

Intro to deep learning

Dr. Janoś Gabler, University of Bonn

Lecture 3: First experience with HuggingFace



What you will learn in this lecture

- How to use pre-trained models from huggingface
- How to clear the model cache to free up space
- How to set some tuning parameters to get what you want
- How to ask a model about the course logistics!

Ken Judd in Bonn

- OSE meetup, Thursday, 6 pm, no registration needed
- Office hours, Thursday afternoon, register [\[here\]](#) (IAME lounge (Lennestr. 43))
- Location of both: IAME lounge (Lennestr. 43)

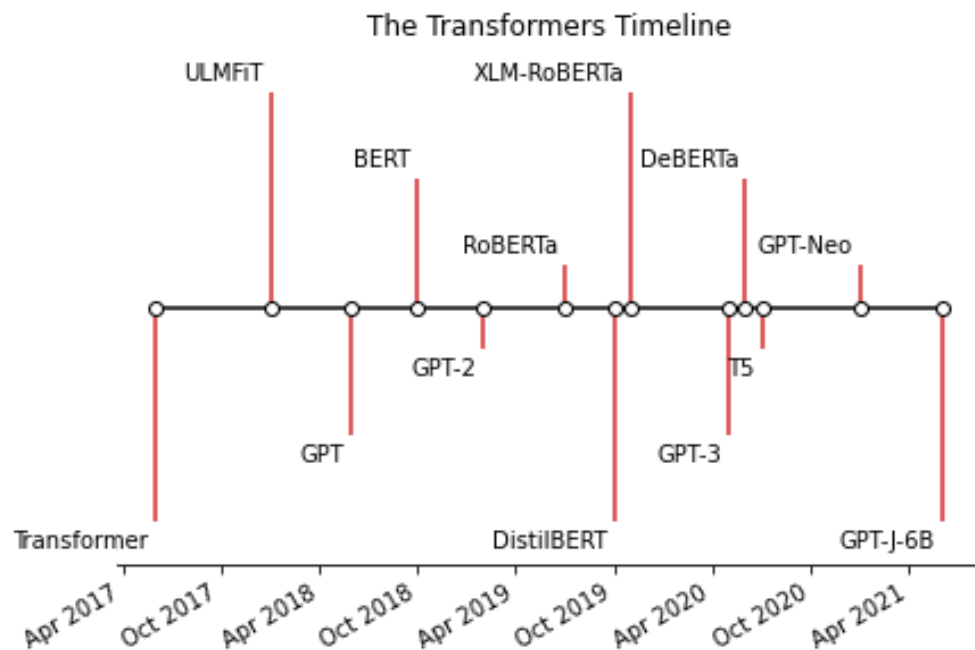
Should we start a bit earlier?

- Some students cannot attend second half
- Currently:
 - Start at 12:15
 - Break from 13:45 to 14:15
- Proposal
 - Start at 12:00
 - Have break from 14:00 to 14:30

A bit of history

- 2010: Start of deep learning revolution
 - CNNs win computer vision contests
 - Training on GPUs becomes mainstream
- Ca. 2015:
 - Transfer learning becomes mainstream in computer vision
 - DNNs outperform everything else in computer vision
- 2017 revolution starts in NLP:
 - Attention is all you need
 - ULMFiT brings transfer learning to NLP

Timeline



What is transfer learning

- Pre-training: Train a NN on giant amounts of unlabeled data
 - Language modelling: Predict next word in a sequence (e.g. GPT)
 - Masked language modelling: Predict a word that was removed (e.g. Bert)
- Domain adaptation: Continue on specialized text corpus
- Fine tuning:
 - Specialize last layer to the target task
 - Train it on labeled data

Plan for today

- Work with pre-trained and pre-fine-tuned models!
- Meet the most relevant NLP tasks:
 - Text classification / sentiment analysis
 - Named entity recognition
 - Question answering
 - Summarization
 - Translation
- Work with the transformers library for the first time

``transformers.pipeline``

- For us: Model is a black box that takes text and returns text
- In reality:
 - Text is converted to numbers (tokenizer)
 - Main model
 - Task specific head
 - Result is converted to text again
 - Post-processing
- ``transformers.pipeline`` hides all of this!

Important

- All the tasks ask you to use specific models instead of default
- This is to limit downloaded file-sizes

An example text

```
text = """Dear Amazon, last week I ordered an Optimus  
Prime action figure from your online store in Germany.  
Unfortunately, when I opened the package, I discovered  
to my horror that I had been sent an action figure of  
Megatron instead! As a lifelong enemy of the Decepticons,  
I hope you can understand my dilemma. To resolve the  
issue, I demand an exchange of Megatron for the Optimus  
Prime figure I ordered. Enclosed are copies of my records  
concerning this purchase. I expect to hear from you soon.  
Sincerely, Bumblebee."""
```

- Does the text have a negative or positive connotation?
- What people/objects are named in the text?
- What are main points of the text?
- What if we need a translation of the text?

