LLM 面试核心公式汇总

AI 助手

May 14, 2025

Contents

1 激活函数 (Activation Functions)

激活函数为神经网络引入非线性，使其能够学习更复杂的模式。

1.1 Sigmoid

主要用于二分类问题的输出层，或作为门控机制。

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

输出范围：$(0, 1)$

1.2 Tanh (双曲正切)

与 Sigmoid 类似，但输出范围是 $(-1, 1)$，通常在隐藏层中表现比 Sigmoid 好。

$$\tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} = 2\sigma(2x) - 1$$

输出范围：$(-1, 1)$

1.3 ReLU (Rectified Linear Unit)

现代神经网络中最常用的激活函数之一。

$$\mathrm{ReLU}(x) = \max(0, x)$$

输出范围：$[0, \infty)$

1.4 GeLU (Gaussian Error Linear Unit)

在 Transformer 模型中广泛使用。

$$\mathrm{GELU}(x) = x \cdot \Phi(x)$$

其中 $\Phi(x)$ 是标准正态分布的累积分布函数 (CDF)。近似计算公式：

$$\mathrm{GELU}(x) \approx 0.5x \left(1 + \tanh\left[\sqrt{\frac{2}{\pi}}\left(x + 0.044715x^{3}\right)\right]\right)$$

1.5 Swish

一个自门控激活函数。函数定义为 x 乘以其自身经过 Sigmoid 函数（并可能由一个可学习的参数 或固定值调节）的结果。与 ReLU 不同，Swish 处处可导，这在优化过程中可能更有利。

- 基本形式 ($\beta = 1$):

$$\mathrm{Swish}(x) = x \cdot \sigma(x) = \frac{x}{1 + e^{-x}}$$

- 带参数 β:

$$\mathrm{Swish}_{\beta}(x) = x \cdot \sigma(\beta x) = \frac{x}{1 + e^{-\beta x}}$$

1.6 Softmax

通常用于多分类问题的输出层，将原始分数（logits）转换为概率分布。对于一个向量 $\mathbf{z} = (z_1, z_2, \ldots, z_K)$：

$$\text{Softmax}(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \quad \text{for } i = 1, \ldots, K$$

输出 $\sum_{i=1}^{K} \text{Softmax}(z_i) = 1$，且每个元素 $\in (0, 1)$。

2 损失函数 (Loss Functions)

损失函数衡量模型预测值与真实值之间的差异。

2.1 均方误差 (Mean Squared Error, MSE)

常用于回归问题。

$$L_{\text{MSE}} = \frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2$$

其中 N 是样本数量，y_i 是真实值，$\hat{y}_i$ 是预测值。

2.2 交叉熵损失 (Cross-Entropy Loss)

常用于分类问题。

- 二分类交叉熵 (Binary Cross-Entropy, BCE):

$$L_{\text{BCE}} = -\frac{1}{N} \sum_{i=1}^{N} [y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i)]$$

  其中 $y_i \in \{0, 1\}$ 是真实标签，$\hat{y}_i$ 是模型预测样本为类别 1 的概率。

- 分类交叉熵 (Categorical Cross-Entropy): 对于单个样本，真实标签 one-hot 编码 $\mathbf{y} = (y_1, \ldots, y_K)$，模型预测概率 $\hat{\mathbf{y}} = (\hat{y}_1, \ldots, \hat{y}_K)$：

$$L_{\text{CE}} = -\sum_{k=1}^{K} y_k \log(\hat{y}_k)$$

2.3 DPO (Direct Preference Optimization) Loss

用于从人类偏好数据中微调 LLM，作为 RLHF 的一种替代方案。给定偏好数据 (x, y_w, y_l) (提示, 偏好回答, 非偏好回答):

$$L_{\text{DPO}}(\pi_\theta; \pi_{\text{ref}}) = -\mathbb{E}_{(x, y_w, y_l) \sim \mathcal{D}} \left[\log \sigma \left(\beta \log \frac{\pi_\theta(y_w|x)}{\pi_{\text{ref}}(y_w|x)} - \beta \log \frac{\pi_\theta(y_l|x)}{\pi_{\text{ref}}(y_l|x)} \right) \right]$$

