

# Intro to Hugging Face and Python

Dirk Wulff



**MAX PLANCK INSTITUTE**  
FOR HUMAN DEVELOPMENT



# OUR SOFTWARE STACK



+



+



# OUR SOFTWARE STACK



+



+



# WHY IS HUGGING FACE?



# WHY IS HUGGING FACE?



**Traditional language modelling pipeline:**

# WHY IS HUGGING FACE?



## **Traditional language modelling pipeline:**

1. Find out the model architecture

# WHY IS HUGGING FACE?

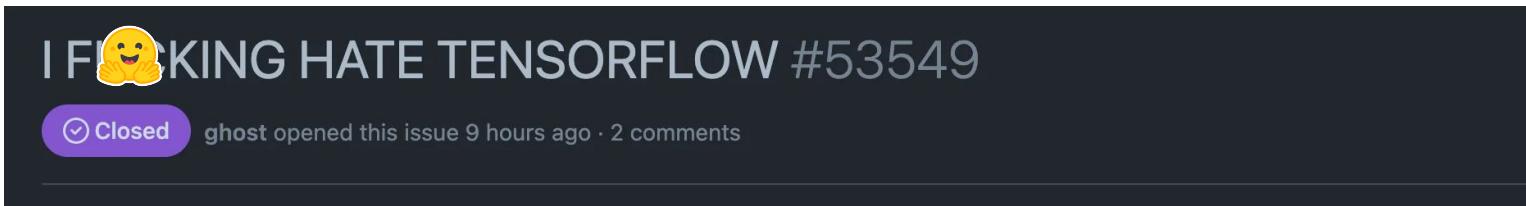


## **Traditional language modelling pipeline:**

1. Find out the model architecture
2. Implement the model architecture in code with deep learning frameworks (e.g PyTorch/Tensorflow).

# 1. DEEP LEARNING LIBRARIES CAN BE DIFFICULT

# 1. DEEP LEARNING LIBRARIES CAN BE DIFFICULT



# WHY HUGGING FACE?

## **Traditional language modelling pipeline:**

1. Find out the model architecture
2. Implement the model architecture in code with deep learning libraries (e.g PyTorch/Tensorflow).

# WHY HUGGING FACE?

## **Traditional language modelling pipeline:**

1. Find out the model architecture
2. Implement the model architecture in code with deep learning libraries (e.g PyTorch/Tensorflow).
3. Load the pretrained weights (if available) from a server.
4. Process the inputs (using the correct tokenizer for the model)
5. Implement data loaders
6. Define a loss function
7. Stick a task-specific “head” on the model

# HUGGING FACE PIPELINES

Import pipeline

```
from transformers import pipeline
```

Initialise  
pipeline

```
pipe = pipeline('text-generation', model='gpt2')
```

Load model  
input

```
prompt = """  
Once upon a time in a land far far away, there was a young prince named  
John. He was known for his bravery and courage. One day, he decided to go on  
an adventure to explore the unknown lands.  
"""`
```

Feed input  
the model

```
output = pipe(prompt, max_length=100)  
print(output)
```

# HUGGING FACE

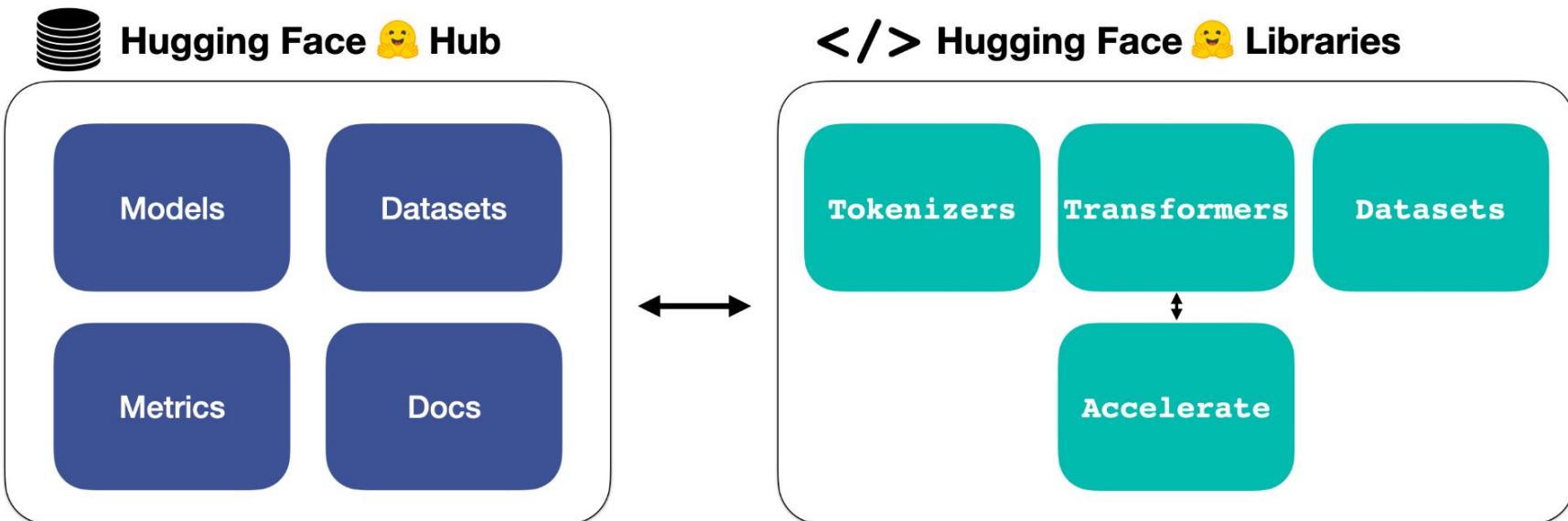
The screenshot shows the Hugging Face homepage. At the top, there's a search bar with placeholder text "Search models, datasets, users...". Below the search bar, a navigation menu includes "Models", "Datasets", "Spaces", "Community", "Docs", "Enterprise", and "Pricing". On the left side, there are several filters and lists:

