Fine-tuning a model with the Trainer API or Keras

File Edit View Insert Runtime Tools Help

+ Code + Text Copy to Drive

Fine-tuning a model with the Trainer API or Keras

Install the Transformers, Datasets, and Evaluate libraries to run this notebook.

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

```
[ ] from datasets import load_dataset
    from transformers import AutoTokenizer, DataCollatorWithPadding

    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)
```

```
[ ] from transformers import AutoModelForSequenceClassification

    model = AutoModelForSequenceClassification.from_pretrained(checkpoint, num_labels=2)
```

```
[ ] from transformers import Trainer
        eval_dataset=tokenized_datasets["validation"],
        data_collator=data_collator,
        tokenizer=tokenizer,
    )
```

```
    (408, 2) (408,)
```

```
[ ] import numpy as np

    preds = np.argmax(predictions.predictions, axis=-1)
    metric.compute(predictions=preds, references=predictions.label_ids)
```

```
    {'accuracy': 0.8578431372549019, 'f1': 0.899653979238754}
```

```
[ ] training_args = TrainingArguments("test-trainer", evaluation_strategy="epoch")
    model = AutoModelForSequenceClassification.from_pretrained(checkpoint, num_labels=2)
        tokenizer=tokenizer,
        compute_metrics=compute_metrics,
    )
```

```
[ ] trainer.train()
```