From Language Models to Large Language Models

October 23, 2025

1 Language Modeling Objectives

Language models provide a probabilistic description of text sequences. For a sequence $\mathbf{x} = (x_1, \ldots, x_T)$ drawn from vocabulary $\mathcal{V}$, the model assigns probability

$$p_\theta(\mathbf{x}) = \prod_{t=1}^{T} p_\theta(x_t \mid x_{<t}), \tag{1}$$

where $x_{<t}$ abbreviates $(x_1, \ldots, x_{t-1})$. Training maximizes the log-likelihood over corpus $\mathcal{D}$:

$$\mathcal{L}(\theta) = -\sum_{\mathbf{x} \in \mathcal{D}} \sum_{t=1}^{T} \log p_\theta(x_t \mid x_{<t}). \tag{2}$$

This objective coincides with minimizing cross-entropy between empirical and model distributions. The exponentiated negative average log-likelihood yields perplexity, a standard evaluation metric:

$$\mathrm{PPL}(\mathcal{D}) = \exp\left(-\frac{1}{|\mathcal{D}|} \sum_{\mathbf{x} \in \mathcal{D}} \frac{1}{T} \log p_\theta(\mathbf{x})\right). \tag{3}$$

Lower perplexity denotes better predictive power. In practice, large-scale training augments maximum likelihood with regularization such as dropout, label smoothing, or gradient clipping. Self-supervised learning underpins language modeling: by masking or predicting tokens from context, models learn representations without manual annotation. Auxiliary tasks, for example next sentence prediction or contrastive sentence ordering, further enrich the objective landscape.

2 Evolution from N-gram to Transformer

2.1 Statistical n-gram Models

Classical n-gram language models approximate the conditional probability with a Markov assumption,

$$p(x_t \mid x_{1:t-1}) \approx p(x_t \mid x_{t-n+1:t-1}), \tag{4}$$

and estimate parameters through frequency counts. Techniques such as Laplace smoothing, Katz back-off, and interpolated Kneser–Ney mitigate sparsity, yet the fixed context window limits long-range dependencies and the parameter space grows exponentially with n.

2.2 Neural Language Models and RNN Family

Neural language models introduced distributed embeddings and nonlinear composition. Recurrent neural networks (RNNs) iteratively update a hidden state $\mathbf{h}_t = f_\theta(x_t, \mathbf{h}_{t-1})$ to summarize history. Long short-term memory (LSTM) networks and gated recurrent units (GRU) employ gating mechanisms—input, forget, and output gates—for stable gradient flow:

$$\mathbf{i}_t = \sigma(\mathbf{W}_i x_t + \mathbf{U}_i \mathbf{h}_{t-1}), \tag{5}$$
$$\mathbf{f}_t = \sigma(\mathbf{W}_f x_t + \mathbf{U}_f \mathbf{h}_{t-1}), \tag{6}$$
$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \tanh(\mathbf{W}_c x_t + \mathbf{U}_c \mathbf{h}_{t-1}). \tag{7}$$

RNNs capture longer contexts than n-gram models, but recurrence hinders parallelization and struggles with extremely long dependencies.

2.3 Attention and Transformers

The transformer architecture replaces recurrence with self-attention. For query, key, and value matrices $\mathbf{Q}$, $\mathbf{K}$, $\mathbf{V}$, scaled dot-product attention computes

$$\mathrm{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \mathrm{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_k}}\right)\mathbf{V}. \tag{8}$$

Multi-head attention, residual connections, and layer normalization enable deep transformers to model global dependencies while benefiting from parallel computation. Subsequent innovations—relative positional encodings, sparse or linear-time attention, and Mixture-of-Experts—extend sequence lengths and improve efficiency, paving the way for large-scale pretraining.

3 Autoregressive vs. Autoencoding Paradigms

3.1 Autoregressive Modeling

Autoregressive (AR) models factorize sequence likelihood left-to-right. Decoder-only transformers (GPT family) adopt causal masking so each token attends only to preceding tokens. Advantages include stable training, straightforward ancestral sampling, and natural compatibility with open-ended generation. Limitations arise from exposure bias between training and inference, as well as the absence of right-context during representation learning.

3.2 Autoencoding Modeling

Autoencoding (AE) models, exemplified by BERT, corrupt input tokens (via masking, deletion, or permutation) and train to reconstruct the original sequence. The masked language modeling (MLM) loss

$$\mathcal{L}_{\mathrm{MLM}} = -\mathbb{E}_{\mathbf{x},\mathbf{m}} \sum_{t \in \mathbf{m}} \log p_\theta(x_t \mid \mathbf{x}_{\setminus \mathbf{m}}) \tag{9}$$

encourages bidirectional context aggregation. AE models excel at understanding tasks—classification, span extraction, sentence similarity—but require encoder-decoder wrapping or iterative refinement for fluent generation.

3.3 Hybrid Approaches

Sequence-to-sequence transformers, span corruption (T5), masked sequence-to-sequence (MASS), and prefix language models bridge AR and AE paradigms. They encode full context while decoding autoregressively, unifying comprehension and generation capabilities.

4 GPT and BERT: Conceptual Comparison

4.1 Architectural Differences

GPT adopts a decoder-only stack with causal masks, enabling each block to attend to all previous tokens. Training uses the next-token prediction objective across massive web-scale corpora, often leveraging curriculum scheduling, adaptive optimizers, and large batch sizes guided by scaling laws. BERT leverages an encoder-only stack with bidirectional self-attention; its training pairs masked language modeling with next sentence prediction (NSP) or sentence order prediction (SOP) to capture inter-sentence coherence.

4.2 Downstream Utilization

GPT-style models are commonly deployed for generative tasks. Prompt design, in-context learning, instruction fine-tuning, and reinforcement learning from human feedback (RLHF) align GPT outputs with user intent and safety guidelines. Conversely, BERT-style encoders feed into lightweight classification heads or are fine-tuned end-to-end for QA, NER, and semantic similarity. Representation quality allows feature extraction for tasks with limited labeled data.

4.3 Scaling and Adaptation

The GPT lineage (GPT-3, GPT-4, PaLM, LLaMA) emphasizes scaling parameters, data, and compute. Enhancements include mixture-of-experts routing, retrieval-augmented inference, and tool integration. Encoder-based descendants (RoBERTa, DeBERTa, ELECTRA) refine the pretraining objective, employ larger corpora, or introduce disentangled attention. Span corruption models (T5) and retrieval-augmented systems (REALM) further adapt the AE paradigm for generation and knowledge-intensive applications.

5 Practical Considerations

- **Data governance:** Deduplication, quality filtering, and multilingual balance improve convergence and reduce memorization risk.
- **Optimization:** Mixed precision (FP16/bfloat16), gradient checkpointing, ZeRO partitioning, and pipeline parallelism are essential for large-scale training.
- **Evaluation and safety:** Comprehensive benchmarks (GLUE, SuperGLUE, MMLU, BIG-Bench) must be coupled with toxicity, bias, and hallucination assessments before deployment.

Further Reading

- Jurafsky and Martin. *Speech and Language Processing*. Chapter 3–12.
- Bengio et al. "A Neural Probabilistic Language Model." JMLR 2003.
- Vaswani et al. "Attention is All You Need." NeurIPS 2017.
- Devlin et al. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding." NAACL 2019.
- Kaplan et al. "Scaling Laws for Neural Language Models." 2020.