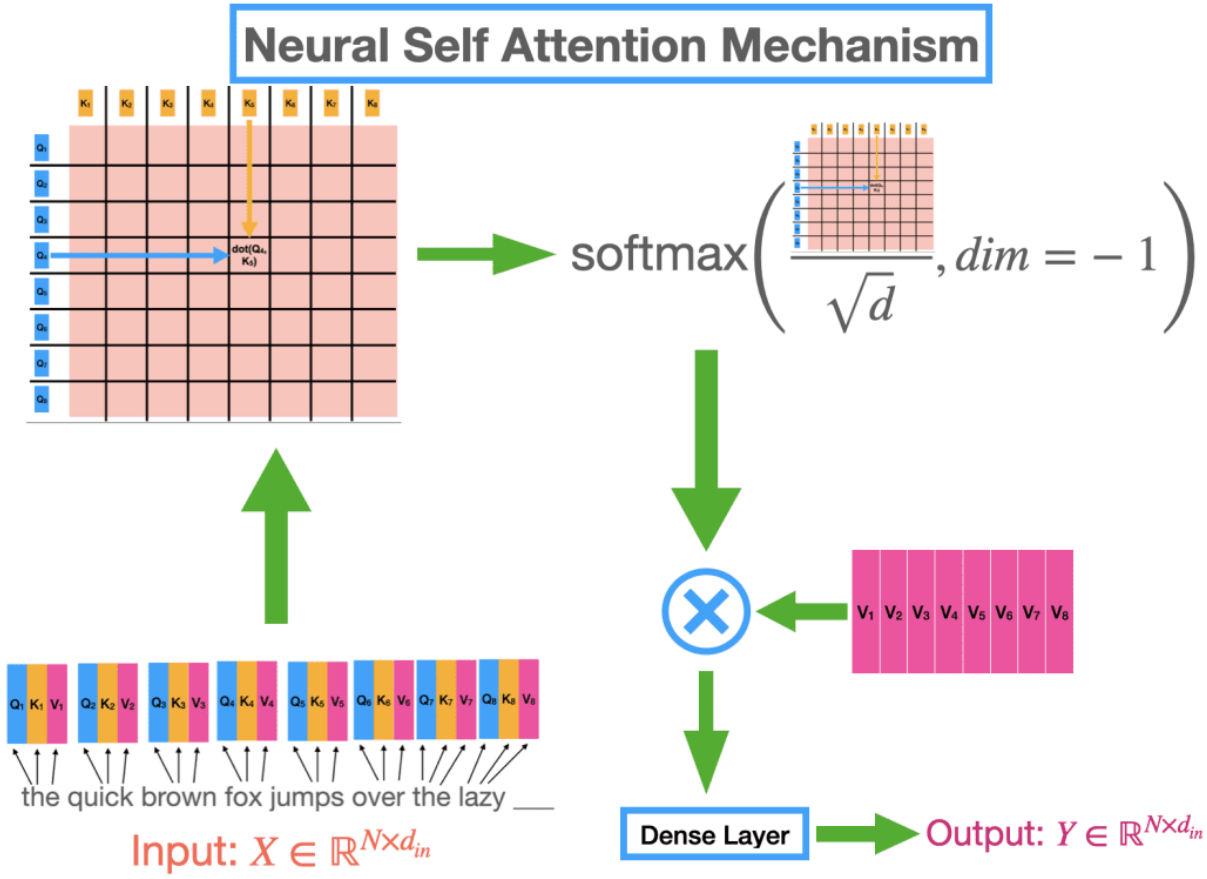
"Intuitively, this implements a mechanism of attention in the decoder. The decoder decides parts of the source sentence to pay attention to. (...) information can be spread throughout the sequence (...)."

- 2014: Primeiro mecanismo de atenção.
  - Contexto de Neural Translation.
  - Bengio é um dos autores.
- 2017: Transformers, self-attention e multi-head attention.
  - Dot product attention (próximos slides)
  - Proposta similar de 1991 por Schmidhuber: Fast Weight Programmer (hoje conhecida como Linear Transformers).
- 2018: Primeiro GPT é proposto.



