In [1]:
```
!pip install opencv-python numpy matplotlib seaborn pandas mlxtend tensorflo
!pip install datasets
!pip install transformers
!pip install tf-keras
```

```
Requirement already satisfied: opencv-python in c:\users\admin\anaconda3\lib
\site-packages (4.10.0.84)
Requirement already satisfied: numpy in c:\users\admin\anaconda3\lib\site-pa
ckages (1.26.4)
Requirement already satisfied: matplotlib in c:\users\admin\anaconda3\lib\si
te-packages (3.8.4)
Requirement already satisfied: seaborn in c:\users\admin\anaconda3\lib\site-
packages (0.13.2)
Requirement already satisfied: pandas in c:\users\admin\anaconda3\lib\site-p
ackages (2.2.2)
Requirement already satisfied: mlxtend in c:\users\admin\anaconda3\lib\site-
packages (0.23.3)
Requirement already satisfied: tensorflow in c:\users\admin\anaconda3\lib\si
te-packages (2.18.0)
Requirement already satisfied: scikit-learn in c:\users\admin\anaconda3\lib
\site-packages (1.4.2)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\admin\anaconda3
\lib\site-packages (from matplotlib) (1.2.0)
Requirement already satisfied: cycler>=0.10 in c:\users\admin\anaconda3\lib
\site-packages (from matplotlib) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\admin\anaconda3
\lib\site-packages (from matplotlib) (4.51.0)
Requirement already satisfied: kiwisolver>=1.3.1 in c:\users\admin\anaconda3
\lib\site-packages (from matplotlib) (1.4.4)
Requirement already satisfied: packaging>=20.0 in c:\users\admin\anaconda3\l
ib\site-packages (from matplotlib) (23.2)
Requirement already satisfied: pillow>=8 in c:\users\admin\anaconda3\lib\sit
e-packages (from matplotlib) (10.3.0)
Requirement already satisfied: pyparsing>=2.3.1 in c:\users\admin\anaconda3
\lib\site-packages (from matplotlib) (3.0.9)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\admin\anacon
da3\lib\site-packages (from matplotlib) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in c:\users\admin\anaconda3\lib
\site-packages (from pandas) (2024.1)
Requirement already satisfied: tzdata>=2022.7 in c:\users\admin\anaconda3\li
b\site-packages (from pandas) (2023.3)
Requirement already satisfied: scipy>=1.2.1 in c:\users\admin\anaconda3\lib
\site-packages (from mlxtend) (1.13.1)
Requirement already satisfied: joblib>=0.13.2 in c:\users\admin\anaconda3\li
b\site-packages (from mlxtend) (1.4.2)
Requirement already satisfied: tensorflow-intel==2.18.0 in c:\users\admin\an
aconda3\lib\site-packages (from tensorflow) (2.18.0)
Requirement already satisfied: absl-py>=1.0.0 in c:\users\admin\anaconda3\li
b\site-packages (from tensorflow-intel==2.18.0->tensorflow) (2.1.0)
Requirement already satisfied: astunparse>=1.6.0 in c:\users\admin\anaconda3
\lib\site-packages (from tensorflow-intel==2.18.0->tensorflow) (1.6.3)
Requirement already satisfied: flatbuffers>=24.3.25 in c:\users\admin\anacon
da3\lib\site-packages (from tensorflow-intel==2.18.0->tensorflow) (24.3.25)
Requirement already satisfied: gast!=0.5.0,!=0.5.1,!=0.5.2,>=0.2.1 in c:\use
rs\admin\anaconda3\lib\site-packages (from tensorflow-intel==2.18.0->tensorf
low) (0.6.0)
Requirement already satisfied: google-pasta>=0.1.1 in c:\users\admin\anacond
a3\lib\site-packages (from tensorflow-intel==2.18.0->tensorflow) (0.2.0)
Requirement already satisfied: libclang>=13.0.0 in c:\users\admin\anaconda3
\lib\site-packages (from tensorflow-intel==2.18.0->tensorflow) (18.1.1)
Requirement already satisfied: opt-einsum>=2.3.2 in c:\users\admin\anaconda3
```

```
\lib\site-packages (from tensorflow-intel==2.18.0->tensorflow) (3.4.0)
Requirement already satisfied: protobuf!=4.21.0,!=4.21.1,!=4.21.2,!=4.21.3,!