- Tasks:** Text Generation, Any-to-Any, Image-Text-to-Text, Image-to-Text, Image-to-Image, Text-to-Image, Text-to-Video, Text-to-Speech (+42).
- Parameters:** A slider ranging from 1B to >500B, currently set at 1B.
- Libraries:** PyTorch, TensorFlow, JAX, Transformers, Diffusers, sentence-transformers, Safetensors, ONNX, GGU, Transformers.js, MLX (+41).
- Apps:** vLLM, TGI, llama.cpp, MLX LM, LM Studio, Ollama, Jan (+13).
- Inference Providers:** Groq, Novita, Nebulus AI, Cerebras, SambaNova, Nscale, fal, Hyperbolic (+11).

The main content area displays a grid of trending models. Each model entry includes the owner's profile picture, the model name, a brief description, and some statistics (e.g., size, last update, number of stars).

Model	Description	Last Update	Size	Stars
Tongyi-MAI/Z-Image-Turbo	Text-to-Image	Updated about 23 hours ago	~111k	1.91k
deepseek-ai/DeepSeek-V3.2	Text Generation	Updated 2 days ago	~5.01k	626
deepseek-ai/DeepSeek-V3.2-Speciale	Text Generation	Updated 2 days ago	~1.87k	439
black-forest-labs/FLUX.2-dev	Image-to-Image	Updated 6 days ago	~185k	850
Comfy-Org/z_image_turbo		Updated 6 days ago	~1.5M	339
tencent/HunyuanOCR	Image-Text-to-Text	Updated 1 day ago	~226k	614
microsoft/Fara-7B	Image-Text-to-Text	Updated 2 days ago	~25.3k	388
facebook/sam3	Mask Generation	Updated 13 days ago	~327k	858
alibaba-pai/Z-Image-Turbo-Fun-Controlnet-Union		Updated 1 day ago	~177	
apple/starflow		Updated 1 day ago	~172	
PrimeIntellect/INTELLECT-3	Text Generation	Updated 6 days ago	~2.75k	170
jayn7/Z-Image-Turbo-GGUF	Text-to-Image	Updated 5 days ago	~70k	154
AIDC-AI/Ovis-Image-7B	Text-to-Image	Updated about 8 hours ago	~1.26k	139
mistralai/Mistral-Large-3-675B-Instruct-2512		Updated about 19 hours ago	~112	113
stepfun-ai/Step-Audio-R1	Audio-Text-to-Text	Updated 1 day ago	~315	110
T5B/Z-Image-Turbo-FP8	Text-to-Image	Updated 6 days ago	~110k	91
salakash/SamKash-Tolstoy	Text Generation	Updated 6 days ago	~7.88k	469
mistralai/Mistral-7-140-TextInstruct-2E12		Updated 1 day ago	~240	87

# THE HUGGING FACE ECOSYSTEM



# HUGGING FACE DOCUMENTATION

## DOC Documentations

Search across all docs

### Hub

Host Git-based models, datasets and Spaces on the Hugging Face Hub.

### Hub Python Library

Client library for the HF Hub: manage repositories from your Python runtime.

### Inference API

Use more than 50k models through our public inference API, with scalability built-in.

### Transformers

State-of-the-art ML for Pytorch, TensorFlow, and JAX.

### Datasets

Access and share datasets for computer vision, audio, and NLP tasks.

### Huggingface.js

A collection of JS libraries to interact with Hugging Face, with TS types included.

### Inference Endpoints

Easily deploy your model to production on dedicated, fully managed infrastructure.

### Diffusers

State-of-the-art diffusion models for image and audio generation in PyTorch.

### Gradio

Build machine learning demos and other web apps, in just a few lines of Python.

### Transformers.js

Community library to run pretrained models from Transformers in your browser.

