

Intro Hugging Face and Python

Dirk Wulff & Zak Hussain



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](#) [Posts](#) [Docs](#) [Enterprise](#) [Pricing](#) [⌵](#) 

Tasks

Libraries

Datasets

Languages

Licenses

Other

Multimodal

Audio-Text-to-Text

Image-Text-to-Text

Visual Question Answering

Document Question Answering

Video-Text-to-Text

Any-to-Any

Computer Vision

Depth Estimation

Image Classification

Object Detection

Image Segmentation

Text-to-Image

Image-to-Text

Image-to-Image

Image-to-Video

Unconditional Image Generation

Video Classification

Text-to-Video

Zero-Shot Image Classification

Mask Generation

Zero-Shot Object Detection

Text-to-3D

Image-to-3D

Image Feature Extraction

Keypoint Detection

Natural Language Processing

Text Classification

Token Classification

Table Question Answering

Question Answering


Models

1,303,814


Filter by name

Full-text search


11 Sort: Trending

 **hexgrad/Kokoro-82M**


Text-to-Speech • Updated 1 day ago • ⬆ 25k • ❤ 1.99k

 **openbmb/MiniCPM-o-2_6**


Any-to-Any • Updated about 2 hours ago • ⬆ 15k • ❤ 608

 **microsoft/phi-4**


Text Generation • Updated 11 days ago • ⬆ 124k • ❤ 1.44k

 **MiniMaxAI/MiniMax-Text-01**


Text Generation • Updated 2 days ago • ⬆ 2.38k • ❤ 404

 **deepseek-ai/DeepSeek-V3**


Updated 21 days ago • ⬆ 155k • ❤ 2.03k

 **NovaSky-AI/Sky-T1-32B-Preview**


Text Generation • Updated 6 days ago • ⬆ 7.51k • ❤ 474

 **jinaai/ReaderLM-v2**


Text Generation • Updated 2 days ago • ⬆ 2.64k • ❤ 257

 **MiniMaxAI/MiniMax-VL-01**

Text Generation • Updated 4 days ago • ⬆ 635 • ❤ 197

 **black-forest-labs/FLUX.1-dev**

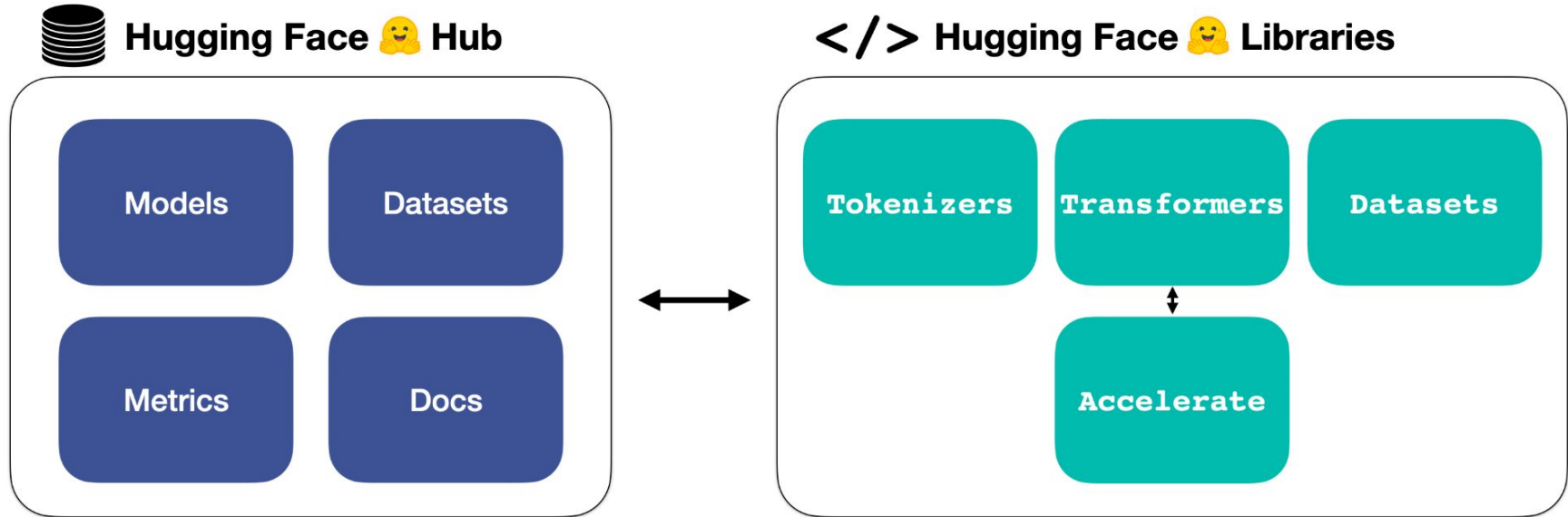
Text-to-Image • Updated Aug 16, 2024 • ⬆ 1.3M • ⚡ • ❤ 8.11k

