University of Southampton

Large Language Models (LLMs): Democratise AI with Low-Code/No-Code for Text, Images, and Audio

Content generation and content analysis in 3 lines of python code

A hands-on 101 tutorial
by students at
the University of Southampton

James Pickford (Year 3 student)
Yifan Chang (Year 2 student)
Aswathy Ajaykumar (Year 2 student)

08 May 2024

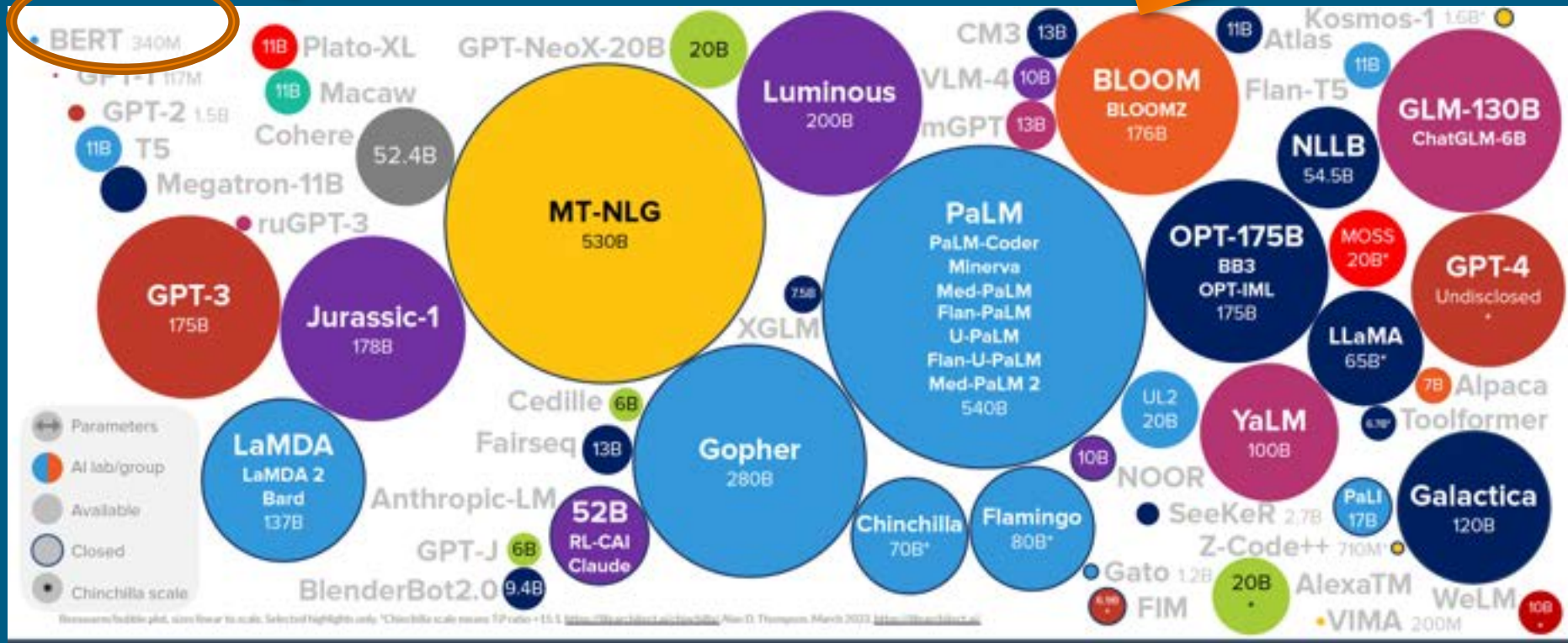
Large Language Models (LLMs)

University of Southampton

Number of Parameters: the smaller models like BERT have **M**illions of parameters

BERT with 340 **M**illion Parameters

BLOOM with 176**B**illion Parameters



LifeArchitect.ai/models

Pipelines to use LLMs from Google, Microsoft, Facebook, OpenAI, …

What are the **pipelines**?

Hugging Face pipelines

Large Language Models (LLMs)

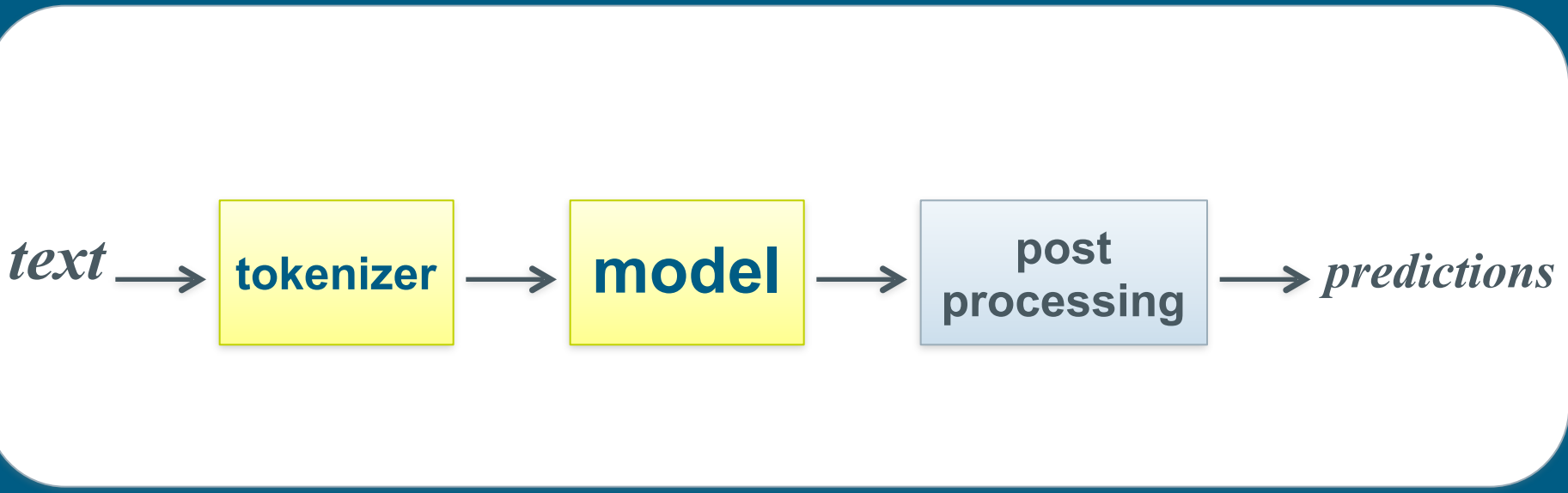
Pipelines to use LLMs from Google, Microsoft, Facebook, OpenAI, …

What are the **pipelines**?

