南京大学
NANJING UNIVERSITY

快速入门GRADIO
你的AI展示利器

陈如莹

南京大学人工智能学院

2024年7月

什么是前端和后端


{ FRONT-END }
CSS
HTML
Javascript
VS
< BACK-END >
PHP
.NET
ExpressionEngine
知乎用户

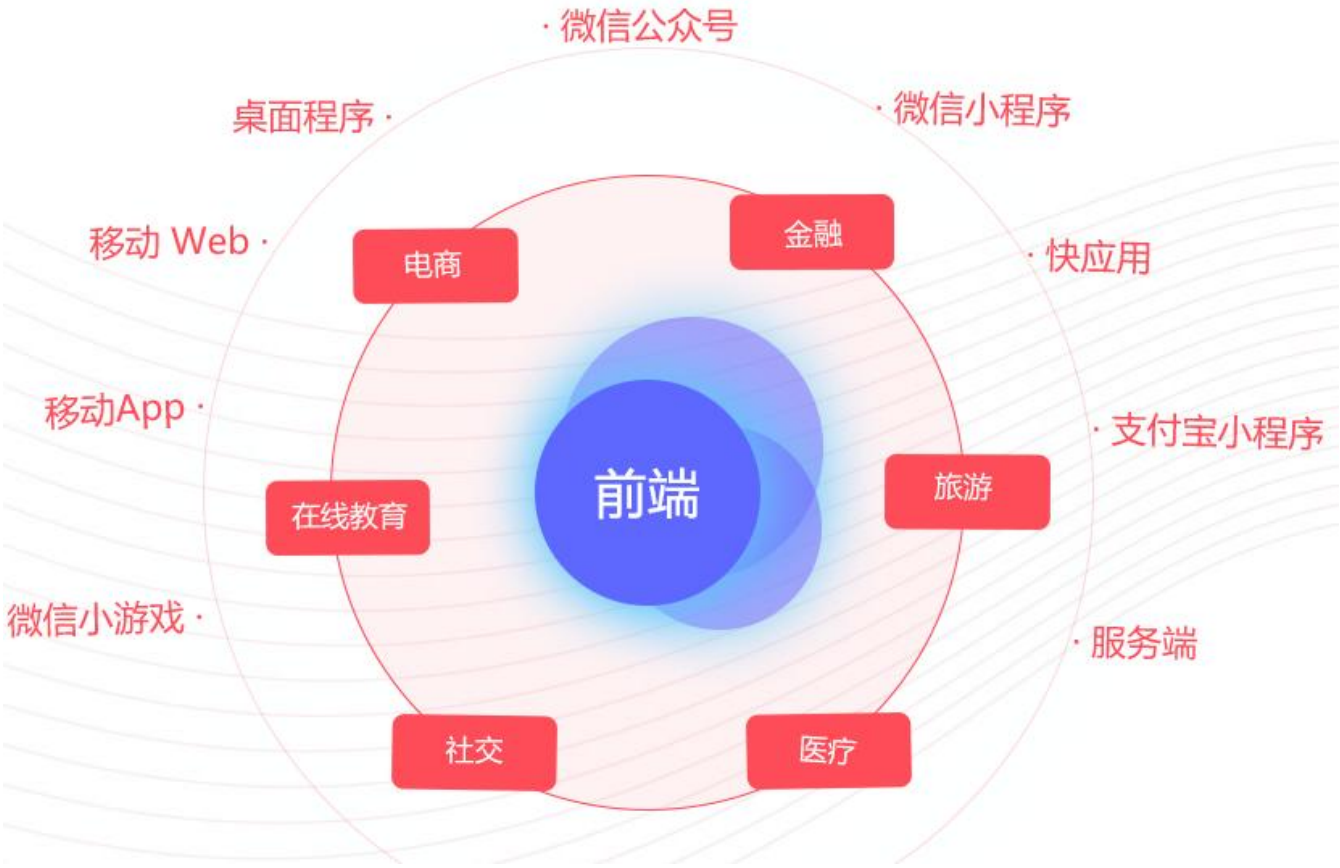
什么是前端和后端

- 前端：“客户端开发”，与用户交互的部分
  - 一般为图形界面的形式，

    也包含语音、手势等其他形式
- 后端：“服务器端开发”
  - 简而言之“系统“后面发生的部分
- 前端和后端之间的工作
  - 缓存数据（本地存储、中间媒介服务器）等
  - 前后端的交互是通过API（应用程序接口）进行。
  - 后端提供API来处理数据和业务逻辑，
  - 而前端调用这些API来获取数据并在用户界面上展示。

什么是前端和后端

- 前端的分类
  - 按呈现形式分类：
    - Web
    - Mobile app
    - IoT devise
    - ...
  - 按语言分类：
    - CSS、HTML、JavaScript...
  - 按框架分类：
    - jQuery、React.js、Vue.js、Angular...