 **internlm/internlm3-8b-instruct**

Text Generation • Updated 3 days ago • ⬆ 3.24k • ❤ 171

Dirk Wulff & Zak Hussain GSERM 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



+



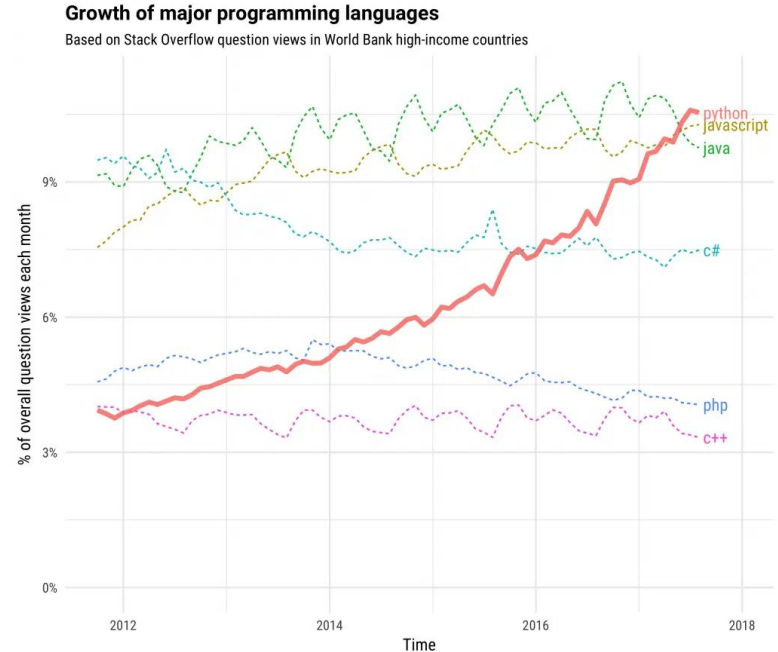
+





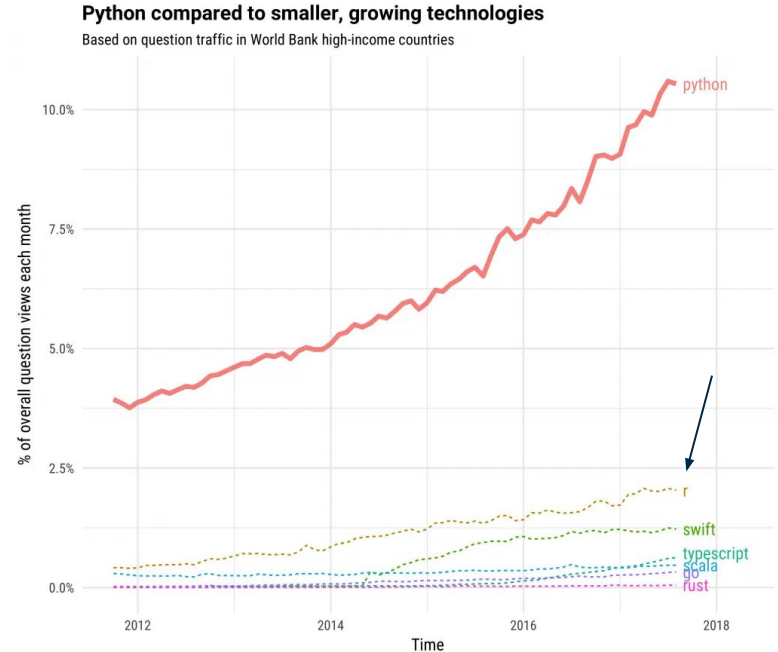


- A **high-level** programming language known for its simplicity and readability.
- Used in various domains such as web development, data analysis, **artificial intelligence**, and scientific computing.