Text classification

```
>>> from transformers import pipeline
>>> classifier = pipeline(task="text-classification")
>>> sentiments = classifier(text)
>>> sentiments
[{'label': 'NEGATIVE', 'score': 0.9015460014343262}]
```

- Create a `pipeline` with the task `"text-classification"`
- A task-specific default model is used, but you can define your model!
- The results are a lists of dictionaries
- Contain predictions and confidence scores.

Task 1

15 minutes

Named entity recognition

```
>>> ner_tagger = pipeline(task="ner")
>>> entities = ner_tagger(text)
>>> entities
[{'entity_group': 'ORG',
  'score': 0.9411546,
  'word': 'amazon',
  'start': 5,
  'end': 11},
 {'entity_group': 'MISC',
  'score': 0.9315696,
  'word': 'optimus prime',
  'start': 36,
  'end': 49},
  ...
]
```

- The task key is `"ner"`
- Additional arguments to define the pipeline
- The output is a list of dictionaries with recognized entities
- Entities are also classified as
 - Organizations
 - People
 - ...

Task 2

10 min

Optional arguments

```
>>> def add(x, y=2):  
...     return x + y
```

```
>>> add(x=4, y=5)  
9
```

```
>>> add(x=4)  
6
```

```
>>> add(y=4)
```

```
-----  
TypeError                                Traceback (most recent call la  
Cell In[15], line 4
```

```
1 def add(x, y=2):  
2     return x + y  
----> 4 add(y=4)
```

```
TypeError: add() missing 1 required positional argument: '
```

- Python functions can have optional arguments
- Use `=` sign in function signature to define them
- The default can be overwritten in function calls

Optional arguments in pipeline

- Which optional arguments can be used, depends on the model / task
- Sometimes a bit hard to find in the documentation
- For the NER task, we look at the argument `aggregation_strategy``

(Default: `simple`). There are several aggregation strategies:

`none`: Every token gets classified without further aggregation.

`simple`: Entities are grouped according to the default schema (B-, I- tags get merged when the tag is similar).

`first`: Same as the `simple` strategy except words cannot end up with different tags. Words will use the tag of the first token when there is ambiguity.

`average`: Same as the `simple` strategy except words cannot end up with different tags. Scores are averaged across tokens and then the maximum label is applied.

`max`: Same as the `simple` strategy except words cannot end up with different tags. Word entity will be the token with the maximum score.

Pro-tip

- Always read the documentation of the libraries you are using
- Knowing and using optional arguments can set you apart from most others!
- Default values are not always chosen well
- Some arguments have a default value but should not

