Lab6-Assignment: Topic Classification

Use the same training, development, and test partitions of the 20 newsgroups text dataset as in Lab6.4-Topic-classification-BERT.ipynb

- **Fine-tune and examine the performance of another transformer-based pretrained language models, e.g., RoBERTa, XLNet**
- **Compare the performance of this model to the results achieved in Lab6.4-Topic-classification-BERT.ipynb and to a conventional machine learning approach (e.g., SVM, Naive Bayes) using bag-of-words or other engineered features of your choice. Describe the differences in performance in terms of Precision, Recall, and F1-score evaluation metrics.**

1. Installs & Imports

In [1]:

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call
drive.mount("/content/drive", force_remount=True).
```

In [2]:

```
!pip install simpletransformers --upgrade
```

```
Requirement already satisfied: simpletransformers in /usr/local/lib/python3.11/dist-packages (
0.70.1)
Requirement already satisfied: numpy in /usr/local/lib/python3.11/dist-packages (from simpletr
ansformers) (1.26.4)
Requirement already satisfied: requests in /usr/local/lib/python3.11/dist-packages (from simpl
etransformers) (2.32.3)
Requirement already satisfied: tqdm>=4.47.0 in /usr/local/lib/python3.11/dist-packages (from s
impletransformers) (4.67.1)
Requirement already satisfied: regex in /usr/local/lib/python3.11/dist-packages (from simpletr
ansformers) (2024.11.6)
Requirement already satisfied: transformers>=4.31.0 in /usr/local/lib/python3.11/dist-packages
(from simpletransformers) (4.48.3)
Requirement already satisfied: datasets in /usr/local/lib/python3.11/dist-packages (from simpl
etransformers) (3.3.2)
Requirement already satisfied: scipy in /usr/local/lib/python3.11/dist-packages (from simpletr
ansformers) (1.13.1)
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.11/dist-packages (from s
impletransformers) (1.6.1)
Requirement already satisfied: seqeval in /usr/local/lib/python3.11/dist-packages (from simple
transformers) (1.2.2)
Requirement already satisfied: tensorboard in /usr/local/lib/python3.11/dist-packages (from si
mpletransformers) (2.18.0)
Requirement already satisfied: tensorboardx in /usr/local/lib/python3.11/dist-packages (from s
impletransformers) (2.6.2.2)
Requirement already satisfied: pandas in /usr/local/lib/python3.11/dist-packages (from simplet
ransformers) (2.2.2)
Requirement already satisfied: tokenizers in /usr/local/lib/python3.11/dist-packages (from sim
pletransformers) (0.21.0)
Requirement already satisfied: wandb>=0.10.32 in /usr/local/lib/python3.11/dist-packages (from
simpletransformers) (0.19.7)
Requirement already satisfied: streamlit in /usr/local/lib/python3.11/dist-packages (from simp
letransformers) (1.42.2)
Requirement already satisfied: sentencepiece in /usr/local/lib/python3.11/dist-packages (from
simpletransformers) (0.2.0)
Requirement already satisfied: filelock in /usr/local/lib/python3.11/dist-packages (from trans
formers>=4.31.0->simpletransformers) (3.17.0)
Requirement already satisfied: huggingface-hub<1.0,>=0.24.0 in /usr/local/lib/python3.11/dist-
```

