Hugging Face入门与实践

💡 分享录屏链接：Hugging Face&Colab实战 2024年8月27日

> 往期回顾：
>
> 零基础学模型 - 从逻辑回归到神经网络 .pdf
>
> 零基础学模型 - 实现一个深度神经网络.pdf
>
> 零基础学模型 - 序列模型与循环神经网络.pdf
>
> 零基础学模型 - Transformer原理剖析
>
> 后期预告：
>
> - 大模型运行与部署
> - 大模型如何支持高并发

1. 存在价值及原因

You need Hugging Face- AI领域的“GitHub"

Bert，Transformer，LLM，模型众多

理论加部署，三年五载，未入门已放弃

希望自己的模型，被更多人发现，被更多人研究，被更多人引用

对于学者：模型没法落地，无法创造价值；

对于工程：空有想法和领域数据，没有好的模型

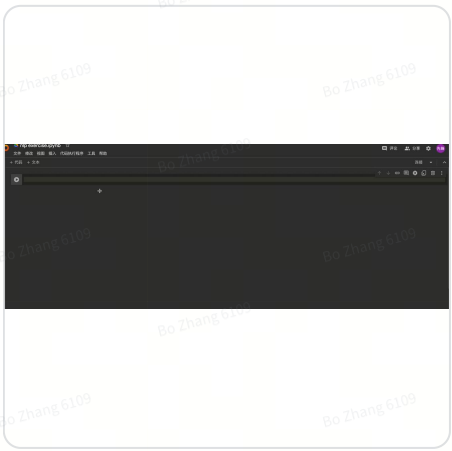
对于新手	对于大佬	学术	工程
拧包入住：即便你对数学一无所知，即便你对代码稀里糊涂，即便你对数据无从下手，你也可以通过HuggingFace开始AI！	**华山论剑**：给大佬们提供了一个舞台，贡献自己的模型，贡献自己的数据，与各位大佬PK，让模型被更多人认识和使用。	**合作共赢**：学术者研究模型，通过Hugging Face开源出去，让更多人更方便的使用	**合作共赢**：工程进行微调，业务探索应用到更多领域，合作共赢

- 简而言之：HuggingFace是菜鸟的福音，大佬的舞台，学术与工程的桥梁！
- 平台链接：https://huggingface.co/

2. 真这么方便吗？

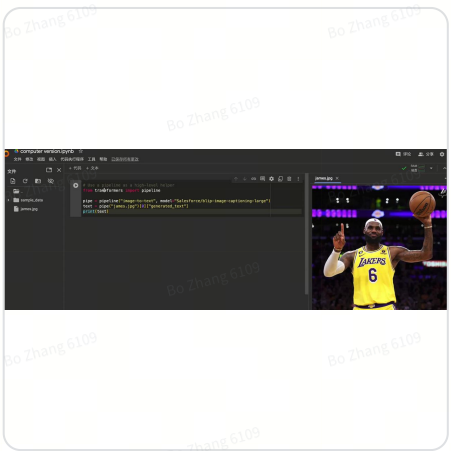
零样本分类

文本摘要

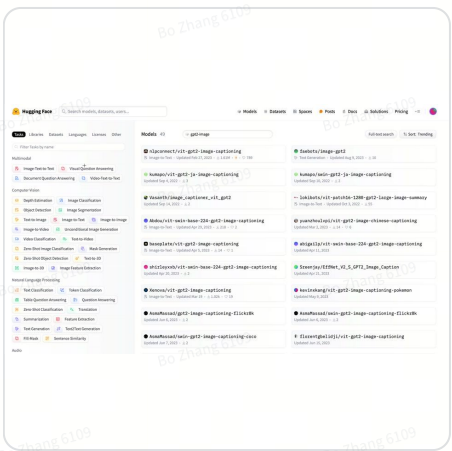


文本生成

图片转文字



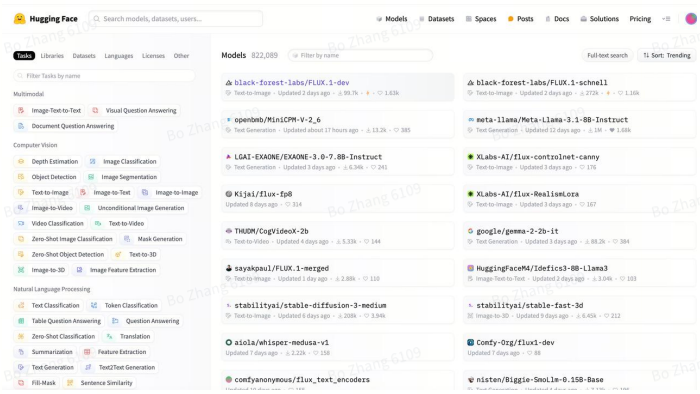
图片识别



文字转图片

3. 揭开神秘面纱

3.1 模型Model

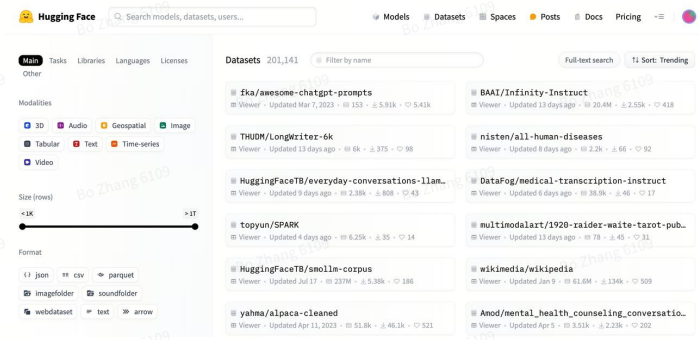


- **自然语言处理**：文本分类、文本生成，文本转语音、文本填空、文本摘要等；
- **计算机视觉**：图片分类、物体识别、文本转图片、题、图片转文本、图片特征提取等；
- **音频**：文本转语音、语音转文本、语音识别、语音分类等；
- **多模态**：Image-Text-to-Text、Video-Text-to_text