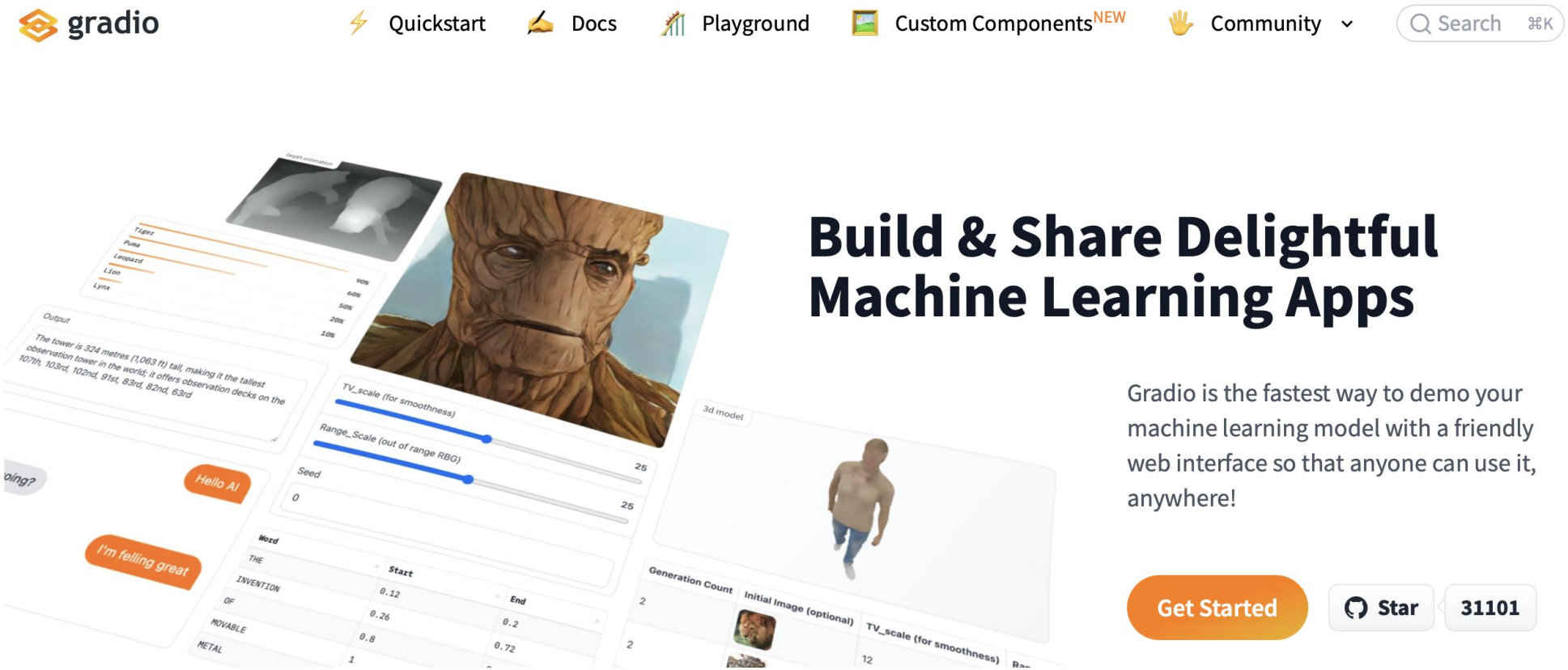


快速生成 AI 应用的框架

	gradio	streamit	dash
主要使用场景	可交互小 Demo	工作流、DashBoard	DashBoard、生产环境的复杂演示应用
上手难度	简单	简单	中等
组件丰富度	低	高	高
综合扩展性	低	中	高
Jupyter Notebook 内支持	是	否	是
是否完全开源	是	是	部分企业级功能未开源
github stars	13.4k	23.1k	18.2k
案例列表	github.com/gradio-app/a...	streamlit.io/gallery	dash.gallery/Portal/

GRADIO

- 一个开源的Python库，由斯坦福大学的四位博士程序员开发
- 初衷是为了快速向其他领域的专家展示AI算法的效果。因为大多数机器学习工程师只懂Python，不懂Web开发，构建完整的web演示还是有点难度的。
- 于2021年被 Hugging Face 收购



