Lecture 1

谢丹
清华大学数学系

2025/08

Large Language Models (LLMs) in Research

LLMs have become increasingly important tools for scientific research:

- Often the most efficient way to learn a new subject
- Writing assistance: editing, proofreading, summarizing
- Code generation: Sage, Mathematica, LaTeX, Python, etc.
- Research agents: Answer questions using tools and resources
- Data analysis and interpretation
- Computation, theorem proof using reasoning feature.

LLM Evolution: GPT-2 to Present

2019-2020

- GPT-2 (1.5B) → GPT-3 (175B)
- Few-shot learning emerges
- T5 unifies text tasks

2021-2022

- GPT-3.5 powers ChatGPT
- PaLM (540B) advances reasoning
- Open models: BLOOM, OPT

2023-2024

- GPT-4: Multimodal
- LLaMA spurs open ecosystem
- Claude 3, Gemini compete

Key Trends

- Scale: 1.5B → ∼1T params
- Access: Proprietary vs open
- Capability: Text → multimodal

Popular LLM Platforms in 2025

- Current LLM landscape:

Closed weights	**Open weights**
OpenAI: ChatGPT	DeepSeek: DeepSeek
Anthropic: Claude	Alibaba: Qwen
Google: Gemini	Zhipu: GLM
xAI: Grok	Moonshot: Kimi
	Meta: LLaMA

- Models are updated frequently with significant capability improvements (especially since 2025): larger context window, reasoning capabilities.
- The performance gap between open and closed models has narrowed considerably
- Many interesting open-weight models available on HuggingFace

Recent Advances in LLMs

- Since 2025, most major LLMs have introduced advanced reasoning capabilities
- Dramatically improved performance on:
  - Complex mathematical problems
  - Physics and scientific reasoning
  - Challenging coding tasks
- LLMs are fundamentally machine learning models
- Our focus will be on the underlying mathematical foundations

Core Components of Machine Learning

The basic ingredients for training ML models:

1. **Model**: Typically probabilistic - reflects the probabilistic nature of reality
2. **Data**: Represented as vectors, matrices, or tensors
3. **Training**: Optimization process to find function minima
4. **Inference**: Making predictions on new data

Machine learning essentially involves careful parameter tuning!

Machine Learning Applications

ML methods can solve diverse problems:

1. Regression analysis (linear and nonlinear curve fitting)
2. Classification tasks
3. Clustering problems
4. Natural language processing:
   - Translation
   - Text generation

Learning Paradigms

- Supervised learning: Regression and classification
- Unsupervised learning: Clustering
- Generative AI: Text generation models

Probability Fundamentals

Basic Probability Concepts

A probability model is described by a density function $p(x)$ satisfying:

$$\int p(x)\, dx = 1$$

For higher dimensions, we have joint probability $p(x, y)$ and:

- Marginal density: $p(x) = \sum_y p(x, y)$
- Conditional probability: $p(x|y)$ or $p(y|x)$

The fundamental relation:

$$\boxed{p(x, y) = p(x|y)p(y)}$$

Relevance to Physics

- Quantum mechanics
- Statistical physics

Statistical distribution

$$\rho(p, q) = \frac{\exp(-E(p, q))}{Z}$$

Such as Ising model, etc.

Probability Characteristics

Key quantities to characterize a probability distribution:

1. **Mean** (expected value):

$$\mu = \mathbb{E}[x] = \int x\, p(x)\, dx$$

2. **Variance** (spread around mean):

$$\sigma^2 = \mathbb{E}[(x-\mu)^2] = \int (x-\mu)^2 p(x)\, dx$$

Information Theory Concepts

Entropy

Measure of uncertainty:

$$H(x) = -\sum p(x) \ln p(x)$$

Kullback-Leibler Divergence

Important in machine learning:

$$\boxed{KL(q\|p) = -\int p(x) \ln \frac{q(x)}{p(x)} dx}$$

1. $KL(q\|p) \geq 0$ (Non-negativity)
2. $KL(q\|p) = 0 \iff p(x) = q(x)$ (Identity)

Important Probability Distributions

1. **Gaussian (Normal) Distribution**:

$$p(x|\mu, \sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

   Mean μ, variance σ^2

2. **Bernoulli Distribution** (discrete):

$$\mathrm{Ber}(x|\mu) = \mu^x (1-\mu)^{1-x}$$

   - $P(x=0) = 1-\mu$, $P(x=1) = \mu$
   - Mean μ, variance $\mu(1-\mu)$

Multivariate Distributions

Multivariate Gaussian

$$P(\mathbf{x}|\boldsymbol{\mu}, \Sigma) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T\Sigma^{-1}(\mathbf{x}-\boldsymbol{\mu})\right)$$

Mean vector $\boldsymbol{\mu}$, covariance matrix Σ

Categorical Distribution

For K classes:

$$P(t=i) = p_i \quad (i = 1, \ldots, K), \quad \sum_{i=1}^{K} p_i = 1$$

Compact representation:

$$P(\mathbf{t}) = \prod_{i=1}^{K} p_i^{t_i}$$

Bayesian Perspective

Machine learning models often begin with a parameterized probability model. Treating parameters $\mathbf{w}$ as random variables:

$$p(\mathbf{w}|\mathbf{x}) = \frac{p(\mathbf{x}|\mathbf{w})p(\mathbf{w})}{p(\mathbf{x})}$$