=4.21.4,!=4.21.5,<6.0.0dev,>=3.20.3 in c:\users\admin\anaconda3\lib\site-pac
kages (from tensorflow-intel==2.18.0->tensorflow) (3.20.3)
Requirement already satisfied: requests<3,>=2.21.0 in c:\users\admin\anacond
a3\lib\site-packages (from tensorflow-intel==2.18.0->tensorflow) (2.32.2)
Requirement already satisfied: setuptools in c:\users\admin\anaconda3\lib\si
te-packages (from tensorflow-intel==2.18.0->tensorflow) (69.5.1)
Requirement already satisfied: six>=1.12.0 in c:\users\admin\anaconda3\lib\s
ite-packages (from tensorflow-intel==2.18.0->tensorflow) (1.16.0)
Requirement already satisfied: termcolor>=1.1.0 in c:\users\admin\anaconda3
\lib\site-packages (from tensorflow-intel==2.18.0->tensorflow) (2.5.0)
Requirement already satisfied: typing-extensions>=3.6.6 in c:\users\admin\an
aconda3\lib\site-packages (from tensorflow-intel==2.18.0->tensorflow) (4.11.
0)
Requirement already satisfied: wrapt>=1.11.0 in c:\users\admin\anaconda3\lib
\site-packages (from tensorflow-intel==2.18.0->tensorflow) (1.14.1)
Requirement already satisfied: grpcio<2.0,>=1.24.3 in c:\users\admin\anacond
a3\lib\site-packages (from tensorflow-intel==2.18.0->tensorflow) (1.68.0)
Requirement already satisfied: tensorboard<2.19,>=2.18 in c:\users\admin\ana
conda3\lib\site-packages (from tensorflow-intel==2.18.0->tensorflow) (2.18.
0)
Requirement already satisfied: keras>=3.5.0 in c:\users\admin\anaconda3\lib
\site-packages (from tensorflow-intel==2.18.0->tensorflow) (3.6.0)
Requirement already satisfied: h5py>=3.11.0 in c:\users\admin\anaconda3\lib
\site-packages (from tensorflow-intel==2.18.0->tensorflow) (3.11.0)
Requirement already satisfied: ml-dtypes<0.5.0,>=0.4.0 in c:\users\admin\ana
conda3\lib\site-packages (from tensorflow-intel==2.18.0->tensorflow) (0.4.1)
Requirement already satisfied: threadpoolctl>=2.0.0 in c:\users\admin\anacon
da3\lib\site-packages (from scikit-learn) (2.2.0)
Requirement already satisfied: wheel<1.0,>=0.23.0 in c:\users\admin\anaconda
3\lib\site-packages (from astunparse>=1.6.0->tensorflow-intel==2.18.0->tenso
rflow) (0.43.0)
Requirement already satisfied: rich in c:\users\admin\anaconda3\lib\site-pac
kages (from keras>=3.5.0->tensorflow-intel==2.18.0->tensorflow) (13.3.5)
Requirement already satisfied: namex in c:\users\admin\anaconda3\lib\site-pa
ckages (from keras>=3.5.0->tensorflow-intel==2.18.0->tensorflow) (0.0.8)
Requirement already satisfied: optree in c:\users\admin\anaconda3\lib\site-p
ackages (from keras>=3.5.0->tensorflow-intel==2.18.0->tensorflow) (0.13.1)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\admin\an
aconda3\lib\site-packages (from requests<3,>=2.21.0->tensorflow-intel==2.18.
