第四讲
提示词工程（I）
Lecture 4
Prompt Engineering (I)

明玉瑞 Yurui Ming

yrming@gmail.com

声明
Disclaimer

语言大模型训练过程
Training of LLM

- 通常情况下，语言大模型的训练涉及到如下几个阶段。这些阶段代表了训练类似GPT模型的过程顺序，使其能够以无监督的方式从大量数据中学习、针对特定任务进行微调，并根据上下文和反馈可能进一步优化其结果，旨在创建生成更准确、上下文相关和流畅的文本的模型：

  - 无监督预训练： 该阶段涉及使用无监督学习方法在大型未标记文本语料库上训练模型。模型学习预测句子中缺失或屏蔽的单词，或预测序列中的下一个单词。这一步帮助模型捕捉一般的语言模式、语法和语义，而不需要特定于任务的标签。

  - 迁移学习/微调： 在对未标记文本进行预训练后，可以使用有标签数据来微调或适应特定任务。通过引入情感分析、文本分类或问答等任务的有标签数据，可以调整模型参数以在这些特定任务上表现出色。这一步使得模型能够专门化并在各种下游任务上表现良好。

  - 上下文学习（或强化学习）： 一些模型可能会采用上下文学习或强化学习策略进一步提高性能。这涉及基于其在特定上下文中生成的回应质量向模型提供反馈。模型可能会根据其响应与所需标准的匹配程度而获得奖励或惩罚，从而根据这种反馈机制改进其输出。

语言大模型训练过程
Training of LLM

- Generally, training of large language models involve the following stages. These stages represent a sequential process in training models like GPT, allowing them to learn from vast amounts of data in an unsupervised manner, fine-tune for specific tasks, and potentially refine their responses based on context and feedback. This iterative process aims to create models that generate more accurate, contextually relevant, and fluent text:

  - Unsupervised Pre-training: This phase involves training the model on a large corpus of unlabeled text using unsupervised learning methods. The model learns to predict missing or masked words within sentences or to predict the next word in a sequence. This step helps the model capture general language patterns, syntax, and semantics without task-specific labels.

  - Transfer Learning/Fine-Tuning: After pre-training on unlabeled text, the model can be fine-tuned or adapted to specific tasks with labeled data. By introducing labeled data for tasks like sentiment analysis, text classification, or question-answering, the model's parameters are adjusted to excel at these particular tasks. This step enables the model to specialize and perform well on various downstream tasks.

  - In-Context Learning (or Reinforcement Learning): Some models might employ in-context learning or reinforcement learning strategies to further enhance performance. This involves providing feedback to the model based on the quality of its generated responses in specific contexts. The model might receive rewards or penalties based on how well its responses align with desired criteria, leading it to improve its outputs based on this feedback mechanism.

提示词工程
Prompt Engineering

- 类似ChatGPT等这样大模型的成功，需要综合用到上面所讲的各项理论与算法，再配合工程实践与经验，才能在基于初始向量，或单词（Token）的基础上，生成对应的文本。下面不再讨论相关技术细节，而是从应用的角度，讨论与Token相关的另一个话题，称为提示词工程。

- 在大语言模型（LLM）的研究与使用中，人们发现输入的Token可以影响模型的表现，包括内容生成与内容生成的质量。有些Token可能与生成的内容直接相关，但有些Token可能是具有指向性的，即指示模型如何工作。此时，这些Token又被称为提示词（Prompt）。为了发展和优化提示词，以便高效地使用语言模型应用于各种研究和主题，人们创建了一个相对较新的学科，称为提示词工程。

- 提示词工程主要包括设计和开发提示，它涵盖了一系列用于与LLM交互和发展的技能和技巧，是连接、构建和理解LLM能力的重要技能：

  - 研究人员可以利用提示词工程来提高LLM在各种常见和复杂任务上的能力，如问题回答和算术推理，以便更好地了解LLM的能力和局限性。

  - 开发人员使用提示工程词设计健壮且有效的提示词技术，与LLM以及其他工具进行交互。

  - 用户可以使用提示工程技术来提高LLM的安全性，并构建新功能，如使用领域知识和外部工具增强LLM。

提示词工程
Prompt Engineering

- The success of large models such as ChatGPT requires a comprehensive combination of the theories and algorithms mentioned above, combined with engineering practice and experience, to generate corresponding text based on initial vectors or tokens. The following will not discuss technical details but from the perspective of application, discuss another topic related to tokens, called prompt engineering.