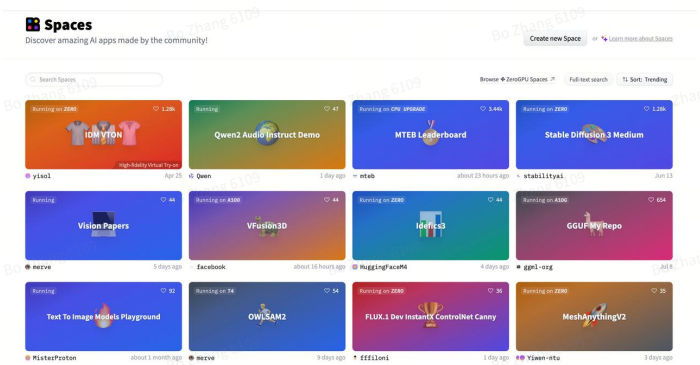
3.2 数据DataSet

互联网开源的一些最标准的语料库，可以用来训练或者微调你的模型，其特点为：

- 包含丰富的数据集：IMDB, CoNLL-2003和GLUE等；
- 简化数据集的下载、预处理操作；
- 提供数据集分割、采样和迭代器的功能；



3.3 应用Space



- 直观的查看模型效果
- 创建自己的Space

3.4 平台番外篇

2016年，法国创业者三名创业者Clément Delangue、Julien Chaumond 和 Thomas Wolf 在纽约成立了Hugging Face。

Hugging Face，它的第一个产品是一个聊天机器人。到2017年，Hugging Face聊天机器人拥有了独特的功能，并可以进行高效的对话。团队将其产品定位为为无聊青少年量身打造的个性鲜明的聊天机器人。2018年5月，完成种子融资。通过本轮融资，Hugging Face团队继续专注于以下领域：改进产品；建立一支优秀的工程师团队；深入研发自然语言对话，并撰写了几篇研究论文。虽然当时产品还没有带来可观的收入，**但团队对核心价值和技术共享的强调为Hugging Face创造了一个转折点。**

2018 年，Hugging Face迎来了关键时刻，Hugging Face的创始人在网上免费分享该应用的部分代码，其中一个重要的开源框架名为Transformers，目前已被下载超过一百万次。GitHub项目获得了上万颗星，这表明开源社区认为它很有价值。微软、谷歌和 Facebook 的研究人员一直在用它做实验，某些公司甚至在生产中使用了它。

Transformers 可用于各种任务，包括文本分类、信息提取、总结、文本生成和对话式人工智能。最终，Hugging Face团队迎来了一个转折点，将公司从一家不太赚钱的AI聊天机器人初创公司转变为未来估值十亿美元的独角兽。

在接下来的几年里，Hugging Face 团队继续专注于产品建设和社区发展，并取得了令人瞩目的成就：

> - Hugging Face 已成为扩展最快的社区和使用最广泛的机器学习平台！平台上有 10 万个预训练模型和 1 万个数据集，涵盖 NLP、语音、时间序列、强化学习、计算机视觉、生物、化学等领域。Hugging Face Hub 已发展成为机器学习构建者开发、协作和部署尖端模型的家园。
> - 目前有10000 多家公司使用 Hugging Face来构建机器学习技术，Hugging Face 帮助这些机器学习工程师和数据科学家团队节省了大量时间，加快了机器学习项目的进度。
> - Hugging Face还领导着 BigScience，一个专注于研究和构建大语言模型的合作研讨会。这项计划汇集了来自不同领域和背景的1000多名研究人员，BigScience 致力于训练世界上最大的开源多语言模型。

Hugging Face为什么能成功?

- 赶上Transformer在AI领域爆火；
- 军阀混战，谷歌TensorFlow Bert, FaceBook PyTorch，跑一下模型需要各种环境；
- 人家是一个舞台，不是一个工具，那后来的人，也只能在这上面玩。

4. 基本操作指南

4.1 数据选择与使用

4.1.1 DataSet选择

1. 语音数据集

💡 语音数据集的特征：

- 小时数：小时数体现了数据集的大小，类似 NLP 数据集中的训练样例数量；
- 领域：数据的来源，比如有声读物、播客、YouTube 还是金融会议等；
- 说话风格：
  - 叙述性（Narrated）：按照给定的文本朗读
  - 自发性（Spontaneous）：没有固定剧本的对话