0->tensorflow) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in c:\users\admin\anaconda3\lib
\site-packages (from requests<3,>=2.21.0->tensorflow-intel==2.18.0->tensorfl
ow) (3.7)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\users\admin\anaconda
3\lib\site-packages (from requests<3,>=2.21.0->tensorflow-intel==2.18.0->ten
sorflow) (2.2.2)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\admin\anaconda
3\lib\site-packages (from requests<3,>=2.21.0->tensorflow-intel==2.18.0->ten
sorflow) (2024.8.30)
Requirement already satisfied: markdown>=2.6.8 in c:\users\admin\anaconda3\l
ib\site-packages (from tensorboard<2.19,>=2.18->tensorflow-intel==2.18.0->te
nsorflow) (3.4.1)
Requirement already satisfied: tensorboard-data-server<0.8.0,>=0.7.0 in c:\u
sers\admin\anaconda3\lib\site-packages (from tensorboard<2.19,>=2.18->tensor
```

```
flow-intel==2.18.0->tensorflow) (0.7.2)
Requirement already satisfied: werkzeug>=1.0.1 in c:\users\admin\anaconda3\l
ib\site-packages (from tensorboard<2.19,>=2.18->tensorflow-intel==2.18.0->te
nsorflow) (3.0.3)
Requirement already satisfied: MarkupSafe>=2.1.1 in c:\users\admin\anaconda3
\lib\site-packages (from werkzeug>=1.0.1->tensorboard<2.19,>=2.18->tensorflo
w-intel==2.18.0->tensorflow) (2.1.3)
Requirement already satisfied: markdown-it-py<3.0.0,>=2.2.0 in c:\users\admi
n\anaconda3\lib\site-packages (from rich->keras>=3.5.0->tensorflow-intel==2.
18.0->tensorflow) (2.2.0)
Requirement already satisfied: pygments<3.0.0,>=2.13.0 in c:\users\admin\ana
conda3\lib\site-packages (from rich->keras>=3.5.0->tensorflow-intel==2.18.0-
>tensorflow) (2.15.1)
Requirement already satisfied: mdurl~=0.1 in c:\users\admin\anaconda3\lib\si
te-packages (from markdown-it-py<3.0.0,>=2.2.0->rich->keras>=3.5.0->tensorfl
ow-intel==2.18.0->tensorflow) (0.1.0)
Requirement already satisfied: datasets in c:\users\admin\anaconda3\lib\site
-packages (3.1.0)
Requirement already satisfied: filelock in c:\users\admin\anaconda3\lib\site
-packages (from datasets) (3.13.1)
Requirement already satisfied: numpy>=1.17 in c:\users\admin\anaconda3\lib\s
ite-packages (from datasets) (1.26.4)
Requirement already satisfied: pyarrow>=15.0.0 in c:\users\admin\anaconda3\l
ib\site-packages (from datasets) (18.0.0)
Requirement already satisfied: dill<0.3.9,>=0.3.0 in c:\users\admin\anaconda
3\lib\site-packages (from datasets) (0.3.8)
Requirement already satisfied: pandas in c:\users\admin\anaconda3\lib\site-p
ackages (from datasets) (2.2.2)
Requirement already satisfied: requests>=2.32.2 in c:\users\admin\anaconda3
\lib\site-packages (from datasets) (2.32.2)
Requirement already satisfied: tqdm>=4.66.3 in c:\users\admin\anaconda3\lib
\site-packages (from datasets) (4.66.4)
Requirement already satisfied: xxhash in c:\users\admin\anaconda3\lib\site-p
ackages (from datasets) (3.5.0)
Requirement already satisfied: multiprocess<0.70.17 in c:\users\admin\anacon
da3\lib\site-packages (from datasets) (0.70.16)
Requirement already satisfied: fsspec<=2024.9.0,>=2023.1.0 in c:\users\admin
\anaconda3\lib\site-packages (from fsspec[http]<=2024.9.0,>=2023.1.0->datase
ts) (2024.3.1)
Requirement already satisfied: aiohttp in c:\users\admin\anaconda3\lib\site-
packages (from datasets) (3.9.5)
Requirement already satisfied: huggingface-hub>=0.23.0 in c:\users\admin\ana
conda3\lib\site-packages (from datasets) (0.26.2)
Requirement already satisfied: packaging in c:\users\admin\anaconda3\lib\sit
e-packages (from datasets) (23.2)
Requirement already satisfied: pyyaml>=5.1 in c:\users\admin\anaconda3\lib\s
ite-packages (from datasets) (6.0.1)
Requirement already satisfied: aiosignal>=1.1.2 in c:\users\admin\anaconda3
\lib\site-packages (from aiohttp->datasets) (1.2.0)
Requirement already satisfied: attrs>=17.3.0 in c:\users\admin\anaconda3\lib
\site-packages (from aiohttp->datasets) (23.1.0)
Requirement already satisfied: frozenlist>=1.1.1 in c:\users\admin\anaconda3
\lib\site-packages (from aiohttp->datasets) (1.4.0)
Requirement already satisfied: multidict<7.0,>=4.5 in c:\users\admin\anacond
a3\lib\site-packages (from aiohttp->datasets) (6.0.4)
Requirement already satisfied: yarl<2.0,>=1.0 in c:\users\admin\anaconda3\li
```

```
b\site-packages (from aiohttp->datasets) (1.9.3)
Requirement already satisfied: typing-extensions>=3.7.4.3 in c:\users\admin
\anaconda3\lib\site-packages (from huggingface-hub>=0.23.0->datasets) (4.11.