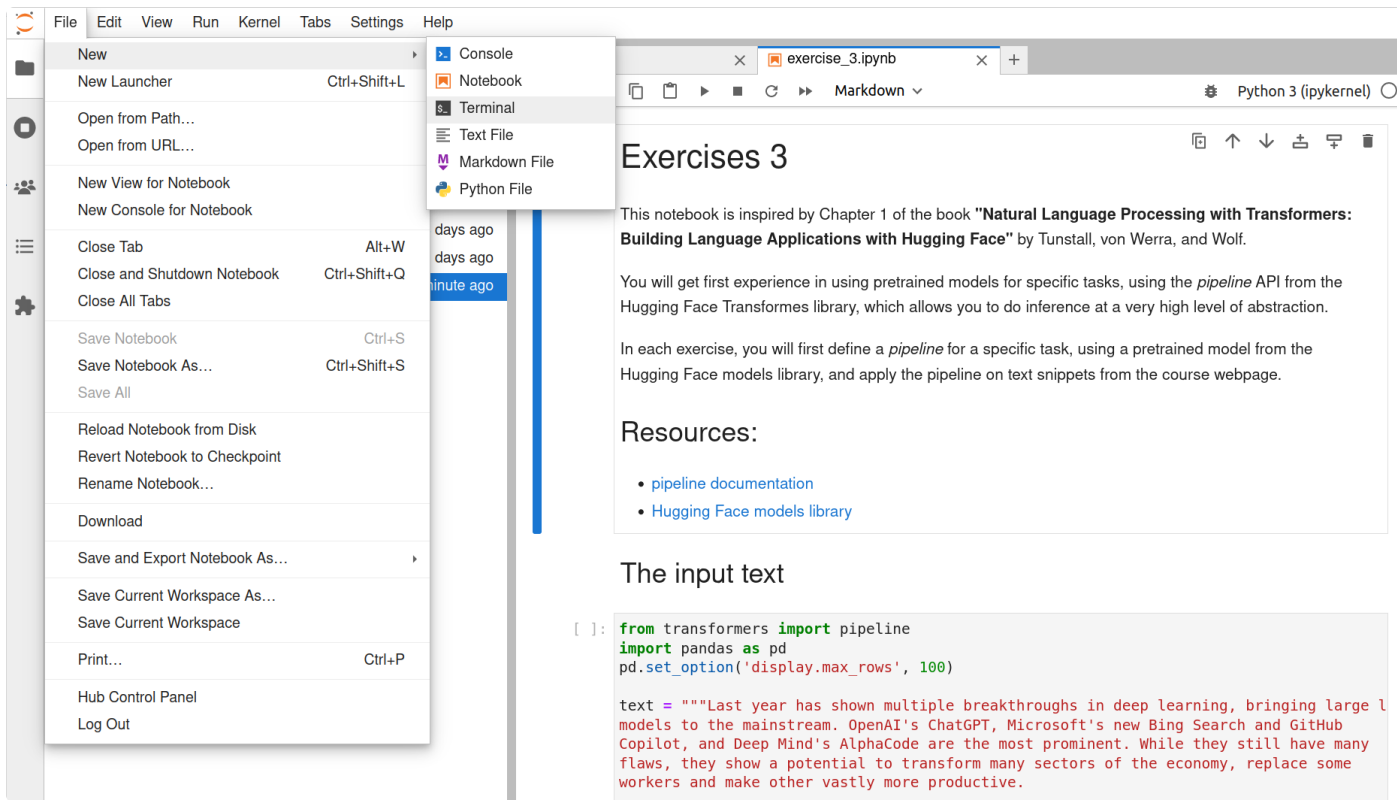
Task 3

(10 min)

Clearing the cache

- Models are large (200 MB to several GB)
- On JupyterHub you only have 3 GB of space
- Even on your laptop, you might run out of space
- Need to learn how to delete models
- Similar on all platforms
- Important: On your laptop you need to activate the environment!

Open a terminal on JupyterHub



The screenshot shows the JupyterLab interface. The 'File' menu is open, and the 'Terminal' option is highlighted. The main notebook area displays 'Exercises 3' with text about natural language processing and a code cell for text processing using transformers and pandas.

File Menu Options:

- New
 - New Launcher (Ctrl+Shift+L)
 - Open from Path...
 - Open from URL...
 - New View for Notebook
 - New Console for Notebook
- Close Tab (Alt+W)
- Close and Shutdown Notebook (Ctrl+Shift+Q)
- Close All Tabs
- Save Notebook (Ctrl+S)
- Save Notebook As... (Ctrl+Shift+S)
- Save All
- Reload Notebook from Disk
- Revert Notebook to Checkpoint
- Rename Notebook...
- Download
- Save and Export Notebook As...
- Save Current Workspace As...
- Save Current Workspace
- Print... (Ctrl+P)
- Hub Control Panel
- Log Out

Terminal Option:

- Console
- Notebook
- Terminal
- Text File
- Markdown File
- Python File

Exercises 3

This notebook is inspired by Chapter 1 of the book "**Natural Language Processing with Transformers: Building Language Applications with Hugging Face**" by Tunstall, von Werra, and Wolf.

You will get first experience in using pretrained models for specific tasks, using the *pipeline* API from the Hugging Face Transformers library, which allows you to do inference at a very high level of abstraction.

In each exercise, you will first define a *pipeline* for a specific task, using a pretrained model from the Hugging Face models library, and apply the pipeline on text snippets from the course webpage.

Resources:

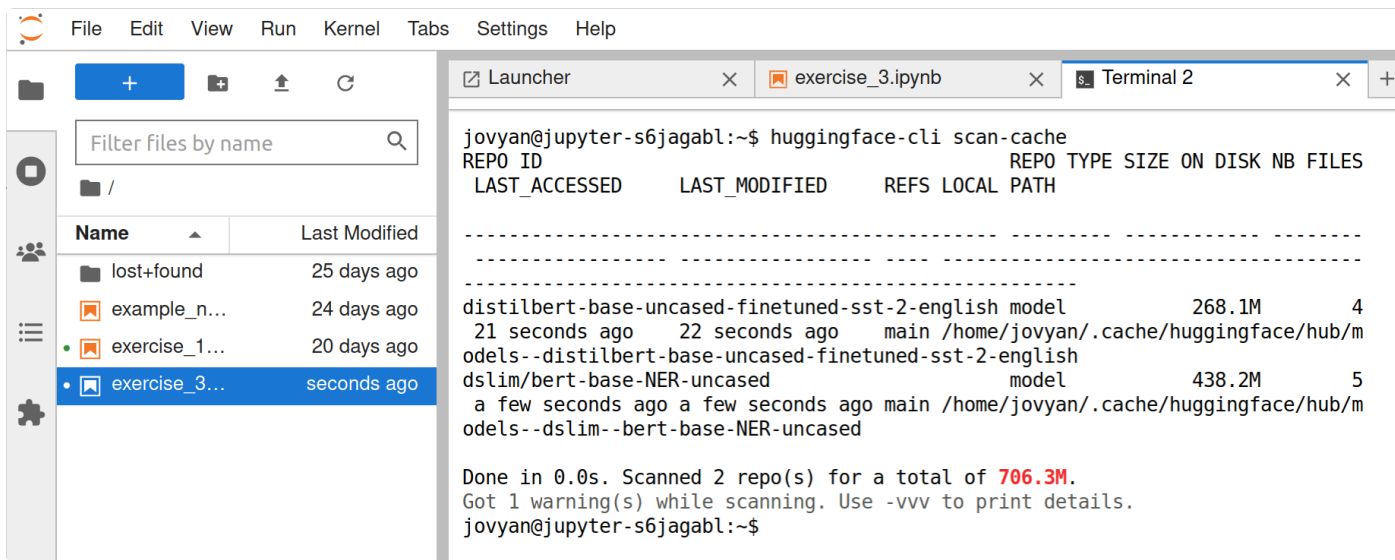
- [pipeline documentation](#)
- [Hugging Face models library](#)

The input text

```
[ ]: from transformers import pipeline
import pandas as pd
pd.set_option('display.max_rows', 100)

text = """Last year has shown multiple breakthroughs in deep learning, bringing large l
models to the mainstream. OpenAI's ChatGPT, Microsoft's new Bing Search and GitHub
Copilot, and Deep Mind's AlphaCode are the most prominent. While they still have many
flaws, they show a potential to transform many sectors of the economy, replace some
workers and make other vastly more productive.
```

List your models with `scan-cache`



The screenshot shows a JupyterLab interface with a file browser on the left and a terminal window on the right. The file browser shows a directory structure with files like 'lost+found', 'example_n...', 'exercise_1...', and 'exercise_3...'. The terminal window shows the output of the command `huggingface-cli scan-cache`.

```
jovyan@jupyter-s6jagabl:~$ huggingface-cli scan-cache
```

REPO ID	REPO TYPE	SIZE ON DISK	NB FILES

distilbert-base-uncased-finetuned-sst-2-english	model	268.1M	4

dsllm-bert-base-NER-uncased	model	438.2M	5

Done in 0.0s. Scanned 2 repo(s) for a total of **706.3M**.
Got 1 warning(s) while scanning. Use -vvv to print details.
jovyan@jupyter-s6jagabl:~\$

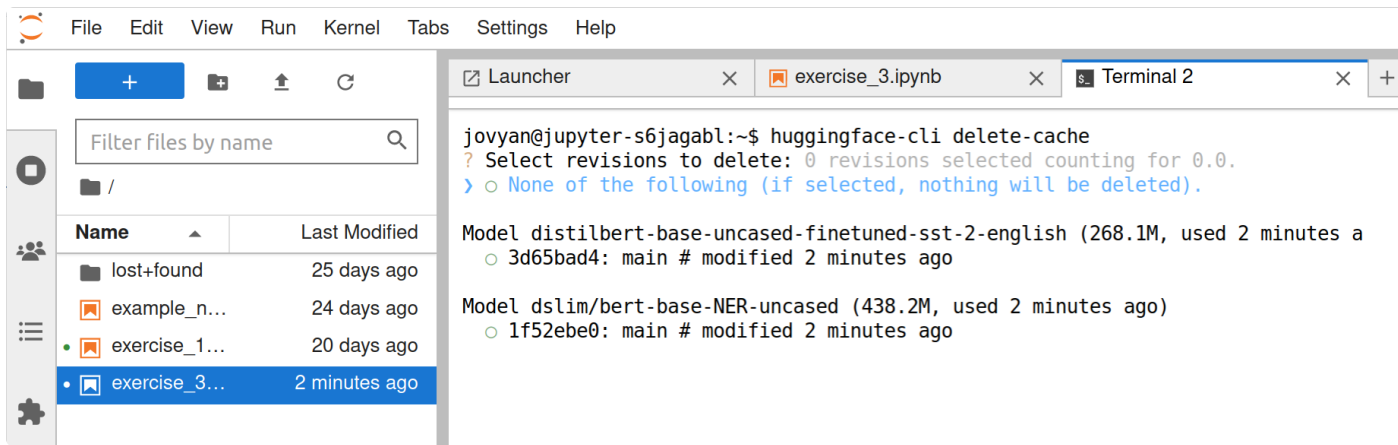
If you forgot to activate the environment

```
~  
> huggingface-cli scan-cache  
huggingface-cli: command not found
```

```
~  
> □
```

