

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

I  KING HATE TENSORFLOW #53549

✓ Closed

ghost opened this issue 9 hours ago · 2 comments

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

 **Hugging Face**

Models

Datasets

Spaces

Community

Docs

Enterprise

Pricing

Main

Tasks

Libraries

Languages

Licenses

Other

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

< 1B

6B

12B

32B

128B

> 500B

Libraries

PyTorch

TensorFlow

JAX

Transformers

Diffusers

sentence-transformers

Safetensors

ONNX

GGUF

Transformers.js

MLX

+ 41

Apps

vLLM

TGI

llama.cpp

MLX LM

LM Studio

Ollama

Jan

+ 13

Inference Providers

Groq

Novita

Nebius AI

Cerebras

SambaNova

Nscale

fal

Hyperbolic

+ 11

Models 2,256,270

Filter by name

Full-text search

Inference Available

11 Sort: Trending

Tongyi-MAI/Z-Image-Turbo

Text-to-Image · Updated about 23 hours ago · ± 111k · ± 1.91k

deepseek-ai/DeepSeek-V3.2

Text Generation · ± 685B · Updated 2 days ago · ± 5.01k · ± 626

deepseek-ai/DeepSeek-Math-V2

Text Generation · ± 685B · Updated 6 days ago · ± 7.21k · ± 616

deepseek-ai/DeepSeek-V3.2-Speciale

Text Generation · ± 685B · 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 · ± 1.0B · Updated 1 day ago · ± 226k · ± 614

nvidia/Orchestrator-8B

Text Generation · ± 8B · Updated 1 day ago · ± 1.04k · ± 269

facebook/sam3

Mask Generation · ± 0.9B · Updated 13 days ago · ± 327k · ± 858

apple/starflow

Updated 1 day ago · ± 172

alibaba-pai/Z-Image-Turbo-Fun-Controlnet-Union

Updated 1 day ago · ± 177

jayn7/Z-Image-Turbo-GGUF

Text-to-Image · ± 6B · Updated 5 days ago · ± 70k · ± 154

PrimeIntellect/INTELLECT-3

Text Generation · ± 107B · Updated 6 days ago · ± 2.75k · ± 170

mistralai/Mistral-large-3-675B-Instruct-2512

Updated about 19 hours ago · ± 112 · ± 113

stepfun-ai/Step-Audio-R1

Audio-Text-to-Text · ± 33B · Updated 1 day ago · ± 315 · ± 110

tencent/HunyuanVideo-1.5

Text-to-Video · Updated 1 day ago · ± 2.63k · ± 791

Supertone/supertonic

Text-to-Speech · Updated 11 days ago · ± 14.4k · ± 383

TSB/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

arcee-ai/Trinity-Mini

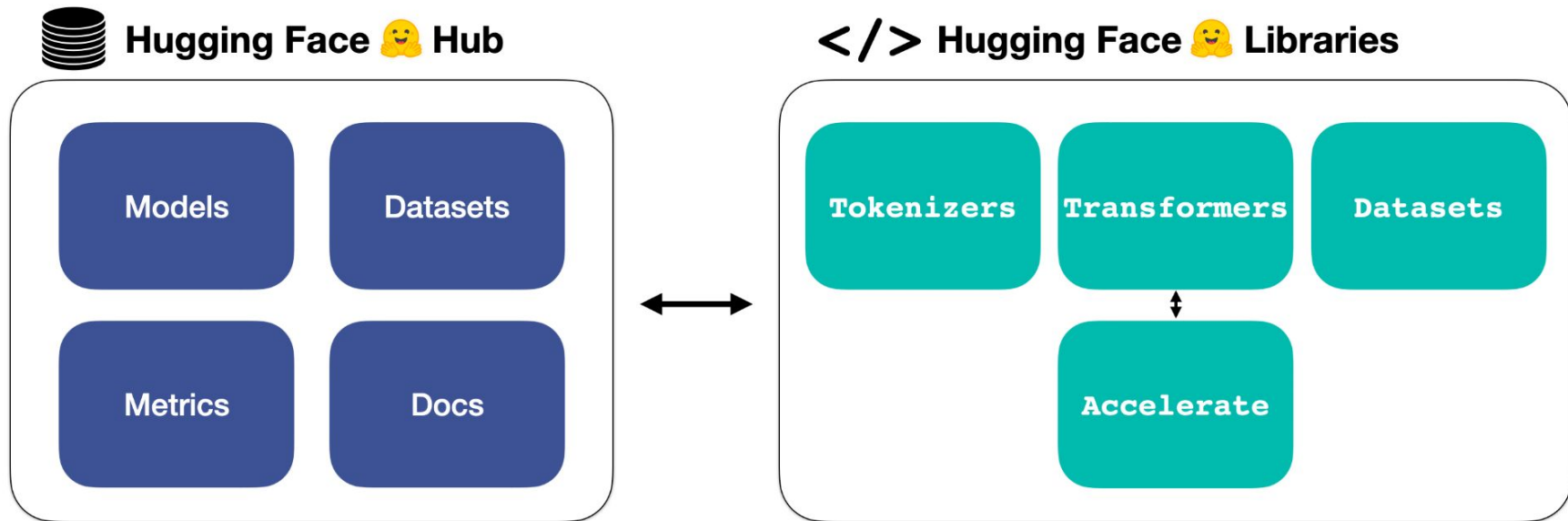
Text Generation · ± 26B · Updated 1 day ago · ± 240 · ± 87

