**调研（偏向于理论）**

**如何使多模态大模型具备推理能力？**

**问题拆解：**

**1.多模态大模型**

**1.1：什么是多模态大模型**

**1.2：与单模态大模型的对比**

**1.3：多模态大模型的应用**

**2.如何让LLM具有推理能力**

**2.1：transfomer核心与自注意力机制**

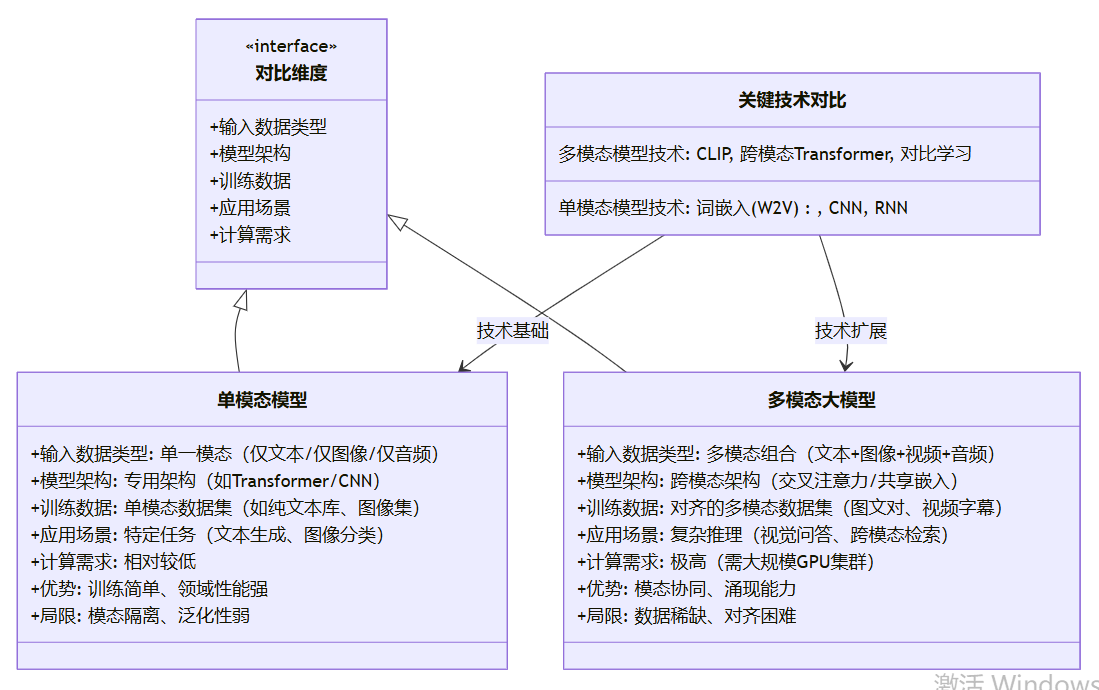
**2.2：prefill decoding KV\_Cache**

1. **多模态大模型**

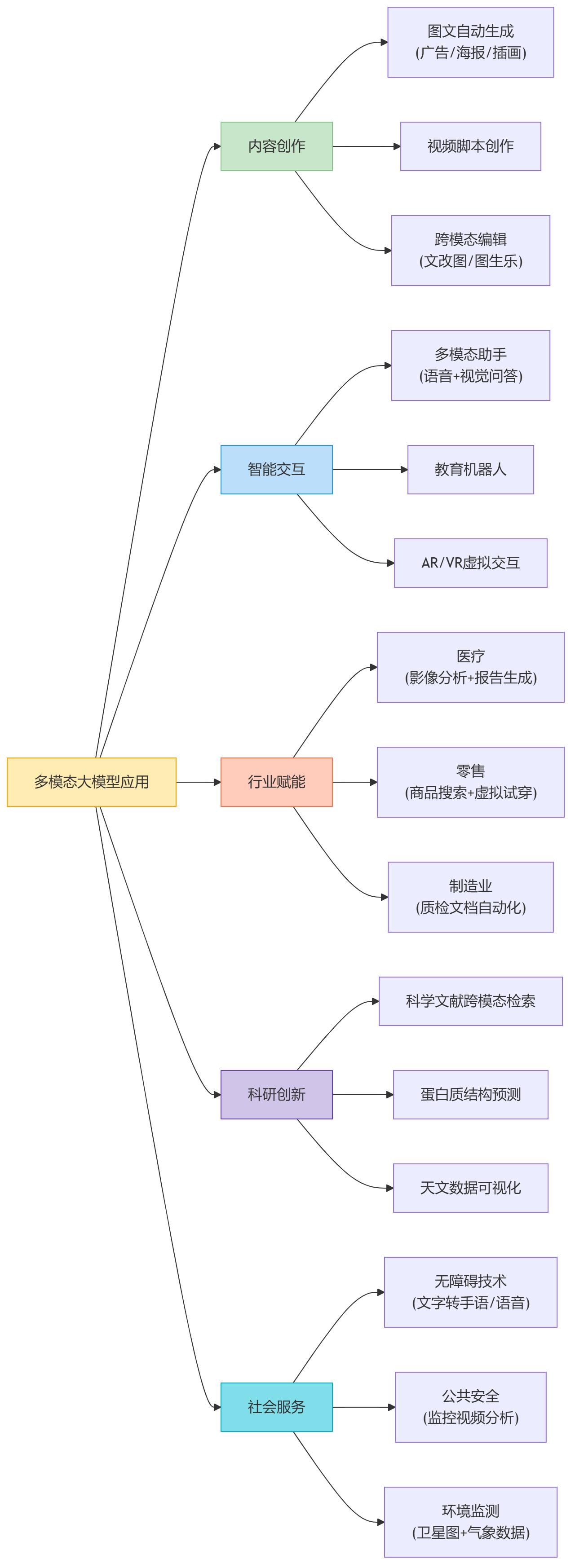
1.1 什么是多模态大模型

多模态大模型都是Transformer based架构，NLP对文本进行embedding，CV对图像patch进行Embedding，从图像、视频、文本、语音数据中提取特征，转换为tokens，进行不同模态特征的对齐，送入Transformer进行运算。

