Rahul_223742152_SIT764_Senti_analysis

May 18, 2025

[2]:
```
import os
os.chdir('/content/drive/MyDrive/SIT764/')
```

[ ]:
```
# requirements = """
# bitsandbytes==0.45.5
# datasets==3.6.0
# gradio==5.29.0
# gradio_client==1.10.0
# peft==0.15.2
# scikit-learn==1.6.1
# sentence-transformers==3.4.1
# tensorflow-datasets==4.9.8
# torch @ https://download.pytorch.org/whl/cu124/torch-2.6.
 ↪0%2Bcu124-cp311-cp311-linux_x86_64.whl
# torchaudio @ https://download.pytorch.org/whl/cu124/torchaudio-2.6.
 ↪0%2Bcu124-cp311-cp311-linux_x86_64.whl
# torchsummary==1.5.1
# torchvision @ https://download.pytorch.org/whl/cu124/torchvision-0.21.
 ↪0%2Bcu124-cp311-cp311-linux_x86_64.whl
# transformers==4.48.3
# vega-datasets==0.9.0
# lime==0.2.0.1
# """

# with open("requirements.txt", "w") as f:
#     f.write(requirements.strip())
```

[3]:
```
!pip install -r requirements.txt
```

```
Collecting torch@ https://download.pytorch.org/whl/cu124/torch-2.6.0%2Bcu124-
cp311-cp311-linux_x86_64.whl (from -r requirements.txt (line 9))
  Downloading https://download.pytorch.org/whl/cu124/torch-2.6.0%2Bcu124-cp311-
cp311-linux_x86_64.whl (768.5 MB)
                           768.5/768.5
MB 1.4 MB/s eta 0:00:00
Collecting torchaudio@ https://download.pytorch.org/whl/cu124/torchaudio-
2.6.0%2Bcu124-cp311-cp311-linux_x86_64.whl (from -r requirements.txt (line 10))
  Downloading https://download.pytorch.org/whl/cu124/torchaudio-2.6.0%2Bcu124-
```

```
cp311-cp311-linux_x86_64.whl (3.4 MB)
                           3.4/3.4 MB
105.8 MB/s eta 0:00:00
Collecting torchvision@ https://download.pytorch.org/whl/cu124/torchvision
-0.21.0%2Bcu124-cp311-cp311-linux_x86_64.whl (from -r requirements.txt (line
12))
  Downloading https://download.pytorch.org/whl/cu124/torchvision-0.21.0%2Bcu124-
cp311-cp311-linux_x86_64.whl (7.3 MB)
                           7.3/7.3 MB
130.9 MB/s eta 0:00:00
Collecting bitsandbytes==0.45.5 (from -r requirements.txt (line 1))
  Downloading bitsandbytes-0.45.5-py3-none-manylinux_2_24_x86_64.whl.metadata
(5.0 kB)
Collecting datasets==3.6.0 (from -r requirements.txt (line 2))
  Downloading datasets-3.6.0-py3-none-any.whl.metadata (19 kB)
Collecting gradio==5.29.0 (from -r requirements.txt (line 3))
  Downloading gradio-5.29.0-py3-none-any.whl.metadata (16 kB)
Collecting gradio_client==1.10.0 (from -r requirements.txt (line 4))
  Downloading gradio_client-1.10.0-py3-none-any.whl.metadata (7.1 kB)
Requirement already satisfied: peft==0.15.2 in /usr/local/lib/python3.11/dist-
packages (from -r requirements.txt (line 5)) (0.15.2)
Requirement already satisfied: scikit-learn==1.6.1 in
/usr/local/lib/python3.11/dist-packages (from -r requirements.txt (line 6))
(1.6.1)
Collecting sentence-transformers==3.4.1 (from -r requirements.txt (line 7))
  Downloading sentence_transformers-3.4.1-py3-none-any.whl.metadata (10 kB)
Requirement already satisfied: tensorflow-datasets==4.9.8 in
/usr/local/lib/python3.11/dist-packages (from -r requirements.txt (line 8))
(4.9.8)
Requirement already satisfied: torchsummary==1.5.1 in
/usr/local/lib/python3.11/dist-packages (from -r requirements.txt (line 11))
(1.5.1)
Collecting transformers==4.48.3 (from -r requirements.txt (line 13))
  Downloading transformers-4.48.3-py3-none-any.whl.metadata (44 kB)
                           44.4/44.4 kB
2.1 MB/s eta 0:00:00
Requirement already satisfied: vega-datasets==0.9.0 in
/usr/local/lib/python3.11/dist-packages (from -r requirements.txt (line 14))
(0.9.0)
Collecting lime==0.2.0.1 (from -r requirements.txt (line 15))
  Downloading lime-0.2.0.1.tar.gz (275 kB)
                           275.7/275.7
kB 9.2 MB/s eta 0:00:00
  Preparing metadata (setup.py) … done
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.11/dist-
packages (from bitsandbytes==0.45.5->-r requirements.txt (line 1)) (2.0.2)
Requirement already satisfied: filelock in /usr/local/lib/python3.11/dist-
```

```
packages (from datasets==3.6.0->-r requirements.txt (line 2)) (3.18.0)
Requirement already satisfied: pyarrow>=15.0.0 in
/usr/local/lib/python3.11/dist-packages (from datasets==3.6.0->-r
requirements.txt (line 2)) (18.1.0)
Requirement already satisfied: dill<0.3.9,>=0.3.0 in
/usr/local/lib/python3.11/dist-packages (from datasets==3.6.0->-r
requirements.txt (line 2)) (0.3.7)
Requirement already satisfied: pandas in /usr/local/lib/python3.11/dist-packages
(from datasets==3.6.0->-r requirements.txt (line 2)) (2.2.2)
Requirement already satisfied: requests>=2.32.2 in
/usr/local/lib/python3.11/dist-packages (from datasets==3.6.0->-r
requirements.txt (line 2)) (2.32.3)
Requirement already satisfied: tqdm>=4.66.3 in /usr/local/lib/python3.11/dist-
packages (from datasets==3.6.0->-r requirements.txt (line 2)) (4.67.1)
Requirement already satisfied: xxhash in /usr/local/lib/python3.11/dist-packages
(from datasets==3.6.0->-r requirements.txt (line 2)) (3.5.0)
Requirement already satisfied: multiprocess<0.70.17 in
/usr/local/lib/python3.11/dist-packages (from datasets==3.6.0->-r
requirements.txt (line 2)) (0.70.15)
Collecting fsspec<=2025.3.0,>=2023.1.0 (from
fsspec[http]<=2025.3.0,>=2023.1.0->datasets==3.6.0->-r requirements.txt (line
2))
  Downloading fsspec-2025.3.0-py3-none-any.whl.metadata (11 kB)
Requirement already satisfied: huggingface-hub>=0.24.0 in
/usr/local/lib/python3.11/dist-packages (from datasets==3.6.0->-r
requirements.txt (line 2)) (0.31.2)
Requirement already satisfied: packaging in /usr/local/lib/python3.11/dist-
packages (from datasets==3.6.0->-r requirements.txt (line 2)) (24.2)
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.11/dist-
packages (from datasets==3.6.0->-r requirements.txt (line 2)) (6.0.2)
Collecting aiofiles<25.0,>=22.0 (from gradio==5.29.0->-r requirements.txt (line
3))
  Downloading aiofiles-24.1.0-py3-none-any.whl.metadata (10 kB)
Requirement already satisfied: anyio<5.0,>=3.0 in
/usr/local/lib/python3.11/dist-packages (from gradio==5.29.0->-r
requirements.txt (line 3)) (4.9.0)
Collecting fastapi<1.0,>=0.115.2 (from gradio==5.29.0->-r requirements.txt (line
3))
  Downloading fastapi-0.115.12-py3-none-any.whl.metadata (27 kB)
Collecting ffmpy (from gradio==5.29.0->-r requirements.txt (line 3))
  Downloading ffmpy-0.5.0-py3-none-any.whl.metadata (3.0 kB)
Collecting groovy~=0.1 (from gradio==5.29.0->-r requirements.txt (line 3))
  Downloading groovy-0.1.2-py3-none-any.whl.metadata (6.1 kB)
Requirement already satisfied: httpx>=0.24.1 in /usr/local/lib/python3.11/dist-
packages (from gradio==5.29.0->-r requirements.txt (line 3)) (0.28.1)
Requirement already satisfied: jinja2<4.0 in /usr/local/lib/python3.11/dist-
packages (from gradio==5.29.0->-r requirements.txt (line 3)) (3.1.6)
Requirement already satisfied: markupsafe<4.0,>=2.0 in
```

```
/usr/local/lib/python3.11/dist-packages (from gradio==5.29.0->-r
requirements.txt (line 3)) (3.0.2)
Requirement already satisfied: orjson~=3.0 in /usr/local/lib/python3.11/dist-
packages (from gradio==5.29.0->-r requirements.txt (line 3)) (3.10.18)
Requirement already satisfied: pillow<12.0,>=8.0 in
/usr/local/lib/python3.11/dist-packages (from gradio==5.29.0->-r
requirements.txt (line 3)) (11.2.1)
Requirement already satisfied: pydantic<2.12,>=2.0 in
/usr/local/lib/python3.11/dist-packages (from gradio==5.29.0->-r
requirements.txt (line 3)) (2.11.4)
Collecting pydub (from gradio==5.29.0->-r requirements.txt (line 3))
  Downloading pydub-0.25.1-py2.py3-none-any.whl.metadata (1.4 kB)
Collecting python-multipart>=0.0.18 (from gradio==5.29.0->-r requirements.txt
(line 3))
  Downloading python_multipart-0.0.20-py3-none-any.whl.metadata (1.8 kB)
Collecting ruff>=0.9.3 (from gradio==5.29.0->-r requirements.txt (line 3))
  Downloading ruff-0.11.10-py3-none-
manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (25 kB)
Collecting safehttpx<0.2.0,>=0.1.6 (from gradio==5.29.0->-r requirements.txt
(line 3))
  Downloading safehttpx-0.1.6-py3-none-any.whl.metadata (4.2 kB)
Collecting semantic-version~=2.0 (from gradio==5.29.0->-r requirements.txt (line
3))
  Downloading semantic_version-2.10.0-py2.py3-none-any.whl.metadata (9.7 kB)
Collecting starlette<1.0,>=0.40.0 (from gradio==5.29.0->-r requirements.txt
(line 3))
  Downloading starlette-0.46.2-py3-none-any.whl.metadata (6.2 kB)
Collecting tomlkit<0.14.0,>=0.12.0 (from gradio==5.29.0->-r requirements.txt
(line 3))
  Downloading tomlkit-0.13.2-py3-none-any.whl.metadata (2.7 kB)
Requirement already satisfied: typer<1.0,>=0.12 in
/usr/local/lib/python3.11/dist-packages (from gradio==5.29.0->-r
requirements.txt (line 3)) (0.15.3)
Requirement already satisfied: typing-extensions~=4.0 in
/usr/local/lib/python3.11/dist-packages (from gradio==5.29.0->-r
requirements.txt (line 3)) (4.13.2)
Collecting uvicorn>=0.14.0 (from gradio==5.29.0->-r requirements.txt (line 3))
  Downloading uvicorn-0.34.2-py3-none-any.whl.metadata (6.5 kB)
Requirement already satisfied: websockets<16.0,>=10.0 in
/usr/local/lib/python3.11/dist-packages (from gradio_client==1.10.0->-r
requirements.txt (line 4)) (15.0.1)
Requirement already satisfied: psutil in /usr/local/lib/python3.11/dist-packages
(from peft==0.15.2->-r requirements.txt (line 5)) (5.9.5)
Requirement already satisfied: accelerate>=0.21.0 in
/usr/local/lib/python3.11/dist-packages (from peft==0.15.2->-r requirements.txt
(line 5)) (1.6.0)
Requirement already satisfied: safetensors in /usr/local/lib/python3.11/dist-
packages (from peft==0.15.2->-r requirements.txt (line 5)) (0.5.3)
```

```
Requirement already satisfied: scipy>=1.6.0 in /usr/local/lib/python3.11/dist-
packages (from scikit-learn==1.6.1->-r requirements.txt (line 6)) (1.15.3)
Requirement already satisfied: joblib>=1.2.0 in /usr/local/lib/python3.11/dist-
packages (from scikit-learn==1.6.1->-r requirements.txt (line 6)) (1.5.0)
Requirement already satisfied: threadpoolctl>=3.1.0 in
/usr/local/lib/python3.11/dist-packages (from scikit-learn==1.6.1->-r
requirements.txt (line 6)) (3.6.0)
Requirement already satisfied: absl-py in /usr/local/lib/python3.11/dist-
packages (from tensorflow-datasets==4.9.8->-r requirements.txt (line 8)) (1.4.0)
Requirement already satisfied: array_record>=0.5.0 in
/usr/local/lib/python3.11/dist-packages (from tensorflow-datasets==4.9.8->-r
requirements.txt (line 8)) (0.7.2)
Requirement already satisfied: dm-tree in /usr/local/lib/python3.11/dist-
packages (from tensorflow-datasets==4.9.8->-r requirements.txt (line 8)) (0.1.9)
Requirement already satisfied: etils>=1.9.1 in /usr/local/lib/python3.11/dist-
packages (from etils[edc,enp,epath,epy,etree]>=1.9.1; python_version >=
"3.11"->tensorflow-datasets==4.9.8->-r requirements.txt (line 8)) (1.12.2)
Requirement already satisfied: immutabledict in /usr/local/lib/python3.11/dist-
packages (from tensorflow-datasets==4.9.8->-r requirements.txt (line 8)) (4.2.1)
