Text Classification Using Transformer Networks (BERT)¶

Some initialization:

In [1]:

```
import random
import torch
import numpy as np
import pandas as pd
from tqdm.notebook import tqdm

# enable tqdm in pandas
tqdm.pandas()

# set to True to use the gpu (if there is one available)
use_gpu = True

# select device
device = torch.device('cuda' if use_gpu and torch.cuda.is_available() else 'cpu')
print(f'device: {device.type}')

# random seed
seed = 1122

# set random seed
if seed is not None:
    print(f'random seed: {seed}')
    random.seed(seed)
    np.random.seed(seed)
    torch.manual_seed(seed)
```

```
device: cuda
random seed: 1122
```

Read the train/dev/test datasets and create a HuggingFace Dataset object:

In [2]:

```
def read_data(filename):
    # read csv file
    df =
pd.read_csv('/kaggle/input/agnews-pytorch-simple-embed-classif-90/AG_NEWS/train.csv',
header=None)
    # add column names
    df.columns = ['label', 'title', 'description']
    # make labels zero-based
    df['label'] -= 1
    # concatenate title and description, and remove backslashes
    df['text'] = df['title'] + " " + df['description']
    df['text'] = df['text'].str.replace('\\', ' ', regex=False)
    return df
```

In [3]:

```
labels = open('/kaggle/input/classes/classes.txt').read().splitlines()
train_df = read_data('data/ag_news_csv/train.csv')
test_df = read_data('data/ag_news_csv/test.csv')
train_df
```

Out[3]:

	label	title	description	text
0	2	Wall St. Bears Claw Back Into the Black (Reuters)	Reuters - Short-sellers, Wall Street's dwindli...	Wall St. Bears Claw Back Into the Black (Reute...
1	2	Carlyle Looks Toward Commercial Aerospace (Reu...	Reuters - Private investment firm Carlyle Grou...	Carlyle Looks Toward Commercial Aerospace (Reu...
2	2	Oil and Economy Cloud Stocks' Outlook (Reuters)	Reuters - Soaring crude prices plus worries\ab...	Oil and Economy Cloud Stocks' Outlook (Reuters...
3	2	Iraq Halts Oil Exports from Main Southern Pipe...	Reuters - Authorities have halted oil export\f...	Iraq Halts Oil Exports from Main Southern Pipe...
4	2	Oil prices soar to all-time record, posing new...	AFP - Tearaway world oil prices, toppling reco...	Oil prices soar to all-time record, posing new...
...	...	...	...	...
119995	0	Pakistan's Musharraf Says Won't Quit as Army C...	KARACHI (Reuters) - Pakistani President Perve...	Pakistan's Musharraf Says Won't Quit as Army C...

	label	title	description	text
119996	1	Renteria signing a top-shelf deal	Red Sox general manager Theo Epstein acknowled...	Renteria signing a top-shelf deal Red Sox gene...
119997	1	Saban not going to Dolphins yet	The Miami Dolphins will put their courtship of...	Saban not going to Dolphins yet The Miami Dolp...
119998	1	Today's NFL games	PITTSBURGH at NY GIANTS Time: 1:30 p.m. Line: ...	Today's NFL games PITTSBURGH at NY GIANTS Time...
119999	1	Nets get Carter from Raptors	INDIANAPOLIS -- All-Star Vince Carter was trad...	Nets get Carter from Raptors INDIANAPOLIS -- A...

120000 rows × 4 columns

In [4]:

```
from sklearn.model_selection import train_test_split

train_df, eval_df = train_test_split(train_df, train_size=0.9)
train_df.reset_index(inplace=True, drop=True)
eval_df.reset_index(inplace=True, drop=True)

print(f'train rows: {len(train_df.index):,}')
```

```
print(f'eval rows: {len(eval_df.index):,}')
print(f'test rows: {len(test_df.index):,}')
```

```
train rows: 108,000
eval rows: 12,000
test rows: 120,000
```

