Huggingface Transformers

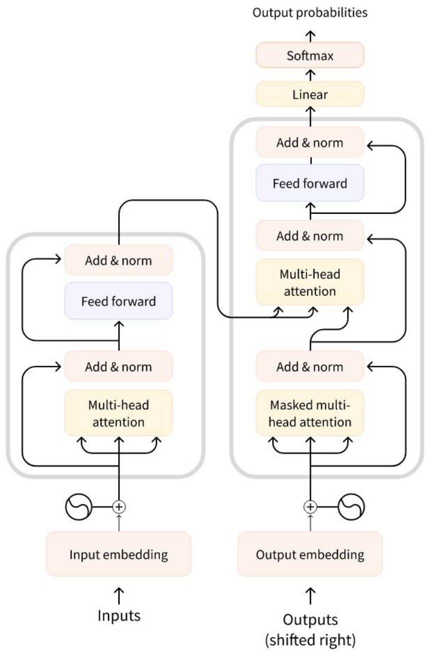
Learning Portfolio 5

universität innsbruck

Huggingface Transformers

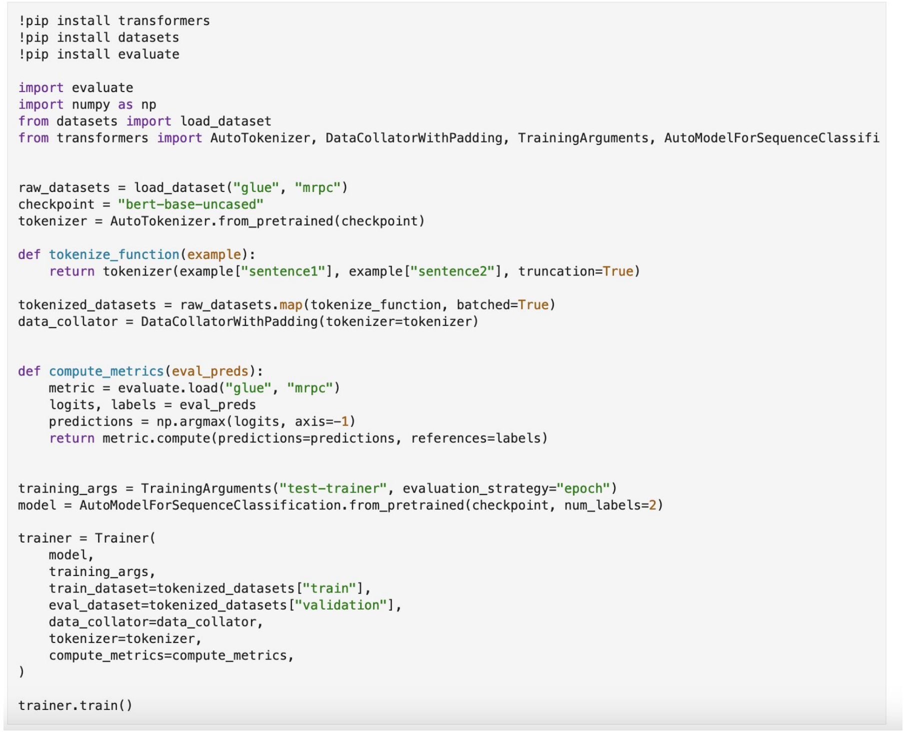
huggingfaces provides through its Transformers library architecture standards for Natural Language Processing (NLP) models. The Transformers library can be used to create and use pre-trained NLP models.

Use-Cases
Text classification, Zero-Shot classification, Text generation, Text completion, Question answering, Summarization, Translation



https://huggingface.co/models
https://huggingface.co/datasets

A full training example



```
!pip install transformers
!pip install datasets
!pip install evaluate

import evaluate
import numpy as np
from datasets import load_dataset
from transformers import AutoTokenizer, DataCollatorWithPadding, TrainingArguments, AutoModelForSequenceClassifi


raw_datasets = load_dataset("glue", "mrpc")
checkpoint = "bert-base-uncased"
tokenizer = AutoTokenizer.from_pretrained(checkpoint)

def tokenize_function(example):
    return tokenizer(example["sentence1"], example["sentence2"], truncation=True)

tokenized_datasets = raw_datasets.map(tokenize_function, batched=True)
data_collator = DataCollatorWithPadding(tokenizer=tokenizer)


def compute_metrics(eval_preds):
    metric = evaluate.load("glue", "mrpc")
    logits, labels = eval_preds
    predictions = np.argmax(logits, axis=-1)
    return metric.compute(predictions=predictions, references=labels)


training_args = TrainingArguments("test-trainer", evaluation_strategy="epoch")
model = AutoModelForSequenceClassification.from_pretrained(checkpoint, num_labels=2)

trainer = Trainer(
    model,
    training_args,
    train_dataset=tokenized_datasets["train"],
    eval_dataset=tokenized_datasets["validation"],
    data_collator=data_collator,
    tokenizer=tokenizer,
    compute_metrics=compute_metrics,
)

trainer.train()
```

Kontakt

Fabian Leuk
12215478
fabian.leuk@student.uibk.ac.at