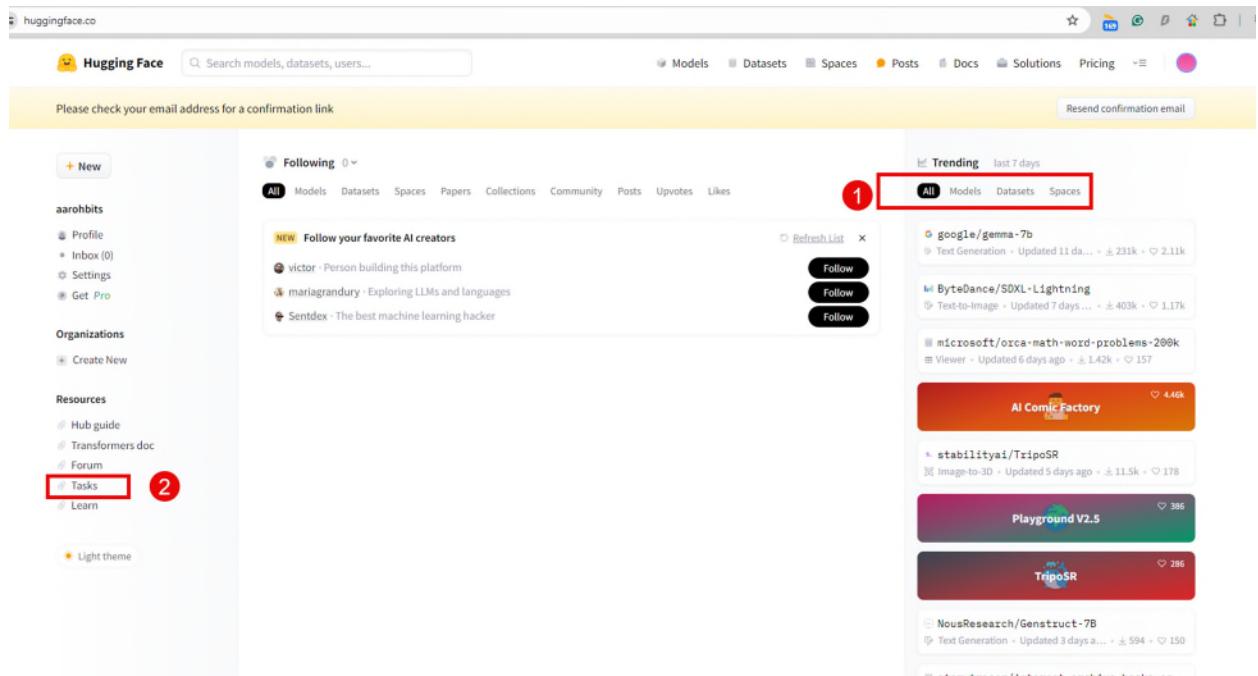


Open Source Models with Hugging Face

Select Models

Open Source Models

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



- Select Automatic Speech Recognition

The screenshot shows the huggingface.co/tasks interface. At the top, there's a search bar with 'huggingface.co/tasks' and some icons. Below it, there are several categories of tasks:

- Audio**: Includes Translation (3,379 models), Zero-Shot Classification (222 models), Audio Classification (1,877 models), Audio-to-Audio (3,703 models), and Automatic Speech Recognition (14,941 models, highlighted with a red box).
- Tabular**: Includes Text-to-Speech (1,814 models) and Tabular Classification (184 models).
- Multimodal**: Includes Tabular Regression (105 models), Document Question Answering (156 models), and Visual Question Answering (170 models).
- Reinforcement Learning**: Includes Reinforcement Learning (38,644 models).

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

1. **Automatic Speech Recognition demo**

This section shows a demo interface for ASR. It has an 'Inputs' section with a video player showing a person speaking, and an 'Output' section displaying a transcript: "Going along slushy country roads and speaking to damp audiences in...".

2. **Models for Automatic Speech Recognition**

This section lists various ASR models:

- openai/whisper-large-v3** (Automatic Speech Recognition - Updated about 1 month ago - ± 873k - ⚡ 1.92k)
- facebook/wav2vec2-base-960h** (Automatic Speech Recognition - Updated Nov 14, 2022 - ± 15.5M - ⚡ 204)
- facebook/s2t-small-mustc-en-fr-st** (Automatic Speech Recognition - Updated Jan 24, 2023 - ± 2.63K - ⚡ 1)

3. **Automatic Speech Recognition**

This section features a video player titled 'Automatic Speech Recognition' with a thumbnail of a man and the text 'with Omar'. Below the video, there are 'Use Cases' and a 'Tasks' section.

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

Please check your email address for a confirmation link

Models 328

Filter by name

French

- openai/whisper-large-v3
- nvidia/canary-1b
- openai/whisper-large-v2
- openai/whisper-small
- openai/whisper-medium
- openai/whisper-large

• Select License

Please check your email address for a confirmation link

Models 243

Filter by name

apache-2.0

- openai/whisper-large-v3
- openai/whisper-small
- openai/whisper-medium
- pierreguillou/whisper-medium-french
- Ilyes/wav2vec2-large-xlsr-53-french
- MehdiHosseiniMoghadam/wav2vec2-large-xlsr-53-French
- Plim/test_lm
- Illyes/wav2vec2-large-xlsr-53-french_punctuation
- Nhut/wav2vec2-large-xlsr-french
- Plim/xls-r-1b-cv_8-fr

• Model Card

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

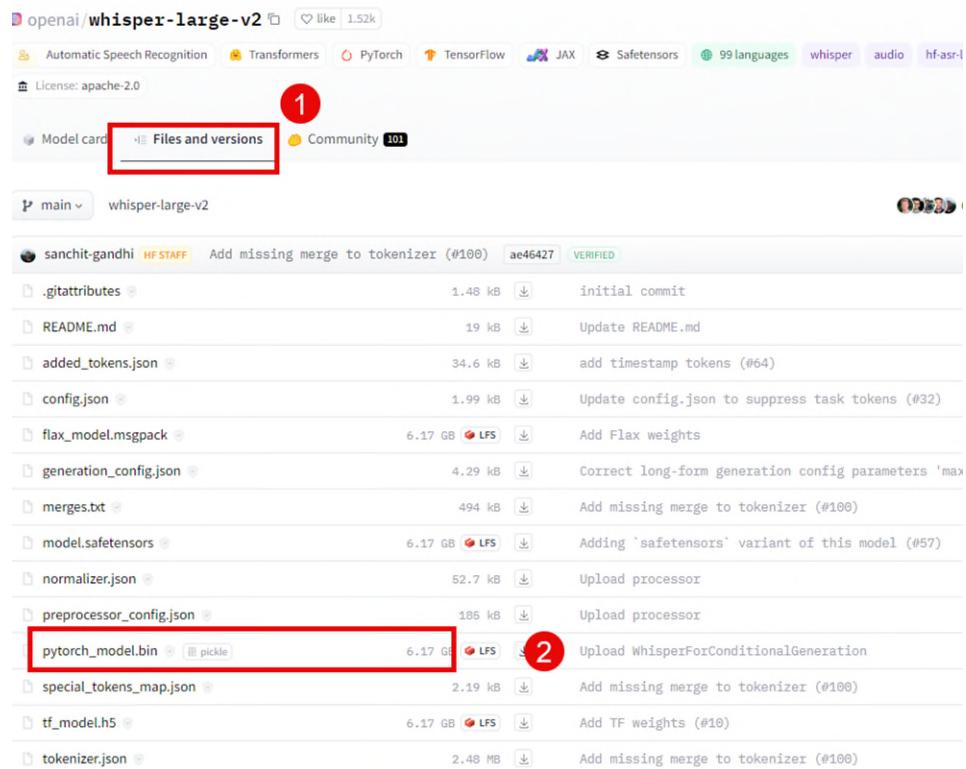
- Inference API

The screenshot shows the Hugging Face Model Hub interface for the Whisper model. It includes:

- Model card:** A detailed page for the Whisper model, including its description, training data, and code repository.
- File and versions:** A section showing the Pytorch_model.bin file size (1550 M).
- Community:** A section showing recent downloads (180,121) and a waveform visualization.
- Inference API:** An interface for audio transcription. It includes a file browser, a microphone icon for recording from the browser, and a play button for audio samples. A green box highlights the transcription text: "going along slushy country roads and speaking to damp audiences in draughty schoolrooms day after day for a fortnight. He'll have to put in an appearance at some place of worship on Sunday morning, and he can come to us immediately afterwards."
- Model details:** A table comparing different model sizes based on size, parameters, English-only support, and multilingual support.

