**Phần 1: Xử lý đầu vào**

* **RAW Text (string)**: Văn bản thô dạng chuỗi.
* **TOKENIZER: SentencePiece BPE**: Chuyển văn bản thành token ID (t\_1, ..., t\_n).
* **Embedding: W\_e[t\_i] ∈ ℝ^(N × 768)**: Ánh xạ token thành vector embedding 768 chiều, tạo ma trận [N × 768].

**Phần 2: Các layer Transformer (Lặp lại 12 lần)**

* **Input H^(l-1) ∈ ℝ^(N × 768)**: Đầu vào layer là ma trận [N × 768].
* **RMSNorm**: Chuẩn hóa RMS.
* **Linear Q, K, V ∈ ℝ^(N × 768)**: Chiếu đầu vào thành Q, K, V.
* **Rotary Positional Embedding (RoPE)**: Áp dụng RoPE cho Q và K.
* **Split into 12 heads: nQ, K, V ∈ ℝ^(N × 64)**: Chia thành 12 head, mỗi head kích thước 64.
* **Attention = softmax(nQ K^T / √64) V**: Tính attention với hệ số scaling √64.
* **Concat heads ∈ ℝ^(N × 768)**: Ghép các head thành [N × 768].
* **Linear projection W\_o → AttnOut**: Chiếu tuyến tính output attention.
* **Residual: H' = H^(l-1) + AttnOut**: Cộng residual.
* **RMSNorm H' → H'**: Chuẩn hóa RMS.
* **Linear W\_1, W\_3 → W\_1 u ∈ ℝ^(N × 3072)**: Chiếu lên 3072 chiều.
* **SiLU(u)**: Áp dụng hàm SiLU.
* **Linear W\_2 → W\_1 v ∈ ℝ^(N × 3072)**: Chiếu thành v.
* **SwiGLU = SiLU(u) ⊙ v**: Kết hợp u và v.
* **Linear W\_4 → SwiGLU x W\_4 ∈ ℝ^(N × 768)**: Chiếu về [N × 768].
* **Residual: H^(l) = H' + FFNOut**: Cộng residual từ FFN.
* **Decision: t < 12?**: Lặp lại 12 lần.

**Phần 3: Xử lý đầu ra**

* **Final H^(12) ∈ ℝ^(N × 768)**: Output sau 12 layer.
* **Final RMSNorm → Y ∈ ℝ^(N × 768)**: Chuẩn hóa RMS cuối cùng.
* **Take h\_t ∈ ℝ^(768) (last row of Y)**: Lấy embedding token cuối.
* **Linear z → h\_t × W\_LM ∈ ℝ^(32000)**: Chiếu thành logits cho 32000 token.
* **Softmax → Probabilities**: Chuyển thành xác suất.
* **Sample / Argmax → Next Token ID**: Lấy token tiếp theo.