```
packages (from transformers>=4.31.0->simpletransformers) (0.28.1)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.11/dist-packages (fro
m transformers>=4.31.0->simpletransformers) (24.2)
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.11/dist-packages (from tr
ansformers>=4.31.0->simpletransformers) (6.0.2)
Requirement already satisfied: safetensors>=0.4.1 in /usr/local/lib/python3.11/dist-packages (
from transformers>=4.31.0->simpletransformers) (0.5.3)
Requirement already satisfied: click!=8.0.0,>=7.1 in /usr/local/lib/python3.11/dist-packages (
from wandb>=0.10.32->simpletransformers) (8.1.8)
Requirement already satisfied: docker-pycreds>=0.4.0 in /usr/local/lib/python3.11/dist-package
s (from wandb>=0.10.32->simpletransformers) (0.4.0)
Requirement already satisfied: gitpython!=3.1.29,>=1.0.0 in /usr/local/lib/python3.11/dist-pac
kages (from wandb>=0.10.32->simpletransformers) (3.1.44)
Requirement already satisfied: platformdirs in /usr/local/lib/python3.11/dist-packages (from w
andb>=0.10.32->simpletransformers) (4.3.6)
Requirement already satisfied: protobuf!=4.21.0,!=5.28.0,<6,>=3.19.0 in /usr/local/lib/python3
.11/dist-packages (from wandb>=0.10.32->simpletransformers) (4.25.6)
Requirement already satisfied: psutil>=5.0.0 in /usr/local/lib/python3.11/dist-packages (from
wandb>=0.10.32->simpletransformers) (5.9.5)
Requirement already satisfied: pydantic<3,>=2.6 in /usr/local/lib/python3.11/dist-packages (fr
om wandb>=0.10.32->simpletransformers) (2.10.6)
Requirement already satisfied: sentry-sdk>=2.0.0 in /usr/local/lib/python3.11/dist-packages (f
rom wandb>=0.10.32->simpletransformers) (2.22.0)
Requirement already satisfied: setproctitle in /usr/local/lib/python3.11/dist-packages (from w
andb>=0.10.32->simpletransformers) (1.3.5)
Requirement already satisfied: setuptools in /usr/local/lib/python3.11/dist-packages (from wan
db>=0.10.32->simpletransformers) (75.1.0)
Requirement already satisfied: typing-extensions<5,>=4.4 in /usr/local/lib/python3.11/dist-pac
kages (from wandb>=0.10.32->simpletransformers) (4.12.2)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.11/dist-pack
ages (from requests->simpletransformers) (3.4.1)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.11/dist-packages (from r
equests->simpletransformers) (3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.11/dist-packages (
from requests->simpletransformers) (2.3.0)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.11/dist-packages (
from requests->simpletransformers) (2025.1.31)
Requirement already satisfied: pyarrow>=15.0.0 in /usr/local/lib/python3.11/dist-packages (fro
m datasets->simpletransformers) (18.1.0)
Requirement already satisfied: dill<0.3.9,>=0.3.0 in /usr/local/lib/python3.11/dist-packages (
from datasets->simpletransformers) (0.3.8)
Requirement already satisfied: xxhash in /usr/local/lib/python3.11/dist-packages (from dataset
s->simpletransformers) (3.5.0)
Requirement already satisfied: multiprocess<0.70.17 in /usr/local/lib/python3.11/dist-packages
(from datasets->simpletransformers) (0.70.16)
Requirement already satisfied: fsspec<=2024.12.0,>=2023.1.0 in /usr/local/lib/python3.11/dist-
packages (from fsspec[http]<=2024.12.0,>=2023.1.0->datasets->simpletransformers) (2024.10.0)
Requirement already satisfied: aiohttp in /usr/local/lib/python3.11/dist-packages (from datase
ts->simpletransformers) (3.11.13)
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.11/dist-packag
es (from pandas->simpletransformers) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.11/dist-packages (from p
andas->simpletransformers) (2025.1)
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.11/dist-packages (from
pandas->simpletransformers) (2025.1)
Requirement already satisfied: joblib>=1.2.0 in /usr/local/lib/python3.11/dist-packages (from
scikit-learn->simpletransformers) (1.4.2)
Requirement already satisfied: threadpoolctl>=3.1.0 in /usr/local/lib/python3.11/dist-packages
(from scikit-learn->simpletransformers) (3.5.0)
Requirement already satisfied: altair<6,>=4.0 in /usr/local/lib/python3.11/dist-packages (from
streamlit->simpletransformers) (5.5.0)
Requirement already satisfied: blinker<2,>=1.0.0 in /usr/local/lib/python3.11/dist-packages (f
rom streamlit->simpletransformers) (1.9.0)
Requirement already satisfied: cachetools<6,>=4.0 in /usr/local/lib/python3.11/dist-packages (
from streamlit->simpletransformers) (5.5.2)
Requirement already satisfied: pillow<12,>=7.1.0 in /usr/local/lib/python3.11/dist-packages (f
rom streamlit->simpletransformers) (11.1.0)
Requirement already satisfied: rich<14,>=10.14.0 in /usr/local/lib/python3.11/dist-packages (f
rom streamlit->simpletransformers) (13.9.4)
Requirement already satisfied: tenacity<10,>=8.1.0 in /usr/local/lib/python3.11/dist-packages
(from streamlit->simpletransformers) (9.0.0)
```