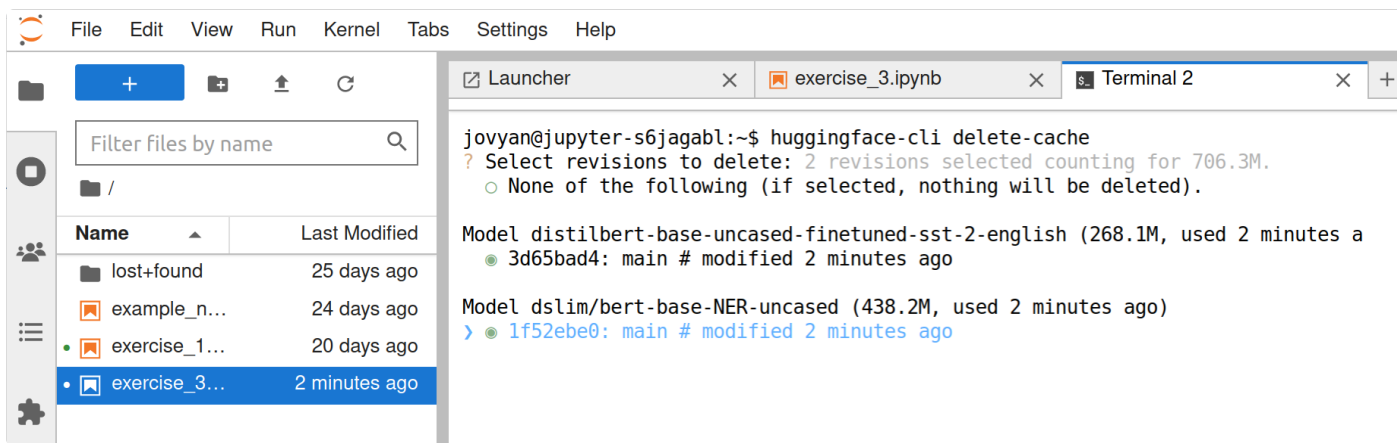
- Read the installation guide for more information

Deleting the cache: command



- Use the arrow keys to move up and down
- Use the space key to toggle a selection

Deleting the cache: selection



The screenshot shows the JupyterLab interface. On the left is the file browser with a search bar and a list of files. The file 'exercise_3...' is selected. On the right is a terminal window showing the command 'huggingface-cli delete-cache' and its output, which lists two models for deletion.

File browser contents:

Name	Last Modified
lost+found	25 days ago
example_n...	24 days ago
exercise_1...	20 days ago
exercise_3...	2 minutes ago

Terminal output:

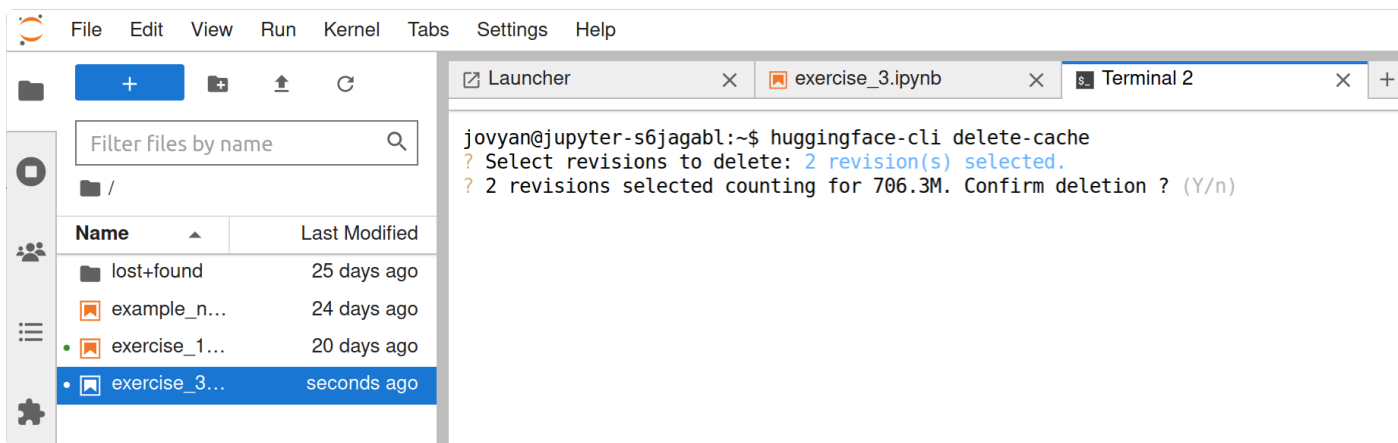
```
jovyan@jupyter-s6jagabl:~$ huggingface-cli delete-cache
? Select revisions to delete: 2 revisions selected counting for 706.3M.
  ○ None of the following (if selected, nothing will be deleted).

Model distilbert-base-uncased-finetuned-sst-2-english (268.1M, used 2 minutes a
  ● 3d65bad4: main # modified 2 minutes ago

Model dslim/bert-base-NER-uncased (438.2M, used 2 minutes ago)
> ● 1f52ebe0: main # modified 2 minutes ago
```