### PEFT

Parameter efficient finetuning methods for large models

# OUR SOFTWARE STACK



+



+





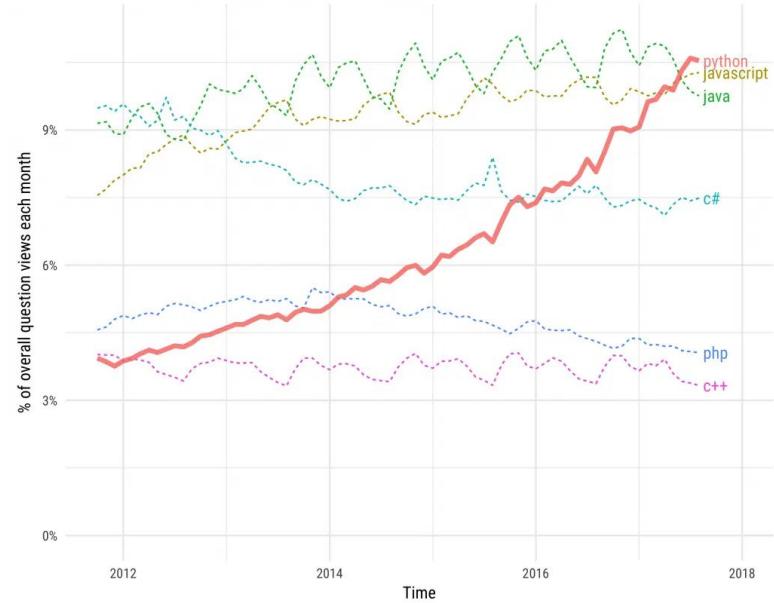
# PYTHON



- A **high-level** programming language known for its simplicity and readability.
- Used in various domains such as web development, data analysis, **artificial intelligence**, and scientific computing.

**Growth of major programming languages**

Based on Stack Overflow question views in World Bank high-income countries



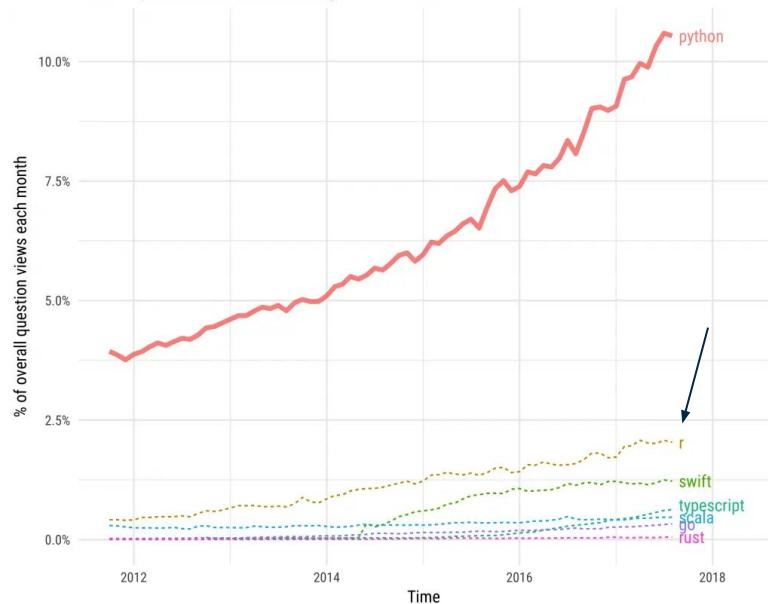
# PYTHON



- A **high-level** programming language known for its simplicity and readability.
- Used in various domains such as web development, data analysis, **artificial intelligence**, and scientific computing.