- 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
```

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)
```

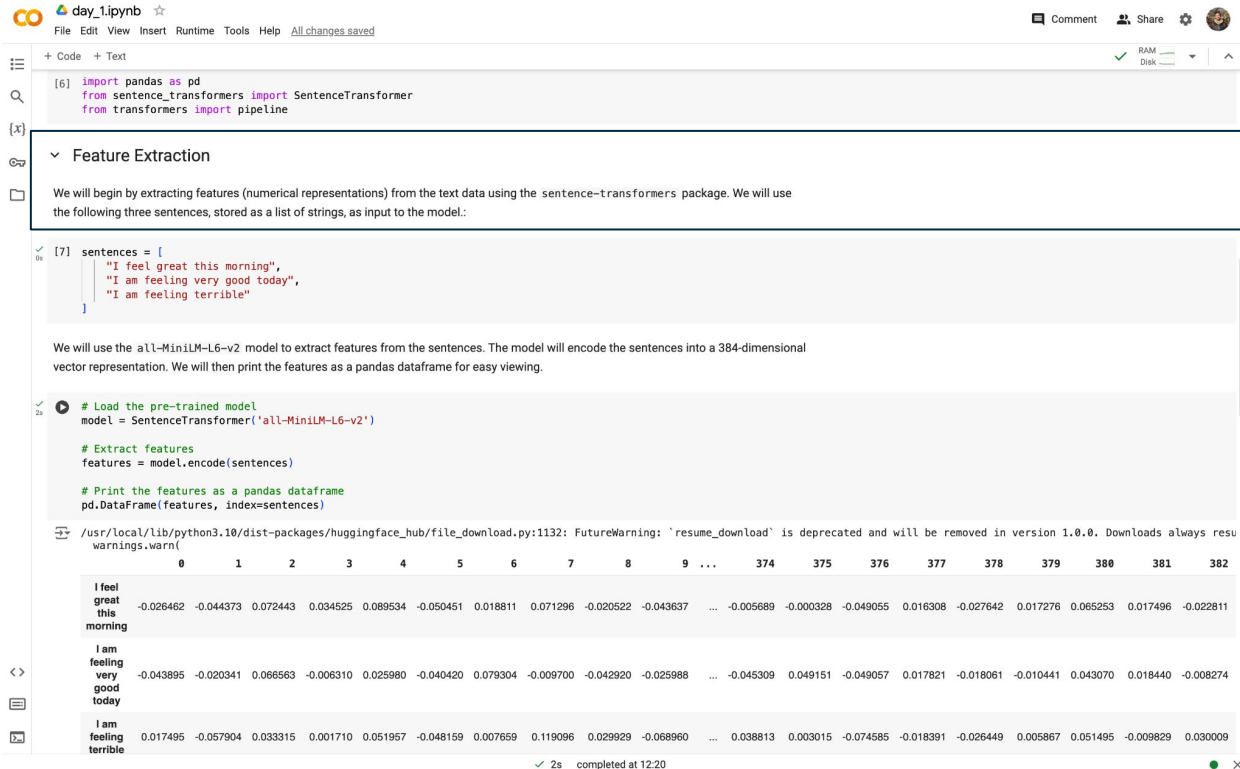
```
/usr/local/lib/python3.10/dist-packages/huggingface_hub/file_download.py:1132: FutureWarning: 'resume_download' is deprecated and will be removed in version 1.0.0. Downloads always resume when possible.
warnings.warn(
```

	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 titled "day_1.ipynb". The interface includes a top menu bar with "File", "Edit", "View", "Insert", "Runtime", "Tools", and "Help". Below the menu is a toolbar with icons for adding code or text cells, a search icon, and a RAM/Disk usage indicator showing 100% RAM and 0% Disk.

The notebook contains two cells:

- Markdown Cell:** Titled "Feature Extraction", 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:"
- Code Cell:** Contains Python code for feature extraction. It imports pandas, sentence-transformers, and transformers. It defines a list of sentences and uses the all-MiniLM-L6-v2 model to extract features. The features are printed as a pandas DataFrame.