GRADIO

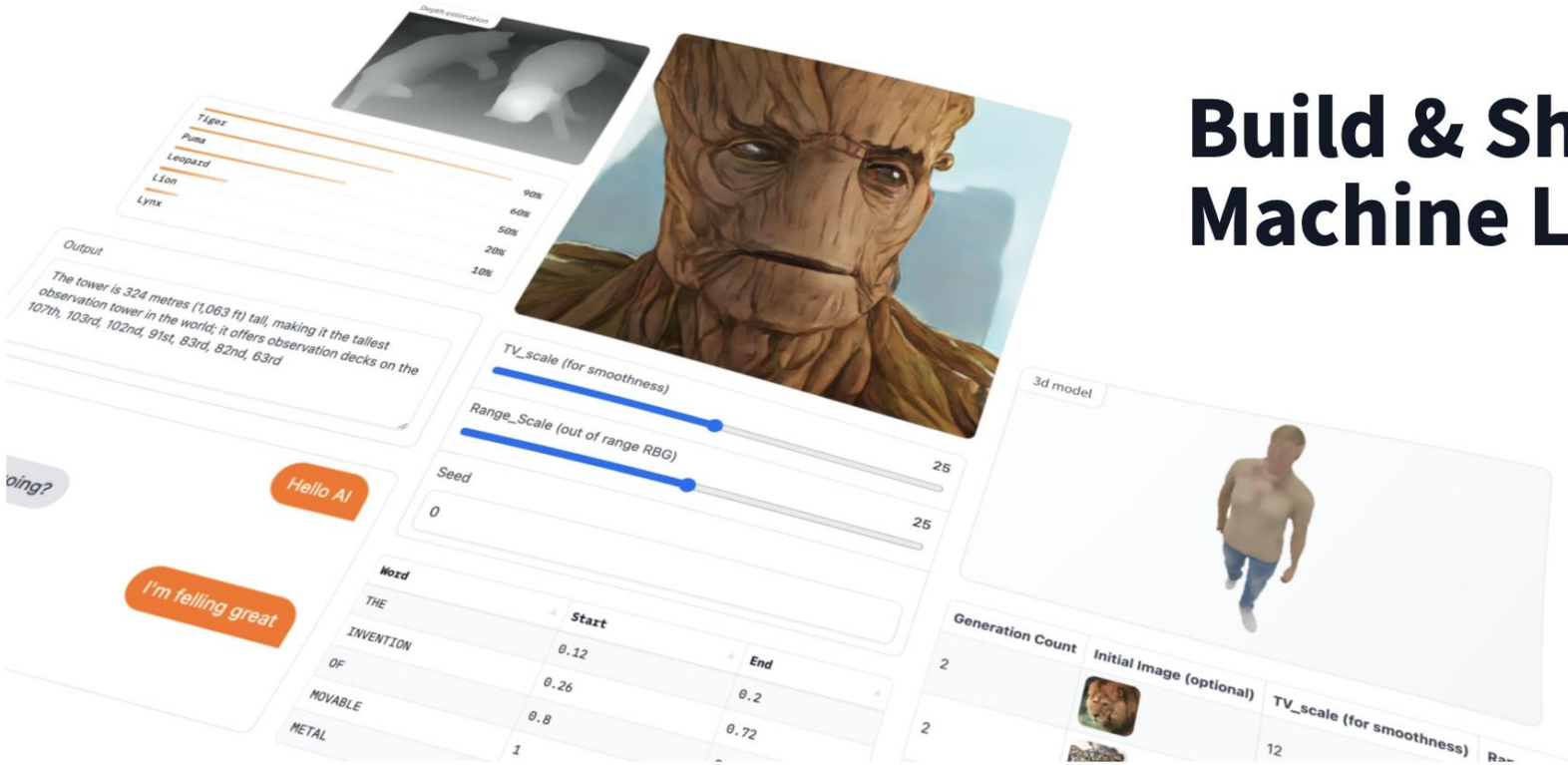
南京大学 NANJING UNIVERSITY

- 优点：
  - gradio 基于 svelte，它的初步使用非常简单
  - 组件的封装程度高，并且尤其适合机器学习模型相关的应用，比如 NLP、CV 领域的模型
  - 可以快速创建可共享的链接，从而方便将链接分享给用户。
  - 可以直接在 Jupyter Notebook 中展示。
- 缺点：
  - 组件的扩展性比较差
  - 使用场景较为简单

GRADIO

NANJING UNIVERSITY

gradio Quickstart Docs Playground Custom Components NEW Community Search ⌘K



Build & Share Delightful Machine Learning Apps

Gradio is the fastest way to demo your machine learning model with a friendly web interface so that anyone can use it, anywhere!