Requirement already satisfied: promise in /usr/local/lib/python3.11/dist-
packages (from tensorflow-datasets==4.9.8->-r requirements.txt (line 8)) (2.3)
Requirement already satisfied: protobuf>=3.20 in /usr/local/lib/python3.11/dist-
packages (from tensorflow-datasets==4.9.8->-r requirements.txt (line 8))
(5.29.4)
Requirement already satisfied: simple_parsing in /usr/local/lib/python3.11/dist-
packages (from tensorflow-datasets==4.9.8->-r requirements.txt (line 8)) (0.1.7)
Requirement already satisfied: tensorflow-metadata in
/usr/local/lib/python3.11/dist-packages (from tensorflow-datasets==4.9.8->-r
requirements.txt (line 8)) (1.17.1)
Requirement already satisfied: termcolor in /usr/local/lib/python3.11/dist-
packages (from tensorflow-datasets==4.9.8->-r requirements.txt (line 8)) (3.1.0)
Requirement already satisfied: toml in /usr/local/lib/python3.11/dist-packages
(from tensorflow-datasets==4.9.8->-r requirements.txt (line 8)) (0.10.2)
Requirement already satisfied: wrapt in /usr/local/lib/python3.11/dist-packages
(from tensorflow-datasets==4.9.8->-r requirements.txt (line 8)) (1.17.2)
Requirement already satisfied: regex!=2019.12.17 in
/usr/local/lib/python3.11/dist-packages (from transformers==4.48.3->-r
requirements.txt (line 13)) (2024.11.6)
Requirement already satisfied: tokenizers<0.22,>=0.21 in
/usr/local/lib/python3.11/dist-packages (from transformers==4.48.3->-r
requirements.txt (line 13)) (0.21.1)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.11/dist-
packages (from lime==0.2.0.1->-r requirements.txt (line 15)) (3.10.0)
Requirement already satisfied: scikit-image>=0.12 in
/usr/local/lib/python3.11/dist-packages (from lime==0.2.0.1->-r requirements.txt
(line 15)) (0.25.2)
Requirement already satisfied: networkx in /usr/local/lib/python3.11/dist-
packages (from torch@ https://download.pytorch.org/whl/cu124/torch-
```

```
2.6.0%2Bcu124-cp311-cp311-linux_x86_64.whl->-r requirements.txt (line 9))
(3.4.2)
Collecting nvidia-cuda-nvrtc-cu12==12.4.127 (from torch@ https://download.pytorc
h.org/whl/cu124/torch-2.6.0%2Bcu124-cp311-cp311-linux_x86_64.whl->-r
requirements.txt (line 9))
  Downloading nvidia_cuda_nvrtc_cu12-12.4.127-py3-none-
manylinux2014_x86_64.whl.metadata (1.5 kB)
Collecting nvidia-cuda-runtime-cu12==12.4.127 (from torch@ https://download.pyto
rch.org/whl/cu124/torch-2.6.0%2Bcu124-cp311-cp311-linux_x86_64.whl->-r
requirements.txt (line 9))
  Downloading nvidia_cuda_runtime_cu12-12.4.127-py3-none-
manylinux2014_x86_64.whl.metadata (1.5 kB)
Collecting nvidia-cuda-cupti-cu12==12.4.127 (from torch@ https://download.pytorc
h.org/whl/cu124/torch-2.6.0%2Bcu124-cp311-cp311-linux_x86_64.whl->-r
requirements.txt (line 9))
  Downloading nvidia_cuda_cupti_cu12-12.4.127-py3-none-
manylinux2014_x86_64.whl.metadata (1.6 kB)
Collecting nvidia-cudnn-cu12==9.1.0.70 (from torch@ https://download.pytorch.org
/whl/cu124/torch-2.6.0%2Bcu124-cp311-cp311-linux_x86_64.whl->-r requirements.txt
(line 9))
  Downloading nvidia_cudnn_cu12-9.1.0.70-py3-none-
manylinux2014_x86_64.whl.metadata (1.6 kB)
Collecting nvidia-cublas-cu12==12.4.5.8 (from torch@ https://download.pytorch.or
g/whl/cu124/torch-2.6.0%2Bcu124-cp311-cp311-linux_x86_64.whl->-r
requirements.txt (line 9))
  Downloading nvidia_cublas_cu12-12.4.5.8-py3-none-
manylinux2014_x86_64.whl.metadata (1.5 kB)
Collecting nvidia-cufft-cu12==11.2.1.3 (from torch@ https://download.pytorch.org
/whl/cu124/torch-2.6.0%2Bcu124-cp311-cp311-linux_x86_64.whl->-r requirements.txt
(line 9))
  Downloading nvidia_cufft_cu12-11.2.1.3-py3-none-
manylinux2014_x86_64.whl.metadata (1.5 kB)
Collecting nvidia-curand-cu12==10.3.5.147 (from torch@ https://download.pytorch.
org/whl/cu124/torch-2.6.0%2Bcu124-cp311-cp311-linux_x86_64.whl->-r
requirements.txt (line 9))
  Downloading nvidia_curand_cu12-10.3.5.147-py3-none-
manylinux2014_x86_64.whl.metadata (1.5 kB)
Collecting nvidia-cusolver-cu12==11.6.1.9 (from torch@ https://download.pytorch.
org/whl/cu124/torch-2.6.0%2Bcu124-cp311-cp311-linux_x86_64.whl->-r
requirements.txt (line 9))
  Downloading nvidia_cusolver_cu12-11.6.1.9-py3-none-
manylinux2014_x86_64.whl.metadata (1.6 kB)
Collecting nvidia-cusparse-cu12==12.3.1.170 (from torch@ https://download.pytorc
h.org/whl/cu124/torch-2.6.0%2Bcu124-cp311-cp311-linux_x86_64.whl->-r
requirements.txt (line 9))
  Downloading nvidia_cusparse_cu12-12.3.1.170-py3-none-
manylinux2014_x86_64.whl.metadata (1.6 kB)
Requirement already satisfied: nvidia-cusparselt-cu12==0.6.2 in
```

```
/usr/local/lib/python3.11/dist-packages (from torch@ https://download.pytorch.or
g/whl/cu124/torch-2.6.0%2Bcu124-cp311-cp311-linux_x86_64.whl->-r
requirements.txt (line 9)) (0.6.2)
Requirement already satisfied: nvidia-nccl-cu12==2.21.5 in
/usr/local/lib/python3.11/dist-packages (from torch@ https://download.pytorch.or
g/whl/cu124/torch-2.6.0%2Bcu124-cp311-cp311-linux_x86_64.whl->-r
requirements.txt (line 9)) (2.21.5)
Requirement already satisfied: nvidia-nvtx-cu12==12.4.127 in
/usr/local/lib/python3.11/dist-packages (from torch@ https://download.pytorch.or
g/whl/cu124/torch-2.6.0%2Bcu124-cp311-cp311-linux_x86_64.whl->-r
requirements.txt (line 9)) (12.4.127)
Collecting nvidia-nvjitlink-cu12==12.4.127 (from torch@ https://download.pytorch
.org/whl/cu124/torch-2.6.0%2Bcu124-cp311-cp311-linux_x86_64.whl->-r
requirements.txt (line 9))
  Downloading nvidia_nvjitlink_cu12-12.4.127-py3-none-
manylinux2014_x86_64.whl.metadata (1.5 kB)
Requirement already satisfied: triton==3.2.0 in /usr/local/lib/python3.11/dist-
packages (from torch@ https://download.pytorch.org/whl/cu124/torch-
2.6.0%2Bcu124-cp311-cp311-linux_x86_64.whl->-r requirements.txt (line 9))
(3.2.0)
Requirement already satisfied: sympy==1.13.1 in /usr/local/lib/python3.11/dist-
packages (from torch@ https://download.pytorch.org/whl/cu124/torch-
2.6.0%2Bcu124-cp311-cp311-linux_x86_64.whl->-r requirements.txt (line 9))
(1.13.1)
Requirement already satisfied: mpmath<1.4,>=1.1.0 in
/usr/local/lib/python3.11/dist-packages (from sympy==1.13.1->torch@ https://down
load.pytorch.org/whl/cu124/torch-2.6.0%2Bcu124-cp311-cp311-linux_x86_64.whl->-r
requirements.txt (line 9)) (1.3.0)
Requirement already satisfied: idna>=2.8 in /usr/local/lib/python3.11/dist-
packages (from anyio<5.0,>=3.0->gradio==5.29.0->-r requirements.txt (line 3))
(3.10)
Requirement already satisfied: sniffio>=1.1 in /usr/local/lib/python3.11/dist-
packages (from anyio<5.0,>=3.0->gradio==5.29.0->-r requirements.txt (line 3))
(1.3.1)
Requirement already satisfied: einops in /usr/local/lib/python3.11/dist-packages
(from etils[edc,enp,epath,epy,etree]>=1.9.1; python_version >=
"3.11"->tensorflow-datasets==4.9.8->-r requirements.txt (line 8)) (0.8.1)
Requirement already satisfied: importlib_resources in
/usr/local/lib/python3.11/dist-packages (from
etils[edc,enp,epath,epy,etree]>=1.9.1; python_version >= "3.11"->tensorflow-
datasets==4.9.8->-r requirements.txt (line 8)) (6.5.2)
Requirement already satisfied: zipp in /usr/local/lib/python3.11/dist-packages
(from etils[edc,enp,epath,epy,etree]>=1.9.1; python_version >=
"3.11"->tensorflow-datasets==4.9.8->-r requirements.txt (line 8)) (3.21.0)
Requirement already satisfied: aiohttp!=4.0.0a0,!=4.0.0a1 in
/usr/local/lib/python3.11/dist-packages (from
fsspec[http]<=2025.3.0,>=2023.1.0->datasets==3.6.0->-r requirements.txt (line
2)) (3.11.15)
```

```
Requirement already satisfied: certifi in /usr/local/lib/python3.11/dist-
packages (from httpx>=0.24.1->gradio==5.29.0->-r requirements.txt (line 3))
(2025.4.26)
Requirement already satisfied: httpcore==1.* in /usr/local/lib/python3.11/dist-
packages (from httpx>=0.24.1->gradio==5.29.0->-r requirements.txt (line 3))
(1.0.9)
Requirement already satisfied: h11>=0.16 in /usr/local/lib/python3.11/dist-
packages (from httpcore==1.*->httpx>=0.24.1->gradio==5.29.0->-r requirements.txt
(line 3)) (0.16.0)
Requirement already satisfied: python-dateutil>=2.8.2 in
/usr/local/lib/python3.11/dist-packages (from pandas->datasets==3.6.0->-r
requirements.txt (line 2)) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.11/dist-
packages (from pandas->datasets==3.6.0->-r requirements.txt (line 2)) (2025.2)
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.11/dist-
packages (from pandas->datasets==3.6.0->-r requirements.txt (line 2)) (2025.2)
Requirement already satisfied: annotated-types>=0.6.0 in
/usr/local/lib/python3.11/dist-packages (from
pydantic<2.12,>=2.0->gradio==5.29.0->-r requirements.txt (line 3)) (0.7.0)
Requirement already satisfied: pydantic-core==2.33.2 in
/usr/local/lib/python3.11/dist-packages (from
pydantic<2.12,>=2.0->gradio==5.29.0->-r requirements.txt (line 3)) (2.33.2)
Requirement already satisfied: typing-inspection>=0.4.0 in
/usr/local/lib/python3.11/dist-packages (from
pydantic<2.12,>=2.0->gradio==5.29.0->-r requirements.txt (line 3)) (0.4.0)
Requirement already satisfied: charset-normalizer<4,>=2 in
/usr/local/lib/python3.11/dist-packages (from
requests>=2.32.2->datasets==3.6.0->-r requirements.txt (line 2)) (3.4.2)
Requirement already satisfied: urllib3<3,>=1.21.1 in
/usr/local/lib/python3.11/dist-packages (from
requests>=2.32.2->datasets==3.6.0->-r requirements.txt (line 2)) (2.4.0)
Requirement already satisfied: imageio!=2.35.0,>=2.33 in
/usr/local/lib/python3.11/dist-packages (from scikit-
image>=0.12->lime==0.2.0.1->-r requirements.txt (line 15)) (2.37.0)
Requirement already satisfied: tifffile>=2022.8.12 in
/usr/local/lib/python3.11/dist-packages (from scikit-
image>=0.12->lime==0.2.0.1->-r requirements.txt (line 15)) (2025.5.10)
Requirement already satisfied: lazy-loader>=0.4 in
/usr/local/lib/python3.11/dist-packages (from scikit-
image>=0.12->lime==0.2.0.1->-r requirements.txt (line 15)) (0.4)
Requirement already satisfied: click>=8.0.0 in /usr/local/lib/python3.11/dist-
packages (from typer<1.0,>=0.12->gradio==5.29.0->-r requirements.txt (line 3))
(8.2.0)
Requirement already satisfied: shellingham>=1.3.0 in
/usr/local/lib/python3.11/dist-packages (from
typer<1.0,>=0.12->gradio==5.29.0->-r requirements.txt (line 3)) (1.5.4)
Requirement already satisfied: rich>=10.11.0 in /usr/local/lib/python3.11/dist-
packages (from typer<1.0,>=0.12->gradio==5.29.0->-r requirements.txt (line 3))
```

```
(13.9.4)
Requirement already satisfied: attrs>=18.2.0 in /usr/local/lib/python3.11/dist-
packages (from dm-tree->tensorflow-datasets==4.9.8->-r requirements.txt (line
8)) (25.3.0)
Requirement already satisfied: contourpy>=1.0.1 in
/usr/local/lib/python3.11/dist-packages (from matplotlib->lime==0.2.0.1->-r
requirements.txt (line 15)) (1.3.2)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.11/dist-
packages (from matplotlib->lime==0.2.0.1->-r requirements.txt (line 15))
(0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in
/usr/local/lib/python3.11/dist-packages (from matplotlib->lime==0.2.0.1->-r
