

A Short Treatise on Large Language Models (LLMs) with Mathematical Foundations

Focusing on Perplexity AI's Model Architecture and Components

Date: May 10, 2025

1. Introduction

Large Language Models (LLMs) like those powering Perplexity AI represent a class of deep neural networks designed to understand and generate human language. These models are built on the Transformer architecture and trained on massive corpora of text to learn statistical patterns of language. Perplexity AI's models, notably the Sonar series (e.g., llama-3.1-sonar-huge-128k-online), extend foundational architectures like Meta's LLaMA with optimizations for real-time inference, extended context windows, and integration of real-time knowledge.

This treatise presents a detailed mathematical and structural overview of such LLMs, covering the underlying architecture, training objectives, inference mechanisms, and recent innovations relevant to Perplexity's models.

2. Transformer Architecture: The Core of LLMs

The Transformer architecture, introduced by Vaswani et al. (2017), is the backbone of modern LLMs. It replaces recurrent and convolutional networks with self-attention mechanisms, enabling efficient parallel processing and long-range dependency modeling.

2.1 Input Representation

Given an input sequence of tokens $x = (x_1, x_2, \dots, x_n)$, each token x_i is mapped to an embedding vector $\mathbf{e}_i \in \mathbb{R}^d$, where d is the embedding dimension. The input embedding matrix $E \in \mathbb{R}^{V \times d}$ (with vocabulary size V) is used such that:

$$\mathbf{e}_i = E[x_i]$$

Positional encodings $\mathbf{p}_i$ are added to incorporate token order:

$$\mathbf{z}_i^{(0)} = \mathbf{e}_i + \mathbf{p}_i$$

2.2 Self-Attention Mechanism

At each Transformer layer l , the self-attention computes context-aware representations by relating each token to every other token.

Define queries Q , keys K , and values V as linear projections of the input:

$$Q = \mathbf{Z}^{(l-1)} W_Q, K = \mathbf{Z}^{(l-1)} W_K, V = \mathbf{Z}^{(l-1)} W_V$$

where $W_Q, W_K, W_V \in \mathbb{R}^{d \times d_k}$ are learned weight matrices, and d_k is the dimension of queries and keys.

The scaled dot-product attention for token i is:

$$\text{Attention}(Q_i, K, V) = \sum_{j=1}^n \alpha_{ij} V_j$$

with attention weights:

$$\alpha_{ij} = \frac{\exp\left(\frac{Q_i \cdot K_j}{\sqrt{d_k}}\right)}{\sum_{m=1}^n \exp\left(\frac{Q_i \cdot K_m}{\sqrt{d_k}}\right)}$$

This computes a weighted sum of value vectors, where weights reflect relevance.

2.3 Multi-Head Attention

To capture diverse relationships, multiple attention heads h are computed in parallel:

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \dots, \text{head}_h) W_O$$

where each head is:

$$\text{head}_i = \text{Attention}(Q W_Q^{(i)}, K W_K^{(i)}, V W_V^{(i)})$$

and $W_O \in \mathbb{R}^{hd_k \times d}$ is the output projection.

2.4 Feedforward Network

Following attention, a position-wise feedforward network applies two linear transformations with a nonlinear activation (usually GELU):

$$\text{FFN}(\mathbf{z}) = \max(0, \mathbf{z}W_1 + b_1)W_2 + b_2$$

where $W_1 \in \mathbb{R}^{d \times d_{\text{ff}}}$, $W_2 \in \mathbb{R}^{d_{\text{ff}} \times d}$, and d_{ff} is the feedforward dimension.

2.5 Residual Connections and Layer Normalization

Each sub-layer (attention, FFN) is wrapped with residual connections and layer normalization:

$$\mathbf{z}^{(l)} = \text{LayerNorm}(\mathbf{z}^{(l-1)} + \text{SubLayer}(\mathbf{z}^{(l-1)}))$$

This stabilizes training and allows deeper networks.

3. Training Objective: Language Modeling

LLMs are trained to model the probability distribution over token sequences. Given a sequence $\mathbf{x} = (x_1, \dots, x_n)$, the model estimates:

$$P(\mathbf{x}) = \prod_{t=1}^n P(x_t | \mathbf{x}_{<t})$$

The training objective minimizes the negative log-likelihood (cross-entropy loss):

$$\mathcal{L} = - \sum_{t=1}^n \log P_{\theta}(x_t | \mathbf{x}_{<t})$$

where θ are model parameters.