Python compared to smaller, growing technologies

Based on question traffic in World Bank high-income countries



# OUR SOFTWARE STACK



+

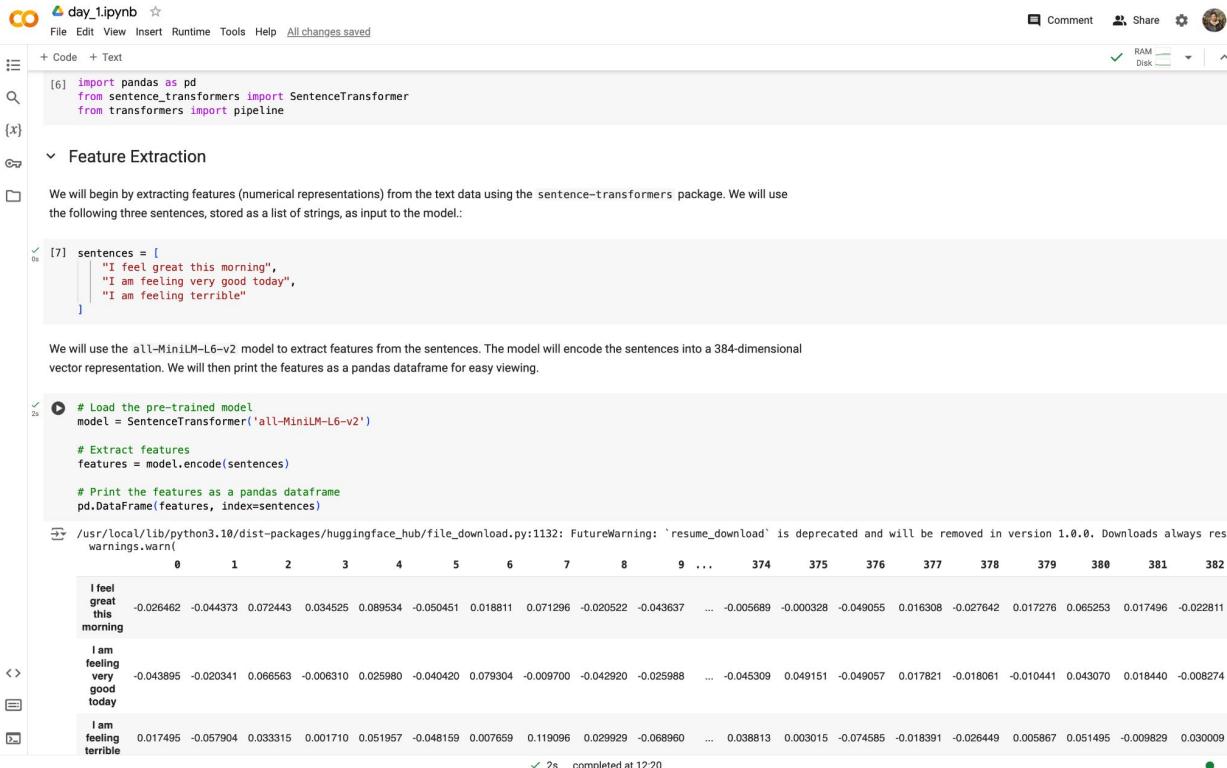


+



PYTHON  + Google Colab 

# PYTHON + Google Colab



The image shows a screenshot of the Google Colab interface. At the top, there's a navigation bar with File, Edit, View, Insert, Runtime, Tools, Help, and a message indicating "All changes saved". On the right side of the header are Comment, Share, and settings icons, along with RAM and Disk status indicators.

The main area is a code editor with a sidebar containing file and search functions. The code editor shows a Jupyter notebook cell with the following code:

```
[6] import pandas as pd
     from sentence_transformers import SentenceTransformer
     from transformers import pipeline
```

Below the code, a section titled "Feature Extraction" is expanded, containing the following text:

We will begin by extracting features (numerical representations) from the text data using the `sentence-transformers` package. We will use the following three sentences, stored as a list of strings, as input to the model:

```
[7] sentences = [
    "I feel great this morning",
    "I am feeling very good today",
    "I am feeling terrible"
]
```

Below the code cell, a note states:

We will use the `all-MiniLM-L6-v2` model to extract features from the sentences. The model will encode the sentences into a 384-dimensional vector representation. We will then print the features as a pandas dataframe for easy viewing.

```
[8] # Load the pre-trained model
model = SentenceTransformer('all-MiniLM-L6-v2')

# Extract features
features = model.encode(sentences)

# Print the features as a pandas dataframe
pd.DataFrame(features, index=sentences)
```

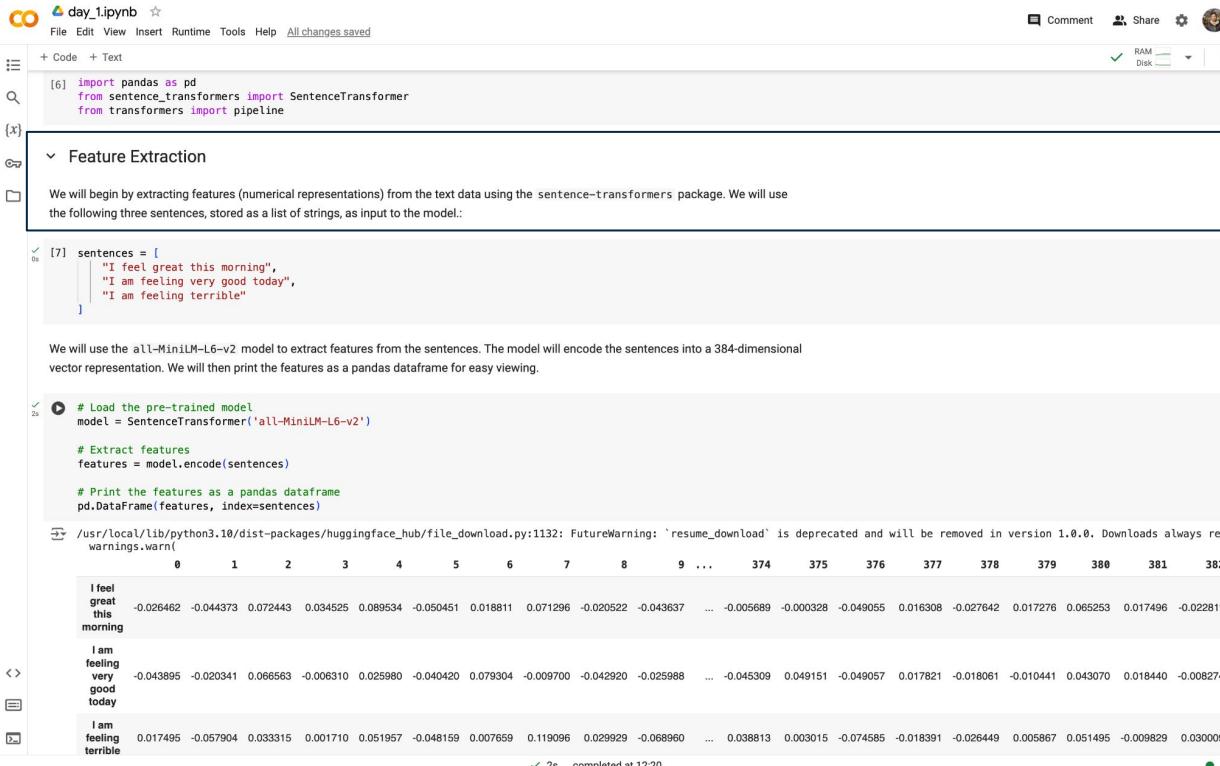
The output of the code cell is a pandas DataFrame with 3 rows and 382 columns. The first few columns are labeled 0 through 8, and the last few columns are labeled 374 through 382. The rows are labeled with the sentences: "I feel great this morning", "I am feeling very good today", and "I am feeling terrible". The data values are floating-point numbers representing the 384-dimensional vector features.