In [5]:

```
from datasets import Dataset, DatasetDict

ds = DatasetDict()
ds['train'] = Dataset.from_pandas(train_df)
ds['validation'] = Dataset.from_pandas(eval_df)
ds['test'] = Dataset.from_pandas(test_df)
ds
```

Out[5]:

```
DatasetDict({
    train: Dataset({
        features: ['label', 'title', 'description', 'text'],
        num_rows: 108000
    })
    validation: Dataset({
        features: ['label', 'title', 'description', 'text'],
        num_rows: 12000
    })
    test: Dataset({
        features: ['label', 'title', 'description', 'text'],
        num_rows: 120000
    })
})
```

Tokenize the texts:

In [6]:

```
from transformers import AutoTokenizer

transformer_name = 'bert-base-cased'
tokenizer = AutoTokenizer.from_pretrained(transformer_name)
```

```
tokenizer_config.json:   0%|          | 0.00/49.0 [00:00<?, ?B/s]
```

```
config.json:   0%|          | 0.00/570 [00:00<?, ?B/s]
```

```
vocab.txt:   0%|          | 0.00/213k [00:00<?, ?B/s]
```

```
tokenizer.json:   0%|          | 0.00/436k [00:00<?, ?B/s]
```

```
/opt/conda/lib/python3.10/site-packages/transformers/tokenization_utils_base.py:1617:
FutureWarning: `clean_up_tokenization_spaces` was not set. It will be set to `True` by default.
This behavior will be deprecated in transformers v4.45, and will be then set to `False` by default.
For more details check this issue: https://github.com/huggingface/transformers/issues/31884
  warnings.warn(
```

In [7]:

```
def tokenize(examples):
    return tokenizer(examples['text'], truncation=True)
```

```
train_ds = ds['train'].map(
    tokenize, batched=True,
    remove_columns=['title', 'description', 'text'],
)
eval_ds = ds['validation'].map(
    tokenize,
    batched=True,
    remove_columns=['title', 'description', 'text'],
)
train_ds.to_pandas()
```

```
Map:   0%|          | 0/108000 [00:00<?, ? examples/s]
```

```
Map:   0%|          | 0/12000 [00:00<?, ? examples/s]
```

Out[7]:

	label	input_ids	token_type_ids	attention_mask
0	2	[101, 16752, 13335, 1186, 2101, 6690, 9717, 11...	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...	[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ...
1	1	[101, 145, 11680, 17308, 9741, 2428, 150, 1469...	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...	[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ...
2	2	[101, 1418, 14099, 27086, 1494, 1114, 4031, 11...	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...	[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ...
3	1	[101, 2404, 117, 6734,	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,	[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,

	label	input_ids	token_type_ids	attention_mask
		1996, 118, 1565, 5465, ...	0, 0, 0, ...	1, 1, 1, ...
4	3	[101, 142, 10044, 27302, 4317, 1584, 3273, 111...	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...	[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ...
...	...	...	...	...
107995	1	[101, 4922, 2274, 1654, 1112, 10503, 1505, 112...	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...	[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ...
107996	3	[101, 10605, 24632, 11252, 21285, 10221, 118, ...	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...	[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ...
107997	2	[101, 13832, 3484, 11300, 4060, 5058, 112, 188...	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...	[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ...
107998	3	[101, 142, 13675, 3756, 5795, 2445, 1104, 109,...	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...	[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ...
107999	2	[101, 157, 16450, 1658, 5302, 185, 7776, 11006...	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...	[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ...

108000 rows × 4 columns

Create the transformer model:

In [8]:

```
from torch import nn
from transformers.modeling_outputs import SequenceClassifierOutput
from transformers.models.bert.modeling_bert import BertModel, BertPreTrainedModel

#
https://github.com/huggingface/transformers/blob/65659a29cf5a079842e61a63d57fa24474288
998/src/transformers/models/bert/modeling_bert.py#L1486

class BertForSequenceClassification(BertPreTrainedModel):
    def __init__(self, config):
        super().__init__(config)
        self.num_labels = config.num_labels
        self.bert = BertModel(config)
        self.dropout = nn.Dropout(config.hidden_dropout_prob)
        self.classifier = nn.Linear(config.hidden_size, config.num_labels)
        self.init_weights()

    def forward(self, input_ids=None, attention_mask=None, token_type_ids=None,
labels=None, **kwargs):
        outputs = self.bert(
            input_ids,
            attention_mask=attention_mask,
            token_type_ids=token_type_ids,
            **kwargs,
        )
        cls_outputs = outputs.last_hidden_state[:, 0, :]
        cls_outputs = self.dropout(cls_outputs)
        logits = self.classifier(cls_outputs)
        loss = None
        if labels is not None:
            loss_fn = nn.CrossEntropyLoss()
            loss = loss_fn(logits, labels)
```

```
        return SequenceClassifierOutput(
            loss=loss,
            logits=logits,
            hidden_states=outputs.hidden_states,
            attentions=outputs.attentions,
        )
```

In [9]:

```
from transformers import AutoConfig

config = AutoConfig.from_pretrained(
    transformer_name,
    num_labels=len(labels),
)

model = (
    BertForSequenceClassification
    .from_pretrained(transformer_name, config=config)
)
```

```
model.safetensors:   0%|          | 0.00/436M [00:00<?, ?B/s]
```

```
Some weights of BertForSequenceClassification were not initialized from the model checkpoint
at bert-base-cased and are newly initialized: ['classifier.bias', 'classifier.weight']
You should probably TRAIN this model on a down-stream task to be able to use it for predictions
and inference.
```

Create the trainer object and train:

In [10]:

```
from transformers import TrainingArguments

num_epochs = 2
batch_size = 24
weight_decay = 0.01
model_name = f'{transformer_name}-sequence-classification'

training_args = TrainingArguments(
    output_dir=model_name,
    log_level='error',
    num_train_epochs=num_epochs,
    per_device_train_batch_size=batch_size,
    per_device_eval_batch_size=batch_size,
    evaluation_strategy='epoch',
    weight_decay=weight_decay,
)
```

```
/opt/conda/lib/python3.10/site-packages/transformers/training_args.py:1545: FutureWarning:
`evaluation_strategy` is deprecated and will be removed in version 4.46 of 🤗 Transformers.
Use `eval_strategy` instead
  warnings.warn(
```

In [11]:

```
from sklearn.metrics import accuracy_score

def compute_metrics(eval_pred):
    y_true = eval_pred.label_ids
    y_pred = np.argmax(eval_pred.predictions, axis=-1)
    return {'accuracy': accuracy_score(y_true, y_pred)}
```

In [12]:

```
from transformers import Trainer

trainer = Trainer(
    model=model,
    args=training_args,
    compute_metrics=compute_metrics,
    train_dataset=train_ds,
    eval_dataset=eval_ds,
    tokenizer=tokenizer,
)
```

In [13]:

```
trainer.train()
```

```
wandb: WARNING The `run_name` is currently set to the same value as
`TrainingArguments.output_dir`. If this was not intended, please specify a different run name by
setting the `TrainingArguments.run_name` parameter.
wandb: Using wandb-core as the SDK backend. Please refer to https://wandb.me/wandb-core
for more information.
wandb: Logging into wandb.ai. (Learn how to deploy a W&B server locally:
https://wandb.me/wandb-server)
wandb: You can find your API key in your browser here: https://wandb.ai/authorize
wandb: Paste an API key from your profile and hit enter, or press ctrl+c to quit:


wandb: Appending key for api.wandb.ai to your netrc file: /root/.netrc



VBox(children=(Label(value='Waiting for wandb.init()...\r'),
FloatProgress(value=0.011113386722222332, max=1.0…

Tracking run with wandb version 0.18.3

Run data is saved locally in /kaggle/working/wandb/run-20241119_133233-f73m1ism
```

Syncing run **bert-base-cased-sequence-classification** to Weights & Biases (docs)

View project at https://wandb.ai/a01198261-tecnol-gico-de-monterrey/huggingface

View run at
https://wandb.ai/a01198261-tecnol-gico-de-monterrey/huggingface/runs/f73m1ism