requirements.txt (line 15)) (4.58.0)
Requirement already satisfied: kiwisolver>=1.3.1 in
/usr/local/lib/python3.11/dist-packages (from matplotlib->lime==0.2.0.1->-r
requirements.txt (line 15)) (1.4.8)
Requirement already satisfied: pyparsing>=2.3.1 in
/usr/local/lib/python3.11/dist-packages (from matplotlib->lime==0.2.0.1->-r
requirements.txt (line 15)) (3.2.3)
Requirement already satisfied: six in /usr/local/lib/python3.11/dist-packages
(from promise->tensorflow-datasets==4.9.8->-r requirements.txt (line 8))
(1.17.0)
Requirement already satisfied: docstring-parser<1.0,>=0.15 in
/usr/local/lib/python3.11/dist-packages (from simple_parsing->tensorflow-
datasets==4.9.8->-r requirements.txt (line 8)) (0.16)
Requirement already satisfied: googleapis-common-protos<2,>=1.56.4 in
/usr/local/lib/python3.11/dist-packages (from tensorflow-metadata->tensorflow-
datasets==4.9.8->-r requirements.txt (line 8)) (1.70.0)
Requirement already satisfied: aiohappyeyeballs>=2.3.0 in
/usr/local/lib/python3.11/dist-packages (from aiohttp!=4.0.0a0,!=4.0.0a1-
>fsspec[http]<=2025.3.0,>=2023.1.0->datasets==3.6.0->-r requirements.txt (line
2)) (2.6.1)
Requirement already satisfied: aiosignal>=1.1.2 in
/usr/local/lib/python3.11/dist-packages (from aiohttp!=4.0.0a0,!=4.0.0a1-
>fsspec[http]<=2025.3.0,>=2023.1.0->datasets==3.6.0->-r requirements.txt (line
2)) (1.3.2)
Requirement already satisfied: frozenlist>=1.1.1 in
/usr/local/lib/python3.11/dist-packages (from aiohttp!=4.0.0a0,!=4.0.0a1-
>fsspec[http]<=2025.3.0,>=2023.1.0->datasets==3.6.0->-r requirements.txt (line
2)) (1.6.0)
Requirement already satisfied: multidict<7.0,>=4.5 in
/usr/local/lib/python3.11/dist-packages (from aiohttp!=4.0.0a0,!=4.0.0a1-
>fsspec[http]<=2025.3.0,>=2023.1.0->datasets==3.6.0->-r requirements.txt (line
2)) (6.4.3)
Requirement already satisfied: propcache>=0.2.0 in
/usr/local/lib/python3.11/dist-packages (from aiohttp!=4.0.0a0,!=4.0.0a1-
>fsspec[http]<=2025.3.0,>=2023.1.0->datasets==3.6.0->-r requirements.txt (line
2)) (0.3.1)
```

```
Requirement already satisfied: yarl<2.0,>=1.17.0 in
/usr/local/lib/python3.11/dist-packages (from aiohttp!=4.0.0a0,!=4.0.0a1-
>fsspec[http]<=2025.3.0,>=2023.1.0->datasets==3.6.0->-r requirements.txt (line
2)) (1.20.0)
Requirement already satisfied: markdown-it-py>=2.2.0 in
/usr/local/lib/python3.11/dist-packages (from
rich>=10.11.0->typer<1.0,>=0.12->gradio==5.29.0->-r requirements.txt (line 3))
(3.0.0)
Requirement already satisfied: pygments<3.0.0,>=2.13.0 in
/usr/local/lib/python3.11/dist-packages (from
rich>=10.11.0->typer<1.0,>=0.12->gradio==5.29.0->-r requirements.txt (line 3))
(2.19.1)
Requirement already satisfied: mdurl~=0.1 in /usr/local/lib/python3.11/dist-
packages (from markdown-it-
py>=2.2.0->rich>=10.11.0->typer<1.0,>=0.12->gradio==5.29.0->-r requirements.txt
(line 3)) (0.1.2)
Downloading bitsandbytes-0.45.5-py3-none-manylinux_2_24_x86_64.whl (76.1 MB)
                         76.1/76.1 MB
29.0 MB/s eta 0:00:00
Downloading datasets-3.6.0-py3-none-any.whl (491 kB)
                         491.5/491.5 kB
37.1 MB/s eta 0:00:00
Downloading gradio-5.29.0-py3-none-any.whl (54.1 MB)
                         54.1/54.1 MB
42.3 MB/s eta 0:00:00
Downloading gradio_client-1.10.0-py3-none-any.whl (322 kB)
                         322.9/322.9 kB
27.9 MB/s eta 0:00:00
Downloading sentence_transformers-3.4.1-py3-none-any.whl (275 kB)
                         275.9/275.9 kB
26.7 MB/s eta 0:00:00
Downloading transformers-4.48.3-py3-none-any.whl (9.7 MB)
                         9.7/9.7 MB
139.0 MB/s eta 0:00:00
Downloading nvidia_cublas_cu12-12.4.5.8-py3-none-manylinux2014_x86_64.whl
(363.4 MB)
                         363.4/363.4 MB
2.4 MB/s eta 0:00:00
Downloading nvidia_cuda_cupti_cu12-12.4.127-py3-none-
manylinux2014_x86_64.whl (13.8 MB)
                         13.8/13.8 MB
130.8 MB/s eta 0:00:00
Downloading nvidia_cuda_nvrtc_cu12-12.4.127-py3-none-
manylinux2014_x86_64.whl (24.6 MB)
                         24.6/24.6 MB
98.2 MB/s eta 0:00:00
Downloading nvidia_cuda_runtime_cu12-12.4.127-py3-none-
manylinux2014_x86_64.whl (883 kB)
```

```
                         883.7/883.7 kB
64.8 MB/s eta 0:00:00
Downloading nvidia_cudnn_cu12-9.1.0.70-py3-none-manylinux2014_x86_64.whl
(664.8 MB)
                         664.8/664.8 MB
1.4 MB/s eta 0:00:00
Downloading nvidia_cufft_cu12-11.2.1.3-py3-none-manylinux2014_x86_64.whl
(211.5 MB)
                         211.5/211.5 MB
10.5 MB/s eta 0:00:00
Downloading nvidia_curand_cu12-10.3.5.147-py3-none-
manylinux2014_x86_64.whl (56.3 MB)
                         56.3/56.3 MB
40.4 MB/s eta 0:00:00
Downloading nvidia_cusolver_cu12-11.6.1.9-py3-none-
manylinux2014_x86_64.whl (127.9 MB)
                         127.9/127.9 MB
7.8 MB/s eta 0:00:00
Downloading nvidia_cusparse_cu12-12.3.1.170-py3-none-
manylinux2014_x86_64.whl (207.5 MB)
                         207.5/207.5 MB
10.6 MB/s eta 0:00:00
Downloading nvidia_nvjitlink_cu12-12.4.127-py3-none-
manylinux2014_x86_64.whl (21.1 MB)
                         21.1/21.1 MB
112.4 MB/s eta 0:00:00
Downloading aiofiles-24.1.0-py3-none-any.whl (15 kB)
Downloading fastapi-0.115.12-py3-none-any.whl (95 kB)
                         95.2/95.2 kB
10.5 MB/s eta 0:00:00
Downloading fsspec-2025.3.0-py3-none-any.whl (193 kB)
                         193.6/193.6 kB
21.0 MB/s eta 0:00:00
Downloading groovy-0.1.2-py3-none-any.whl (14 kB)
Downloading python_multipart-0.0.20-py3-none-any.whl (24 kB)
Downloading ruff-0.11.10-py3-none-manylinux_2_17_x86_64.manylinux2014_x86_64.whl
(11.6 MB)
                         11.6/11.6 MB
136.7 MB/s eta 0:00:00
Downloading safehttpx-0.1.6-py3-none-any.whl (8.7 kB)
Downloading semantic_version-2.10.0-py2.py3-none-any.whl (15 kB)
Downloading starlette-0.46.2-py3-none-any.whl (72 kB)
                         72.0/72.0 kB
8.2 MB/s eta 0:00:00
Downloading tomlkit-0.13.2-py3-none-any.whl (37 kB)
Downloading uvicorn-0.34.2-py3-none-any.whl (62 kB)
                         62.5/62.5 kB
6.5 MB/s eta 0:00:00
```

```
Downloading ffmpy-0.5.0-py3-none-any.whl (6.0 kB)
Downloading pydub-0.25.1-py2.py3-none-any.whl (32 kB)
Building wheels for collected packages: lime
  Building wheel for lime (setup.py) … done
  Created wheel for lime: filename=lime-0.2.0.1-py3-none-any.whl size=283834
sha256=8795b46c604f34024dfb7ad1d8a0d48878db19ff95fa897c5f22dddd66fa2e11
  Stored in directory: /root/.cache/pip/wheels/85/fa/a3/9c2d44c9f3cd77cf4e533b58
900b2bf4487f2a17e8ec212a3d
Successfully built lime
Installing collected packages: pydub, uvicorn, tomlkit, semantic-version, ruff,
python-multipart, nvidia-nvjitlink-cu12, nvidia-curand-cu12, nvidia-cufft-cu12,
nvidia-cuda-runtime-cu12, nvidia-cuda-nvrtc-cu12, nvidia-cuda-cupti-cu12,
nvidia-cublas-cu12, groovy, fsspec, ffmpy, aiofiles, starlette, nvidia-cusparse-
cu12, nvidia-cudnn-cu12, safehttpx, nvidia-cusolver-cu12, lime, gradio_client,
fastapi, transformers, gradio, datasets, sentence-transformers, bitsandbytes
  Attempting uninstall: nvidia-nvjitlink-cu12
    Found existing installation: nvidia-nvjitlink-cu12 12.5.82
    Uninstalling nvidia-nvjitlink-cu12-12.5.82:
      Successfully uninstalled nvidia-nvjitlink-cu12-12.5.82
  Attempting uninstall: nvidia-curand-cu12
    Found existing installation: nvidia-curand-cu12 10.3.6.82
    Uninstalling nvidia-curand-cu12-10.3.6.82:
      Successfully uninstalled nvidia-curand-cu12-10.3.6.82
  Attempting uninstall: nvidia-cufft-cu12
    Found existing installation: nvidia-cufft-cu12 11.2.3.61
    Uninstalling nvidia-cufft-cu12-11.2.3.61:
      Successfully uninstalled nvidia-cufft-cu12-11.2.3.61
  Attempting uninstall: nvidia-cuda-runtime-cu12
    Found existing installation: nvidia-cuda-runtime-cu12 12.5.82
    Uninstalling nvidia-cuda-runtime-cu12-12.5.82:
      Successfully uninstalled nvidia-cuda-runtime-cu12-12.5.82
  Attempting uninstall: nvidia-cuda-nvrtc-cu12
    Found existing installation: nvidia-cuda-nvrtc-cu12 12.5.82
    Uninstalling nvidia-cuda-nvrtc-cu12-12.5.82:
      Successfully uninstalled nvidia-cuda-nvrtc-cu12-12.5.82
  Attempting uninstall: nvidia-cuda-cupti-cu12
    Found existing installation: nvidia-cuda-cupti-cu12 12.5.82
    Uninstalling nvidia-cuda-cupti-cu12-12.5.82:
      Successfully uninstalled nvidia-cuda-cupti-cu12-12.5.82
  Attempting uninstall: nvidia-cublas-cu12
    Found existing installation: nvidia-cublas-cu12 12.5.3.2
    Uninstalling nvidia-cublas-cu12-12.5.3.2:
      Successfully uninstalled nvidia-cublas-cu12-12.5.3.2
  Attempting uninstall: fsspec
    Found existing installation: fsspec 2025.3.2
    Uninstalling fsspec-2025.3.2:
      Successfully uninstalled fsspec-2025.3.2
  Attempting uninstall: nvidia-cusparse-cu12
```

```
    Found existing installation: nvidia-cusparse-cu12 12.5.1.3
    Uninstalling nvidia-cusparse-cu12-12.5.1.3:
      Successfully uninstalled nvidia-cusparse-cu12-12.5.1.3
  Attempting uninstall: nvidia-cudnn-cu12
    Found existing installation: nvidia-cudnn-cu12 9.3.0.75
    Uninstalling nvidia-cudnn-cu12-9.3.0.75:
      Successfully uninstalled nvidia-cudnn-cu12-9.3.0.75
  Attempting uninstall: nvidia-cusolver-cu12
    Found existing installation: nvidia-cusolver-cu12 11.6.3.83
    Uninstalling nvidia-cusolver-cu12-11.6.3.83:
      Successfully uninstalled nvidia-cusolver-cu12-11.6.3.83
  Attempting uninstall: transformers
    Found existing installation: transformers 4.51.3
    Uninstalling transformers-4.51.3:
      Successfully uninstalled transformers-4.51.3
  Attempting uninstall: datasets
    Found existing installation: datasets 2.14.4
    Uninstalling datasets-2.14.4:
      Successfully uninstalled datasets-2.14.4
  Attempting uninstall: sentence-transformers
    Found existing installation: sentence-transformers 4.1.0
    Uninstalling sentence-transformers-4.1.0:
      Successfully uninstalled sentence-transformers-4.1.0
ERROR: pip's dependency resolver does not currently take into account all
the packages that are installed. This behaviour is the source of the following
dependency conflicts.
gcsfs 2025.3.2 requires fsspec==2025.3.2, but you have fsspec 2025.3.0 which is
incompatible.

Successfully installed aiofiles-24.1.0 bitsandbytes-0.45.5 datasets-3.6.0
fastapi-0.115.12 ffmpy-0.5.0 fsspec-2025.3.0 gradio-5.29.0 gradio_client-1.10.0
groovy-0.1.2 lime-0.2.0.1 nvidia-cublas-cu12-12.4.5.8 nvidia-cuda-cupti-
cu12-12.4.127 nvidia-cuda-nvrtc-cu12-12.4.127 nvidia-cuda-runtime-cu12-12.4.127
nvidia-cudnn-cu12-9.1.0.70 nvidia-cufft-cu12-11.2.1.3 nvidia-curand-
cu12-10.3.5.147 nvidia-cusolver-cu12-11.6.1.9 nvidia-cusparse-cu12-12.3.1.170
nvidia-nvjitlink-cu12-12.4.127 pydub-0.25.1 python-multipart-0.0.20 ruff-0.11.10
safehttpx-0.1.6 semantic-version-2.10.0 sentence-transformers-3.4.1
starlette-0.46.2 tomlkit-0.13.2 transformers-4.48.3 uvicorn-0.34.2
```