Hugging Face Hub 上最受欢迎的英语语音识别数据集整理如下：

数据集	小时数	领域	说话风格	大小写	标点	许可证	推荐用途
LibriSpeech	960	有声读物	叙述性	❌	❌	CC-BY-4.0	学术基准测试
Common Voice 11	3000	维基百科	叙述性	✅	✅	CC0-1.0	非母语发言者
VoxPopuli	540	欧洲议会	演讲式	❌	✅	CC0	非母语发言者
TED-LIUM	450	TED 演讲	演讲式	❌	❌	CC-BY-NC-ND 3.0	技术主题
GigaSpeech	10000	有声读物、播客、YouTube	叙述性、自发性	❌	✅	apache-2.0	多领域鲁棒性
SPGISpeech	5000	金融会议	演讲式、自发性	✅	✅	User Agreement	完全格式化的转写
Earnings-22	119	金融会议	演讲式、自发性	✅	✅	CC-BY-SA-4.0	口音多样性
AMI	100	会议	自发性	✅	✅	CC-BY-4.0	嘈杂语音环境

Hugging Face Hub 上最受欢迎的中文语音识别数据集整理如下：

数据集	小时数	说明
Chinese_dialect	25000	各种方言
Mandarin chinese	15000	普通话
miexed_speech_chinese_english	2000	普通话和英语混合

更多中文语音数据集，可以看这里：https://huggingface.co/datasets?modality=modality:audio&sort=trending&search=chinese

多语言语音识别数据集，省略了训练小时数列（因为这取决于每个数据集有哪些语言，占比多少），改成了每个数据集的语言数量：

数据集	语言数量	领域	说话风格	大小写	标点	许可证	推荐用途

Multilingual LibriSpeech	6	有声读物	叙述性	❌	❌	CC-BY-4.0	学术基准测试
Common Voice 13	108	维基百科文本和众筹的言语	叙述性	✅	✅	CC0-1.0	多样化的发言者集合
VoxPopuli	15	欧洲议会	自发性	❌	✅	CC0	欧洲语言
FLEURS	101	欧洲议会	自发性	❌	❌	CC-BY-4.0	多语言评估

2. 文本数据集

数据集	数据集描述
IMDB数据集	大型电影评论数据集，为用户提供了超过5万条的电影评论
Amazon Polarity数据集	该数据集包含来自亚马逊的超过3500万条产品评论
Common Voice数据集	此数据集是音频数据和文本数据的混合。包含超过9000小时的录音信息及其书面记录文本

4.1.2 DataSet使用

```
# 导入数据集
from datasets import load_dataset
```

加载托管的数据集：

```
dataset = load_dataset("imdb")
```

加载远程数据集：

```
url = "https://github.com/crux82/squad-it/raw/master/"
data_files = {
    "train": url + "SQuAD_it-train.json.gz",
    "test": url + "SQuAD_it-test.json.gz",
}
squad_it_dataset = load_dataset("json", data_files=data_files, field="data")
```

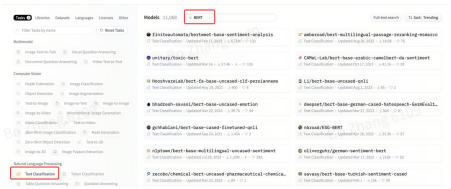
加载本地数据集：

```
load_dataset("csv",data_files="my_file.csv")
```