Size	Parameters	English-only	Multilingual
tiny	39 M	✓	✓
base	74 M	✓	✓
small	244 M	✓	✓
medium	769 M	✓	✓
large	1550 M	x	✓
large-v2	1550 M	x	✓

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



- o 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**

The screenshot shows the Hugging Face model card for the 'whisper-large-v2' model. The 'Files and versions' tab is active. At the top right, there are buttons for 'Train', 'Deploy', and 'Use in Transformers', with the last one being highlighted by a red box. Below the tabs, there's a commit history table showing recent changes made by 'sanchit.gandhi'. The commits include updates to the tokenizer, README, config.json, and Flax weights, along with timestamp tokens and safetensors variants. The table includes columns for author, commit message, file, size, and date.

Use Pipeline object

This screenshot shows the 'How to use from the library' section of the model card. It contains two code snippets. The first snippet demonstrates how to use a pipeline as a high-level helper:

```
# Use a pipeline as a high-level helper
from transformers import pipeline

pipe = pipeline("automatic-speech-recognition", model="openai/whisper-large-v2")
```

The second snippet shows how to load the model directly:

```
# Load model directly
from transformers import AutoProcessor, AutoModelForSpeechSeq2Seq

processor = AutoProcessor.from_pretrained("openai/whisper-large-v2")
model = AutoModelForSpeechSeq2Seq.from_pretrained("openai/whisper-large-v2")
```

Below the code, there's a 'Quick Links' section with links to model documentation, docs on high-level pipeline, and learning resources.

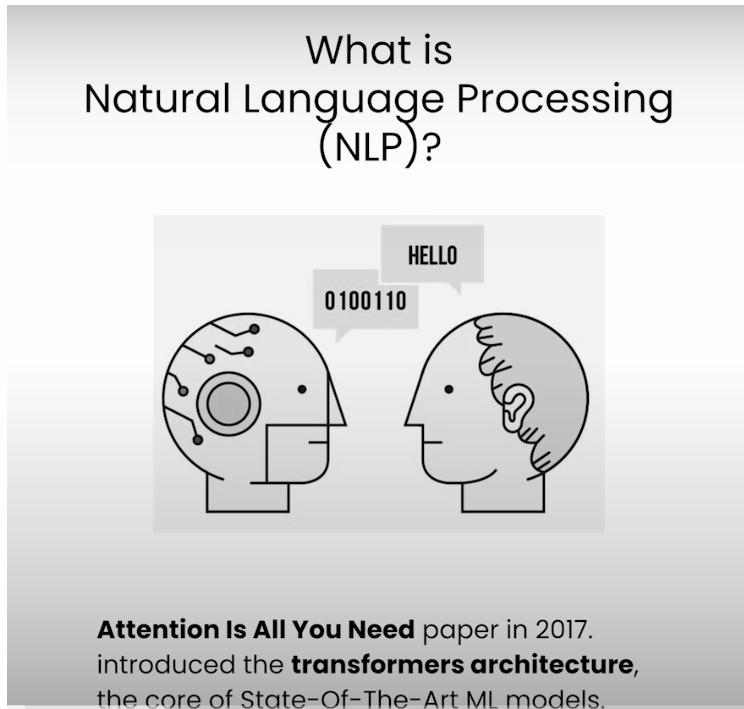
Natural Language Processing (NLP)

- 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**

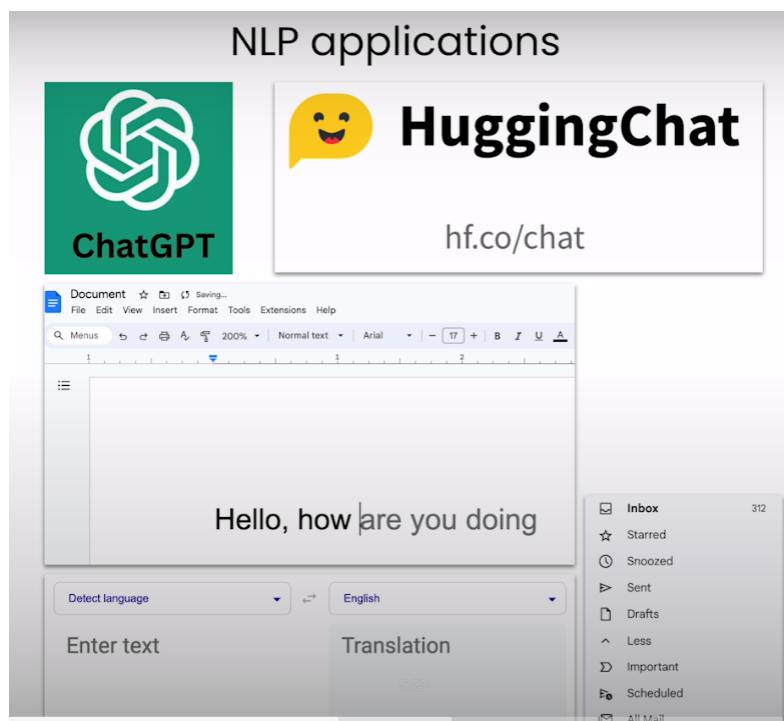
```
from transformers.utils import logging  
logging.set_verbosity_error()
```