The **pipelines** are **wrappers**, **i.e. python functions or classes**

There are **task-specific** pipelines for **audio**, **computer vision**, **natural language processing (NLP)**, and **multimodal tasks** (e.g. text-to-video)

Pipelines for Natural Language Processing (NLP)



Using LLMs for various tasks

overview

the 3 lines of python code

instantiate the pipeline

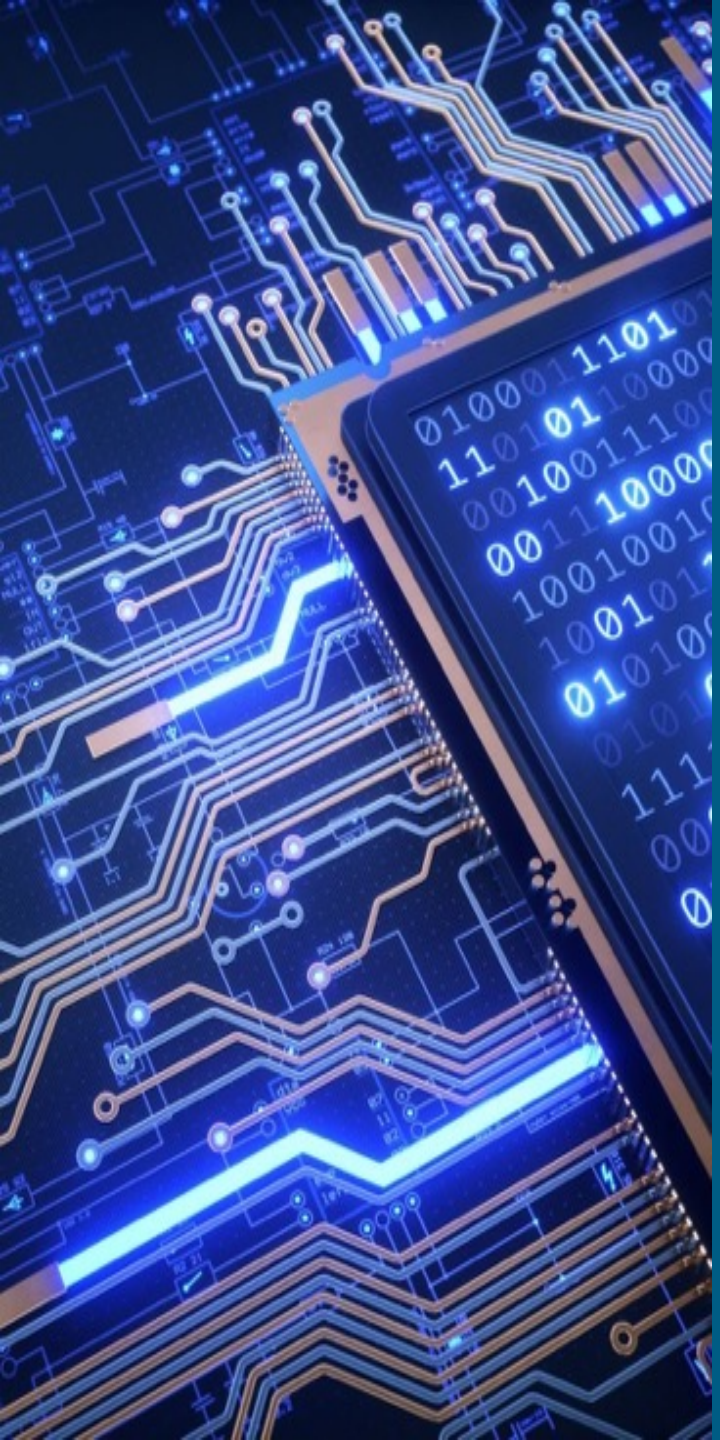
What is the task? What is the model?

the input data for the pipeline

a data sample or data instance

the output of the pipeline with the data given

the result of executing the pipeline



University of Southampton

Text Classification: Sentiment Analysis (SA)

Hugging Face pipeline

example

the 3 lines of python code

instantiate the pipeline

What is the task? What is the model?

```
pipe = pipeline(task="sentiment-analysis",
                model='siebert/sentiment-roberta-large-english')
```

the input data for the pipeline

a data sample or data instance

```
### a vector (one-dimensional array) with 2 string values
text_data = ["I love you", "I hate you"]
```

University of Southampton

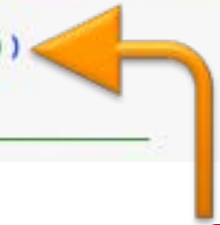
Hugging Face pipeline

example

the 3 lines of python code

Google Colab

```
# _____________________________
###  Text Classification: Sentiment Analysis (SA)
# _____________________________

#--- --- --- install and load libraries
!pip install transformers
from transformers import pipeline

#--- --- --- Transformers pipeline: 3 lines of code

pipe = pipeline(task="sentiment-analysis", model='siebert/sentiment-roberta-large-english')

text_data = ["I love you", "I hate you"] ### a vector (one-dimensional array) with 2 string values

print(pipe(text_data))

# _____________________________
```



