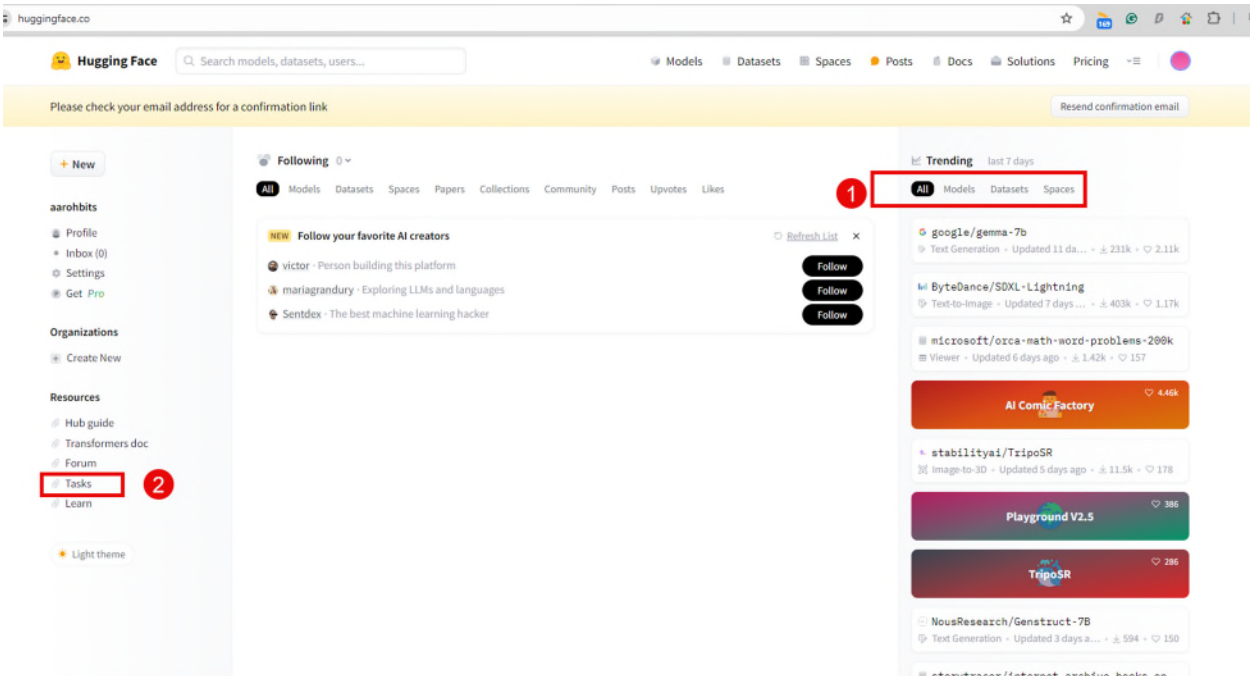
Open Source Models with Hugging Face

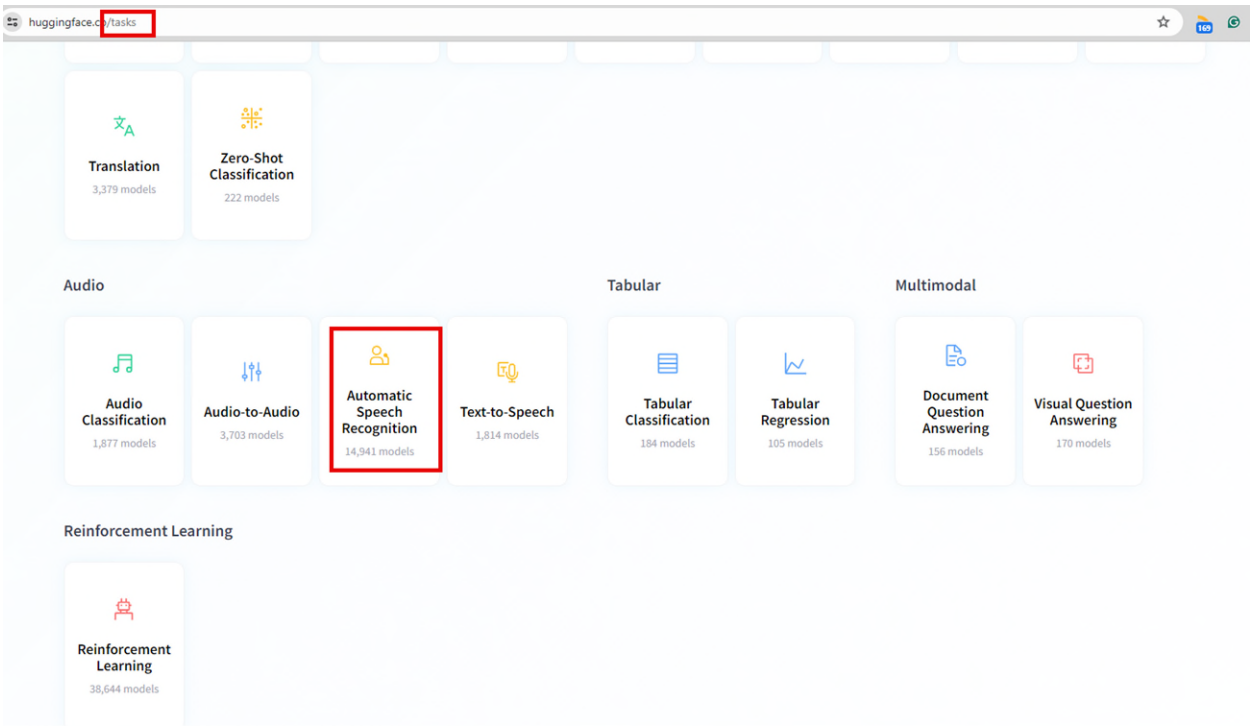
Select Models

Open Source Models

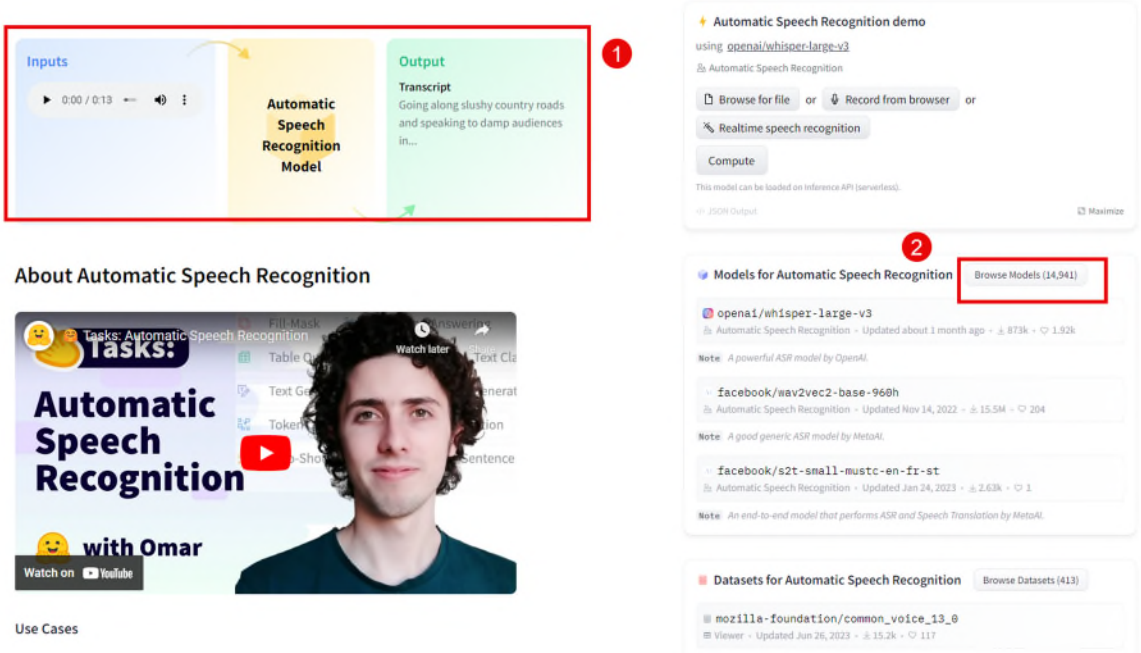
- Create your Hugging Face account.
- Select Models and select appropriate tasks, libraries, datasets



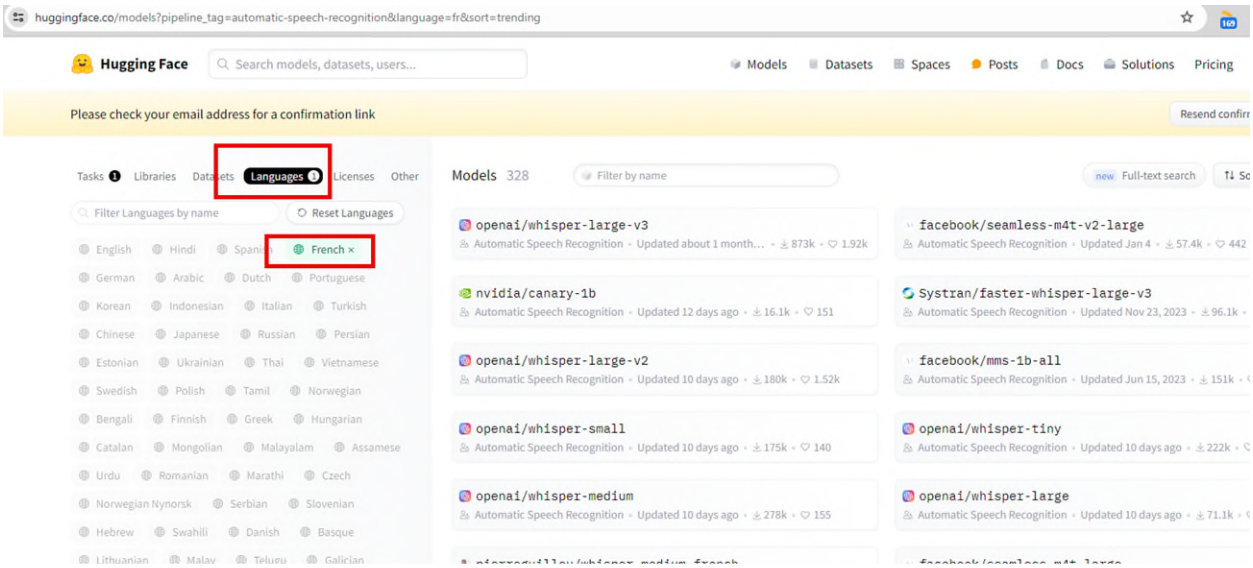
- Select **Automatic Speech Recognition**



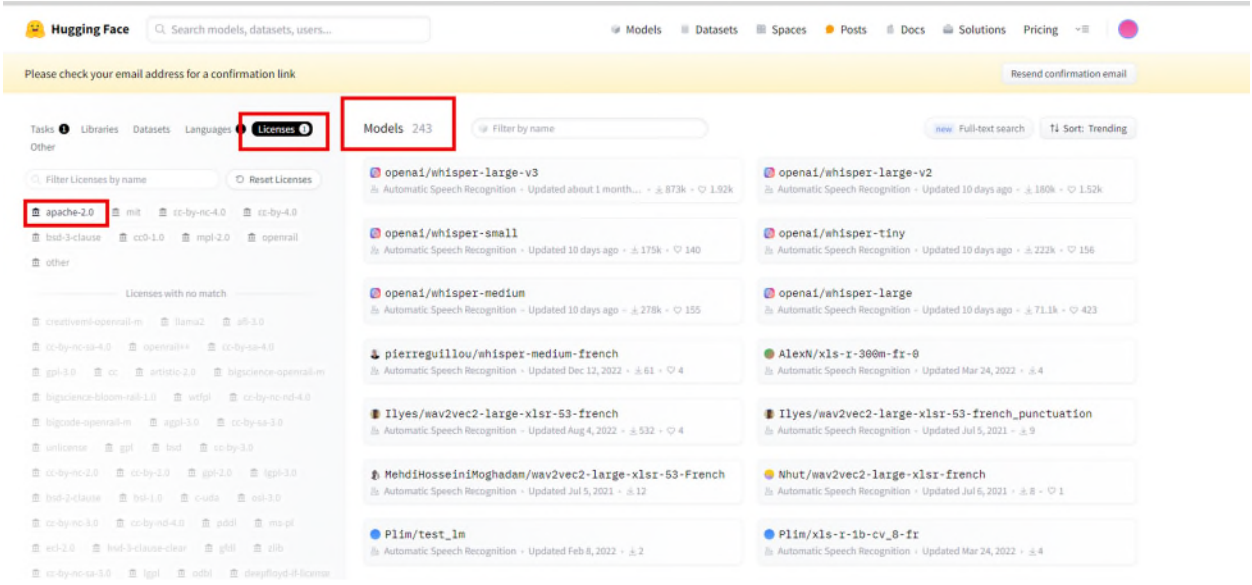
- You can view input and output.
- Select Browse Models



- Select the **Language** and we will choose French.