The code cell output shows a FutureWarning and a pandas DataFrame with 382 columns and 3 rows of numerical features.

```
[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.

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

# Extract features
features = model.encode(sentences)

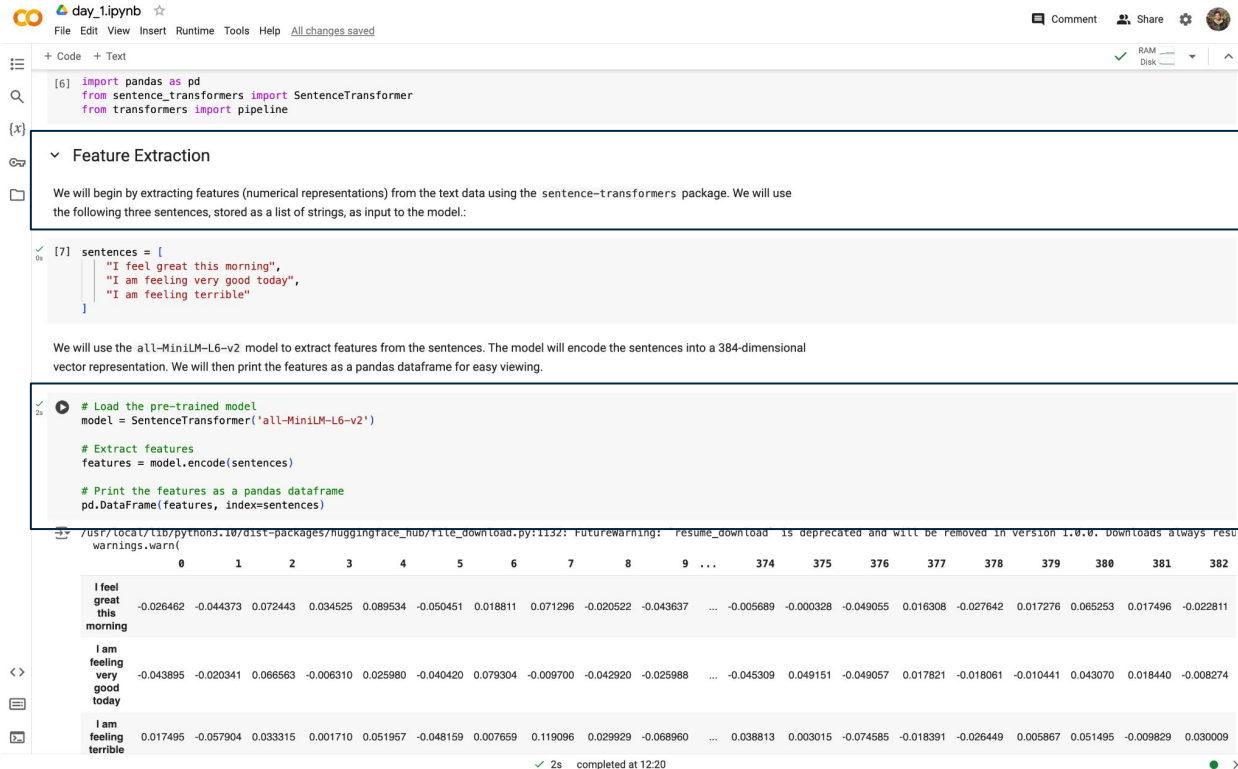
# Print the features as a pandas dataframe
pd.DataFrame(features, index=sentences)
```

/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. To avoid this warning, you can set the `resume_download` parameter 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	0.072443	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 notebook contains a code cell and a markdown cell. The code cell imports pandas, sentence-transformers, and transformers, and defines a list of sentences. The markdown cell explains the goal of extracting features using the sentence-transformers package and the all-MiniLM-L6-v2 model. The output of the code cell is a pandas DataFrame showing the 384-dimensional vector representations for the three sentences.

```
[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.