- 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)*

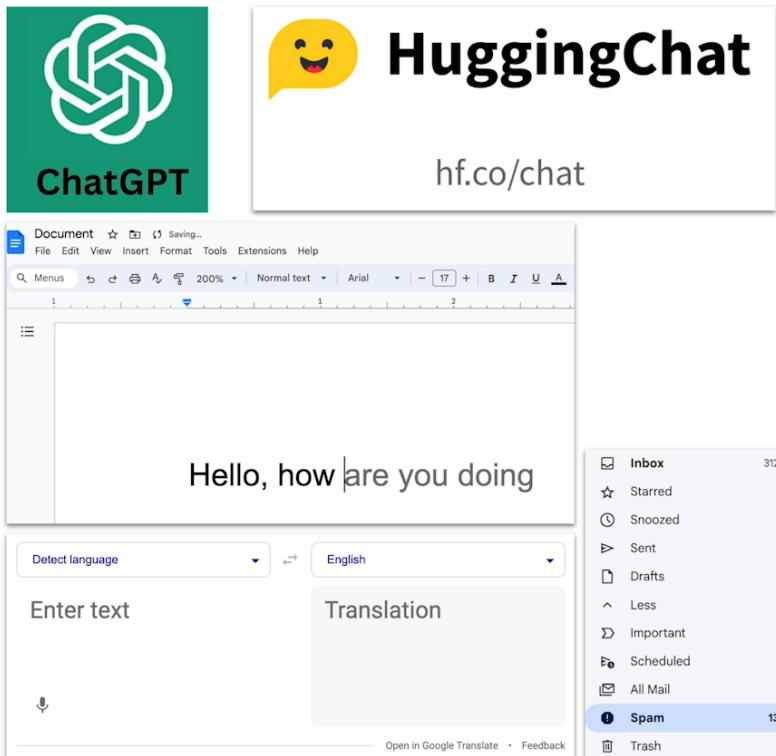


NLP Applications

🤗 Hugging Face

.DeepLearning.AI

NLP applications



NLP Tasks



Hugging Face



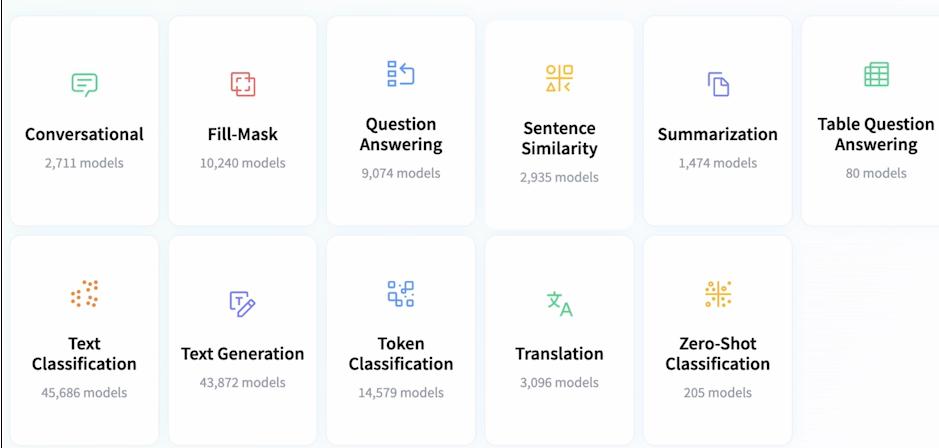
DeepLearning.AI

NLP tasks

Here are some common NLP tasks:

- Text generation
- Sentence similarity
- Summarization
- Machine translation

Natural Language Processing



Which one to choose from?

As there are so many open source models.

The screenshot shows the Hugging Face Model Hub interface. At the top, there are two logos: "Hugging Face" on the left and "DeepLearning.AI" on the right. Below the logos, the title "Let's get NLP models" is displayed. A navigation bar at the top includes tabs for "Tasks", "Libraries", "Datasets", "Languages", "Licenses", and "Other". A search bar with the placeholder "Filter Tasks by name" is also present. The main content area is titled "Models 469,367" and features a "Filter by name" input field. The models are listed in a grid format, each with a thumbnail, the model name, a brief description, and metrics like updates, size, and downloads. A red box highlights the "Natural Language Processing" section, which contains sub-categories: Text Classification, Token Classification, Table Question Answering, Question Answering, Zero-Shot Classification, Translation, Summarization, Conversational, Text Generation, Text2Text Generation, and Fill-Mask. Below the grid, the URL "https://huggingface.co/models" is shown.

Select **Filters** and use **Transformers**

The screenshot shows a pop-up window titled "How to use from the • Transformers ⚡ library". It contains code snippets for using the library. The first snippet shows how to use a pipeline as a high-level helper, and the second snippet shows how to load a model directly. Both snippets have a "Copy" button next to them, with the first one highlighted by a red box. Below the code snippets, there is a "Quick Links" section with three items: "Read model documentation", "Read docs on high-level-pipeline", and "Read our learning resources".

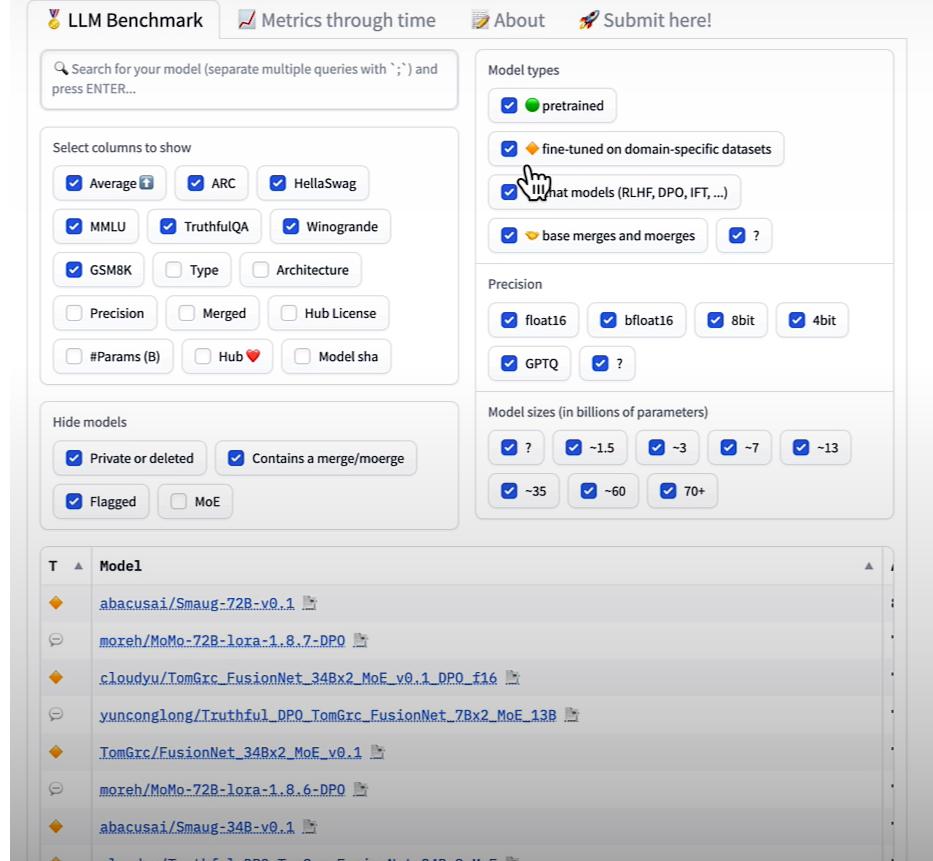
LLM Leaderboard

- Pretrained are modelled from scratch.

great [Eleuther AI Language Model Evaluation Harness](#) - read more details in the "About" page!

Other cool leaderboards:

- [LLM safety](#)
- [LLM performance](#)

The screenshot shows the LLM Benchmark interface. At the top, there's a search bar with placeholder text: "Search for your model (separate multiple queries with `;` and press ENTER...)" and tabs for "Metrics through time", "About", and "Submit here!". Below the search bar are sections for "Model types", "Precision", and "Model sizes (in billions of parameters)". The "Model types" section includes checkboxes for "pretrained", "fine-tuned on domain-specific datasets", "Chat models (RLHF, DPO, IFT, ...)", and "base merges and moerges". The "Precision" section includes checkboxes for "float16", "bf16", "8bit", "4bit", and "GPTQ". The "Model sizes" section includes checkboxes for parameter counts like "?", "-1.5", "-3", "-7", "-13", "-35", "-60", and "70+". A large table below these filters lists various LLM models, each with a small icon and a link. The columns are labeled "Model", "Type", "Dataset", "Precision", "Size", and "MoE". Some models listed include "abacusai/Smaug-72B-v0.1", "moreh/MoMo-72B-lora-1.8.7-DPO", "cloudyu/TomGrc_FusionNet_34Bx2_MoE_v0.1_DPO_f16", "yunconglong/Truthful_DPO_TomGrc_FusionNet_7Bx2_MoE_13B", "TomGrc/FusionNet_34Bx2_MoE_v0.1", "moreh/MoMo-72B-lora-1.8.6-DPO", and "abacusai/Smaug-34B-v0.1".