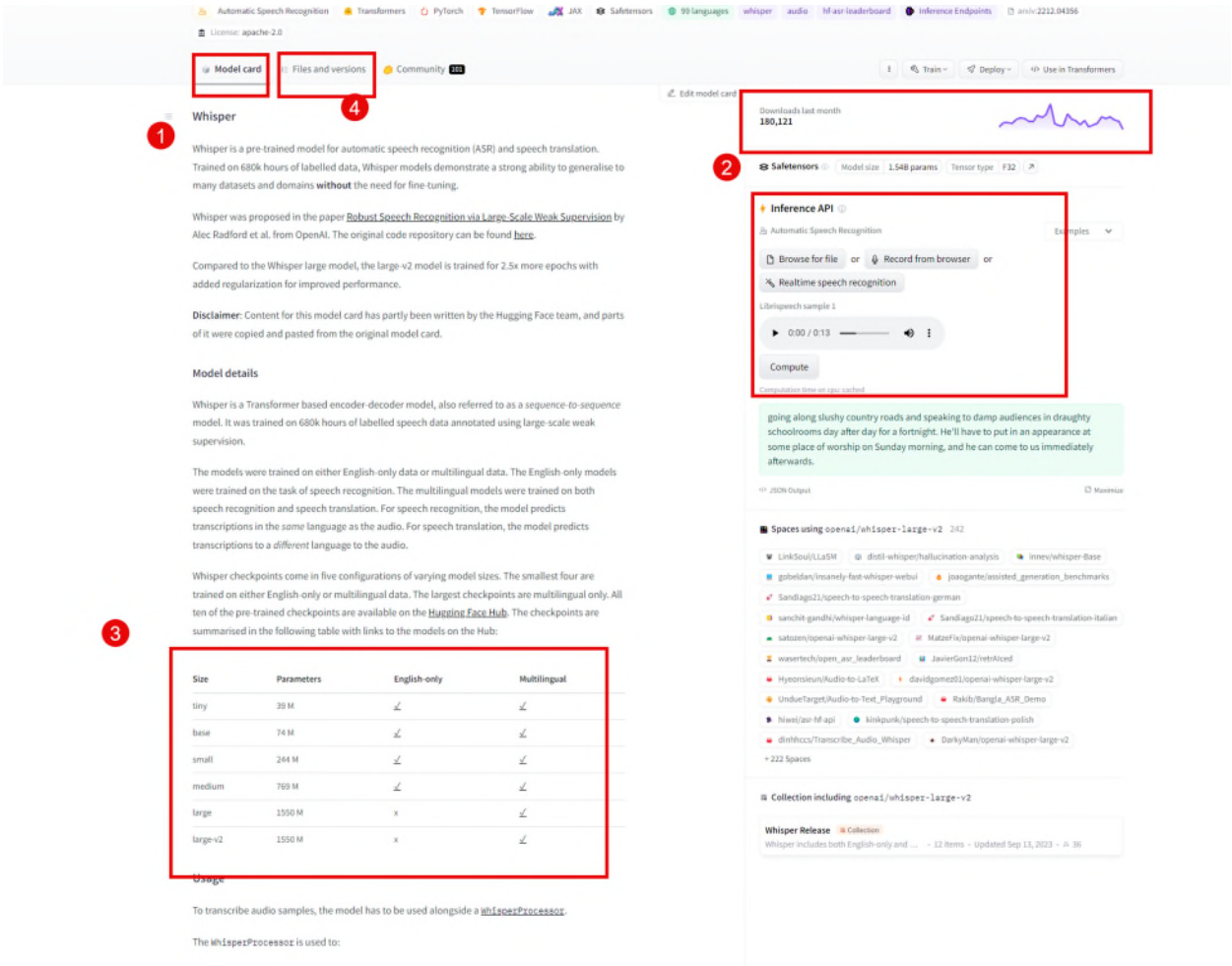


- Select **License**

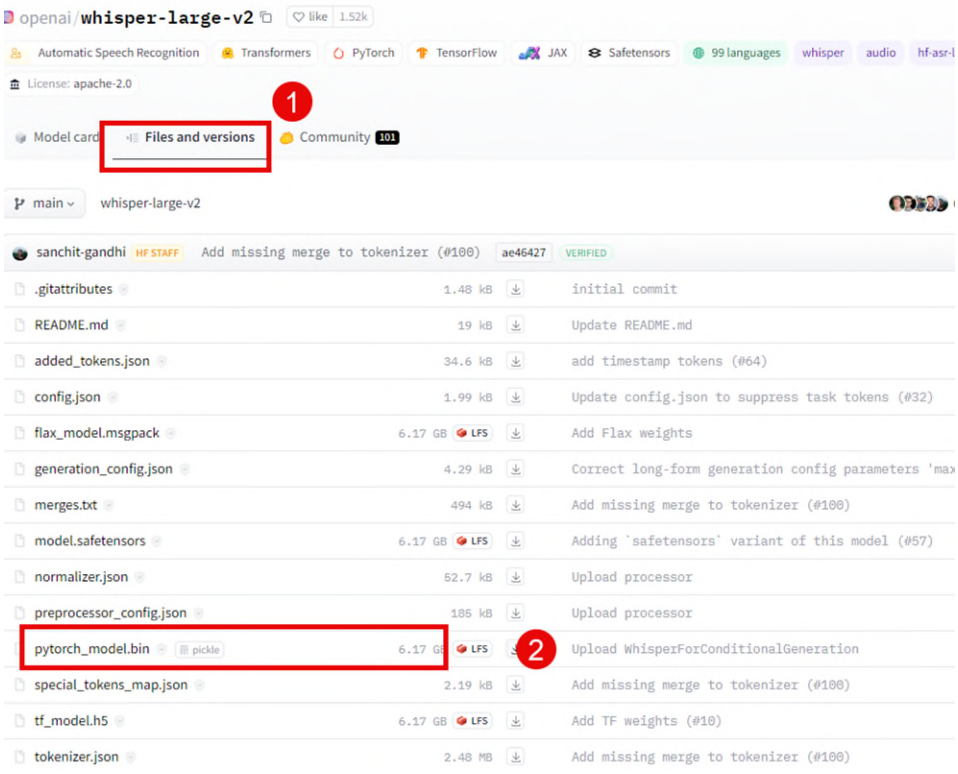


- Model Card
  - Find more details on model such as
  - Size, Parameters.
  - Usage

- Inference API



- You can get estimated model size by selecting **File and versions** and find Pytorch_model.bin size

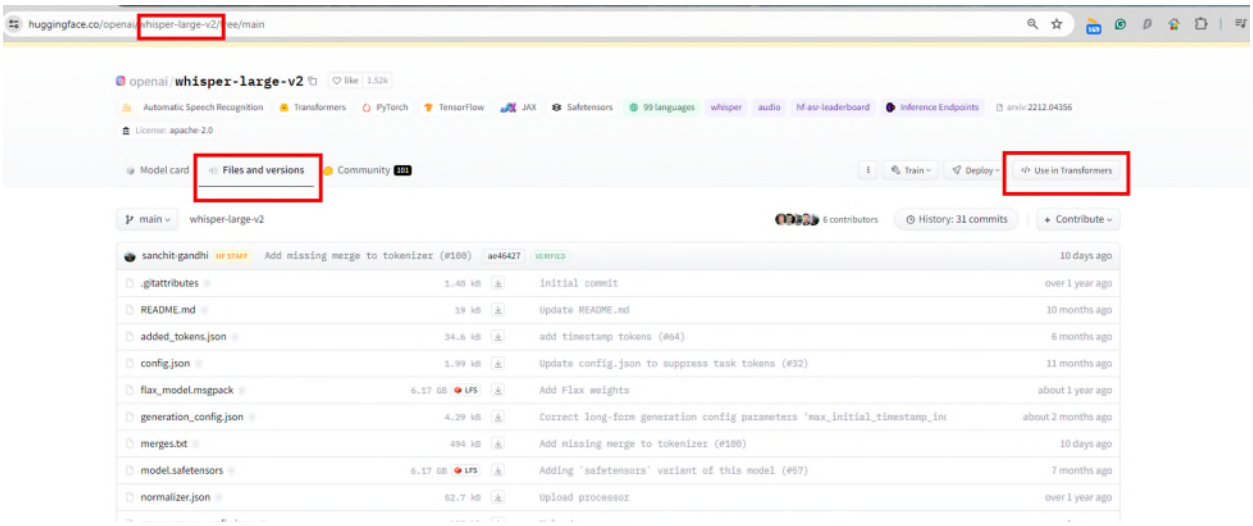


- Multiple by 1.2* times and that will give to you how much memory will be required.

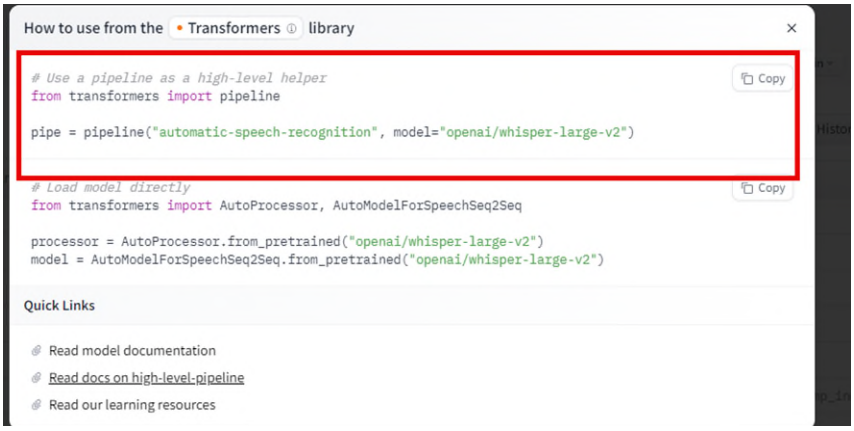
What is transformer?

So, in simple terms, a transformer in machine learning is like a super-efficient assistant that reads, understands, and processes large amounts of text data, helping to make sense of it all in a way that's faster and smarter than traditional methods.

Under File and versions, select on **Transformer**



Use Pipeline object



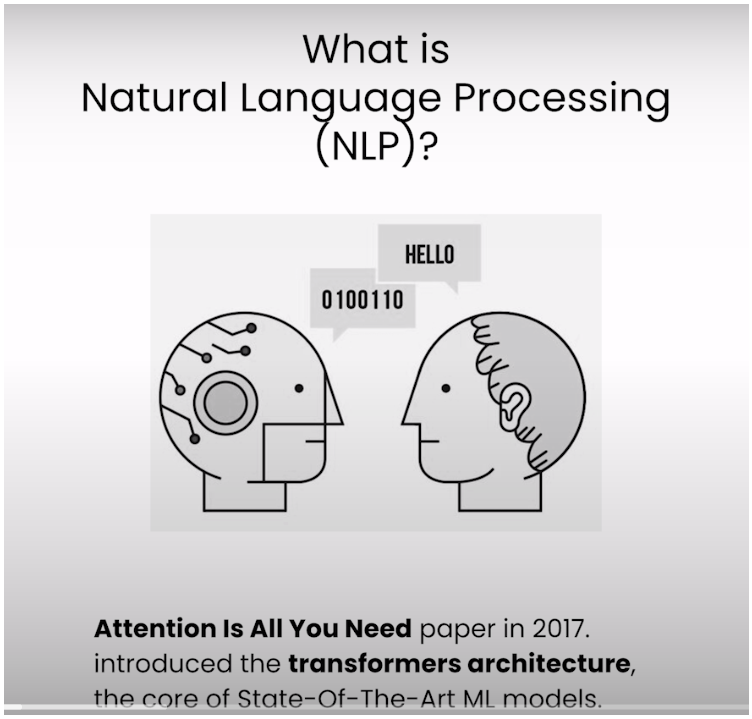
Natural Language Processing (NLP)

- Use Hugging Face library to perform NLP tasks
- Build a chatbot with Open Source model by Meta

What is NLP?

- **Attention is all you need** for a paper in 2017.

- Introduced the transformer architecture, the core of **State-of-the-Art** ML Models.



- Install transformer library

  !pip install transformers

- Build the chatbot pipeline using 🟠 Transformers Library

  from transformers.utils import logging
  logging.set_verbosity_error()

- Define the conversation pipeline

  *chatbot = pipeline(**task**="conversational", **model**="./models/facebook/blenderbot-400M-distill")*

- *user_message = """What are some fun activities I can do in the winter?"""*
- *from transformers import Conversation*
- *conversation = Conversation(user_message)*
- *print(conversation)*

- *conversation = chatbot(conversation)*

- *print(conversation)*

  You can continue the conversation with the chatbot with:
- *print(chatbot(Conversation("What else do you recommend?")))*

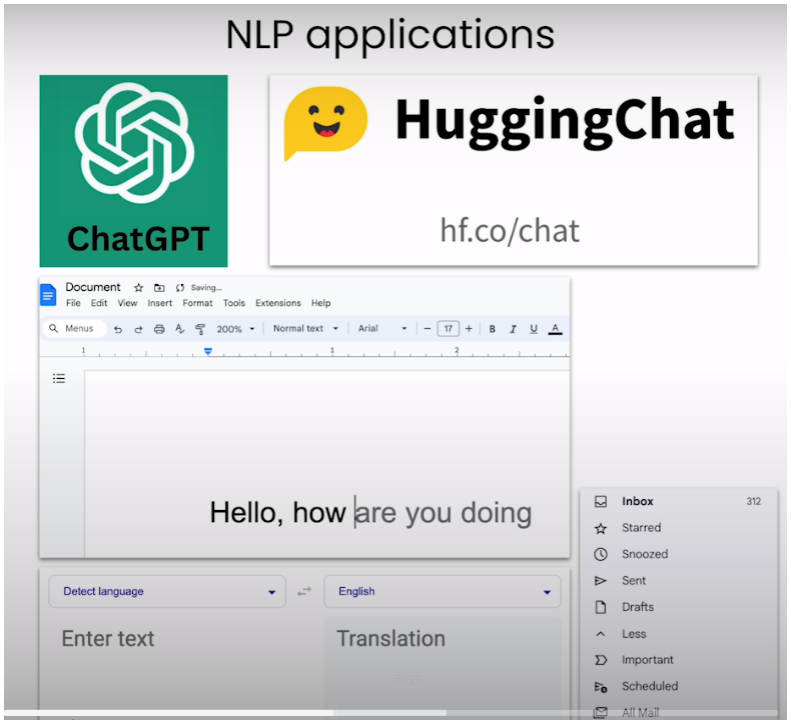
- However, the chatbot may provide an unrelated response because it does not have memory of any prior conversations.
- To include prior conversations in the LLM's context, you can add a 'message' to include the previous chat history.
-