```
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)
```

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=False.

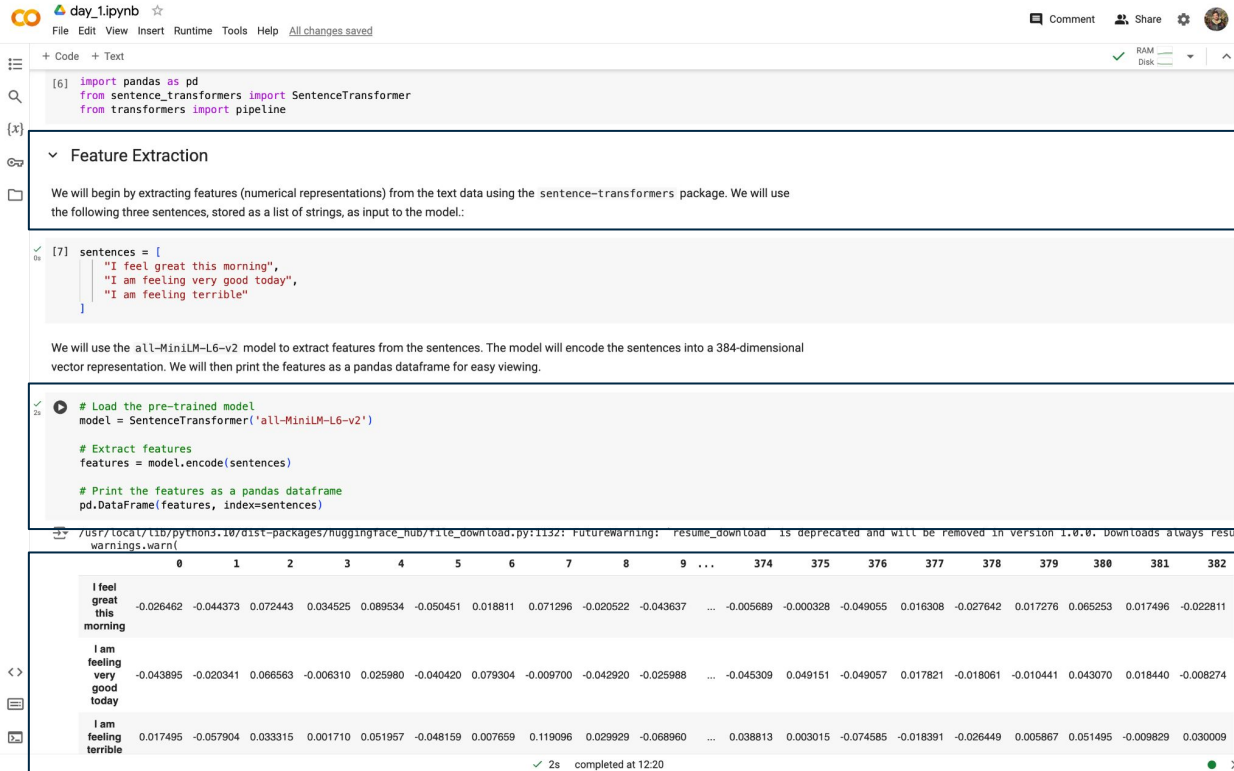
	0	1	2	3	4	5	6	7	8	9	...	374	375	376	377	378	379	380	381	382
I feel great this morning	-0.026462	-0.044373	0.072443	0.034525	0.089534	-0.050451	0.018811	0.071296	-0.020522	-0.043637	...	-0.005689	-0.000328	-0.049055	0.016308	-0.027642	0.017276	0.065253	0.017496	-0.022811
I am feeling very good today	-0.043895	-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

2s completed at 12:20

2. Code

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 adding code or text, searching, and managing the notebook. The notebook content is divided into three sections:

- Code Section:** Contains two code cells. The first cell imports pandas, sentence-transformers, and transformers. The second cell defines a list of sentences:

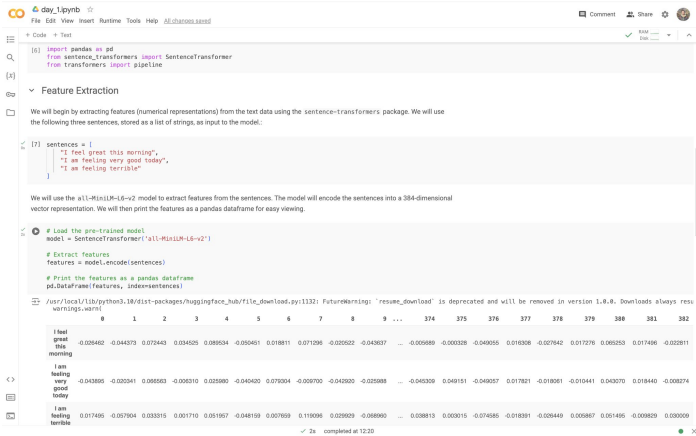
```
sentences = ["I feel great this morning", "I am feeling very good today", "I am feeling terrible"]
```
- Markdown Section:** Contains two text blocks. The first block, titled "Feature Extraction", explains the goal of extracting numerical representations from text using the sentence-transformers package. The second block explains that the all-MiniLM-L6-v2 model will be used to encode the sentences into a 384-dimensional vector representation.
- Printout Section:** Contains a code cell that loads the pre-trained model, extracts features, and prints them as a pandas dataframe. The output is a table with 384 columns (0 to 382) and 3 rows of data corresponding to the 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

2. Code

3. Printouts

PYTHON + Google Colab