the output of the pipeline with the data given

the result of executing the pipeline

University of Southampton

Hugging Face pipeline

running the code

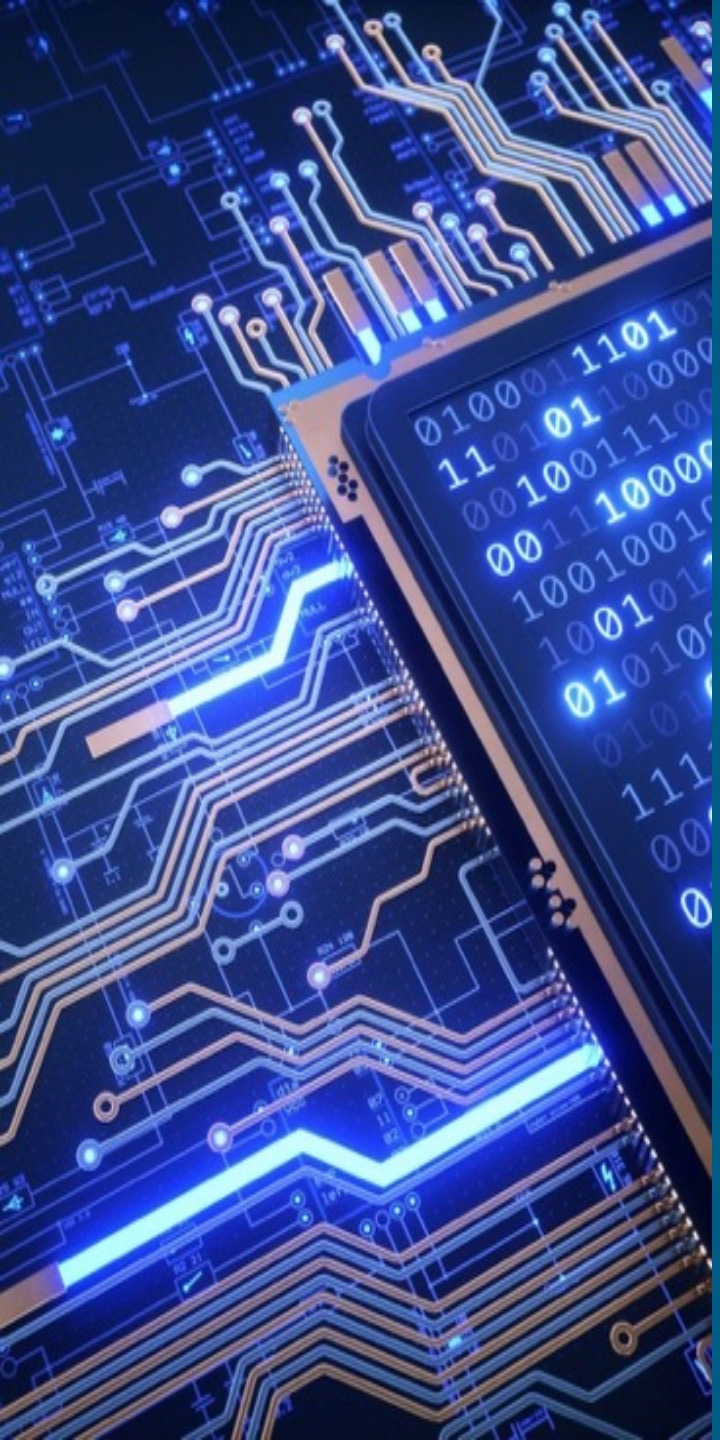
the 3 lines of python code

Google Colab

```
# _____________________________
###  Text Classification: Sentiment Analysis (SA)
# _____________________________

#--- --- --- install and load libraries
!pip install transformers
from transformers import pipeline

#--- --- --- Transformers pipeline: 3 lines of code

pipe = pipeline(task="sentiment-analysis", model='siebert/sentiment-roberta-large-english')

text_data = ["I love you", "I hate you"] ### a vector (one-dimensional array) with 2 string values

print(pipe(text_data))

# _____________________________
```

the output of the pipeline with the data given

[{'label': 'POSITIVE', 'score': 0.998561680316925},
{'label': 'NEGATIVE', 'score': 0.9991401433944702}]



University of Southampton

Token Classification: Named-Entity Recognition (NER)

University of Southampton

Hugging Face pipeline

example

the 3 lines of python code

instantiate the pipeline

What is the task? What is the model?

```
model_path = "d4data/biomedical-ner-all”
pipe = pipeline(task="ner",
          model=AutoModelForTokenClassification.from_pretrained(model_path),
          tokenizer= AutoTokenizer.from_pretrained(model_path))
```

the input data for the pipeline

a data sample or data instance

```
text_data =  "I have a mild headache"
```

University of Southampton

Hugging Face pipeline

example

the 3 lines of python code

Google Colab

```
# _____________________________
### Token Classification: Named-entity recognition (NER)
# _____________________________

#--- --- --- install and load libraries
!pip install transformers
from transformers import pipeline, AutoTokenizer, AutoModelForTokenClassification

#--- --- --- preliminaries
model_path = "d4data/biomedical-ner-all"

#--- --- --- Transformers pipeline: 3 lines of code

pipe = pipeline(task="ner", model= AutoModelForTokenClassification.from_pretrained(model_path),
                tokenizer= AutoTokenizer.from_pretrained(model_path))

text_data = "I have a mild headache"

print(pipe(text_data))
# _____________________________
```

the output of the pipeline with the data given

the result of executing the pipeline

University of Southampton

Hugging Face pipeline

running the code

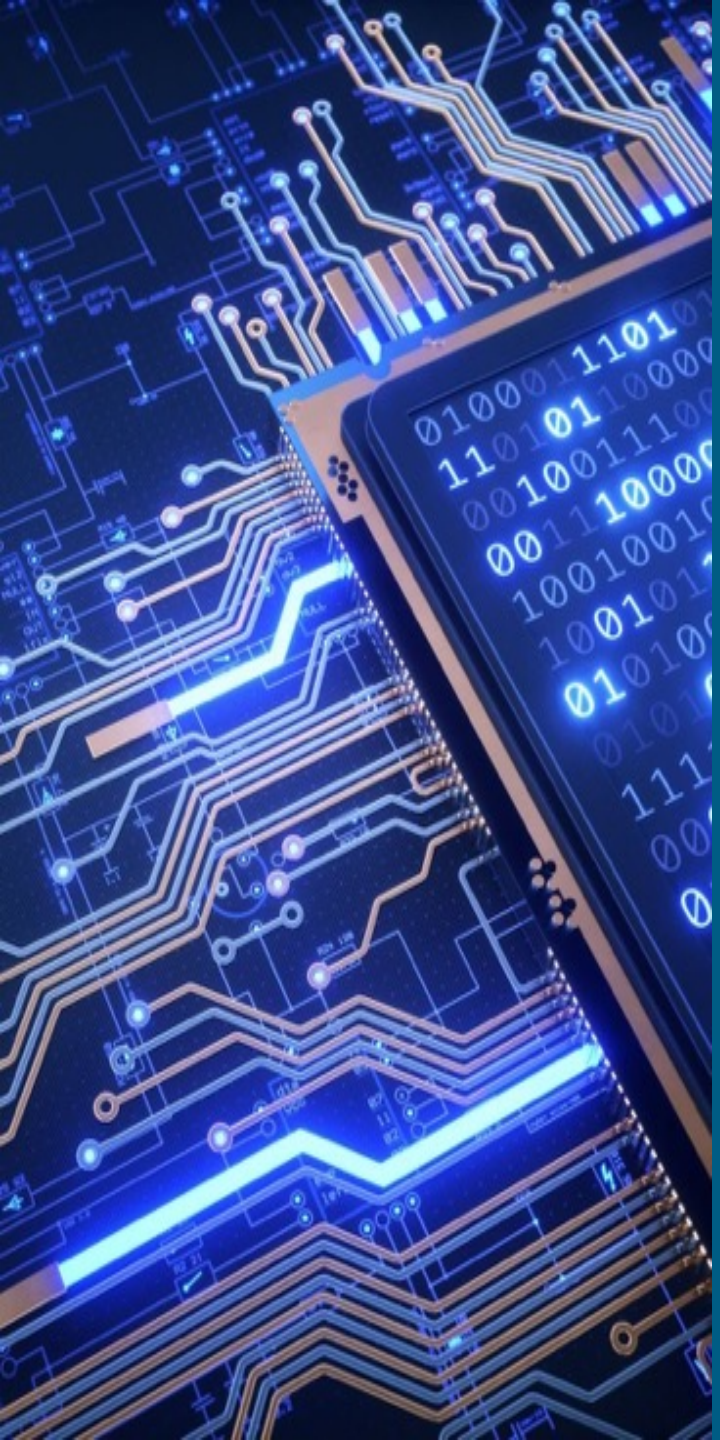
the 3 lines of python code

Google Colab

```
# _____________________________
### Token Classification: Named-entity recognition (NER)
# _____________________________

#--- --- --- install and load libraries
!pip install transformers
from transformers import pipeline, AutoTokenizer, AutoModelForTokenClassification

#--- --- --- preliminaries
model_path = "d4data/biomedical-ner-all"

#--- --- --- Transformers pipeline: 3 lines of code

pipe = pipeline(task="ner", model= AutoModelForTokenClassification.from_pretrained(model_path),
                tokenizer= AutoTokenizer.from_pretrained(model_path))

text_data = "I have a mild headache"

print(pipe(text_data))
# _____________________________
```

the output of the pipeline with the data given

[{'entity': '**B-Severity**', 'score': 0.9997888, 'index': 4, 'word': '**mild**', 'start': 9, 'end': 13},
{'entity': '**B-Sign_symptom**', 'score': 0.99993956, 'index': 5, 'word': '**headache**', 'start': 14, 'end': 22}]



University of Southampton

Text Classification: SA

Token Classification: NER

No-Code AI using the Hugging Face pipelines (**changing the prompt Not the code**)

Hugging Face pipeline

example

the 3 lines of python code

instantiate the pipeline

What is the task? What is the model?