或者等价地写成隐式奖励形式：

$$\hat{r}_\theta(x, y) = \beta \log \frac{\pi_\theta(y|x)}{\pi_{\text{ref}}(y|x)}$$

$$L_{\text{DPO}} = -\mathbb{E}_{(x, y_w, y_l) \sim \mathcal{D}} \left[\log \sigma \left(\hat{r}_\theta(x, y_w) - \hat{r}_\theta(x, y_l) \right) \right]$$

其中 π_θ 是当前模型策略，π_{ref} 是参考模型策略，β 是超参数，σ 是 Sigmoid 函数。

2.4 PPO (Proximal Policy Optimization) Loss

PPO 是一种常用的强化学习算法，通过限制策略更新的幅度来稳定训练过程，常用于 LLM 的对齐微调 (RLHF)。其核心目标是最大化一个替代目标函数，该函数会惩罚新策略与旧策略之间过大的差异。

给定状态 s_t，动作 a_t，旧策略 $\pi_{\theta_{\text{old}}}(a_t|s_t)$，当前策略 $\pi_\theta(a_t|s_t)$，以及优势函数估计 $\hat{A}_t$：定义概率比率 (probability ratio):

$$r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}$$

PPO 的裁剪替代目标函数 (Clipped Surrogate Objective) $L^{\text{CLIP}}(\theta)$ 通常是最大化的目标，因此其对应的损失函数是该目标的负值：

$$L_{\text{PPO}}^{\text{CLIP}}(\theta) = -\mathbb{E}_t \left[\min \left(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t \right) \right]$$

其中:

- $\mathbb{E}_t[\dots]$ 表示对所有时间步（或 mini-batch 中的样本）取期望。
- $\hat{A}_t$ 是在时间步 t 的优势函数估计值，表示采取动作 a_t 相对于平均动作的好坏程度。
- $r_t(\theta)$ 是当前策略 π_θ 和旧策略 $\pi_{\theta_{\text{old}}}$ 对动作 a_t 的概率比。
- ϵ 是一个小的超参数（例如 0.1 或 0.2），用于定义裁剪范围 $\text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)$，它将 $r_t(\theta)$ 限制在 $[1-\epsilon, 1+\epsilon]$ 区间内。

在实际应用中，完整的 PPO 损失函数通常还包括一个价值函数损失项（用于更新价值网络，以更好地估计 $\hat{A}_t$）和一个熵奖励项（以鼓励探索）。

3 正则化 (Regularization)

正则化用于防止模型过拟合。

3.1 L1 正则化 (Lasso Regression)

向损失函数添加权重的绝对值之和。

$$L_{\text{total}} = L_{\text{original}} + \lambda \sum_{j=1}^{M} |w_j|$$

其中 M 是权重数量，λ 是正则化强度。

3.2 L2 正则化 (Ridge Regression / Weight Decay)

向损失函数添加权重的平方和。

$$L_{\text{total}} = L_{\text{original}} + \frac{\lambda}{2} \sum_{j=1}^{M} w_j^2$$

3.3 Dropout

训练时，以概率 p 随机将一些神经元的输出置为 0。推断时，通常关闭 Dropout 并将权重乘以 $(1-p)$（如果使用 Inverted Dropout 则推断时无需操作）。

4 Transformer 核心组件

Transformer 是现代 LLM 的基础架构。

4.1 Scaled Dot-Product Attention

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

其中 Q: Queries, K: Keys, V: Values, d_k: Key/Query 的维度。

4.2 Multi-Head Attention

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \ldots, \text{head}_h)W^O$$

其中每个头 head_i 是：

$$\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$

W_i^Q, W_i^K, W_i^V 是特定于头的投影矩阵，W^O 是输出投影矩阵。

4.3 Positional Encoding (Sinusoidal)

提供序列中 Token 的位置信息。

$$PE_{(pos,2i)} = \sin(pos/10000^{2i/d_{\text{model}}})$$
$$PE_{(pos,2i+1)} = \cos(pos/10000^{2i/d_{\text{model}}})$$

其中 pos: 位置, i: 维度索引, d_{model}: 模型隐藏层维度。

4.4 Layer Normalization

对每个样本的特征进行归一化。对于隐藏层向量 $\mathbf{x} = (x_1, \ldots, x_D)$：

$$\mu_j = \frac{1}{D}\sum_{i=1}^{D} x_i$$
$$\sigma_j^2 = \frac{1}{D}\sum_{i=1}^{D}(x_i - \mu_j)^2$$
$$\text{LayerNorm}(\mathbf{x})_i = \gamma_i \frac{x_i - \mu_j}{\sqrt{\sigma_j^2 + \epsilon}} + \beta_i$$