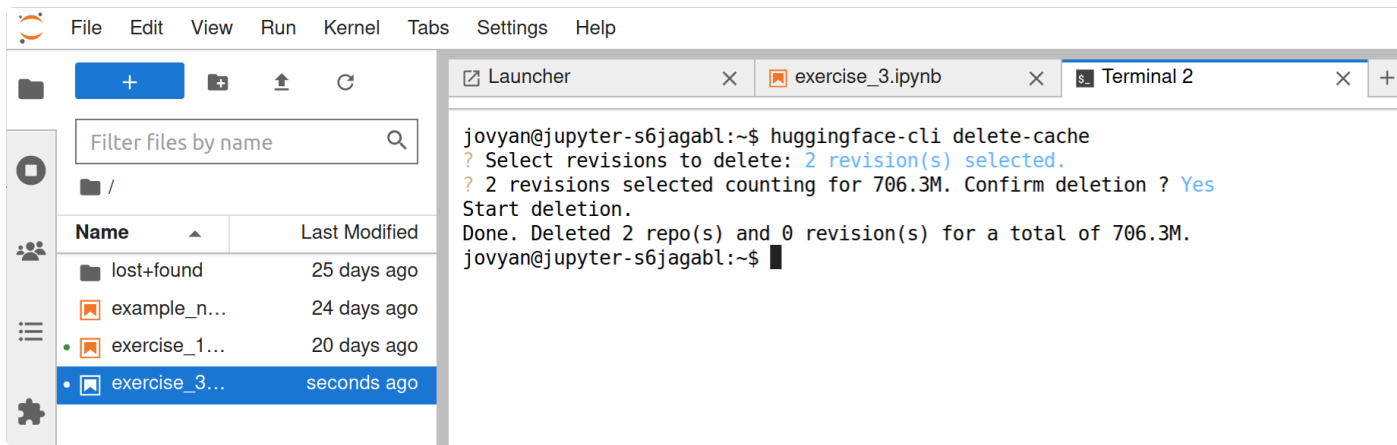
- press enter

Deleting the cache: confirmation



- confirm by typing y

Deleting the cache: done



The screenshot shows the JupyterLab interface. On the left is a file browser with a search bar and a table of files. On the right is a terminal window showing the execution of a command to delete the cache.

File Browser:

Name	Last Modified
lost+found	25 days ago
example_n...	24 days ago
exercise_1...	20 days ago
exercise_3...	seconds ago

Terminal 2:

```
jovyan@jupyter-s6jagabl:~$ huggingface-cli delete-cache
? Select revisions to delete: 2 revision(s) selected.
? 2 revisions selected counting for 706.3M. Confirm deletion ? Yes
Start deletion.
Done. Deleted 2 repo(s) and 0 revision(s) for a total of 706.3M.
jovyan@jupyter-s6jagabl:~$
```

Task 4

7 min

Question answering

```
>>> reader = pipeline(task="question-answering")
>>> question = "What does the customer want?"
>>> answers = reader(question=question, context=text)
>>> answers
{'score': 0.6312923431396484,
 'start': 335,
 'end': 358,
 'answer': 'an exchange of Megatron'}
```

- The task key is `"question-answering"`
- Define your question(s) as a string (or list of strings)
- Pass your question and the text as a context to the pipeline
- The output is a list of dictionaries with answers to questions

Task 5

12 min

Summarization

```
>>> summarizer = pipeline(task="summarization")
>>> summaries = summarizer(
...     text,
...     max_length=45,
...     min_length=45,
...     clean_up_tokenization_spaces=True
... )
>>> summaries
[{"summary_text": "Amazon's Optimus Prime action figure\ Wassent to a German online store in Germany. Bumblebee\ asks for an exchange of Megatron for the Optimus Prime\ figure he ordered. He expects to hear from you soon."}]
```

- The task key is `"summarization"`
- Summarize the text using additional arguments
- The output is a list of dictionaries with short summaries of the input text