Translation and Summarization

- English to French – (Translation)

Open Source Models with
Hugging Face

Translation
and Summarization



Hugging Face



DeepLearning.AI

Zero-Shot Audio Classification

Open Source Models with
Hugging Face

Zero-Shot
Audio Classification

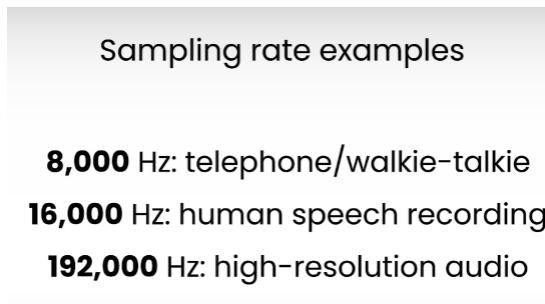
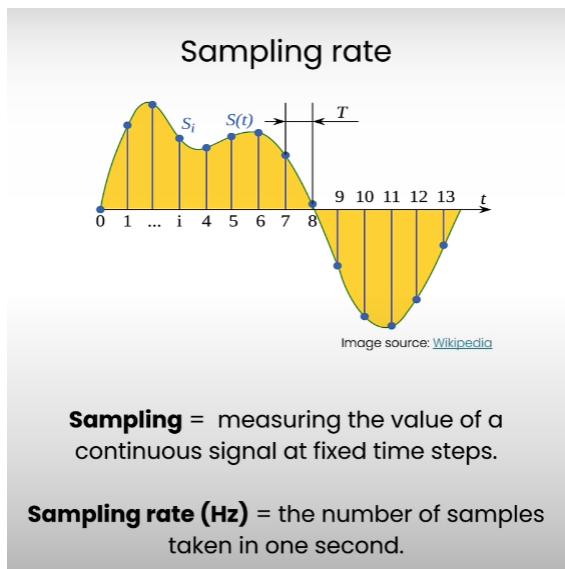
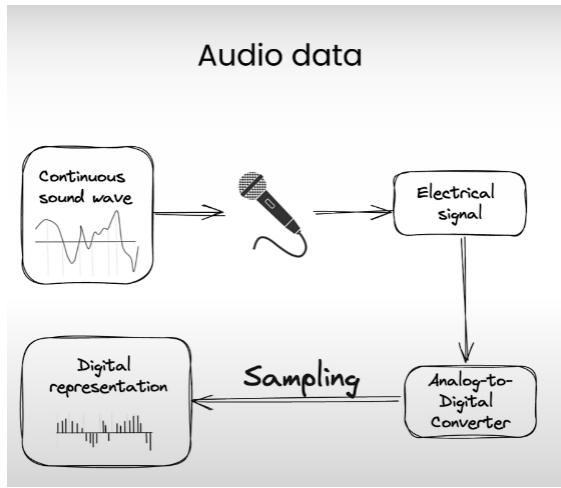


Hugging Face



DeepLearning.AI

Audio Data



Sampling rate for transformer models

5-second sound

At 8,000 Hz => $5 \times 8000 = 40,000$ signal values

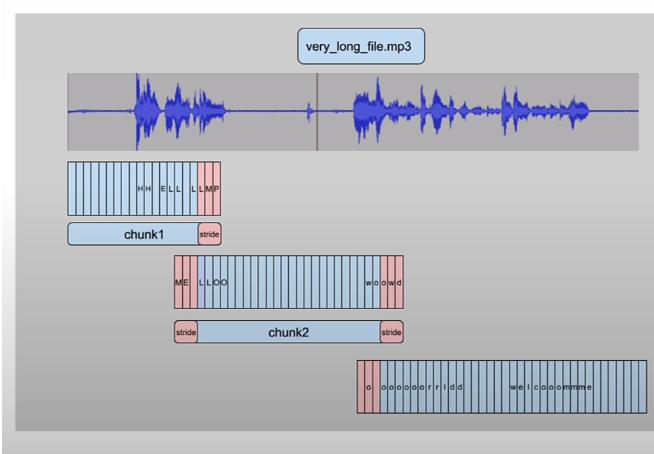
At 16,000 Hz => $5 \times 16000 = 80,000$ signal values

At 192,000 Hz => $5 \times 192000 = 960,000$ 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 ($60 \times 16,000 = 960,000$).

#Lesson 6: LAB 1: Automatic Speech Recognition

Transcribing long recordings

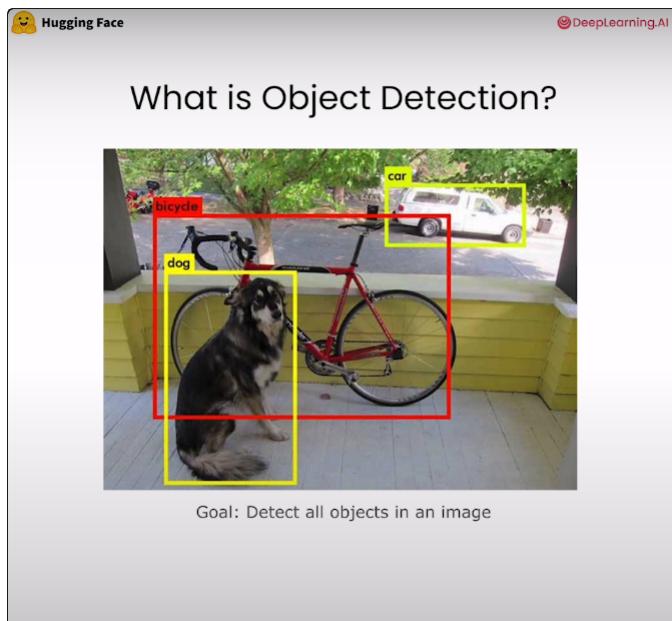


Lesson 7: Text to Speech

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

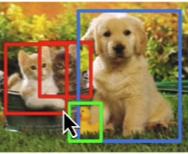
- Helping visually impaired person what is in the picture.
- Info about [facebook/detr-resnet-50](#)
- Explore more of the [Hugging Face Hub for more object detection models](#)

What is object detection?



 **Hugging Face** 

What is Object Detection?

Classification	Classification + Localization	Object Detection
 CAT	 CAT	 CAT, DOG, DUCK

With Object detection you can classify and localise at the same time!

 **Hugging Face** 

How to browse SoTA Object detection models?

You Only Look Once: Unified, Real-Time Object Detection

Tasks  Libraries Datasets Languages Licenses Other

Filter Tasks by name  Models 1,304 Filter by name 

-  [microsoft/table-transformer-detection](#)  Object Detection - Updated Sep 6 -  1.39M  122
-  [facebook/detr-resnet-50](#)  Object Detection - Updated Oct 17 -  5.85k  353
-  [nustvl/yolos-tiny](#)  Object Detection - Updated Jun 5 -  283k  159
-  [keremberke/yolov5n-garbage](#)  Object Detection - Updated Jan 5 -  633  8
-  [hustvl/yolos-small](#)  Object Detection - Updated Jun 27, 2022 -  82.5k  40
-  [nickmuchi/yolos-small-finetuned-license-plate-detec...](#)  Object Detection - Updated Jan 10 -  3.15k  21

Computer Vision

- Depth Estimation Image Classification
- Object Detection Image Segmentation
- Image-to-Image Unconditional Image Generation
- Video Classification Zero-Shot Image Classification
- Mask Generation Zero-Shot Object Detection

View all tasks  Just like in YOLOv5, our approach is motivated by the multi-task learning paradigm in DETR. We highlight on the comment embeddings for localizing the four extremities and predicting the bounding box, which increases the need for high-quality content embeddings.

Figure 1. Comparison of spatial attention weight maps for our con...

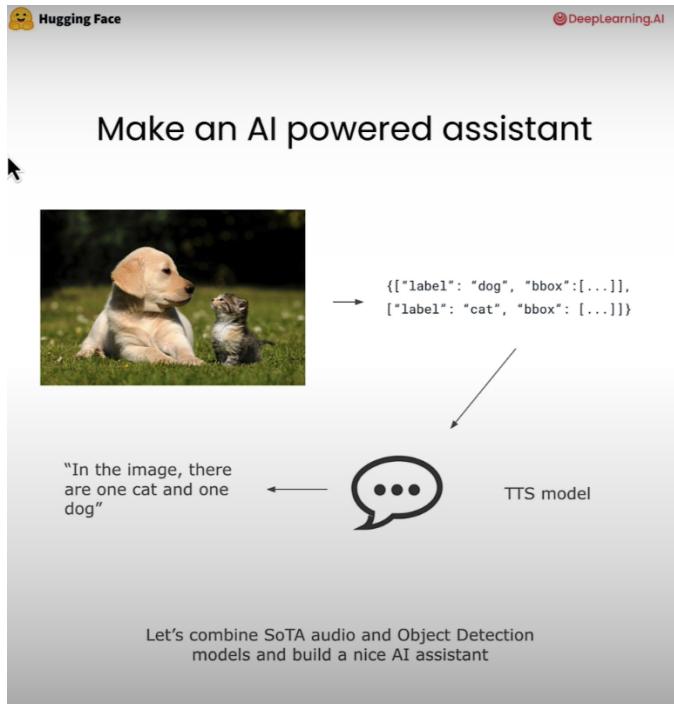
Sample Image



Final demo

The screenshot shows a web-based application interface for generating images. At the top left is the Hugging Face logo, and at the top right is the DeepLearning.AI logo. The main title "Make a Gradio demo!" is centered above a input field labeled "Prompt". The prompt "a lion in the water" is entered. To the right of the prompt is a generated image of a lion's head above water. Below the image is a "Regenerate" button. At the bottom of the screen, a message reads: "With Gradio you can showcase your work to anyone easily!"

