Soundverse - AI Assignment

Objective

We're looking for your implementation of key components from the transformer architecture: Self-Attention and Multi-Head Attention mechanisms.

Deliverables (More details below)

1. **Code Submission**:
    - Implement **Self-Attention** and **Multi-Head Attention** mechanisms in Python using `numpy`.
    - Please ensure your implementation follows the structure provided below.

2. **Video Explanation**:
    - Record a **Loom video** where you walk us through your code and explain your reasoning behind the implementation. Send it to sourabh@soundverse.ai with the title "Audio AI - (Your Name)" as the headline, and link to the github.
    - Cover key concepts like attention scores, softmax normalization, and the rationale behind multi-head attention splitting.

3. **Research Task**:
    - Read and familiarize yourself with diffusion models for music generation.
    - To get you started, we'd like you to read the following 3 research papers:

Task 1: Implement Self-Attention Mechanism

Task: Implement the Self-Attention Mechanism

- Your task is to implement the self-attention mechanism, which is a fundamental component of transformer models, widely used in natural language processing and computer vision tasks. The self-attention mechanism allows a model to dynamically focus on different parts of the input sequence when generating a contextualized representation.
- Your function should return the self-attention output as a numpy array.

Example:

Input:

```
import numpy as np

X = np.array([[1, 0], [0, 1]])
```

```
W_q = np.array([[1, 0], [0, 1]])
W_k = np.array([[1, 0], [0, 1]])
W_v = np.array([[1, 2], [3, 4]])

Q, K, V = compute_qkv(X, W_q, W_k, W_v)
output = self_attention(Q, K, V)

print(output)
```

Output:

```
# [[1.660477 2.660477]
#  [2.339523 3.339523]]
```

Reasoning:

The self-attention mechanism calculates the attention scores for each input, determining how much focus to put on other inputs when generating a contextualized representation. The output is the weighted sum of the values based on the attention scores.

Task 2: Implement Multi-Head Attention

Implement the multi-head attention mechanism, a critical component of transformer models. Given Query (Q), Key (K), and Value (V) matrices, compute the attention outputs for multiple heads and concatenate the results.

Example:

Input:

```
Q = np.array([[1, 0], [0, 1]]), K = np.array([[1, 0], [0, 1]]), V = np.array([[1, 0], [0, 1]]), n_heads = 2
```

Output:

```
[[1., 0.], [0., 1.]]
```

Reasoning:

Multi-head attention is computed for 2 heads using the input Q, K, and V matrices. The resulting outputs for each head are concatenated to form the final attention output.

Task 3: Read Top 3 Diffusion Model Papers for Music Generation:

Your interview will have questions from these papers

DiffRhythm: Blazingly Fast and Embarrassingly Simple End-to-End Full-Length Song Generation with Latent Diffusion
Published: March 5, 2025
This paper introduces DiffRhythm, the first latent diffusion-based song generation model capable of producing complete songs with synchronized vocals and instrumentals in a streamlined process. Notably, DiffRhythm can generate full-length songs up to 4 minutes and 45 seconds long in just 10 seconds. Hugging Face+1arXiv+1

QA-MDT: Quality-aware Masked Diffusion Transformer for Enhanced Music Generation
Published: May 24, 2024
This work presents a novel paradigm for high-quality music generation by incorporating a quality-aware training strategy. The proposed masked diffusion transformer (MDT) model demonstrates enhanced musicality and addresses issues related to low-quality datasets. arXiv+1OpenReview+1

Long-form Music Generation with Latent Diffusion
Published: April 16, 2024
This paper demonstrates that by training a generative model on long temporal contexts, it is possible to produce long-form music of up to 4 minutes and 45 seconds. The model utilizes a diffusion-transformer operating on a highly downsampled continuous latent representation and achieves state-of-the-art results in audio quality and prompt alignment. arXiv