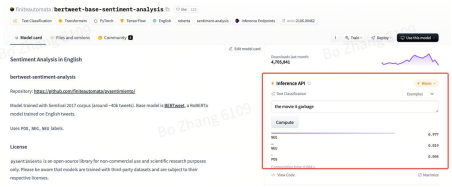
4.2 模型选择与使用

4.2.1 Model选择

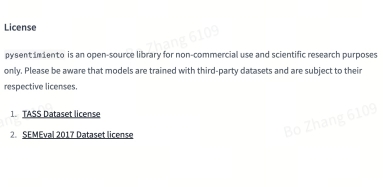
1.选择模型



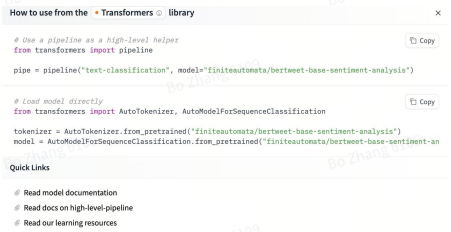
2.看看效果



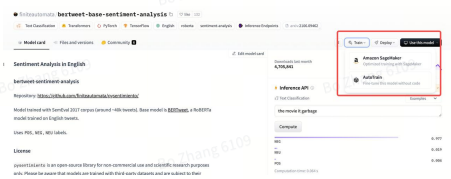
3.查看协议



4.开始使用



5.微调模型



https://neptune.ai/blog/hugging-face-pre-trained-models-find-the-best

4.2.2 模型使用

> repo： https://github.com/huggingface/transformers

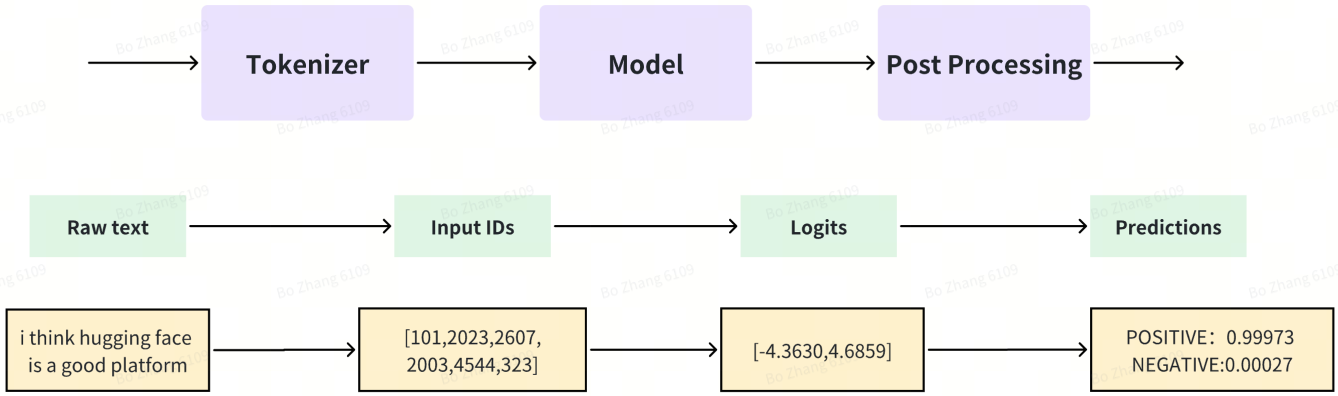
Transformers 库中最基本的对象是 pipeline() 函数。它将模型与其必要的预处理和后处理步骤连接起来，使我们能够通过直接输入任何文本并获得最终的答案：

```
from transformers import pipeline

classifier = pipeline("sentiment-analysis")
classifier("I think hugging face is a good platform")
```

```
No model was supplied, defaulted to distilbert/distilbert-base-uncased-finetuned-sst-2-english and revision af0f99b (https://huggingface.co/distilbert/distilbert-base-uncased-finetuned-sst-2-english).
```

```
Using a pipeline without specifying a model name and revision in production is not recommended.
[{'label': 'POSITIVE', 'score': 0.9997380375862122}]
```

pipeline将三步合在一起: preprocessing, passing the inputs through the model, and postprocessing:



将一些文本传递到pipeline时涉及三个主要步骤：

1. 文本被**预处理**为模型可以理解的格式。
2. 预处理的输入被传递给模型。
3. 模型处理后输出最终人类可以理解的结果。

4.3 选择评价指标

查看有哪些评价指标：

```
from datasets import list_metrics
metrics_list = list_metrics()
len(metrics_list)
print(metrics_list)
```

> 输出：274个，具体为：
>
> ['accuracy', 'bertscore', 'bleu', 'bleurt', 'brier_score', 'cer', 'character', 'charcut_mt', 'chrf', 'code_eval', 'comet', 'competition_math', 'confusion_matrix', 'coval', 'cuad',.....]

加载评价指标：

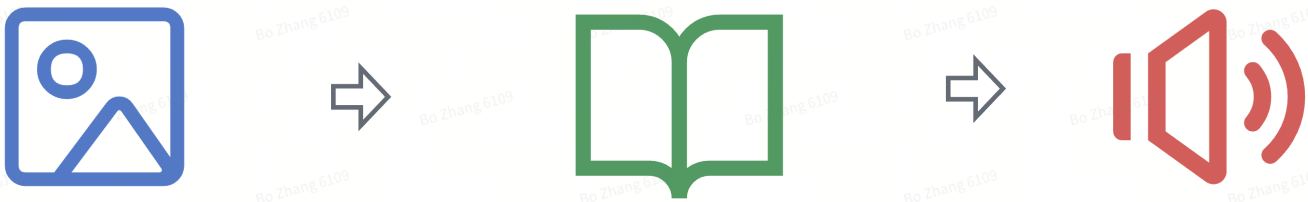
```
from datasets import load_metric
```

```
metric = load_metric('wer')
```

5. 搓搓激动的小手

推荐使用Colaboratory，浏览器上就可以开始跑模型。相关介绍：写在最前面 - 工具推荐。

5.1 验证想法



- 首先，Image-to-text，把图片转换为文字；
- 然后，Generate Text，生成故事；
- 最后，text-to-audio，文字转换为语音

5.1.1 图片转换为文字

```
from transformers import pipeline

def img2text(url):
  image_to_text = pipeline("image-to-text",model="Salesforce/blip-image-captioning-base")
  text = image_to_text(url)[0]["generated_text"]
  print(text)
  return text

img2text("basketball.jpg")
```

```python
from transformers import pipeline

def img2text(url):
  image_to_text = pipeline("image-to-text",model="Salesforce/blip-image-captioning-base")
  text = image_to_text(url)[0]["generated_text"]
  print(text)
  return text

img2text("basketball.jpg")
```

```
/usr/local/lib/python3.10/dist-packages/transformers/generation/utils.py:1249: UserWarning: Using the model-agnostic default `max_length` (=20) to control
  warnings.warn(
two boys playing basketball
'two boys playing basketball'
```

