Lecture 6: Attention Mechanism and Transformers - II

COMP 5801H/4900A: Generative AI and LLMs

2026-01-22

Sriram Subramanian
Assistant Professor & Canada Research Chair, Carleton University
Faculty Affiliate, Vector Institute for Artificial Intelligence
Faculty Affiliate, Schwartz Reisman Institute for Technology and Society

Carleton University

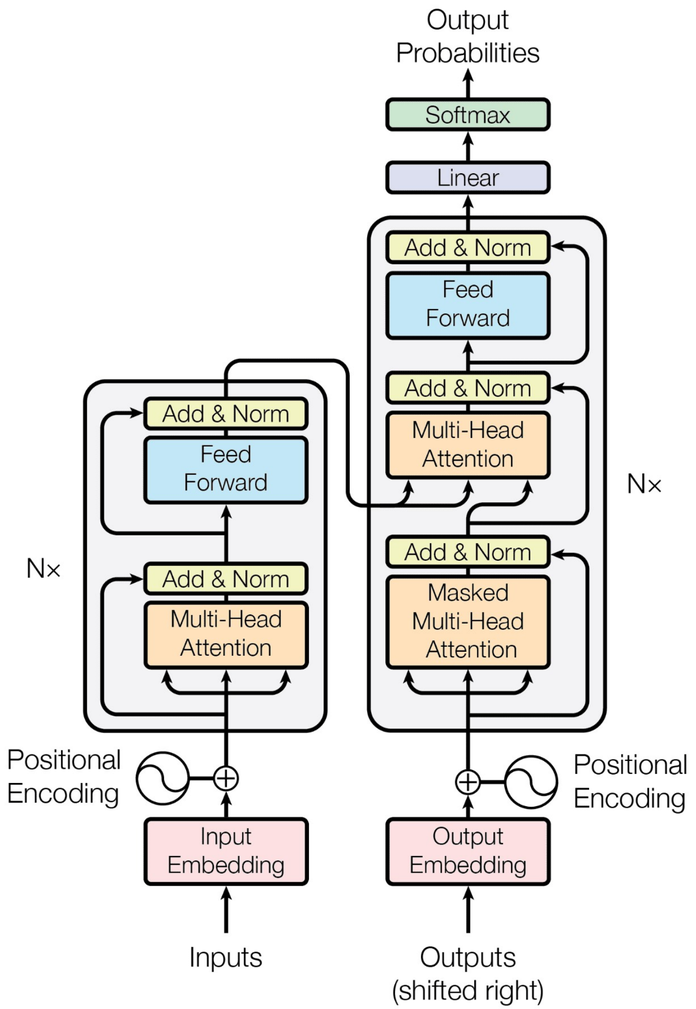
VECTOR INSTITUTE

SCHWARTZ REISMAN INSTITUTE FOR TECHNOLOGY AND SOCIETY

Lecture Outline

- **The Bottleneck Problem**
  - Why recurrence fails for long-range dependencies
  - The "Single Sticky Note" problem (Fixed-size context vectors)
- **Neural Machine Translation (NMT) with Attention**
  - The Bahdanau breakthrough: Scoring and Alignment
  - Query (Q), Key (K), and Value (V) analogies
- **From RNNs to Self-Attention**
  - The "Attention is All You Need" philosophy
  - Parallelization vs. Sequential Processing
- **The Transformer Architecture**
  - Positional Encoding, Multi-Head Attention, and Layer Norm
- **Scalability and the Generative Era**
  - How Attention enabled the rise of Large Language Models (LLMs)

Transformer Architecture



Credit: Vaswani et al. “Attention is all you need”

Transformer Encoder Block - I

- **The Input: Embeddings + Positional Encoding**
  - Static Meaning: Words are converted into 512-D vectors
  - Order Restoration: We add the Sinusoidal Positional Encoding ($E + PE$) so the model knows the sequence order
  - $X_{in} = \text{Embedding(tokens)} + PE$
- **The Core: Multi-Head Attention (MHA)**
  - The "Committee": X_{in} is projected into $Q, K,$ and V matrices across multiple heads
  - Relational Mapping: Each head calculates how much every word should "attend" to every other word
  - Aggregation: The results of all heads are concatenated and projected back to 512-D