uncloth/One2One-80B-A3B-TextInstruct-GGUF

mistralai/Mistral-7B-Instruct-v0.3

Dirk Wulff MetaRep 2025

THE HUGGING FACE ECOSYSTEM



HUGGING FACE DOCUMENTATION

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



+



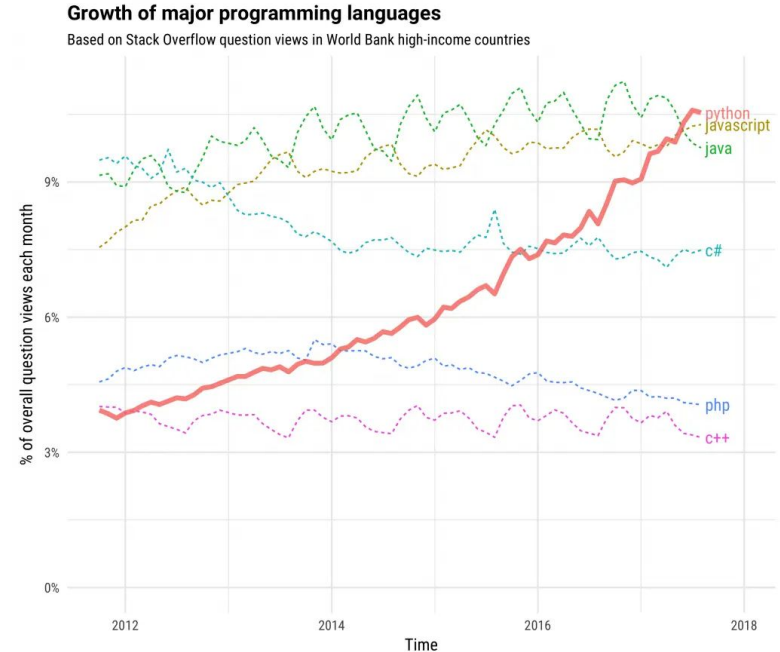
+





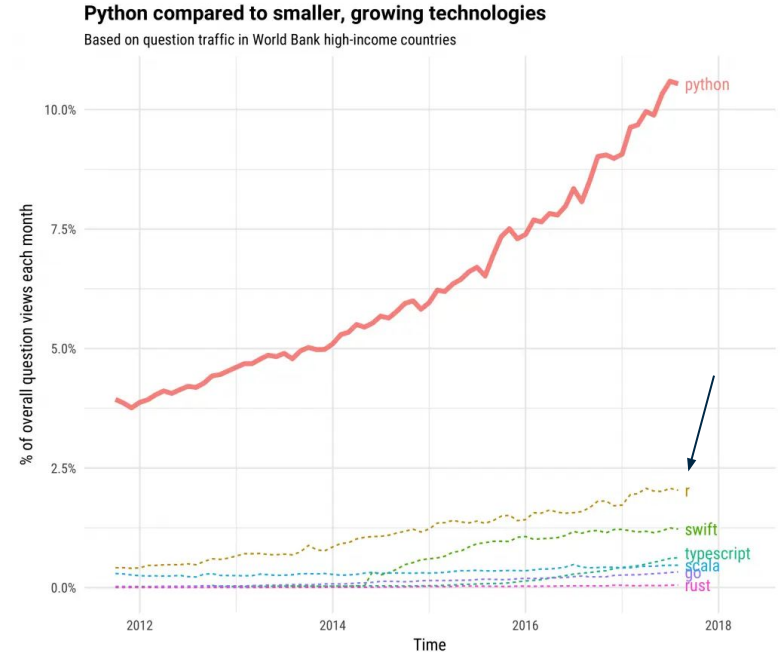


- A **high-level** programming language known for its simplicity and readability.
- Used in various domains such as web development, data analysis, **artificial intelligence**, and scientific computing.





- 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.



