LAB P1-01: Implementação do Mecanismo de Self-Attention

1. Objetivo

O trabalho deve implementar a lógica central do mecanismo de **Scaled Dot-Product Attention**, conforme descrito no paper *"Attention Is All You Need"*. O foco é entender a transformação das matrizes de **Query (Q)**, **Key (K)** e **Value (V)**.

2. Requisitos Técnicos

A implementação deve ser feita sem o uso de bibliotecas de alto nível de Deep Learning (como camadas prontas do Keras ou PyTorch nn.MultiheadAttention).

- **Permitido:** Bibliotecas de álgebra linear (NumPy, Eigen, etc.).
- **Linguagem:** Livre escolha do aluno.
- **Entrega:** Exclusivamente via link de repositório Git (GitHub, GitLab ou Bitbucket).

3. Estrutura Esperada no Repositório

Para ser considerado um trabalho de nível profissional, o repositório deve seguir esta estrutura mínima:

1. **Código Fonte:** Implementação da função ou classe de Attention.
2. **Scripts de Teste:** Um arquivo (ex: test_attention.py) que valide a saída com um exemplo numérico simples.
3. **README.md:** Documentação clara contendo:
    - Instruções de como rodar o código.
    - Explicação breve de como a normalização ($\sqrt{d_k}$) foi aplicada.
    - Exemplo de input e o output esperado.

4. O que será cobrado (Critérios de Avaliação)

A correção seguirá os seguintes pesos:

Critério	**Descrição**	**Peso**
Logística de Matrizes	Cálculo correto do produto escalar QK^T e aplicação do Softmax.	40%

Scaling Factor	Divisão correta pela raiz quadrada da dimensão das chaves ($\sqrt{d_k}$).	20%
Engenharia de Código	Organização, nomes de variáveis semânticos e ausência de "código sujo".	20%
Documentação/Git	Histórico de commits coerente e README explicativo.	20%

---

5. Equação de Referência

O aluno deve obrigatoriamente basear sua implementação na fórmula:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

Nota para o aluno: Lembre-se que o Softmax deve ser aplicado em cada linha da matriz resultante do produto escalar.

Importante: Trabalho Individual