At the bottom of the code cell, there's a warning message:

```
/usr/local/lib/python3.10/dist-packages/huggingface_hub/file_download.py:1132: FutureWarning: 'resume_download' is deprecated and will be removed in version 1.0.0. Downloads always resume.
```

At the very bottom of the interface, there's a footer with a green checkmark icon, the text "2s completed at 12:20", and a close button.

# PYTHON + Google Colab



The screenshot shows a Google Colab notebook titled "day\_1.ipynb". The code cell at the top imports pandas and SentenceTransformer from the sentence-transformers package. A section titled "Feature Extraction" is highlighted with a blue border. Inside this section, the code defines a list of sentences: "I feel great this morning", "I am feeling very good today", and "I am feeling terrible". A note explains that these sentences will be used to extract features using the "all-MiniLM-L6-v2" model. The code then loads the pre-trained model and extracts features as a pandas DataFrame. The resulting DataFrame is displayed, showing numerical values for each sentence across 382 dimensions. The first few rows of the DataFrame are:

	0	1	2	3	4	5	6	7	8	9	...	374	375	376	377	378	379	380	381	382
I feel great this morning	-0.026462	-0.044373	0.072443	0.034525	0.089534	-0.050451	0.018811	0.071296	-0.020522	-0.043637	...	-0.005689	-0.000328	-0.049055	0.016308	-0.027642	0.017276	0.065253	0.017496	-0.022811
I am feeling very good today	-0.043895	-0.020341	0.066563	-0.006310	0.025980	-0.040420	0.079304	-0.009700	-0.042920	-0.025988	...	-0.045309	0.049151	-0.049057	0.017821	-0.018061	-0.010441	0.043070	0.018440	-0.008274
I am feeling terrible	0.017495	-0.057904	0.033315	0.001710	0.051957	-0.048159	0.007659	0.119096	0.029929	-0.068960	...	0.038813	0.003015	-0.074585	-0.018391	-0.026449	0.005867	0.051495	-0.009829	0.030009

At the bottom, a status bar indicates the cell completed at 12:20.

## 1. Markdown

# PYTHON + Google Colab

The screenshot shows a Google Colab notebook titled "day\_1.ipynb". The code cell at the top imports pandas and sentence-transformers, and defines a list of sentences. A section titled "Feature Extraction" is expanded, containing a note about extracting features from text data using the sentence-transformers package. Below this, another code cell loads a pre-trained model and encodes the sentences into a pandas DataFrame. The resulting DataFrame is displayed, showing numerical features for each sentence.

```
[6] import pandas as pd
     from sentence_transformers import SentenceTransformer
     from transformers import pipeline

{x}
  ✓ Feature Extraction

  We will begin by extracting features (numerical representations) from the text data using the sentence-transformers package. We will use the following three sentences, stored as a list of strings, as input to the model:

[7] sentences = [
    "I feel great this morning",
    "I am feeling very good today",
    "I am feeling terrible"
]

We will use the all-MiniLM-L6-v2 model to extract features from the sentences. The model will encode the sentences into a 384-dimensional vector representation. We will then print the features as a pandas dataframe for easy viewing.

[25] # Load the pre-trained model
model = SentenceTransformer('all-MiniLM-L6-v2')

# Extract features
features = model.encode(sentences)

# Print the features as a pandas dataframe
pd.DataFrame(features, index=sentences)
```

	0	1	2	3	4	5	6	7	8	9	...	374	375	376	377	378	379	380	381	382
I feel great this morning	-0.026462	-0.044373	0.072443	0.034525	0.089534	-0.050451	0.018811	0.071296	-0.020522	-0.043637	...	-0.005689	-0.000328	-0.049055	0.016308	-0.027642	0.017276	0.065253	0.017496	-0.022811
I am feeling very good today	-0.043895	-0.020341	0.066563	-0.006310	0.025980	-0.040420	0.079304	-0.009700	-0.042920	-0.025988	...	-0.045309	0.049151	-0.049057	0.017821	-0.018061	-0.010441	0.043070	0.018440	-0.008274
I am feeling terrible	0.017495	-0.057904	0.033315	0.001710	0.051957	-0.048159	0.007659	0.119096	0.029929	-0.068960	...	0.038813	0.003015	-0.074585	-0.018391	-0.026449	0.005867	0.051495	-0.009829	0.030009

## 1. Markdown

## 2. Code

# PYTHON + Google Colab

The screenshot shows a Google Colab notebook titled "day\_1.ipynb". The interface includes a menu bar (File, Edit, View, Insert, Runtime, Tools, Help), a toolbar with icons for Comment, Share, and settings, and a sidebar with a file tree.