5.1.2 故事生成

```python
from transformers import pipeline

generator = pipeline("text-generation")
generator("two boys playing basketball")
```

```
No model was supplied, defaulted to openai-community/gpt2 and revision 6c0e608 (https://huggingfac
Using a pipeline without specifying a model name and revision in production is not recommended.
/usr/local/lib/python3.10/dist-packages/huggingface_hub/utils/_token.py:89: UserWarning:
The secret `HF_TOKEN` does not exist in your Colab secrets.
To authenticate with the Hugging Face Hub, create a token in your settings tab (https://huggingfac
You will be able to reuse this secret in all of your notebooks.
Please note that authentication is recommended but still optional to access public models or datas
  warnings.warn(
config.json: 100% 665/665 [00:00<00:00, 17.2kB/s]
model.safetensors: 100% 548M/548M [00:05<00:00, 123MB/s]
generation_config.json: 100% 124/124 [00:00<00:00, 1.60kB/s]
tokenizer_config.json: 100% 26.0/26.0 [00:00<00:00, 683B/s]
vocab.json: 100% 1.04M/1.04M [00:00<00:00, 6.10MB/s]
merges.txt: 100% 456k/456k [00:00<00:00, 6.85MB/s]
tokenizer.json: 100% 1.36M/1.36M [00:00<00:00, 15.3MB/s]
Setting `pad_token_id` to `eos_token_id`:50256 for open-end generation.
[{'generated_text': 'two boys playing basketball at the time of the crash, but police say they
have received no reports of anyone being injured. One of the juveniles was hit by the other
vehicle before he was hit. The boys, who were wearing helmets and white shirt,'}]
```

生成内容如下：

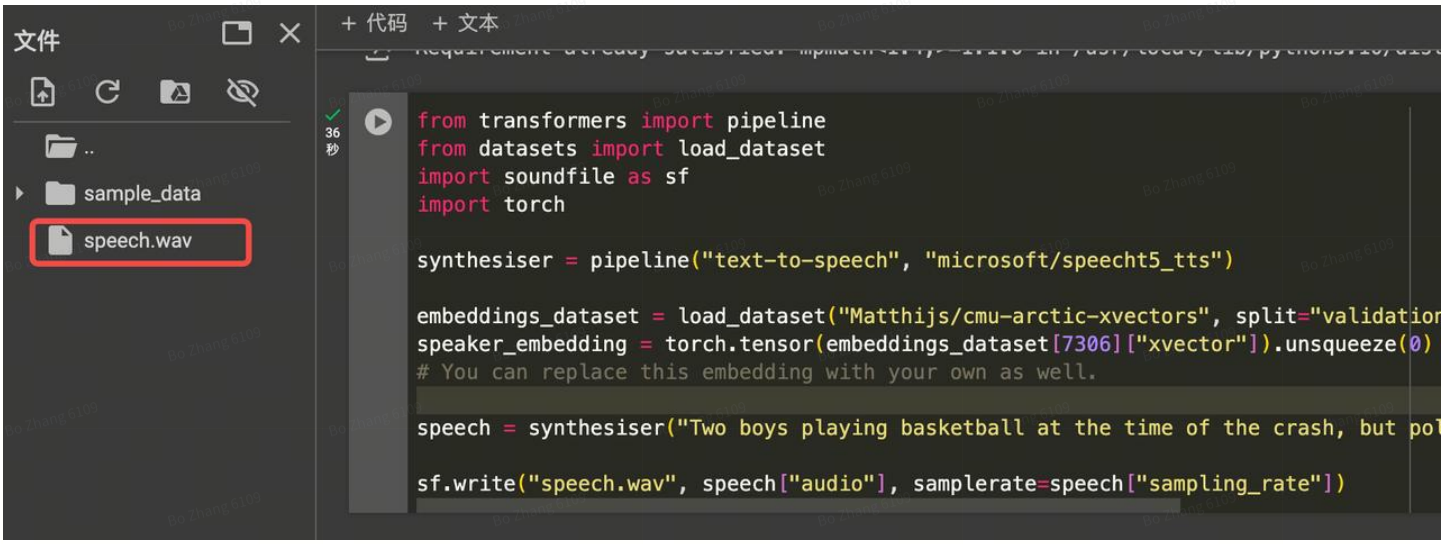
> Two boys playing basketball at the time of the crash, but police say they have received no reports of anyone being injured. One of the juveniles was hit by the other vehicle before he was hit. The boys, who were wearing helmets and white shirt

```python
from langchain_core.prompts import PromptTemplate
from langchain_openai import OpenAI
from langchain.chains import LLMChain

def generated_story(scenario):
  template = """
  你是一位专业的篮球解说员，下面CONTEXT中的内容是一个外国人说的一个英文，请求根据这句话延展处一个比赛，最好比较激烈一些

  CONTEXT: [scenario]
  STORY:
  """
  prompt = PromptTemplate(template=template, input_variables=["scenario"])
  story_llm = LLMChain(llm=OpenAI(model_name="gpt-3.5-turbo",temperature=1),prompt=prompt,verbose=True)
  story = story_llm.predict(scenario=scenario)
  return story
```