Perplexity, a key metric, is defined as:

$$\text{Perplexity} = \exp \left(\frac{1}{n} \sum_{t=1}^n -\log P_{\theta}(x_t | \mathbf{x}_{<t}) \right)$$

Lower perplexity indicates better predictive performance.

4. Extended Context Windows and Real-Time Information Integration

Perplexity AI's Sonar models extend the Transformer's context window up to **128K tokens**, enabling long document understanding and multi-turn dialogue retention.

4.1 Efficient Attention for Long Contexts

Standard self-attention scales as $O(n^2)$ with sequence length n , which is prohibitive for very long inputs. To handle 128K tokens, Perplexity employs optimized attention variants such as:

- **Sparse Attention:** Only attend to a subset of tokens based on locality or learned patterns.
- **Memory Compression:** Summarize distant context into compressed representations.
- **Segmented Attention:** Process inputs in overlapping chunks with cross-segment communication.

Mathematically, sparse attention modifies the attention mask $M \in \{0,1\}^{n \times n}$ to zero out irrelevant pairs:

$$\alpha_{ij} = 0 \text{ if } M_{ij} = 0$$

reducing computation.

4.2 Real-Time Knowledge Integration

Unlike static LLMs, Perplexity's models incorporate **online inference** with real-time web data retrieval. This involves:

- Querying external knowledge sources during inference.
- Conditioning generation on retrieved documents.

Formally, let R be retrieved relevant documents for query q . The model estimates:

$$P(x|q, R) = \prod_{t=1}^n P(x_t|x_{<t}, q, R)$$

This conditioning is implemented by concatenating retrieved text or embedding it into the context window.

5. Inference and Decoding

Given a prompt $x_{\{<t\}}$, the model generates the next token x_t by sampling from the predicted distribution $P_{\theta}(x_t | x_{\{<t\}})$.

Common decoding strategies include:

- **Greedy decoding:** Select token with highest probability.
- **Beam search:** Maintain multiple candidate sequences to optimize overall likelihood.
- **Sampling:** Draw tokens probabilistically to increase diversity.
- **Top-k / Top-p (nucleus) sampling:** Restrict sampling to top probable tokens or cumulative probability mass.

Mathematically, sampling from top-p nucleus with threshold p :

1. Sort tokens by probability $p_1 \geq p_2 \geq \dots$.
2. Select smallest set S such that:

$$\sum_{i \in S} p_i \geq p$$

3. Sample next token from S .

6. Model Variants and Sizes

Perplexity's Sonar models come in multiple sizes (small, large, huge) with varying parameters N (number of layers), d (embedding size), and h (attention heads). Larger models generally yield better performance but require more computation.

Model Size	Layers N	Embedding Dim d	Heads h	Context Window
Small	~24	1024	16	128K tokens
Large	~48	2048	32	128K tokens
Huge	~72	4096	64	128K tokens

7. Fine-Tuning and Alignment

Perplexity models are fine-tuned from base LLaMA checkpoints with additional training on curated datasets emphasizing:

- Real-time web data integration.
- Safety and ethical alignment.
- Task-specific capabilities (e.g., question answering, summarization).

Fine-tuning optimizes parameters θ via gradient descent on combined losses, including supervised learning and reinforcement learning with human feedback (RLHF).

8. API and Usage Structure

Perplexity exposes its models via an API compatible with OpenAI's chat completion schema, supporting:

- Message-based interactions with roles (user, assistant).
- Extended context windows for long conversations.
- Detailed token usage and cost reporting.

Example API request includes:

```
{
  "model": "llama-3.1-sonar-huge-128k-online",
  "messages": [
    {"role": "user", "content": "Explain quantum computing advances in 2025."}
  ],
  "max_tokens": 150
}
```

9. Summary

Perplexity's LLMs are advanced Transformer-based models built upon Meta's LLaMA architecture, enhanced for:

- Massive context windows (up to 128K tokens).
- Real-time knowledge integration via online retrieval.

- Scalable model sizes for diverse use cases.
- Fine-tuning for alignment, safety, and task specialization.

Mathematically, they implement self-attention, feedforward layers, and probabilistic language modeling with innovations in efficient long-context processing and dynamic knowledge conditioning.

References

- Vaswani et al., “Attention is All You Need,” NeurIPS 2017.
- Touvron et al., “LLaMA: Open and Efficient Foundation Language Models,” 2023.
- Perplexity AI Technical FAQ and API Documentation, 2025.
- [APIpie.ai](#) Model Overview and Usage Guide, 2025.

This treatise provides a foundational mathematical understanding of how models like Perplexity’s Sonar series operate, suitable for research, development, and academic study.

**