OUR SOFTWARE STACK



+




+



PYTHON  + Google Colab 


PYTHON + Google Colab

 day_1.ipynb ☆


File Edit View Insert Runtime Tools Help [All changes saved](#)


+ Code + Text


RAM
Disk


 [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.



```
# 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)
```

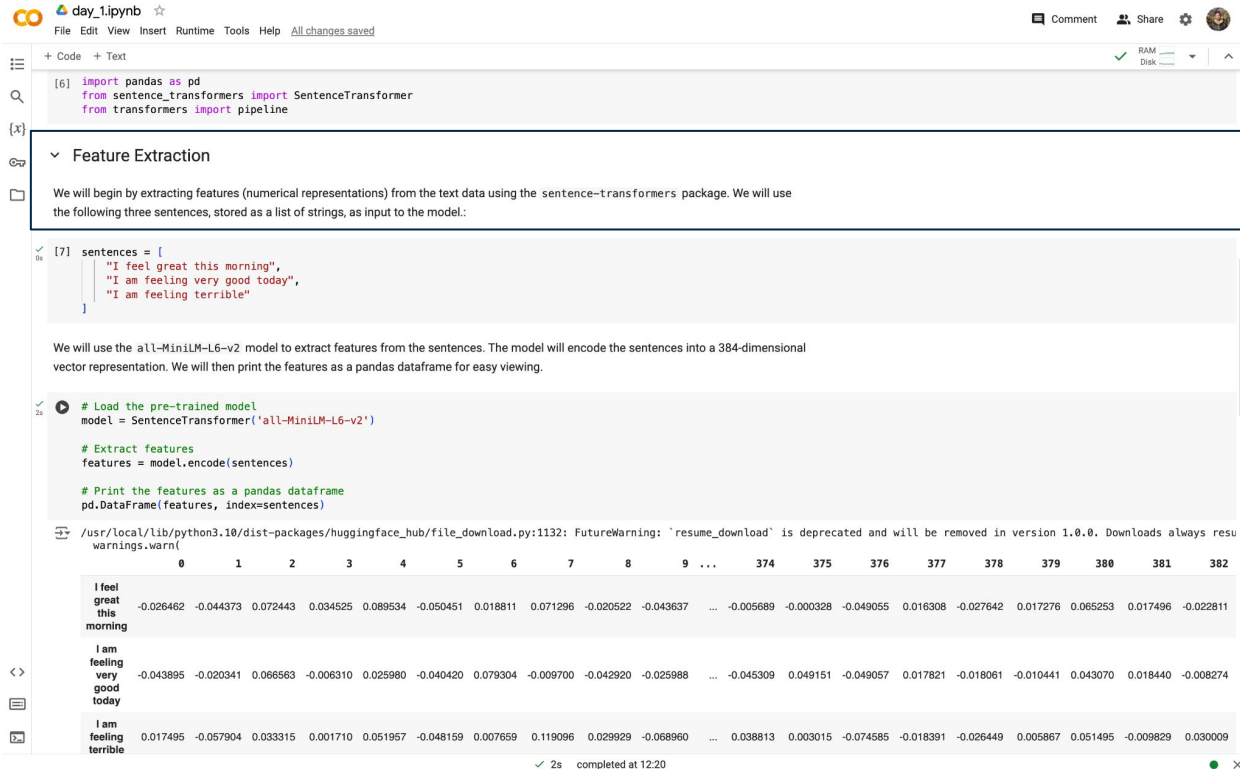
 `/usr/local/lib/python3.10/dist-packages/huggingface_hub/file_download.py:1132: FutureWarning: 'resume_download' is deprecated and will be removed in version 1.0.0. Downloads always resu`
`warnings.warn(`