```
Requirement already satisfied: toml<2,>=0.10.1 in /usr/local/lib/python3.11/dist-packages (fro
m streamlit->simpletransformers) (0.10.2)
Requirement already satisfied: watchdog<7,>=2.1.5 in /usr/local/lib/python3.11/dist-packages (
from streamlit->simpletransformers) (6.0.0)
Requirement already satisfied: pydeck<1,>=0.8.0b4 in /usr/local/lib/python3.11/dist-packages (
from streamlit->simpletransformers) (0.9.1)
Requirement already satisfied: tornado<7,>=6.0.3 in /usr/local/lib/python3.11/dist-packages (f
rom streamlit->simpletransformers) (6.4.2)
Requirement already satisfied: absl-py>=0.4 in /usr/local/lib/python3.11/dist-packages (from t
ensorboard->simpletransformers) (1.4.0)
Requirement already satisfied: grpcio>=1.48.2 in /usr/local/lib/python3.11/dist-packages (from
tensorboard->simpletransformers) (1.70.0)
Requirement already satisfied: markdown>=2.6.8 in /usr/local/lib/python3.11/dist-packages (fro
m tensorboard->simpletransformers) (3.7)
Requirement already satisfied: six>1.9 in /usr/local/lib/python3.11/dist-packages (from tensor
board->simpletransformers) (1.17.0)
Requirement already satisfied: tensorboard-data-server<0.8.0,>=0.7.0 in /usr/local/lib/python3
.11/dist-packages (from tensorboard->simpletransformers) (0.7.2)
Requirement already satisfied: werkzeug>=1.0.1 in /usr/local/lib/python3.11/dist-packages (fro
m tensorboard->simpletransformers) (3.1.3)
Requirement already satisfied: jinja2 in /usr/local/lib/python3.11/dist-packages (from altair<
6,>=4.0->streamlit->simpletransformers) (3.1.5)
Requirement already satisfied: jsonschema>=3.0 in /usr/local/lib/python3.11/dist-packages (fro
m altair<6,>=4.0->streamlit->simpletransformers) (4.23.0)
Requirement already satisfied: narwhals>=1.14.2 in /usr/local/lib/python3.11/dist-packages (fr
om altair<6,>=4.0->streamlit->simpletransformers) (1.28.0)
Requirement already satisfied: aiohappyeyeballs>=2.3.0 in /usr/local/lib/python3.11/dist-packa
ges (from aiohttp->datasets->simpletransformers) (2.4.6)
Requirement already satisfied: aiosignal>=1.1.2 in /usr/local/lib/python3.11/dist-packages (fr
om aiohttp->datasets->simpletransformers) (1.3.2)
Requirement already satisfied: attrs>=17.3.0 in /usr/local/lib/python3.11/dist-packages (from
aiohttp->datasets->simpletransformers) (25.1.0)
Requirement already satisfied: frozenlist>=1.1.1 in /usr/local/lib/python3.11/dist-packages (f
rom aiohttp->datasets->simpletransformers) (1.5.0)
Requirement already satisfied: multidict<7.0,>=4.5 in /usr/local/lib/python3.11/dist-packages
(from aiohttp->datasets->simpletransformers) (6.1.0)
Requirement already satisfied: propcache>=0.2.0 in /usr/local/lib/python3.11/dist-packages (fr
om aiohttp->datasets->simpletransformers) (0.3.0)
Requirement already satisfied: yarl<2.0,>=1.17.0 in /usr/local/lib/python3.11/dist-packages (f
rom aiohttp->datasets->simpletransformers) (1.18.3)
Requirement already satisfied: gitdb<5,>=4.0.1 in /usr/local/lib/python3.11/dist-packages (fro
m gitpython!=3.1.29,>=1.0.0->wandb>=0.10.32->simpletransformers) (4.0.12)
Requirement already satisfied: annotated-types>=0.6.0 in /usr/local/lib/python3.11/dist-packag
es (from pydantic<3,>=2.6->wandb>=0.10.32->simpletransformers) (0.7.0)
Requirement already satisfied: pydantic-core==2.27.2 in /usr/local/lib/python3.11/dist-package
s (from pydantic<3,>=2.6->wandb>=0.10.32->simpletransformers) (2.27.2)
Requirement already satisfied: markdown-it-py>=2.2.0 in /usr/local/lib/python3.11/dist-package
s (from rich<14,>=10.14.0->streamlit->simpletransformers) (3.0.0)
Requirement already satisfied: pygments<3.0.0,>=2.13.0 in /usr/local/lib/python3.11/dist-packa
ges (from rich<14,>=10.14.0->streamlit->simpletransformers) (2.18.0)
Requirement already satisfied: MarkupSafe>=2.1.1 in /usr/local/lib/python3.11/dist-packages (f
rom werkzeug>=1.0.1->tensorboard->simpletransformers) (3.0.2)
Requirement already satisfied: smmap<6,>=3.0.1 in /usr/local/lib/python3.11/dist-packages (fro
m gitdb<5,>=4.0.1->gitpython!=3.1.29,>=1.0.0->wandb>=0.10.32->simpletransformers) (5.0.2)
Requirement already satisfied: jsonschema-specifications>=2023.03.6 in /usr/local/lib/python3.
11/dist-packages (from jsonschema>=3.0->altair<6,>=4.0->streamlit->simpletransformers) (2024.1
0.1)
Requirement already satisfied: referencing>=0.28.4 in /usr/local/lib/python3.11/dist-packages
(from jsonschema>=3.0->altair<6,>=4.0->streamlit->simpletransformers) (0.36.2)
Requirement already satisfied: rpds-py>=0.7.1 in /usr/local/lib/python3.11/dist-packages (from
jsonschema>=3.0->altair<6,>=4.0->streamlit->simpletransformers) (0.23.1)
Requirement already satisfied: mdurl~=0.1 in /usr/local/lib/python3.11/dist-packages (from mar
kdown-it-py>=2.2.0->rich<14,>=10.14.0->streamlit->simpletransformers) (0.1.2)
```

In [1]:

```python
import pandas as pd
import torch
from collections import Counter
import matplotlib.pyplot as plt
```

```
from sklearn.datasets import fetch_20newsgroups
from sklearn import svm
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.metrics import classification_report
from sklearn.model_selection import train_test_split
from simpletransformers.classification import ClassificationArgs, ClassificationModel
```

2. Dataset Loading & Splitting

In [2]:

```
# Load only the four specific categories we need
categories = ["alt.atheism", "comp.graphics", "sci.med", "sci.space"]

# Strip out headers, footers, and quoted text to prevent overfitting
train_groups = fetch_20newsgroups(
    subset="train",
    remove=("headers", "footers", "quotes"),
    categories=categories,
    random_state=42,
)
test_groups = fetch_20newsgroups(
    subset="test",
    remove=("headers", "footers", "quotes"),
    categories=categories,
    random_state=42,
)
```

In [3]:

```
# Check if identical to notebook 6.4
print("Distribution in Training Set:")
print(dict(sorted(Counter(train_groups.target).items(), key=lambda x: x[0])))
print("\nDistribution in Test Set:")
print(dict(sorted(Counter(test_groups.target).items(), key=lambda x: x[0])))
```

```
Distribution in Training Set:
{0: 480, 1: 584, 2: 594, 3: 593}

Distribution in Test Set:
{0: 319, 1: 389, 2: 396, 3: 394}
```

In [4]:

```
# Convert the training and test sets to dataframes
train_df = pd.DataFrame({"text": train_groups.data, "labels": train_groups.target})
test_df = pd.DataFrame({"text": test_groups.data, "labels": test_groups.target})

# Split the training set into two, so that 10% of it can be used as validation set
train_df, dev_df = train_test_split(
    train_df, test_size=0.1, random_state=0, stratify=train_df[["labels"]]
)
```

In [5]:

```
# Check if identical to notebook 6.4
print("Distribution in Training Set:")
print(dict(sorted(Counter(train_df["labels"]).items(), key=lambda x: x[0])))
print("\nDistribution in Validation Set:")
print(dict(sorted(Counter(dev_df["labels"]).items(), key=lambda x: x[0])))
```

```
Distribution in Training Set:
{0: 432, 1: 525, 2: 534, 3: 534}

Distribution in Validation Set:
{0: 48, 1: 59, 2: 60, 3: 59}
```

3. Finetune RoBERTa for Topic Classification on the Dataset

In [ ]:

```
# Model Configuration
model_args = ClassificationArgs()