0)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\admin\an
aconda3\lib\site-packages (from requests>=2.32.2->datasets) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in c:\users\admin\anaconda3\lib
\site-packages (from requests>=2.32.2->datasets) (3.7)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\users\admin\anaconda
3\lib\site-packages (from requests>=2.32.2->datasets) (2.2.2)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\admin\anaconda
3\lib\site-packages (from requests>=2.32.2->datasets) (2024.8.30)
Requirement already satisfied: colorama in c:\users\admin\anaconda3\lib\site
-packages (from tqdm>=4.66.3->datasets) (0.4.6)
Requirement already satisfied: python-dateutil>=2.8.2 in c:\users\admin\anac
onda3\lib\site-packages (from pandas->datasets) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in c:\users\admin\anaconda3\lib
\site-packages (from pandas->datasets) (2024.1)
Requirement already satisfied: tzdata>=2022.7 in c:\users\admin\anaconda3\li
b\site-packages (from pandas->datasets) (2023.3)
Requirement already satisfied: six>=1.5 in c:\users\admin\anaconda3\lib\site
-packages (from python-dateutil>=2.8.2->pandas->datasets) (1.16.0)
Requirement already satisfied: transformers in c:\users\admin\anaconda3\lib
\site-packages (4.46.3)
Requirement already satisfied: filelock in c:\users\admin\anaconda3\lib\site
-packages (from transformers) (3.13.1)
Requirement already satisfied: huggingface-hub<1.0,>=0.23.2 in c:\users\admi
n\anaconda3\lib\site-packages (from transformers) (0.26.2)
Requirement already satisfied: numpy>=1.17 in c:\users\admin\anaconda3\lib\s
ite-packages (from transformers) (1.26.4)
Requirement already satisfied: packaging>=20.0 in c:\users\admin\anaconda3\l
ib\site-packages (from transformers) (23.2)
Requirement already satisfied: pyyaml>=5.1 in c:\users\admin\anaconda3\lib\s
ite-packages (from transformers) (6.0.1)
Requirement already satisfied: regex!=2019.12.17 in c:\users\admin\anaconda3
\lib\site-packages (from transformers) (2023.10.3)
Requirement already satisfied: requests in c:\users\admin\anaconda3\lib\site
-packages (from transformers) (2.32.2)
Requirement already satisfied: tokenizers<0.21,>=0.20 in c:\users\admin\anac
onda3\lib\site-packages (from transformers) (0.20.3)
Requirement already satisfied: safetensors>=0.4.1 in c:\users\admin\anaconda
3\lib\site-packages (from transformers) (0.4.5)
Requirement already satisfied: tqdm>=4.27 in c:\users\admin\anaconda3\lib\si
te-packages (from transformers) (4.66.4)
Requirement already satisfied: fsspec>=2023.5.0 in c:\users\admin\anaconda3
\lib\site-packages (from huggingface-hub<1.0,>=0.23.2->transformers) (2024.
