目　录

基础知识

基础知识

AI 基础知识概览

本文整理了大语言模型（LLM）及其核心概念、架构和工作原理，适合初学者快速理解 AI 系统。

一、底层引擎：大语言模型（LLM）

核心定义与架构

- LM 全称：Large Language Model（大语言模型），简称大模型
- 底层架构：基于 Transformer（2017 年 Google 提出的 Attention 机制）
- 工作原理：通过预测下一个最可能的词生成文本

发展里程碑

时间	事件	意义
2017 年	Transformer 架构提出	奠定大模型技术基础
2022 年底	GPT-3.5 发布	首个达到可用级别的大模型
2023 年3月	GPT-4 发布	大幅提升 AI 能力上限
2023 年后	Claude、Gemini 等模型涌现	AI 赛道竞争加剧，OpenAI AI 技术多方竞争

二、数据处理单元：Token

核心特性

- 定义：文本的最小处理单元，通过 Tokenizer 分词
- 编码过程：文本 → Token → Token ID（数字表示）
- 解码过程：Token → 文本

Token 与语言单位关系

语言单位	与 Token 的关系	示例
中文词语	可能拆分	“工作坊” → “工作” + “坊”

语言单位	与 Token 的关系	示例
英文单词	常见单词对应 1 个 Token	"hello" → 1 Token
复杂单词	可能拆分	"helpful" → "help" + "ful"
特殊符号	可能多个 Token 表示	→ 3 Token

Token 参考量

- 1 Token ≈ 0.75 个英文单词
- 1 Token ≈ 1.5-2 个中文字符
- 40 万 Token ≈ 30 万英文单词或 60-80 万汉字

OpenAI Tokenizer 工具

- 网址：https://platform.openai.com/tokenizer
- 功能：
  - 输入文本可直观查看对应 Token
  - 支持不同模型 Token 计算（如 GPT-3、GPT-4、GPT-5）
  - 帮助理解文本长度和 Context Window 消耗
- 用途：
  - 预测 Token 消耗，控制上下文容量
  - 优化 Prompt，避免超过模型限制
  - 调试模型输入输出

三、记忆单元：Context

核心概念

- 定义：模型每次处理任务时能访问的信息总和，类似“临时记忆”
- 组成部分：用户问题、对话历史、工具调用、System Prompt 等
- 容量限制：由 Context Window（上下文窗口）定义

主流模型 Context Window 对比

模型	Context Window（Token）	约合汉字数量
GPT-5.4	105 万	~157.5 万
Gemini 3.1 Pro	100 万	~150 万
Claude Opus 4.6	100 万	~150 万

突破 Context Window 限制

- RAG 技术（检索增强生成）：通过检索相关文档片段，降低 Token 消耗

四、指令交互：Prompt

定义与分类

- Prompt：给 AI 的任务指令，决定输出内容
- 分类：
  - User Prompt：用户具体任务，如“写一首诗”
  - System Prompt：系统设定规则，如“保持幽默风格”

Prompt 工程（Prompt Engineering）

- 核心原则：清晰、具体、明确
- 提示设计技巧：
  - 避免模糊指令
  - 拆分复杂任务
  - 结合上下文提供信息

五、外部能力扩展：Tool

核心作用

- 定义：模型调用外部工具或接口，实现能力扩展
- 工作流程：
  i. 用户输入或平台转发

ii. 模型生成工具调用指令

iii. 工具执行并返回结果

iv. 模型生成最终输出

六、工具标准化：MCP

- 全称：Model Context Protocol（模型上下文协议）
- 作用：统一工具接口，解决多平台调用不一致问题
- 典型示例：OpenAI、Anthropic、Google 各自有接入规范，通过 MCP 统一

七、命令行工具（CLI）

核心概念

- CLI（Command-Line Interface） 是通过命令行与 AI 或工具交互的方式
- 常用于：
  - 快速调用模型功能
  - 自动化任务
  - 与脚本或系统集成
- 优势：
  - 高效轻量，不依赖图形界面
  - 可组合管道命令完成复杂任务
  - 适合开发者、数据科学家和运维场景

典型 CLI 示例

- OpenAI CLI：调用模型生成、管理 API keys
- Gemini CLI / Claude CLI：调度 Agent 执行任务
- Codex CLI：编程辅助和代码生成

八、自主执行系统：Agent

- 定义：能够自主决策、调用工具、完成任务的系统
- 核心能力：多步骤推理、工具选择、流程控制
- 代表产品：Claude Code、Codex、Gemini CLI 等
- 典型模式：React、Plan and Execute

九、任务定制：Agent Skill

- 定义：给 Agent 的能力模块，包含任务规则和执行步骤
- 核心功能：
  - 名称与描述
  - 任务目标
  - 执行步骤、判断规则、输出格式
- 技术实现：
  - Markdown 文档管理
  - 存放于专用目录，方便调用
  - 可附加用户问题映射规则，实现智能指令

十、概念体系关系

LLM（大脑） → Token（数据单元） → Context（记忆空间） → Prompt（交互指令） → Tool（外部能力）→ MCP（工具标准化） → CLI（命令行工具） → Agent（决策系统） → Agent Skill（任务定制）

补充说明

- Transformer 架构奠定基础
- Token 化处理文本，是模型理解的核心
- CLI 是现实场景中操作 AI 的重要接口
- Agent 与 Skill 扩展了 LLM 的实际应用能力
- RAG 技术可突破 Context Window 限制，提高信息调用效率

> *本文整理了 AI 核心概念、工作流程、扩展模块、CLI 工具及历史发展，便于快速入门和实践。*