- *conversation.add_message({"role": "user", "content": """What else do you recommend?""" })*

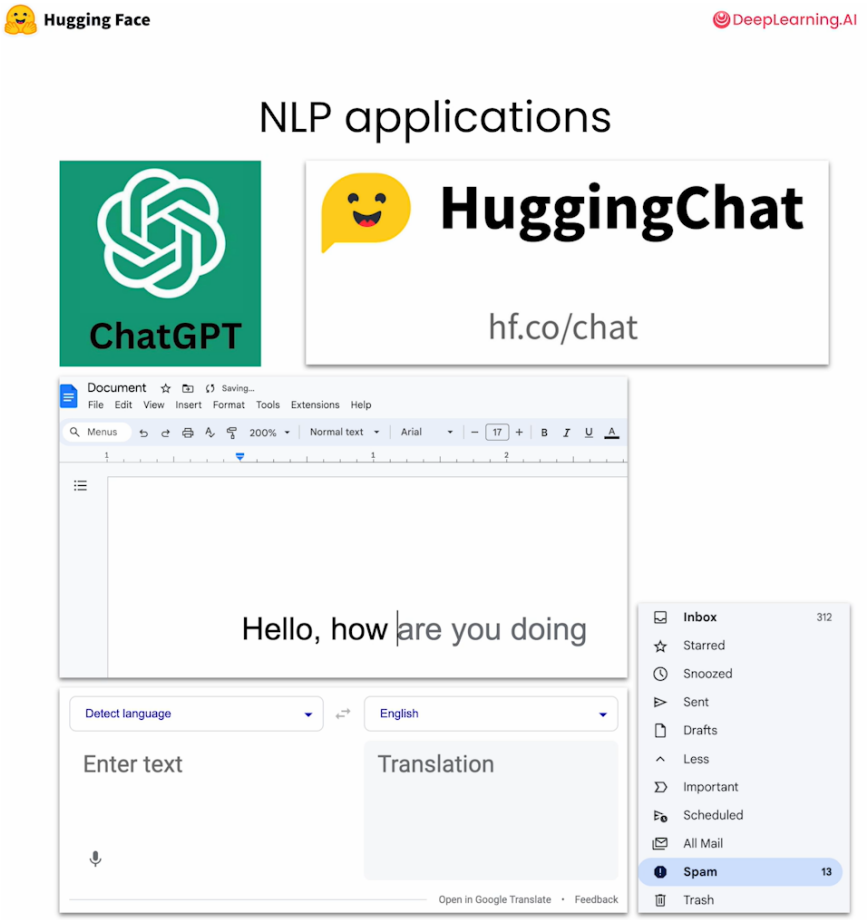
- *print(conversation)*

- *conversation = chatbot(conversation)*

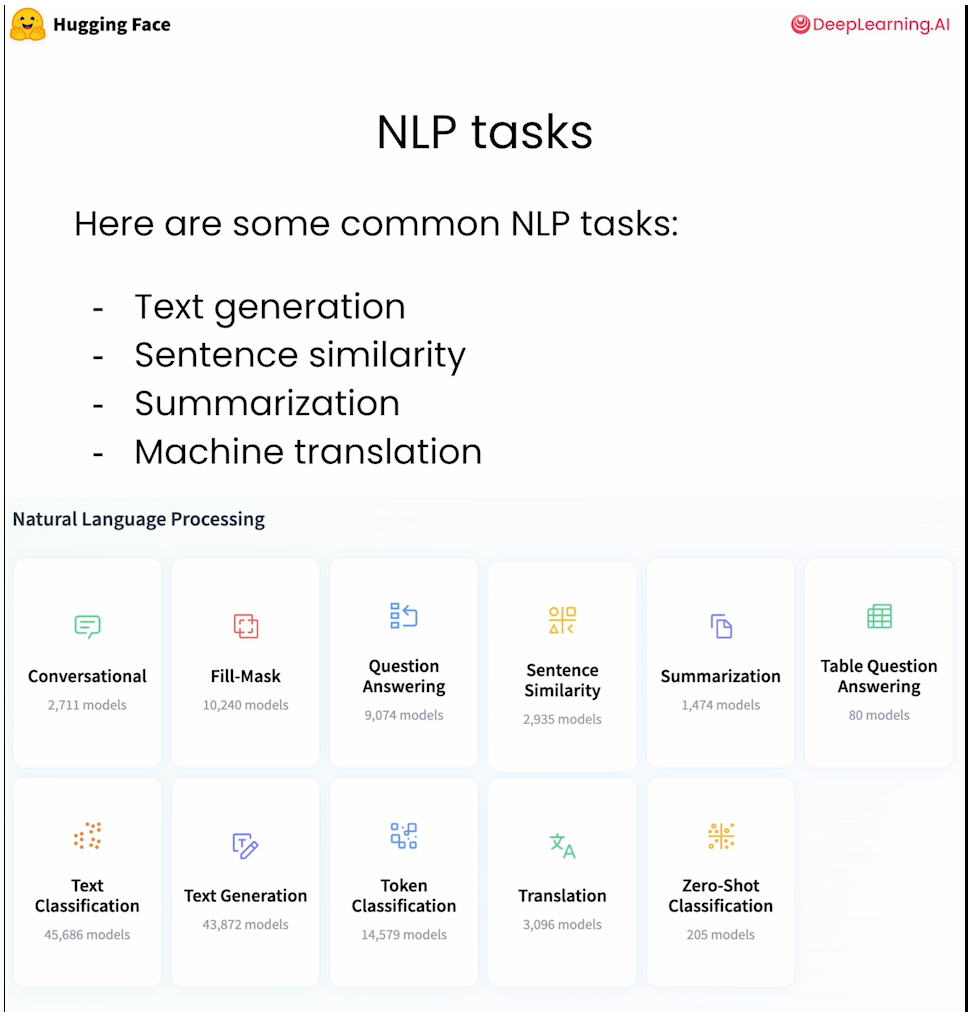
- *print(conversation)*



NLP Applications

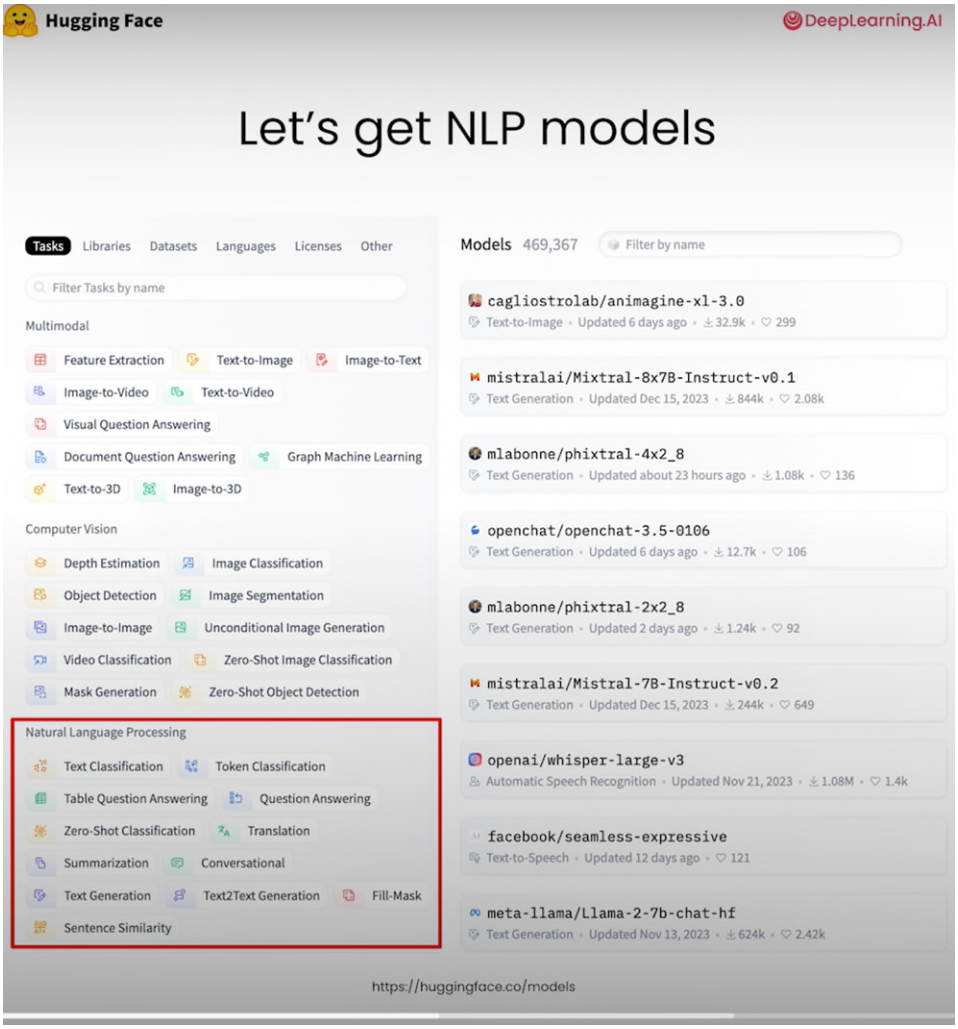


NLP Tasks

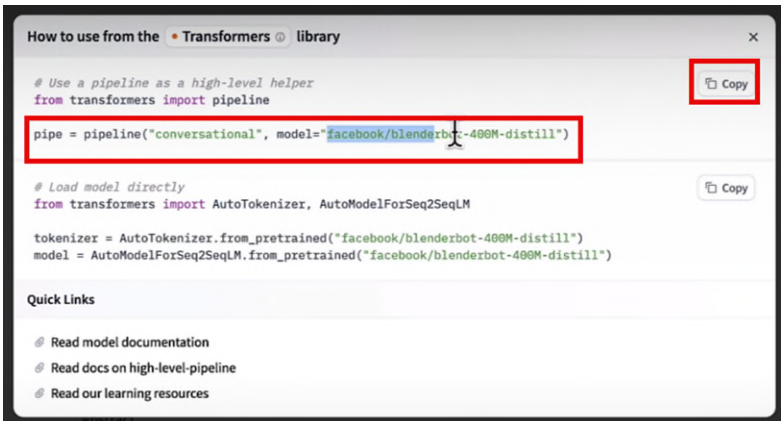


Which one to choose from?

As there are so many open source models.



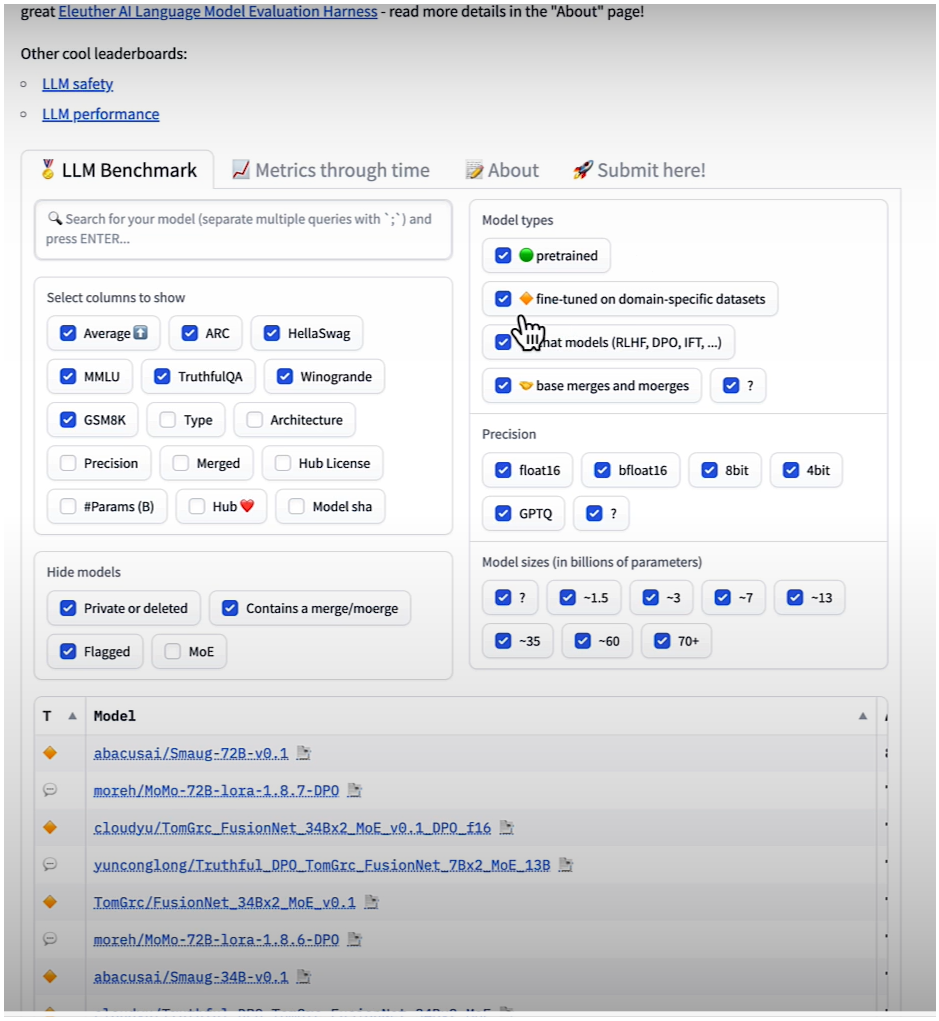
*Select **Filters** and use Transformers*



LLM Leaderboard

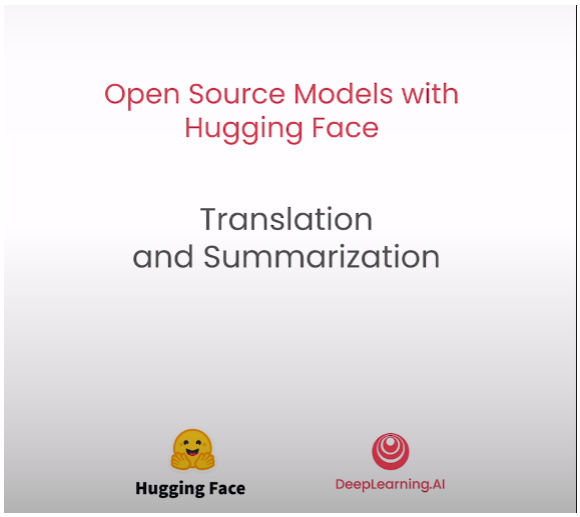
‘

- Pretrained are modelled from scratch.