[4]:
```
os.environ["HUGGINGFACE_TOKEN"] = "<YOUR_TOKEN>"
```

[5]:
```
from huggingface_hub import login
import os

# Load token from environment variable
hf_token = os.environ.get("HUGGINGFACE_TOKEN")
```

```
# Login using token
login(hf_token)
```

[6]:
```
import torch
print("GPU available:", torch.cuda.is_available())
```

```
GPU available: True
```

[7]:
```
import sys
print(sys.version)
!python --version
```

```
3.11.12 (main, Apr  9 2025, 08:55:54) [GCC 11.4.0]
Python 3.11.12
```

[ ]:
```
# ==================== 1. Setup ==================== #
from datasets import load_dataset, Dataset
from transformers import AutoTokenizer,␣
 ↪AutoModelForCausalLM,DataCollatorForLanguageModeling, Trainer,␣
 ↪TrainingArguments
from sklearn.utils import resample
from sklearn.metrics import accuracy_score,␣
 ↪classification_report,confusion_matrix, ConfusionMatrixDisplay,f1_score
import seaborn as sns
import pandas as pd
import matplotlib.pyplot as plt
import torch
from tqdm import tqdm
import time
import psutil
from itertools import product

# Load dataset
dataset = load_dataset("Zakia/drugscom_reviews")
train_df = pd.DataFrame(dataset["train"])
test_df = pd.DataFrame(dataset["test"])

# Load tokenizer and model
model_id = "meta-llama/Llama-2-7b-hf"
tokenizer = AutoTokenizer.from_pretrained(model_id, use_auth_token=True)
tokenizer.pad_token = tokenizer.eos_token
model = AutoModelForCausalLM.from_pretrained(
    model_id,
    torch_dtype=torch.float16,
    device_map="auto"
)
```

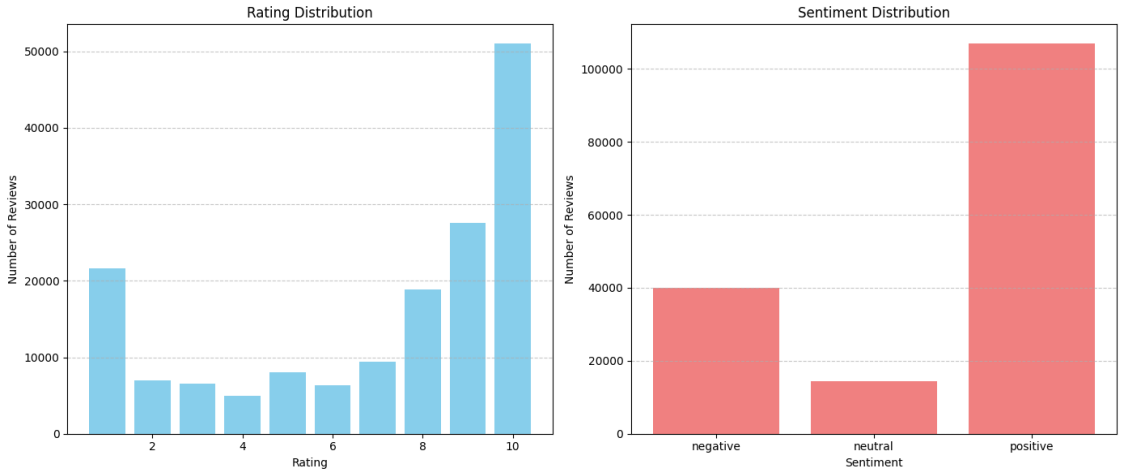
```
/usr/local/lib/python3.11/dist-packages/huggingface_hub/utils/_auth.py:94:
```

```
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://huggingface.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 datasets.
  warnings.warn(

README.md:   0%|          | 0.00/6.72k [00:00<?, ?B/s]

train.tsv:   0%|          | 0.00/84.3M [00:00<?, ?B/s]

test.tsv:   0%|          | 0.00/28.1M [00:00<?, ?B/s]

Generating train split:   0%|          | 0/161297 [00:00<?, ? examples/s]

Generating test split:   0%|          | 0/53766 [00:00<?, ? examples/s]

/usr/local/lib/python3.11/dist-
packages/transformers/models/auto/tokenization_auto.py:823: FutureWarning: The
`use_auth_token` argument is deprecated and will be removed in v5 of
Transformers. Please use `token` instead.
  warnings.warn(

tokenizer_config.json:   0%|          | 0.00/776 [00:00<?, ?B/s]

tokenizer.model:   0%|          | 0.00/500k [00:00<?, ?B/s]

tokenizer.json:   0%|          | 0.00/1.84M [00:00<?, ?B/s]

special_tokens_map.json:   0%|          | 0.00/414 [00:00<?, ?B/s]

config.json:   0%|          | 0.00/609 [00:00<?, ?B/s]

model.safetensors.index.json:   0%|          | 0.00/26.8k [00:00<?, ?B/s]

Downloading shards:   0%|          | 0/2 [00:00<?, ?it/s]

model-00001-of-00002.safetensors:   0%|          | 0.00/9.98G [00:00<?, ?B/s]

model-00002-of-00002.safetensors:   0%|          | 0.00/3.50G [00:00<?, ?B/s]

Loading checkpoint shards:   0%|          | 0/2 [00:00<?, ?it/s]

generation_config.json:   0%|          | 0.00/188 [00:00<?, ?B/s]
```

[ ]:
```python
# ==================== 2. Preprocessing ==================== #
def map_sentiment(rating):
    if rating <= 4:
        return "negative"
    elif rating <= 6:
        return "neutral"
    else:
        return "positive"
```

```
for df in [train_df, test_df]:
    df['sentiment'] = df['rating'].apply(map_sentiment)
    df['prompt'] = "Review: " + df['review'].astype(str) + "\nSentiment:"
    df['target'] = df['sentiment']
```

[ ]:
```
# ==================== 3. Visualization ==================== #
rating_counts = train_df['rating'].value_counts().sort_index()
sentiment_counts = train_df['sentiment'].value_counts().reindex(["negative",␣
 ↪"neutral", "positive"])

fig, axes = plt.subplots(1, 2, figsize=(14, 6))
axes[0].bar(rating_counts.index, rating_counts.values, color='skyblue')
axes[0].set_title("Rating Distribution")
axes[0].set_xlabel("Rating")
axes[0].set_ylabel("Number of Reviews")
axes[0].grid(axis='y', linestyle='--', alpha=0.7)

axes[1].bar(sentiment_counts.index, sentiment_counts.values, color='lightcoral')
axes[1].set_title("Sentiment Distribution")
axes[1].set_xlabel("Sentiment")
axes[1].set_ylabel("Number of Reviews")
axes[1].grid(axis='y', linestyle='--', alpha=0.7)
plt.tight_layout()
plt.show()
```



[ ]:
```
# ==================== 4. Downsampling ==================== #
def downsample(df):
    groups = [df[df['sentiment'] == s] for s in ['negative', 'neutral',␣
 ↪'positive']]
```

```
    min_size = min(len(g) for g in groups)
    return pd.concat([resample(g, replace=False, n_samples=min_size,␣
 ↪random_state=42) for g in groups])

balanced_df = downsample(train_df).sample(frac=1, random_state=42).
 ↪reset_index(drop=True)

# Sampling

sample_df = pd.concat([
    test_df[test_df['sentiment'] == 'positive'].sample(500, random_state=42),
    test_df[test_df['sentiment'] == 'neutral'].sample(500, random_state=42),
    test_df[test_df['sentiment'] == 'negative'].sample(500, random_state=42)
]).sample(frac=1, random_state=42).reset_index(drop=True)
```

Prompt Engineering

[ ]:
```
# ====================== 5.1 Zero-Shot Prompting ====================== #

def generate_sentiment_zeroshot(review_text, max_new_tokens=5):
    prompt = f"Review: {review_text}\nSentiment:"  # few-shot
    # prompt = f"Review: {review_text}\nSentiment:"               # Zero-shot
    input_ids = tokenizer(prompt, return_tensors="pt").to(model.device)
    with torch.no_grad():
        output = model.generate(
            **input_ids,
            max_new_tokens=max_new_tokens,
            temperature=0.7,
            top_p=0.9,
            do_sample=True,
            pad_token_id=tokenizer.eos_token_id
        )
    decoded = tokenizer.decode(output[0], skip_special_tokens=True)
    return decoded.split("Sentiment:")[-1].strip().split("\n")[0].lower()

def extract_sentiment(text):
    for label in ["positive", "negative", "neutral"]:
        if label in text.lower():
            return label
    return "unknown"
```

[ ]:
```
# ====================== 6.1  Inference and Evaluation - Zero-shot␣
 ↪====================== #
# Measure runtime
```

```
start_time = time.time()
latencies = []
predictions = []

for review in tqdm(sample_df['review']):
    t0 = time.time()
    raw_pred = generate_sentiment_zeroshot(review)
    clean_pred = extract_sentiment(raw_pred)
    t1 = time.time()
    latencies.append(t1 - t0)
    predictions.append(clean_pred)

end_time = time.time()

sample_df["predicted_sentiment"] = predictions

# Classification Metrics
true_labels = sample_df['target'].str.lower()
pred_labels = sample_df['predicted_sentiment']
pred_labels_cleaned = pred_labels.apply(lambda x: next((s for s in ["negative",␣
 ↪"neutral", "positive"] if s in x), "unknown"))
```

```
100%|      | 1500/1500 [07:55<00:00,  3.15it/s]
```

[ ]:
```
import numpy as np
import psutil
print("\n====================== Classification Metrics ======================")
print("Accuracy:", round(accuracy_score(true_labels, pred_labels_cleaned), 4))
report = classification_report(true_labels, pred_labels_cleaned,␣
 ↪output_dict=True)
macro = report['macro avg']
print("Precision (Macro):", round(macro['precision'], 4))
print("Recall (Macro):", round(macro['recall'], 4))
print("F1-Score (Macro):", round(macro['f1-score'], 4))
print("\nPer-Class Metrics:")
print(classification_report(true_labels, pred_labels_cleaned))

# Runtime Performance
total_time = end_time - start_time
avg_latency = np.mean(latencies)
percentile_95 = np.percentile(latencies, 95)
percentile_99 = np.percentile(latencies, 99)
throughput = len(sample_df) / total_time
memory_usage = psutil.Process().memory_info().rss / (1024 * 1024)  # in MB

print("\n====================== Runtime Performance ======================")
print(f"Total Samples Processed: {len(sample_df)}")
```

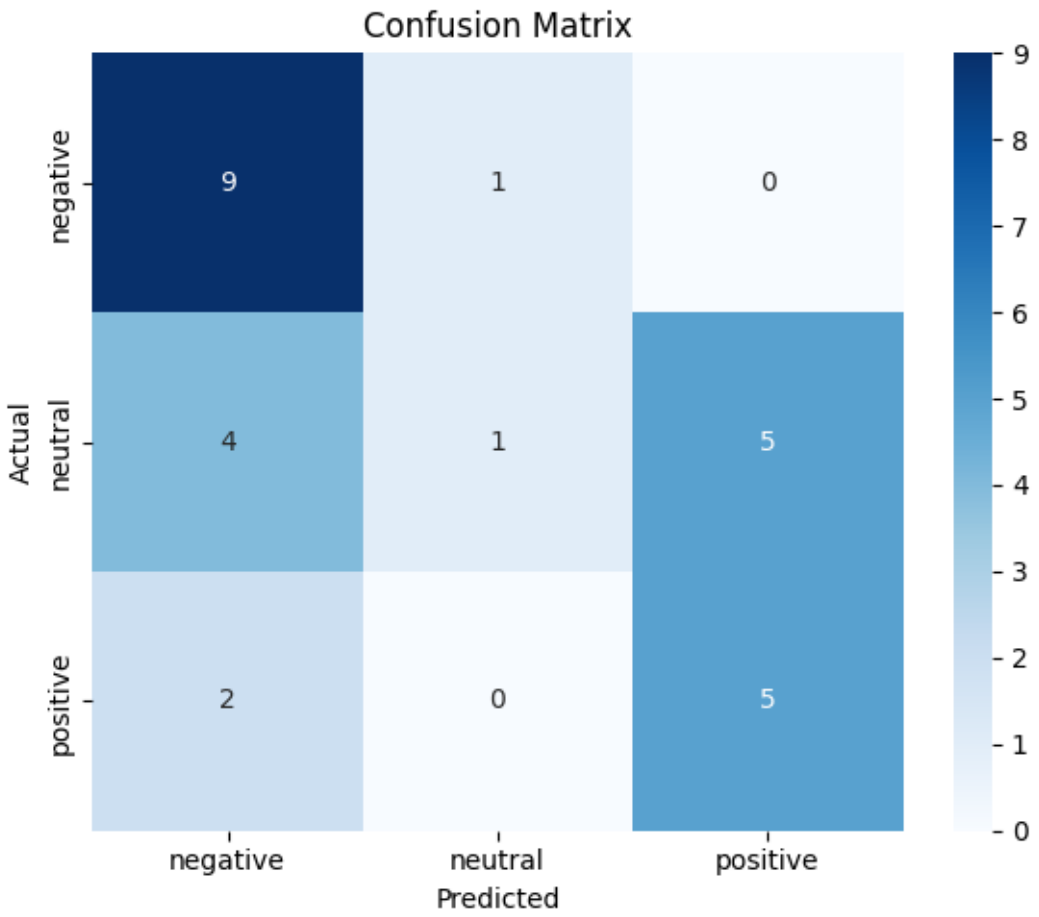
```python
print(f"Total Inference Time: {total_time:.2f} seconds")
print("...")
print(f"95th Percentile Latency: {percentile_95:.4f} sec")
print(f"99th Percentile Latency: {percentile_99:.4f} sec")
print(f"Throughput: {throughput:.2f} samples/sec")
print(f"Memory Usage (RSS): {memory_usage:.2f} MB")
```

```
====================== Classification Metrics ======================
Accuracy: 0.01
Precision (Macro): 0.4
Recall (Macro): 0.0075
F1-Score (Macro): 0.0146

Per-Class Metrics:
              precision    recall  f1-score   support

    negative       0.60      0.02      0.03       500
     neutral       0.50      0.00      0.00       500
    positive       0.50      0.01      0.02       500
     unknown       0.00      0.00      0.00         0

    accuracy                           0.01      1500
   macro avg       0.40      0.01      0.01      1500
weighted avg       0.53      0.01      0.02      1500


====================== Runtime Performance ======================
Total Samples Processed: 1500
Total Inference Time: 475.64 seconds
…
95th Percentile Latency: 0.3392 sec
99th Percentile Latency: 0.3755 sec
Throughput: 3.15 samples/sec
Memory Usage (RSS): 2768.17 MB
```

```
/usr/local/lib/python3.11/dist-packages/sklearn/metrics/_classification.py:1565:
UndefinedMetricWarning: Recall is ill-defined and being set to 0.0 in labels
with no true samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
/usr/local/lib/python3.11/dist-packages/sklearn/metrics/_classification.py:1565:
UndefinedMetricWarning: Recall is ill-defined and being set to 0.0 in labels
with no true samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
/usr/local/lib/python3.11/dist-packages/sklearn/metrics/_classification.py:1565:
UndefinedMetricWarning: Recall is ill-defined and being set to 0.0 in labels
with no true samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
```

```
/usr/local/lib/python3.11/dist-packages/sklearn/metrics/_classification.py:1565:
UndefinedMetricWarning: Recall is ill-defined and being set to 0.0 in labels
with no true samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
/usr/local/lib/python3.11/dist-packages/sklearn/metrics/_classification.py:1565:
UndefinedMetricWarning: Recall is ill-defined and being set to 0.0 in labels
with no true samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
/usr/local/lib/python3.11/dist-packages/sklearn/metrics/_classification.py:1565:
UndefinedMetricWarning: Recall is ill-defined and being set to 0.0 in labels
with no true samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
```