3.1)
Requirement already satisfied: typing-extensions>=3.7.4.3 in c:\users\admin
\anaconda3\lib\site-packages (from huggingface-hub<1.0,>=0.23.2->transformer
s) (4.11.0)
Requirement already satisfied: colorama in c:\users\admin\anaconda3\lib\site
-packages (from tqdm>=4.27->transformers) (0.4.6)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\admin\an
aconda3\lib\site-packages (from requests->transformers) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in c:\users\admin\anaconda3\lib
\site-packages (from requests->transformers) (3.7)
```

```
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\users\admin\anaconda
3\lib\site-packages (from requests->transformers) (2.2.2)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\admin\anaconda
3\lib\site-packages (from requests->transformers) (2024.8.30)
Requirement already satisfied: tf-keras in c:\users\admin\anaconda3\lib\site
-packages (2.18.0)
Requirement already satisfied: tensorflow<2.19,>=2.18 in c:\users\admin\anac
onda3\lib\site-packages (from tf-keras) (2.18.0)
Requirement already satisfied: tensorflow-intel==2.18.0 in c:\users\admin\an
aconda3\lib\site-packages (from tensorflow<2.19,>=2.18->tf-keras) (2.18.0)
Requirement already satisfied: absl-py>=1.0.0 in c:\users\admin\anaconda3\li
b\site-packages (from tensorflow-intel==2.18.0->tensorflow<2.19,>=2.18->tf-k
eras) (2.1.0)
Requirement already satisfied: astunparse>=1.6.0 in c:\users\admin\anaconda3
\lib\site-packages (from tensorflow-intel==2.18.0->tensorflow<2.19,>=2.18->t
f-keras) (1.6.3)
Requirement already satisfied: flatbuffers>=24.3.25 in c:\users\admin\anacon
da3\lib\site-packages (from tensorflow-intel==2.18.0->tensorflow<2.19,>=2.18
->tf-keras) (24.3.25)
Requirement already satisfied: gast!=0.5.0,!=0.5.1,!=0.5.2,>=0.2.1 in c:\use
rs\admin\anaconda3\lib\site-packages (from tensorflow-intel==2.18.0->tensorf
low<2.19,>=2.18->tf-keras) (0.6.0)
Requirement already satisfied: google-pasta>=0.1.1 in c:\users\admin\anacond
a3\lib\site-packages (from tensorflow-intel==2.18.0->tensorflow<2.19,>=2.18-
>tf-keras) (0.2.0)
Requirement already satisfied: libclang>=13.0.0 in c:\users\admin\anaconda3
\lib\site-packages (from tensorflow-intel==2.18.0->tensorflow<2.19,>=2.18->t
f-keras) (18.1.1)
Requirement already satisfied: opt-einsum>=2.3.2 in c:\users\admin\anaconda3
\lib\site-packages (from tensorflow-intel==2.18.0->tensorflow<2.19,>=2.18->t
f-keras) (3.4.0)
Requirement already satisfied: packaging in c:\users\admin\anaconda3\lib\sit
e-packages (from tensorflow-intel==2.18.0->tensorflow<2.19,>=2.18->tf-keras)
(23.2)
Requirement already satisfied: protobuf!=4.21.0,!=4.21.1,!=4.21.2,!=4.21.3,!