Translation and Summarization

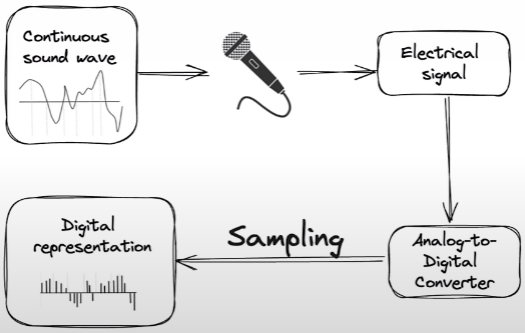
- English to Frech – (Translation)



Zero-Shot Audio Classification

Audio Data

Audio data



Sampling rate

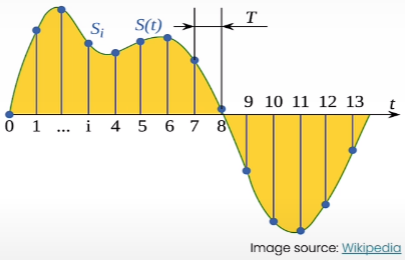


Image source: Wikipedia

Sampling = measuring the value of a continuous signal at fixed time steps.

Sampling rate (Hz) = the number of samples taken in one second.

Sampling rate examples

8,000 Hz: telephone/walkie-talkie

16,000 Hz: human speech recording

192,000 Hz: high-resolution audio

Sampling rate for transformer models

5-second sound

At 8,000 Hz => 5*8000 = 40,000 signal values

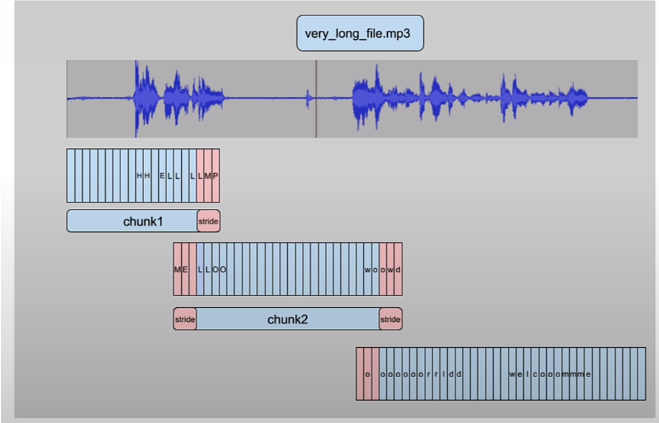
At 16,000 Hz => 5*16000 = 80,000 signal values

At 192,000 Hz => 5*192000 = 960,000 signal values

For a transformer trained on 16 kHz audio, an array of 960,000 values will look like a 60-second recording at 16kHz (60*16,000 = 960,000).

#Lesson 6: LAB 1: Automatic Speech Recognition

Transcribing long recordings

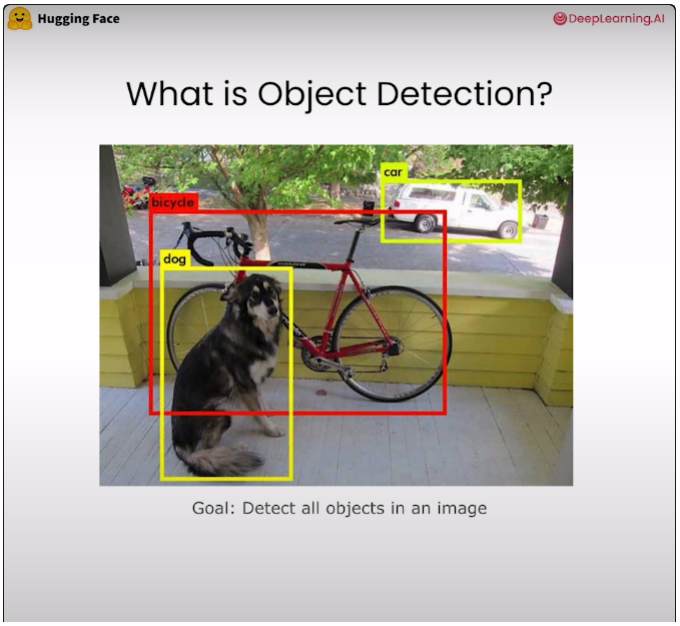


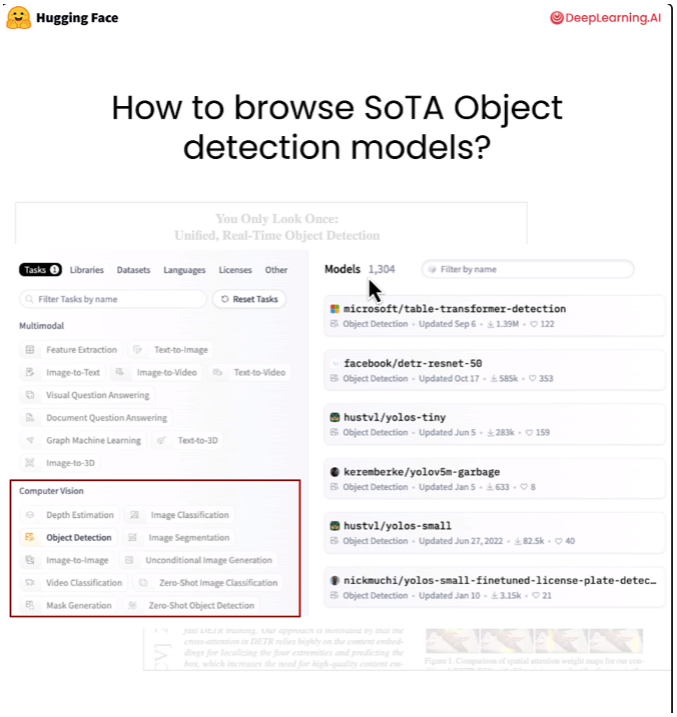
Lesson 7: Text to Speech

Lesson 8: LAB 2: Object Detection (visually impaired person)

- Helping visually impaired person what is in the picture.
- Info about facebook/detr-resnet-50
- Explore more of the Hugging Face Hub for more object detection models

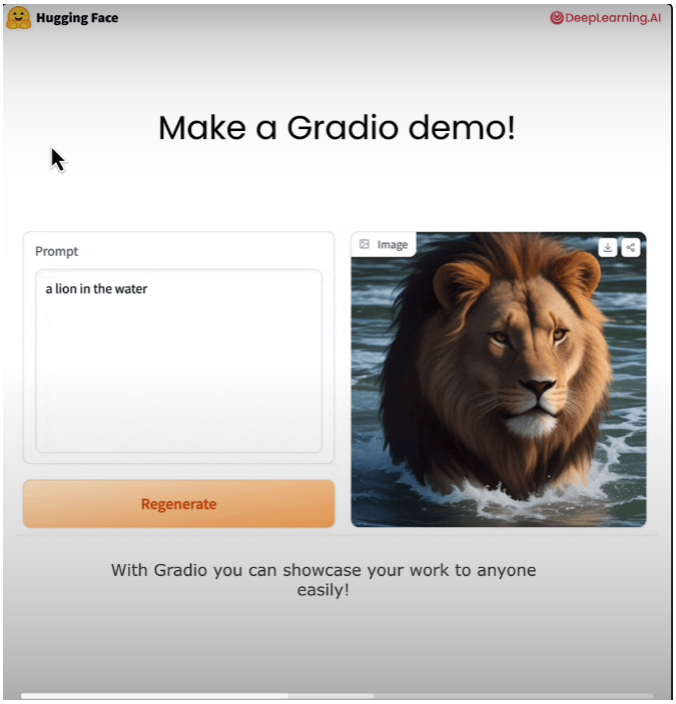
What is object detection?



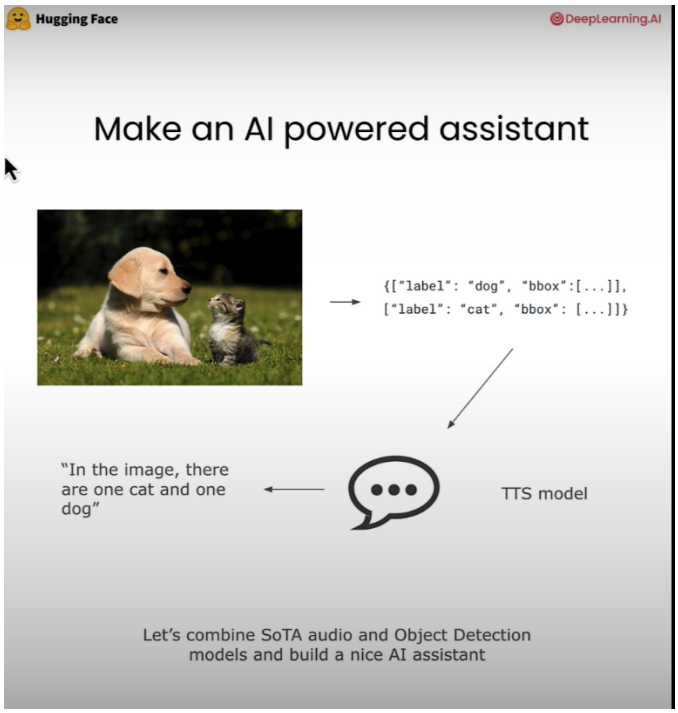


Sample Image

Final demo



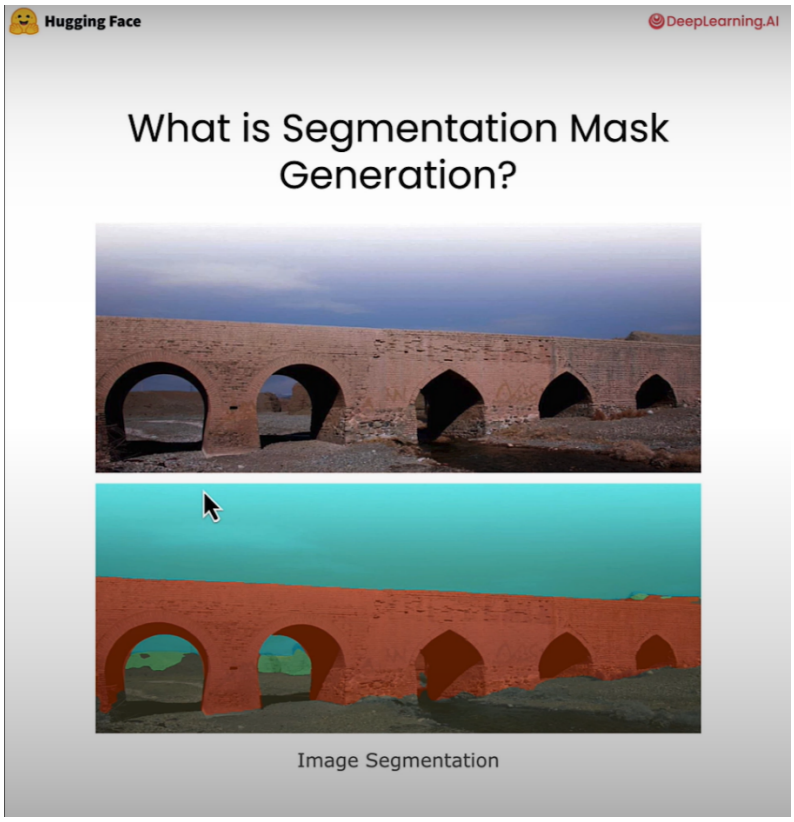
LAB 2: Make an AI Powered assistant



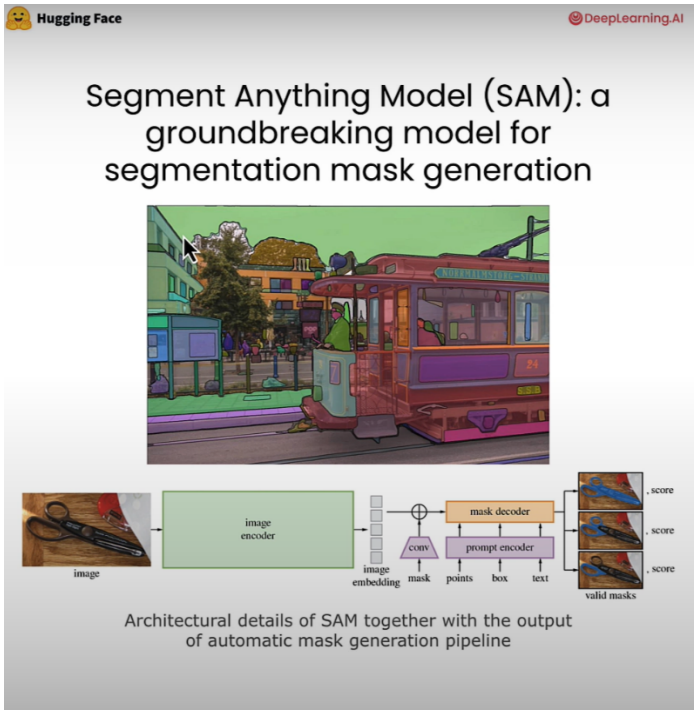
Lesson 9: Segmentation

- Visual prompting
- Meta AI

What is Segmentation Mask Generation?