[ ]:
```
from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay
import seaborn as sns

# Compute confusion matrix
labels = ["negative", "neutral", "positive"]
cm = confusion_matrix(true_labels, pred_labels_cleaned, labels=labels)

# Plot using seaborn heatmap
plt.figure(figsize=(6, 5))
sns.heatmap(cm, annot=True, fmt="d", cmap="Blues", xticklabels=labels,␣
 ↪yticklabels=labels)
plt.title("Confusion Matrix")
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.tight_layout()
plt.show()
```



[ ]:
```
# ====================== 5.2 Few-Shot Prompting ====================== #
few_shot = """
Review: I love this medication. It really helped me sleep and improved my mood.
Sentiment: positive

Review: This drug made me feel worse. I had terrible side effects and had to␣
 ↪stop taking it.
Sentiment: negative

Review: It worked okay, but I still experienced some side effects. Not great,␣
 ↪not terrible.
Sentiment: neutral

"""

def generate_sentiment_fewshot(review_text, max_new_tokens=5):
    prompt = few_shot + f"Review: {review_text}\nSentiment:"  # few-shot
```

```
    # prompt = f"Review: {review_text}\nSentiment:"                # Zero-shot
    input_ids = tokenizer(prompt, return_tensors="pt").to(model.device)
    with torch.no_grad():
        output = model.generate(
            **input_ids,
            max_new_tokens=max_new_tokens,
            temperature=0.7,
            top_p=0.9,
            do_sample=True,
            pad_token_id=tokenizer.eos_token_id
        )
    decoded = tokenizer.decode(output[0], skip_special_tokens=True)
    return decoded.split("Sentiment:")[-1].strip().split("\n")[0].lower()
```

[ ]:
```
# ====================== 6.2 Inference and Evaluation - Few-shot␣
↪====================== #
# Measure runtime
start_time = time.time()
latencies = []
predictions = []

for review in tqdm(sample_df['review']):
    t0 = time.time()
    raw_pred = generate_sentiment_fewshot(review)
    clean_pred = extract_sentiment(raw_pred)
    t1 = time.time()
    latencies.append(t1 - t0)
    predictions.append(clean_pred)

end_time = time.time()

sample_df["predicted_sentiment"] = predictions

# Classification Metrics
true_labels = sample_df['target'].str.lower()
pred_labels = sample_df['predicted_sentiment']
pred_labels_cleaned = pred_labels.apply(lambda x: next((s for s in ["negative",␣
↪"neutral", "positive"] if s in x), "unknown"))
```

```
100%|      | 1500/1500 [08:43<00:00,  2.87it/s]
```

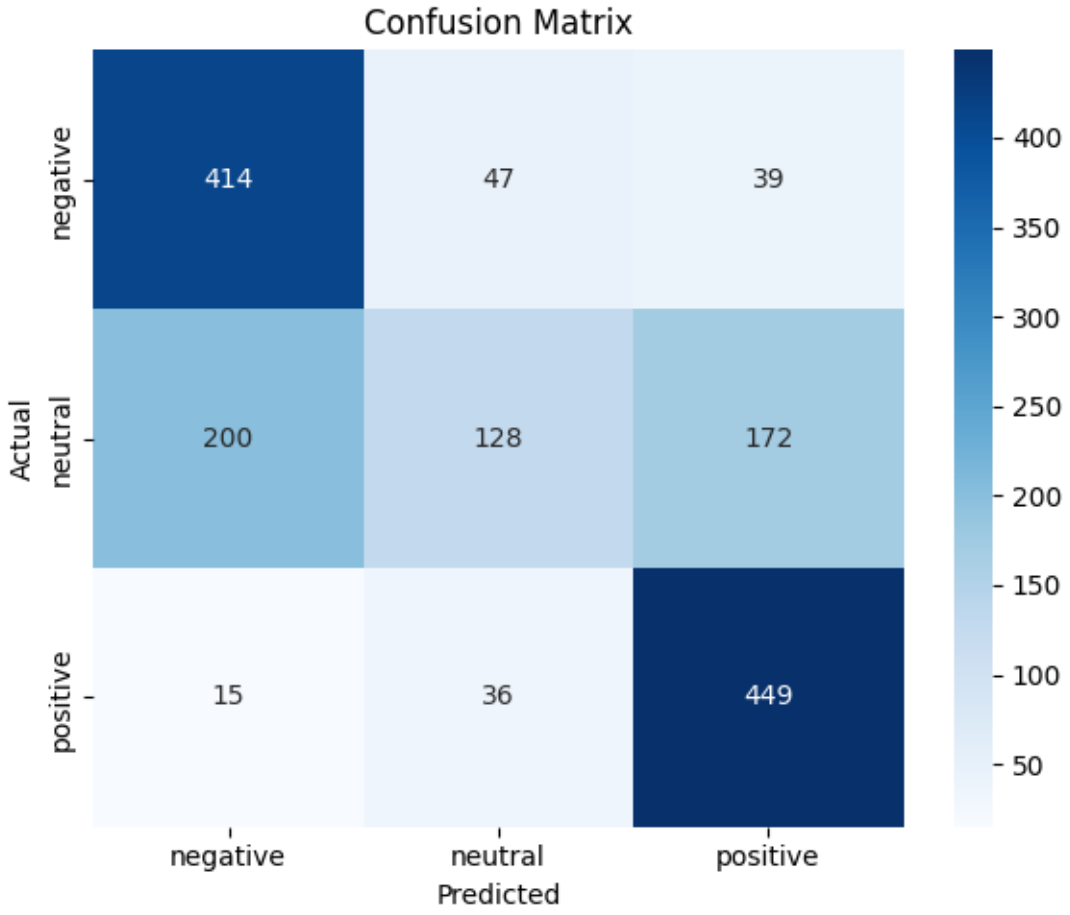
[ ]:
```
import numpy as np
import psutil
print("\n====================== Classification Metrics ======================")
print("Accuracy:", round(accuracy_score(true_labels, pred_labels_cleaned), 4))
report = classification_report(true_labels, pred_labels_cleaned,␣
↪output_dict=True)
```

```python
macro = report['macro avg']
print("Precision (Macro):", round(macro['precision'], 4))
print("Recall (Macro):", round(macro['recall'], 4))
print("F1-Score (Macro):", round(macro['f1-score'], 4))
print("\nPer-Class Metrics:")
print(classification_report(true_labels, pred_labels_cleaned))

# Runtime Performance
total_time = end_time - start_time
avg_latency = np.mean(latencies)
percentile_95 = np.percentile(latencies, 95)
percentile_99 = np.percentile(latencies, 99)
throughput = len(sample_df) / total_time
memory_usage = psutil.Process().memory_info().rss / (1024 * 1024)  # in MB

print("\n====================== Runtime Performance ======================")
print(f"Total Samples Processed: {len(sample_df)}")
print(f"Total Inference Time: {total_time:.2f} seconds")
print("...")
print(f"95th Percentile Latency: {percentile_95:.4f} sec")
print(f"99th Percentile Latency: {percentile_99:.4f} sec")
print(f"Throughput: {throughput:.2f} samples/sec")
print(f"Memory Usage (RSS): {memory_usage:.2f} MB")
```

```
====================== Classification Metrics ======================
Accuracy: 0.6607
Precision (Macro): 0.6484
Recall (Macro): 0.6607
F1-Score (Macro): 0.6225

Per-Class Metrics:
              precision    recall  f1-score   support

    negative       0.66      0.83      0.73       500
     neutral       0.61      0.26      0.36       500
    positive       0.68      0.90      0.77       500

    accuracy                           0.66      1500
   macro avg       0.65      0.66      0.62      1500
weighted avg       0.65      0.66      0.62      1500


====================== Runtime Performance ======================
Total Samples Processed: 1500
Total Inference Time: 523.03 seconds
…
95th Percentile Latency: 0.3883 sec
```

```
99th Percentile Latency: 0.3914 sec
Throughput: 2.87 samples/sec
Memory Usage (RSS): 2772.43 MB
```

[ ]:
```python
from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay
import seaborn as sns

# Compute confusion matrix
labels = ["negative", "neutral", "positive"]
cm = confusion_matrix(true_labels, pred_labels_cleaned, labels=labels)

# Plot using seaborn heatmap
plt.figure(figsize=(6, 5))
sns.heatmap(cm, annot=True, fmt="d", cmap="Blues", xticklabels=labels,␣
 ↪yticklabels=labels)
plt.title("Confusion Matrix")
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.tight_layout()
plt.show()
```



0.0.1 Hyperparameter tuning with Grid Search

[ ]:
```
def generate_with_params(review_text, temperature=0.7, top_p=0.9,␣
↪max_new_tokens=5):
    prompt = few_shot + f"Review: {review_text}\nSentiment:"
    input_ids = tokenizer(prompt, return_tensors="pt").to(model.device)

    with torch.no_grad():
        output = model.generate(
            **input_ids,
            max_new_tokens=max_new_tokens,
            temperature=temperature,
            top_p=top_p,
            do_sample=True,
            pad_token_id=tokenizer.eos_token_id
        )

    decoded = tokenizer.decode(output[0], skip_special_tokens=True)
    return decoded.split("Sentiment:")[-1].strip().split("\n")[0].lower()
```

[ ]:
```
temp_list = [0.5, 0.7, 0.9]
top_p_list = [0.8, 0.9]
max_tokens_list = [3, 5]

# Track results
grid_results = []

# Grid search loop
for temp in temp_list:
    for top_p in top_p_list:
        for max_tokens in max_tokens_list:
            predictions = []
            latencies = []
            start_time = time.time()

            for review in tqdm(sample_df['review'], desc=f"T={temp}, P={top_p},␣
↪M={max_tokens}"):
                t0 = time.time()
                raw_pred = generate_with_params(
                    review,
                    temperature=temp,
```

```
                    top_p=top_p,
                    max_new_tokens=max_tokens
                )
                clean_pred = extract_sentiment(raw_pred)
                t1 = time.time()
                latencies.append(t1 - t0)
                predictions.append(clean_pred)

            end_time = time.time()
            total_time = end_time - start_time
            avg_latency = sum(latencies) / len(latencies)

            # Evaluate
            true_labels = sample_df['target'].str.lower()
            pred_labels = pd.Series(predictions)
            pred_labels_cleaned = pred_labels.apply(lambda x: next((s for s in␣
 ↪["negative", "neutral", "positive"] if s in x), "unknown"))

            acc = accuracy_score(true_labels, pred_labels_cleaned)
            f1 = f1_score(true_labels, pred_labels_cleaned, average="macro",␣
 ↪zero_division=0)

            grid_results.append({
                "temperature": temp,
                "top_p": top_p,
                "max_new_tokens": max_tokens,
                "accuracy": acc,
                "f1_score": f1,
                "avg_latency": avg_latency
            })

# Convert to DataFrame
results_df = pd.DataFrame(grid_results)

# Select best params
best_row = results_df.sort_values("f1_score", ascending=False).iloc[0]
best_params = best_row.to_dict()
```

```
T=0.5, P=0.8, M=3: 100%|      | 1500/1500 [05:50<00:00,  4.28it/s]
T=0.5, P=0.8, M=5: 100%|      | 1500/1500 [08:53<00:00,  2.81it/s]
T=0.5, P=0.9, M=3: 100%|      | 1500/1500 [05:52<00:00,  4.25it/s]
T=0.5, P=0.9, M=5: 100%|      | 1500/1500 [08:52<00:00,  2.82it/s]
T=0.7, P=0.8, M=3: 100%|      | 1500/1500 [05:52<00:00,  4.25it/s]
T=0.7, P=0.8, M=5: 100%|      | 1500/1500 [08:52<00:00,  2.82it/s]
T=0.7, P=0.9, M=3: 100%|      | 1500/1500 [05:52<00:00,  4.25it/s]
T=0.7, P=0.9, M=5: 100%|      | 1500/1500 [08:52<00:00,  2.82it/s]
T=0.9, P=0.8, M=3: 100%|      | 1500/1500 [05:52<00:00,  4.25it/s]
```

```
T=0.9, P=0.8, M=5: 100%|      | 1500/1500 [08:52<00:00,  2.82it/s]
T=0.9, P=0.9, M=3: 100%|      | 1500/1500 [05:52<00:00,  4.25it/s]
T=0.9, P=0.9, M=5: 100%|      | 1500/1500 [08:52<00:00,  2.82it/s]
```

[ ]:
```
best_params
```

[ ]:
```
{'temperature': 0.9,
 'top_p': 0.8,
 'max_new_tokens': 3.0,
 'accuracy': 0.6693333333333333,
 'f1_score': 0.631705254058616,
 'avg_latency': 0.23449335098266602}
```

[ ]:
```
results_df
```

[ ]:

	temperature	top_p	max_new_tokens	accuracy	f1_score	avg_latency
0	0.5	0.8	3	0.664000	0.608067	0.232934
1	0.5	0.8	5	0.668000	0.460790	0.354739
2	0.5	0.9	3	0.653333	0.603067	0.234535
3	0.5	0.9	5	0.658667	0.610659	0.354301
4	0.7	0.8	3	0.666000	0.621059	0.234614
5	0.7	0.8	5	0.666000	0.621546	0.354396
6	0.7	0.9	3	0.650000	0.610019	0.234558
7	0.7	0.9	5	0.671333	0.475183	0.354309
8	0.9	0.8	3	0.669333	0.631705	0.234493
9	0.9	0.8	5	0.660667	0.467718	0.354284
10	0.9	0.9	3	0.658000	0.470046	0.234564
11	0.9	0.9	5	0.644000	0.607999	0.354380