LAB 2: Make an AI Powered assistant



Lesson 9: Segmentation

- Visual prompting
- Meta AI

What is Segmentation Mask Generation?



Hugging Face



DeepLearning.AI

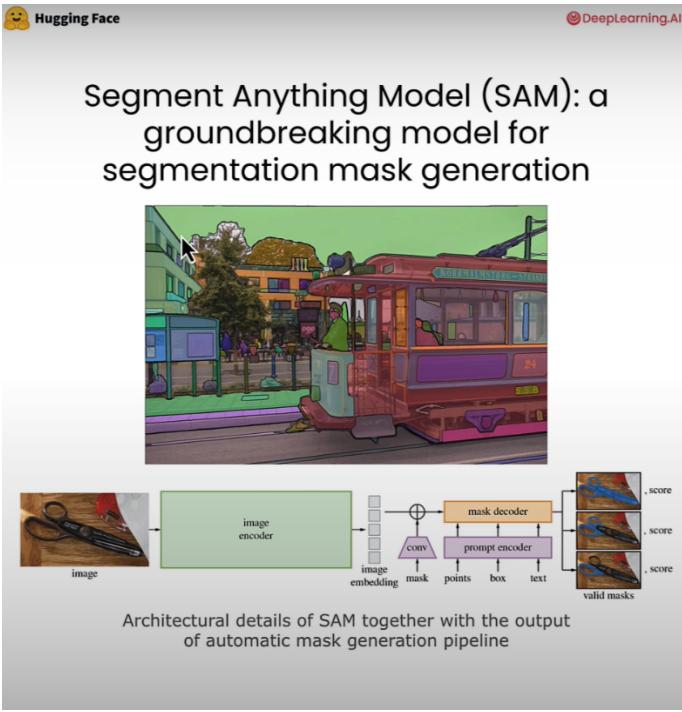
What is Segmentation Mask
Generation?



Image Segmentation



Segment Any model (SAM)



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

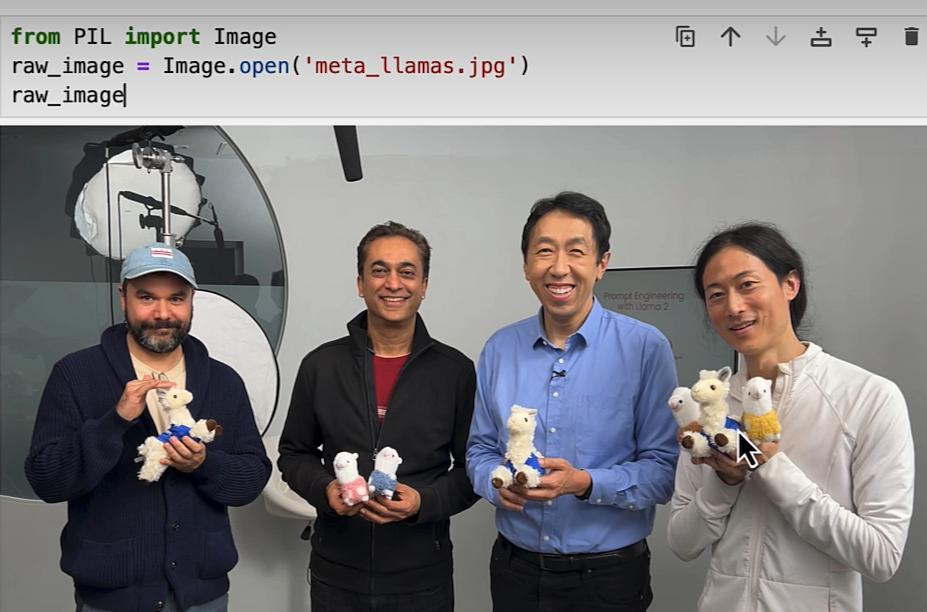
Abstract

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 alternative fashion. Further-



284v2 [cs.CV] 12 Dec 2023

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



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

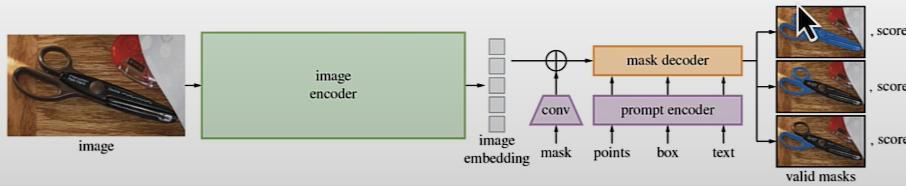
```
from helper import show_pipe_masks_on_image
```

↶ ↗ ↘ ↙ ↛ ↜

```
show_pipe_masks_on_image(raw_image, output)|
```



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



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

```
output = prediction.squeeze().numpy()
formatted = (output * 255 / np.max(output)).astype("uint8")
depth = Image.fromarray(formatted)
```

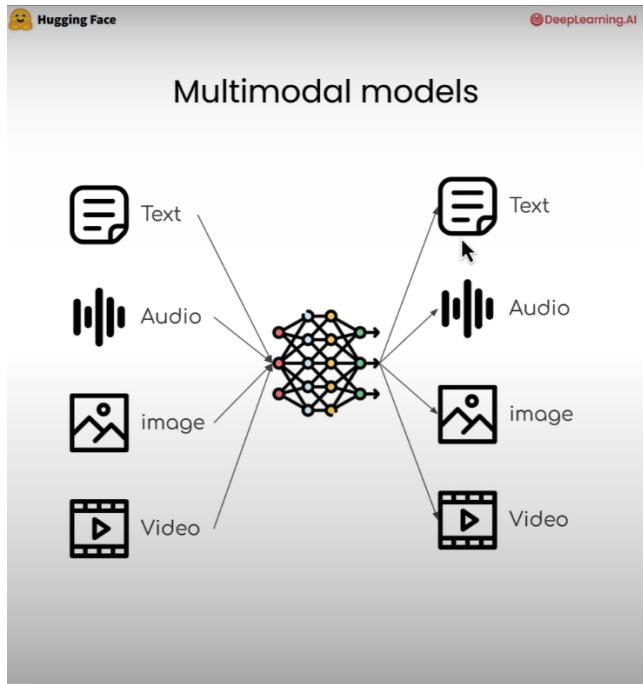
depth|



Lesson 10: Image Retrieval

- An Image Retrieval from Salesforce.

- Multimodal models



Hugging Face DeepLearning.AI

Applications

ChatGPT can now see, hear, and speak

We are beginning to roll out new voice and image capabilities in ChatGPT. They offer a new, more intuitive type of interface by allowing you to have a voice conversation or show ChatGPT what you're talking about.

IDEFICS - the open multimodal ChatGPT

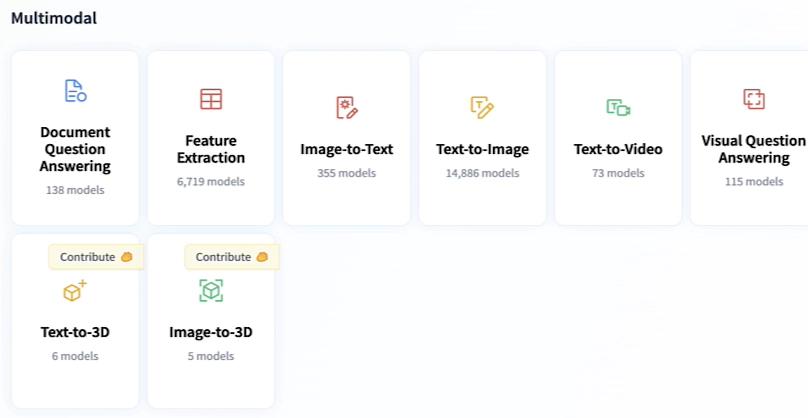
Based on the image is there "coke" in the fridge?