Splitting and combining strings

```
zen = """The Zen of Python, by Tim Peters
```

```
Beautiful is better than ugly.  
Explicit is better than implicit.
```

```
Simple is better than complex.  
Complex is better than complicated."""
```

```
>>> parts = zen.split("\n\n")
```

```
>>> parts
```

```
['The Zen of Python, by Tim Peters',  
 'Beautiful is better than ugly.\nExplicit is better than  
 'Simple is better than complex.\nComplex is better than c
```

```
>>> print("\n---\n".join(parts))
```

```
The Zen of Python, by Tim Peters
```

```
---
```

```
Beautiful is better than ugly.  
Explicit is better than implicit.
```

```
---
```


```
Simple is better than complex.  
Complex is better than complicated.
```


- New lines are represented as ``\\n`` in python strings
- ``.split`` can split a string into a list of strings
- using ``"\n\n"`` as delimiter splits by paragraphs
- ``.join`` is the inverse of split

Task 7

(7 min)

https://huggingface.co/models

 **Hugging Face**

[Models](#) [Datasets](#) [Spaces](#) [Docs](#) [Solutions](#) [Pricing](#) [⌵](#) 

Tasks

Libraries

Datasets

Languages

Licenses

Other

Multimodal

Computer Vision

Natural Language Processing

Audio

Tabular

Reinforcement Learning

Feature Extraction

Text-to-Image

Image-to-Text

Text-to-Video

Visual Question Answering

Document Question Answering

Graph Machine Learning

Depth Estimation

Image Classification

Object Detection

Image Segmentation

Image-to-Image

Unconditional Image Generation

Video Classification

Zero-Shot Image Classification

Text Classification

Token Classification

Table Question Answering

Question Answering

Zero-Shot Classification

Translation

Summarization

Conversational

Text Generation

Text2Text Generation

Fill-Mask

Sentence Similarity

Text-to-Speech

Automatic Speech Recognition

Audio-to-Audio

Audio Classification

Voice Activity Detection

Tabular Classification

Tabular Regression

Reinforcement Learning

Robotics

Models 183,501

Filter by name

Full-text search

Sort: Most Downloads

bert-base-uncased

Updated Nov 16, 2022 · 42.4M · 744

jonatasgrosman/wav2vec2-large-xlsr-53-english

Updated 29 days ago · 40.4M · 85

Davlan/distilbert-base-multilingual-cased-nor-hrl

Updated Jun 27, 2022 · 29.9M · 40

gpt2

Updated Dec 16, 2022 · 18.9M · 928

xlm-roberta-base

Updated 16 days ago · 15.3M · 253

openai/clip-vit-large-patch14

Updated Oct 4, 2022 · 8.62M · 347

microsoft/layoutlmv3-base

Updated 11 days ago · 7.63M · 131

distilbert-base-uncased

Updated Nov 16, 2022 · 7.37M · 176

roberta-base

Updated Mar 6 · 7.12M · 152

t5-base

Updated 17 days ago · 5.59M · 198

openai/clip-vit-base-patch32

Updated Oct 4, 2022 · 5.58M · 166

distilroberta-base

Updated Nov 17, 2022 · 5.35M · 62

xlm-roberta-large

Updated 17 days ago · 5.11M · 131

bert-base-cased

Updated Nov 16, 2022 · 4.64M · 95

bert-base-multilingual-cased

Updated Nov 17, 2022 · 3.96M · 141

prajjwalibert-tiny

Updated Oct 27, 2021 · 3.14M · 52

albert-base-v2

Updated 5 days ago · 2.94M · 49

princeton-nlp/unsup-simcse-roberta-base

Updated Jun 16, 2021 · 2.88M · 8

roberta-large

Updated Mar 22 · 2.27M · 101

runwayml/stable-diffusion-v1-5

Updated Jan 27 · 2.17M · 7.19K

Helsinki-NLP/opus-mt-en-es

Updated Jan 24 · 2.11M · 37

cardiffnlp/twitter-roberta-base-sentiment

Updated Jan 20 · 2.11M · 158


google/electra-base-discriminator

Updated Apr 30, 2021 · 2.06M · 21

pyannote/segmentation

Updated 24 days ago · 1.91M · 131

Use filters to find models

 **Hugging Face**

Models Datasets Spaces Docs Solutions Pricing

Tasks Libraries Datasets Languages Licenses Other

Multimodal

Feature Extraction

Text-to-Image

Image-to-Text

Text-to-Video

Visual Question Answering

Document Question Answering

Graph Machine Learning

Computer Vision

Depth Estimation

Image Classification

Object Detection

Image Segmentation

Image-to-Image

Unconditional Image Generation

Video Classification

Zero-Shot Image Classification

Natural Language Processing

Text Classification

Token Classification

