datacamp

Working with Hugging Face

Hugging Face Basics

Learn Hugging Face online at www.DataCamp.com

What is Hugging Face?

Hugging Face is an ecosystem for discovering, running, training, and sharing machine learning models and datasets, with a strong emphasis on open-source and reproducibility.

The "core four" libraries are: `transformers` (models + pipelines), `tokenizers` (fast tokenization), `datasets` (data loading/processing), and `huggingface_hub` (Hub interaction + versioning).

> The Hugging Face Hub

The Hub is a Git-backed platform for hosting **Models**, **Datasets**, and **Spaces** (interactive demos), plus Community features for sharing and discovery.

> Key definitions

- A **model** is a pretrained checkpoint; a **tokenizer** converts raw text into tokens; a **pipeline** bundles preprocessing, inference, and postprocessing for a task.
- A **dataset** is an Arrow-backed collection of data with **splits** (train/validation/test).
- A **checkpoint** is a saved snapshot of model weights/config; **inference** means running a trained model on new inputs; a **repo** is a Git-backed Hub unit storing models/datasets/Spaces.

> Model Cards and Dataset Cards

- A **Model Card** explains intended use, training data, evaluation, limitations/biases, and licensing.
- A **Dataset Card** describes data sources, schema/splits, known issues/biases, ethics, and licensing.

Use cards to assess fitness-for-purpose, risk, and reproducibility.

> Where to run inference?

- Run **locally** for control, lower latency, and offline use (you manage hardware/dependencies).
- Use an **inference provider** for fast setup and scalability (trade control for network latency and usage-based costs).

> Workflows - Inference (Transformers)

Quickstart: Run inference with a pipeline

```python
from transformers import pipeline

# Create a pipeline by specifying a task and model ID
analyze_sentiment = pipeline(
   "sentiment-analysis",
   model="distilbert-base-uncased-finetuned-sst-2-english"
)

# Run inference on input text
analyze_sentiment("Hugging Face makes NLP workflows easy!")
```

Text summarization

```python
from transformers import pipeline

# Create a summarization pipeline
summarize_text = pipeline(
   "summarization",
   model="facebook/bart-large-cnn"
)

# Summarize input text
summarize_text("Long document text goes here...")
```

Document question answering

```python
from transformers import pipeline

# Create a document QA pipeline
answer_question = pipeline(
   "document-question-answering",
   model="impira/layoutlm-document-qa"
)

# Ask a question about a document image
answer_question(
   image="invoice.png",
 question="What is the invoice total?"
)
```

Run inference manually

```python
import torch
from transformers import AutoTokenizer, AutoModelForSequenceClassification

model_id = "distilbert-base-uncased-finetuned-sst-2-english"

tokenizer = AutoTokenizer.from_pretrained(model_id)
model = AutoModelForSequenceClassification.from_pretrained(model_id)

inputs = tokenizer("Hugging Face is great", return_tensors="pt")

with torch.no_grad():
   outputs = model(**inputs)

outputs.logits.argmax(dim=-1).item()
```

Learn Data Skills Online at www.DataCamp.com

datacamp

> Data processing workflows (datasets)

Load and slice datasets

```python
from datasets import load_dataset

movie_reviews = load_dataset("imdb")

train_reviews = movie_reviews["train"]
train_reviews[0]

small_sample = train_reviews.select(range(100))
```

Preprocess a dataset

```python
from datasets import load_dataset
from transformers import AutoTokenizer

dataset = load_dataset("imdb")
tokenizer = AutoTokenizer.from_pretrained("distilbert-base-uncased")

def tokenize_batch(batch):
   return tokenizer(
       batch["text"],
       truncation=True,
       padding="max_length",
       max_length=256
   )

tokenized_dataset = dataset.map(
   tokenize_batch,
   batched=True,
   remove_columns=["text"]
)
```

> Working with the Hub (huggingface_hub)

Save locally and reload

```python
from transformers import AutoTokenizer, AutoModelForSequenceClassification

model_id = "distilbert-base-uncased-finetuned-sst-2-english"

tokenizer = AutoTokenizer.from_pretrained(model_id)
model = AutoModelForSequenceClassification.from_pretrained(model_id)

tokenizer.save_pretrained("local_tokenizer")
model.save_pretrained("local_model")

AutoTokenizer.from_pretrained("local_tokenizer")
AutoModelForSequenceClassification.from_pretrained("local_model")
```

Log in to the Hub

```python
from huggingface_hub import login

login()
```

Upload (push) a model to the Hub

```python
from transformers import AutoTokenizer, AutoModelForSequenceClassification

repo_id = "your-username/my-model"

tokenizer = AutoTokenizer.from_pretrained("distilbert-base-uncased")
model = AutoModelForSequenceClassification.from_pretrained("distilbert-
base-uncased")

tokenizer.push_to_hub(repo_id)
model.push_to_hub(repo_id)
```