```
day_1pythn
File Edit View Insert Runtime Tools Help 88.04000.0000
+ 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)

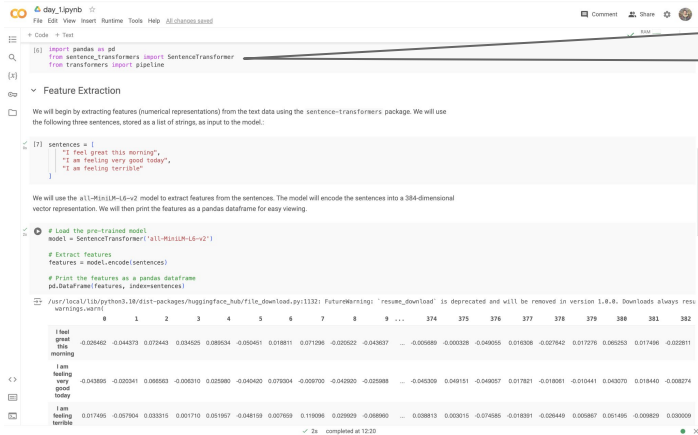
/usr/local/lib/python3.10/dist-packages/huggingface_hub/file_download.py:1132: FutureWarning: 'resume_download' is deprecated and will be removed in version 1.0.0. Downloads always resume when possible.
0 1 2 3 4 5 6 7 8 9 ... 374 375 376 377 378 379 380 381 382
I feel great this morning -0.026462 -0.044373 0.072943 0.034026 0.089504 -0.050461 0.018811 0.071296 -0.020522 -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 -0.006210 0.025890 -0.040403 0.077604 -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 -0.057604 0.033315 0.001710 0.051857 -0.048159 0.007769 0.119006 0.029929 -0.088900 ... 0.008813 0.003015 -0.074585 -0.018391 -0.026440 0.005867 0.051405 -0.006829 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 88.48000.0000
+ 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.

# 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)
```

Warning: 'resume_download' is deprecated and will be removed in version 1.0.0. Downloads always resume when possible. To avoid this message, please use `resume_from_checkpoint=True`.

	0	1	2	3	4	5	6	7	8	9	...	374	375	376	377	378	379	380	381	382
I feel great this morning	-0.020462	-0.044373	0.072943	0.034026	0.089304	-0.050461	0.018811	0.071296	-0.020522	-0.043037	...	-0.005089	-0.000328	-0.049005	0.016308	-0.027642	0.017276	0.060253	0.017406	-0.020811
I am feeling very good today	-0.043886	-0.020541	0.080903	-0.006210	0.025890	-0.040403	0.077604	-0.008700	-0.040262	-0.020588	...	-0.046309	0.068151	-0.048057	0.017821	-0.018061	-0.010441	0.040270	0.018440	-0.008274
I am feeling terrible	0.017495	-0.057604	0.033315	0.001710	0.051867	-0.048169	0.007669	0.119306	0.022929	-0.048900	...	0.008813	0.003015	-0.074585	-0.018391	-0.026440	0.005867	-0.051405	-0.008629	0.030008

2x completed at 12:20

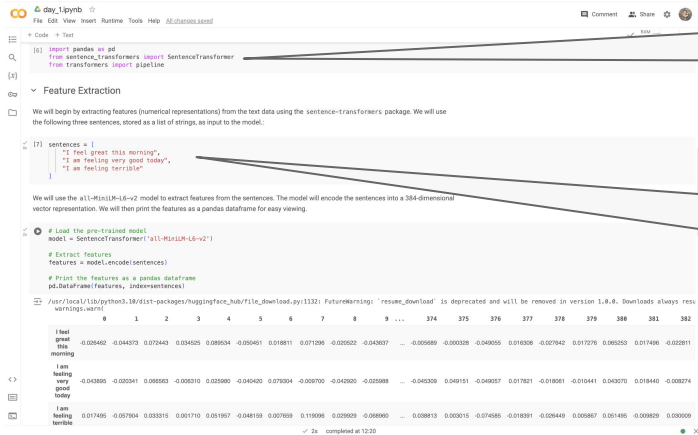
PYTHON + Google Colab

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"
]
```



```
day_1.py:nb
File Edit View Insert Runtime Tools Help
[1] 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)
```