[ ]:
```
# Final evaluation using best parameters
best_predictions = []
latencies = []
start_time = time.time()

for review in tqdm(sample_df['review'], desc="Final Inference (Best Params)"):
    t0 = time.time()
    raw_pred = generate_with_params(
        review,
        temperature=best_params['temperature'],
        top_p=best_params['top_p'],
        max_new_tokens=int(best_params['max_new_tokens'])
    )
    clean_pred = extract_sentiment(raw_pred)
    t1 = time.time()
    latencies.append(t1 - t0)
    best_predictions.append(clean_pred)

end_time = time.time()
```

```
sample_df["predicted_sentiment"] = best_predictions
```

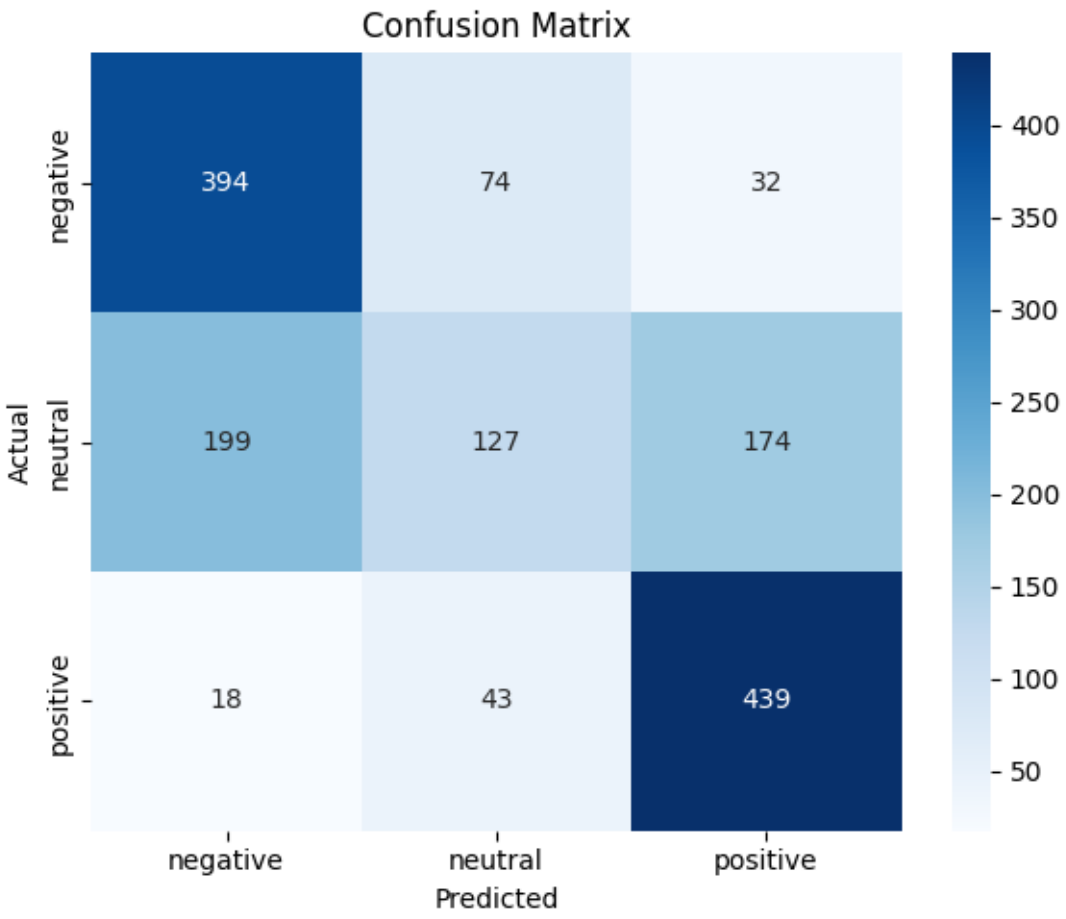
```
Final Inference (Best Params): 100%|      | 1500/1500 [05:52<00:00,
4.25it/s]
```

[ ]:
```
print("\n====================== Classification Metrics ======================")
print("Accuracy:", round(accuracy_score(true_labels, pred_labels_cleaned), 4))
report = classification_report(true_labels, pred_labels_cleaned,␣
 ↪output_dict=True)
macro = report['macro avg']
print("Precision (Macro):", round(macro['precision'], 4))
print("Recall (Macro):", round(macro['recall'], 4))
print("F1-Score (Macro):", round(macro['f1-score'], 4))
print("\nPer-Class Metrics:")
print(classification_report(true_labels, pred_labels_cleaned))

# Runtime Performance
total_time = end_time - start_time
avg_latency = np.mean(latencies)
percentile_95 = np.percentile(latencies, 95)
percentile_99 = np.percentile(latencies, 99)
throughput = len(sample_df) / total_time
memory_usage = psutil.Process().memory_info().rss / (1024 * 1024)  # in MB

print("\n====================== Runtime Performance ======================")
print(f"Total Samples Processed: {len(sample_df)}")
print(f"Total Inference Time: {total_time:.2f} seconds")
print("...")
print(f"95th Percentile Latency: {percentile_95:.4f} sec")
print(f"99th Percentile Latency: {percentile_99:.4f} sec")
print(f"Throughput: {throughput:.2f} samples/sec")
print(f"Memory Usage (RSS): {memory_usage:.2f} MB")
```

```
====================== Classification Metrics ======================
Accuracy: 0.64
Precision (Macro): 0.6153
Recall (Macro): 0.64
F1-Score (Macro): 0.6058

Per-Class Metrics:
              precision    recall  f1-score   support

    negative       0.64      0.79      0.71       500
     neutral       0.52      0.25      0.34       500
    positive       0.68      0.88      0.77       500

    accuracy                           0.64      1500
```

```
   macro avg       0.62      0.64      0.61      1500
weighted avg       0.62      0.64      0.61      1500


====================== Runtime Performance ======================
Total Samples Processed: 1500
Total Inference Time: 522.77 seconds
…
95th Percentile Latency: 0.3883 sec
99th Percentile Latency: 0.3918 sec
Throughput: 2.87 samples/sec
Memory Usage (RSS): 2805.12 MB
```

[ ]:
```python
from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay
import seaborn as sns

# Compute confusion matrix
labels = ["negative", "neutral", "positive"]
cm = confusion_matrix(true_labels, pred_labels_cleaned, labels=labels)

# Plot using seaborn heatmap
plt.figure(figsize=(6, 5))
sns.heatmap(cm, annot=True, fmt="d", cmap="Blues", xticklabels=labels,␣
 ↪yticklabels=labels)
plt.title("Confusion Matrix")
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.tight_layout()
plt.show()
```



0.0.2 Fine-tune with QLoRa

[ ]:
```
# ==================== Tokenization ==================== #
balanced_df_1 = pd.concat([
    balanced_df[balanced_df['target'] == 'positive'].sample(500,␣
 ↪random_state=42),
    balanced_df[balanced_df['target'] == 'neutral'].sample(500,␣
 ↪random_state=42),
    balanced_df[balanced_df['target'] == 'negative'].sample(500,␣
 ↪random_state=42)
]).sample(frac=1, random_state=42).reset_index(drop=True)

train_dataset = Dataset.from_pandas(balanced_df_1[["prompt", "target"]])
test_dataset = Dataset.from_pandas(sample_df[["prompt", "target"]])
```

```
max_length = 128

def tokenize(example):
    input_text = example['prompt'] + " " + example['target']
    return tokenizer(input_text, truncation=True, padding="max_length",␣
 ↪max_length=max_length)

train_tokenized = train_dataset.map(tokenize)
test_tokenized = test_dataset.map(tokenize)


data_collator = DataCollatorForLanguageModeling(tokenizer=tokenizer, mlm=False)
```

```
Map:   0%|          | 0/1500 [00:00<?, ? examples/s]
Map:   0%|          | 0/1500 [00:00<?, ? examples/s]
```

[ ]:
```
test_dataset
```

[ ]:
```
Dataset({
    features: ['prompt', 'target'],
    num_rows: 1500
})
```

[ ]:
```
from peft import LoraConfig, get_peft_model, TaskType,␣
 ↪prepare_model_for_kbit_training
from transformers import BitsAndBytesConfig
```