# Overwrite existing saved models in the same directory
model_args.overwrite_output_dir = True

# Enable evaluation during training to monitor performance
model_args.evaluate_during_training = True

# Training parameters
model_args.num_train_epochs = 10  # Train for 10 epochs
model_args.train_batch_size = 32  # Process 32 samples per batch
model_args.learning_rate = 4e-6  # Learning rate for optimization
model_args.max_seq_length = 256  # Max token length per input (the higher the number, the long
er it takes)

# Early stopping helps prevent overfitting by stopping training
# when validation loss stops improving
model_args.use_early_stopping = True
model_args.early_stopping_delta = 0.01  # Minimum improvement in loss required to continue
training
model_args.early_stopping_metric = "eval_loss"  # The metric to monitor
model_args.early_stopping_metric_minimize = True  # Lower eval_loss is better
model_args.early_stopping_patience = 2  # Stop training if no improvement in 2 evaluations

# Run validation every 32 training steps to track progress
model_args.evaluate_during_training_steps = 32

# Change output directory to be inside Google Drive
model_args.output_dir = "/content/drive/MyDrive/outputs"
model_args.best_model_dir = "/content/drive/MyDrive/outputs/best_model"
```

In [ ]:

```
model = ClassificationModel(
    model_type = "roberta",
    model_name = "roberta-large",
    num_labels = 4,
    args = model_args,
    use_cuda = torch.cuda.is_available(),
)

# Preview the parameters of the model
print("\n".join(str(model.args).split(",")))
```

```
/usr/local/lib/python3.11/dist-packages/huggingface_hub/utils/_auth.py:94: UserWarning:
The secret `HF_TOKEN` does not exist in your Colab secrets.
To authenticate with the Hugging Face Hub, create a token in your settings tab (https://huggin
gface.co/settings/tokens), set it as secret in your Google Colab and restart your session.
You will be able to reuse this secret in all of your notebooks.
Please note that authentication is recommended but still optional to access public models or d
atasets.
  warnings.warn(
Some weights of RobertaForSequenceClassification were not initialized from the model checkpoin
t at roberta-large and are newly initialized: ['classifier.dense.bias', 'classifier.dense.weig
ht', 'classifier.out_proj.bias', 'classifier.out_proj.weight']
You should probably TRAIN this model on a down-stream task to be able to use it for prediction
s and inference.
```

```
ClassificationArgs(adafactor_beta1=None
 adafactor_clip_threshold=1.0
 adafactor_decay_rate=-0.8
 adafactor_eps=(1e-30
 0.001)
 adafactor_relative_step=True
 adafactor_scale_parameter=True
 adafactor_warmup_init=True
 adam_betas=(0.9
 0.999)
```

```
adam_epsilon=1e-08
best_model_dir='/content/drive/MyDrive/outputs/best_model'
cache_dir='cache_dir/'
config={}
cosine_schedule_num_cycles=0.5
custom_layer_parameters=[]
custom_parameter_groups=[]
dataloader_num_workers=0
do_lower_case=False
dynamic_quantize=False
early_stopping_consider_epochs=False
early_stopping_delta=0.01
early_stopping_metric='eval_loss'
early_stopping_metric_minimize=True
early_stopping_patience=2
encoding=None
eval_batch_size=100
evaluate_during_training=True
evaluate_during_training_silent=True
evaluate_during_training_steps=32
evaluate_during_training_verbose=False
evaluate_each_epoch=True
fp16=True
gradient_accumulation_steps=1
learning_rate=4e-06
local_rank=-1
logging_steps=50
loss_type=None
loss_args={}
manual_seed=None
max_grad_norm=1.0
max_seq_length=256
model_name='roberta-large'
model_type='roberta'
multiprocessing_chunksize=-1
n_gpu=1
no_cache=False
no_save=False
not_saved_args=[]
num_train_epochs=10
optimizer='AdamW'
output_dir='/content/drive/MyDrive/outputs'
overwrite_output_dir=True
polynomial_decay_schedule_lr_end=1e-07
polynomial_decay_schedule_power=1.0
process_count=1
quantized_model=False
reprocess_input_data=True
save_best_model=True
save_eval_checkpoints=True
save_model_every_epoch=True
save_optimizer_and_scheduler=True
save_steps=2000
scheduler='linear_schedule_with_warmup'
silent=False
skip_special_tokens=True
tensorboard_dir=None
thread_count=None
tokenizer_name='roberta-large'
tokenizer_type=None
train_batch_size=32
train_custom_parameters_only=False
trust_remote_code=False
use_cached_eval_features=False
use_early_stopping=True
use_hf_datasets=False
use_multiprocessing=True
use_multiprocessing_for_evaluation=True
wandb_kwargs={}
wandb_project=None
warmup_ratio=0.06
```

```
 warmup_steps=0
 weight_decay=0.0
 model_class='ClassificationModel'
 labels_list=[0
 1
 2
 3]
 labels_map={}
 lazy_delimiter='\t'
 lazy_labels_column=1
 lazy_loading=False
 lazy_loading_start_line=1
 lazy_text_a_column=None
 lazy_text_b_column=None
 lazy_text_column=0
 onnx=False
 regression=False
 sliding_window=False
 special_tokens_list=[]
 stride=0.8
 tie_value=1)
```

In [ ]:

```
# Train the model
training_results = model.train_model(train_df, eval_df = dev_df)

history = training_results[1]
```

```
/usr/local/lib/python3.11/dist-
packages/simpletransformers/classification/classification_model.py:882: FutureWarning: `torch.
cuda.amp.GradScaler(args...)` is deprecated. Please use `torch.amp.GradScaler('cuda', args...)
` instead.
  scaler = amp.GradScaler()
```

```
/usr/local/lib/python3.11/dist-
packages/simpletransformers/classification/classification_model.py:905: FutureWarning: `torch.
cuda.amp.autocast(args...)` is deprecated. Please use `torch.amp.autocast('cuda', args...)` in
stead.
  with amp.autocast():
```

