

# textpipeline

February 21, 2026

```
[ ]: !pip install datasets evaluate transformers[sentencepiece]
```

```
[1]: from transformers import pipeline
```

**Zero-shot-classification:** Classify text without prior training on specific labels

```
[5]: classifier=pipeline('zero-shot-classification')
classifier(
    'This is a course about the Transformers library',
    candidate_labels=['education', 'politics', 'business'],
)
```

No model was supplied, defaulted to facebook/bart-large-mnli and revision d7645e1.

Using a pipeline without specifying a model name and revision in production is not recommended.

/usr/local/lib/python3.12/dist-packages/huggingface\_hub/utils/\_auth.py:94:

UserWarning:

The secret `HF\_TOKEN` does not exist in your Colab secrets.

To authenticate with the Hugging Face Hub, create a token in your settings tab (<https://huggingface.co/settings/tokens>), set it as secret in your Google Colab and restart your session.

You will be able to reuse this secret in all of your notebooks.

Please note that authentication is recommended but still optional to access public models or datasets.

```
    warnings.warn(
```

```
config.json: 0.00B [00:00, ?B/s]
```

```
model.safetensors:  0% | 0.00/1.63G [00:00<?, ?B/s]
```

```
Loading weights:  0% | 0/515 [00:00<?, ?it/s]
```

Warning: You are sending unauthenticated requests to the HF Hub. Please set a HF\_TOKEN to enable higher rate limits and faster downloads.

WARNING:huggingface\_hub.utils.\_http:Warning: You are sending unauthenticated requests to the HF Hub. Please set a HF\_TOKEN to enable higher rate limits and faster downloads.

```
tokenizer_config.json:  0% | 0.00/26.0 [00:00<?, ?B/s]
```

```
vocab.json: 0.00B [00:00, ?B/s]
merges.txt: 0.00B [00:00, ?B/s]
tokenizer.json: 0.00B [00:00, ?B/s]

[5]: {'sequence': 'This is a course about the Transformers library',
      'labels': ['education', 'business', 'politics'],
      'scores': [0.8445985913276672, 0.11197450757026672, 0.04342687129974365]}
```

```
[9]: generator=pipeline('text-generation')
generator('What is machine learning !')
```

No model was supplied, defaulted to openai-community/gpt2 and revision 607a30d. Using a pipeline without specifying a model name and revision in production is not recommended.

```
Loading weights: 0% | 0/148 [00:00<?, ?it/s]
GPT2LMHeadModel LOAD REPORT from: openai-community/gpt2
Key           | Status    | |
-----+-----+---+
h.{0...11}.attn.bias | UNEXPECTED | |
```

Notes:

- UNEXPECTED : can be ignored when loading from different task/architecture; not ok if you expect identical arch.

Setting `pad\_token\_id` to `eos\_token\_id`:50256 for open-end generation.  
Both `max\_new\_tokens` (=256) and `max\_length`(=50) seem to have been set.  
`max\_new\_tokens` will take precedence. Please refer to the documentation for more information.