/opt/conda/lib/python3.10/site-packages/torch/nn/parallel/parallel_apply.py:79: FutureWarning: `torch.cuda.amp.autocast(args...)` is deprecated. Please use `torch.amp.autocast('cuda', args...)` instead.
  with torch.cuda.device(device), torch.cuda.stream(stream), autocast(enabled=autocast_enabled):
/opt/conda/lib/python3.10/site-packages/torch/nn/parallel/_functions.py:68: UserWarning: Was asked to gather along dimension 0, but all input tensors were scalars; will instead unsqueeze and return a vector.
  warnings.warn('Was asked to gather along dimension 0, but all '

[4500/4500 53:13, Epoch 2/2]

Epoch	Training Loss	Validation Loss	Accuracy
1	0.188400	0.174404	0.940833
2	0.101500	0.164861	0.945917

/opt/conda/lib/python3.10/site-packages/torch/nn/parallel/parallel_apply.py:79: FutureWarning: `torch.cuda.amp.autocast(args...)` is deprecated. Please use `torch.amp.autocast('cuda', args...)` instead.
  with torch.cuda.device(device), torch.cuda.stream(stream), autocast(enabled=autocast_enabled):
/opt/conda/lib/python3.10/site-packages/torch/nn/parallel/_functions.py:68: UserWarning: Was asked to gather along dimension 0, but all input tensors were scalars; will instead unsqueeze and return a vector.
  warnings.warn('Was asked to gather along dimension 0, but all '
/opt/conda/lib/python3.10/site-packages/torch/nn/parallel/parallel_apply.py:79: FutureWarning: `torch.cuda.amp.autocast(args...)` is deprecated. Please use `torch.amp.autocast('cuda', args...)` instead.