=4.21.4,!=4.21.5,<6.0.0dev,>=3.20.3 in c:\users\admin\anaconda3\lib\site-pac
kages (from tensorflow-intel==2.18.0->tensorflow<2.19,>=2.18->tf-keras) (3.2
0.3)
Requirement already satisfied: requests<3,>=2.21.0 in c:\users\admin\anacond
a3\lib\site-packages (from tensorflow-intel==2.18.0->tensorflow<2.19,>=2.18-
>tf-keras) (2.32.2)
Requirement already satisfied: setuptools in c:\users\admin\anaconda3\lib\si
te-packages (from tensorflow-intel==2.18.0->tensorflow<2.19,>=2.18->tf-kera
s) (69.5.1)
Requirement already satisfied: six>=1.12.0 in c:\users\admin\anaconda3\lib\s
ite-packages (from tensorflow-intel==2.18.0->tensorflow<2.19,>=2.18->tf-kera
s) (1.16.0)
Requirement already satisfied: termcolor>=1.1.0 in c:\users\admin\anaconda3
\lib\site-packages (from tensorflow-intel==2.18.0->tensorflow<2.19,>=2.18->t
f-keras) (2.5.0)
Requirement already satisfied: typing-extensions>=3.6.6 in c:\users\admin\an
aconda3\lib\site-packages (from tensorflow-intel==2.18.0->tensorflow<2.19,>=
2.18->tf-keras) (4.11.0)
Requirement already satisfied: wrapt>=1.11.0 in c:\users\admin\anaconda3\lib
\site-packages (from tensorflow-intel==2.18.0->tensorflow<2.19,>=2.18->tf-ke
ras) (1.14.1)
```

```
Requirement already satisfied: grpcio<2.0,>=1.24.3 in c:\users\admin\anacond
a3\lib\site-packages (from tensorflow-intel==2.18.0->tensorflow<2.19,>=2.18-
>tf-keras) (1.68.0)
Requirement already satisfied: tensorboard<2.19,>=2.18 in c:\users\admin\ana
conda3\lib\site-packages (from tensorflow-intel==2.18.0->tensorflow<2.19,>=
2.18->tf-keras) (2.18.0)
Requirement already satisfied: keras>=3.5.0 in c:\users\admin\anaconda3\lib
\site-packages (from tensorflow-intel==2.18.0->tensorflow<2.19,>=2.18->tf-ke
ras) (3.6.0)
Requirement already satisfied: numpy<2.1.0,>=1.26.0 in c:\users\admin\anacon
da3\lib\site-packages (from tensorflow-intel==2.18.0->tensorflow<2.19,>=2.18
->tf-keras) (1.26.4)
Requirement already satisfied: h5py>=3.11.0 in c:\users\admin\anaconda3\lib
\site-packages (from tensorflow-intel==2.18.0->tensorflow<2.19,>=2.18->tf-ke
ras) (3.11.0)
Requirement already satisfied: ml-dtypes<0.5.0,>=0.4.0 in c:\users\admin\ana
conda3\lib\site-packages (from tensorflow-intel==2.18.0->tensorflow<2.19,>=
2.18->tf-keras) (0.4.1)
Requirement already satisfied: wheel<1.0,>=0.23.0 in c:\users\admin\anaconda
3\lib\site-packages (from astunparse>=1.6.0->tensorflow-intel==2.18.0->tenso
rflow<2.19,>=2.18->tf-keras) (0.43.0)
Requirement already satisfied: rich in c:\users\admin\anaconda3\lib\site-pac
kages (from keras>=3.5.0->tensorflow-intel==2.18.0->tensorflow<2.19,>=2.18->
tf-keras) (13.3.5)
Requirement already satisfied: namex in c:\users\admin\anaconda3\lib\site-pa
ckages (from keras>=3.5.0->tensorflow-intel==2.18.0->tensorflow<2.19,>=2.18-
>tf-keras) (0.0.8)
Requirement already satisfied: optree in c:\users\admin\anaconda3\lib\site-p
ackages (from keras>=3.5.0->tensorflow-intel==2.18.0->tensorflow<2.19,>=2.18
->tf-keras) (0.13.1)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\admin\an
aconda3\lib\site-packages (from requests<3,>=2.21.0->tensorflow-intel==2.18.