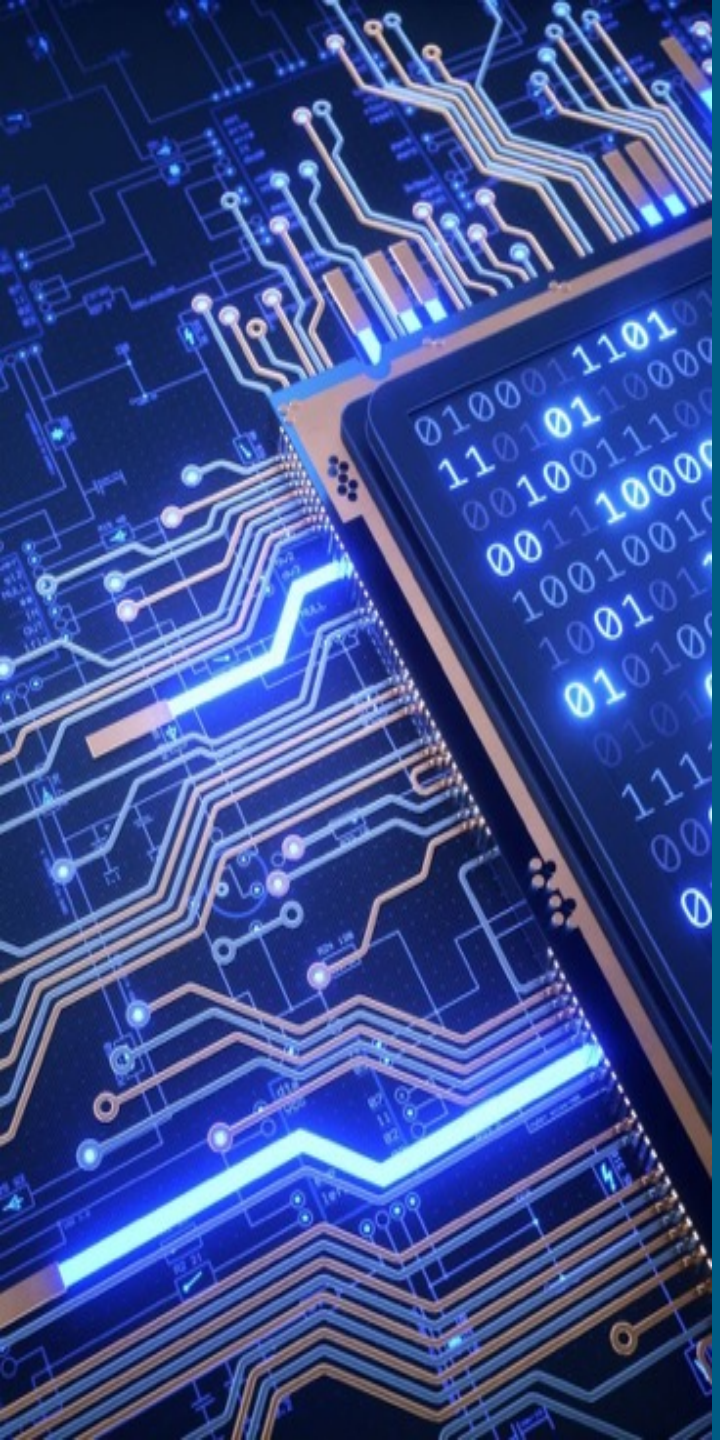
```
pipe = pipeline(task="text-generation",
                model='mistralai/Mistral-7B-Instruct-v0.2')
```

the input data for the pipeline

a data sample or data instance

text_prompt = "Classify the text into neutral, negative or positive.
Text: **I hate you**. Sentiment:" SA

text_prompt = "Return a list with the medical entities in the text.
Text: **I have a mild headache**. Named entities:" NER



University of Southampton

Zero shot (open vocabulary) image classification

University of Southampton

Hugging Face pipeline

example

the 3 lines of python code

instantiate the pipeline

What is the task? What is the model?

```
pipe = pipeline(task="zero-shot-image-classification",
                model="openai/clip-vit-large-patch14")
```

the input data for the pipeline

a data sample or data instance

```
img_data = Image.open(requests.get("https://www.metmuseum.org/-/media/images/visit/met-fifth-avenue/fifthave_teaser.jpg", stream=True).raw)
```

University of Southampton

Reusing existing images

example

the data

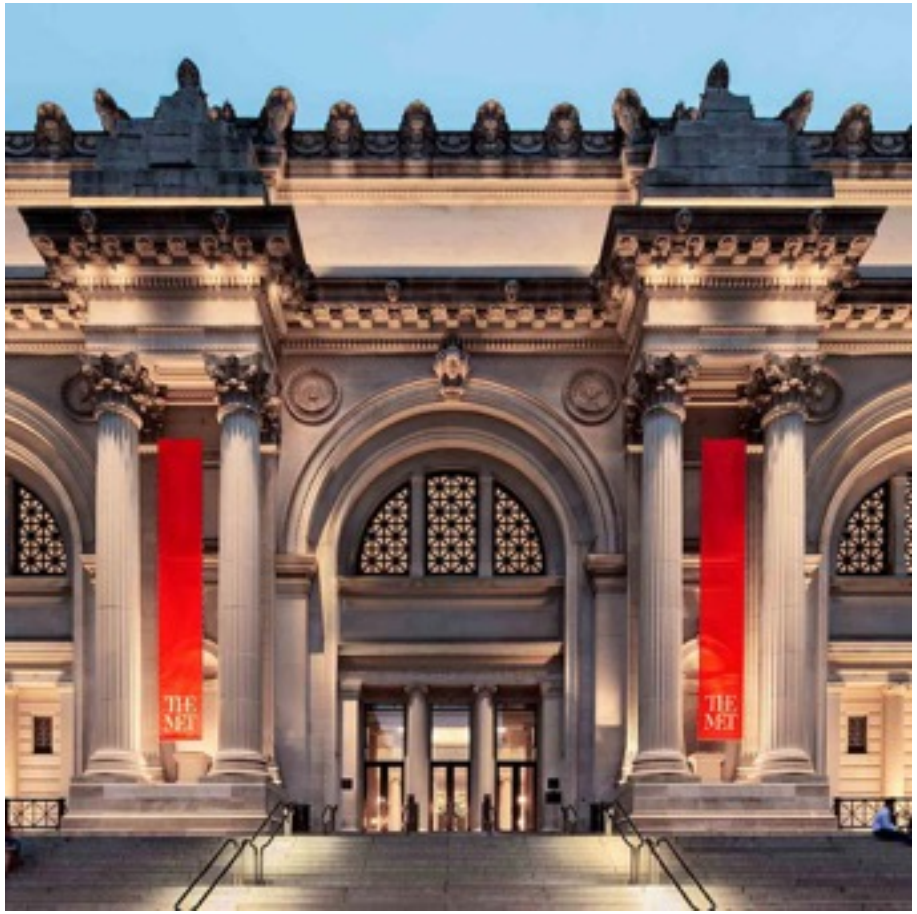


image URL

https://www.metmuseum.org/-/media/images/visit/met-fifth-avenue/fifthave_teaser.jpg