Segment Any model (SAM)



Hugging Face DeepLearning.AI

Segment Anything Model (SAM): a groundbreaking model for segmentation mask generation

0.1% Data Makes Segment Anything Slim

Zigeng Chen , Gongfan Fang , Xinyin Ma , Xinchao Wang*
National University of Singapore
zigeng99@u.nus.edu, gongfan@u.nus.edu, xinyinma@u.nus.edu, xinchao@nus.edu.sg

Abstract

The formidable model size and demanding computational requirements of Segment Anything Model (SAM) have rendered it cumbersome for deployment on resource-constrained devices. Existing approaches for SAM compression typically involve training a new network from scratch, posing a challenging trade-off between compression costs and model performance. To address this issue, this paper introduces SlimSAM, a novel SAM compression method that achieves superior performance with remarkably low training costs. This is achieved by the efficient reuse of pre-trained SAMs through a unified pruning-distillation framework. To enhance knowledge inheritance from the original SAM, we employ an innovative alternate slimming strategy that partitions the compression process into a progressive procedure. Diverging from prior pruning techniques, we meticulously prune and distill decoupled model structures in an alternating fashion. Further-

For the lesson, we will use SlimSAM: a compressed version of SAM

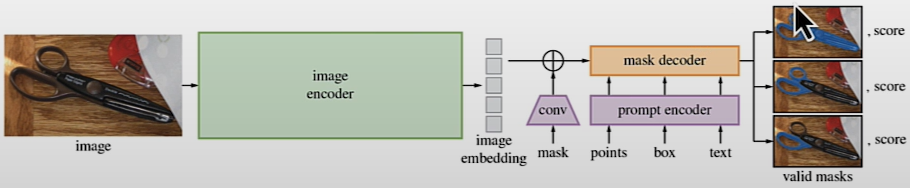
```
from PIL import Image
raw_image = Image.open('meta_llamas.jpg')
raw_image
```

```
output = sam_pipe(raw_image, points_per_batch=32)
```

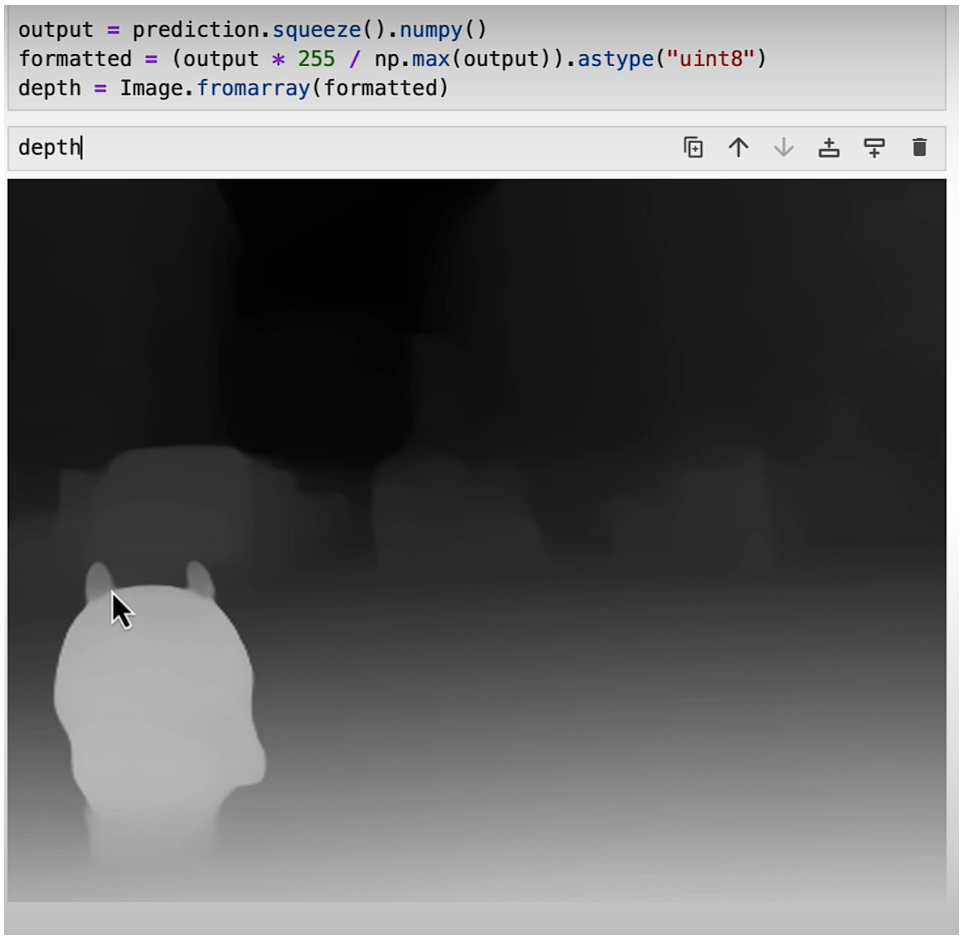
```
from helper import show_pipe_masks_on_image

show_pipe_masks_on_image(raw_image, output)
```

Hugging Face DeepLearning.AI

Segment Anything Model (SAM): a groundbreaking model for segmentation mask generation



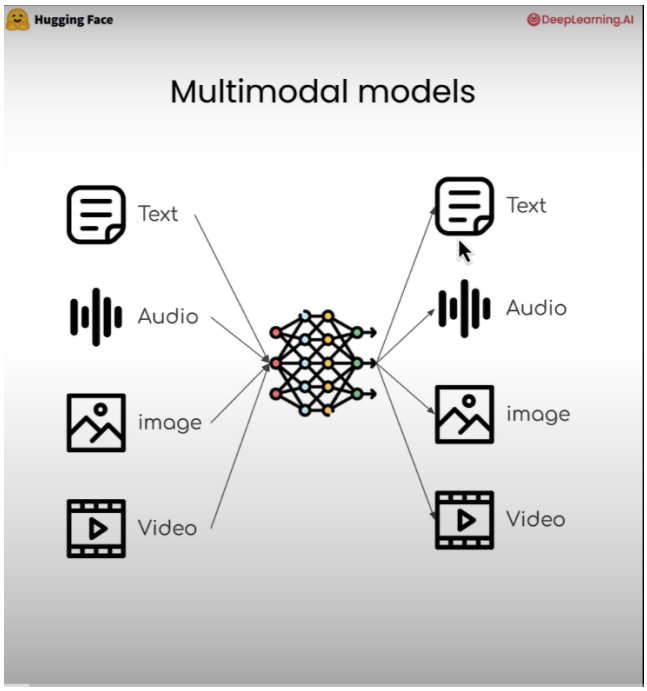
Architectural details of SAM together with the output of automatic mask generation pipeline

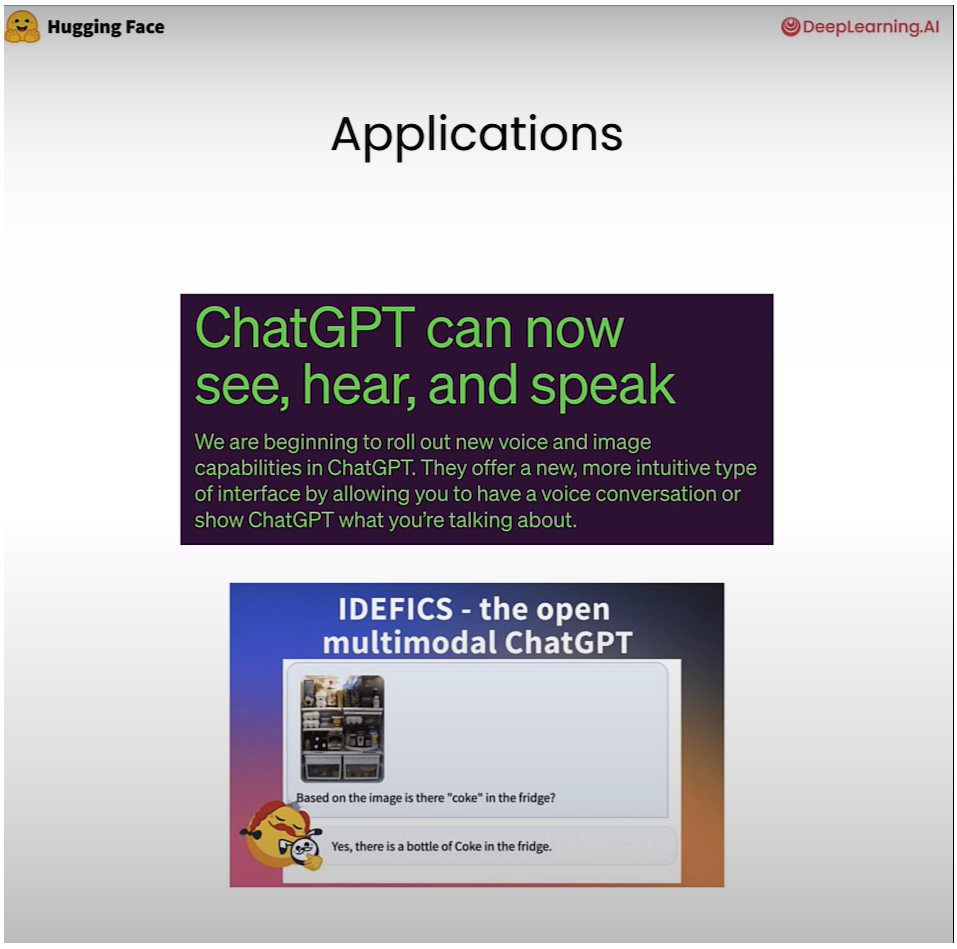


Lesson 10: Image Retrieval

- An Image Retrieval from Salesforce.

- Multimodal models

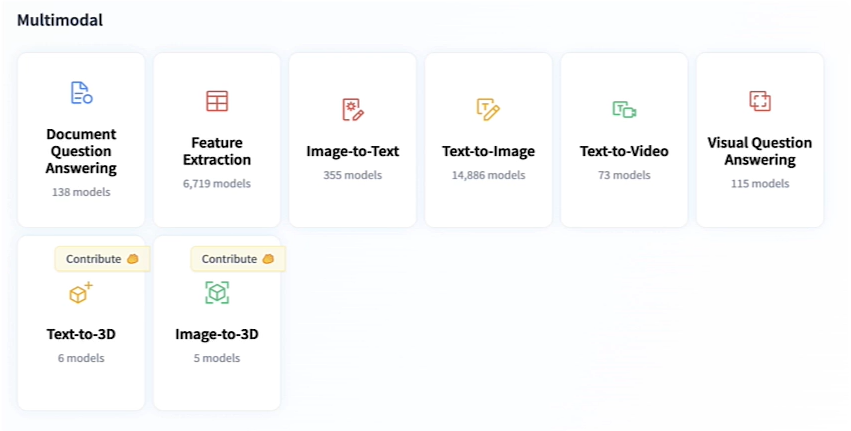




Multimodal Tasks

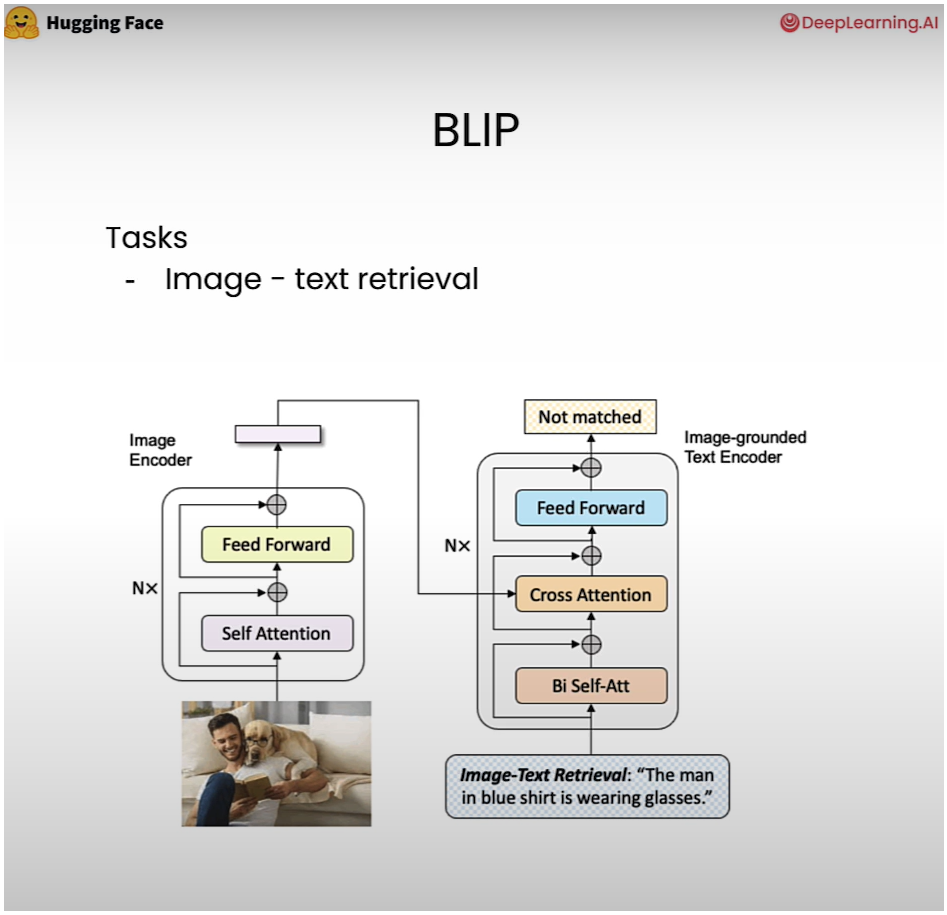
Here are some common multimodal tasks that we will go through:

- Image to text matching
- Image captioning
- Visual Q&A
- Zero-shot image classification

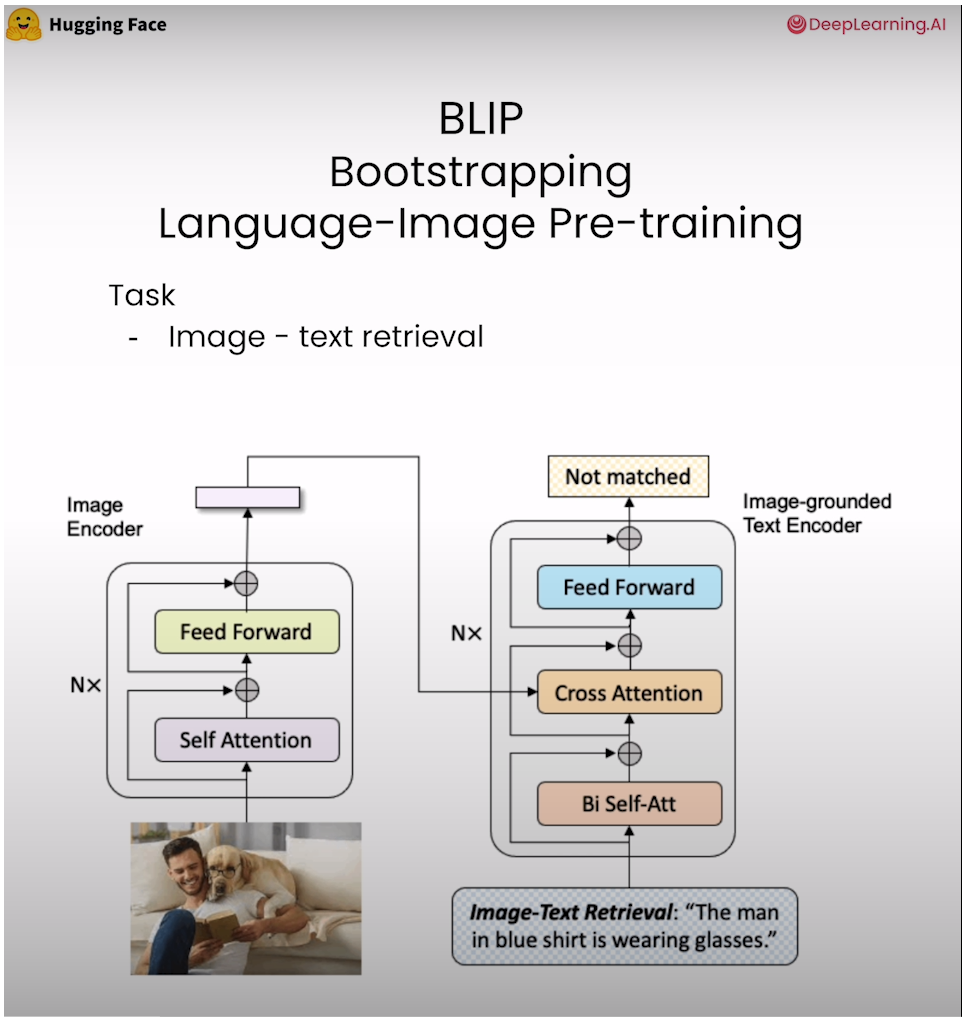


Multimodal Tasks from Hugging Face Hub

BLIP



Lesson 11: Image Captioning

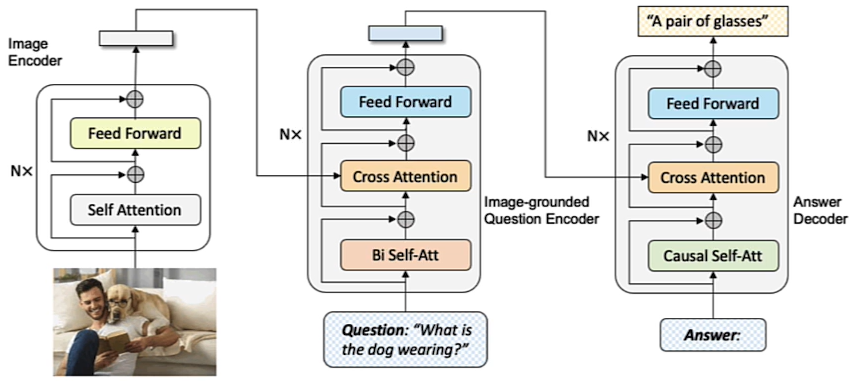


Lesson 12: Visual Question & Answering

Hugging Face DeepLearning.AI

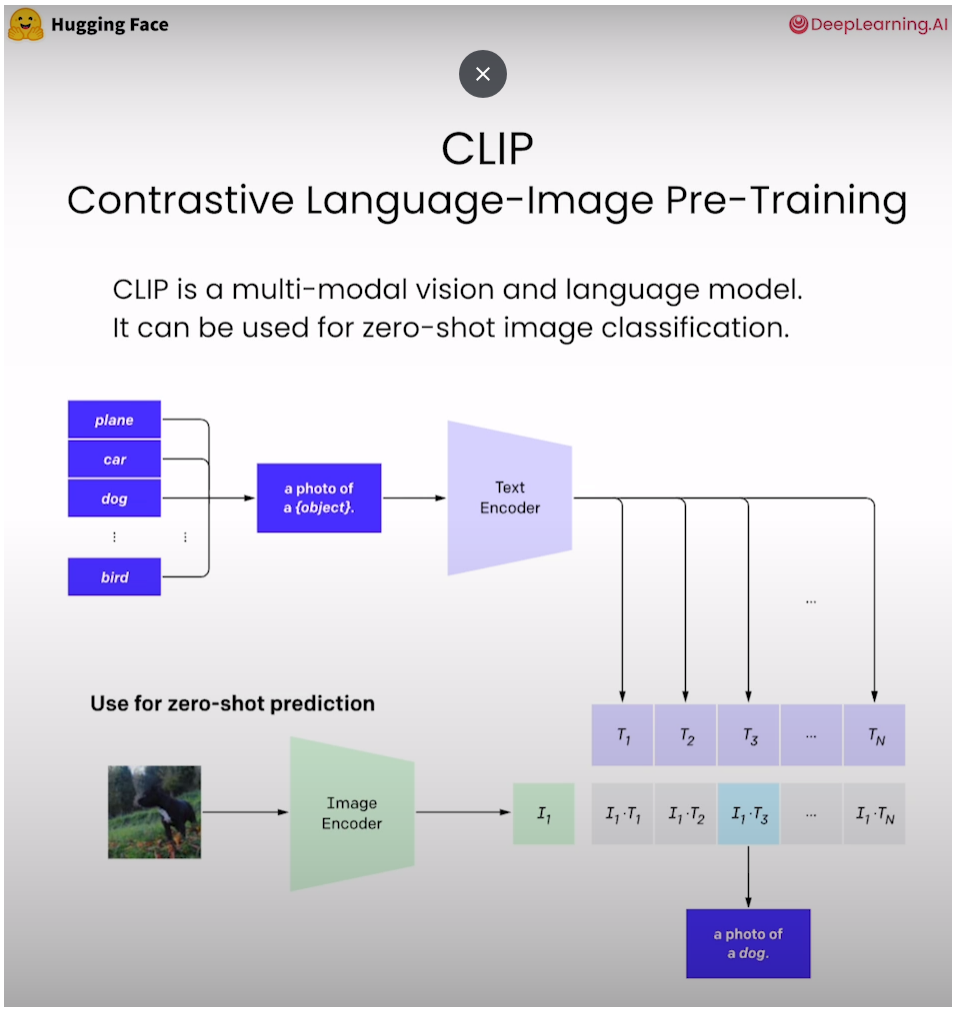
BLIP

Tasks
- Visual Q&A



Lesson 13: Zero-Shot Image Classification

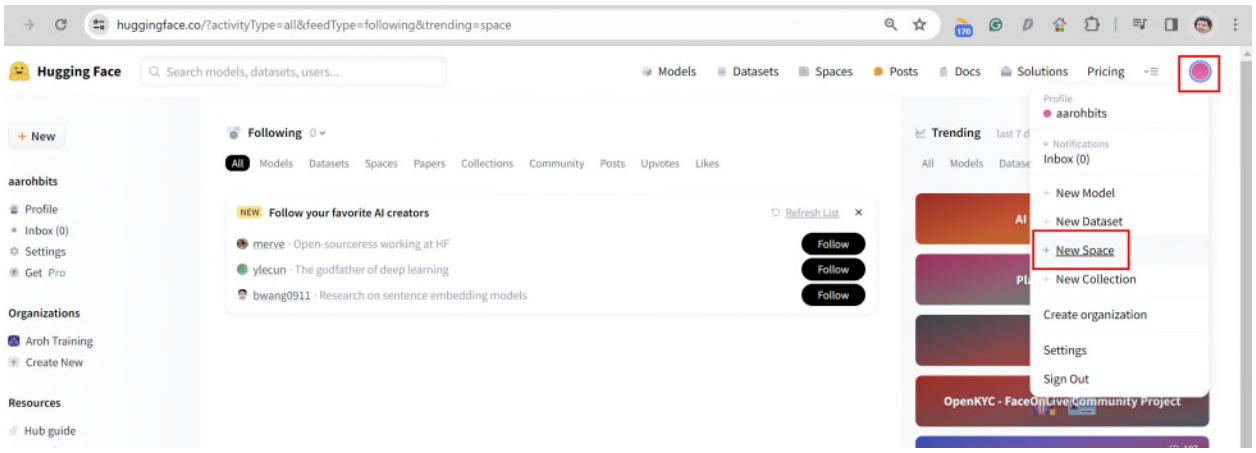
- OpenAI

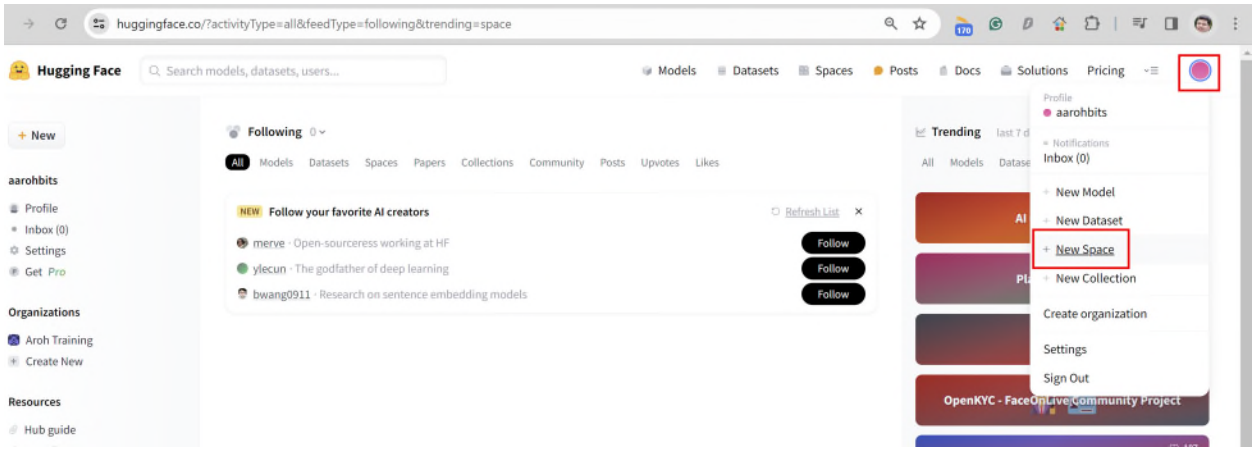
- 

Lesson 14: Deploy ML Models on 🤗 Hub using Gradio

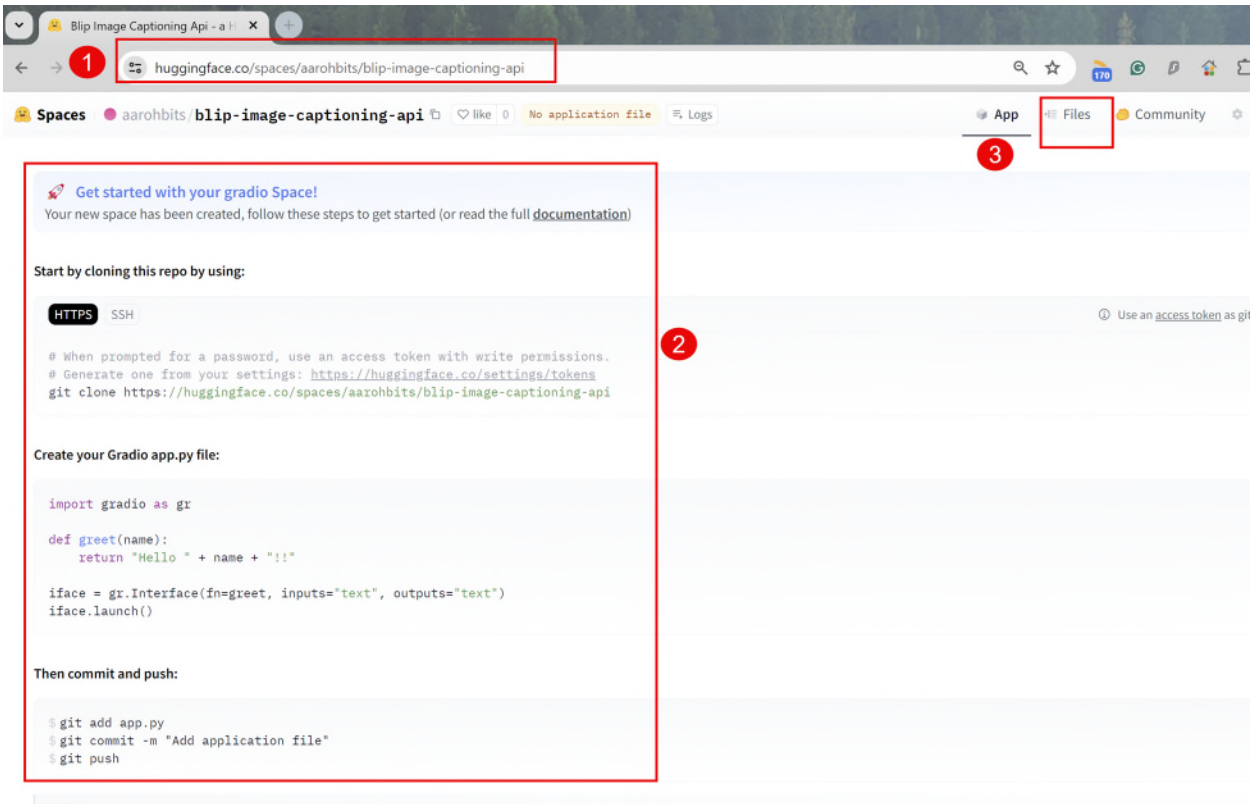
- Salesforce

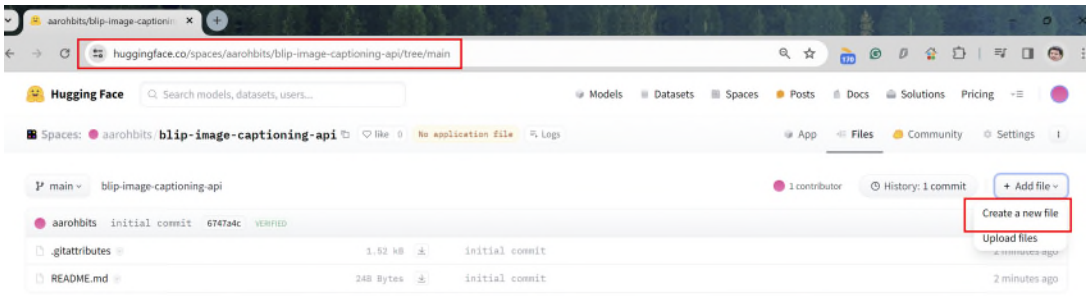
Create a new space



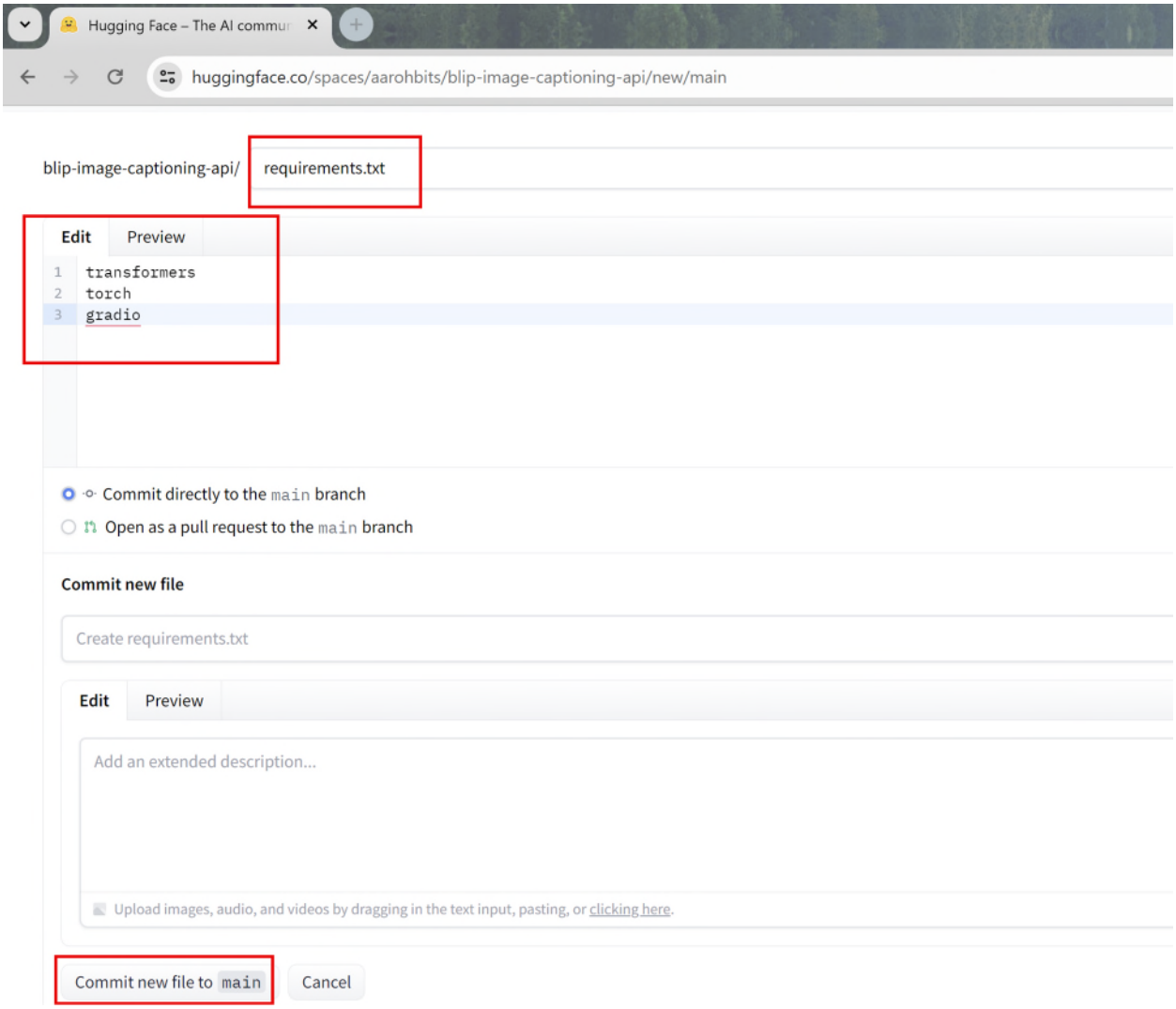


Need to create 2 files for this project


Blip Image Captioning Api - a
huggingface.co/spaces/aarohbits/blip-image-captioning-api
Spaces
aarohbits / blip-image-captioning-api
like 0
No application file
Logs
App
Files
Community
1
2
3
Get started with your gradio Space!
Your new space has been created, follow these steps to get started (or read the full documentation)