([https://huggingface.co/docs/transformers/main/en/main\\_classes/text\\_generation](https://huggingface.co/docs/transformers/main/en/main_classes/text_generation))

```
[9]: [{'generated_text': 'What is machine learning !?\n\nMachine learning is a system of algorithms that are designed to perform tasks in a logical order and are then trained to perform those tasks in a logical manner. This is a common way to train a computer to perform certain tasks in a logical way.\n\nThe good news is that we can improve the way we build our algorithms by improving the way we train them, but the bad news is that we are very reliant on AI.\n\nThis is why it was important to create a system where we could train the computer to perform certain tasks in a logical way. By doing so, we allow our AI to perform tasks in a logical manner, but by allowing our machine to perform those tasks in a logical way, we are allowing our AI to perform certain tasks in a bad way.\n\nIn order to ensure that our systems are able to do these tasks efficiently, we need to learn how to learn how to do them correctly.\n\nThis is where machine learning comes in.\n\nMachine learning is a kind of machine learning that is designed to learn and learn new things.\n\nIn order to find more ways to learn new things, we need to learn more about new things.\n\nThis is where machine learning comes in.\n\nMachine learning is'}]
```

```
[10]: generator = pipeline("text-generation", model="HuggingFaceTB/SmolLM2-360M")
generator(
    "In this course, we will teach you how to",
    max_length=30,
    num_return_sequences=2,
)
```

```
config.json: 0% | 0.00/689 [00:00<?, ?B/s]
model.safetensors: 0% | 0.00/724M [00:00<?, ?B/s]
Loading weights: 0% | 0/290 [00:00<?, ?it/s]
generation_config.json: 0% | 0.00/111 [00:00<?, ?B/s]
tokenizer_config.json: 0.00B [00:00, ?B/s]
tokenizer.json: 0.00B [00:00, ?B/s]
special_tokens_map.json: 0% | 0.00/831 [00:00<?, ?B/s]

Passing `generation_config` together with generation-related
arguments=({'num_return_sequences', 'max_length'}) is deprecated and will be
removed in future versions. Please pass either a `generation_config` object OR
all generation parameters explicitly, but not both.
Setting `pad_token_id` to `eos_token_id`:0 for open-end generation.
Both `max_new_tokens` (=256) and `max_length` (=30) seem to have been set.
`max_new_tokens` will take precedence. Please refer to the documentation for
more information.
(https://huggingface.co/docs/transformers/main/en/main\_classes/text\_generation)
```

```
[10]: [{'generated_text': 'In this course, we will teach you how to use this tool to
create a range of useful and highly effective data visualizations. For instance,
you can create heatmaps with different colors to represent different values in a
dataset. You can also use this tool to create graphs that show the correlation
between different variables. You can use this tool to create bar charts, pie
charts, and line charts. This tool can help you create a range of useful and
highly effective data visualizations.\n\n## What to watch out for?\n\nWhen
working with data, it is important to understand how to interpret the data
correctly. For instance, if you are working with data that contains a large
number of outliers, it is important to understand how to interpret the outliers
correctly.\n\n## Conclusion\n\nData visualization is a powerful tool that can
help you to better understand your data. In this blog post, we have taught you
how to use this tool to create data visualizations. We have also explained how
to interpret data and how to use the data visualization tools that are provided
with the tool. We have also explained how to use the data visualization tools
that are provided with the tool. We hope that this has helped you to better
understand how to use the data visualization tools that are provided with the
tool.'},
 {'generated_text': 'In this course, we will teach you how to read a document in
a language other than your native tongue and write in the correct way. You will'}
```

learn how to deal with accents and spelling mistakes in a foreign language. We will show you how to translate between different languages and how to explain your ideas in another language.\n\n\nWhat will I learn?\n\n\n • Understand how to make a language translation\n • Use a dictionary to translate between different languages\n • Understand how to explain an idea in a foreign language\n • Use a word processor to write in a foreign language\n\nWe will show you how to do all this by using a real life example.\n\nCourse requirements\nTo complete the course, you will need to have the following:\n\n • An open mind. We want to show you how to see things differently.\n • A computer with internet access. This will enable us to write in a foreign language.\n • A dictionary or a translation app.\n • A notebook in which you will write down your ideas, and a pen.\n • One hour per week to study.\n\nWho is this course for?\nThis course is for anyone who wants to learn a foreign language. You do not need to have any knowledge of other languages to join, but if you do, we']

**Mask filling** is an NLP task where a model predicts missing or hidden words in a sentence.

```
[18]: from transformers import pipeline

unmasker = pipeline("fill-mask")
unmasker("This course will teach you all about <mask> models.", top_k=3)
```

No model was supplied, defaulted to distilbert/distilroberta-base and revision fb53ab8.

Using a pipeline without specifying a model name and revision in production is not recommended.