**1. Markdown**

We will begin by extracting features (numerical representations) from the text data using the `sentence-transformers` package. We will use the following three sentences, stored as a list of strings, as input to the model:

```
[7] sentences = [
    "I feel great this morning",
    "I am feeling very good today",
    "I am feeling terrible"
]
```

We will use the `all-MiniLM-L6-v2` model to extract features from the sentences. The model will encode the sentences into a 384-dimensional vector representation. We will then print the features as a pandas dataframe for easy viewing.

```
[25] # Load the pre-trained model
model = SentenceTransformer('all-MiniLM-L6-v2')

# Extract features
features = model.encode(sentences)

# Print the features as a pandas dataframe
pd.DataFrame(features, index=sentences)
```

**2. Code**

**3. Printouts**

	0	1	2	3	4	5	6	7	8	9	...	374	375	376	377	378	379	380	381	382
I feel great this morning	-0.026462	-0.044373	0.072443	0.034525	0.089534	-0.050451	0.018811	0.071296	-0.020522	-0.043637	...	-0.005689	-0.000328	-0.049055	0.016308	-0.027642	0.017276	0.065253	0.017496	-0.022811
I am feeling very good today	-0.043895	-0.020341	0.066563	-0.006310	0.025980	-0.040420	0.079304	-0.009700	-0.042920	-0.025988	...	-0.045309	0.049151	-0.049057	0.017821	-0.018061	-0.010441	0.043070	0.018440	-0.008274
I am feeling terrible	0.017495	-0.057904	0.033315	0.001710	0.051957	-0.048159	0.007659	0.119096	0.029929	-0.068960	...	0.038813	0.003015	-0.074585	-0.018391	-0.026449	0.005867	0.051495	-0.009829	0.030009

# PYTHON + Google Colab



```
# Day 1: Python + Google Colab

File Edit View Insert Runtime Tools Help All changes saved
+ Code + Test
(1) import pandas as pd
from sentence_transformers import SentenceTransformer
from transformers import pipeline
(x)
Feature Extraction
We will begin by extracting features (numerical representations) from the text data using the sentence-transformers package. We will use the following three sentences, stored as a list of strings, as input to the model.
(2) sentences = ["I am great this morning",
               "I am feeling very good today",
               "I am feeling terrible"]
We will use the all-MiniLM-L6-v2 model to extract features from the sentences. The model will encode the sentences into a 384-dimensional vector representation. We will then print the features as a pandas dataframe for easy viewing.
# Load the pre-trained model.
model = SentenceTransformer('all-MiniLM-L6-v2')

# Extract features
features = model.encode(sentences)

# Print the features as a pandas dataframe
pd.DataFrame(features, index=sentences)
/usr/local/lib/python3.10/dist-packages/huggingface_hub/file_download.py:1132: FutureWarning: 'resume_download' is deprecated and will be removed in version 1.8.0. Downloads always resume.
warnings.warn(
    0   1   2   3   4   5   6   7   8   9   ...   374   375   376   377   378   379   380   381   382
I am great this morning  -0.044602  -0.044701  0.072483  0.034625  0.030564  -0.005461  0.019811  0.012096  -0.020622  -0.044807  ...  -0.002689  -0.003298  -0.040055  0.016326  -0.027462  0.012276  0.063253  0.074696  -0.028817
I am feeling very good today  -0.043865  -0.020341  0.085583  -0.005310  0.023590  -0.040400  0.079304  -0.009700  -0.042920  -0.025088  ...  -0.040309  0.049151  -0.049057  0.017821  -0.019081  -0.010441  0.043270  0.018440  -0.008274
I am feeling terrible  0.017495  -0.057904  0.033515  0.001710  0.051957  -0.048159  0.007669  0.119096  ...  0.039813  0.000015  -0.074586  -0.016891  0.009449  0.009867  0.051465  -0.009829  0.030009

```

# PYTHON + Google Colab

## package imports



```
# Code + Test
File Edit View Insert Runtime Help All changes saved
[1]: import pandas as pd
from sentence_transformers import SentenceTransformer
from transformers import pipeline
(x)
Feature Extraction
We will begin by extracting features (numerical representations) from the text data using the sentence-transformers package. We will use the following three sentences, stored as a list of strings, as input to the model.
[1]: sentences = [
    "I am breath this morning",
    "I am feeling very good today",
    "I am feeling terrible"
]
We will use the all-MiniLM-L6-v2 model to extract features from the sentences. The model will encode the sentences into a 384-dimensional vector representation. We will then print the features as a pandas dataframe for easy viewing.
[2]: # Load the pre-trained model
model = SentenceTransformer('all-MiniLM-L6-v2')

# Extract features
features = model.encode(sentences)

# Print the features as a pandas dataframe
pd.DataFrame(features, index=sentences)
/notebook/python3.10/dist-packages/huggingface_hub/file_download.py:132: FutureWarning: 'resume_download' is deprecated and will be removed in version 1.8.0. Downloads always resume.
warnings.warn(
    0   1   2   3   4   5   6   7   8   9 ... 374 375 376 377 378 379 380 381 382
  I feel
great
this
morning
  I am
feeling
very
good
today
  I am
feeling
terrible
-0.036462 -0.044379 0.072483 0.034625 0.039564 -0.005461 0.019811 0.012196 -0.020262 -0.044807 ... -0.005889 -0.003328 -0.040055 0.016326 -0.027462 0.017276 0.063253 0.071746 -0.028171
-0.043865 -0.020341 0.085583 0.005310 0.023590 -0.040480 0.079304 -0.009700 -0.042920 -0.023588 ... -0.046309 0.049151 -0.049057 0.017821 -0.010981 -0.010441 0.043070 0.018440 -0.008274
  ✓ 2s completed at 12:20
```

```
import pandas as pd
from sentence_transformers import SentenceTransformer
from transformers import pipeline
```