Get Started | Star 31101

GRADIO QUICKSTART

南京大学 NANJING UNIVERSITY

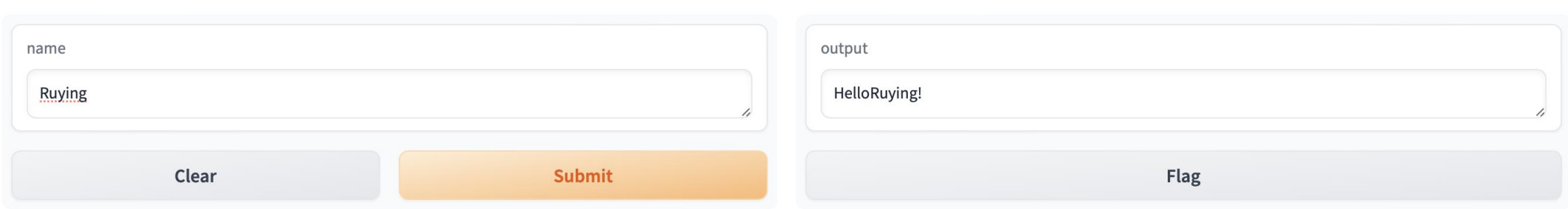
- 核心组件, Interface 类通过以下三个参数进行初始化：
  - fn：包装的函数
  - inputs：输入组件类型，（例如：“text”、"image）
  - ouputs：输出组件类型，（例如：“text”、"image）

GRADIO QUICKSTART

南京大学 NANJING UNIVERSITY

Code

```python
import gradio as gr

def greet(name):
    return "Hello " + name + "!"

demo = gr.Interface(fn=greet, inputs="textbox", outputs="textbox")

if __name__ == "__main__":
    demo.launch()

```



GRADIO QUICKSTART

南京大学 NANJING UNIVERSITY

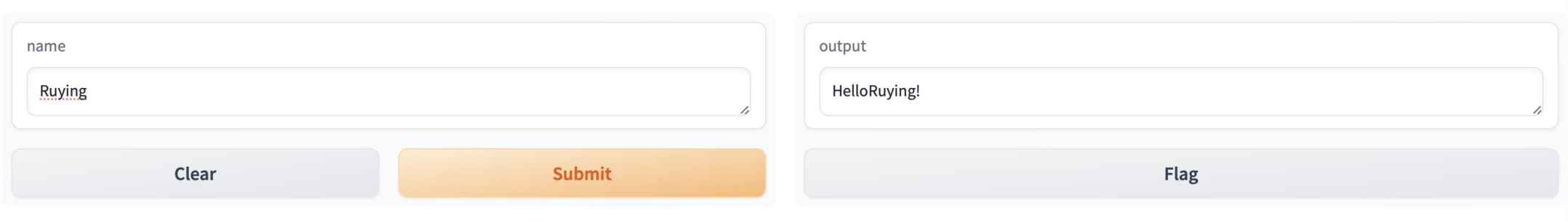
- 核心组件, 常用的基础模块
  - 应用界面：gr.Interface(简易场景), gr.Blocks(定制化场景)
  - 输入输出：gr.Image(图像), gr.Textbox(文本框), gr.DataFrame(数据框), gr.Dropdown(下拉选项), gr.Number(数字), gr.Markdown, gr.Files
  - 控制组件：gr.Button(按钮)
  - 布局组件：gr.Tab(标签页), gr.Row(行布局), gr.Column(列布局)

GRADIO QUICKSTART

南京大学 NANJING UNIVERSITY

Code

```python
# Write your own Gradio code here and see what it looks like!
import gradio as gr

def greet(name):
    return "Hello " + name + "!"
demo = gr.Interface(fn=greet, inputs=gr.Textbox(), outputs=gr.Textbox())
demo.launch()
```