Table Question Answering

Question Answering

Zero-Shot Classification

Translation

Summarization

Conversational

Text Generation

Text2Text Generation

Fill-Mask

Sentence Similarity

Models 4,372

Full-text search

deepset/roberta-base-squad2

Updated 30 days ago · 1.31M · 272

distilbert-base-uncased-distilled-squad

Updated 17 days ago · 351k · 26

sultan/BioM-ELECTRA-Large-SQuAD2

Updated Aug 7, 2021 · 247k · 6

deepset/minilm-uncased-squad2

Updated Dec 5, 2022 · 106k · 29

deepset/roberta-base-squad2-covid

Updated Nov 18, 2022 · 87k · 5

monologg/koelectra-small-v2-distilled-korquad-384

Updated Jun 4, 2020 · 64.7k · 1

thatdramebaazguy/roberta-base-squad

Updated Jul 1, 2022 · 50.1k · 1

philschmid/distilbert-onnx

Updated Feb 16, 2023 · 46k · 3

bert-large-uncased-whole-word-masking-finetuned-squad

Updated 17 days ago · 612k · 81

distilbert-base-cased-distilled-squad

Updated 11 days ago · 304k · 112

deepset/bert-large-uncased-whole-word-masking-squad

Updated Dec 5, 2022 · 148k · 17

Rakib/roberta-base-on-cuad

Updated Jan 18 · 90.8k · 2

valhalla/longformer-base-4096-finetuned-squadv1

Updated Feb 10, 2021 · 86.1k · 10

deepset/roberta-large-squad2

Updated Jul 25, 2022 · 59.1k · 14


deepset/tinyroberta-squad2



Updated 30 days ago · 48.9k · 24



deepset/bert-base-cased-squad2

Updated Feb 10, 2021 · 48.9k · 24

Check model-card for more information


 **Hugging Face**

[Models](#) [Datasets](#) [Spaces](#) [Docs](#) [Solutions](#) [Pricing](#)  

deepset/**roberta-base-squad2**   like 272

[Question Answering](#) [PyTorch](#) [TensorFlow](#) [JAX](#) [Rust](#) [Safetensors](#) [Transformers](#) [squad_v2](#) [English](#) [roberta](#) [Eval Results](#) [AutoTrain Compatible](#) [License: cc-by-4.0](#)

[Model card](#) [Files and versions](#) [Community 12](#)

 [Train](#) [Deploy](#) [Use in Transformers](#)

[Edit model card](#)

roberta-base for QA

This is the [roberta-base](#) model, fine-tuned using the [SQuAD2.0](#) dataset. It's been trained on question-answer pairs, including unanswerable questions, for the task of Question Answering.


Overview



Language model: roberta-base
Language: English
Downstream-task: Extractive QA
Training data: SQuAD 2.0
Eval data: SQuAD 2.0
Code: See [an example QA pipeline on Haystack](#)
Infrastructure: 4x Tesla v100



Hyperparameters

```
batch_size = 96
n_epochs = 2
base_LM_model = "roberta-base"
max_seq_len = 386
learning_rate = 3e-5
lr_schedule = LinearWarmup
warmup_proportion = 0.2
doc_stride=128
max_query_length=64
```

Downloads last month
1,309,088



 **Hosted inference API** 

 Question Answering [Examples](#) 


[Compute](#)

Context


Computation time on Intel Xeon 3rd Gen Scalable cpu: cached

Berlin



0.919



 JSON Output [Maximize](#)



Dataset used to train deepset/roberta-base-squad2



 **squad_v2**
[Preview](#) · Updated 18 days ago · ↓ 42.9k · ♥ 27

Spaces using deepset/roberta-base-squad2 194

 microsoft/HuggingGPT  seduer/semantic_search

 anakin87/who-killed-laura-palmer  AmazonScience/QA-NLU

 european-agency/play-with-transformer  razakhan/text-summarizer

 Hellisotheroeoole/HF-SHAP  Femansleendevrived/Study For Me AI

Translation

```
>>> translator = pipeline(task="translation_en_to_de")
>>> translations = translator(
...     text,
...     clean_up_tokenization_spaces=True,
...     min_length=50
... )
>>> translations
[{"translation_text": "Sehr geehrter Amazon,\nletzte Woche habe ich eine Optimus Prime Action\nFigur aus Ihrem Online-Shop in Deutschland bestellt.\nLeider, als ich das Paket öffnete, entdeckte ich zu\meinem Entsetzen, dass ich stattdessen eine Action\nFigur von Megatron geschickt worden war! Als\nlebenslanger Feind der Decepticons, Ich hoffe,\nSie können mein Dilemma verstehen. Um das Problem\nzu lösen, Ich fordere einen Austausch von Megatron\nfür die Optimus Prime Figur habe ich bestellt."}]
```

- The task key is `"translation_xx_to_yy"`
- Translate your text using additional arguments
- The output is a list of dictionaries with the translated text

Task 8

10 min