	0	1	2	3	4	5	6	7	8	9	...	374	375	376	377	378	379	380	381	382
I feel great this morning	-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	0.068563	-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

2s completed at 12:20

PYTHON + Google Colab

1. Markdown



The screenshot shows a Google Colab notebook interface. At the top, the title bar says "day_1.ipynb" with a star icon. Below it are tabs for "File", "Edit", "View", "Insert", "Runtime", "Tools", and "Help". A status bar on the right shows "Comment", "Share", and a user profile icon. The notebook content is divided into two main sections. The first section is a markdown cell titled "Feature Extraction" with a downward arrow icon. It contains the 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:". The second section is a code cell containing Python code. The code imports pandas as pd, sentence_transformers import SentenceTransformer, and transformers import pipeline. It then defines a list of sentences: ["I feel great this morning", "I am feeling very good today", "I am feeling terrible"]. Below the code, there is a warning message: "FutureWarning: 'resume_download' is deprecated and will be removed in version 1.0.0. Downloads always resu". At the bottom of the code cell, there is a table with 382 columns and 3 rows of numerical data. The columns are labeled 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, ..., 374, 375, 376, 377, 378, 379, 380, 381, 382. The rows are labeled "I feel great this morning", "I am feeling very good today", and "I am feeling terrible".

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

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.

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

# Extract features
features = model.encode(sentences)

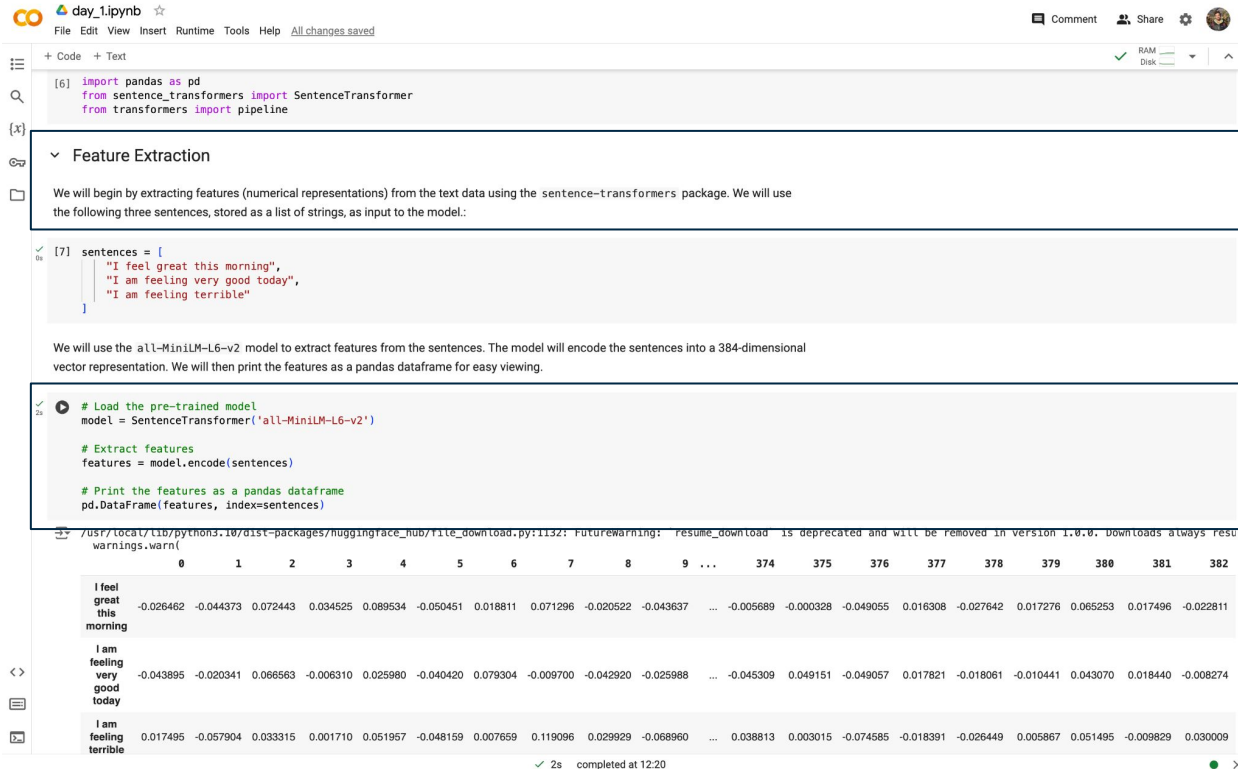
# Print the features as a pandas dataframe
pd.DataFrame(features, index=sentences)
```

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

1. Markdown



The screenshot shows a Google Colab notebook titled "day_1.ipynb". The interface includes a menu bar (File, Edit, View, Insert, Runtime, Tools, Help) and a toolbar with icons for code and text. The notebook content is divided into two main sections:

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:

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

[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.