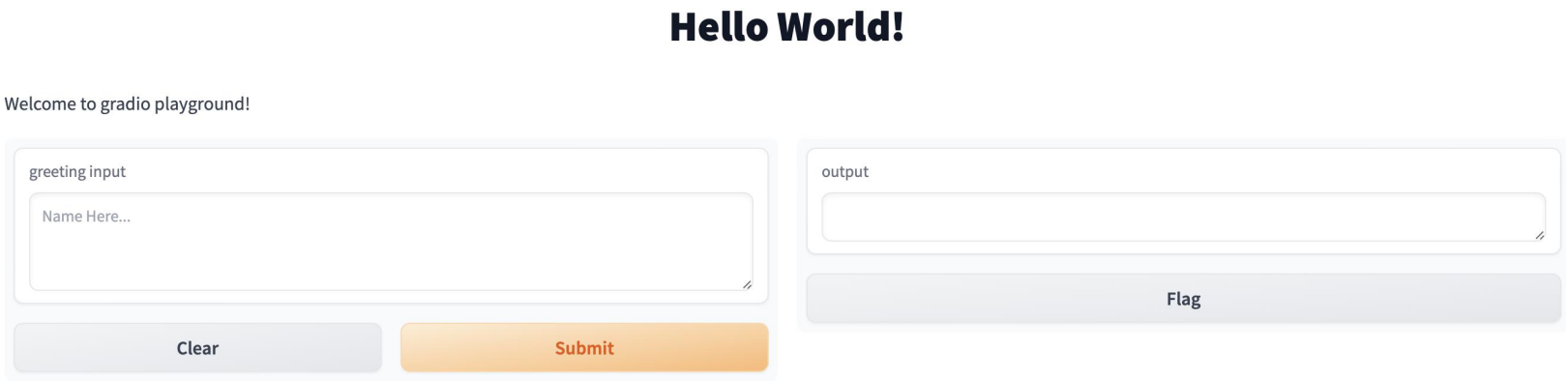


GRADIO QUICKSTART

南京大学 NANJING UNIVERSITY

Code

```python
# Write your own Gradio code here and see what it looks like!
import gradio as gr

def greet(name):
    return "Hello" + name + "!"
demo = gr.Interface(
    fn=greet,
    inputs=gr.Textbox(lines=3, placeholder="Name Here...",label="greeting input"),
    outputs=gr.Textbox(),
    title="Hello World!",
    description="Welcome to gradio playground!"
)
demo.launch()
```



GRADIO QUICKSTART

南京大学 NANJING UNIVERSITY

- 输入组件 (Inputs)
  - Audio：允许用户上传音频文件或直接录音。参数：source: 指定音频来源（如麦克风）、type: 指定返回类型。 示例：gr.Audio(source="microphone", type="filepath")
  - Checkbox：提供复选框，用于布尔值输入。参数：label: 显示在复选框旁边的文本标签。 示例：gr.Checkbox(label="同意条款")
  - CheckboxGroup：允许用户从一组选项中选择多个。参数：choices: 字符串数组，表示复选框的选项、label: 标签文本。示例：gr.CheckboxGroup(["选项1", "选项2", "选项3"], label="选择你的兴趣")

GRADIO QUICKSTART

南京大学 NANJING UNIVERSITY

- 输入组件 (Inputs)
  - ColorPicker：用于选择颜色，通常返回十六进制颜色代码。参数：default: 默认颜色值。示例：gr.ColorPicker(default="#ff0000")
  - Dataframe：允许用户上传CSV文件或输入DataFrame。参数：headers: 列标题数组、row_count: 初始显示的行数。示例：gr.Dataframe(headers=["列1", "列2"], row_count=5)
  - Dropdown：下拉菜单，用户可以从中选择一个选项。参数：choices: 字符串数组，表示下拉菜单的选项、label: 标签文本。示例：gr.Dropdown(["选项1", "选项2", "选项3"], label="选择一个选项")

GRADIO QUICKSTART

南京大学 NANJING UNIVERSITY

- 输入组件 (Inputs)
  - File：用于上传任意文件，支持多种文件格式。参数：file_count: 允许上传的文件数量，如"single"或"multiple"、type: 返回的数据类型，如"file"或"auto"。示例：gr.File(file_count="single", type="file")
  - Image：用于上传图片，支持多种图像格式。参数：type图像类型，如pil。示例：gr.Image(type='pil')
  - Number：数字输入框，适用于整数和浮点数。参数：default: 默认数字、label: 标签文本。示例：gr.Number(default=0, label="输入一个数字")
  - Radio：单选按钮组，用户从中选择一个选项。参数：choices: 字符串数组，表示单选按钮的选项、label: 标签文本。示例：gr.Radio(["选项1", "选项2", "选项3"], label="选择一个选项")

GRADIO QUICKSTART

南京大学 NANJING UNIVERSITY

- 输入组件 (Inputs)
  - Slider：滑动条，用于选择一定范围内的数值。参数：minimum: 最小值、maximum: 最大值、step: 步长、label: 标签文本。示例：gr.Slider(minimum=0, maximum=10, step=1, label="调整数值")
  - Textbox：单行文本输入框，适用于简短文本。参数：default: 默认文本、placeholder: 占位符文本。示例：gr.Textbox(default="默认文本", placeholder="输入文本")
  - Textarea：多行文本输入区域，适合较长的文本输入。参数：lines: 显示行数、placeholder: 占位符文本。示例：gr.Textarea(lines=4, placeholder="输入长文本")

GRADIO QUICKSTART

南京大学 NANJING UNIVERSITY

- 输入组件 (Inputs)
  - Time：用于输入时间。参数：label: 标签文本。示例：gr.Time(label="选择时间")
  - Video：视频上传组件，支持多种视频格式。参数：label: 标签文本。示例：gr.Video(label="上传视频")
  - Data：用于上传二进制数据，例如图像或音频的原始字节。参数：type: 数据类型，如"auto"自动推断。示例：gr.Data(type="auto", label="上传数据")

GRADIO QUICKSTART

南京大学 NANJING UNIVERSITY

- 输出组件 (Outputs)
  - Audio：播放音频文件。参数：type 指定输出格式。示例：gr.Audio(type="auto")
  - Carousel：以轮播方式展示多个输出Dataframe：展示Pandas DataFrame，适用于表格数据。参数：type 指定返回的DataFrame类型。示例：gr.Dataframe(type="pandas")
  - Gallery：以画廊形式展示一系列图像。
  - HTML：展示HTML内容，适用于富文本或网页布局。
  - Image：展示图像。参数：type 指定图像格式。 示例：gr.Image(type="pil")
  - JSON：以JSON格式展示数据，便于查看结构化数据。

GRADIO QUICKSTART

- 输出组件 (Outputs)
  - KeyValues：以键值对形式展示数据。
  - Label：展示文本标签，适用于简单的文本输出。
  - Markdown：支持Markdown格式的文本展示。
  - Plot：展示图表，如matplotlib生成的图表。
  - Text：用于显示文本，适合较长的输出。
  - Video：播放视频文件。

GRADIO 与预训练模型相结合

南京大学 NANJING UNIVERSITY

- 选择模型源
  - 常见的来源包括TensorFlow Hub、PyTorch Hub或Hugging Face模型库。
- 加载模型：
  - 根据所选模型的文档加载模型。这通常涉及导入相应的库并加载预训练权重。
- 创建处理函数：
  - 编写一个处理函数，该函数接收Gradio输入，并使用模型进行预测或处理，然后返回输出。
- 构建Gradio界面：
  - 根据模型的输入输出类型，选择合适的Gradio输入输出组件。然后将处理函数绑定到Gradio界面。

GRADIO 与预训练模型相结合

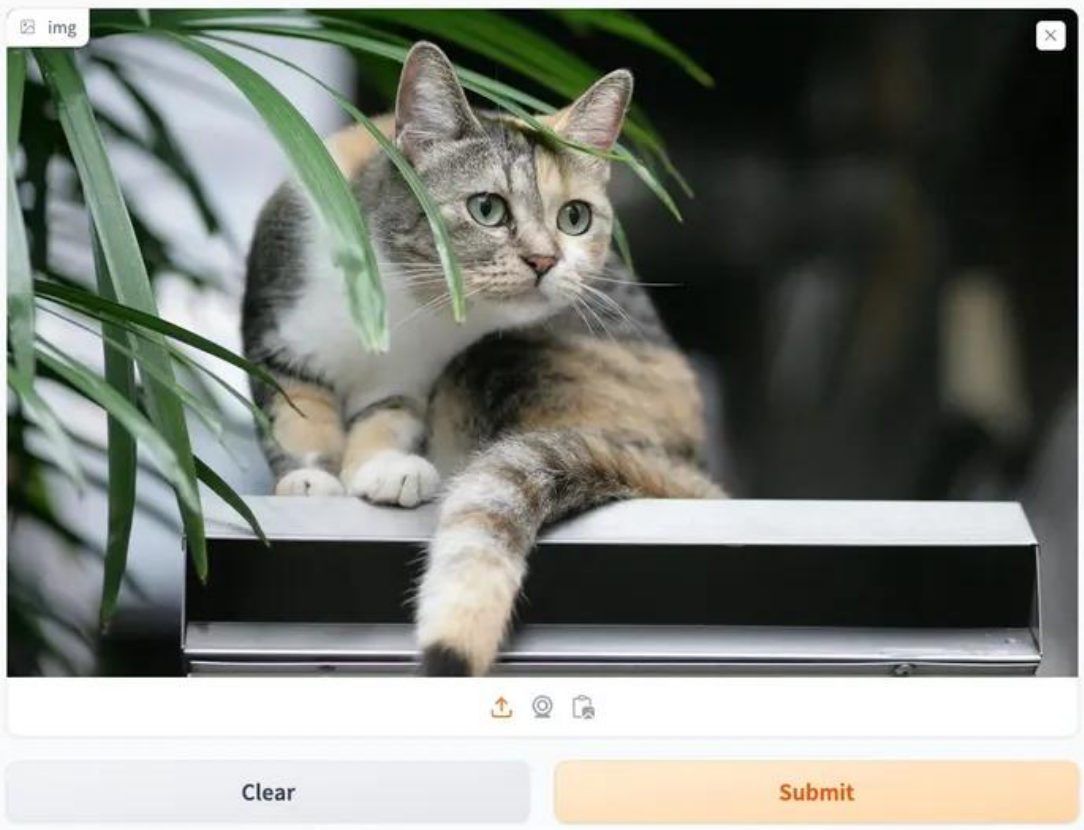
南京大学 NANJING UNIVERSITY

- 使用Hugging Face的预训练图像分类模型。

```
import gradio as gr
from transformers import pipeline

# 加载预训练模型
model = pipeline('image-classification')

# 定义处理函数
def classify_image(img):
    return {i['label']: i['score'] for i in model(img)}

# 创建Gradio界面
iface = gr.Interface(
    fn=classify_image,
    inputs=gr.Image(type="pil"),
    outputs=gr.Label(num_top_classes=5))
iface.launch()
```

GRADIO 与预训练模型相结合

- 使用Hugging Face的预训练图像分类模型。并显示前五个最可能的类别



GRADIO 使用CHATINTERFACE

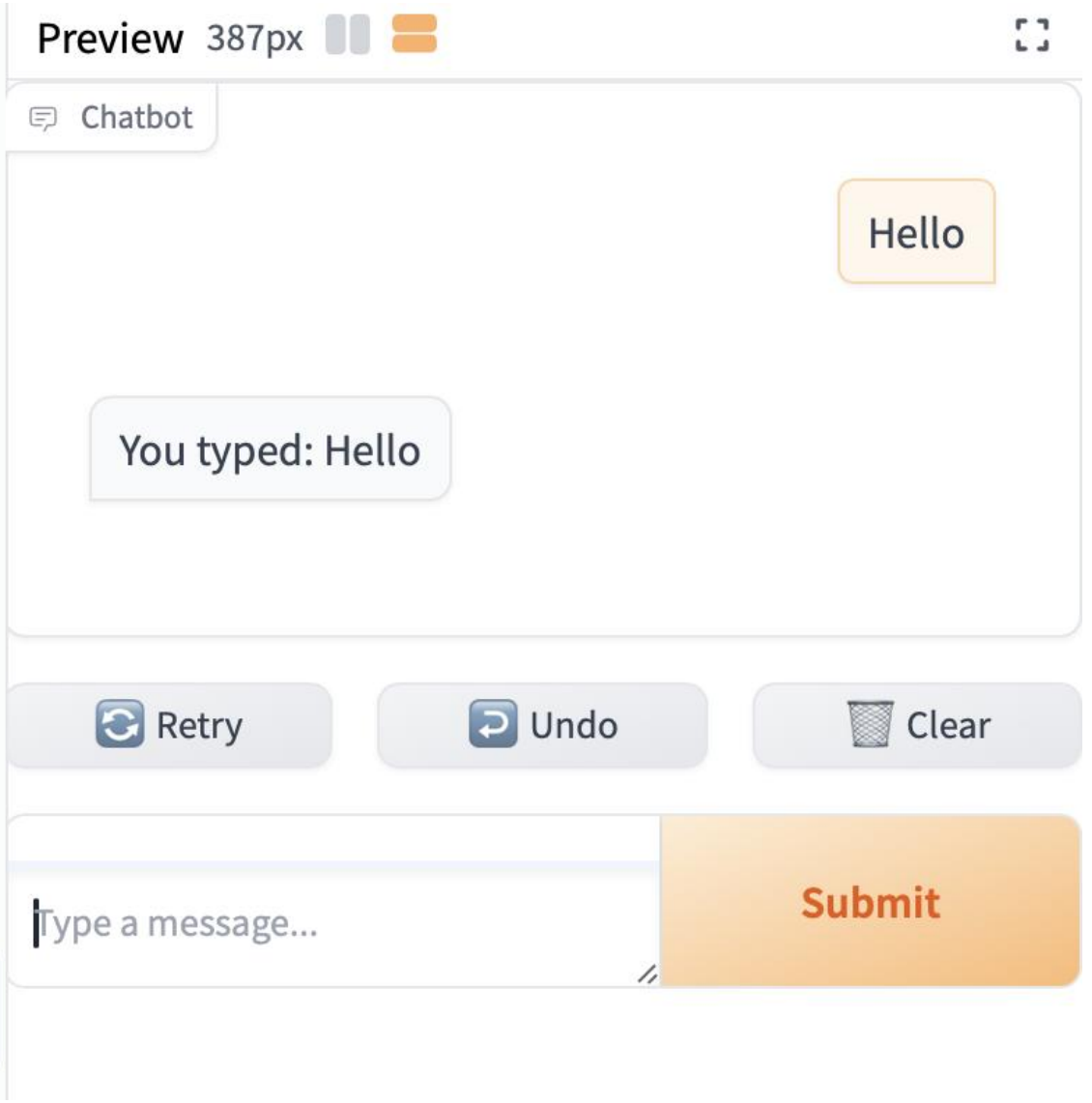
- Defining a chat functio:
  - message: a str representing the user's input.
  - history: a list of list representing the conversations up until that point. Each inner list consists of two str representing a pair: [user input, bot response].

GRADIO 使用CHATINTERFACE

南京大学 NANJING UNIVERSITY

Code

```python
import time
import gradio as gr


def slow_echo(message, history):
    for i in range(len(message)):
        time.sleep(0.05)
        yield "You typed: " + message[: i + 1]


demo = gr.ChatInterface(slow_echo)

if __name__ == "__main__":
    demo.launch()
```



开发一个CHATBOT系统的过程和细节

- 大模型线上服务端部署
- 前端界面流式的交互
  - 推理过程中，不断返回下一个词的动态交互体验
- 服务端的并发控制
  - 大模型的开销很大，不能对每个请求都实时计算
- 知识库文本向量化、分词向量化
- 向量数据库的使用
- ...

https://www.modelscope.cn/studios/qwen/Qwen-1_8B-Chat-Demo

TONGYI Qwen

Qwen1.5-1.8B-Chat Bot👾

通义千问1.5-1.8B（Qwen1.5-1.8B） 是阿里云研发的通义千问大模型系列的70亿参数规模的模型。

VLLM 和 VLLM-CHAT区别（以QWEN为例）

南京大学 NANJING UNIVERSITY

- 1.8B预训练版本，训练数据：
  - 语料：百度文库
  - 输入：英国航空，中文简称英航，是英国的国家航空公司，也是寰宇一家的创始成员及国际航空集团旗下子公司。<|endoftext|>
  - 输出：英航的主要枢纽为伦敦希思罗机场及伦敦盖特威克机场。英航是欧洲第二大的航空公司、西欧最大的航空公司及全球三间其中一间曾拥有协和客机的航空公司，其余两间为法国航空和新加坡航空。<|endoftext|>

VLLM 和 VLLM-CHAT区别（以QWEN为例）

南京大学 NANJING UNIVERSITY

- 1.8B-Chat版本，基于1.8B预训练版本进行微调（SFT，S监督学习，FT微调）

  训练数据：

  - 输入：<|im_start|>system\nYou are a helpful assitant.\n<|im_end|> \n<|im_start|>user\n了解英国航空吗？ \n<|im_end|><|im_start|>assitant:历史回答A\n<|im_end|> \n<|im_start|>user\n历史提问B？ \n<|im_end|><|im_start|>assitant:历史回答B\n<|im_end|> \n<|im_start|>user\n了解英国航空么？ \n<|im_end|><|im_start|>assitant:\n<|endoftext|>
  - 输出：英国航空，中文简称英航，是英国的国家航空公司。<|im_end|><|endoftext|>

VLLM 和 VLLM-CHAT区别（以QWEN为例）

- 1.8B-Chat版本，基于1.8B预训练版本进行微调（SFT，S监督学习，FT微调）

  训练数据：

  - 输入：<|im_start|>system\nYou are a helpful assitant.\n<|im_end|> \n<|im_start|>user\n了解英国航空吗？ \n<|im_end|><|im_start|>assitant:历史回答A\n<|im_end|> \n<|im_start|>user\n历史提问B？ \n<|im_end|><|im_start|>assitant:历史回答B\n<|im_end|> \n<|im_start|>user\n了解英国航空么？ \n<|im_end|><|im_start|>assitant:\n<|endoftext|>
  - 输出：英国航空，中文简称英航，是英国的国家航空公司。<|im_end|><|endoftext|>

GRADIO 安装

南京大学 NANJING UNIVERSITY

- 安装与配置
  - Prerequisite: Gradio requires Python 3.8 or higher

```
pip install gradio
```

  - 更多安装信息：https://www.gradio.app/main/guides/installing-gradio-in-a-virtual-environment
  - 检查安装：安装完成后，可以通过运行以下Python代码来检查Gradio是否正确安装

```
import gradio as gr
print(gr.__version__)
# 4.15.0
```

GRADIO 安装

- 本地运行程序
  - 在 Terminal 运行 python your_file.py 指令
  - 打开 http://localhost:7860 即可看到网页效果
- 如果需要分享你的Demo
  - 设置 share=True in launch(), and a publicly accessible URL will be created for your demo.

GRADIO参考资料

南京大学 NANJING UNIVERSITY

- https://www.gradio.app/guides/quickstart
- https://github.com/gradio-app/gradio
- 吴恩达的公开课《使用Gradio构建生成式人工智能应用程序/Building Generative AI Applications with Gradio》，官方地址：https://www.deeplearning.ai/short-courses/building-generative-ai-applications-with-gradio/
- https://zhuanlan.zhihu.com/p/679668818
- 通义千问官方部署文档 https://github.com/QwenLM/Qwen?tab=readme-ov-file#deployment