Start by cloning this repo by using:
HTTPS
SSH
Use an access token as git
When prompted for a password, use an access token with write permissions.
Generate one from your settings: https://huggingface.co/settings/tokens
git clone https://huggingface.co/spaces/aarohbits/blip-image-captioning-api
Create your Gradio app.py file:
import gradio as gr
def greet(name):
return "Hello " + name + "!!"
iface = gr.Interface(fn=greet, inputs="text", outputs="text")
iface.launch()
Then commit and push:
$ git add app.py
$ git commit -m "Add application file"
$ git push

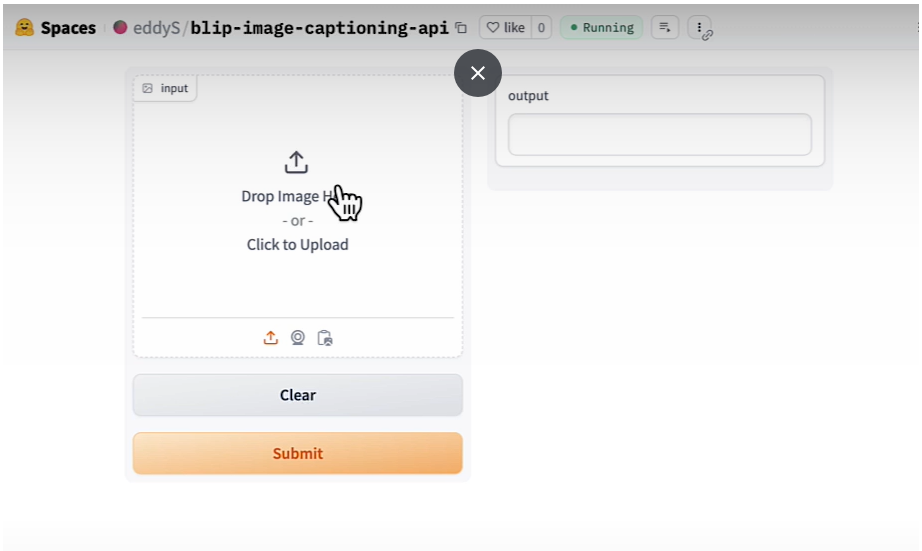

huggingface.co/spaces/aarohbits/blip-image-captioning-api/tree/main
Hugging Face
Search models, datasets, users...
Models
Datasets
Spaces
Posts
Docs
Solutions
Pricing
Spaces: aarohbits / blip-image-captioning-api
like 0
No application file
Logs
App
Files
Community
Settings
main
blip-image-captioning-api
1 contributor
History: 1 commit
+ Add file
Create a new file
Upload files
aarohbits
initial commit
6747a4c
VERIFIED
.gitattributes
1.52 kB
initial commit
README.md
248 Bytes
initial commit
2 minutes ago