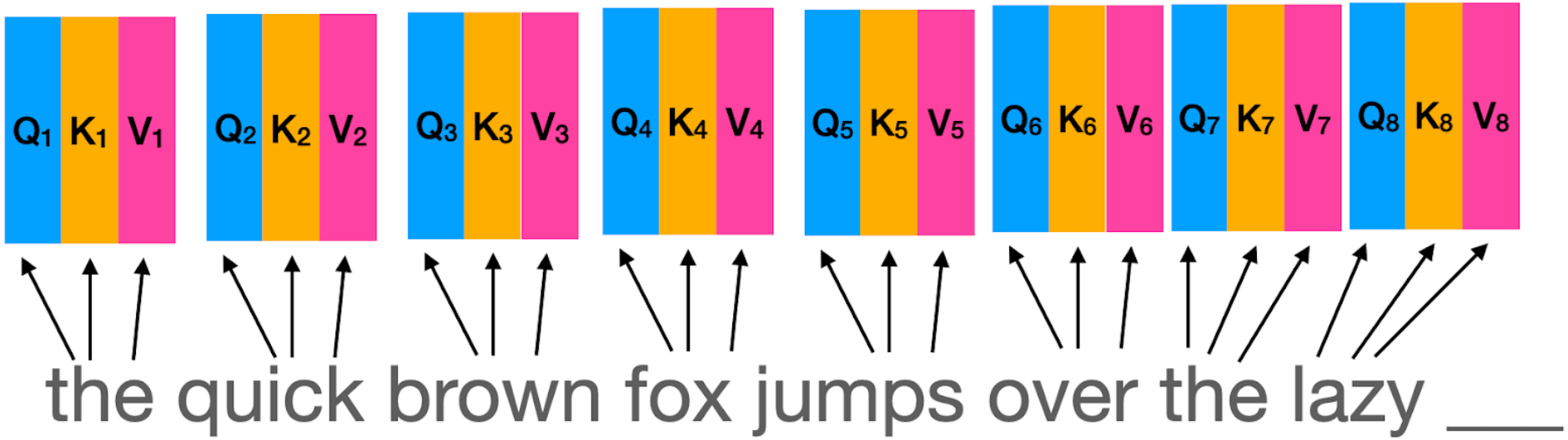
Primeiro mecanismo de atenção. Fonte: https://arxiv.org/abs/1409.0473.

Self-Attention: Visão Geral



Visão geral do mecanismo de atenção em Transformers. Fonte: Jaiyam Sharma @LearnOpenCV.

Self-Attention: Query, Key, Value



Primeiro passo no mecanismo de atenção. Fonte: Jaiyam Sharma @LearnOpenCV.

- Projeções lineares onde os embeddings dos tokens são multiplicados por matrizes de parâmetros W^K, W^Q e W^V da rede neural (aprendidos durante o treinamento).

- K: representam o token de origem.
- Q: representam os tokens de destino.
- V: representam a semântica e contexto dos tokens.

Detalhes do funcionamento na lousa

Self-Attention

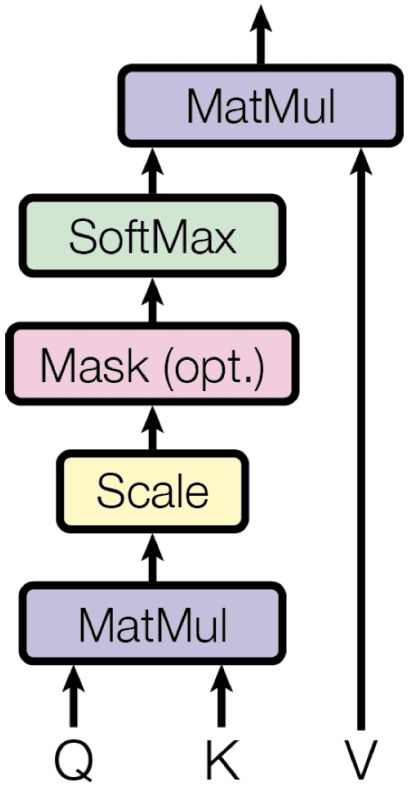
$$\text{Attention}(Q, K, V) = \text{Softmax}\left(\frac{Q\,K^\top}{\sqrt{d_k}}\right) V$$

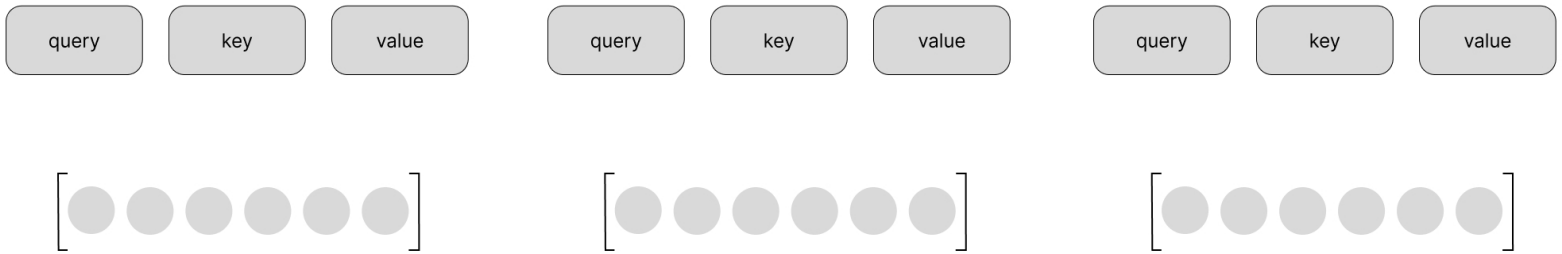
onde:

- $X \in \mathbb{R}^{n \times d_{\text{model}}}$ é a matriz de embeddings da sequência (n = comprimento).
- $W_Q, W_K, W_V \in \mathbb{R}^{d_{\text{model}} \times d_k}$ são as matrizes de projeção treináveis.

$$Q = X\,W_Q, \qquad K = X\,W_K, \qquad V = X\,W_V$$

- d_k é a dimensionalidade dos vetores query e key (usado para estabilidade numérica).
- Cada palavra cria uma query e recebe keys e values das demais palavras.
- Dot product $Q \cdot K^\top$ mede a similaridade entre queries e keys.
- Softmax esses números em pesos (probabilidades) $\rightarrow$ pesos de atenção.
- Os values são então somados ponderadamente, produzindo uma representação que leva em conta todas as palavras relevantes.





Mecanismo self-attention. Fonte:Jeremy Jordan.

Self-Attention

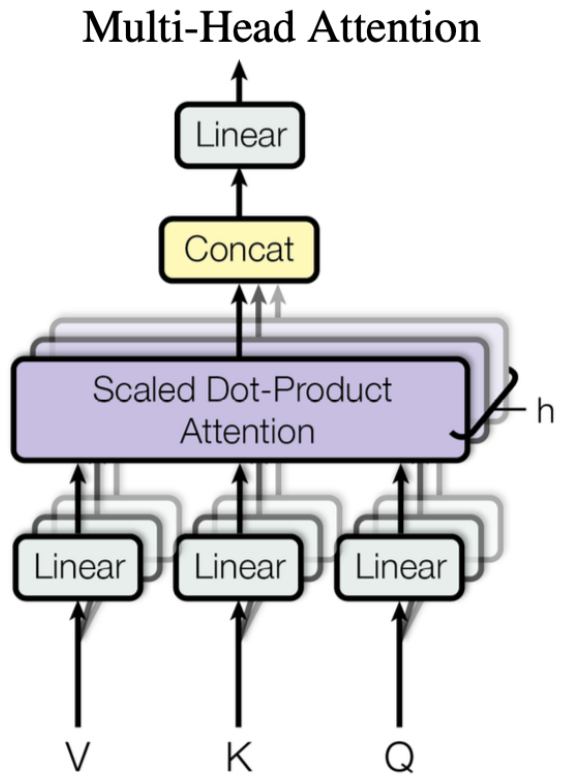
Elemento	Intuição	Como aparece na prática?
Queries (Q)	**Pergunta**: Cada palavra da frase está "fazendo uma pergunta" sobre quais outras palavras ela quer saber.	Vetor que representa a própria palavra, gerado por multiplicação do embedding pela matriz W_Q.
Keys (K)	**Chaves de um armário**: As demais palavras têm "chaves" que podem ser comparadas com as perguntas. Se uma chave for semelhante à pergunta, ela "abre" a porta para a informação relevante.	Vetor gerado pela mesma palavra, mas usando W_K.
Values (V)	**Conteúdo guardado nas portas**: Quando a porta abre, o que vem dentro é a informação que a palavra quer transmitir à pergunta.	Vetor resultante da multiplicação do embedding por W_V.

Analogia com o YouTube

- Query (Q): o vídeo que você está assistindo agora funciona como a "pergunta". Ele pede informações sobre quais outros vídeos podem ser relevantes.
- Key (K): Cada vídeo em sua lista de recomendações tem um “título + descrição” que atua como uma chave; o algoritmo compara essa chave com a pergunta para ver quão semelhante é.
- Value (V): Quando a chave corresponde, o valor é o próprio vídeo (ou seu link). Ele contém tudo o que você recebe: título, thumbnail, descrição, etc.

Resumo e Próximos Passos

- **Objetivo:** Permitir que cada token acesse e combine informação de **todas** as posições da sequência simultaneamente.
- **Queries (Q):** Vetores "perguntas" gerados a partir do próprio token.
- **Keys (K):** Vetores "chaves" que representam o conteúdo de cada token na mesma sequência.
- **Values (V):** Vetores contendo a informação real que será combinada.
- $A = \text{Softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right)V$
- **Próximos Passos**: Compreender **Multi-Head Attention** $\longrightarrow$ Repete o mecanismo em (h) sub-espaços diferentes e concatena os resultados, permitindo capturar múltiplas relações simultaneamente.



Multi-Head Attention. Fonte: Jeremy Jordan.