```
26 # 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 showing the 384-dimensional vector representations for the three sentences. The DataFrame has columns indexed from 0 to 382. The first three rows correspond to the sentences: "I feel great this morning", "I am feeling very good today", and "I am feeling terrible".

	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

2. Code

PYTHON + Google Colab

1. Markdown

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.

2. Code

```
# 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)
```

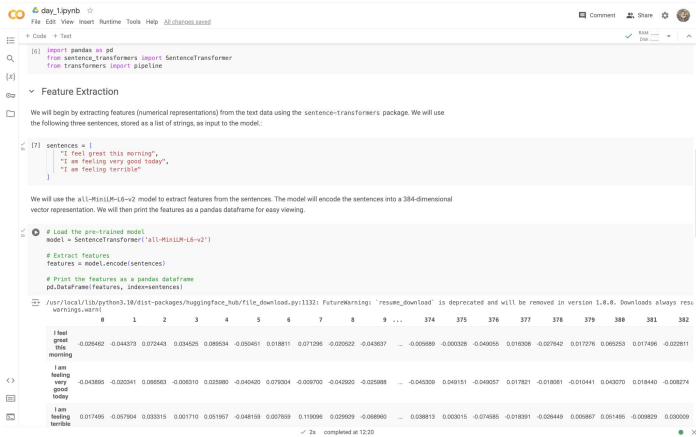
3. Printouts

```
/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 resum
warnings.warnl
```

	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	0.068563	-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

✓ 2s completed at 12:20

PYTHON + Google Colab



```
day_1pythb
File Edit View Insert Runtime Tools Help 88.0a000.0000
+ Code + Text
[5] import pandas as pd
from sentence_transformers import SentenceTransformer
from transformers import pipeline

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.

[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)

/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 when possible.
FutureWarning: 'resume_download' is deprecated and will be removed in version 1.0.0. Downloads always resume when possible.

I feel great this morning
-0.036462 -0.044373 0.072943 0.034026 0.089534 -0.050461 0.018811 0.071396 -0.020522 -0.043037 ... -0.005089 -0.000328 -0.049005 0.016308 -0.027642 0.017276 0.060253 0.017406 -0.002811

I am feeling very good today
-0.043886 -0.020341 0.080903 -0.006310 0.025890 -0.040403 0.076004 -0.008700 -0.040262 -0.025088 ... -0.040309 0.068151 -0.048057 0.017821 -0.018061 -0.010441 0.040270 0.018440 -0.008274

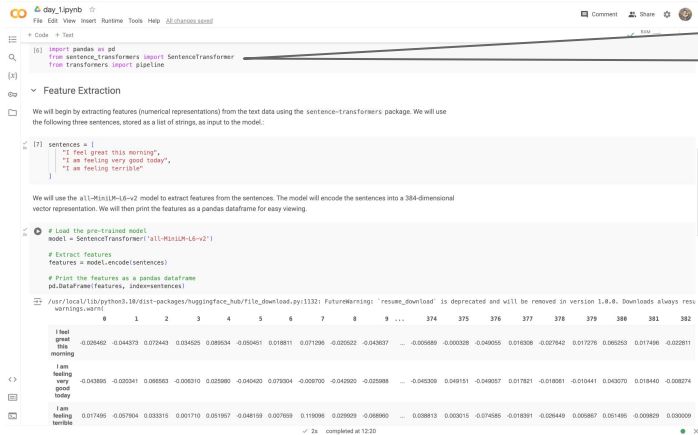
I am feeling terrible
0.017495 -0.057904 0.033315 0.061710 0.051957 -0.048159 0.007659 0.119006 0.029929 -0.088900 ... 0.008813 0.003015 -0.074585 -0.018391 -0.026410 0.005867 0.051405 -0.008629 0.030009

2s completed at 12:20
```