Hugging Face pipeline

example

the 3 lines of python code

Google Colab

```
# ____________________________
### zero-shot-image-classification
# ____________________________

#--- --- --- install and load libraries
!pip install transformers
!pip install pillow

import requests
from transformers import pipeline
from PIL import Image

#--- --- --- Transformers pipeline: 3 lines of code

pipe = pipeline(task="zero-shot-image-classification", model="openai/clip-vit-large-patch14")

img_data = Image.open(requests.get("https://www.metmuseum.org/-/media/images/visit/met-fifth-avenue/fifthave_teaser.jpg", stream=True).raw)

print(pipe(img_data, candidate_labels=["building", "column", "car", "animal", "cat", "flowers"]))

# ____________________________
```

the output of the pipeline
with the data given

the result of executing the pipeline

University of Southampton

Hugging Face pipeline

running the code

the 3 lines of python code

Google Colab

```
# ____________________________
### zero-shot-image-classification
# ____________________________

#--- --- --- install and load libraries
!pip install transformers
!pip install pillow

import requests
from transformers import pipeline
from PIL import Image

#--- --- --- Transformers pipeline: 3 lines of code

pipe = pipeline(task="zero-shot-image-classification", model="openai/clip-vit-large-patch14")

img_data = Image.open(requests.get("https://www.metmuseum.org/-/media/images/visit/met-fifth-avenue/fifthave_teaser.jpg", stream=True).raw)

print(pipe(img_data, candidate_labels=["building", "column", "car", "animal", "cat", "flowers"]))

# ____________________________
```

the output of the pipeline with the data given

[{'score': **0.7278850078582764**, 'label': '**building**'}, {'score': **0.259612143039703337**, 'label': '**column**'}, {'score': 0.004940211772918701, 'label': 'car'}, {'score': 0.0047919861972332, 'label': 'animal'}, {'score': 0.0025786142796278, 'label': 'cat'}, {'score': 0.00019203458214178681, 'label': 'flowers'}]

Multimodal Large Language Models

State-of-the-art of prompting

Can a textual prompt generate a video?

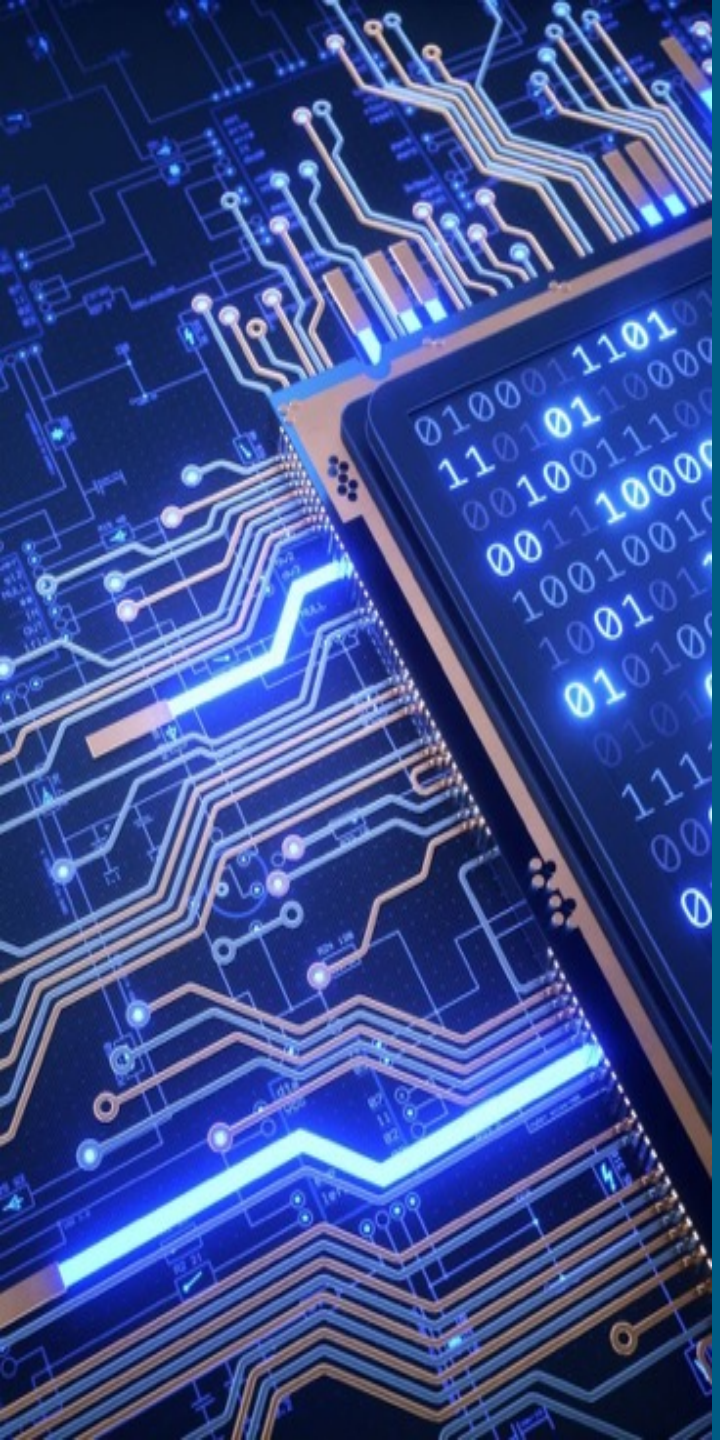
Can this be done with Hugging Face pipelines in 3 lines of code?

Multimodal Large Language Models

State-of-the-art of prompting

Can a textual prompt generate a video? **Yes**

Can this be done with Hugging Face pipelines in 3 lines of code? **Yes**



University of Southampton

text to video with multimodal LLMs

Task-specific pipeline

example

the 3 lines of python code

instantiate the pipeline

What is the task? What is the model?

```
pipe = DiffusionPipeline.from_pretrained("damo-vilab/text-to-video-ms-1.7b",
                                         torch_dtype=torch.float16, variant="fp16").to("cuda")
```

the input data for the pipeline

a data sample or data instance

```
text_prompt = "Darth Vader surfing a wave"
```

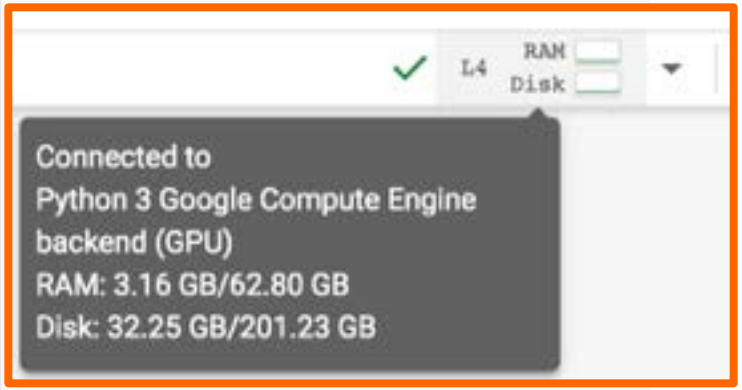
Task-specific pipeline

example

the 3 lines of python code

```
# ____________________________
### text to video
# ____________________________
#--- --- --- install and load libraries
!pip install transformers accelerate diffusers

import torch
from diffusers import DiffusionPipeline
from diffusers.utils import export_to_video

#--- --- --- preliminaries
###--- access to Google drive
from google.colab import drive
drive.mount('/content/drive')
###--- path to save the video generated
v_path = "/content/drive/MyDrive/newTut101_files/DarthVader_ex.mp4"


#--- --- --- Transformers pipeline: 3 lines of code

pipe = DiffusionPipeline.from_pretrained("damo-vilab/text-to-video-ms-1.7b", torch_dtype=torch.float16, variant="fp16").to("cuda")

text_prompt = "Darth Vader surfing a wave"

video_frames = pipe(text_prompt, num_frames=32).frames[0]

###--- extra line: save the video
video_path = export_to_video(video_frames, fps=10, output_video_path=v_path)

# ____________________________
```

Google Colab



the output of the pipeline with the data given

the result of executing the pipeline

Task-specific pipeline

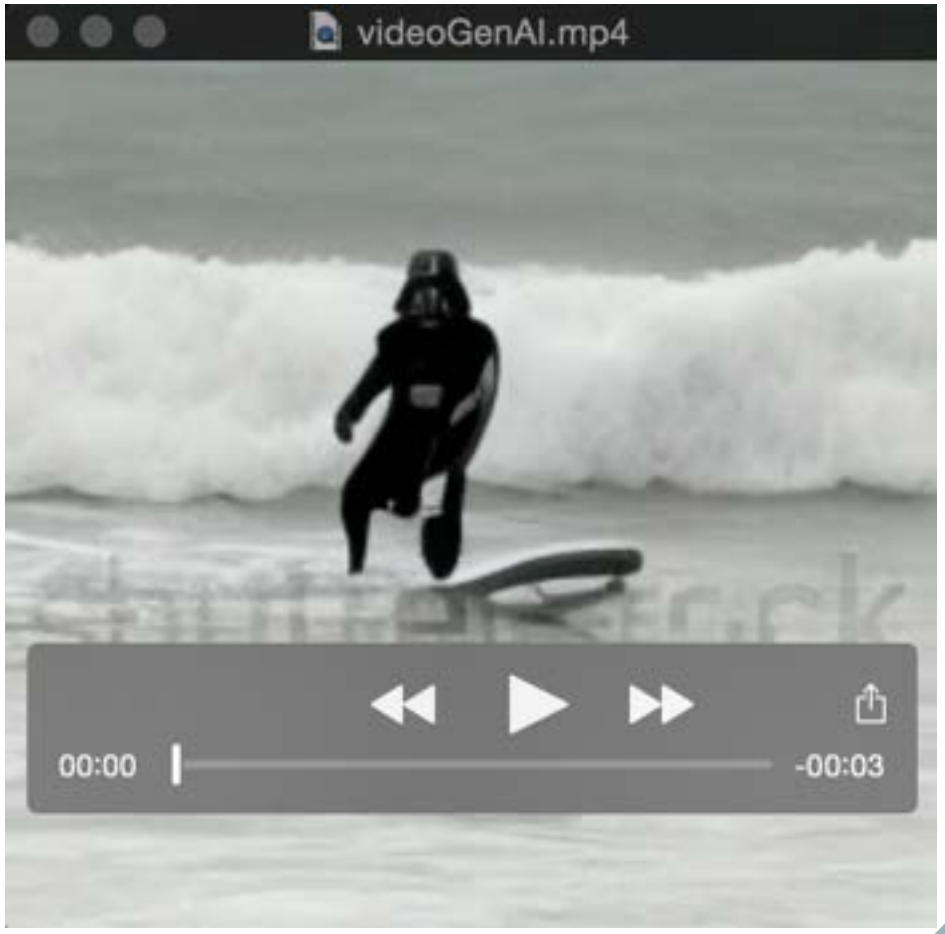
running the code

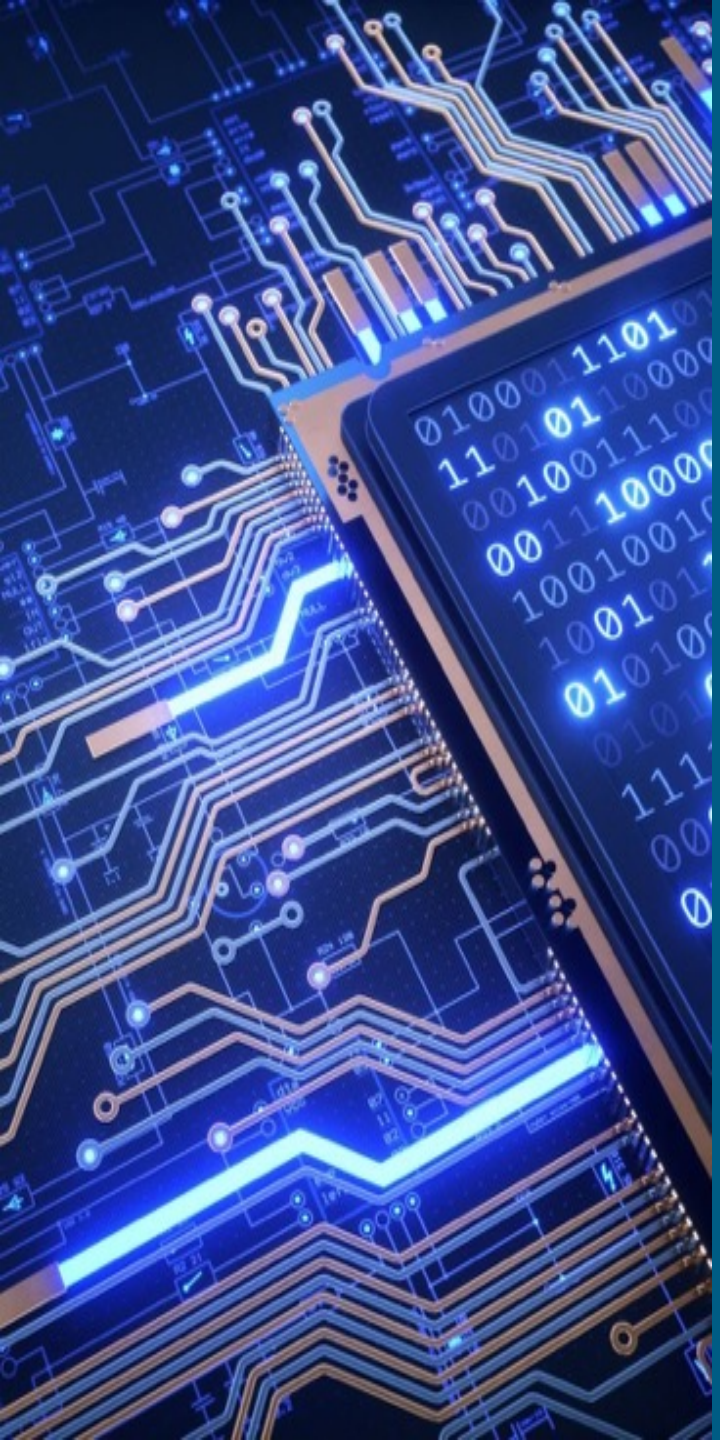
the 3 lines of python code

the input data for the pipeline

```
text_prompt = "Darth Vader surfing a wave"
```

the output of the pipeline with the data given





University of Southampton

Audio classification: Language Identification

University of Southampton

Hugging Face pipeline

example

the 3 lines of python code

instantiate the pipeline

What is the task? What is the model?

```
pipe = pipeline(task="audio-classification",
                model="facebook/mms-lid-4017")
```

the input data for the pipeline

a data sample or data instance

```
audio_data = audio_sample["audio"]["array"]
```

University of Southampton

Reusing existing datasets

example

the data

```
# _____________________________
### Using existing datasets:
### loading and playing audio files
# _____________________________

#--- --- --- install and load libraries
!pip install datasets
from datasets import load_dataset, Audio

#--- --- --- load audio file
### first file from train dataset of Minds-14: e-banking speech dataset Intent Classification
minds14 = load_dataset("PolyAI/minds14", name="en-AU", split="train[:1]")
audio_files = minds14.cast_column("audio", Audio(sampling_rate=16_000))
audio_sample = audio_files[0]

#--- --- --- play audio file
from IPython.display import Audio
Audio(audio_sample["audio"]["array"], rate=audio_sample["audio"]["sampling_rate"])
```

audio file

I would like to pay my electricity bill using my card, can you please assist?

University of Southampton

Hugging Face pipeline

example

the 3 lines of python code

Google Colab

```
# _____________________________
###  Audio Classification: Language Identification (LID)
# _____________________________

#--- --- --- install and load libraries
!pip install transformers datasets
from transformers import pipeline
from datasets import load_dataset, Audio

#--- --- --- preliminaries
### first file from train dataset of Minds-14: e-banking speech dataset Intent Classification
minds14 = load_dataset("PolyAI/minds14", name="en-AU", split="train[:1]")
audio_files = minds14.cast_column("audio", Audio(sampling_rate=16_000))
audio_sample = audio_files[0]

#--- --- --- Transformers pipeline: 3 lines of code

pipe = pipeline(task="audio-classification", model="facebook/mms-lid-4017")

audio_data = audio_sample["audio"]["array"]

print(pipe(audio_data))
# _____________________________
```

the output of the pipeline with the data given

the result of executing the pipeline

University of Southampton

Hugging Face pipeline

running the code

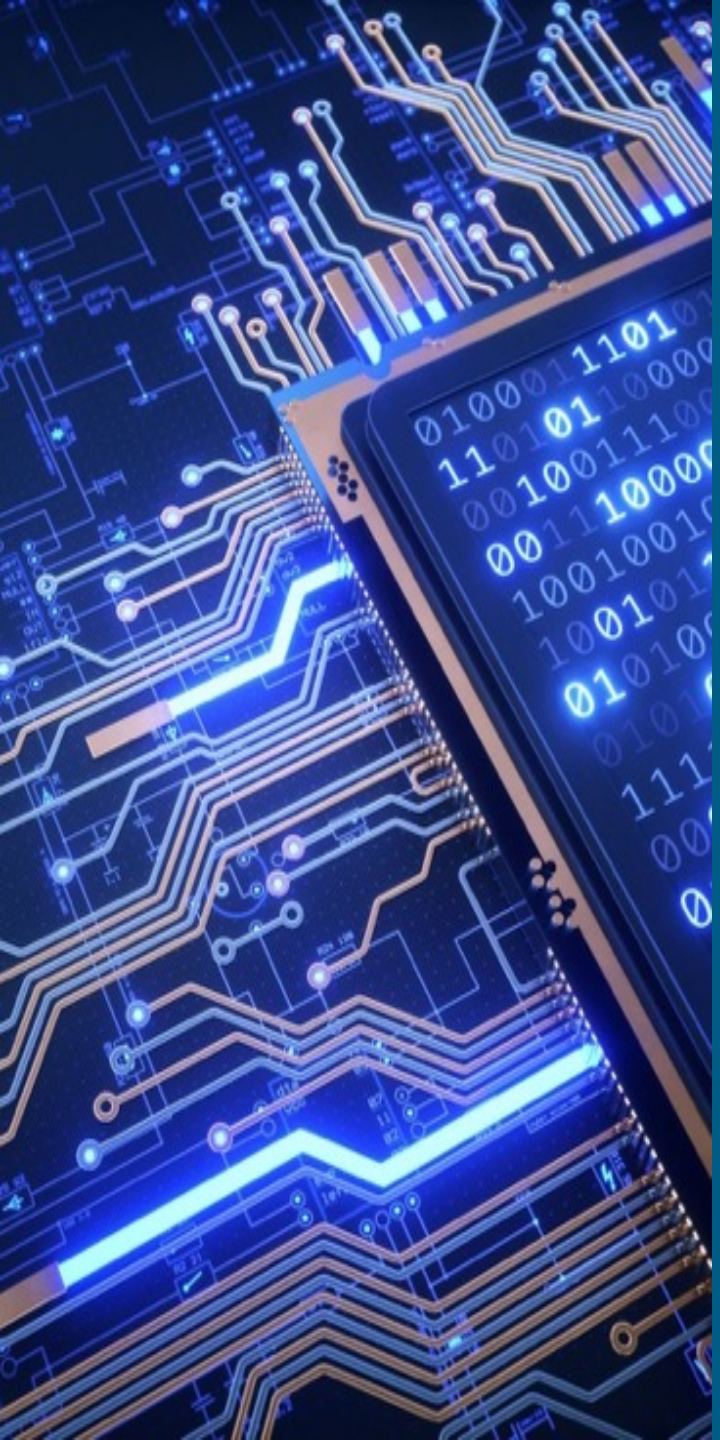
the 3 lines of python code

Google Colab

```
# ____________________________
###  Audio Classification: Language Identification (LID)
# ____________________________

#--- --- --- install and load libraries
!pip install transformers datasets
from transformers import pipeline
from datasets import load_dataset, Audio

#--- --- --- preliminaries
### first file from train dataset of Minds-14: e-banking speech dataset Intent Classification
minds14 = load_dataset("PolyAI/minds14", name="en-AU", split="train[:1]")
audio_files = minds14.cast_column("audio", Audio(sampling_rate=16_000))
audio_sample = audio_files[0]

#--- --- --- Transformers pipeline: 3 lines of code

pipe = pipeline(task="audio-classification", model="facebook/mms-lid-4017")

audio_data = audio_sample["audio"]["array"]

print(pipe(audio_data))
# ____________________________
```

the output of the pipeline with the data given

[{'score': **0.9155595302581787**, 'label': '**eng**'}, {'score': 0.030366547405719757, 'label': 'mri'}, {'score': 0.0034650196321308613, 'label': 'cym'}, {'score': 0.0018054329557344317, 'label': 'haw'}, {'score': 0.0013475015293806791, 'label': 'lat'}]



University of Southampton

Audio classification: Keyword Spotting

University of Southampton

Hugging Face pipeline

example

the 3 lines of python code

instantiate the pipeline

What is the task? What is the model?

```
pipe = pipeline(task="audio-classification",
                model="anton-l/xtreme_s_xlsr_300m_minds14")
```

the input data for the pipeline

a data sample or data instance

```
audio_data = audio_sample["audio"]["array"]
```

University of Southampton

Reusing existing datasets

example

the data

```
# _____________________________
### Using existing datasets:
### loading and playing audio files
# _____________________________

#--- --- --- install and load libraries
!pip install datasets
from datasets import load_dataset, Audio

#--- --- --- load audio file
### first file from train dataset of Minds-14: e-banking speech dataset Intent Classification
minds14 = load_dataset("PolyAI/minds14", name="en-AU", split="train[:1]")
audio_files = minds14.cast_column("audio", Audio(sampling_rate=16_000))
audio_sample = audio_files[0]

#--- --- --- play audio file
from IPython.display import Audio
Audio(audio_sample["audio"]["array"], rate=audio_sample["audio"]["sampling_rate"])
```

audio file

I would like to pay my electricity bill using my card, can you please assist?

University of Southampton

Hugging Face pipeline

example

the 3 lines of python code

Google Colab

```
# _____________________________
###  Audio Classification: Keyword Spotting (KWS)
# _____________________________

#--- --- --- install and load libraries
!pip install transformers datasets
from transformers import pipeline
from datasets import load_dataset, Audio

#--- --- --- preliminaries
### first file from train dataset of Minds-14: e-banking speech dataset Intent Classification
minds14 = load_dataset("PolyAI/minds14", name="en-AU", split="train[:1]")
audio_files = minds14.cast_column("audio", Audio(sampling_rate=16_000))
audio_sample = audio_files[0]

#--- --- --- Transformers pipeline: 3 lines of code

pipe = pipeline(task="audio-classification", model="anton-l/xtreme_s_xlsr_300m_minds14")

audio_data = audio_sample["audio"]["array"]

print(pipe(audio_data))

# _____________________________
```

the output of the pipeline with the data given

the result of executing the pipeline

University of Southampton

Hugging Face pipeline

running the code

the 3 lines of python code

Google Colab

```
# ____________________________
###  Audio Classification: Keyword Spotting (KWS)
# ____________________________

#--- --- --- install and load libraries
!pip install transformers datasets
from transformers import pipeline
from datasets import load_dataset, Audio

#--- --- --- preliminaries
### first file from train dataset of Minds-14: e-banking speech dataset Intent Classification
minds14 = load_dataset("PolyAI/minds14", name="en-AU", split="train[:1]")
audio_files = minds14.cast_column("audio", Audio(sampling_rate=16_000))
audio_sample = audio_files[0]

#--- --- --- Transformers pipeline: 3 lines of code

pipe = pipeline(task="audio-classification", model="anton-l/xtreme_s_xlsr_300m_minds14")

audio_data = audio_sample["audio"]["array"]

print(pipe(audio_data))

# ____________________________
```

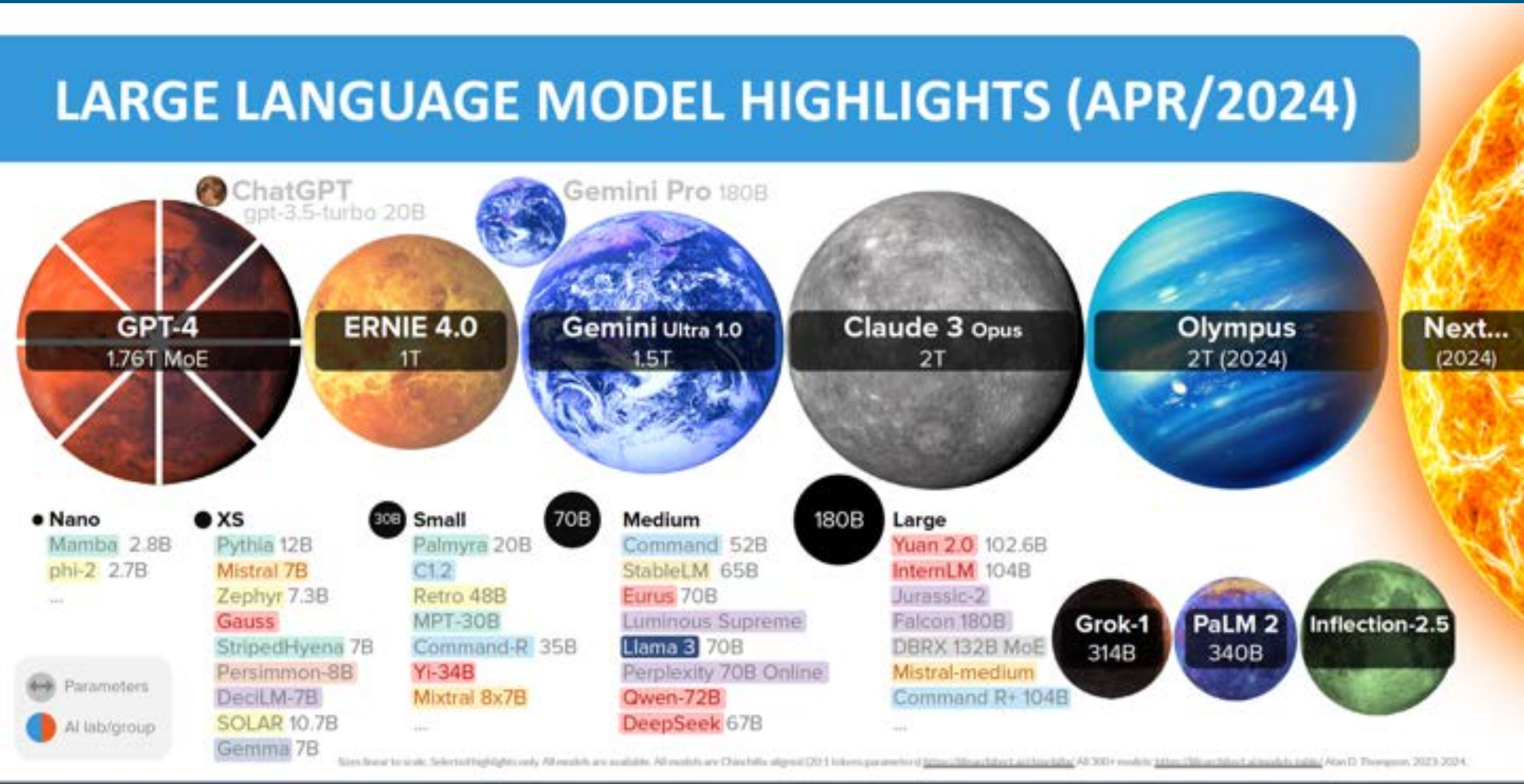
the output of the pipeline with the data given

[{'score': **0.9625307917594 91**, 'label': '**pay_bill**'}, {'score': 0.02867300808429718, 'label': 'freeze'}, {'score': 0.003349815495312214, 'label': 'card_issues'}, {'score': 0.0020058136433362 96, 'label': 'abroad'}, {'score': 0.0008484353311359882, 'label': 'high_value_payment'}]

University of Southampton

What's next?

Parameters (number of variables): from **M** (millions), to **B** (billions), to **T** (trillions)



Images shown may have a copyright

University of Southampton

Questions

Special thanks to:
Prof Max Bramer (Chair of BCS SGAI) and
Dhruv Khanna (President of AI Society at University of Southampton)

M Arguello-Casteleiro (University of Southampton) and T Furmston (University of Manchester) for the Hands-on 101 Tutorial Democratise AI with Low-Code/No-Code for Text, Images, and Audio