1.2与单模态大模型的对比



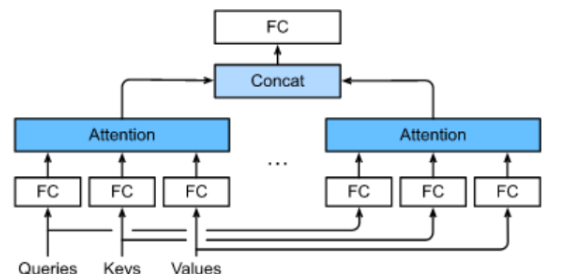
1.3多模态大模型的应用



**2.如何让LLM具有推理能力**

**2.1：transfomer核心与自注意力机制**

Transformer 是一种基于自注意力机制（self-attention）的序列到序列（sequence-to-sequence）模型。它使用了多头注意力机制，能够并行地关注不同位置的信息。Transformer 在自然语言处理领域中广泛应用，如机器翻译和文本生成。Masked-MultiHead Attetion：独立学习得到 head 组不同的 线性投影（linear projections）来变换查询、键和值。



核心思想：动态注意力

传统模型（如RNN）：按顺序处理词，无法直接捕捉远距离关系。

Transformer：让每个词直接“查看”其他所有词，通过注意力权重决定关注谁。  
2. 三大关键机制

(1) 自注意力（Self-Attention）

Query（Q）, Key（K）, Value（V）：

每个词生成这三组向量，用来计算与其他词的关系。

权重计算：

通过Q和K的点积，得到词与词之间的相关性分数（如“猫”和“吃”分数高）。

加权输出：

用权重对V加权求和，得到新词表示（包含上下文信息）。

(2) 多头注意力（Multi-Head Attention）

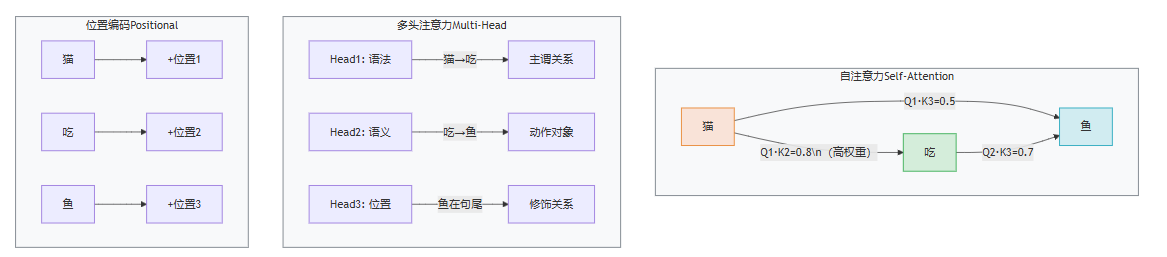
并行运行多组自注意力，捕捉不同角度的关系：

例：一组关注语法，另一组关注语义。

(3) 位置编码（Positional Encoding）

为词向量添加位置信息，解决“无顺序”问题：

例：“猫追狗” 和 “狗追猫” 位置编码不同。



**2.2：prefill decoding KV\_Cache**

1. Prefill（预填充）阶段

目标：初始化输入文本的上下文表示，为后续生成做准备。

输入："The cat sat on the \_\_\_"

执行过程：

Token化与嵌入

将输入文本拆分为token序列：["The", "cat", "sat", "on", "the"]。

每个token转换为向量（词嵌入），并添加位置编码（保留词序信息）。

并行计算注意力

模型一次性计算所有token的隐藏状态（Hidden States）。

对每个token生成Key（K）和Value（V）矩阵，用于后续注意力计算。



2. Decoding（解码）阶段

目标：基于Prefill的上下文，逐步生成补全词。

输入：Prefill阶段的KV Cache + 已生成的部分结果（初始为空）。

执行过程：

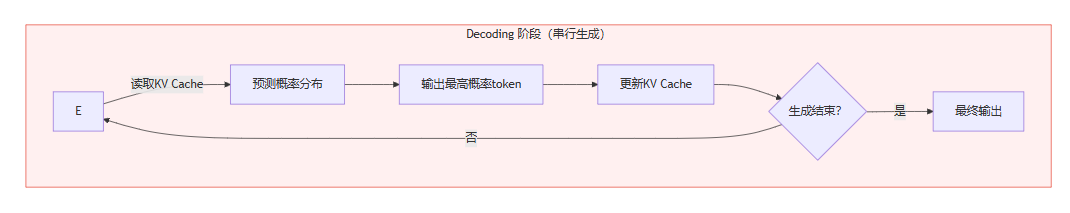
第1步生成

模型读取KV Cache中所有token的K/V（The到the）。

计算当前上下文（"The cat sat on the"）的Query（Q），并与历史K计算注意力权重：

发现"sat"和"the"权重较高（语义相关）。

预测下一个token的概率分布：



1. KV Cache与二者的关系

KV Cache是连接Prefill和Decoding的核心桥梁：

Prefill为其填充初始燃料（输入token的K/V），

Decoding依赖其实现高效自回归生成，并动态扩展它。

两者通过KV Cache的读写协作，实现了Transformer推理的效率与灵活性。