PYTHON + Google Colab

package imports

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



```
6 day_1.pytnb
File Edit View Insert Runtime Tools Help
+ Code + Text
[5] import pandas as pd
from sentence_transformers import SentenceTransformer
from transformers import pipeline

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.

# 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)
```

FutureWarning: 'resume_download' is deprecated and will be removed in version 1.0.0. Downloads always resume when possible. To avoid this warning, you can set the variable 'resume_download' to False

	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	0.072943	0.034026	0.089504	-0.050461	0.018811	0.071296	-0.022522	-0.043037	...	-0.005889	-0.000328	-0.049005	0.016308	-0.027642	0.017276	0.060253	0.017406	-0.022811
I am feeling very good today	-0.043886	-0.020541	0.080903	-0.006210	0.025890	-0.040403	0.076004	-0.008700	-0.040262	-0.023568	...	-0.046309	0.048151	-0.048037	0.017821	-0.018061	-0.010441	0.040270	0.018440	-0.008274
I am feeling terrible	0.017495	-0.057904	0.033315	0.001710	0.051857	-0.048159	0.007659	0.119006	0.022929	-0.048900	...	0.008813	0.003015	-0.074585	-0.018391	-0.026440	0.005867	-0.051405	-0.006829	0.030008

2s completed at 12:20

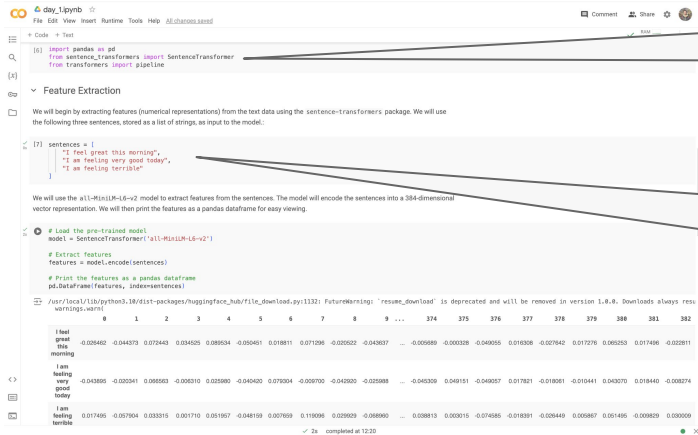
PYTHON + Google Colab

package imports

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

variable assignment, lists, strings

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



```
day_1.pytnb
File Edit View Insert Runtime Tools Help
[15] import pandas as pd
from sentence_transformers import SentenceTransformer
from transformers import pipeline

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.

[17] 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.

[18] # 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	-0.036462	-0.044373	0.072943	0.034326	0.089534	-0.050461	0.018811	0.071296	-0.020522	-0.043037	...	-0.005889	-0.000328	-0.049005	0.016306	-0.027642	0.012726	0.060253	0.017406	-0.020811	
I am feeling very good today	-0.043886	-0.020341	0.088953	-0.006310	0.025890	-0.040430	0.076204	-0.020700	-0.042620	-0.023588	...	-0.043309	0.068151	-0.048037	0.017821	-0.018061	-0.010441	0.040270	0.018440	-0.008274	
I am feeling terrible	0.017495	-0.057904	0.033315	0.001710	0.051867	-0.048159	0.007659	0.119306	0.022929	-0.088900	...	0.008813	0.003015	-0.074685	-0.018331	-0.026440	0.005867	0.051405	-0.006829	0.030008	

2x completed at 12:20

PYTHON + Google Colab

package imports

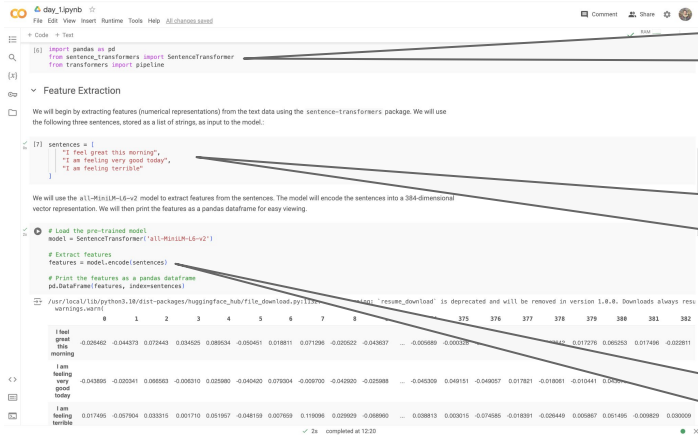
```
import pandas as pd
from sentence_transformers import SentenceTransformer
from transformers import pipeline
```