with torch.cuda.device(device), torch.cuda.stream(stream),
autocast(enabled=autocast_enabled):
/opt/conda/lib/python3.10/site-packages/torch/nn/parallel/_functions.py:68: UserWarning: Was asked to gather along dimension 0, but all input tensors were scalars; will instead unsqueeze and return a vector.
warnings.warn('Was asked to gather along dimension 0, but all '
/opt/conda/lib/python3.10/site-packages/torch/nn/parallel/parallel_apply.py:79: FutureWarning: `torch.cuda.amp.autocast(args...)` is deprecated. Please use `torch.amp.autocast('cuda', args...)` instead.
with torch.cuda.device(device), torch.cuda.stream(stream),
autocast(enabled=autocast_enabled):
/opt/conda/lib/python3.10/site-packages/torch/nn/parallel/_functions.py:68: UserWarning: Was asked to gather along dimension 0, but all input tensors were scalars; will instead unsqueeze and return a vector.
warnings.warn('Was asked to gather along dimension 0, but all '
/opt/conda/lib/python3.10/site-packages/torch/nn/parallel/parallel_apply.py:79: FutureWarning: `torch.cuda.amp.autocast(args...)` is deprecated. Please use `torch.amp.autocast('cuda', args...)` instead.
with torch.cuda.device(device), torch.cuda.stream(stream),
autocast(enabled=autocast_enabled):
/opt/conda/lib/python3.10/site-packages/torch/nn/parallel/_functions.py:68: UserWarning: Was asked to gather along dimension 0, but all input tensors were scalars; will instead unsqueeze and return a vector.
warnings.warn('Was asked to gather along dimension 0, but all '
/opt/conda/lib/python3.10/site-packages/torch/nn/parallel/parallel_apply.py:79: FutureWarning: `torch.cuda.amp.autocast(args...)` is deprecated. Please use `torch.amp.autocast('cuda', args...)` instead.
with torch.cuda.device(device), torch.cuda.stream(stream),
autocast(enabled=autocast_enabled):
/opt/conda/lib/python3.10/site-packages/torch/nn/parallel/_functions.py:68: UserWarning: Was asked to gather along dimension 0, but all input tensors were scalars; will instead unsqueeze and return a vector.
warnings.warn('Was asked to gather along dimension 0, but all '
/opt/conda/lib/python3.10/site-packages/torch/nn/parallel/parallel_apply.py:79: FutureWarning: `torch.cuda.amp.autocast(args...)` is deprecated. Please use `torch.amp.autocast('cuda', args...)` instead.
with torch.cuda.device(device), torch.cuda.stream(stream),
autocast(enabled=autocast_enabled):
/opt/conda/lib/python3.10/site-packages/torch/nn/parallel/_functions.py:68: UserWarning: Was asked to gather along dimension 0, but all input tensors were scalars; will instead unsqueeze and return a vector.
warnings.warn('Was asked to gather along dimension 0, but all '
/opt/conda/lib/python3.10/site-packages/torch/nn/parallel/parallel_apply.py:79: FutureWarning: `torch.cuda.amp.autocast(args...)` is deprecated. Please use `torch.amp.autocast('cuda', args...)` instead.
with torch.cuda.device(device), torch.cuda.stream(stream),
autocast(enabled=autocast_enabled):
/opt/conda/lib/python3.10/site-packages/torch/nn/parallel/_functions.py:68: UserWarning: Was

```
asked to gather along dimension 0, but all input tensors were scalars; will instead unsqueeze and return a vector.
  warnings.warn('Was asked to gather along dimension 0, but all '
/opt/conda/lib/python3.10/site-packages/torch/nn/parallel/parallel_apply.py:79: FutureWarning: `torch.cuda.amp.autocast(args...)` is deprecated. Please use `torch.amp.autocast('cuda', args...)` instead.
  with torch.cuda.device(device), torch.cuda.stream(stream), autocast(enabled=autocast_enabled):
/opt/conda/lib/python3.10/site-packages/torch/nn/parallel/_functions.py:68: UserWarning: Was asked to gather along dimension 0, but all input tensors were scalars; will instead unsqueeze and return a vector.
  warnings.warn('Was asked to gather along dimension 0, but all '
/opt/conda/lib/python3.10/site-packages/torch/nn/parallel/parallel_apply.py:79: FutureWarning: `torch.cuda.amp.autocast(args...)` is deprecated. Please use `torch.amp.autocast('cuda', args...)` instead.
  with torch.cuda.device(device), torch.cuda.stream(stream), autocast(enabled=autocast_enabled):
/opt/conda/lib/python3.10/site-packages/torch/nn/parallel/_functions.py:68: UserWarning: Was asked to gather along dimension 0, but all input tensors were scalars; will instead unsqueeze and return a vector.
  warnings.warn('Was asked to gather along dimension 0, but all '
/opt/conda/lib/python3.10/site-packages/torch/nn/parallel/parallel_apply.py:79: FutureWarning: `torch.cuda.amp.autocast(args...)` is deprecated. Please use `torch.amp.autocast('cuda', args...)` instead.
  with torch.cuda.device(device), torch.cuda.stream(stream), autocast(enabled=autocast_enabled):
/opt/conda/lib/python3.10/site-packages/torch/nn/parallel/_functions.py:68: UserWarning: Was asked to gather along dimension 0, but all input tensors were scalars; will instead unsqueeze and return a vector.
  warnings.warn('Was asked to gather along dimension 0, but all '
```

Out[13]:

```
TrainOutput(global_step=4500, training_loss=0.16177347649468315, metrics={'train_runtime': 3223.872, 'train_samples_per_second': 67.0, 'train_steps_per_second': 1.396, 'total_flos': 1.5600315493990656e+16, 'train_loss': 0.16177347649468315, 'epoch': 2.0})
```

Evaluate on the test partition:

In [14]:

```
test_ds = ds['test'].map(
    tokenize,
    batched=True,
    remove_columns=['title', 'description', 'text'],
)
test_ds.to_pandas()
```

```
Map:   0%|          | 0/120000 [00:00<?, ? examples/s]
```

Out[14]:

	label	input_ids	token_type_ids	attention_mask
0	2	[101, 6250, 1457, 119, 10169, 140, 9598, 4388,...	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...	[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ...
1	2	[101, 19879, 1513, 15218, 27674, 10472, 19417,...	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...	[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ...
2	2	[101, 9105, 1105, 14592, 11804, 9924, 1116, 11...	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...	[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ...
3	2	[101, 5008, 12193, 2145, 9105, 18947, 13245, 1...	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...	[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ...