- $p(\mathbf{w})$: Prior probability
- $p(\mathbf{w}|\mathbf{x})$: Posterior probability

Maximum Likelihood Estimation:

$$\mathbf{w}^* = \arg\max_{\mathbf{w}} \prod_{i=1}^{N} p(x_i|\mathbf{w})$$

Linear Regression

Probabilistic Model

$$P(y|\mathbf{x}, \mathbf{w}, \sigma^2) = \mathcal{N}(y|\mathbf{w}^T\mathbf{x}, \sigma^2)$$

Negative log-likelihood gives the loss function:

$$\boxed{J(\mathbf{w}) = \sum_{n=1}^{N}(y_n - \mathbf{w}^T\mathbf{x}_n)^2}$$

Regularization

To prevent overfitting:

$$E(\mathbf{w}) = \sum_{n=1}^{N}(y_n - \mathbf{w}^T\mathbf{x}_n)^2 + \lambda\|\mathbf{w}\|_2^2$$

λ: Hyperparameter controlling regularization strength

Logistic Regression (Classification)

Binary Classification Model

$$P(t|\mathbf{x}, \mathbf{w}) = \sigma(\mathbf{w}^T\mathbf{x})^t(1 - \sigma(\mathbf{w}^T\mathbf{x}))^{1-t}$$

where $\sigma(x) = \frac{1}{1+\exp(-x)}$ is the sigmoid function

Loss Function

Negative log-likelihood:

$$J(\mathbf{w}) = -\sum_{n=1}^{N} \left[t_n \log \sigma(\mathbf{w}^T\mathbf{x}_n) + (1 - t_n) \log(1 - \sigma(\mathbf{w}^T\mathbf{x}_n))\right]$$

Multiclass Classification

Softmax Regression

Probability for class i:

$$p_i = \frac{\exp(\mathbf{w}_i^T \mathbf{x})}{\sum_{j=1}^{K} \exp(\mathbf{w}_j^T \mathbf{x})}$$

- Generalization of logistic regression
- Similar loss function derived via maximum likelihood
- Uses cross-entropy loss for optimization

模型训练流程

参数确定方法

给定模型后，确定未知参数$\mathbf{w}$的步骤:

1. **数据**：训练样本 (X, t)，其中：
   - X：输入特征（如前$n-1$个词语）
   - t：目标输出（如第n个词语）
2. **损失函数** $E(y(\mathbf{w}; X), t)$：
   - 衡量模型预测y与真实值t的差异
   - 概率模型通常导出特定的损失函数形式
3. **参数优化**：
   - 寻找使E最小的$\mathbf{w}$
   - 最优参数使模型预测最接近真实数据

梯度下降法

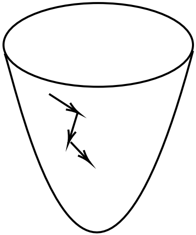
优化核心方法

解决复杂函数极值问题的基本方法:

- 牛顿法
- 梯度下降法（更常用）

$$E(\mathbf{w}_0 + \delta\mathbf{w}) = E(\mathbf{w}_0) + \nabla E(\mathbf{w}_0)^T \delta\mathbf{w} + \mathcal{O}(\|\delta\mathbf{w}\|^2)$$

取步长 $\delta\mathbf{w} = -\eta\nabla E(\mathbf{w}_0)$，保证函数值下降



优化算法实现

基本流程

- 初始化：随机选取参数$\mathbf{w}_0$
- 迭代更新：
  1. 计算梯度$\nabla E(\mathbf{w}_k)$
  2. 更新参数：$\mathbf{w}_{k+1} = \mathbf{w}_k - \eta \nabla E(\mathbf{w}_k)$

数学挑战

- 收敛速度: improvement: momentum based method
- 数值稳定性
- 局部极小值与全局极小值
- 参数λ和学习速率η的选择

大规模优化挑战

主要困难

- **参数量巨大**：百万、十亿甚至万亿级别参数
- **数据量庞大**：海量训练样本

现代机器学习成功要素

- **高效矩阵运算**：训练过程主要依赖矩阵操作
- **计算硬件进步**：
  - GPU加速
  - 并行算法优化
  - 专用张量处理单元

Stochastic Gradient Descent (SGD)

Key Idea

Instead of computing the full gradient using all training data, SGD:

- Uses small random subsets (batches) of data
- Computes gradient estimates from these batches
- Updates parameters more frequently

Batch Size

- **Batch size**: Number of samples in each subset
- Common choices:
  - Small batches (32-256 samples): Faster updates, noisier gradients
  - Large batches: Smoother gradients, more memory needed
- One epoch = complete pass through all batches

Advantages

- Faster convergence per computation time
- Better escape from local minima
- Enables training on large datasets

模型推断(Inference)

预测阶段

参数固定后，可进行两类预测：

1. **回归预测**：
   - 给定新数据x_{n+1}
   - 计算预测值$y_{n+1} = f(\mathbf{w}^*, x_{n+1})$
2. **分类预测**：
   - 计算类别概率$\sigma(\mathbf{w}^* x_{n+1})$
   - 选择最大概率对应的类别

贝叶斯方法

可进一步提供预测的不确定性估计

核心总结

机器学习训练过程本质上就是

参数的优化过程！

Remark: Renormalization group flow