5.1.3 文字转语音

```python
from transformers import pipeline
```

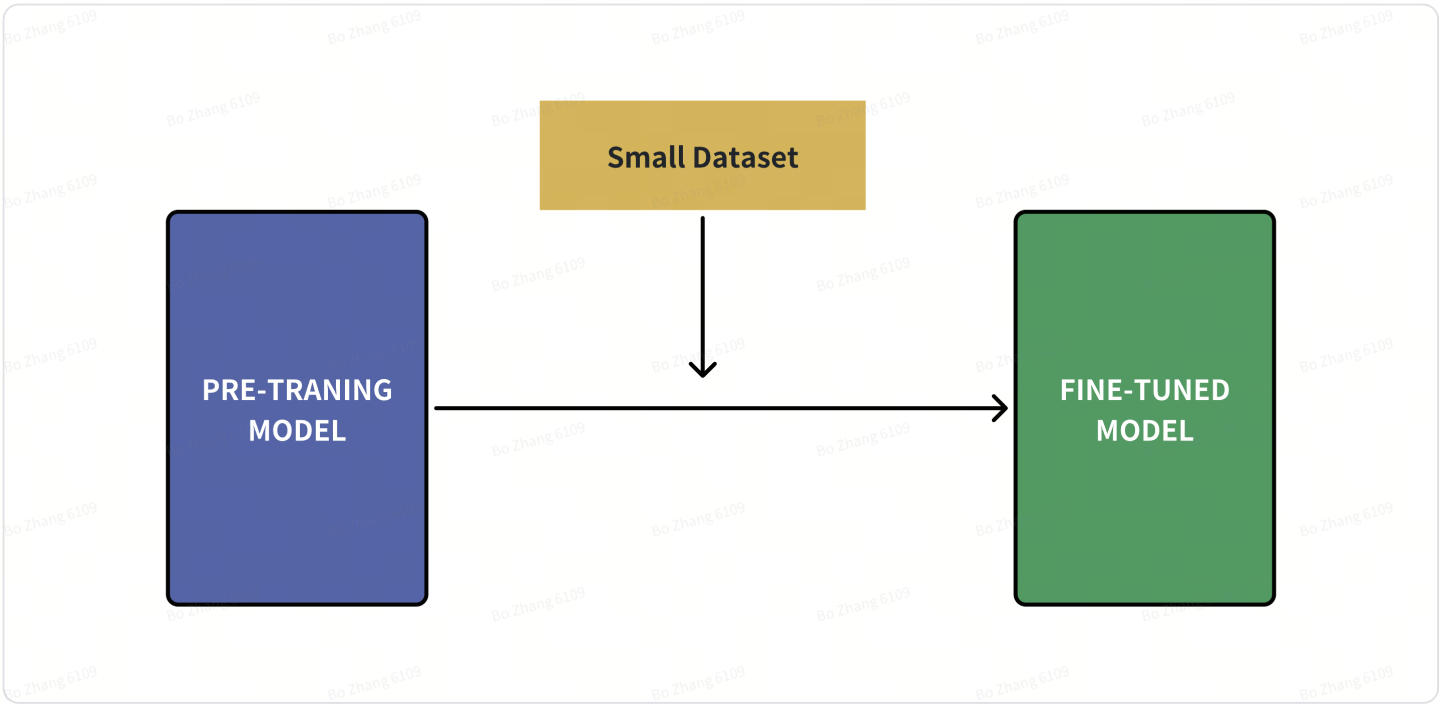
```python
from datasets import load_dataset
import soundfile as sf
import torch

synthesiser = pipeline("text-to-speech", "microsoft/speecht5_tts")

embeddings_dataset = load_dataset("Matthijs/cmu-arctic-xvectors", split="validation")
speaker_embedding = torch.tensor(embeddings_dataset[7306]["xvector"]).unsqueeze(0)
# You can replace this embedding with your own as well.

speech = synthesiser("Two boys playing basketball at the time of the crash, but police say they have received no reports of anyone being injured. One of the juveniles was hit by the other vehicle before he was hit. The boys, who were wearing helmets and white shirt", forward_params={"speaker_embeddings": speaker_embedding})

sf.write("speech.wav", speech["audio"], samplerate=speech["sampling_rate"])
```



speech

5.2 微调模型

https://kedion.medium.com/fine-tuning-nlp-models-with-hugging-face-f92d55949b66

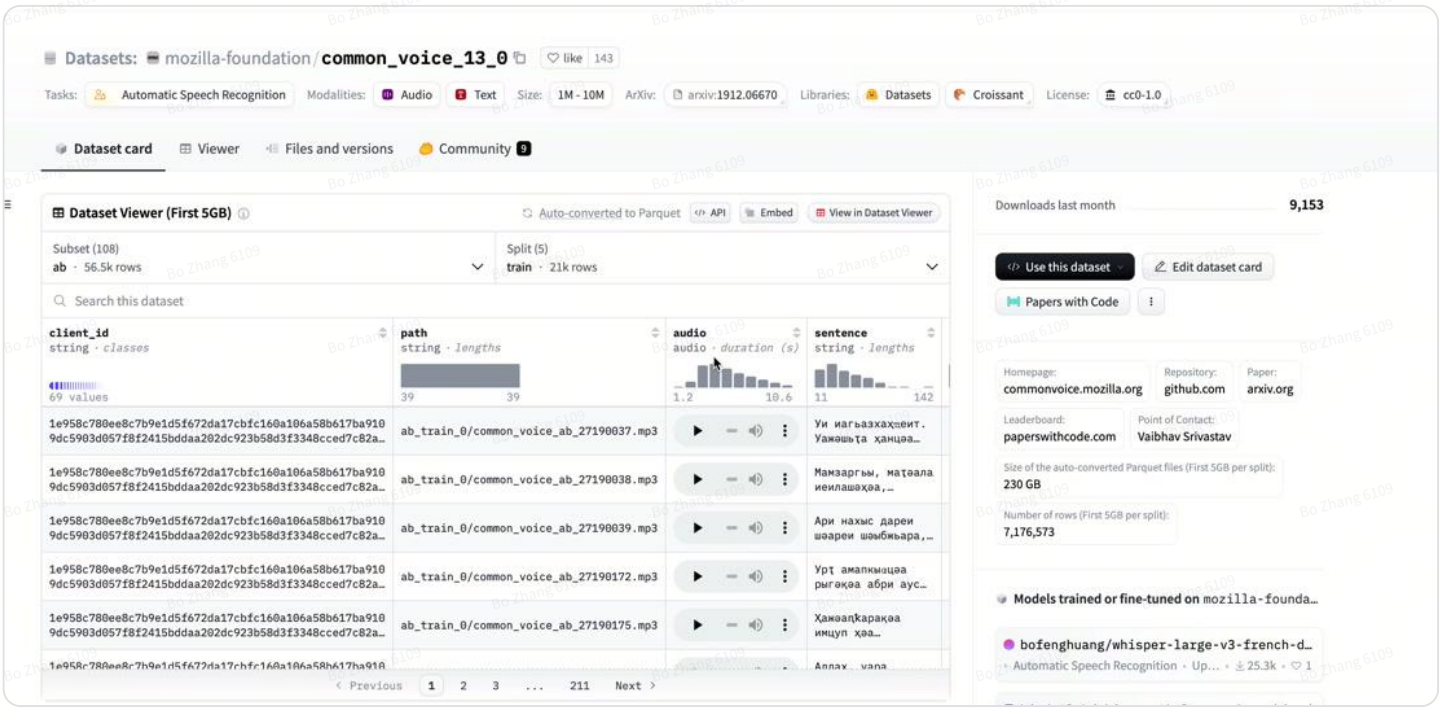


我们选择微调一个语音识别模型，选择一个语音识别模型，以及微调的数据集：

- Whisper: https://huggingface.co/openai/whisper-small
- Common-voice: https://huggingface.co/datasets/mozilla-foundation/common_voice_13_0/viewer/zh-CN

5.2.1 处理微调数据

我们选择用common_voice来进行微调，这是一个多语言数据集，如下所示：



Load the data set

查看数据集划分信息：

```
>>> from datasets import get_dataset_split_names

>>> get_dataset_split_names("mozilla-foundation/common_voice_13_0","zh-CN")
['train', 'validation', 'test', 'other', 'invalidated']
```

```
from datasets import load_dataset, DatasetDict

common_voice = DatasetDict()
common_voice["train"] = load_dataset(
    "mozilla-foundation/common_voice_13_0", "zh-CN", split="train+validation"
)
common_voice["test"] = load_dataset(
    "mozilla-foundation/common_voice_13_0", "zh-CN", split="test"
)

common_voice = common_voice.select_columns(["audio", "sentence"])

print(common_voice)
```

Pre-process the data set

```
from transformers import WhisperProcessor

processor = WhisperProcessor.from_pretrained(
    "openai/whisper-small", language="zh", task="transcribe"
)
```

```
from datasets import Audio

sampling_rate = processor.feature_extractor.sampling_rate
common_voice = common_voice.cast_column("audio",
Audio(sampling_rate=sampling_rate))
```

5.2.2 开始微调训练

从whisper-small模型，开始微调训练：

```
from transformers import WhisperForConditionalGeneration

```

```
model = WhisperForConditionalGeneration.from_pretrained("openai/whisper-small")
```

```
[ ] from transformers import WhisperForConditionalGeneration
    model = WhisperForConditionalGeneration.from_pretrained("openai/whisper-small")
```


```python
from functools import partial

# 在训练期间不使用缓存，因为它和梯度检查点不兼容
model.config.use_cache = False

# 为生成设置语言和任务，并重新启用缓存
model.generate = partial(
    model.generate, language="zh", task="transcribe", use_cache=True
)
```

设置训练参数：

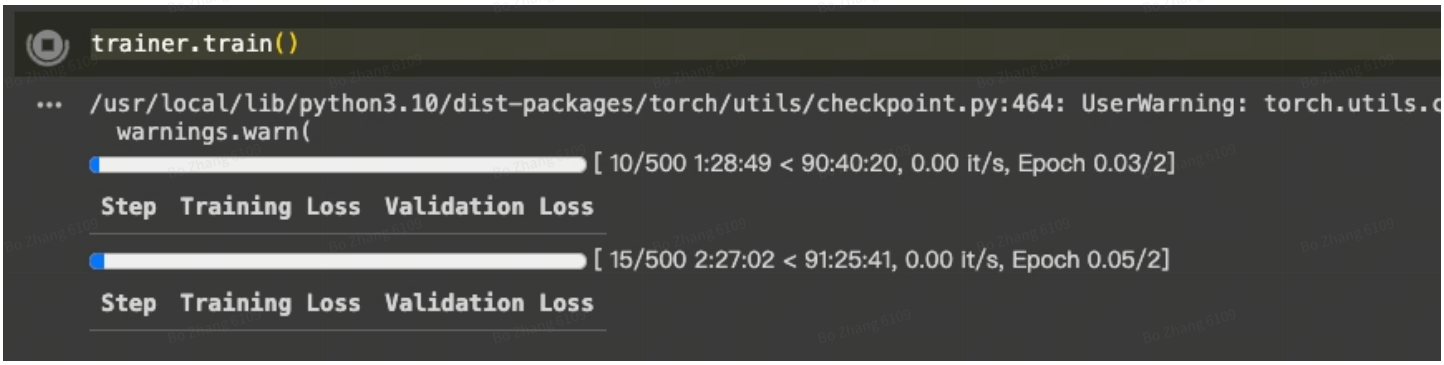
```python
from transformers import Seq2SeqTrainingArguments

training_args = Seq2SeqTrainingArguments(
    output_dir="./whisper-small-xianfeng",  # 在 HF Hub 上的输出目录的名字
    per_device_train_batch_size=16,
    gradient_accumulation_steps=1,  # 每次 batch size 下调到一半就把这个参数上调到两倍
    learning_rate=1e-5,
    lr_scheduler_type="constant_with_warmup",
    warmup_steps=50,
    max_steps=500,  # 如果您有自己的 GPU 或者 Colab 付费计划，上调到 4000
    gradient_checkpointing=True,
    fp16=True,
    fp16_full_eval=True,
    evaluation_strategy="steps",
    per_device_eval_batch_size=16,
    predict_with_generate=True,
    generation_max_length=225,
    save_steps=500,
    eval_steps=500,
```

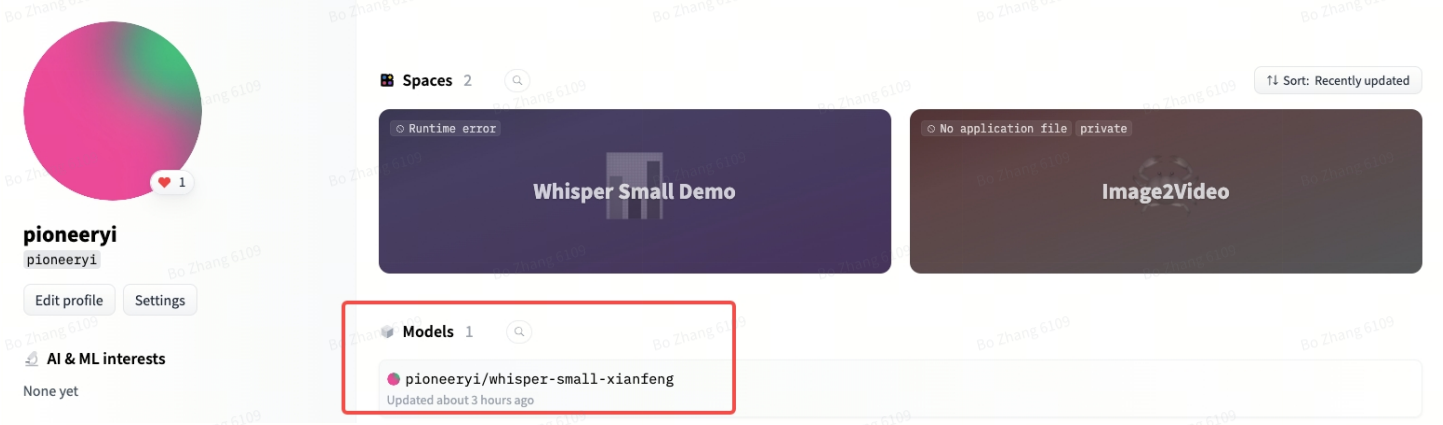
```
    logging_steps=25,
    report_to=["tensorboard"],
    load_best_model_at_end=True,
    metric_for_best_model="wer",
    greater_is_better=False,
    push_to_hub=True,
)
```

开始训练：

```
from transformers import Seq2SeqTrainer

trainer = Seq2SeqTrainer(
    args=training_args,
    model=model,
    train_dataset=common_voice["train"],
    eval_dataset=common_voice["test"],
    data_collator=data_collator,
    compute_metrics=compute_metrics,
    tokenizer=processor,
)

trainer.train()
```

训练效果如下所示：





https://colab.research.google.com/drive/1p88W6KZKa4wLTGCq91drJWXrZuFc3_c_?usp=sharing

https://neptune.ai/blog/hugging-face-pre-trained-models-find-the-best

5.3 模型部署

HuggingFace模型部署

探讨在字节云部署 Hugging Face 模型的方案

6. 学习资料

资料名	资料说明
Hugging Face Hub 和开源生态介绍	Hugging Face中文社区负责人，对于Hugging Face的介绍和分享视频
AI 快速发展年，来自 Hugging Face 的开源最新进展	HuggingFace机器学习工程师，对于HuggingFace的进展分享和介绍视频
Hugging Face NLP Course	官方课程，包含NLP基本介绍，相关数据集介绍，以及如何微调一个模型
Hugging Face Audio Course	官方课程，包含音频基础知识，音频数据处理，transformer结构，语音识别，从文本到语音
https://huggingface.co/docs	官方介绍文档

7. 课后作业

💡 通过huggingface，选一个NLP模型，然后再选一个数据集（最好是中文数据集），进行微调一下！

参考资料

https://huggingface.co/ByteDance

NLP Projects to Boost Your Resume

Natural Language Processing with Hugging Face and Transformers