Loading weights: 0% | 0/106 [00:00<?, ?it/s]

| RobertaForMaskedLM LOAD REPORT from: distilbert/distilroberta-base |            |  |
|--|------------|--|
| Key  | Status     |  |
| roberta.pooler.dense.weight  | UNEXPECTED |  |
| roberta.pooler.dense.bias  | UNEXPECTED |  |

Notes:

- UNEXPECTED : can be ignored when loading from different task/architecture; not ok if you expect identical arch.

```
[18]: [ {'score': 0.19619713723659515,
  'token': 30412,
  'token_str': ' mathematical',
  'sequence': 'This course will teach you all about mathematical models.'},
  {'score': 0.040527280420064926,
  'token': 38163,
  'token_str': ' computational',
  'sequence': 'This course will teach you all about computational models.'},
```

```
{'score': 0.03301792964339256,
'token': 27930,
'token_str': ' predictive',
'sequence': 'This course will teach you all about predictive models.']}
```

## 1 Named entity recognition

Named entity recognition (NER) is a task where the model has to find which parts of the input text correspond to entities such as persons, locations, or organizations. Let's look at an example:

```
[25]: ner = pipeline("ner")
ner("My name is maheshkannemadugu and I work at Telstra in Bengaluru.")
```

No model was supplied, defaulted to dbmdz/bert-large-cased-finetuned-conll03-english and revision 4c53496.

Using a pipeline without specifying a model name and revision in production is not recommended.

Loading weights: 0% | 0/391 [00:00<?, ?it/s]

BertForTokenClassification LOAD REPORT from: dbmdz/bert-large-cased-finetuned-conll03-english

| Key                      | Status     |  |  |
|--------------------------|------------|--|--|
| bert.pooler.dense.weight | UNEXPECTED |  |  |
| bert.pooler.dense.bias   | UNEXPECTED |  |  |

Notes:

- UNEXPECTED : can be ignored when loading from different task/architecture; not ok if you expect identical arch.

```
[25]: [{"entity': 'I-PER',
'score': np.float32(0.974722),
'index': 4,
'word': 'ma',
'start': 11,
'end': 13},
{'entity': 'I-PER',
'score': np.float32(0.9742966),
'index': 5,
'word': '##hes',
'start': 13,
'end': 16},
{'entity': 'I-PER',
'score': np.float32(0.98616344),
'index': 6,
'word': '##h',
'start': 16,
```

```
'end': 17},
{'entity': 'I-PER',
 'score': np.float32(0.98592687),
 'index': 7,
 'word': '##kan',
 'start': 17,
 'end': 20},
{'entity': 'I-PER',
 'score': np.float32(0.97786033),
 'index': 8,
 'word': '##ne',
 'start': 20,
 'end': 22},
{'entity': 'I-PER',
 'score': np.float32(0.8874486),
 'index': 9,
 'word': '##mad',
 'start': 22,
 'end': 25},
{'entity': 'I-PER',
 'score': np.float32(0.6474023),
 'index': 10,
 'word': '##ug',
 'start': 25,
 'end': 27},
{'entity': 'I-PER',
 'score': np.float32(0.6504101),
 'index': 11,
 'word': '##u',
 'start': 27,
 'end': 28},
{'entity': 'I-ORG',
 'score': np.float32(0.99937266),
 'index': 16,
 'word': 'Tel',
 'start': 43,
 'end': 46},
{'entity': 'I-ORG',
 'score': np.float32(0.99901116),
 'index': 17,
 'word': '##stra',
 'start': 46,
 'end': 50},
{'entity': 'I-LOC',
 'score': np.float32(0.99555063),
 'index': 19,
 'word': 'Ben',
```

```

'start': 54,
'end': 57},
{'entity': 'I-LOC',
'score': np.float32(0.95189714),
'index': 20,
'word': '##gu',
'start': 57,
'end': 59},
{'entity': 'I-LOC',
'score': np.float32(0.94492304),
'index': 21,
'word': '##lu',
'start': 59,
'end': 61},
{'entity': 'I-LOC',
'score': np.float32(0.9900241),
'index': 22,
'word': '##ru',
'start': 61,
'end': 63}]

```

## 2 Question answering

```
[28]: question_answerer = pipeline("question-answering")
question_answerer(
    question="Where is the Eiffel Tower located?",
    context="The Eiffel Tower is located in Paris and was built in 1889.",
)
```

No model was supplied, defaulted to distilbert/distilbert-base-cased-distilled-squad and revision 564e9b5.

Using a pipeline without specifying a model name and revision in production is not recommended.

Loading weights: 0% | 0/102 [00:00<?, ?it/s]

```
[28]: {'score': 0.9887961149215698, 'start': 31, 'end': 36, 'answer': 'Paris'}
```

```
[35]: summarizer = pipeline("text-generation", model="google/flan-t5-base")

text = """summarize: Artificial Intelligence (AI) refers to the simulation of ↵
human intelligence in machines..."""

result = summarizer(text)

print(result)
```

```

model.safetensors: 0% | 0.00/990M [00:00<?, ?B/s]
Loading weights: 0% | 0/282 [00:00<?, ?it/s]

The tied weights mapping and config for this model specifies to tie
shared.weight to lm_head.weight, but both are present in the checkpoints, so we
will NOT tie them. You should update the config with `tie_word_embeddings=False`
to silence this warning

generation_config.json: 0% | 0.00/147 [00:00<?, ?B/s]
tokenizer_config.json: 0.00B [00:00, ?B/s]
spiece.model: 0% | 0.00/792k [00:00<?, ?B/s]
tokenizer.json: 0.00B [00:00, ?B/s]
special_tokens_map.json: 0.00B [00:00, ?B/s]

The model 'T5ForConditionalGeneration' is not supported for text-generation.
Supported models are ['PeftModelForCausallM', 'AfmoeForCausallM',
'ApertusForCausallM', 'ArceeForCausallM', 'AriaTextForCausallM',
'BambaForCausallM', 'BartForCausallM', 'BertLMHeadModel',
'BertGenerationDecoder', 'BigBirdForCausallM', 'BigBirdPegasusForCausallM',
'BioGptForCausallM', 'BitNetForCausallM', 'BlenderbotForCausallM',
'BlenderbotSmallForCausallM', 'BloomForCausallM', 'BltForCausallM',
'CamembertForCausallM', 'LlamaForCausallM', 'CodeGenForCausallM',
'CohereForCausallM', 'Cohere2ForCausallM', 'CpmAntForCausallM',
'CTRLLMHeadModel', 'CwmForCausallM', 'Data2VecTextForCausallM',
'DbrxForCausallM', 'DeepseekV2ForCausallM', 'DeepseekV3ForCausallM',
'DiffLlamaForCausallM', 'DogeForCausallM', 'Dots1ForCausallM',
'ElectraForCausallM', 'Emu3ForCausallM', 'ErnieForCausallM',
'Ernie4_5ForCausallM', 'Ernie4_5_MoeForCausallM', 'Exaone4ForCausallM',
'FalconForCausallM', 'FalconH1ForCausallM', 'FalconMambaForCausallM',
'FlexOlmoForCausallM', 'FuyuForCausallM', 'GemmaForCausallM',
'Gemma2ForCausallM', 'Gemma3ForConditionalGeneration', 'Gemma3ForCausallM',
'Gemma3nForConditionalGeneration', 'Gemma3nForCausallM', 'GitForCausallM',
'GlmForCausallM', 'Glm4ForCausallM', 'Glm4MoeForCausallM',
'Glm4MoeLiteForCausallM', 'GotOcr2ForConditionalGeneration', 'GPT2LMHeadModel',
'GPT2LMHeadModel', 'GPTBigCodeForCausallM', 'GPTNeoForCausallM',
'GPTNeoXForCausallM', 'GPTNeoXJapaneseForCausallM', 'GptOssForCausallM',
'GPTJForCausallM', 'GraniteForCausallM', 'GraniteMoeForCausallM',
'GraniteMoeHybridForCausallM', 'GraniteMoeSharedForCausallM',
'HeliumForCausallM', 'HunYuanDenseV1ForCausallM', 'HunYuanMoEV1ForCausallM',
'Jais2ForCausallM', 'JambaForCausallM', 'JetMoeForCausallM', 'Lfm2ForCausallM',
'Lfm2MoeForCausallM', 'LlamaForCausallM', 'Llama4ForCausallM',
'Llama4ForCausallM', 'LongcatFlashForCausallM', 'MambaForCausallM',
'Mamba2ForCausallM', 'MarianForCausallM', 'MBartForCausallM',
'MegatronBertForCausallM', 'MiniMaxForCausallM', 'MiniMaxM2ForCausallM',
'MinistralfForCausallM', 'Ministralf3ForCausallM', 'MistralForCausallM',
'MixtralForCausallM', 'MllamaForCausallM', 'ModernBertDecoderForCausallM',
'MoshiForCausallM', 'MptForCausallM', 'MusicgenForCausallM',

```

```
'MusicgenMelodyForCausallM', 'MvpForCausallM', 'NanoChatForCausallM',
'NemotronForCausallM', 'OlmoForCausallM', 'Olmo2ForCausallM',
'Olmo3ForCausallM', 'OlmoeForCausallM', 'OpenAIGPTLMHeadModel',
'OPTForCausallM', 'PegasusForCausallM', 'PersimmonForCausallM',
'PhiForCausallM', 'Phi3ForCausallM', 'Phi4MultimodalForCausallM',
'PhimoeForCausallM', 'PLBartForCausallM', 'ProphetNetForCausallM',
'Qwen2ForCausallM', 'Qwen2MoeForCausallM', 'Qwen3ForCausallM',
'Qwen3MoeForCausallM', 'Qwen3NextForCausallM', 'RecurrentGemmaForCausallM',
'ReformerModelWithLMHead', 'RemBertForCausallM', 'RobertaForCausallM',
'RobertaPreLayerNormForCausallM', 'RoCBertForCausallM', 'RoFormerForCausallM',
'RwkvForCausallM', 'SeedOssForCausallM', 'SmolLM3ForCausallM',
'SolarOpenForCausallM', 'StableLmForCausallM', 'Starcoder2ForCausallM',
'TrOCRForCausallM', 'VaultGemmaForCausallM', 'WhisperForCausallM',
'XGLMForCausallM', 'XLMWithLMHeadModel', 'XLMRobertaForCausallM',
'XLMRobertaXLForCausallM', 'XLNetLMHeadModel', 'xLSTMForCausallM',
'XmodForCausallM', 'ZambaForCausallM', 'Zamba2ForCausallM'].

Both `max_new_tokens` (=256) and `max_length` (=20) seem to have been set.
`max_new_tokens` will take precedence. Please refer to the documentation for
more information.
```

([https://huggingface.co/docs/transformers/main/en/main\\_classes/text\\_generation](https://huggingface.co/docs/transformers/main/en/main_classes/text_generation))

```
[{'generated_text': 'summarize: Artificial Intelligence (AI) refers to the
simulation of human intelligence in
machines...logiecshare...iol=inpjware%25v5wd71'}]
```

```
[ ]: from transformers import pipeline

translator = pipeline(
    "text-to-text",
    model="google/flan-t5-base"
)

translator("Translate French to English: Ce cours est produit par Hugging Face.
˓→")
```

### 3 Image and audio pipelines

```
[39]: image_classifier = pipeline(
    task="image-classification", model="google/vit-base-patch16-224"
)
result = image_classifier(
    "https://huggingface.co/datasets/huggingface/documentation-images/resolve/
˓→main/pipeline-cat-chonk.jpeg"
)
print(result)
```

Loading weights: 0% | 0/200 [00:00<?, ?it/s]

```
Fast image processor class <class  
'transformers.models.vit.image_processing_vit_fast.ViTImageProcessorFast'> is  
available for this model. Using slow image processor class. To use the fast  
image processor class set `use_fast=True`.
```

```
[{'label': 'lynx, catamount', 'score': 0.4334994852542877}, {'label': 'cougar,  
puma, catamount, mountain lion, painter, panther, Felis concolor', 'score':  
0.03479618951678276}, {'label': 'snow leopard, ounce, Panthera uncia', 'score':  
0.03240194171667099}, {'label': 'Egyptian cat', 'score': 0.023944808170199394},  
{'label': 'tiger cat', 'score': 0.022889263927936554}]
```

```
[3]: from transformers import pipeline  
  
transcriber = pipeline(  
    task="automatic-speech-recognition", model="openai/whisper-large-v3"  
)  
result = transcriber(  
    "https://huggingface.co/datasets/Narsil/asr_dummy/resolve/main/mlk.flac"  
)  
print(result)
```

```
Loading weights: 0%| 0/1259 [00:00<?, ?it/s]
```

```
{'text': ' I have a dream that one day this nation will rise up and live out the  
true meaning of its creed.'}
```