# PYTHON + Google Colab

## package imports

```
import pandas as pd
from sentence_transformers import SentenceTransformer
from transformers import pipeline
```

A screenshot of a Google Colab notebook titled '6-day-lipynn'. The code cell contains the following imports:

```
(1) import pandas as pd
from sentence_transformers import SentenceTransformer
from transformers import pipeline
```

The output cell shows the following text:

We will begin by extracting features (numerical representations) from the text data using the `sentence-transformers` package. We will use the following three sentences, stored as a list of strings, as input to the model.

```
(1) sentences = [
    "I feel great this morning",
    "I am feeling very good today",
    "I am feeling terrible"
]
```

We will use the `all-MiniLM-L6-v2` model to extract features from the sentences. The model will encode the sentences into a 384-dimensional vector representation. We will then print the features as a pandas dataframe for easy viewing.

```
(2) # Load the pre-trained model
model = SentenceTransformer('all-MiniLM-L6-v2')

# Extract features
features = model.encode(sentences)

# Print the features as a pandas dataframe
pd.DataFrame(features, index=sentences)
```

The output shows the following DataFrame:

	0	1	2	3	4	5	6	7	8	9	374	375	376	377	378	379	380	381	382
I feel great this morning	-0.036462	-0.044373	0.027443	0.034625	0.036954	-0.005463	0.019811	0.012196	-0.026262	-0.048037	-0.005889	-0.003328	-0.040035	0.016326	-0.027462	0.017276	0.065253	0.074746	-0.028711
I am feeling very good today	-0.043865	-0.020341	0.065563	0.005310	0.023590	-0.040420	0.079304	-0.009700	-0.042920	-0.023588	-0.046309	0.049151	-0.049057	0.017821	-0.010981	-0.010441	0.043270	0.018440	-0.008274
I am feeling terrible	0.017495	-0.057904	0.033515	0.001710	0.051957	-0.048159	0.007669	0.119096	0.020969	-0.086960	0.039813	0.000015	-0.074586	-0.016891	0.009449	0.008967	0.051465	-0.009829	0.030000

2s completed at 12:20

## variable assignment, lists, strings

```
sentences = [
    "I feel great this morning",
    "I am feeling very good today",
    "I am feeling terrible"
]
```

# PYTHON + Google Colab

```
# Code + Test
# Feature Extraction
# We will begin by extracting features (numerical representations) from the text data using the sentence-transformers package. We will use the following three sentences, stored as a list of strings, as input to the model.
sentences = [
    "I feel great this morning",
    "I am feeling very good today",
    "I am feeling terrible"
]

We will use the all-MiniLM-L6-v2 model to extract features from the sentences. The model will encode the sentences into a 384-dimensional vector representation. We will then print the features as a pandas data frame for easy viewing.

# Load the pre-trained model.
model = SentenceTransformer('all-MiniLM-L6-v2')

# Extract features
features = model.encode(sentences)

# Print the features as a pandas data frame
pd.DataFrame(features, index=sentences)
```

```
import pandas as pd
from sentence_transformers import SentenceTransformer
from transformers import pipeline
```

## package imports

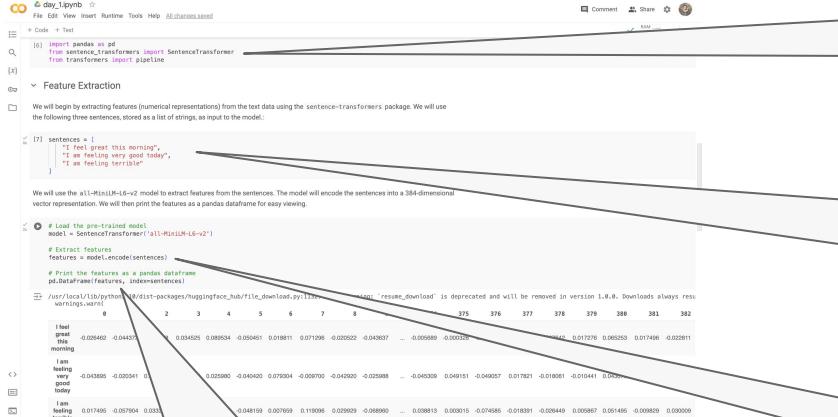
## variable assignment, lists, strings

```
sentences = [
    "I feel great this morning",
    "I am feeling very good today",
    "I am feeling terrible"
]
```

