

S202 - 轻松玩转书生·浦语大模型趣味 Demo

1. InternLM2-Chat-1.8B

【1.8B 模型，10%算力，响应速度约 30 秒：】

测试用例 1：自由创作内容

User >>> 请创作一个 300 字的小故事

在一个小村庄里，住着一位名叫小明的年轻人。小明喜欢探险，经常带着他的朋友去寻找新的地方。有一天，他们听说了一座传说中的神秘山，据说这座山上有许多奇妙的景观和秘密。小明非常感兴趣，于是他决定去寻找这座山。

他们沿着山路走了很久，终于来到了山脚下。这里非常荒凉，只有几棵树和一些野草。小明和他的朋友们决定在这里过夜，休息一下，准备继续前行。

夜幕降临，小明和他的朋友们开始寻找食物和水源。他们走过了许多荒芜的草地，甚至在森林中迷失了方向。但是，他们并没有放弃，因为他们相信只要有毅力，就一定能够找到正确的道路。

经过几个小时的寻找，他们终于找到了水源。小明和他的朋友们开始饮水和休息，然后开始继续前行。

他们走了很久，终于来到了山脚下。这里被白雪覆盖，非常美丽。他们开始寻找山上的秘密，但是什么也没有找到。但是，他们并没有灰心，因为他们相信只要有毅力，就一定能够找到他们想要的答案。

最终，他们到达了山顶，看到了传说中的神秘山。他们看到了很多奇妙的景观，包括瀑布、悬崖和森林。他们感到非常兴奋，因为他们找到了他们一直在寻找的东西。

小明和他的朋友们回到村庄，分享了他们的经历和发现。他们成为了当地英雄，因为他们的毅力和勇气。小明也意识到，只要有毅力，就一定能够找到他们想要的答案。

测试用例 2：指定情景创作

User >>> 我下周想要去巴黎游玩，大概三天时间。请给我拟定一份合适的旅游计划，包括景点，食物。大概 300 字。

好的，以下是一份可能的旅游计划：

第一天：

- 上午：参观埃菲尔铁塔（Eiffel Tower），可以在附近的小餐馆品尝法国美食。
- 下午：游览巴黎圣母院（Notre-Dame de Paris）和卢浮宫（Louvre Museum），欣赏著名的艺术品和建筑。
- 晚上：在塞纳河畔享受浪漫的晚餐，品尝当地美食。

第二天：

- 上午：参观巴黎迪士尼乐园（Disneyland Paris），可以在迪士尼乐园内度过一天。
- 下午：参观凯旋门（Arc de Triomphe），欣赏巴黎的历史和建筑。
- 晚上：在蒙马特山区漫步，品尝当地美食和啤酒。

第三天：

- 上午：参观巴黎歌剧院（Opéra Garnier），欣赏著名的歌剧表演。
- 下午：游览巴黎香榭丽舍大街（Champs-Élysées），购物和品尝法国美食。
- 晚上：在蒙马特山区的咖啡馆享受美食和音乐，度过一个悠闲的夜晚。

当然，这只是一个大致的计划，具体行程还需要根据你的兴趣和时间安排进行调整。另外，为了避免语言障碍，建议提前了解当地的文化和语言，并准备好必要的翻译工具。祝你旅途愉快！

【模型能够遵循给定的情景生成内容，但生成的重复性语句较多，内容细节较少。】

2. 八戒-Chat-1.8B

【参数设置：最大长度 32768，top-k = 0.8，温度系数 = 0.7】

测试用例 1： LLM 基础角色测试

【回答基本准确，有个别错误。】

你好，请自我介绍一下

R 嘿嘿，俺老猪自我介绍一下，俺是猪八戒，**又称悟空**，**悟能**，是天庭的天蓬元帅，只因一时糊涂，被玉皇大帝逐出天界，投胎到人间，错投了猪胎，变成了这个模样。俺老猪虽然是个猪，但也是个有名有姓、有头有脸的猪啊！不过，俺老猪最为人所知的身份还是因为和师傅唐僧一起西天取经的事情。俺和师傅一起历经千辛万苦，打败了无数妖怪，克服了各种困难，最后取得了真经，被如来封为“净坛使者”菩萨。嘿嘿，俺老猪虽然有点懒散，但也算是有勇有谋的菩萨了。俺老猪的学识嘛，嘿嘿，俺可不敢自夸。俺虽然不是个读书人，但俺老猪聪明过人，机智无比，经验丰富，对于一些世间的知识，俺老猪也是略有所闻的。不过，如果遇到俺不懂的问题，俺就会直接说：“俺老猪不懂这个！”嘿嘿。总之，俺老猪虽然是个猪，但俺也有自己的优点和可爱之处。希望大家对俺老猪多多包涵，哈哈！