- In the research and use of language models (LMs), it has been found that input tokens can affect the performance and quality of content generation of the model. Some tokens may be directly related to the generated content, but some tokens may be directional, indicating how the model works. These tokens are then called prompts. To develop and optimize prompts for efficient use of language models in various research and topics, a relatively new discipline called prompt engineering has been created.

- Prompt engineering mainly includes designing and developing prompts, covering a range of skills and techniques for interacting and developing with large language models (LLMs), which are important skills for connecting, building, and understanding LLM capabilities:

  - Researchers can use prompt engineering to improve the ability of LLM on various common and complex tasks, such as question answering and arithmetic reasoning, to better understand the capabilities and limitations of LLM.

  - Developers use prompt engineering to design robust and effective prompt techniques to interact with LLMs and other tools.

  - Users can use prompt engineering techniques to enhance the security of LLM and build new features, such as using domain knowledge and external tools to enhance LLM.

提示词工程
Prompt Engineering

- 简单地讲，我们将由不同类型的提示词组成的模型输入称为提示。

- 目前比较流行的大型语言模型，国外有OpenAI的ChatGPT，Google的Bard，以及Hugging Face上的多种模型；国内的有文心一言、智谱清言等。使用提示与LLM交互时，一般情况下会用户界面进行；有些情况下亦可以通过API直接与LLM进行交互。不论直接使用或使用API，用户可以通过配置一些参数以获得不同的提示结果。下面介绍了最重要的两个参数：

  - Temperature：简单来说，temperature的参数值越小，模型就会返回越确定的一个结果。如果调高该参数值，大语言模型可能会返回更随机的结果，也就是说这可能会带来更多样化或更具创造性的输出。例如，在实际应用方面，对于质量保障（QA）等任务，我们可以设置更低的temperature值，以促使模型基于事实返回更真实和简洁的结果。 对于诗歌生成或其他创造性任务，用户可以适当调高 temperature 参数值。

  - Top_p：同样，使用top_p（与temperature一起称为核采样的技术），可以用来控制模型返回结果的真实性。如果用户需要准确和事实的答案，就把参数值调低。如果用户想要更多样化的答案，就把参数值调高一些。

提示词工程
Prompt Engineering

▶ Simply put, we call the model input consisting of different types of prompts as a prompt.

▶ Currently popular large language models include OpenAI's ChatGPT, Google's Bard, and various models on Hugging Face; domestically, there are Wenxin Yiyuan, Zhipu Qingyan, and others. When interacting with LLM using prompts, it is generally done through a user interface, and in some cases, it can also be done directly with LLM through an API. Whether via interface or via an API, users can obtain different prompt results by configuring some parameters. The two most important parameters are described below:

▶ Temperature: Simply put, the smaller the temperature parameter value, the more certain the result the model returns. If the parameter value is increased, LLMs may return more random results, which may bring more diversified or more creative outputs. For example, in practical applications such as quality assurance (QA) tasks, we can set a lower temperature value to prompt the model to return more realistic and concise results based on facts. For poetry generation or other creative tasks, users can appropriately increase the temperature parameter value.

▶ Top_p: Similarly, top_p (a technique called nucleus sampling together with temperature) can be used to control the authenticity of the model's returned results. If the user needs accurate and factual answers, they should set the parameter value lower. If the user wants more diverse answers, they should increase the parameter value.

提示词工程
Prompt Engineering

▶ 提示词可以包含传递到模型的指令（Instruction），指示模型怎么做；或用户期待模型解决的问题（Problem）等信息，也可以包含其他详细信息，如上下文（Context）、输入（Input）或示例（Example）等。用户可以通过简单的提示词，获得许多结果，但结果的质量一般与提示词的数量和完善度有关。用户可以通过不同类型的提示词来更好地指导模型，并因此获得更好的结果。

▶ 提示可以包含以下任意要素：

   ▶ 指令：想要模型执行的特定任务或指令。

   ▶ 上下文：包含外部信息或额外的上下文信息，引导语言模型更好地响应。

   ▶ 输入数据：用户输入的内容或问题。

   ▶ 输出指示：指定输出的类型或格式。

▶ 通常，提示所需的格式取决于用户想要语言模型完成的任务类型，并非所有以上要素都是必须的。

提示词工程
Prompt Engineering

▶ Prompts can contain instructions that are passed to the model, indicating how the model should perform, or information about the problems that users expect the model to solve. Prompts may also contain other detailed information such as context, input, or examples. Users can obtain many results through simple prompts, but the quality of the results is generally related to the quantity and completeness of the prompts. Users can better guide the model through different types of prompts and thereby obtain better results.

