Summary

- **Core Innovations**:
    - **Self-Attention**: Enables the model to assess how each word in a sequence relates to all other words during prediction.
    - **Positional Encoding**: Captures word order and sequential patterns, crucial because Transformers process all tokens simultaneously, unlike Recurrent Neural Networks (RNNs).
- **Decoder-Only Transformer Architecture**:
    - Composed of multiple identical **decoder blocks** stacked vertically.
    - Training involves pairing each input sequence with a target sequence that is **shifted forward by one token**, similar to RNN-based language models.
- **Decoder Block Components**:
    - Each decoder block consists of two main sub-layers: **Self-Attention** and a **Position-Wise Multilayer Perceptron (MLP)**.
    - The block processes input token embeddings (e.g., 6-dimensional vectors).
- **Self-Attention Mechanism**:
    - Relies on three sets of trainable parameters (tensors or matrices): **Query (Q)**, **Key (K)**, and **Value (V)**. These are derived by multiplying input embeddings with corresponding weight matrices (**W_Q, W_K, W_V**).
    - **Attention Scores**: Computed by taking the **dot product** of each query vector (**q_i**) with every key vector (**k_j**).
    - **Scaled Scores**: Attention scores are divided by the square root of the key vector's dimensionality (e.g., $\sqrt{6}$) to prevent gradients from becoming too small after softmax.
    - **Causal Mask**: Applied to the scaled scores to prevent tokens from attending to **future positions** in the sequence. This is essential for maintaining the **autoregressive nature** of language models, ensuring predictions rely only on previous and current inputs. Masked scores with `-inf` become zero after softmax.
    - **Attention Weights**: Produced by applying the **softmax function** to the masked scores.
    - **Output Vector (g_i)**: Computed as a **weighted sum of the Value vectors** using the attention weights.
    - **Interpretation of Q, K, V**: Query seeks information, Key is what other positions offer, and Value is the information selected and combined.

    - **Historical Context**: The concept of attention emerged before the Transformer, notably in 2014 from Dzmitry Bahdanau's work on RNNs for machine translation.
- **Position-Wise Multilayer Perceptron (MLP)**:
    - After self-attention, each output vector (**g_i**) is independently processed by an MLP, applying a sequence of transformations with learned parameters.
    - This component is often called a feedforward, dense, or fully connected layer, but it specifically uses **two weight matrices, two bias vectors, and a ReLU activation function**. The first linear transformation typically **expands the dimensionality** (e.g., 4 times) before compressing it back to the original embedding dimensionality.
- **Rotary Position Embedding (RoPE)**:
    - Addresses the Transformer's inherent lack of word order awareness.
    - Encodes positional information by **rotating pairs of adjacent dimensions** within the query and key vectors before attention computation.
    - A key benefit is its ability to **generalize effectively to sequences longer than those seen during training**.
    - The angle between any two rotated vectors encodes the distance between their positions.
    - Rotation is performed using **matrix multiplication** with a rotation matrix.
    - The rotation frequency varies across dimensions, allowing fine-grained local position information in early dimensions and coarse-grained global information in later dimensions.
    - **Value vectors do not need RoPE** because positional relationships are captured in the query-key alignment.
- **Multi-Head Attention**:
    - An enhanced version of self-attention that allows the model to **focus on multiple aspects of information simultaneously** (e.g., syntactic, semantic, long-range dependencies).
    - For each "head," there's a separate triplet of Q, K, V matrices.
    - Outputs from multiple heads are **concatenated** along the embedding dimension and then transformed by a **projection matrix (W_O)** to integrate information.
    - Modern LLMs can use up to 128 heads.
- **Residual Connections (Skip Connections)**:
    - **Essential for training deep neural networks** by solving the **vanishing gradient problem**.
    - The input of a layer is **added directly to its output** (y = f(x) + x), creating shortcuts in the gradient computation path.

    - Mathematically, they introduce additional terms in the gradient calculation, preventing it from vanishing to zero, even with small weights.
    - Each decoder block includes two residual connections.
- **Root Mean Square Normalization (RMSNorm)**:
    - Applies root mean square normalization to the input vector **before** it enters the self-attention layer and the position-wise MLP.
    - Calculates the **RMS** of the vector, divides each component by it, and then applies a trainable **scale factor (γ)** element-wise.
    - Primary purpose is to **stabilize training** by maintaining a consistent scale for layer inputs, preventing excessively large or small gradient updates.
- **Key-Value Caching**:
    - An optimization for **autoregressive inference** (generating tokens one at a time).
    - Avoids recalculating key and value matrices for previous tokens by **saving (caching)** them after they are computed once.
    - When a new token arrives, its key and value vectors are computed and appended to the cache. Query vectors are not cached as they depend on the current token.
    - RoPE is compatible with caching as new tokens simply take the next available position index.
- **Performance and Scale**:
    - The Transformer model implemented in the chapter achieved a perplexity of 55.19, which is better than the RNN's 72.23 (for comparable parameter counts).
    - The true strengths of Transformers become apparent at **larger scales** of model size, context length, and training data.

This chapter lays the foundational understanding for comprehending Large Language Models (LLMs), which are a specific type of Transformer.