[ ]:
```
from pathlib import Path
import json
# ========== CONFIGURATION ==========

bnb_config = BitsAndBytesConfig(
    load_in_4bit=True,
    bnb_4bit_use_double_quant=True,
    bnb_4bit_quant_type="nf4",
    bnb_4bit_compute_dtype=torch.float16
)

lora_config = LoraConfig(
    r=8,
    lora_alpha=16,
    target_modules=["q_proj", "v_proj"],
    lora_dropout=0.05,
    bias="none",
    task_type="CAUSAL_LM"
```

```
)

# ========== GRID SETUP ==========
batch_sizes = [2]
learning_rates = [1e-5, 2e-5, 3e-5]
epochs = [2]
results = []


# ========== GRID SEARCH LOOP ==========
for bs in batch_sizes:
    for lr in learning_rates:
        for ep in epochs:
            output_dir = f"output_bs{bs}_lr{lr}_ep{ep}"
            checkpoint_path = None

            if os.path.exists(output_dir):
                checkpoints = sorted(Path(output_dir).glob("checkpoint-*"))
                if checkpoints:
                    checkpoint_path = str(checkpoints[-1])

            # Load and prepare model
            base_model = AutoModelForCausalLM.from_pretrained(
                model_id,
                quantization_config=bnb_config,
                device_map={"": 0},
                torch_dtype=torch.float16,
                use_auth_token=True
            )
            base_model = prepare_model_for_kbit_training(base_model)
            model = get_peft_model(base_model, lora_config)

            training_args = TrainingArguments(
                output_dir=output_dir,
                evaluation_strategy="epoch",
                logging_strategy="epoch",
                per_device_train_batch_size=bs,
                per_device_eval_batch_size=8,
                num_train_epochs=ep,
                learning_rate=lr,
                weight_decay=0.01,
                logging_dir=f"logs_bs{bs}_lr{lr}_ep{ep}",
                save_total_limit=1,
                fp16=True,
                report_to="none"
            )
```

```
            trainer = Trainer(
                model=model,
                args=training_args,
                train_dataset=train_tokenized,
                eval_dataset=test_tokenized,
                tokenizer=tokenizer,
                data_collator=data_collator
            )

            # Resume from checkpoint if exists
            if checkpoint_path:
                trainer.train(resume_from_checkpoint=checkpoint_path)
            else:
                trainer.train()

            trainer.save_model(output_dir)

            # ========== EVALUATION ==========
            start_time = time.time()
            latencies, predictions = [], []

            for review in tqdm(sample_df['review'], desc=f"Evaluating bs={bs},␣
↪lr={lr}"):
                prompt = f"Review: {review}\nSentiment:"
                input_ids = tokenizer(prompt, return_tensors="pt").to(model.
↪device)
                t0 = time.time()
                with torch.no_grad():
                    output = model.generate(
                        **input_ids,
                        max_new_tokens=3,
                        temperature=0.9,
                        top_p=0.8,
                        do_sample=True,
                        pad_token_id=tokenizer.eos_token_id
                    )
                t1 = time.time()
                decoded = tokenizer.decode(output[0], skip_special_tokens=True)
                pred = decoded.split("Sentiment:")[-1].strip().split("\n")[0].
↪lower()
                latencies.append(t1 - t0)
                predictions.append(pred)

            end_time = time.time()

            def extract_sentiment(text):
                for label in ["positive", "neutral", "negative"]:
```

```
                    if label in text.lower():
                        return label
                return "unknown"

            sample_df["predicted_sentiment"] = pd.Series(predictions).
 ↪apply(extract_sentiment)

            # Metrics
            true_labels = sample_df["target"].str.lower()
            pred_labels = sample_df["predicted_sentiment"]
            acc = round(accuracy_score(true_labels, pred_labels), 4)
            f1 = round(f1_score(true_labels, pred_labels, average="macro",␣
 ↪zero_division=0), 4)
            runtime = round(end_time - start_time, 2)
            clf_report = classification_report(true_labels, pred_labels,␣
 ↪output_dict=True, zero_division=0)
            conf_matrix = confusion_matrix(true_labels, pred_labels,␣
 ↪labels=["positive", "neutral", "negative"]).tolist()

            result = {
                "lr": lr,
                "batch_size": bs,
                "epochs": ep,
                "accuracy": acc,
                "f1": f1,
                "runtime_sec": runtime,
                "classification_report": clf_report,
                "confusion_matrix": conf_matrix
            }

            # Save JSON per run
            with open(f"{output_dir}/eval_results.json", "w") as f:
                json.dump(result, f, indent=2)

            results.append(result)

# ========== SUMMARY ==========
summary_df = pd.DataFrame(results)[["lr", "batch_size", "epochs", "accuracy",␣
 ↪"f1", "runtime_sec"]]
summary_df.sort_values("f1", ascending=False, inplace=True)
summary_df.reset_index(drop=True, inplace=True)
summary_df
```

```
/usr/local/lib/python3.11/dist-
packages/transformers/models/auto/auto_factory.py:471: FutureWarning: The
`use_auth_token` argument is deprecated and will be removed in v5 of
Transformers. Please use `token` instead.
```

```
  warnings.warn(

Loading checkpoint shards:   0%|          | 0/2 [00:00<?, ?it/s]

/usr/local/lib/python3.11/dist-packages/transformers/training_args.py:1575:
FutureWarning: `evaluation_strategy` is deprecated and will be removed in
version 4.46 of  Transformers. Use `eval_strategy` instead
  warnings.warn(
<ipython-input-15-7258f53d4162>:66: FutureWarning: `tokenizer` is deprecated and
will be removed in version 5.0.0 for `Trainer.__init__`. Use `processing_class`
instead.
  trainer = Trainer(
/usr/local/lib/python3.11/dist-packages/torch/_dynamo/eval_frame.py:745:
UserWarning: torch.utils.checkpoint: the use_reentrant parameter should be
passed explicitly. In version 2.5 we will raise an exception if use_reentrant is
not passed. use_reentrant=False is recommended, but if you need to preserve the
current default behavior, you can pass use_reentrant=True. Refer to docs for
more details on the differences between the two variants.
  return fn(*args, **kwargs)

<IPython.core.display.HTML object>

/usr/local/lib/python3.11/dist-packages/torch/_dynamo/eval_frame.py:745:
UserWarning: torch.utils.checkpoint: the use_reentrant parameter should be
passed explicitly. In version 2.5 we will raise an exception if use_reentrant is
not passed. use_reentrant=False is recommended, but if you need to preserve the
current default behavior, you can pass use_reentrant=True. Refer to docs for
more details on the differences between the two variants.
  return fn(*args, **kwargs)
/usr/local/lib/python3.11/dist-packages/torch/_dynamo/eval_frame.py:745:
UserWarning: torch.utils.checkpoint: the use_reentrant parameter should be
passed explicitly. In version 2.5 we will raise an exception if use_reentrant is
not passed. use_reentrant=False is recommended, but if you need to preserve the
current default behavior, you can pass use_reentrant=True. Refer to docs for
more details on the differences between the two variants.
  return fn(*args, **kwargs)
Evaluating bs=2, lr=1e-05: 100%|      | 1500/1500 [08:17<00:00,  3.02it/s]
/usr/local/lib/python3.11/dist-
packages/transformers/models/auto/auto_factory.py:471: FutureWarning: The
`use_auth_token` argument is deprecated and will be removed in v5 of
Transformers. Please use `token` instead.
  warnings.warn(

Loading checkpoint shards:   0%|          | 0/2 [00:00<?, ?it/s]

/usr/local/lib/python3.11/dist-packages/transformers/training_args.py:1575:
FutureWarning: `evaluation_strategy` is deprecated and will be removed in
version 4.46 of  Transformers. Use `eval_strategy` instead
  warnings.warn(
<ipython-input-15-7258f53d4162>:66: FutureWarning: `tokenizer` is deprecated and
will be removed in version 5.0.0 for `Trainer.__init__`. Use `processing_class`
```

```
instead.
  trainer = Trainer(
/usr/local/lib/python3.11/dist-packages/torch/_dynamo/eval_frame.py:745:
UserWarning: torch.utils.checkpoint: the use_reentrant parameter should be
passed explicitly. In version 2.5 we will raise an exception if use_reentrant is
not passed. use_reentrant=False is recommended, but if you need to preserve the
current default behavior, you can pass use_reentrant=True. Refer to docs for
more details on the differences between the two variants.
  return fn(*args, **kwargs)

<IPython.core.display.HTML object>

/usr/local/lib/python3.11/dist-packages/torch/_dynamo/eval_frame.py:745:
UserWarning: torch.utils.checkpoint: the use_reentrant parameter should be
passed explicitly. In version 2.5 we will raise an exception if use_reentrant is
not passed. use_reentrant=False is recommended, but if you need to preserve the
current default behavior, you can pass use_reentrant=True. Refer to docs for
more details on the differences between the two variants.
  return fn(*args, **kwargs)
/usr/local/lib/python3.11/dist-packages/torch/_dynamo/eval_frame.py:745:
UserWarning: torch.utils.checkpoint: the use_reentrant parameter should be
passed explicitly. In version 2.5 we will raise an exception if use_reentrant is
not passed. use_reentrant=False is recommended, but if you need to preserve the
current default behavior, you can pass use_reentrant=True. Refer to docs for
more details on the differences between the two variants.
  return fn(*args, **kwargs)
Evaluating bs=2, lr=2e-05: 100%|      | 1500/1500 [08:16<00:00,  3.02it/s]
/usr/local/lib/python3.11/dist-
packages/transformers/models/auto/auto_factory.py:471: FutureWarning: The
`use_auth_token` argument is deprecated and will be removed in v5 of
Transformers. Please use `token` instead.
  warnings.warn(

Loading checkpoint shards:   0%|          | 0/2 [00:00<?, ?it/s]

/usr/local/lib/python3.11/dist-packages/transformers/training_args.py:1575:
FutureWarning: `evaluation_strategy` is deprecated and will be removed in
version 4.46 of  Transformers. Use `eval_strategy` instead
  warnings.warn(
<ipython-input-15-7258f53d4162>:66: FutureWarning: `tokenizer` is deprecated and
will be removed in version 5.0.0 for `Trainer.__init__`. Use `processing_class`
instead.
  trainer = Trainer(
/usr/local/lib/python3.11/dist-packages/torch/_dynamo/eval_frame.py:745:
UserWarning: torch.utils.checkpoint: the use_reentrant parameter should be
passed explicitly. In version 2.5 we will raise an exception if use_reentrant is
not passed. use_reentrant=False is recommended, but if you need to preserve the
current default behavior, you can pass use_reentrant=True. Refer to docs for
more details on the differences between the two variants.
  return fn(*args, **kwargs)
```

```
<IPython.core.display.HTML object>

/usr/local/lib/python3.11/dist-packages/torch/_dynamo/eval_frame.py:745:
UserWarning: torch.utils.checkpoint: the use_reentrant parameter should be
passed explicitly. In version 2.5 we will raise an exception if use_reentrant is
not passed. use_reentrant=False is recommended, but if you need to preserve the
current default behavior, you can pass use_reentrant=True. Refer to docs for
more details on the differences between the two variants.
  return fn(*args, **kwargs)
/usr/local/lib/python3.11/dist-packages/torch/_dynamo/eval_frame.py:745:
UserWarning: torch.utils.checkpoint: the use_reentrant parameter should be
passed explicitly. In version 2.5 we will raise an exception if use_reentrant is
not passed. use_reentrant=False is recommended, but if you need to preserve the
current default behavior, you can pass use_reentrant=True. Refer to docs for
more details on the differences between the two variants.
  return fn(*args, **kwargs)
Evaluating bs=2, lr=3e-05: 100%|      | 1500/1500 [08:19<00:00,  3.00it/s]
```

[ ]:

	lr	batch_size	epochs	accuracy	f1	runtime_sec
0	0.00003	2	2	0.6607	0.6580	499.51
1	0.00002	2	2	0.6553	0.6489	496.59
2	0.00001	2	2	0.6240	0.4600	497.01

[ ]:
```python
# 1. Select best config from previous results (based on F1)
best_run = max(results, key=lambda x: x["f1"])
best_dir =␣
 ↪f"output_bs{best_run['batch_size']}_lr{best_run['lr']}_ep{best_run['epochs']}"
print(f"Best Model: {best_dir} - F1: {best_run['f1']}")

# 2. Reload fine-tuned model
base_model = AutoModelForCausalLM.from_pretrained(
    model_id,
    quantization_config=bnb_config,
    device_map={"": 0},
    torch_dtype=torch.float16,
    use_auth_token=True
)
base_model = prepare_model_for_kbit_training(base_model)
model = get_peft_model(base_model, lora_config)
model.load_adapter(best_dir)
```

[ ]:
```python
# 3. Evaluate on sample test_df
latencies = []
predictions = []
start_time = time.time()

for review in tqdm(sample_df['review'], desc="Evaluating best model on test␣
 ↪set"):
```

```
    prompt = f"Review: {review}\nSentiment:"
    input_ids = tokenizer(prompt, return_tensors="pt").to(model.device)
    t0 = time.time()
    with torch.no_grad():
        output = model.generate(
            **input_ids,
            max_new_tokens=3,
            temperature=0.7,
            top_p=0.9,
            do_sample=True,
            pad_token_id=tokenizer.eos_token_id
        )
    t1 = time.time()
    decoded = tokenizer.decode(output[0], skip_special_tokens=True)
    pred = decoded.split("Sentiment:")[-1].strip().split("\n")[0].lower()
    latencies.append(t1 - t0)
    predictions.append(pred)

end_time = time.time()

sample_df['predicted_sentiment'] = pd.Series(predictions).
 ↪apply(extract_sentiment)

# Print detailed classification + runtime report
true_labels = sample_df["target"].str.lower()
pred_labels = sample_df["predicted_sentiment"]
```

```
Evaluating best model on test set: 100%|     | 1500/1500 [08:19<00:00,
3.00it/s]
```

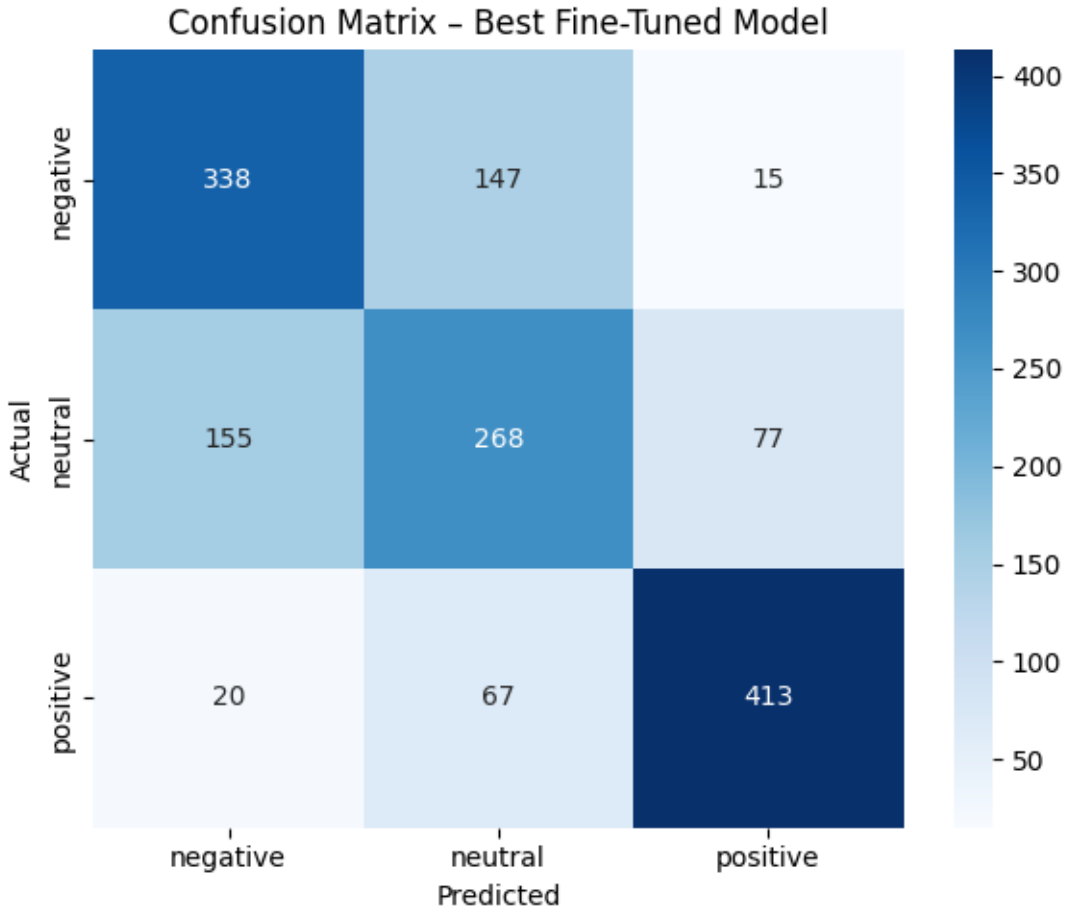
[ ]:
```
import numpy as np
# ==================== Classification Metrics ====================
print("\n" + "="*22, "Classification Metrics", "="*22)
acc = round(accuracy_score(true_labels, pred_labels), 4)
report = classification_report(true_labels, pred_labels, output_dict=True,␣
 ↪zero_division=0)
macro = report["macro avg"]
print("Accuracy:", acc)
print("Precision (Macro):", round(macro["precision"], 4))
print("Recall (Macro):", round(macro["recall"], 4))
print("F1-Score (Macro):", round(macro["f1-score"], 4))

print("\nPer-Class Metrics:")
print(classification_report(true_labels, pred_labels, zero_division=0))

# ==================== Runtime Performance ====================
print("\n" + "="*22, "Runtime Performance", "="*22)
```

```
total_time = end_time - start_time
avg_latency = np.mean(latencies)
percentile_95 = np.percentile(latencies, 95)
percentile_99 = np.percentile(latencies, 99)
throughput = len(sample_df) / total_time
memory_usage = psutil.Process().memory_info().rss / (1024 * 1024)

print(f"Total Samples Processed: {len(sample_df)}")
print(f"Total Inference Time: {total_time:.2f} seconds")
print(f"Average Latency: {avg_latency:.4f} sec")
print(f"95th Percentile Latency: {percentile_95:.4f} sec")
print(f"99th Percentile Latency: {percentile_99:.4f} sec")
print(f"Throughput: {throughput:.2f} samples/sec")
print(f"Memory Usage (RSS): {memory_usage:.2f} MB")
```

```
====================== Classification Metrics ======================
Accuracy: 0.6793
Precision (Macro): 0.6776
Recall (Macro): 0.6793
F1-Score (Macro): 0.6783

Per-Class Metrics:
              precision    recall  f1-score   support

    negative       0.66      0.68      0.67       500
     neutral       0.56      0.54      0.55       500
    positive       0.82      0.83      0.82       500

    accuracy                           0.68      1500
   macro avg       0.68      0.68      0.68      1500
weighted avg       0.68      0.68      0.68      1500


====================== Runtime Performance ======================
Total Samples Processed: 1500
Total Inference Time: 499.31 seconds
Average Latency: 0.3307 sec
95th Percentile Latency: 0.4021 sec
99th Percentile Latency: 0.4361 sec
Throughput: 3.00 samples/sec
Memory Usage (RSS): 3076.50 MB
```

[ ]:
```
# ==================== Confusion Matrix ====================
labels = ["negative", "neutral", "positive"]
cm = confusion_matrix(true_labels, pred_labels, labels=labels)
```