```
/usr/local/lib/python3.11/dist-
packages/simpletransformers/classification/classification_model.py:1505: FutureWarning: `torch
.cuda.amp.autocast(args...)` is deprecated. Please use `torch.amp.autocast('cuda', args...)` i
nstead.
  with amp.autocast():
/usr/local/lib/python3.11/dist-
packages/simpletransformers/classification/classification_model.py:905: FutureWarning: `torch.
cuda.amp.autocast(args...)` is deprecated. Please use `torch.amp.autocast('cuda', args...)` in
stead.
  with amp.autocast():
```

```
/usr/local/lib/python3.11/dist-
packages/simpletransformers/classification/classification_model.py:1505: FutureWarning: `torch
.cuda.amp.autocast(args...)` is deprecated. Please use `torch.amp.autocast('cuda', args...)` i
nstead.
  with amp.autocast():
```

```
/usr/local/lib/python3.11/dist-
packages/simpletransformers/classification/classification_model.py:1505: FutureWarning: `torch
.cuda.amp.autocast(args...)` is deprecated. Please use `torch.amp.autocast('cuda', args...)` i
nstead.
  with amp.autocast():
```

```
/usr/local/lib/python3.11/dist-
packages/simpletransformers/classification/classification_model.py:905: FutureWarning: `torch.
cuda.amp.autocast(args...)` is deprecated. Please use `torch.amp.autocast('cuda', args...)` in
stead.
  with amp.autocast():
```

```
  with amp.autocast():
```

```
/usr/local/lib/python3.11/dist-
packages/simpletransformers/classification/classification_model.py:1505: FutureWarning: `torch
.cuda.amp.autocast(args...)` is deprecated. Please use `torch.amp.autocast('cuda', args...)` i
nstead.
  with amp.autocast():
/usr/local/lib/python3.11/dist-
packages/simpletransformers/classification/classification_model.py:905: FutureWarning: `torch.
cuda.amp.autocast(args...)` is deprecated. Please use `torch.amp.autocast('cuda', args...)` in
stead.
  with amp.autocast():
```

```
/usr/local/lib/python3.11/dist-
packages/simpletransformers/classification/classification_model.py:1505: FutureWarning: `torch
.cuda.amp.autocast(args...)` is deprecated. Please use `torch.amp.autocast('cuda', args...)` i
nstead.
  with amp.autocast():
```

```
/usr/local/lib/python3.11/dist-
packages/simpletransformers/classification/classification_model.py:1505: FutureWarning: `torch
.cuda.amp.autocast(args...)` is deprecated. Please use `torch.amp.autocast('cuda', args...)` i
nstead.
  with amp.autocast():
```

```
/usr/local/lib/python3.11/dist-
packages/simpletransformers/classification/classification_model.py:905: FutureWarning: `torch.
cuda.amp.autocast(args...)` is deprecated. Please use `torch.amp.autocast('cuda', args...)` in
stead.
  with amp.autocast():
```

```
/usr/local/lib/python3.11/dist-
packages/simpletransformers/classification/classification_model.py:1505: FutureWarning: `torch
.cuda.amp.autocast(args...)` is deprecated. Please use `torch.amp.autocast('cuda', args...)` i
nstead.
  with amp.autocast():
/usr/local/lib/python3.11/dist-
packages/simpletransformers/classification/classification_model.py:905: FutureWarning: `torch.
cuda.amp.autocast(args...)` is deprecated. Please use `torch.amp.autocast('cuda', args...)` in
stead.
  with amp.autocast():
```

```
/usr/local/lib/python3.11/dist-
packages/simpletransformers/classification/classification_model.py:1505: FutureWarning: `torch
.cuda.amp.autocast(args...)` is deprecated. Please use `torch.amp.autocast('cuda', args...)` i
nstead.
  with amp.autocast():
```

```
/usr/local/lib/python3.11/dist-
packages/simpletransformers/classification/classification_model.py:1505: FutureWarning: `torch
.cuda.amp.autocast(args...)` is deprecated. Please use `torch.amp.autocast('cuda', args...)` i
nstead.
  with amp.autocast():
```

```
/usr/local/lib/python3.11/dist-
packages/simpletransformers/classification/classification_model.py:905: FutureWarning: `torch.
cuda.amp.autocast(args...)` is deprecated. Please use `torch.amp.autocast('cuda', args...)` in
stead.
  with amp.autocast():
```

```
/usr/local/lib/python3.11/dist-
packages/simpletransformers/classification/classification_model.py:1505: FutureWarning: `torch
.cuda.amp.autocast(args...)` is deprecated. Please use `torch.amp.autocast('cuda', args...)` i
nstead.
  with amp.autocast():
/usr/local/lib/python3.11/dist-
packages/simpletransformers/classification/classification_model.py:905: FutureWarning: `torch.
cuda.amp.autocast(args...)` is deprecated. Please use `torch.amp.autocast('cuda', args...)` in
stead.
```

```
  with amp.autocast():
```

```
/usr/local/lib/python3.11/dist-
packages/simpletransformers/classification/classification_model.py:1505: FutureWarning: `torch
.cuda.amp.autocast(args...)` is deprecated. Please use `torch.amp.autocast('cuda', args...)` i
nstead.
  with amp.autocast():
```

```
/usr/local/lib/python3.11/dist-
packages/simpletransformers/classification/classification_model.py:1505: FutureWarning: `torch
.cuda.amp.autocast(args...)` is deprecated. Please use `torch.amp.autocast('cuda', args...)` i
nstead.
  with amp.autocast():
```