	0	1	2	3	4	5	6	7	8	...	374	375	376	377	378	379	380	381	382	
I feel great this morning	-0.026462	-0.044373	0.072943	0.034326	0.089534	-0.050461	0.018811	0.071296	-0.022522	-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	-0.006210	0.025890	-0.040430	0.076204	-0.020700	-0.042620	-0.023588	...	-0.043339	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	...	0.028813	0.003015	-0.074585	-0.018331	-0.026440	0.005867	0.051405	-0.008629	0.030009

2x completed at 12:20

PYTHON + Google Colab

package imports

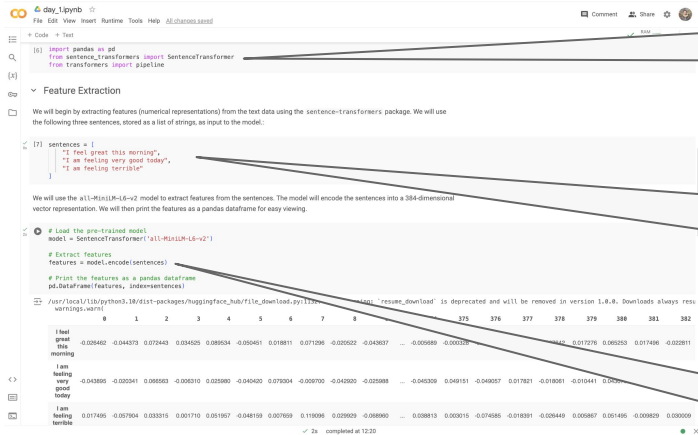
```
import pandas as pd
from sentence_transformers import SentenceTransformer
from transformers import pipeline
```

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:

```
File Edit View Insert Runtime Tools Help
+ Code + Text
[1] 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)
```

The output shows a pandas DataFrame with 3 rows and 384 columns. The first row corresponds to "I feel great this morning", the second to "I am feeling very good today", and the third to "I am feeling terrible". The values are numerical representations of the sentences.

	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	0.017276	0.060203	0.017406	-0.020811	
I am feeling very good today	-0.043886	-0.020341	0.080903	-0.006310	0.025990	-0.040403	0.077604	-0.020700	-0.040260	-0.020588	...	-0.046309	0.068151	-0.048037	0.017821	-0.018061	-0.010441	0.000000
I am feeling terrible	0.017495	-0.057904	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

✓ 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"
]
```

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



LLM4BeSci_Ljubljana2025

Public

Pin

Unwatch 1

Fork 1

Star 0

main 1 Branch 0 Tags

Go to file

Add file

<> Code

Zak-Hussain update

b1c1106 · 5 days ago 26 Commits

day_1	add bonus task	2 weeks ago
day_2	typo	2 weeks ago
day_3	typos	2 weeks ago
day_4	typos	2 weeks ago
day_5	typo	2 weeks ago
.gitignore	Initial commit	2 weeks ago
LICENSE.txt	populate	2 weeks ago
README.md	re-order installation steps	5 days ago
notes.md	update	5 days ago

README License

LLM4BeSci at GSERM, Ljubljana 2025

The course introduces the use of open-source large language models (LLMs) from the Hugging Face ecosystem for research in the behavioral and social sciences.

About

The course introduces the use of open-source large language models (LLMs) from the Hugging Face ecosystem for research in the behavioral and social sciences.

Readme

View license

Activity

0 stars

1 watching

1 fork

Releases

No releases published

Create a new release

Packages

No packages published

Publish your first package

Languages

Jupyter Notebook 100.0%