Transformer Encoder Block - II

- **The Stability: Residuals & Layer Norm (Add & Norm)**
  - Residual Connection: Add the input of the layer to its output: X + Sublayer(X), prevents gradients from vanishing
  - Normalization: Layer Norm ensures the mean and variance of the activations stay stable, allowing for deeper stacking

- The Refinement: **Position-Wise Feed-Forward Network (FFN)**
  - Each token's vector is processed through two linear layers with a ReLU (or GeLU) activation. This allows the model to process the "new" context-aware information gathered by the attention heads

The Transformer Decoder Block - I

- **Masked Self-Attention (The "Blinders")**
  - Prevents the model from looking at "future" tokens during training.
  - Ensures that when predicting the i-th word, the model only uses words $1 \ldots i-1$.
- **Encoder-Decoder Attention (The "Bridge")**
  - This is where the Decoder "looks back" at the input.
  - Query (Q): Comes from the Decoder (the word we are currently trying to generate).
  - Key (K) & Value (V): Come from the final output of the Encoder.
  - Logic: "Based on what I've generated so far (Q), which parts of the original source sentence (K, V) should I focus on next?”
- **Linear & Softmax Head**
  - Converts the 512-D vector into a probability distribution over the entire vocabulary

How Transformers are Trained — Parallel Learning

- **Teacher Forcing**
  - During training, we don't give the model its own (potentially wrong) previous guess
  - Instead, we provide the Ground Truth (the actual correct sentence) as the input to the Decoder, shifted by one position
- **The Optimization Loop**
  - Forward Pass: The entire source sentence goes through the Encoder. The (shifted) target sentence goes through the Decoder
  - The Prediction: The model outputs a probability distribution for every word in the sequence at once
  - The Comparison: We compare the predicted probabilities against the actual correct words

How Transformers are Trained — Parallel Learning

- **The Math**

$$L = -\sum_{c=1}^{V} y_{o,c} \log(p_{o,c})$$

- $y_{o,c}$: A binary indicator (0 or 1) if class c is the correct label for observation o
- $p_{o,c}$: The predicted probability that observation o belongs to class c
- **What it does**
  - It heavily penalizes the model if it is confident about the wrong word
  - It rewards the model for putting a high probability (close to 1.0) on the correct word

Backpropagation — The Global Update

- Because the Transformer is a directed graph of differentiable operations, we use **Backpropagation Through Time (BPTT) effectively without the “Time"**
- **The Gradient Flow**
  - The loss is calculated for every word in the output sequence
  - Gradients are calculated using the **Chain Rule**, flowing backward from the Loss, through the Linear layer, the Decoder blocks, the Attention heads, and all the way back to the Encoder embeddings
- **Parameter Update**
  - The model uses an optimizer (typically **Adam**) to slightly nudge the weights (W^Q, W^K, W^V, etc.) in the direction that reduces the loss
  - Because of **Residual Connections**, the gradients can "skip" layers, reaching the earliest parts of the network without vanishing

From "Attention" to "Scale" — The Rise of LLMs

- **Solving the Sequential Constraint:** All words are processed in parallel
- **Constant Path Length (Long-Range Dependencies):** In an LLM, any word can "look" at any other word with a single operation
- **Scaling Laws: "More is Different"**
  - **Emergent Abilities**: Researchers discovered that as you increase the number of parameters and the amount of data (Scaling Laws), models suddenly gain abilities they were not explicitly trained for, like coding, reasoning, and zero-shot translation
  - **Attention's Role**: Attention is a highly expressive mechanism. Unlike the rigid gates of a GRU, Attention can learn almost any relationship between data points if given enough scale

Question: “We know that Transformers allow for parallel processing of all words. However, as we scale to models with millions of tokens in context, what becomes the new computational bottleneck compared to the old RNN approach?”

Conclusion

- **The Past**: The limitations of Recurrence and the Bahdanau bridge
- **The Present**: The Q, K, V mechanics, Layer Norms, and Residuals that make the Transformer work
- **The Future**: How these mechanics enabled the current explosion of Large Language Models