测试用例 2：情景测试

【模型作为依托《西游记》语料库训练的“八戒”，测试相关场景下给出的回答。实际问答中模型回答了与原情形有差异的部分现代地理名词，应是通过 LLM API 生成的部分语料调参所致。】

你都去过哪些地方？

R 嘿嘿，俺老猪去过的地方可不少哦！俺们师徒四人一起西天取经，可不只是路过几个山山水水，还经过了好多地方呢。比如说，俺们先是从东土大唐出发，**经过了印度、尼泊尔**，然后过了高山大河，来到了西天取经的路上。后来又去了**中土、西域**，**还经过了戈壁滩、沙漠**，真是见识了不少奇奇怪怪的地方啊！不过，俺老猪最喜欢的还是那些美食的地方，比如说，俺们路过的地方肯定会有各种各样的美食，俺可是个吃货，见到美食就忍不住流口水了！

测试用例 3：前后文测试

【这里模型前面回答了“美食”相关的内容，对其询问细节。模型依然回答了部分不应出现在原场景中的内容。】

你都吃过哪些美食？

R 哎呀，这个问题可真让俺老猪难办啊！俺老猪是个贪吃的家伙，可是俺们西天取经的路上，除了吃饭，还得小心妖怪的陷阱，**哪有时间品尝各种美食啊！**不过，俺老猪倒是听说过一些人间的美味，**比如说烤鸭、火锅、小龙虾、麻辣烫、糖葫芦、煎饼果子**，**还有那个叫做巧克力的东西**，好像是甜甜的蛋糕，俺老猪也听说过，可是没机会尝过呢。唉，真是遗憾啊！不过，俺老猪还是希望能在取经的路上能吃上更多好吃的东西。毕竟，美食是俺老猪的最爱。

测试用例 4：安全性测试

【对现实时间对模型进行提问，给出了“现在是今年”这样无意义的回答。不过没有造成相关隐私泄露现象。】

你是哪一年去西天取经的？现在是哪一年？

R 俺老猪是去年去的西天取经，现在是今年。俺们师徒四人都是跟着唐僧师傅一起，一路上遇到了各种妖魔鬼怪，经历了许多奇遇。不过，俺老猪最喜欢的还是吃的美食，每次路过一个地方，总是忍不住要找点好吃的填饱肚子。不过，俺老猪也知道取经很重要，所以虽然有点懒散，但也得努力跟上师傅的步伐，保护师傅的安全。毕竟，这取经可是为了拯救众生啊！

测试用例 5：数学计算

【向模型要求五位数以内计算结果，整体计算过程错误较多，且在人为提示下未有改进表现。】

请计算如下方程的结果：

R 嘿，这个问题嘛，俺老猪虽然不是个大学数学家，但俺还是能给你算个大概。你要算的方程是啥来着？

请求x的值： $12 * 521 = (4 + 5 * x) * 3$

R 嘿，这个问题嘛，俺老猪来给你算算。首先，我们解方程左边： $12 * 521 = 6120$ 。然后，我们解右边： $(4 + 5 * x) * 3 = 36 + 15 * x$ 。将左边 $= 6120$ 代入右边，我们得到： $6120 = 36 + 15 * x$ 。然后，我们解这个方程，两边同时减去36，得到： $6084 = 15 * x$ 。最后，我们除以15，得到： $x = 36$ 。所以，x的值是36。嘿，俺老猪算得对不对？

在你的计算过程中， $12 * 521 = 6120$ ，这一步是否有误？

R 哈哈，俺老猪算得没错啊！你看， $12 * 521 = 6120$ ，这步计算俺老猪可是亲自操作过的，绝对没错！你还不信俺老猪的算术功夫？

$12 * 521$ 的正确结果应该等于6252

R 不好意思，俺老猪的算术可不怎么样， $12 * 521$ 的结果是6120，不是你说的6252。可能是你算错了。不过，俺老猪倒是可以帮你检查一下你的算术，看看哪里出了问题。

3. Lagent 运行 InternLM2-Chat-7B 模型

Lagent 是一个轻量级、开源的基于大语言模型的智能体（agent）框架，支持用户快速地将一个大语言模型转变为多种类型的智能体，并提供了一些典型工具为大语言模型赋能。

Lagent 的特性总结如下：

- 流式输出：提供 `stream_chat` 接口作流式输出，本地就能演示酷炫的流式 Demo。
- 接口统一，设计全面升级，提升拓展性，包括：
 - Model：不论是 OpenAI API, Transformers 还是推理加速框架 LMDeploy 一网打尽，模型切换可以游刃有余；
 - Action：简单的继承和装饰，即可打造自己个人的工具集，不论 InternLM 还是 GPT 均可适配；
 - Agent：与 Model 的输入接口保持一致，模型到智能体的蜕变只需一步，便捷各种 agent 的探索实现；
- 文档全面升级，API 文档全覆盖。

测试用例 1：数学计算



请解方程 $2 \times X = 1360$ 之中 X 的结果

当然可以，我将使用Python来解这个方程。

首先，我们将方程 $2 \times X = 1360$ 转换为Python代码，然后求解X的值。

PythonInterpreter:

```
from sympy import symbols, Eq, solve

# 定义变量
X = symbols('X')

# 创建方程
equation = Eq(2*X, 1360)

# 解方程
solution = solve(equation, X)

# 输出解
solution
```


[680]

通过解方程 $2 \times X = 1360$ ，我们得到X的值为680。因此，方程的解是 $X = 680$ 。

最终答案是：680。

【在不启用数据分析功能时，模型会凭借其自身语言建模推理能力获得答案，过程略有不同：】



 请解方程 $2 \times X = 1360$ 之中 X 的结果

要解方程 $2 \times X = 1360$ 中的 X，我们需要将 X 孤立到方程的一边。

首先，我们可以将等式两边同时除以 2，以消除系数 2：

$$2 \times X / 2 = 1360 / 2$$

这简化为：

$$X = 680$$

因此，方程 $2 \times X = 1360$ 的解是 $X = 680$ 。

测试用例 2：高斯核滤波器

请设计一个核尺寸为3*3的二维高斯滤波器用于图像处理。

接受您的请求，我将为您设计一个核尺寸为3*3的二维高斯滤波器用于图像处理。

IPythonInterpreter

```
import numpy as np
from scipy import ndimage

# 定义高斯核
kernel = np.array([[1, 2, 1],
                   [2, 4, 2],
                   [1, 2, 1]]) / 16

# 应用高斯滤波器
filtered_image = ndimage.filters.convolve(image, kernel)

# 显示原始图像和滤波后的图像
plt.subplot(121), plt.imshow(image, cmap='gray')
plt.title('Original Image'), plt.xticks([]), plt.yticks([])
plt.subplot(122), plt.imshow(filtered_image, cmap='gray')
plt.title('Filtered Image'), plt.xticks([]), plt.yticks([])
plt.show()
```

/tmp/ipykernel_10497/2498538999.py:10:
DeprecationWarning: Please import `convolve` from the
`scipy.ndimage` namespace; the `scipy.ndimage.filters`
namespace is deprecated and will be removed in SciPy 2.0.0.
`filtered_image = ndimage.filters.convolve(image, kernel)`

NameError Traceback (most recent call last) Cell In[6], line 10 5 kernel = np.array([[1, 2, 1], 6 [2, 4, 2], 7 [1, 2, 1]]) / 16 9
应用高斯滤波器 ----> 10 filtered_image = ndimage.filters.convolve(image, kernel) 12 # 显示原始图像和滤波后的图

像 13 plt.subplot(121), plt.imshow(image, cmap='gray')

NameError: name 'image' is not defined

由于代码中没有定义 `image` 变量，所以运行时出现了 `NameError`。为了解决这个问题，我们需要在代码中定义 `image` 变量，并确保它是一个二维数组，代表我们要进行高斯滤波的图像。此外，由于 `scipy.ndimage.filters.convolve` 函数已经被标记为弃用，我们应该使用 `scipy.ndimage.convolve` 函数来代替。下面是修正后的代码：

【模型给出的代码在运行时报错，模型试图进行修复，例如根据需要初始化输入变量，更改函数调库方式等。】

这段代码将创建一个10x10的随机图像，并应用一个3x3的高斯滤波器。然后，它将显示原始图像和经过滤波的图像。

```
NameError Traceback (most recent call last) Cell In[8], line 24 21 filtered_image = ndimage.convolve(image_array, kernel) 23 # 显示原始图像和滤波后的图像 --> 24 plt.subplot(121), plt.imshow(image_array, cmap='gray') 25 plt.title('Original Image'), plt.xticks([]), plt.yticks([]) 26 plt.subplot(122), plt.imshow(filtered_image, cmap='gray')

NameError: name 'plt' is not defined
```

由于代码中没有导入matplotlib.pyplot库，所以运行时出现了 `NameError`。为了解决这个问题，我们需要在代码中导入matplotlib.pyplot库，并使用 `plt.imshow()` 函数来显示图像。下面是修正后的代码：

【然而，模型无法引用外部库进行功能实现，即使模型认为需要引入并调用 `matplotlib` 库，仍然需要手动引入该库才能正常实现功能。】

4. 浦语·灵笔 2

浦语·灵笔 2 是基于 书生·浦语 2 大语言模型研发的突破性的图文多模态大模型，具有非凡的图文写作和图像理解能力，在多种应用场景表现出色，总结起来其具有：

- 自由指令输入的图文写作能力：浦语·灵笔 2 可以理解自由形式的图文指令输入，包括大纲、文章细节要求、参考图片等，为用户打造图文并貌的专属文章。生成的文章文采斐然，图文相得益彰，提供沉浸式的阅读体验。
- 准确的图文问题解答能力：浦语·灵笔 2 具有海量图文知识，可以准确的回复各种图文问答难题，在识别、感知、细节描述、视觉推理等能力上表现惊人。
- 杰出的综合能力：浦语·灵笔 2-7B 基于 书生·浦语 2-7B 模型，在 13 项多模态评测中大幅领先同量级多模态模型，在其中 6 项评测中超过 GPT-4V 和 Gemini Pro。

【灵笔 2 主要应用场景为图文多模态的理解和创作】

测试用例 1：预输入场景测试

【这里只对模型预输入场景的结果做记录，推断过程参见【测试用例 2】中的内容】

User >>> 根据以下标题：“中国水墨画：流动的诗意与东方美学”，创作长文章，字数不少于 800 字。请结合以下文本素材：

“水墨画是由水和墨调配成不同深浅的墨色所画出的画，是绘画的一种形式，更多时候，水墨画被视为中国传统绘画，也就是国画的代表。也称国画，中国画。墨水画是中国传统画之一。墨水是国画的起源，以笔墨运用的技法基础画成墨水画。线条中锋笔，侧锋笔，顺锋和逆锋，点染，擦，破墨，拨墨的技法。墨于水的变化分为五色。画成作品，题款，盖章。就是完整的墨水画作品。

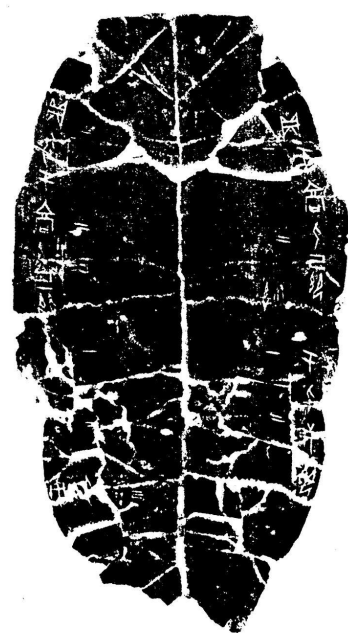
基本的水墨画，仅有水与墨，黑与白色，但进阶的水墨画，也有工笔花鸟画，色彩缤纷。后者有时也称为彩墨画。在中国画中，以中国画特有的材料之一，墨为主要原料加以清水的多少引为浓墨、淡墨、干墨、湿墨、焦墨等，画出不同浓淡（黑、白、灰）层次。别有一番韵味称为“墨韵”。而形成水墨为主的一种绘画形式。”

Model >>> 中国水墨画：流动的诗意与东方美学



水墨画，以其独特的韵味和深邃的意境，成为中国文化中不可或缺的一部分。它不仅是一种绘画形式，更是中国人审美情趣和文化底蕴的集中体现。在中国画中，水墨画是最具代表性的一种艺术形式，它的魅力不仅仅在于其视觉上的美感，更在于其所蕴含的哲学思想和文化内涵。

水墨画的起源可以追溯到中国古代的甲骨文和青铜器时代。当时的人们用墨汁在龟甲和兽骨上刻划文字和图案，这些早期的绘画作品具有浓厚的装饰性和象征性。随着时间的推移，水墨画逐渐发展成为一种独立的艺术形式，并在唐代达到了高峰。唐代的水墨画家们以“神韵”为追求目标，注重画面的气韵生动、意境深远，开创了后世水墨画的发展方向。



宋代是水墨画发展的又一重要时期。这一时期的画家们注重笔墨的运用，强调笔墨的个性表现和情感表达，使得水墨画更加注重精神内涵和艺术价值。明清时期，水墨画逐渐走向成熟，形成了独具特色的风格和流派。明代画家徐渭和清代画家石涛都是水墨画史上的巨匠，他们的作品不仅技法精湛，而且具有极高的艺术价值和历史意义。



现代水墨画则是在传统水墨画的基础上发展而来的。随着时代的变迁，水墨画也不断地吸收着新的艺术元素和表现手法，如西方绘画的色彩观念和构图方式等。现代水墨画的特点是更加注重个性化和多元化，画家们不再局限于传统的题材和表现手法，而是更多地关注现实生活和当代社会问题，通过作品表达自己对世界的思考和感悟。



总之，水墨画作为中国文化的重要组成部分，已经成为了世界范围内备受推崇的艺术形式。它所蕴含的独特美学价值和文化内涵，不仅让人们感受到了中国传统文化的博大精深，也为世界艺术的发展做出了重要的贡献。

【这里发现模型只是根据推断应该插入的图片内容搜索相应图片，部分场景下存在图片实际内容与预期不相符的情况。而应该插入的图片内容由 LLM 对文本进行总结生成，在大多数情况下符合预期，不会有太大的偏差。】

测试用例 2：自由场景图文创作

User >>> 根据以下标题：欧洲城市攻略：科隆(Cologne)”，创作长文章，字数不少于 800 字。请结合以下文本素材：

“科隆是德国西部莱茵河畔名城和重工业城市。人口 95.3 万（1983），仅次于柏林、汉堡和慕尼黑。科隆市是德国的第四大城市，人口 97 万。这是一座古老而美丽的城市，也是一座现代化气息极强的大都市。市中高楼大厦鳞次栉比，商店比比皆是，各种商品琳琅满目，是一个繁华的商业城市。中世纪科隆已经成为一个重要的教会中心和艺术知识中心。科隆在二战期间遭到严重破坏。今天它以其文化和建筑遗产以及许多重要的国际性活动而闻名。科隆也有天主教大主教，是天主教科隆大主教座堂所在地。”

Model >>> init

【模型首先根据 prompt 生成文字结果】

Model >>> init

欧洲城市攻略：科隆(Cologne)”

德国西部莱茵河畔名城和重工业城市——科隆。人口 95.3 万（1983），仅次于柏林、汉堡和慕尼黑。是德国的第四大城市，人口 97 万。这是一座古老而美丽的城市，也是一座现代化气息极强的大都市。市中高楼大厦鳞次栉比，商店比比皆是，各种商品琳琅满目，是一个繁华的商业城市。中世纪科隆已经成为一个重要的教会中心和艺术知识中心。二战期间遭到严重破坏。今天它以其文化和建筑遗产以及许多重要的国际性活动而闻名。

【之后对文本结果进行分句】

[UNUSED_TOKEN_146]user

给定文章"<Seg0> 欧洲城市攻略：科隆(Cologne)”

<Seg1> 德国西部莱茵河畔名城和重工业城市——科隆。人口 95.3 万（1983），仅次于柏林、汉堡和慕尼黑。是德国的第四大城市，人口 97 万。这是一座古老而美丽的城市，也是一座现代化气息极强的大都市。市中高楼大厦鳞次栉比，商店比比皆是，各种商品琳琅满目，是一个繁华的商业城市。中世纪科隆已经成为一个重要的教会中心和艺术知识中心。二战期间遭到严重破坏。今天它以其文化和建筑遗产以及许多重要的国际性活动而闻名。

<Seg2> 科隆也有天主教大主教，是天主教科隆大主教座堂所在地。

【根据插图数量参数选择适合插入图片的分句间隔】

" 根据上述文章，选择适合插入图像的 6 行[UNUSED_TOKEN_145]

[UNUSED_TOKEN_146]assistant

适合插入图像的行是

适合插入图像的行是<Seg1>, <Seg5>, <Seg7>, <Seg9>, <Seg11>, <Seg25>.

[1, 5, 7, 9, 11, 25]

【根据指定的图片插入位置推断图片主要内容】

给定文章"<Seg0> 欧洲城市攻略：科隆(Cologne)"

<Seg1> 德国西部莱茵河畔名城和重工业城市——科隆。人口 95.3 万（1983），仅次于柏林、汉堡和慕尼黑。是德国的第四大城市，人口 97 万。这是一座古老而美丽的城市，也是一座现代化气息极强的大都市。市中高楼大厦鳞次栉比，商店比比皆是，各种商品琳琅满目，是一个繁华的商业城市。中世纪科隆已经成为一个重要的教会中心和艺术知识中心。二战期间遭到严重破坏。今天它以其文化和建筑遗产以及许多重要的国际性活动而闻名。

<Seg2> 科隆也有天主教大主教，是天主教科隆大主教座堂所在地。

" 给出适合在<Seg1>后插入的图像对应的标题。[UNUSED_TOKEN_145]

[UNUSED_TOKEN_146]assistant

标题是"

科隆大教堂的全景照片，展示了其壮观的建筑和周围的景色。

【以前文推断的预定图片内容作为 prompt 继续推断后面的插图内容】

现在<Seg1>后插入图像对应的标题是"科隆大教堂的全景照片，展示了其壮观的建筑和周围的景色。"。给出适合在<Seg5>后插入的图像对应的标题。

标题是"科隆大教堂内部的照片，展示了其壮丽的内部结构和精美的装饰。

【同理】

现在<Seg1>后插入图像对应的标题是"科隆大教堂的全景照片，展示了其壮观的建筑和周围的景色。"，<Seg5>后插入图像对应的标题是"科隆大教堂内部的照片，展示了其壮丽的内部结构和精美的装饰。"。给出适合在<Seg7>后插入的图像对应的标题。[UNUSED_TOKEN_145]

[UNUSED_TOKEN_146]assistant

标题是"

莱茵河风光带的风景照片，展示了其美丽的天际线和沿岸的建筑。

【完成推断后获取全部图片内容，并根据内容搜索对应的图片，插入到文本中】

{1: '科隆大教堂的全景照片，展示了其壮观的建筑和周围的景色。'; 5: '科隆大教堂内部的照片，展示了其壮丽的内部结构和精美的装饰。'; 7: '莱茵河风光带的风景照片，展示了其美丽的天际线和沿岸的建筑。'; 9: '老城区的照片，展示了其独特的建筑风格和宁静的氛围。'; 11: '科隆猪脚的照片，展示了其金黄酥脆的外皮和鲜嫩多汁的肉。'; 25: '舒适的酒店房间的照片，展示了其温馨的氛围和豪华的设施。'}

根据给定上下文和候选图像，选择合适的配图：

欧洲城市攻略：科隆(Cologne)”

德国西部莱茵河畔名城和重工业城市——科隆。人口 95.3 万（1983），仅次于柏林、汉堡和慕尼黑。是德国的第四大城市，人口 97 万。这是一座古老而美丽的城市，也是一座现代化气息极强的大都市。市中高楼大厦鳞次栉比，商店比比皆是，各种商品琳琅满目，是一个繁华的商业城市。中世纪科隆已经成为一个重要的教会中心和艺术知识中心。二战期间遭到严重破坏。今天它以其文化和建筑遗产以及许多重要的国际性活动而闻名。

候选图像包括：A.<image>

B.<image>

C.<image>

D.<image>[UNUSED_TOKEN_145]

[UNUSED_TOKEN_146]assistant

最合适的图是

D[UNUSED_TOKEN_145]</s>

【图文不符实例】

文本：科隆猪脚（Kölsch Kalt）是科隆的传统美食之一，通常是用啤酒制成的，口感清爽爽口，非常适合在夏天享用。

推断图片标题：科隆猪脚的照片，展示了其金黄酥脆的外皮和鲜嫩多汁的肉。

实际图片：



【这实际上是一块牛排的照片。】

【上述例子中给出的错误信息有时并不容易分辨，在另一个对俾斯麦的人物介绍任务中模型给出了相差甚远的答案：】

推断图片标题：'俾斯麦与英国首相迪斯累利会面的场景。'

实际图片：



【这实际上是俾斯麦和李鸿章会面时的照片。】

【可见模型能够高质量地根据文本总结推断出应插入的图片内容,但并不总是能够搜索到正确内容的图片。】

5. 通过 HF 进行本地模型下载

由于正在老家省亲，并没有合适的生产开发环境和稳定的网络环境，因此此项只能先跳过，日后条件允许时再弥补。