▶ Prompts can contain any of the following elements:

▶ Instructions: specific tasks or instructions that the user wants the model to perform.

▶ Context: external information or additional contextual information that guides the language model to respond better.

▶ Input data: the content or questions entered by the user.

▶ Output instruction: specifies the type or format of the output.

▶ The format required for prompts generally depends on the type of task that the user wants the language model to complete, and not all of the above elements are necessary.

提示词工程
Prompt Engineering

▶ 提示的设计往往是一个迭代的过程，通常需要大量的实验来获得最佳结果。尽管可以使用像ChatGPT或智谱清言这样的平台作为尝试的起点，但不同的模型可能对同一提示返回的结果可能不同。

▶ 在人们尝试的过程中，逐渐形成以下共识及经验做法：

➢ 从简单的提示开始

　▶ 通过不断添加更多的元素和上下文，比较生成的结果并评估是否需要对提示词进行调整。对于基于大模型开展业务的公司来说，开发人员甚至可能在此过程中对提示进行版本控制。一般情况下，具体性的、简洁性的和简明性的提示，带来的结果通常已经足够好。

　▶ 例如，当有一个涉及许多不同子任务的大任务时，可以尝试将任务分解为更简单的子任务，并随着获得更好的结果而不断构建。这避免了在提示设计过程中一开始就添加过多的复杂性。假设我们期望LLM帮助设计一个复杂软件系统，这种情况下可以将整个系统分解成若干子系统，然后再将子系统分解成具体的模块，基于这些模块与LLM交互，得到示例性代码，最后将这些代码由底向上集成，得到所期望的系统。

提示词工程
Prompt Engineering

- The design of prompts is often an iterative process that typically requires a lot of experimentation to obtain the best results. Although platforms like ChatGPT or Zhipu Qingyan can be used as a starting point for exploration, different models may return different results for the same prompt.

- As people experiment, the following consensus and empirical practices gradually emerge:

- Start with simple prompts

  - Compare generated results by continuously adding more elements and context, and evaluate whether adjustments to the prompt are necessary. For companies that engage in business based on large models, developers may even version-control the prompts during this process. Generally, specific, concise, and clear prompts usually produce good enough results.

  - For example, when there is a large task involving many different sub-tasks, try breaking down the task into simpler sub-tasks and build upon them as better results are obtained. This avoids adding too much complexity at the beginning of the prompt design process. Suppose we expect LLM to help design a complex software system. In this case, the entire system can be decomposed into several subsystems, and then each subsystem can be broken down into specific modules. Based on these modules interacting with LLM, exemplary code can be obtained, and finally, these codes can be integrated from bottom to top to obtain the desired system.

提示词工程
Prompt Engineering

➢ 开宗明义地使用指令

  ▶ 用户可以使用命令来指示模型执行各种简单任务，例如"分类"、"总结"、"翻译"、"排序"等，从而为各种简单任务设计有效的提示。通常，在不同的上下文中，指令可能具有不同的意义，因此可能需要尝试结合使用不同的关键字、上下文和数据组合，进行一定的实验，才能看出对于特定的用例和任务，哪种方法最有效。

  ▶ 通常情况下，在指示模型执行具体任务的过程中，上下文越具体和任务越相关，效果越好。

  ▶ 下面是一些通过指令与LLM交互的例子：

  1. 请将如下文字翻译成英文

  2. 请告诉我在Excel中如何求均值与方差的函数

  3. 请推荐适合素食主义者的食谱

  4. 请在网络技术的背景下解释英文缩写"IP"的意义

提示词工程
Prompt Engineering

➤ Use commands explicitly

▶ Users can use commands to instruct the model to perform various simple tasks, such as "classify", "summarize", "translate", "sort", etc., in order to design effective prompts for various simple tasks. Typically, commands may have different meanings in different contexts, so it may be necessary to experiment with different keyword, contextual, and data combinations to determine which method is most effective for a specific use case or task.

▶ Generally, in the process of instructing the model to perform a specific task, the more specific and relevant the context is to the task, the better the effect.

▶ Here are some examples of interacting with LLM through commands:

1. Please translate the following text into English.

2. Please tell me the functions for calculating mean and variance in Excel.

3. Please recommend recipes suitable for vegetarians.

4. Please explain the meaning of the English abbreviation "IP" in the context of network technology.

提示词工程
Prompt Engineering

➤ 体现具体性

▶ 对期望模型执行的指令和任务，提示应该非常具体。通常，提示越具体和详细，结果就越好。但这和指令的简洁性并不矛盾，因为仍需模型从提示中得到用户的意图。

▶ 当用户知道所期望的结果或生成样式时，这一点尤为重要。具有良好的格式和描述性提示会导致更好的结果。实际上，在提示中提供示例非常有效，可以以特定格式获得所需的输出。

➤ Be specific

▶ Prompts should be very specific for the instructions and tasks that the user expects the model to perform. Generally, the more specific and detailed the prompt, the better the results. However, this is not contradictory to the conciseness of the instruction because the model still needs to understand the user's intention from the prompt.

▶ This is particularly important when the user knows the expected result or generation style. Good prompts with good formatting and descriptiveness lead to better results. In fact, providing examples in prompts is very effective and can obtain the desired output in a specific format.

提示词工程
Prompt Engineering

▶ 下面是指示模型具体输出格式与实际输出的例子：

统计如下几个学生的成绩情况，以表格的形式输出：表格的每一行是一个学生的成绩信息，第一列是学生姓名，第二列是语文成绩，第三列是数学成绩，第四列是英语成绩。最后一列是学生各科成绩的平均成绩。最后一行是所有学生单个学科的平均成绩。下面是成绩信息：小明，语文98，数学96，英语93；小张，语文93，数学86，英语83；小李，语文88，数学93，英语89。

▶ Here are examples of specific instructions for the model output format and actual output:

Calculate the grades of the following students and output them in table form: each row of the table represents a student's grade information, with the first column being the student's name, the second column being the Chinese score, the third column being the math score, and the fourth column being the English score. The last column is the average score of all subjects for each student. The last row shows the average score of each subject for all students. Here are the grade information: Xiaoming, Chinese 98, Math 96, English 93; Xiaozhang, Chinese 93, Math 86, English 83; Xiaoli, Chinese 88, Math 93, English 89.

名称	语文成绩	数学成绩	英语成绩	平均分
小明	98	96	93	95.67
小张	93	86	83	87.33
小李	88	93	89	90
平均分	93	91.67	88.33	-

提示词工程
Prompt Engineering

➢ 避免不精确

  ▶ 在关于详细和格式改进的提示中，很容易陷入过于聪明的提示陷阱，从而可能创建不精确的描述。一般地说，与模型的交互类似于有效的沟通——越直接，信息传递就越有效。因此，通常需要在具体和直接之间做出一个平衡。

  ▶ 另外，由于一些文化差异，例如，我们常说的盐少许、油少许，可能在西方文化里没有办法理解，但实际准备菜肴时，基本没有谁实际去量。但对于模型来说，可以量化的提示一般可以得到更好的结果。例如，我们可能希望模型向高中生简短的解释提示词工程，则提示"使用简短的话向高中学生解释提示词工程"，可能不如"使用2—3句话向高中学生解释提示词工程"。尽管"简短"表明了使用者的倾向，但"2—3"则明确地表明了对输出的约束。

➢ 明确做什么，而不是不做什么

  ▶ 设计提示时的另一个常见技巧是避免说不要做什么，而是说要做什么。在鼓励做什么的过程中，用户会潜在关注导致模型产生良好响应的细节，这其实是提示设计的正向引导与迭代。

  ▶ 例如，"生成二分查找的代码，变量的名称不要过于简短"，其效果可能不如提示"生成二分查找的代码，变量的名称要尽可能长"，得到的代码更能体现用户对代码易读性的要求。

提示词工程
Prompt Engineering

- Avoid impreciseness

  - Generally speaking, interacting with a model is similar to effective communication--the more direct, the more effective the information transfer. Therefore, there is usually a need to strike a balance between being specific and direct.

  - For example, common expressions "a pinch of salt" or "a little oil," in Chinese may not be understood in Western culture, but when preparing dishes, few people actually measure them out. However, prompts that can be quantified generally lead to better results for models. For example, if we want a model to explain keyword engineering in a short way to high school students, then "Using a short sentence to explain keyword engineering to high school students" may not be as good as "Using 2-3 sentences to explain keyword engineering to high school students." Although "short" indicates the user's tendency.

- Be clear about what to do, rather than what not to do

  - Another common technique when designing prompts is to avoid saying what not to do and instead say what to do. In the process of encouraging what to do, users will potentially pay attention to details that lead to good responses from the model. This is actually positive guidance and iteration in prompt design.

  - For example, "Generate code for binary search, but don't use overly short variable names" may be less effective than "Generate code for binary search, using variable names that are as long as possible"

提示词工程
Prompt Engineering

➤ 必要时进行样本提示

- ▶ 经过大量数据训练并调整指令的LLM一般情况下具有较好的泛化能力，即使对没有见过的任务也有可能可以完成，我们称模型能执行零样本任务。当零样本不起作用时，可以设计少样本提示，帮助模型完成特定任务。

- ▶ 例如，当我们直接输入以下提示时，模型可能不知道要做什么；但当我们给模型足够的提示后，模型可能会很快理解了任务并进行执行。

➤ Provide sample prompts when necessary

- ▶ LLMs that have been trained with a large amount of data and adjusted instructions generally have good generalization abilities, which means that they can even complete tasks that have not been exposed to the models before. We call this ability to perform zero-shot tasks. When zero-shot is not effective, few-shot prompts can be designed to help the model complete specific tasks.

- ▶ For example, when we directly input the following prompt, the model may not know what to do; but when we give the model enough prompts, it may quickly understand the task and execute it.

提示词工程
Prompt Engineering

➤ 下面例子展示了样本提示的用途:

The following example illustrates the use of sample prompts:

① Q: 哇，那部电影太棒了！
A:

② 是的，有些电影真的是非常棒！如果您愿意分享电影的名称或一些具体内容，我也很乐意听听您的观点和体会。

➤ 模型起初并不知道用户要进行情感分类，当用样本进行提示后，模型学习到了用户的企图，并成功完成任务。

Initially, the model had no idea that the user wanted to perform sentiment classification. However, after being prompted with a sample, the model learned about the user's intent and successfully completed the task.

③ Q: 哇，那部电影太棒了！
A: Positive
Q: 今天太糟糕了。
A: Negative
Q: 今天有个会议要参加
A: Neutral
Q: 那个发言人都没有将问题讲清楚！
A:

④ Negative

提示词工程
Prompt Engineering

➢ 强制链式思考提示

▶ 链式思考（CoT）是一种问题解决和决策制定的方法，也被称为连锁思维或切题方法。其核心思想是将复杂的问题分解为更小、更具体的问题，然后逐步深入分析每个小问题，直到找到最后的答案。一些语言大模型，可能在进行推理复杂任务时性能欠佳。此时，引入的链式思考（CoT），即通过提示，强制LLM输出中间推理步骤，有时可以实现复杂的推理能力。

➢ Enforce Chain-of-Thought Prompting

▶ Chain of Thoughts (CoT) is a method of problem-solving and decision-making, also known as chain thinking or problem-solving methods. Its core idea is to decompose complex problems into smaller, more specific problems, and then gradually analyze each small problem until the final answer is found. Some large language models may have poor performance when performing complex reasoning tasks. In such cases, the introduction of Chain of Thoughts (CoT), that is, by prompting, forcing LLMs to output intermediate reasoning steps, sometimes can achieve complex reasoning abilities.

提示词工程
Prompt Engineering

➤ 右边是强制链式思考的例子：

On the right is an example of forced chain thinking:

> 👤 我去市场买了10个苹果。我给了邻居2个苹果和修理工2个苹果。然后我去买了5个苹果并吃了1个。我还剩下多少苹果？请直接告诉我答案。

> ⬤ 你现在还剩下 12 个苹果。

> 👤 我去市场买了10个苹果。我给了邻居2个苹果和修理工2个苹果。然后我去买了5个苹果并吃了1个。我还剩下多少苹果？请逐步解答。

> ⬤ 首先，你在市场买了 10 个苹果。
>
> 然后，你分别给了邻居和修理工 2 个苹果，这就是 4 个苹果。那么你剩下的苹果数量是 10-4=6 个。
>
> 接着，你说你又买了 5 个苹果，那么你现在有 6+5=11 个苹果。
>
> 最后，你说你吃了 1 个苹果，所以你现在还剩下 11-1=10 个苹果。
>
> 所以，你现在还剩下 10 个苹果。