## dot notation, methods, attributes

```
# Extract features
features = model.encode(sentences)
```

# PYTHON + Google Colab



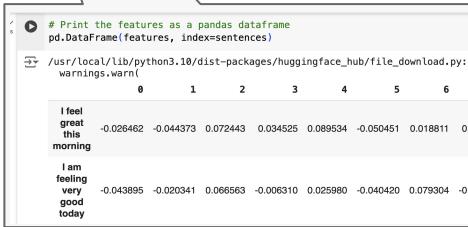
A screenshot of a Google Colab notebook titled 'A-day-Lipynn.ipynb'. The code cell contains imports for pandas, sentence-transformers, and transformers, along with a list of three sentences. The output cell shows the sentences and a warning about the 'resume\_download' method being deprecated.

```
# Code + Test
File Edit View Insert Runtime Help All changes saved
[1]: import pandas as pd
from sentence_transformers import SentenceTransformer
from transformers import pipeline
(x)
Feature Extraction
We will begin by extracting features (numerical representations) from the text data using the sentence-transformers package. We will use the following three sentences, stored as a list of strings, as input to the model.
[1]: sentences = [
    "I feel great this morning",
    "I am feeling very good today",
    "I am feeling terrible"
]
We will use the 'all-MiniLM-L6-v2' model to extract features from the sentences. The model will encode the sentences into a 384-dimensional vector representation. We will then print the features as a pandas dataframe for easy viewing.
[2]: # Load the pre-trained model.
model = SentenceTransformer('all-MiniLM-L6-v2')

# Extract features
features = model.encode(sentences)

# Print the features as a pandas dataframe
pd.DataFrame(features, index=sentences)
/usr/local/lib/python3.10/dist-packages/huggingface_hub/file_download.py:123: UserWarning: 'resume_download' is deprecated and will be removed in version 1.8.0. Downloads always resume.
warnings.warn(
I feel great this morning
  0   1   2   3   4   5   6   7   8   375   376   377   378   380   381   382
  I feel great this morning -0.026462 -0.044373  0.072443  0.034525  0.089534 -0.050451  0.018811  0.
  I am feeling very good today  0.023580 -0.040400  0.079004 -0.009700 -0.042900 -0.023588 ... -0.040309  0.049151 -0.049057  0.017821 -0.010981 -0.010441  0.0050
  I am feeling terrible  0.017495 -0.057904  0.0335 ... 0.048159  0.057659  0.119096 -0.029609 -0.089690 ... 0.058813  0.000015 -0.074586 -0.018891  0.005449  0.008967  0.051465 -0.009829  0.000000
  ✓ 2s completed at 12:20
```

printing



A screenshot of a Google Colab notebook showing the output of a code cell that prints the extracted features as a pandas DataFrame. The output shows the sentences and their corresponding 384-dimensional vector representations.

```
# Print the features as a pandas dataframe
pd.DataFrame(features, index=sentences)

/usr/local/lib/python3.10/dist-packages/huggingface_hub/file_download.py:123: UserWarning: 'resume_download' is deprecated and will be removed in version 1.8.0. Downloads always resume.
warnings.warn(
I feel great this morning
  0   1   2   3   4   5   6
  I feel great this morning -0.026462 -0.044373  0.072443  0.034525  0.089534 -0.050451  0.018811  0.
  I am feeling very good today  0.023580 -0.040400  0.079004 -0.009700 -0.042900 -0.023588 ... -0.040309  0.049151 -0.049057  0.017821 -0.010981 -0.010441  0.0050
  I am feeling terrible  0.017495 -0.057904  0.0335 ... 0.048159  0.057659  0.119096 -0.029609 -0.089690 ... 0.058813  0.000015 -0.074586 -0.018891  0.005449  0.008967  0.051465 -0.009829  0.000000
  ✓ 2s completed at 12:20
```

```
import pandas as pd
from sentence_transformers import SentenceTransformer
from transformers import pipeline
```

package imports

variable assignment, lists, strings

```
sentences = [
    "I feel great this morning",
    "I am feeling very good today",
    "I am feeling terrible"
]
```

dot notation, methods, attributes

```
# Extract features
features = model.encode(sentences)
```

# OUR SOFTWARE STACK



+



+



Time to install  
our stack...



MAX PLANCK INSTITUTE  
FOR HUMAN DEVELOPMENT





dwulff / LLM4BeSci\_2025MetaRep

Type  to search

Code

Issues

Pull requests

Actions

Projects

Wiki

Security

Insights

Settings



LLM4BeSci\_2025MetaRep

Public

Pin

Watch 0

Fork 0

Star 0

main

1 Branch 0 Tags

Go to file

Add file

Code



dwulff Fix link formatting for Taisia Tikhomirova

72cb358 · 7 minutes ago

12 Commits

day\_1

init

yesterday

day\_2

..

13 minutes ago

day\_3

..

13 minutes ago

day\_4

init

yesterday

day\_5

init

yesterday

.DS\_Store

..

13 minutes ago

LICENSE.txt

init

yesterday

README.md

Fix link formatting for Taisia Tikhomirova

7 minutes ago

cover\_metarep.png

Add files via upload

43 minutes ago

notes.txt

init

yesterday

README License



LLM4BeSci at MetaRep, MPIB Berlin, Dec 2025

## About

No description, website, or topics provided.

Readme

View license

Activity

0 stars

0 watching

0 forks

## Releases

No releases published

[Create a new release](#)

## Packages

No packages published

[Publish your first package](#)

## Languages

● Jupyter Notebook 100.0%