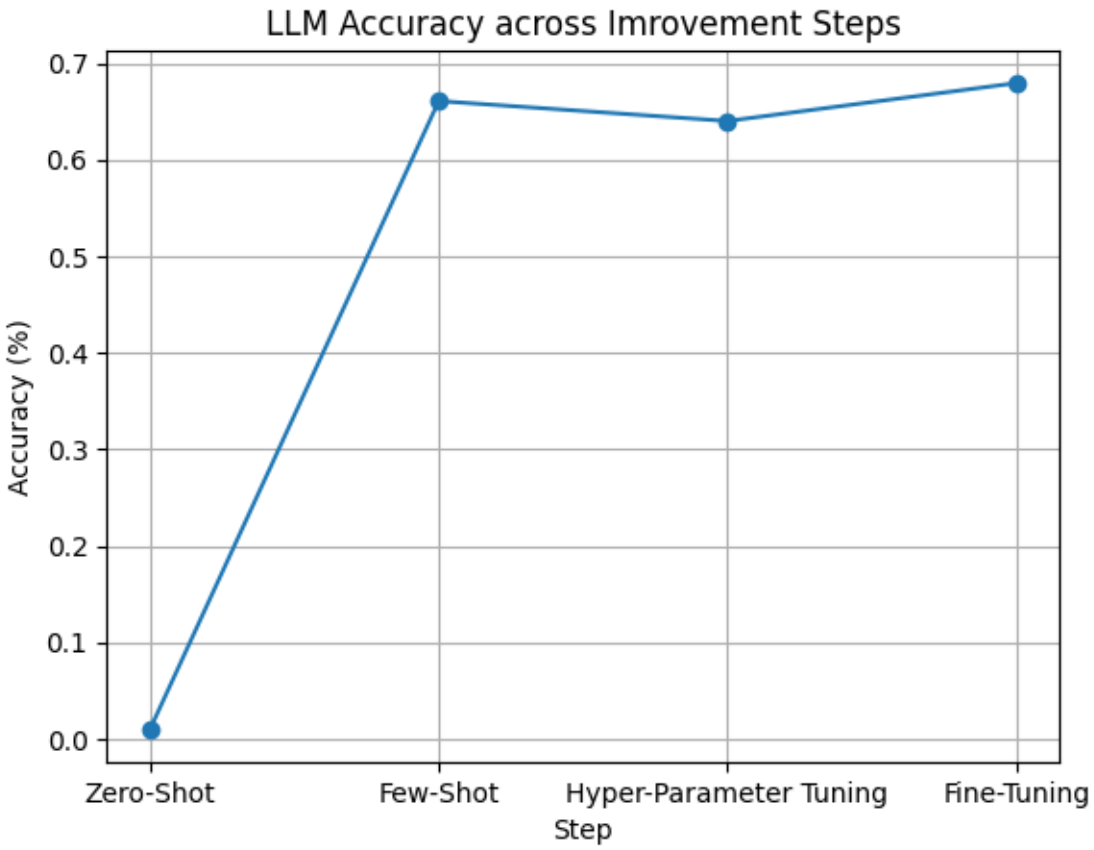
```
plt.figure(figsize=(6, 5))
sns.heatmap(cm, annot=True, fmt="d", cmap="Blues", xticklabels=labels,␣
 ↪yticklabels=labels)
plt.title("Confusion Matrix - Best Fine-Tuned Model")
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.tight_layout()
plt.show()
```



From the above results, accuracy and F1-score has improved with Fine-tuning.

[9]:
```
import matplotlib.pyplot as plt

steps = ['Zero-Shot', 'Few-Shot', 'Hyper-Parameter Tuning', 'Fine-Tuning']
scores = [0.01, 0.6607, 0.64, 0.6793]

plt.plot(steps,scores, marker = 'o')
plt.title("LLM Accuracy across Imrovement Steps")
```

```
plt.xlabel("Step")
plt.ylabel("Accuracy (%)")
plt.grid(True)
plt.show()
```



0.1 Gradio

[ ]:
```
import os
os.chdir('/content/drive/MyDrive/SIT764/')
```

[8]:
```
# Load the pretrained model with QLoRA
from transformers import AutoModelForCausalLM, AutoTokenizer,BitsAndBytesConfig
from peft import PeftModel
import torch

# Load quantized base model
base_model_id = "meta-llama/Llama-2-7b-hf"
adapter_path = "output_bs2_lr3e-05_ep2"
```

```
bnb_config = BitsAndBytesConfig(
    load_in_4bit=True,
    bnb_4bit_use_double_quant=True,
    bnb_4bit_quant_type="nf4",
    bnb_4bit_compute_dtype=torch.float16
)

tokenizer = AutoTokenizer.from_pretrained(adapter_path, use_auth_token=True)
base_model = AutoModelForCausalLM.from_pretrained(
    base_model_id,
    quantization_config=bnb_config,
    device_map="auto",
    torch_dtype=torch.float16,
    use_auth_token=True
)
model = PeftModel.from_pretrained(base_model, adapter_path)
model.eval()
```

```
/usr/local/lib/python3.11/dist-
packages/transformers/models/auto/tokenization_auto.py:823: FutureWarning: The
`use_auth_token` argument is deprecated and will be removed in v5 of
Transformers. Please use `token` instead.
  warnings.warn(
/usr/local/lib/python3.11/dist-
packages/transformers/models/auto/auto_factory.py:471: FutureWarning: The
`use_auth_token` argument is deprecated and will be removed in v5 of
Transformers. Please use `token` instead.
  warnings.warn(
/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://huggingface.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 datasets.
  warnings.warn(
config.json:   0%|          | 0.00/609 [00:00<?, ?B/s]
model.safetensors.index.json:   0%|          | 0.00/26.8k [00:00<?, ?B/s]
Downloading shards:   0%|          | 0/2 [00:00<?, ?it/s]
model-00001-of-00002.safetensors:   0%|          | 0.00/9.98G [00:00<?, ?B/s]
model-00002-of-00002.safetensors:   0%|          | 0.00/3.50G [00:00<?, ?B/s]
Loading checkpoint shards:   0%|          | 0/2 [00:00<?, ?it/s]
```

```
generation_config.json:   0%|          | 0.00/188 [00:00<?, ?B/s]
```

```
[8]: PeftModelForCausalLM(
       (base_model): LoraModel(
         (model): LlamaForCausalLM(
           (model): LlamaModel(
             (embed_tokens): Embedding(32000, 4096)
             (layers): ModuleList(
               (0-31): 32 x LlamaDecoderLayer(
                 (self_attn): LlamaAttention(
                   (q_proj): lora.Linear4bit(
                     (base_layer): Linear4bit(in_features=4096, out_features=4096,
     bias=False)
                     (lora_dropout): ModuleDict(
                       (default): Dropout(p=0.05, inplace=False)
                     )
                     (lora_A): ModuleDict(
                       (default): Linear(in_features=4096, out_features=8,
     bias=False)
                     )
                     (lora_B): ModuleDict(
                       (default): Linear(in_features=8, out_features=4096,
     bias=False)
                     )
                     (lora_embedding_A): ParameterDict()
                     (lora_embedding_B): ParameterDict()
                     (lora_magnitude_vector): ModuleDict()
                   )
                   (k_proj): Linear4bit(in_features=4096, out_features=4096,
     bias=False)
                   (v_proj): lora.Linear4bit(
                     (base_layer): Linear4bit(in_features=4096, out_features=4096,
     bias=False)
                     (lora_dropout): ModuleDict(
                       (default): Dropout(p=0.05, inplace=False)
                     )
                     (lora_A): ModuleDict(
                       (default): Linear(in_features=4096, out_features=8,
     bias=False)
                     )
                     (lora_B): ModuleDict(
                       (default): Linear(in_features=8, out_features=4096,
     bias=False)
                     )
                     (lora_embedding_A): ParameterDict()
                     (lora_embedding_B): ParameterDict()
                     (lora_magnitude_vector): ModuleDict()
```

```
              )
              (o_proj): Linear4bit(in_features=4096, out_features=4096,
bias=False)
            )
            (mlp): LlamaMLP(
              (gate_proj): Linear4bit(in_features=4096, out_features=11008,
bias=False)
              (up_proj): Linear4bit(in_features=4096, out_features=11008,
bias=False)
              (down_proj): Linear4bit(in_features=11008, out_features=4096,
bias=False)
              (act_fn): SiLU()
            )
            (input_layernorm): LlamaRMSNorm((4096,), eps=1e-05)
            (post_attention_layernorm): LlamaRMSNorm((4096,), eps=1e-05)
          )
        )
        (norm): LlamaRMSNorm((4096,), eps=1e-05)
        (rotary_emb): LlamaRotaryEmbedding()
      )
      (lm_head): Linear(in_features=4096, out_features=32000, bias=False)
    )
  )
)
```

[11]:
```python
## Define Functions:
def predict_sentiment(review):
    prompt = f"Review: {review}\nSentiment:"
    input_ids = tokenizer(prompt, return_tensors="pt").to(model.device)
    with torch.no_grad():
        output = model.generate(
            **input_ids,
            max_new_tokens=3,
            temperature=0.7,  # deterministic for explanation
            top_p=0.9,
            do_sample=False,
            pad_token_id=tokenizer.eos_token_id
        )
    decoded = tokenizer.decode(output[0], skip_special_tokens=True)
    sentiment = decoded.split("Sentiment:")[-1].strip().split("\n")[0].lower()
    return sentiment

def predict_sentiment_ui(review_text):
    sentiment = predict_sentiment(review_text)
    sentiment = sentiment.lower()
    if "positive" in sentiment:
        return "Positive"
```

```
    elif "neutral" in sentiment:
        return "Neutral"
    elif "negative" in sentiment:
        return "Negative"
    else:
        return "Unknown"

from lime.lime_text import LimeTextExplainer
import numpy as np

class_names = ["negative", "neutral", "positive"]
explainer = LimeTextExplainer(class_names=class_names)


def llm_predict_proba(texts):
    # Accepts a list of texts and returns one-hot probs
    predictions = []
    for text in texts:
        pred = predict_sentiment(text)
        if "negative" in pred:
            predictions.append([1.0, 0.0, 0.0])
        elif "neutral" in pred:
            predictions.append([0.0, 1.0, 0.0])
        elif "positive" in pred:
            predictions.append([0.0, 0.0, 1.0])
        else:
            predictions.append([0.33, 0.33, 0.34])  # fallback
    return np.array(predictions)

def explain_prediction_plot_with_label(review_text):
    # Get LIME explanation
    explanation = explainer.explain_instance(
        review_text,
        llm_predict_proba,
        num_features=6,
        num_samples=20
    )

    import matplotlib.pyplot as plt

    # Plot bar chart
    words, weights = zip(*explanation.as_list())
    colors = ["green" if w > 0 else "red" for w in weights]

    plt.figure(figsize=(8, 4))
    plt.barh(words, weights, color=colors)
    plt.xlabel("Contribution to Prediction")
```

```
    plt.title("LIME Explanation")
    plt.tight_layout()

    # Predict sentiment label using your main function
    prediction = predict_sentiment(review_text).lower()
    if "positive" in prediction:
        label = "Predicted Sentiment:  Positive"
    elif "neutral" in prediction:
        label = "Predicted Sentiment:  Neutral"
    elif "negative" in prediction:
        label = "Predicted Sentiment:  Negative"
    else:
        label = "Predicted Sentiment:  Unknown"

    return label, plt
```

[13]:
```
import gradio as gr

with gr.Blocks(title="LLM Drug Review Sentiment Analyzer") as demo:
    gr.Markdown("# Drug Review Sentiment Analyzer")
    gr.Markdown("Powered by fine-tuned LLaMA-2 using QLoRA. Enter a drug review␣
 ↪and receive its predicted sentiment.")

    with gr.Tab("  Predict Sentiment"):
        with gr.Row():
            review_input = gr.Textbox(lines=6, label="Enter your drug review␣
 ↪here")
            sentiment_output = gr.Radio(
                ["Positive", "Neutral", "Negative", "Unknown"],
                label="Predicted Sentiment",
                interactive=False
            )
        predict_btn = gr.Button("Analyze")

        predict_btn.click(
            fn=predict_sentiment_ui,
            inputs=review_input,
            outputs=sentiment_output
        )

    with gr.Tab("Submit Feedback"):
        gr.Markdown("If you think the model prediction was incorrect or␣
 ↪misleading, flag it below.")
        flagged_review = gr.Textbox(lines=4, label="Review text")
        feedback_reason = gr.Textbox(lines=2, label="Why are you flagging this?
 ↪")
        flag_button = gr.Button("Submit Feedback")
```

```
        def flag_callback(text, reason):
            # Save flagged text and reason to a log file or database
            with open("flagged_feedback.txt", "a") as f:
                f.write(f"Review: {text}\nReason: {reason}\n---\n")
            return "Thanks! Your feedback has been recorded."

        flag_output = gr.Textbox(label="", interactive=False)

        flag_button.click(fn=flag_callback, inputs=[flagged_review,␣
↪feedback_reason], outputs=flag_output)

    with gr.Tab("  Explain Prediction"):
        lime_input = gr.Textbox(lines=4, label="Enter a review to explain")
        label_output = gr.Textbox(label="Prediction")
        plot_output = gr.Plot(label="LIME Word Contributions")

        gr.Button("Generate Explanation").click(
            fn=explain_prediction_plot_with_label,
            inputs=lime_input,
            outputs=[label_output, plot_output]
        )


demo.launch(share=True, debug=True)
```

```
Colab notebook detected. This cell will run indefinitely so that you can see
errors and logs. To turn off, set debug=False in launch().
* Running on public URL: https://24c73194f7bd64bb87.gradio.live

This share link expires in 1 week. For free permanent hosting and GPU upgrades,
run `gradio deploy` from the terminal in the working directory to deploy to
Hugging Face Spaces (https://huggingface.co/spaces)
```

```
<IPython.core.display.HTML object>
```

```
/usr/local/lib/python3.11/dist-
packages/transformers/generation/configuration_utils.py:628: UserWarning:
`do_sample` is set to `False`. However, `temperature` is set to `0.7` -- this
flag is only used in sample-based generation modes. You should set
`do_sample=True` or unset `temperature`.
  warnings.warn(
/usr/local/lib/python3.11/dist-
packages/transformers/generation/configuration_utils.py:633: UserWarning:
`do_sample` is set to `False`. However, `top_p` is set to `0.9` -- this flag is
only used in sample-based generation modes. You should set `do_sample=True` or
unset `top_p`.
  warnings.warn(
```

```
Keyboard interruption in main thread… closing server.
Killing tunnel 127.0.0.1:7860 <> https://24c73194f7bd64bb87.gradio.live
```

[13]:

[ ]: