

```
# Transformers installation
! pip install transformers
# To install from source instead of the last release, comment the command above and uncomment the following one.
# ! pip install git+https://github.com/huggingface/transformers.git
```

```
Collecting transformers
  Downloading https://files.pythonhosted.org/packages/b0/9e/5b80becd952d5f7250eaf8fc64b957077b12ccfe73e9c03d37146ab29712/transformers-4.1.0-py3-none-any.whl (2.3MB)
  2.3MB 13.4MB/s
Collecting tokenizers<0.11,>=0.10.1
  Downloading https://files.pythonhosted.org/packages/ae/04/5b870f26a858552025a62f1649c20d29d2672c02ff3c3fb4c688ca46467a/tokenizers-0.10.1-cp39-cp39-macosx_10_9_universal2.whl (3.3MB)
  3.3MB 53.2MB/s
Requirement already satisfied: regex!=2019.12.17 in /usr/local/lib/python3.7/dist-packages (from transformers) (2019.12.20)
Requirement already satisfied: tqdm>=4.27 in /usr/local/lib/python3.7/dist-packages (from transformers) (4.41.1)
Collecting huggingface-hub==0.0.8
  Downloading https://files.pythonhosted.org/packages/a1/88/7b1e45720ecf59c6c6737ff332f41c955963090a18e72acbcbeac6b25e86/huggingface_hub-0.0.8-py3-none-any.whl (901kB)
  901kB 68.5MB/s
Requirement already satisfied: importlib-metadata; python_version < "3.8" in /usr/local/lib/python3.7/dist-packages (from transformers) (2.0.9)
Requirement already satisfied: packaging in /usr/local/lib/python3.7/dist-packages (from transformers) (20.9)
Requirement already satisfied: requests in /usr/local/lib/python3.7/dist-packages (from transformers) (2.23.0)
Collecting sacremoses
  Downloading https://files.pythonhosted.org/packages/75/ee/67241dc87f266093c533a2d4d3d69438e57d7a90abb216fa076e7d475d4a/sacremoses-0.0.45-py3-none-any.whl (901kB)
  901kB 68.5MB/s
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.7/dist-packages (from transformers) (1.19.5)
Requirement already satisfied: filelock in /usr/local/lib/python3.7/dist-packages (from transformers) (3.0.12)
Requirement already satisfied: typing-extensions>=3.6.4; python_version < "3.8" in /usr/local/lib/python3.7/dist-packages (from transformers) (3.7.4)
Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.7/dist-packages (from transformers) (0.5.2)
Requirement already satisfied: pyparsing>=2.0.2 in /usr/local/lib/python3.7/dist-packages (from transformers) (2.4.7)
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in /usr/local/lib/python3.7/dist-packages (from transformers) (1.25.1)
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.7/dist-packages (from transformers) (2.10)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.7/dist-packages (from transformers) (2020.12.5)
Requirement already satisfied: chardet<4,>=3.0.2 in /usr/local/lib/python3.7/dist-packages (from transformers) (3.0.4)
Requirement already satisfied: six in /usr/local/lib/python3.7/dist-packages (from transformers) (1.15.0)
Requirement already satisfied: joblib in /usr/local/lib/python3.7/dist-packages (from transformers) (1.0.1)
Requirement already satisfied: click in /usr/local/lib/python3.7/dist-packages (from transformers) (8.0.0)
Installing collected packages: tokenizers, huggingface-hub, sacremoses, transformers
Successfully installed huggingface-hub-0.0.8 sacremoses-0.0.45 tokenizers-0.10.2 transformers-4.1.0
```

```
from transformers import pipeline
classifier = pipeline('sentiment-analysis')
```

```
Downloading: 100% 629/629 [00:00<00:00, 21.4kB/s]

Downloading: 100% 268M/268M [00:10<00:00, 26.5MB/s]

Downloading: 100% 232k/232k [00:01<00:00, 136kB/s]

Downloading: 100% 48.0/48.0 [00:00<00:00, 745B/s]
```

```
classifier('We are very happy to show you the 🤖 Transformers library.')
```

```
[{'label': 'POSITIVE', 'score': 0.9997795224189758}]
```


```
classifier('The pizza is not that great but the crust is awesome.')
```

```
[{'label': 'POSITIVE', 'score': 0.9998460412025452}]
```


```
results = classifier(["We are very happy to show you the 🤖 Transformers library.",
                    "We hope you don't hate it."])
for result in results:
    print(f"label: {result['label']}, with score: {round(result['score'], 4)}")
```

```
label: POSITIVE, with score: 0.9998
label: NEGATIVE, with score: 0.5309
```

```
classifier = pipeline('sentiment-analysis', model="nlpTown/bert-base-multilingual-uncased-sentiment")
```

 Downloading: 100% 953/953 [00:00<00:00, 2.59kB/s]
 Downloading: 100% 669M/669M [00:21<00:00, 31.7MB/s]
 Downloading: 100% 872k/872k [00:02<00:00, 420kB/s]
 Downloading: 100% 112/112 [00:00<00:00, 197B/s]
 Downloading: 100% 39.0/39.0 [00:00<00:00, 182B/s]

```
classifier("Esperamos que no lo odie.")
```


 [{"label": '3 stars', 'score': 0.3368821442127228}]

```
from transformers import AutoTokenizer, TFAutoModelForSequenceClassification
```

```


model_name = "nlpTown/bert-base-multilingual-uncased-sentiment"
# This model only exists in PyTorch, so we use the `from_pt` flag to import that model in TensorFlow.
model = TFAutoModelForSequenceClassification.from_pretrained(model_name, from_pt=True)
tokenizer = AutoTokenizer.from_pretrained(model_name)
classifier = pipeline('sentiment-analysis', model=model, tokenizer=tokenizer)

```

 All PyTorch model weights were used when initializing TFBertForSequenceClassification.

All the weights of TFBertForSequenceClassification were initialized from the PyTorch model.
 If your task is similar to the task the model of the checkpoint was trained on, you can already use TFBertForSequenceClassification for

```
classifier("I am a good boy")
```

 [{"label": '4 stars', 'score': 0.42292696237564087}]

✓ Under the hood: pretrained models

```

from transformers import AutoTokenizer, TFAutoModelForSequenceClassification
model_name = "distilbert-base-uncased-finetuned-sst-2-english"
tf_model = TFAutoModelForSequenceClassification.from_pretrained(model_name)
tokenizer = AutoTokenizer.from_pretrained(model_name)

```

 Downloading: 100% 268M/268M [00:04<00:00, 56.2MB/s]


Some layers from the model checkpoint at distilbert-base-uncased-finetuned-sst-2-english were not used when initializing TFDistilBertForSequenceClassification
 - This IS expected if you are initializing TFDistilBertForSequenceClassification from the checkpoint of a model trained on another task
 - This IS NOT expected if you are initializing TFDistilBertForSequenceClassification from the checkpoint of a model that you expect to
 Some layers of TFDistilBertForSequenceClassification were not initialized from the model checkpoint at distilbert-base-uncased-finetune
 You should probably TRAIN this model on a down-stream task to be able to use it for predictions and inference.

✓ Using the tokenizer

```
inputs = tokenizer("We are very happy to show you the 🤖 Transformers library.")
```

Start [coding](#) or [generate](#) with AI.

```
print(inputs)
```

 {'input_ids': [101, 2057, 2024, 2200, 3407, 2000, 2265, 2017, 1996, 100, 19081, 3075, 1012, 102], 'attention_mask': [1, 1, 1, 1, 1, 1,

```

tf_batch = tokenizer(
    ["We are very happy to show you the 🤖 Transformers library.", "We hope you don't hate it."],
    padding=True,

```

```

truncation=True,
max_length=512,
return_tensors="tf"
)

for key, value in tf_batch.items():
    print(f"{key}: {value.numpy().tolist()}")

↗ input_ids: [[101, 2057, 2024, 2200, 3407, 2000, 2265, 2017, 1996, 100, 19081, 3075, 1012, 102], [101, 2057, 3246, 2017, 2123, 1005, 105
attention_mask: [[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1], [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0]]

```

You can learn more about tokenizers [here](#).

✓ Using the model

```

tf_outputs = tf_model(tf_batch)

print(tf_outputs)

↗ (<tf.Tensor: shape=(2, 2), dtype=float32, numpy=
array([[ -4.0832963 ,  4.336414  ],
       [ 0.08181786, -0.04179301]], dtype=float32)>,)

```

Let's apply the SoftMax activation to get predictions.

```

import tensorflow as tf
tf_predictions = tf.nn.softmax(tf_outputs[0], axis=-1)

```

We can see we get the numbers from before:

```

print(tf_predictions)

↗ tf.Tensor(
[[2.2042994e-04 9.9977952e-01]
 [5.3086340e-01 4.6913657e-01]], shape=(2, 2), dtype=float32)

```

If you have labels, you can provide them to the model, it will return a tuple with the loss and the final activations.

```

import tensorflow as tf
tf_outputs = tf_model(tf_batch, labels = tf.constant([1, 0]))

```

```

tokenizer.save_pretrained(save_directory)
model.save_pretrained(save_directory)

```

```

tokenizer = AutoTokenizer.from_pretrained(save_directory)
model = TFAutoModel.from_pretrained(save_directory, from_pt=True)

```

and if you are loading a saved TensorFlow model in a PyTorch model, you should use the following code:

```

tokenizer = AutoTokenizer.from_pretrained(save_directory)
model = AutoModel.from_pretrained(save_directory, from_tf=True)

```

Lastly, you can also ask the model to return all hidden states and all attention weights if you need them:

```

tf_outputs = tf_model(tf_batch, output_hidden_states=True, output_attentions=True)
all_hidden_states, all_attentions = tf_outputs[-2:]

```

✓ Accessing the code

```

from transformers import DistilBertTokenizer, TFDistilBertForSequenceClassification
model_name = "distilbert-base-uncased-finetuned-sst-2-english"
model = TFDistilBertForSequenceClassification.from_pretrained(model_name)

```

```
tokenizer = DistilBertTokenizer.from_pretrained(model_name)
```

✓ Customizing the model

```
from transformers import DistilBertConfig, DistilBertTokenizer, TFDistilBertForSequenceClassification
config = DistilBertConfig(n_heads=8, dim=512, hidden_dim=4*512)
tokenizer = DistilBertTokenizer.from_pretrained('distilbert-base-uncased')
model = TFDistilBertForSequenceClassification(config)
```

```
from transformers import DistilBertConfig, DistilBertTokenizer, TFDistilBertForSequenceClassification
model_name = "distilbert-base-uncased"
model = TFDistilBertForSequenceClassification.from_pretrained(model_name, num_labels=10)
tokenizer = DistilBertTokenizer.from_pretrained(model_name)
```