Yes, there is a bottle of Coke in the fridge.

Multimodal Tasks

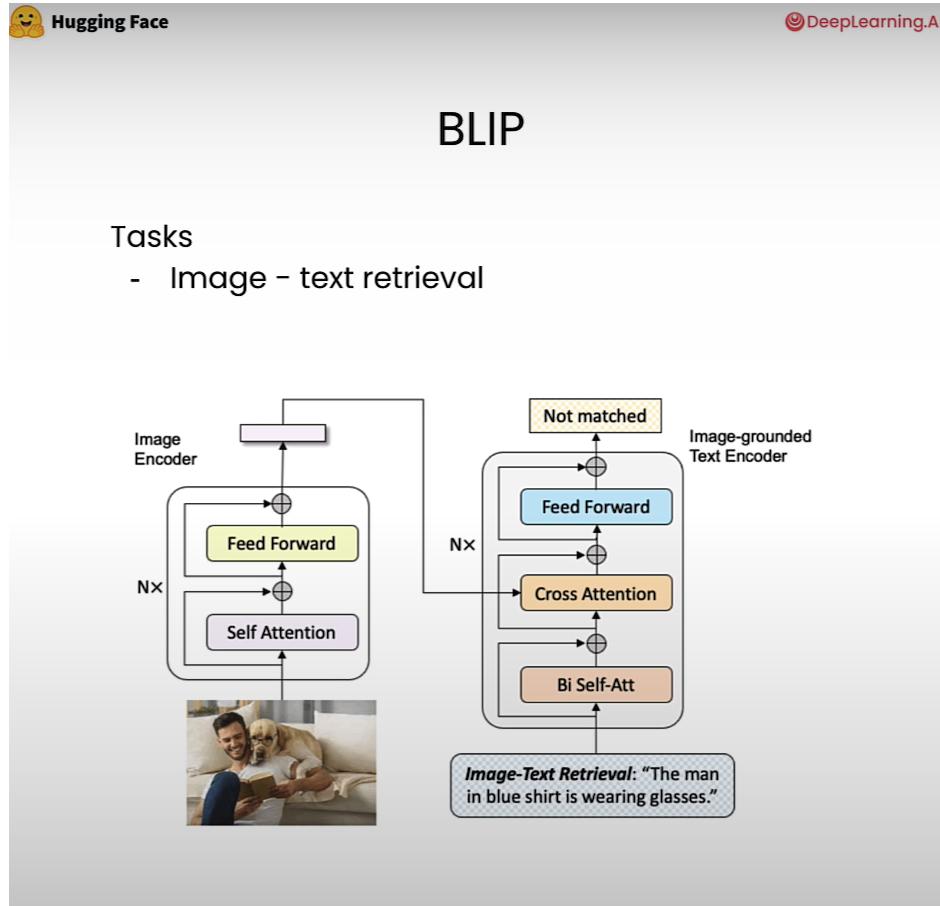
Here are some common multimodal tasks that we will go through:

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



Multimodal Tasks from Hugging Face Hub

BLIP



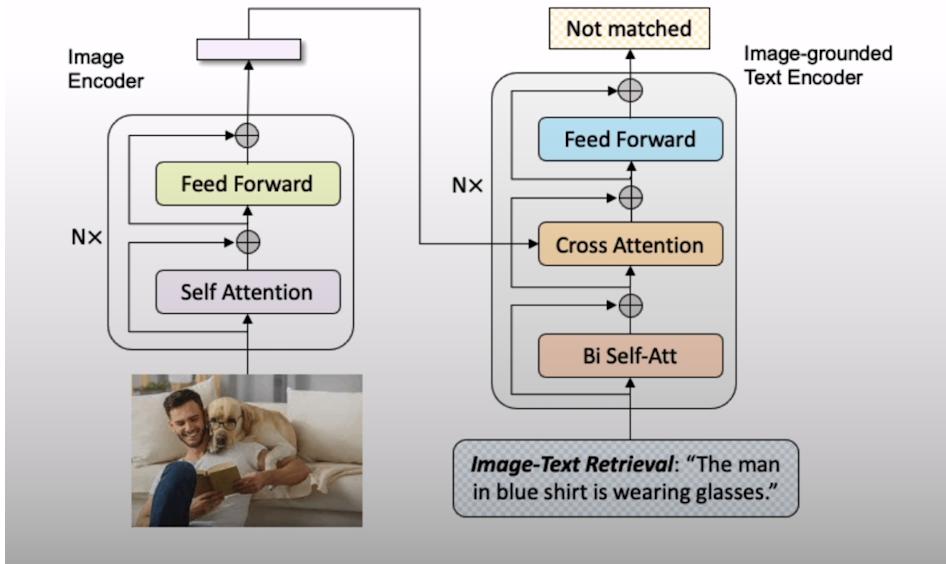
Lesson 11: Image Captioning

BLIP

Bootstrapping Language-Image Pre-training

Task

- Image - text retrieval

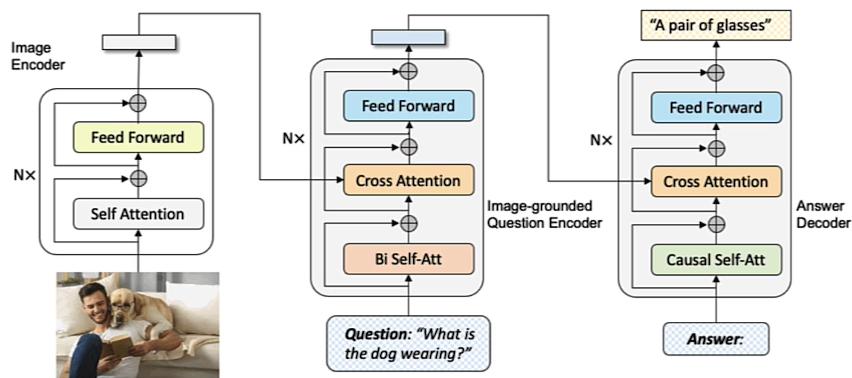


Lesson 12: Visual Question & Answering

BLIP

Tasks

- Visual Q&A



Lesson 13: Zero-Shot Image Classification

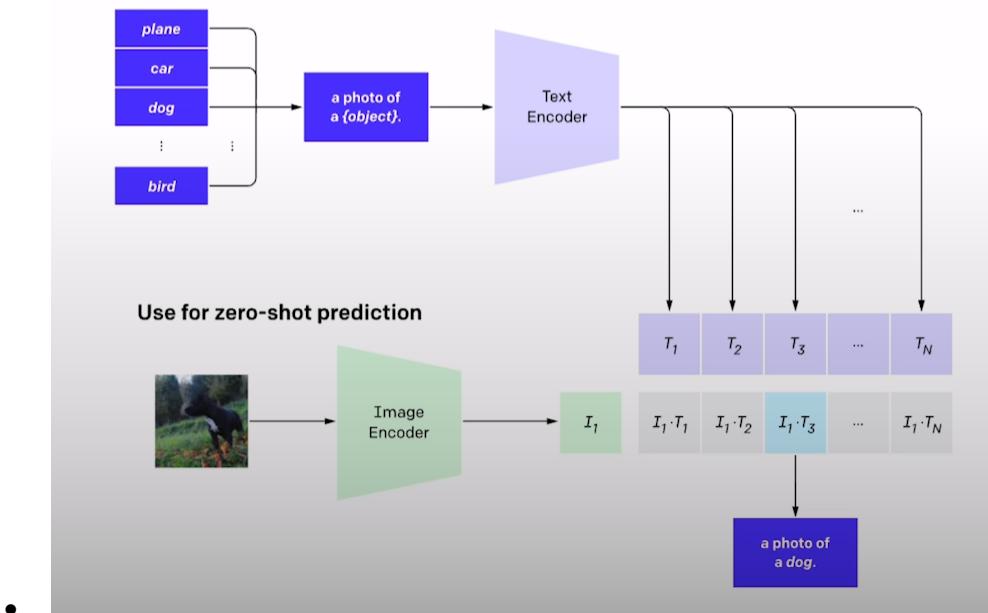
- OpenAI



CLIP

Contrastive Language-Image Pre-Training

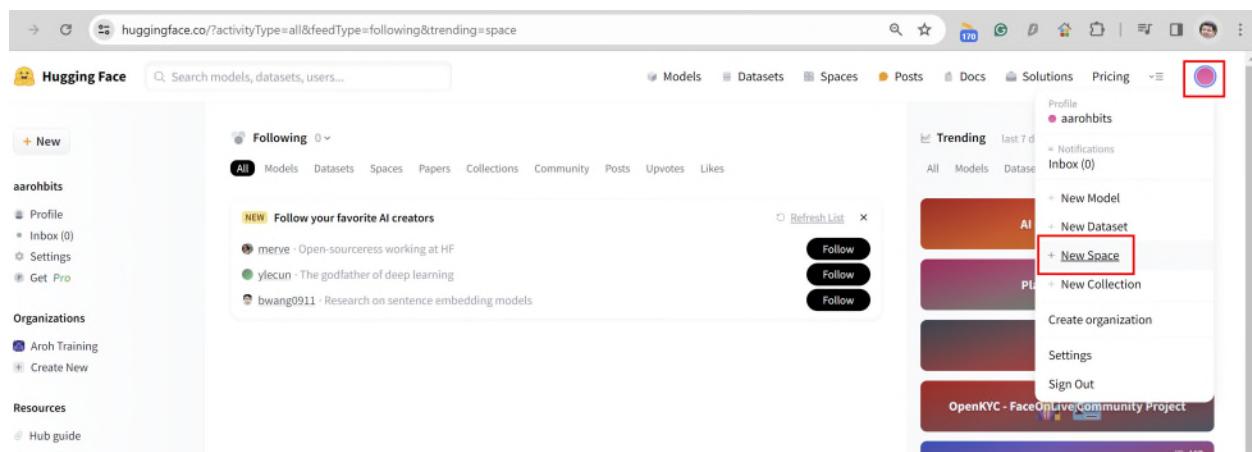
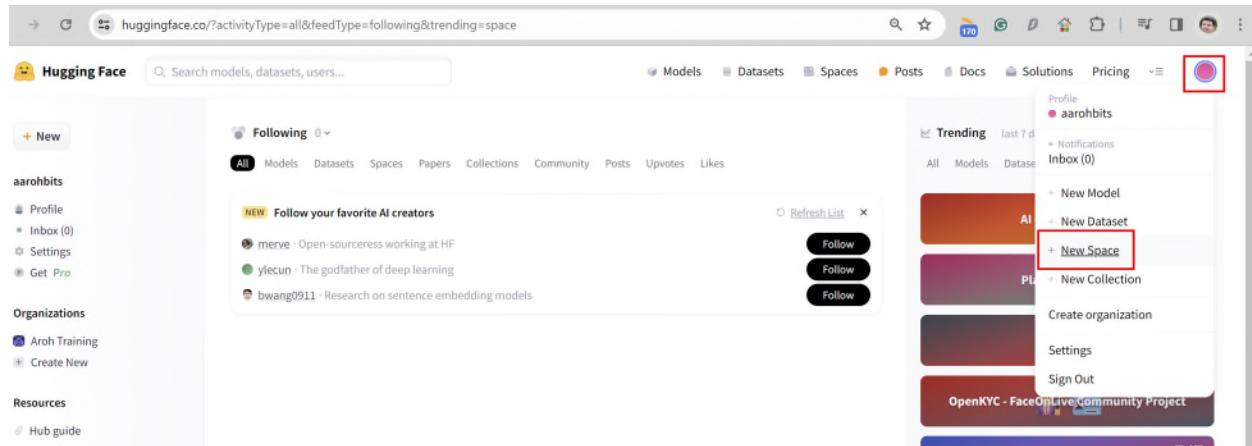
CLIP is a multi-modal vision and language model.
It can be used for zero-shot image classification.



Lesson 14: Deploy ML Models on 🤗 Hub using Gradio

- Salesforce

Create a new space



Need to create 2 files for this project

1 huggingface.co/spaces/aarohbits/blip-image-captioning-api

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

```
# 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
```

1 huggingface.co/spaces/aarohbits/blip-image-captioning-api/tree/main

2

3

Hugging Face Search models, datasets, users...

Spaces: aarohbits blip-image-captioning-api 0 like 0 No application file Logs

App Files Community Settings

main · blip-image-captioning.api

aarohbits initial commit 6747a4c VERIFIED

.gitattributes 1.52 kB initial commit

README.md 248 Bytes initial commit

1 contributor History: 1 commit

+ Add file v Create a new file Upload files 2 minutes ago

1. Requirements.txt

The screenshot shows a web-based form for creating a requirements.txt file. At the top, there's a navigation bar with a back arrow, forward arrow, and a refresh icon. The URL in the address bar is `huggingface.co/spaces/aarohbits/blip-image-captioning-api/new/main`. Below the address bar, there's a header with the project name "blip-image-captioning-api/" and the file name "requirements.txt". A red box highlights the "Edit" tab of the file content area. The content of the file is:

```
1 transformers
2 torch
3 gradio
```

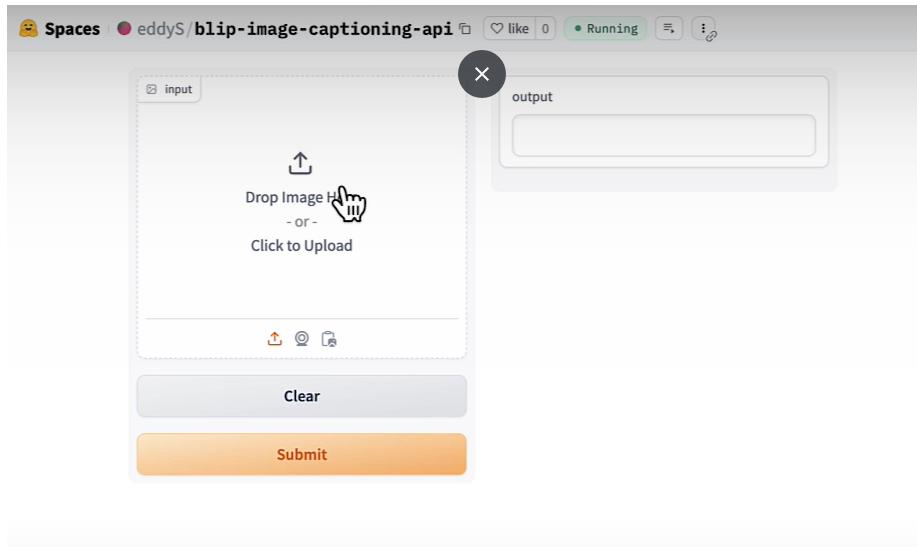
Below the file content, there are two options for committing:

- Commit directly to the `main` branch
- Open as a pull request to the `main` branch

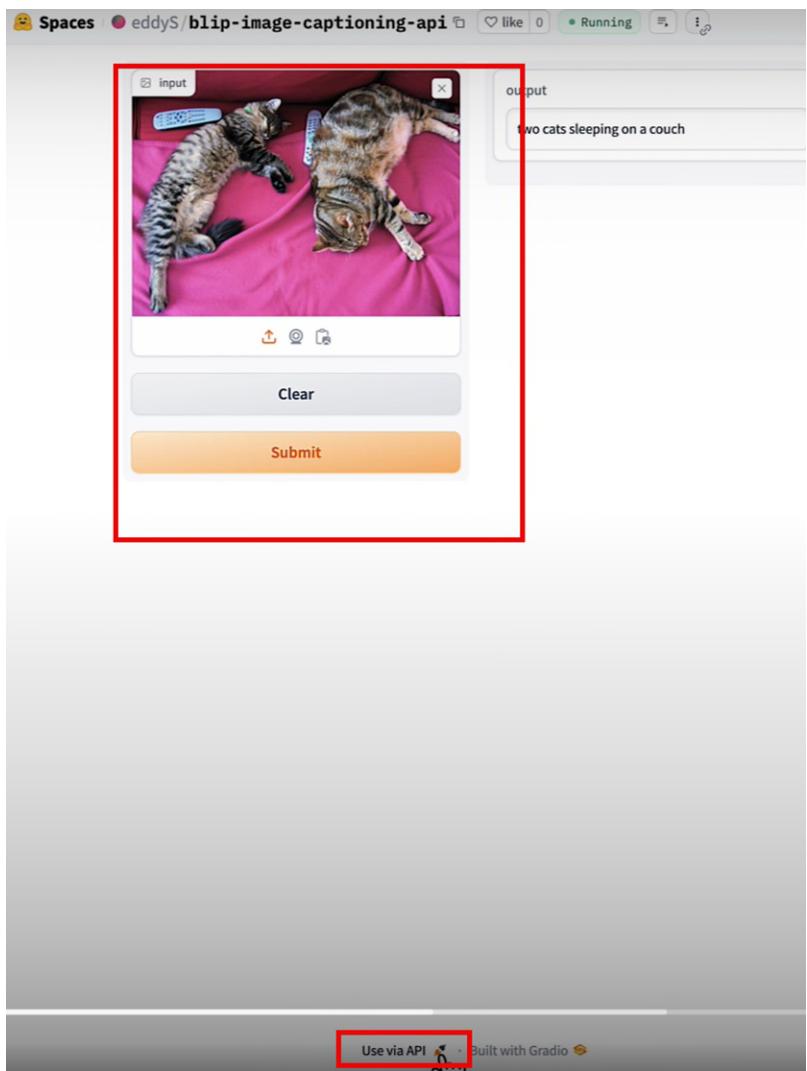
Underneath, there's a section for "Commit new file" with a text input field labeled "Create requirements.txt" and a red box highlighting the "Edit" tab. There's also a text area for "Add an extended description..." and a note about uploading files. At the bottom, there are two buttons: "Commit new file to `main`" (highlighted with a red box) and "Cancel".

2. App.py

Copy and paste the python code



Use via API



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.

```
$ 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/8swgs")
result = client.predict(
    "https://raw.githubusercontent.com/gradio-app/gradio/main/test/test_files/bear.jpg",
    api_name="/predict"
)
print(result)
```

• Return Type(s)

```
# str representing output in 'output' Textbox component
```

Make Space private

The screenshot shows the 'Settings' page for a space on huggingface.co. The top navigation bar includes tabs for Models, Datasets, Spaces, Posts, Docs, Solutions, Pricing, App, Files, Community, and Settings (which is highlighted with a red box). Below the navigation is a search bar and a breadcrumb trail showing the space name.

Space Hardware (Display price: per hour • per month)

Choose a hardware for your Space. You'll be billed on a per minute basis. View usage in your [billing settings](#).

Hardware Type	Configuration	Price
CPU basic	2 vCPU 16 GB RAM	Free
CPU upgrade	8 vCPU 32 GB RAM	\$0.03/hour
Nvidia T4 small	4 vCPU 15 GB RAM 16 GB VRAM	\$0.60/hour
Nvidia T4 medium	8 vCPU 30 GB RAM 16 GB VRAM	\$0.90/hour
Nvidia A10G small	4 vCPU 15 GB RAM 24 GB VRAM	\$1.05/hour
Nvidia A10G large	12 vCPU 46 GB RAM 24 GB VRAM	\$3.15/hour
Nvidia A100 large	12 vCPU 142 GB RAM 40 GB VRAM	\$4.13/hour
Nvidia 2xA10G large	24 vCPU 92 GB RAM 48 GB VRAM	\$5.70/hour
Nvidia 4xA10G large	48 vCPU 184 GB RAM 96 GB VRAM	\$10.80/hour
Nvidia H100	24 vCPU 250 GB RAM 80 GB VRAM	\$8.70/hour

Sleep time settings (Display price: per hour • per month)

Sleep after 48 hours of inactivity. Upgrade to a paid Hardware to set a custom sleep time.

Pause Space (Display price: per hour • per month)

Building something cool as a side project? Apply for a [community GPU grant](#).

Persistent Storage (Display price: per hour • per month)

Choose a storage tier for your Space. You'll be billed on a per minute basis. View usage in your [billing settings](#).

Tier	Capacity	Price
Small	20 GB Persistent across restarts	\$0.01/hour
Medium	150 GB Persistent across restarts	\$0.03/hour
Large	1 TB Persistent across restarts	\$0.14/hour

Restart this Space (Display price: per hour • per month)

Click this button to trigger a restart of your Space.

Factory rebuild (Display price: per hour • per month)

Click this button to trigger a factory rebuild of your Space. This will invalidate Docker layer caches and rebuild your space from scratch, reinstalling all dependencies.

Change space visibility (Display price: per hour • per month)

This space is currently **public**. Anyone on the internet can see this space. Only you (personal space) or members of your organization (organization space) can commit.

Variables and secrets (Display price: per hour • per month)

Variables Public

Variable	Type	Value
No variables		

Secrets Private

New variable New secret

Access Tokens

The screenshot shows the Hugging Face Spaces settings page for the space 'blip-image-captioning-api'. On the left, there's a section titled 'Space Hardware' with a sub-section 'Sleep time settings'. In the center, there are several hardware upgrade options listed in a grid:

Hardware Type	Model	Specs	Cost
CPU basic	2 vCPU - 16 GB RAM	Free	
	8 vCPU - 32 GB RAM	\$0.03/hour	Selected
Nvidia T4	Small (4 vCPU - 15 GB RAM - 16 GB VRAM)	\$0.60/hour	
	Medium (8 vCPU - 30 GB RAM - 16 GB VRAM)	\$0.90/hour	
Nvidia A10G	Small (4 vCPU - 15 GB RAM - 24 GB VRAM)	\$1.05/hour	
	Large (12 vCPU - 142 GB RAM - 40 GB VRAM)	\$4.13/hour	
Nvidia 2xA10G	Large (24 vCPU - 92 GB RAM - 48 GB VRAM)	\$5.70/hour	
	Large (48 vCPU - 184 GB RAM - 96 GB VRAM)	\$10.80/hour	
Nvidia 4xA10G	Large (48 vCPU - 184 GB RAM - 96 GB VRAM)	\$10.80/hour	
	H100 (24 vCPU - 250 GB RAM - 80 GB VRAM)	\$8.70/hour	

On the right side, there's a sidebar with user profile information ('Profile: aarohbits') and a list of actions: 'New Model', 'New Dataset', 'New Space', 'New Collection', 'Create organization', 'Settings' (which is selected and highlighted with a red box), and 'Sign Out'.

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

The screenshot shows a modal dialog titled 'Create a new access token'. It has fields for 'Name' (containing 'blip-captioning-app'), 'Role' (containing 'read'), and a large button labeled 'Generate a token'.

- You can copy the access token.

The screenshot shows the Hugging Face user settings interface. On the left, there's a sidebar with options like Profile, Account, Authentication, Organizations, Billing, and several sections under 'Access Tokens' (SSH and GPG Keys, Webhooks, Papers, Notifications, Content Preferences). The 'Access Tokens' section is highlighted with a red box. On the right, the main content area is also titled 'Access Tokens' and contains a sub-section 'User Access Tokens'. It explains what access tokens are and how they work. Below this, there's a table showing a single token entry: 'blip-captioning-app READ' with a 'Manage' dropdown and a 'Show' button. This table is also highlighted with a red box. At the bottom of the main content area, there's a 'New token' button.

```
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.