实例
Demonstration

➤ 辅助写作

Writing Assistant

▶ 请写一段太阳在海面升起的文字。文字中对景物的描写要尽可能地多用比喻；句子之间要尽可能多用排比。对太阳、海面、云彩、波涛、轮船的描述都要涉及。

👤 请写一段太阳在海面升起的文字。文字中对景物的描写要尽可能地多用比喻；句子之间要尽可能多用排比。对太阳、海面、云彩、波涛、轮船的描述都要涉及。

🌀 晨曦微破，太阳犹如一颗明珠，缓缓从海平线升起。海面波光粼粼，犹如无数银针闪烁，与天空中云彩交相辉映。云彩宛如棉花糖般洁白，柔和地映衬着太阳，形成一幅美妙的画卷。

太阳逐渐升高，犹如一枚金币，光芒四射，洒向大地。海面上的波涛宛如骏马奔腾，翻滚着白色的浪花，冲刷着海岸线。一艘轮船在海面上行驶，如同一位勇士驾驭着波涛，勇往直前。

随着太阳的升起，天空中的云彩被染上了一层金黄色，宛如大师级画家勾勒的画卷。阳光照耀在波涛上，犹如黄金铺设的地毯，熠熠生辉。轮船的影子在波光中摇曳，仿佛是天空中的云彩在水中嬉戏。

此刻，太阳已跃居天空之中，犹如一位王者，照耀着一切。海面上的波涛在此刻变得温顺，犹如婴儿的摇篮，宁静而美丽。轮船继续前行，留下的白色浪花宛如珍珠项链，镶嵌在蓝色的海面上。

太阳越发辉煌，天空中的云彩被分割成无数片，犹如一幅泼墨山水画。海面上的波涛在此刻翻滚，犹如骏马奔腾，展示着生命的活力。轮船行驶过的地方，留下了一道美丽的风景线，让人陶醉不已。

在这如诗如画的景象中，太阳、海面、云彩、波涛和轮船共同谱写了一曲赞美大自然的赞歌，令人陶醉其中，流连忘返。

示例
Demonstration

➤ 代码生成

Code Generation

▸ 由于许多大模型的语料库涵盖Github上的许多源代码，因此，包括ChatGPT在内的许多LLMs的代码生成能力较强。已经有一些公司将该种能力运用到了日常工作中，在一定程度上减轻了信息技术人员特别是软件工程师的负担。特别的，专门为程序设计工作着设计的Copilot，也得到了越来越多的应用。下面展示了代码生成的例子：

冒泡排序（Bubble Sort）是一种简单的排序算法。它重复地遍历待排序的数列，一次比较两个元素，如果顺序错误就把它们交换过来。遍历数列的工作是重复地进行直到没有再需要交换，也就是说该数列已经排序完成。

以下是一个使用 Python 实现的冒泡排序示例代码：

```python
def bubble_sort(arr):
    n = len(arr)

    # 遍历所有数组元素
    for i in range(n):
        # Last i elements are already in place, so we don't need to check them
        for j in range(0, n - i - 1):
            # 交换如果发现元素不是按顺序排列的
            if arr[j] > arr[j + 1]:
                arr[j], arr[j + 1] = arr[j + 1], arr[j]
```

示例
Demonstration

- 个人助理

  Personal Assistant

  - 由于语言大模型基于丰富的语料库，因此，可以解决工作与生活中的许多常见问题。右边是一个出行安排的例子，假设要去一个陌生的地方游览，可以通过与语言大模型进行交互，得到相应的景点推荐与形成安排：

我准备去巴黎玩，请帮我推荐5个景点。每个景点我计划玩一天，请同时以表格的形式帮我安排好日程。

以下是为您推荐的五个巴黎景点以及相应的日程安排：

景点名称	推荐理由	游玩时间
卢浮宫	世界著名的艺术博物馆，拥有众多珍贵文物和艺术品，如蒙娜丽莎、断臂的维纳斯等。	第一天
埃菲尔铁塔	巴黎地标建筑，登上铁塔可以俯瞰整个巴黎市区美景。	第二天
巴黎圣母院	古老而宏伟的天主教堂，因雨果的同名小说而声名大噪。	第三天
香榭丽舍大街	巴黎最美丽的街道，世界著名的名品街，感受法国浪漫气息。	第四天
蒙马特高地	遇见巴黎的另类文艺，富有艺术气息的地方。	第五天