1. Requirements.txt

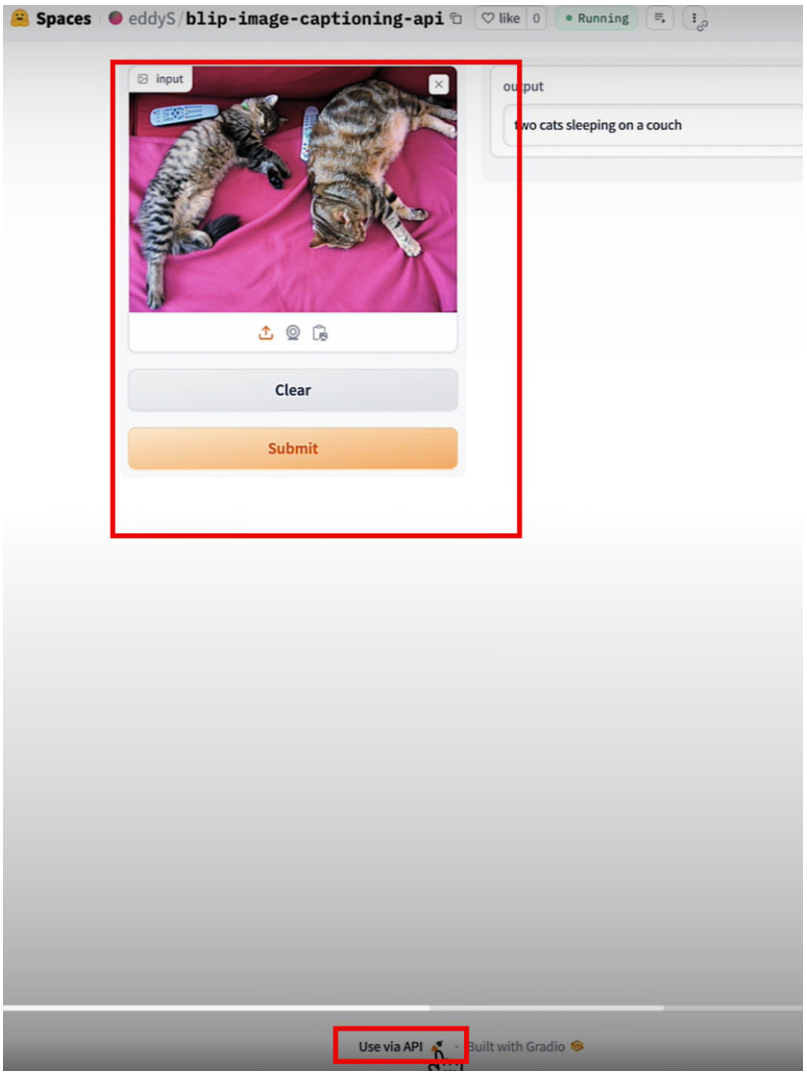


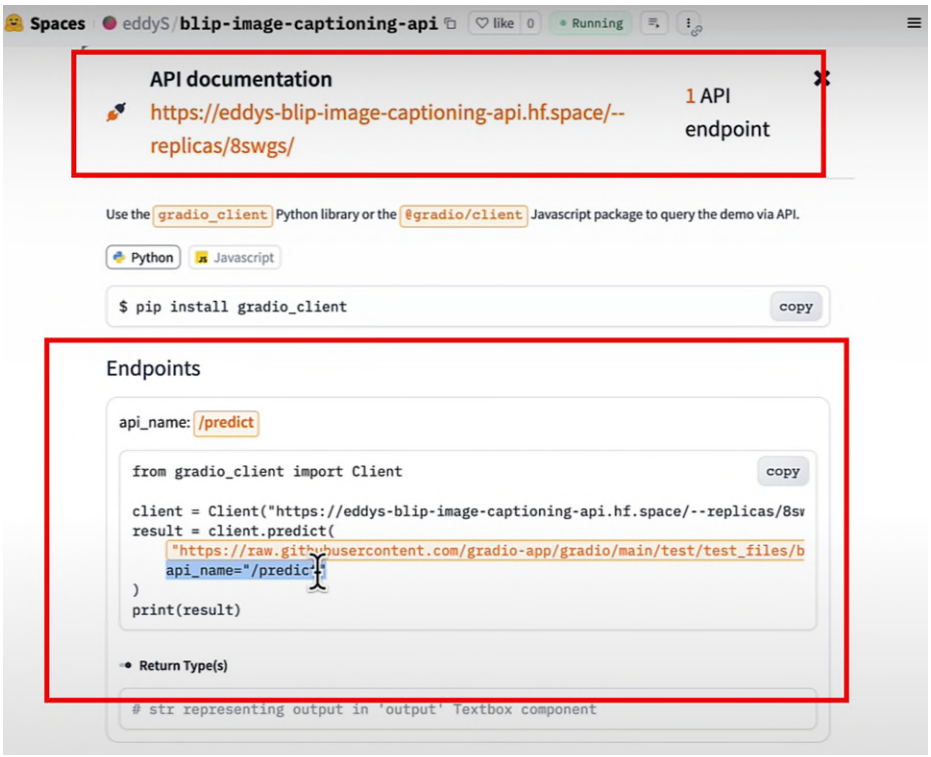
2. App.py

Copy and paste the python code

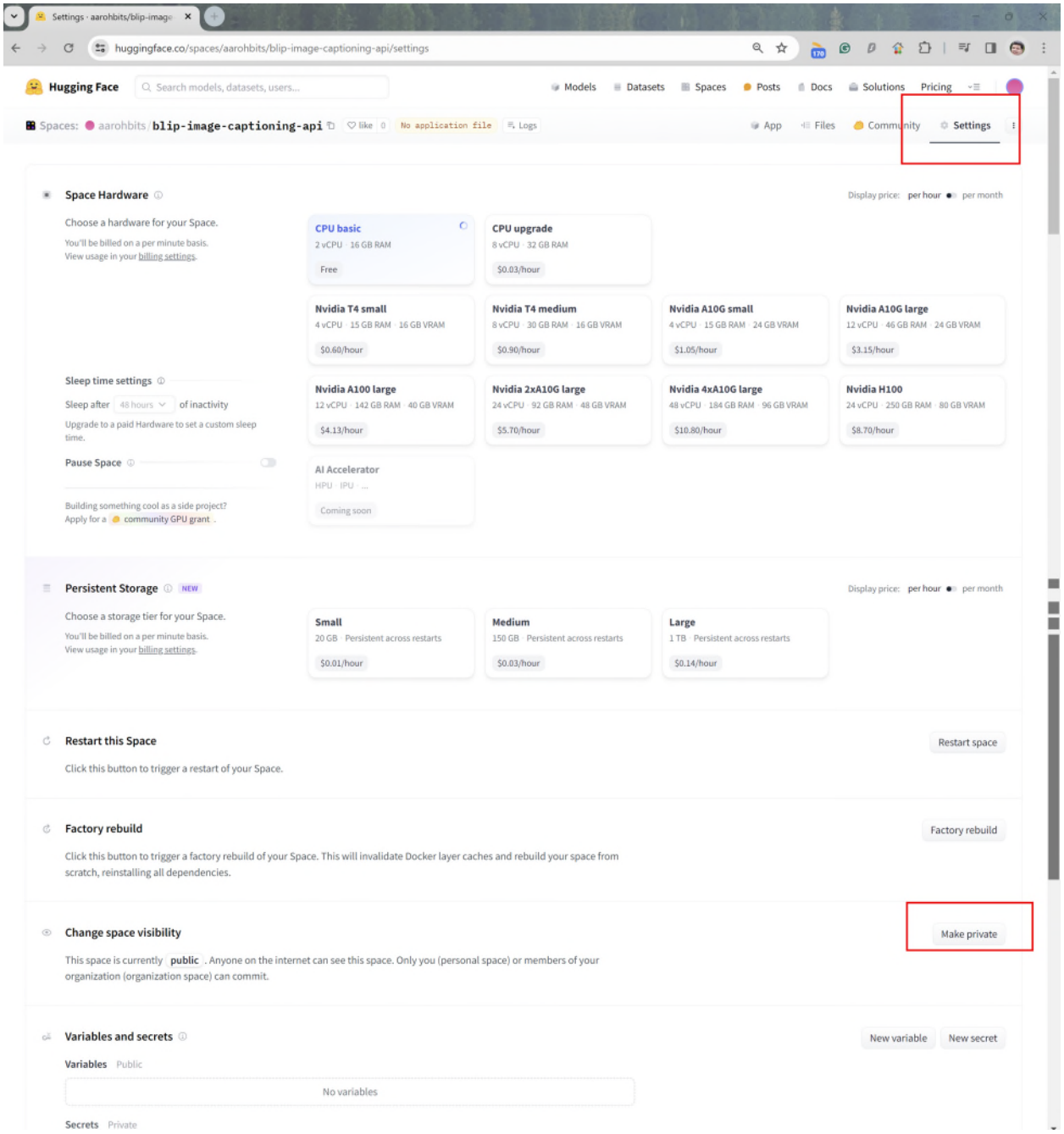

Spaces
eddyS/blip-image-captioning-api
like 0
Running
input
output
Drop Image H
- or -
Click to Upload
Clear
Submit

Use via API

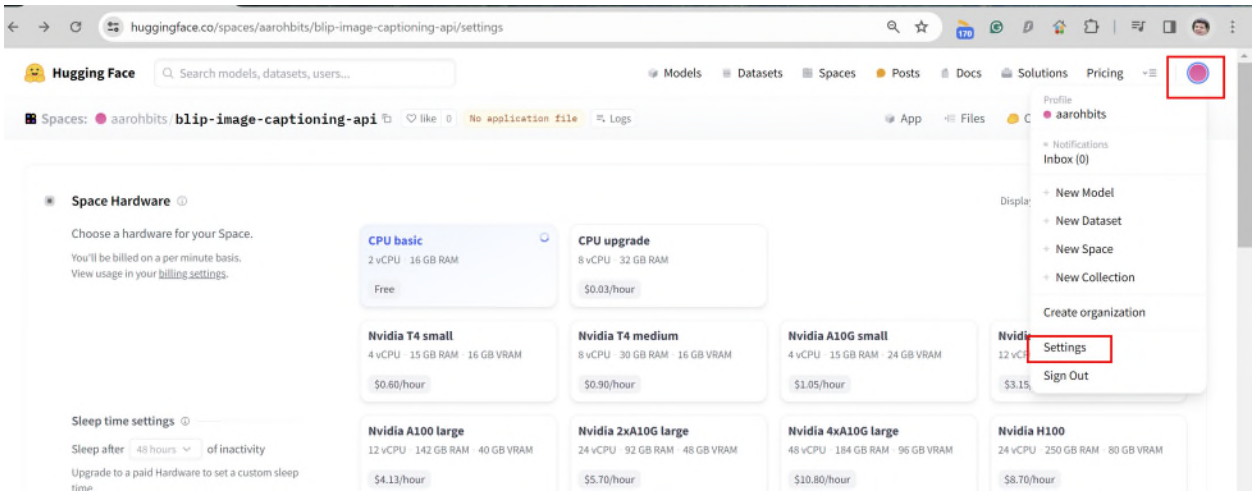

Spaces
eddyS / blip-image-captioning-api
like 0
Running
input
output
two cats sleeping on a couch
Clear
Submit
Use via API
Built with Gradio


Spaces
eddyS / blip-image-captioning-api
like 0
Running
API documentation
https://eddys-blip-image-captioning-api.hf.space/--replicas/8swgs/
1 API endpoint
Use the gradio_client Python library or the @gradio/client Javascript package to query the demo via API.
Python
Javascript
$ pip install gradio_client
copy
Endpoints
api_name: /predict
from gradio_client import Client
copy
client = Client("https://eddys-blip-image-captioning-api.hf.space/--replicas/8sv
result = client.predict(
"https://raw.githubusercontent.com/gradio-app/gradio/main/test/test_files/b
api_name="/predict"
)
print(result)
Return Type(s)
str representing output in 'output' Textbox component

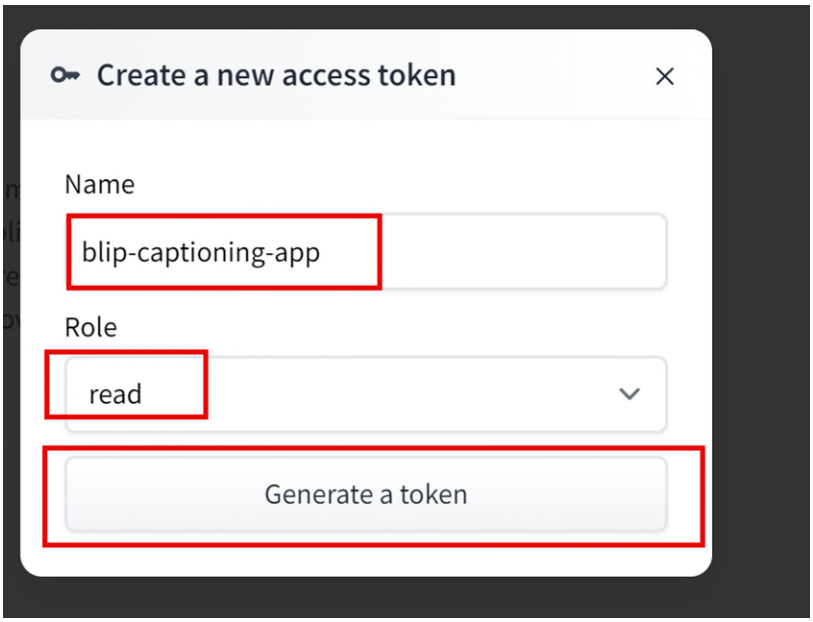
Make Space private



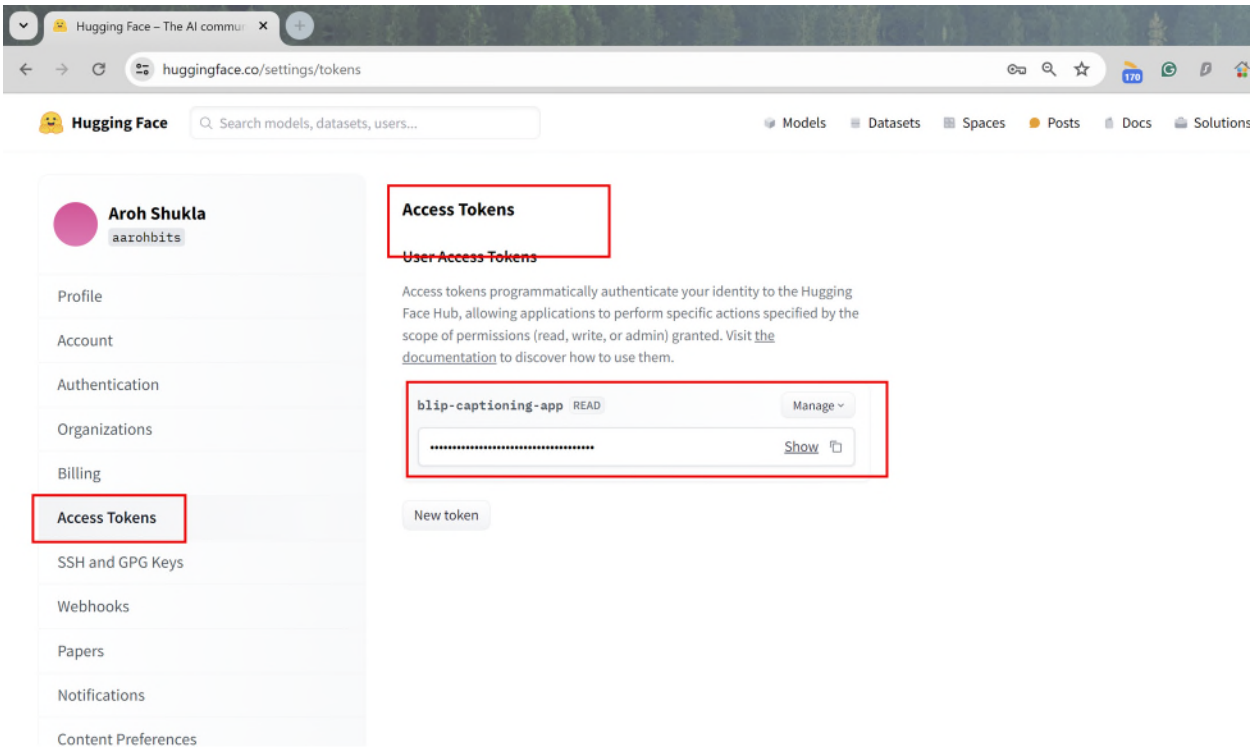
Access Tokens



- In your profile settings, on the left side menu, click "Access Tokens".
- Click "New token".



- You can copy the access token.




```
from gradio_client import Client


client = Client("eddyS/blip-image-captioning-2",
                hf_token=hf_access_token
               )
result = client.predict(
        "kittens.jpg",
        api_name="/predict"
)
print(result)
# client = Client("abidlabs/whisper-large-v2",
)
```

Saving your access token securely

GPU Zero Space

A place to spin free GPUs on demand for your spaces.

https://huggingface.co/zero-gpu-explorers

You can click "request to join this org".

It may take a few days/weeks for this to be approved.

Conclusion

Reference

1. Open Source Models with Hugging Face
   https://learn.deeplearning.ai/courses/open-source-models-hugging-face/

2.