	label	input_ids	token_type_ids	attention_mask
4	2	[101, 9105, 7352, 1177, 1813, 1106, 1155, 118,...	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...	[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ...
...	...	...	...	...
119995	0	[101, 3658, 112, 188, 19569, 5480, 10582, 2087...	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...	[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ...
119996	1	[101, 16513, 2083, 1465, 6086, 170, 1499, 118,...	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...	[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ...
119997	1	[101, 17784, 7167, 1136, 1280, 1106, 19112, 18...	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...	[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ...
119998	1	[101, 3570, 112, 188, 4279, 1638, 153, 12150, ...	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...	[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ...
119999	1	[101, 20820, 1116, 1243, 5007, 1121, 21196, 50...	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...	[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ...

120000 rows × 4 columns

In [15]:

```
output = trainer.predict(test_ds)
output
```

```
/opt/conda/lib/python3.10/site-packages/torch/nn/parallel/parallel_apply.py:79: FutureWarning:
`torch.cuda.amp.autocast(args...)` is deprecated. Please use `torch.amp.autocast('cuda',
args...)` instead.
  with torch.cuda.device(device), torch.cuda.stream(stream),
autocast(enabled=autocast_enabled):
/opt/conda/lib/python3.10/site-packages/torch/nn/parallel/_functions.py:68: UserWarning: Was
asked to gather along dimension 0, but all input tensors were scalars; will instead unsqueeze
and return a vector.
  warnings.warn('Was asked to gather along dimension 0, but all '
```

Out[15]:

```
PredictionOutput(predictions=array([[ 0.26401708, -5.151869  ,  4.684026  , -1.0956956 ],
       [-0.84335977, -4.805707  ,  4.112665  ,  0.7136828 ],
       [-0.22219476, -4.4022207 ,  5.2064476 , -1.5156815 ],
       ...,
       [-1.8854737 ,  7.2194147 , -2.5020251 , -2.9388943 ],
       [-1.3959128 ,  7.084009  , -3.0517933 , -3.1712253 ],
       [-1.6128936 ,  7.777843  , -2.7770045 , -3.351252  ]],
      dtype=float32), label_ids=array([2, 2, 2, ..., 1, 1, 1]), metrics={'test_loss':
0.07129824161529541, 'test_accuracy': 0.9768333333333333, 'test_runtime': 562.4616,
'test_samples_per_second': 213.348, 'test_steps_per_second': 4.445})
```

In [16]:

```
from sklearn.metrics import classification_report

y_true = output.label_ids
y_pred = np.argmax(output.predictions, axis=-1)
target_names = labels
print(classification_report(y_true, y_pred, target_names=target_names))
```

	precision	recall	f1-score	support
World	0.99	0.98	0.98	30000
Sports	1.00	1.00	1.00	30000
Business	0.97	0.96	0.96	30000
Sci/Tech	0.96	0.97	0.97	30000
accuracy			0.98	120000
macro avg	0.98	0.98	0.98	120000
weighted avg	0.98	0.98	0.98	120000

El código toma datos de texto etiquetados en forma de títulos y descripciones de películas y este entrena un modelo BERT para clasificar los textos en diferentes categorías. El flujo del código seria de esta forma:

- Al inicio se ponen todas las librerías, se detecta si hay GPU y se escoge una semilla .
- Lectura de los datos, ajuste de las etiquetas y renombramiento a las columnas y se limpian caracteres no deseados.
- División de los datos en 90% de train y 10% de test y conversión de los datos8
- Creación de tokens de texto
- Se define el modelo mediante BertForSequenceClassification y se entrena el modelo con BERT
- Y por último los resultados de las medidas de evaluación como precision, recall y F1-score.

En conclusión este código es un ejemplo de cómo el modelo BERT es útil para problemas de clasificación, como se vio en clase este modelo nos puede ayudar a mejorar los scores y tener predicciones más correctas y acertadas.