```
/usr/local/lib/python3.11/dist-
packages/simpletransformers/classification/classification_model.py:905: FutureWarning: `torch.
cuda.amp.autocast(args...)` is deprecated. Please use `torch.amp.autocast('cuda', args...)` in
stead.
  with amp.autocast():
```

```
/usr/local/lib/python3.11/dist-
packages/simpletransformers/classification/classification_model.py:1505: FutureWarning: `torch
.cuda.amp.autocast(args...)` is deprecated. Please use `torch.amp.autocast('cuda', args...)` i
nstead.
  with amp.autocast():
/usr/local/lib/python3.11/dist-
packages/simpletransformers/classification/classification_model.py:905: FutureWarning: `torch.
cuda.amp.autocast(args...)` is deprecated. Please use `torch.amp.autocast('cuda', args...)` in
stead.
  with amp.autocast():
```

```
/usr/local/lib/python3.11/dist-
packages/simpletransformers/classification/classification_model.py:1505: FutureWarning: `torch
.cuda.amp.autocast(args...)` is deprecated. Please use `torch.amp.autocast('cuda', args...)` i
nstead.
  with amp.autocast():
```

```
/usr/local/lib/python3.11/dist-
packages/simpletransformers/classification/classification_model.py:1505: FutureWarning: `torch
.cuda.amp.autocast(args...)` is deprecated. Please use `torch.amp.autocast('cuda', args...)` i
nstead.
  with amp.autocast():
```

```
/usr/local/lib/python3.11/dist-
packages/simpletransformers/classification/classification_model.py:905: FutureWarning: `torch.
cuda.amp.autocast(args...)` is deprecated. Please use `torch.amp.autocast('cuda', args...)` in
stead.
  with amp.autocast():
```

```
/usr/local/lib/python3.11/dist-
packages/simpletransformers/classification/classification_model.py:1505: FutureWarning: `torch
.cuda.amp.autocast(args...)` is deprecated. Please use `torch.amp.autocast('cuda', args...)` i
nstead.
  with amp.autocast():
/usr/local/lib/python3.11/dist-
packages/simpletransformers/classification/classification_model.py:905: FutureWarning: `torch.
cuda.amp.autocast(args...)` is deprecated. Please use `torch.amp.autocast('cuda', args...)` in
stead.
  with amp.autocast():
```

```
/usr/local/lib/python3.11/dist-
packages/simpletransformers/classification/classification_model.py:1505: FutureWarning: `torch
.cuda.amp.autocast(args...)` is deprecated. Please use `torch.amp.autocast('cuda', args...)` i
nstead.
  with amp.autocast():
```

```
/usr/local/lib/python3.11/dist-
packages/simpletransformers/classification/classification_model.py:1505: FutureWarning: `torch
.cuda.amp.autocast(args...)` is deprecated. Please use `torch.amp.autocast('cuda', args...)` i
```

```
nstead.
  with amp.autocast():
```

```
/usr/local/lib/python3.11/dist-packages/simpletransformers/classification/classification_model.py:905: FutureWarning: `torch.cuda.amp.autocast(args...)` is deprecated. Please use `torch.amp.autocast('cuda', args...)` instead.
  with amp.autocast():
```

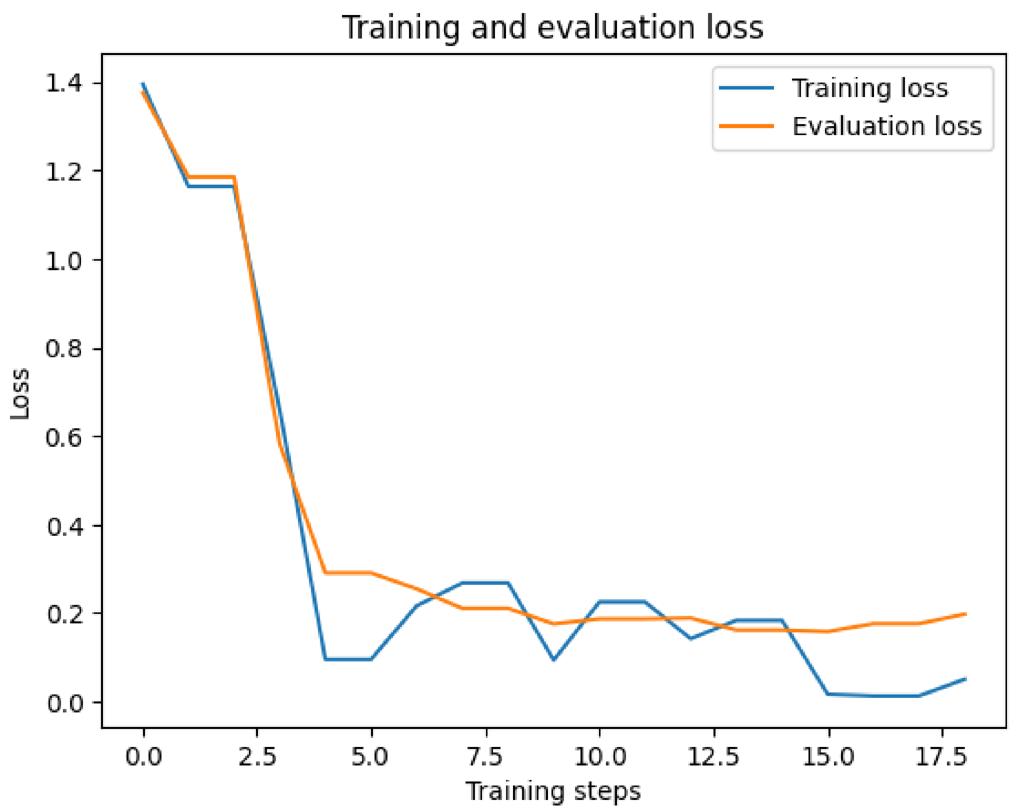
```
/usr/local/lib/python3.11/dist-packages/simpletransformers/classification/classification_model.py:1505: FutureWarning: `torch.cuda.amp.autocast(args...)` is deprecated. Please use `torch.amp.autocast('cuda', args...)` instead.
  with amp.autocast():
```

In [11]:

```
# Training and evaluation loss
train_loss = history['train_loss']
eval_loss = history['eval_loss']
plt.plot(train_loss, label='Training loss')
plt.plot(eval_loss, label='Evaluation loss')
plt.title('Training and evaluation loss')
plt.xlabel('Training steps')
plt.ylabel('Loss')
plt.legend()
```

Out[11]:

```
<matplotlib.legend.Legend at 0x7ef7153f6f50>
```



In [12]:

```
# Evaluate the model
predicted, probabilities = model.predict(test_df.text.to_list())
test_df_copy = test_df.copy()
test_df_copy['predicted'] = predicted
test_df_copy.head(10)
```

```
/usr/local/lib/python3.11/dist-packages/simpletransformers/classification/classification_model.py:2188: FutureWarning: `torch.cuda.amp.autocast(args...)` is deprecated. Please use `torch.amp.autocast('cuda', args...)` instead.
```

```
    with amp.autocast():
```

Out[12]:

	text	labels	predicted
0	\nAnd guess who's here in your place.\n\nPleas...	1	1
1	Does anyone know if any of Currier and Ives et...	1	1
2	=FLAME ON\n=\n=Reading through the posts about...	2	2
3	\nBut in this case I said I hoped that BCCI wa...	0	0
4	\nIn the kind I have made I used a Lite sour c...	2	2
5	\n\n\tThe "R Us" is not trademarked, but the "...	0	0
6	\nI think you must have the same hygiene teach...	2	2
7	\n\nIt may be a good way to catch a cold. It'...	2	2
8	Archive-name: graphics/resources-list/part2\nL...	1	1
9	can someone tell me where i could find ansi or...	1	1

In [13]:

```
# Generate classification report
print(classification_report(test_df_copy['labels'], test_df_copy['predicted']))
```

```
              precision    recall  f1-score   support

           0       0.87      0.82      0.84       319
           1       0.82      0.95      0.88       389
           2       0.93      0.88      0.91       396
           3       0.89      0.85      0.87       394

    accuracy                           0.88      1498
   macro avg       0.88      0.87      0.88      1498
weighted avg       0.88      0.88      0.88      1498
```

Comparison of Fine-tuned RoBERTa to Fine-Tuned BERT

BERT classification report:

```
              precision    recall  f1-score   support

           0       0.84      0.80      0.82       319
           1       0.89      0.91      0.90       389
           2       0.94      0.88      0.91       396
           3       0.79      0.86      0.82       394

    accuracy                           0.86      1498
   macro avg       0.87      0.86      0.86      1498
weighted avg       0.87      0.86      0.86      1498
```

In comparison to the model presented in **Lab6.4-Topic-classification-BERT.ipynb**, the RoBERTa-based approach achieves a consistently higher performance across nearly every metric:

- **Overall Accuracy** rises from **0.86** to **0.88**.
- **Precision** gets slight gains in most classes, showing fewer false positives.
- **Recall** shows a strong improvement, especially for Class 1 (reaching **0.95**), meaning fewer false negatives.
- **F1-scores** follow an upward trend, reflecting an overall enhancement in the model's classification capabilities.

These improvements show RoBERTa's greater capacity for contextual understanding. Where BERT may have missed subtle semantic cues, RoBERTa's transformer-based embeddings help it differentiate among topics more accurately.

thereby increasing its **Precision, Recall,** and **F1-scores**.

Per Topic Comparisons

Below is a per class breakdown of **Precision, Recall,** and **F1** scores for both models:

Class	BERT (Precision, Recall, F1-score)	RoBERTa (Precision, Recall, F1-score)
0	(0.84, 0.80, 0.82)	(0.87, 0.82, 0.84)
1	(0.89, 0.91, 0.90)	(0.82, 0.95, 0.88)
2	(0.94, 0.88, 0.91)	(0.93, 0.88, 0.91)
3	(0.79, 0.86, 0.82)	(0.89, 0.85, 0.87)

1. **Class 0**
   RoBERTa reduces false positives more effectively (Precision **0.87** compared to **0.84**), while getting a higher proportion of true instances (Recall **0.82** compared to **0.80**). This leads to increasing its F1-score to **0.84** as well.
2. **Class 1**
   RoBERTa gets a lower precision (**0.82** compared to **0.89**), while getting a higher proportion of true instances (Recall **0.95** compared to **0.91**). This leads to an F1-score of **0.88**.
3. **Class 2**
   RoBERTa exhibits a drop in precision (**0.93** compared to **0.94**), while retaining the same recall (**0.88**). This maintains an overall F1-score of **0.91**, matching BERT's performance.
4. **Class 3**
   RoBERTa raises precision (**0.89** compared to **0.79**), while nearly matching the recall (0.85 vs. 0.86). This leads to an improved F1-score of **0.87**.

Overall, these findings highlight RoBERTa's refined ability to disambiguate among classes, reducing errors, improving classifications, and getting higher performance metrics across different set of topics compared to BERT.

4. Conventional ML Approach: SVM with Bag-of-Words

In [6]:

```
# Merging train and validation sets from above as validation set isn't needed with SVM/NB
X_train = pd.concat([train_df["text"], dev_df["text"]])
X_test = test_df["text"]
y_train = pd.concat([train_df["labels"], dev_df["labels"]])
y_test = test_df["labels"]
```

In [ ]:

```
# BoW
vectorizer = CountVectorizer(stop_words="english")  # By default (when no tokenizer is given), CountVectorizer uses a pattern to tokenize words (r"(?u)\b\w\w+\b") that excludes punctuation and words shorter than 2 letters.
                                                    # It also lowercases all words by default. Hence, we decided that there is no need to set tokenizer=nltk.word_tokenize. When we tested the two options, they returned identical results.
X_train_bow = vectorizer.fit_transform(X_train)
X_test_bow = vectorizer.transform(X_test)

# Initialize and train the SVM
svm_model = svm.LinearSVC(max_iter=20000)  # Increase the max iterations to remove convergence warning
svm_model.fit(X_train_bow, y_train)

# Evaluate the model
y_pred = svm_model.predict(X_test_bow)
print(classification_report(y_test, y_pred, target_names=categories))
```

```
               precision    recall  f1-score   support

  alt.atheism       0.73      0.72      0.73       319
comp.graphics       0.73      0.87      0.79       389
      sci.med       0.81      0.71      0.75       396
```

```
     sci.med       0.81      0.71      0.75       396
   sci.space       0.75      0.71      0.73       394

    accuracy                           0.75      1498
   macro avg       0.75      0.75      0.75      1498
weighted avg       0.76      0.75      0.75      1498
```

Comparison of Fine-Tuned RoBERTa to SVM with Bag-of-Words Representation

Overall Comparison

To compare the fine-tuned RoBERTa model to SVM with Bag-of-Words representation, we can firstly look at the overall accuracy. The accuracy of RoBERTa is significantly higher than the accuracy of SVM (0.88 vs 0.75) --> RoBERTa outperforms SVM with BoW by 13%.

The precision, recall and F1-score are also higher for RoBERTa:

- **Precision: RoBERTa exhibits higher precision than SVM (0.88 vs 0.75) --> there are fewer false positives;**
- **Recall: RoBERTa outperforms SVM (0.87 vs 0.75) --> RoBERTa is better at capturing actual positive cases;**
- **F1-score: RoBERTa is also better than SVM (0.88 vs 0.75) --> overall better results and balance between precision and recall.**

Therefore, RoBERTa significantly outperforms SVM with BoW representation in all major metrics (accuracy, precision, recall, f1 score).

Per Topic Comparisons

Below is a per class breakdown of Precision, Recall, and F1-score for both models:

Class	SVM (Precision, Recall, F1-score)	RoBERTa (Precision, Recall, F1-score)
0	(0.73, 0.72, 0.73)	(0.87, 0.82, 0.84)
1	(0.73, 0.87, 0.79)	(0.82, 0.95, 0.88)
2	(0.81, 0.71, 0.75)	(0.93, 0.88, 0.91)
3	(0.75, 0.71, 0.73)	(0.89, 0.85, 0.87)

1. **Class 0 (alt.atheism)**
RoBERTa outperforms SVM in all metrics, showing particulary high numbers in precision and F1-score. The difference in precision of 14% means that RoBERTa is better at reducing false positives --> it is more accurate in distinguishing `alt.atheism` from other classes. The higher recall for RoBERTa means that it captures a bigger part of relevant instances, and the SVM might miss more cases.
2. **Class 1 (comp.graphics)**
For this class, RoBERTa outperforms SVM with BoW, but the difference is less harsh compared to class 0. RoBERTa has a precision of 0.82 and a recall of 0.95. SVM with BoW has a precision of 0.73 and a recall of 0.87. Important thing to notice here is the difference in recall values. RoBERTa is capturing nearly all relevant instances of `comp.graphics` (recall is 0.95), while SVM is missing many of them. However, the precision for RoBERTa compared to recall for RoBERTa is lower. This means that while it is retrieving more instances correctly, it may also be introducing slightly more false positives.
3. **Class 2 (sci.med)**
Again, RoBERTa is performing better than SVM. In this class, especially, it shows drastically higher results. The major difference in recall highlights that SVM is missing larger number of `sci.med` instances. The precision difference further suggests that SVM is more prone to misclassifying texts from other categories --> higher rate of false positives compared to RoBERTa. This considerable difference and strong performance is likely due to RoBERTa's ability to capture nuanced medical terminology and contextual relationships, which SVM with BoW struggles with.
4. **Class 3 (sci.space)**
For class 3, RoBERTa performs better than SVM, however, the gap is smaller than for class 2. RoBERTa's higher precision (0.89 vs 0.75) means it makes fewer false positives, and its better recall (0.85 vs 0.71) shows it is capable of correct identification of instances from this class. Therefore, we can obvserve that RoBERTa can understand scientific language better than SVM, which is relying on more simple word patterns.

Conclusion (RoBERTa vs SVM with BoW)

The fine-tuned RoBERTa model significantly outperforms SVM with Bag-of-Words across all evaluation metrics, including accuracy, precision, recall, and F1-score. This advantage is evident both in the overall performance and within each individual class, where RoBERTa consistently demonstrates better precision in reducing false positives and higher recall in capturing relevant instances. Its ability to capture complex context makes it especially effective for specialized topics like `sci.med`, where SVM struggles. Thus, the results suggest that RoBERTa's is better for the topic modelling task than simpler models like SVM with BoW.

All Models Conclusion

In conclusion, RoBERTa consistently outperforms both fine-tuned BERT and SVM with Bag-of-Words, demonstrating superior performance across all key metrics. Compared to SVM, RoBERTa achieves a 13% higher accuracy (0.88 vs. 0.75), along with significant improvements in precision, recall, and F1-score, which showcases its ability to capture contextual relationships more effectively than the frequency-based approach of SVM with BoW. While the improvements over BERT are more subtle, RoBERTa still achieves higher recall and F1-scores, particularly demonstrating strength in reducing false negatives, which improves its accuracy in topic classification. These findings highlight RoBERTa's greater capacity for semantic understanding and differentiation of context, making it the most effective model for the task of topic modelling.