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)
```



The screenshot shows a Google Colab notebook with the following content:

```
1 import pandas as pd
2 from sentence_transformers import SentenceTransformer
3 from transformers import pipeline

4
5 # Feature Extraction
6
7 We will begin by extracting features (numerical representations) from the text data using the sentence-transformers package. We will use
8 the following three sentences, stored as a list of strings, as input to the model.
9
10 sentences = [
11     "I feel great this morning",
12     "I am feeling very good today",
13     "I am feeling terrible"
14 ]
15
16 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
17 vector representation. We will then print the features as a pandas dataframe for easy viewing.
18
19 # Load the pre-trained model
20 model = SentenceTransformer('all-MiniLM-L6-v2')
21
22 # Extract features
23 features = model.encode(sentences)
24
25 # Print the features as a pandas dataframe
26 pd.DataFrame(features, index=sentences)
```

The output shows a pandas DataFrame with 3 rows and 384 columns. The first three columns are labeled with the sentences, and the remaining columns contain numerical values representing the features.

	0	1	2	3	4	5	6	7	8	...	375	376	377	378	379	380	381	382
I feel great this morning	-0.026462	-0.044373	0.072943	0.034026	0.089504	-0.050461	0.018811	0.071296	-0.020502	-0.043037	...	-0.009389	-0.002705	0.014432	0.017276	0.060203	0.017406	-0.020811
I am feeling very good today	-0.043886	-0.020341	0.080903	-0.006310	0.025990	-0.040403	0.076004	-0.020700	-0.040202	-0.020508	...	-0.040309	0.068151	-0.040657	0.017821	-0.018061	-0.010441	0.000000
I am feeling terrible	0.017495	-0.027604	0.033315	0.001710	0.051907	-0.048109	0.007069	0.119006	0.022929	-0.088900	...	0.008813	0.003015	-0.074685	-0.018391	-0.020440	0.005867	0.001405

PYTHON + Google Colab

package imports

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

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)
```

printing

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

	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
I am feeling very good today	-0.043895	-0.020341	0.066563	-0.006310	0.025980	-0.040420	0.079304
I am feeling terrible	0.017495	-0.079034	0.033333	0.048159	0.007659	0.119006	0.022929

OUR SOFTWARE STACK



+



+



Time to install our stack...



MAX PLANCK INSTITUTE
FOR HUMAN DEVELOPMENT



dwulff

LLM4BeSci_2025MetaRep

Q

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

Q

Go to file

t

Add file

<> Code

dwulff

Fix link formatting for Taisiia 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 Taisiia 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%