γ, β 是可学习参数，ϵ 是小常数。

4.4.1 RMS Norm (Root Mean Square Layer Normalization)

Layer Normalization 的简化版本，移除了均值中心化。对于输入向量 $\mathbf{x} = (x_1, \ldots, x_d)$：

$$\text{RMS}(\mathbf{x}) = \sqrt{\frac{1}{d}\sum_{i=1}^{d} x_i^2}$$

$$\text{RMSNorm}(\mathbf{x})_i = \frac{x_i}{\text{RMS}(\mathbf{x}) + \epsilon} \cdot g_i$$

其中 g_i 是可学习的增益参数。

4.5 Feed-Forward Network (FFN) / Position-wise FFN

Transformer 块中的子层，包含两个线性变换加一个激活，是 Transformer 参数量最大的模块。

$$\text{FFN}(x) = \text{Activate}(xW_1 + b_1)W_2 + b_2$$

'Activate' 通常是 ReLU 或 GeLU。

5 门控线性单元及其变体 (Gated Linear Units and Variants)

GLU 通过门控机制动态控制信息流，常用于 Transformer 的 FFN 层。其核心思想是将输入通过两个线性变换，其中一个变换的结果经过 Sigmoid 函数作为门，与另一个线性变换的结果进行逐元素相乘。

5.1 门控线性单元 (GLU - Gated Linear Unit) 基础

原始 GLU 定义：

$$\text{GLU}(X, W, V, b, c) = (XW + b) \odot \sigma(XV + c)$$

其中 σ 是 Sigmoid 函数, $\odot$ 是逐元素乘法。

5.2 Transformer FFN 中的 GLU 变体

通用形式（通常省略偏置 b_1, b_2 以简化表示，并在 W_3 后添加）：

$$\text{FFN}_{\text{GLU}}(x) = (\phi(xW_1) \odot (xW_2))W_3$$

其中 $x \in \mathbb{R}^{d_{model}}$, $W_1, W_2 \in \mathbb{R}^{d_{model} \times d_{ff}}$, $W_3 \in \mathbb{R}^{d_{ff} \times d_{model}}$。$\phi$ 是一个激活函数。

5.2.1 GeGLU (Gaussian Error Linear Unit Gated Linear Unit)

$\phi = \text{GeLU}$

$$\text{FFN}_{\text{GeGLU}}(x) = (\text{GeLU}(xW_1) \odot (xW_2))W_3$$

5.2.2 SwiGLU (Swish Gated Linear Unit)

$$\text{SwiGLU}(x) = (xW_1) \odot Swish(xW_2)$$

5.2.3 ReGLU (ReLU Gated Linear Unit)

$\phi = \text{ReLU}$

$$\text{FFN}_{\text{ReGLU}}(x) = (\text{ReLU}(xW_1) \odot (xW_2))W_3$$

6 评估指标 (Evaluation Metrics)

6.1 Perplexity (PPL)

衡量语言模型好坏，越低越好。对于序列 $W = w_1, \ldots, w_N$:

$$\text{PPL}(W) = P(w_1, \ldots, w_N)^{-\frac{1}{N}}$$

或基于交叉熵：

$$\text{PPL}(W) = \exp\left(\text{CrossEntropyLoss}\right) = \exp\left(-\frac{1}{N}\sum_{i=1}^{N}\log P(w_i|w_1, \ldots, w_{i-1})\right)$$

7 优化器相关 (Optimizer Related)

7.1 SGD with Momentum

v_t 是动量项，γ 是动量系数，η 是学习率。

$$v_t = \gamma v_{t-1} + \eta \nabla_\theta L(\theta_{t-1})$$
$$\theta_t = \theta_{t-1} - v_t$$

7.2 Adam (Adaptive Moment Estimation) - 核心思想

$g_t = \nabla_\theta L(\theta_t)$ 是梯度。

- 一阶矩 (动量): $m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t$
- 二阶矩 (平方梯度): $v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2$
- 偏差修正: $\hat{m}_t = \frac{m_t}{1-\beta_1^t}, \quad \hat{v}_t = \frac{v_t}{1-\beta_2^t}$
- 参数更新: $\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t}+\epsilon}\hat{m}_t$