0->tensorflow<2.19,>=2.18->tf-keras) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in c:\users\admin\anaconda3\lib
\site-packages (from requests<3,>=2.21.0->tensorflow-intel==2.18.0->tensorfl
ow<2.19,>=2.18->tf-keras) (3.7)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\users\admin\anaconda
3\lib\site-packages (from requests<3,>=2.21.0->tensorflow-intel==2.18.0->ten
sorflow<2.19,>=2.18->tf-keras) (2.2.2)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\admin\anaconda
3\lib\site-packages (from requests<3,>=2.21.0->tensorflow-intel==2.18.0->ten
sorflow<2.19,>=2.18->tf-keras) (2024.8.30)
Requirement already satisfied: markdown>=2.6.8 in c:\users\admin\anaconda3\l
ib\site-packages (from tensorboard<2.19,>=2.18->tensorflow-intel==2.18.0->te
nsorflow<2.19,>=2.18->tf-keras) (3.4.1)
Requirement already satisfied: tensorboard-data-server<0.8.0,>=0.7.0 in c:\u
sers\admin\anaconda3\lib\site-packages (from tensorboard<2.19,>=2.18->tensor
flow-intel==2.18.0->tensorflow<2.19,>=2.18->tf-keras) (0.7.2)
Requirement already satisfied: werkzeug>=1.0.1 in c:\users\admin\anaconda3\l
ib\site-packages (from tensorboard<2.19,>=2.18->tensorflow-intel==2.18.0->te
nsorflow<2.19,>=2.18->tf-keras) (3.0.3)
Requirement already satisfied: MarkupSafe>=2.1.1 in c:\users\admin\anaconda3
\lib\site-packages (from werkzeug>=1.0.1->tensorboard<2.19,>=2.18->tensorflo
w-intel==2.18.0->tensorflow<2.19,>=2.18->tf-keras) (2.1.3)
Requirement already satisfied: markdown-it-py<3.0.0,>=2.2.0 in c:\users\admi
n\anaconda3\lib\site-packages (from rich->keras>=3.5.0->tensorflow-intel==2.
```

```
18.0->tensorflow<2.19,>=2.18->tf-keras) (2.2.0)
Requirement already satisfied: pygments<3.0.0,>=2.13.0 in c:\users\admin\ana
conda3\lib\site-packages (from rich->keras>=3.5.0->tensorflow-intel==2.18.0-
>tensorflow<2.19,>=2.18->tf-keras) (2.15.1)
Requirement already satisfied: mdurl~=0.1 in c:\users\admin\anaconda3\lib\si
te-packages (from markdown-it-py<3.0.0,>=2.2.0->rich->keras>=3.5.0->tensorfl
ow-intel==2.18.0->tensorflow<2.19,>=2.18->tf-keras) (0.1.0)
```

In [2]:

```
import cv2
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd
from mlxtend.preprocessing import TransactionEncoder
from mlxtend.frequent_patterns import apriori, fpmax, fpgrowth, association_
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense
from sklearn.model_selection import train_test_split

import warnings
warnings.filterwarnings("ignore", message=".*utcfromtimestamp.*")
import warnings
warnings.filterwarnings("ignore", category=DeprecationWarning, module="googl
```

```
C:\Users\ADMIN\anaconda3\Lib\site-packages\google\protobuf\internal\well_kno
wn_types.py:91: DeprecationWarning: datetime.datetime.utcfromtimestamp() is
deprecated and scheduled for removal in a future version. Use timezone-aware
objects to represent datetimes in UTC: datetime.datetime.fromtimestamp(times
tamp, datetime.UTC).
  _EPOCH_DATETIME_NAIVE = datetime.datetime.utcfromtimestamp(0)
```

In [3]:

```
path = r'C:\Users\ADMIN\Desktop\nidhi python\dataset'
```

In [4]:

```
dataset = pd.read_csv(r'C:\Users\ADMIN\Desktop\nidhi python\dataset\Grocery_
```

In [62]:

```
import pandas as pd
from mlxtend.frequent_patterns import apriori, association_rules
import seaborn as sns
import matplotlib.pyplot as plt

file_path = r'C:\Users\ADMIN\Desktop\nidhi python\dataset\Grocery_Items_26.c
data = pd.read_csv(file_path)

# Exploding the items in each transaction to analyze individually
all_items = data.apply(pd.Series.explode).stack()

# Getting unique items
unique_items = all_items.unique()

# Getting the counts of each item
item_counts = all_items.value_counts()

# Number of unique items
num_unique_items = len(unique_items)
```

```
# Number of records (transactions)
num_records = len(data)

# Most popular item
most_popular_item = item_counts.idxmax()

# Number of transactions containing the most popular item
most_popular_count = item_counts.max()

# Output the results
print(f"Number of unique items: {num_unique_items}")
print(f"Number of records (transactions): {num_records}")
print(f"Most popular item: {most_popular_item}")
print(f"Number of transactions containing the most popular item: {most_popul
```

```
Number of unique items: 166
Number of records (transactions): 8000
Most popular item: whole milk
Number of transactions containing the most popular item: 1364
```

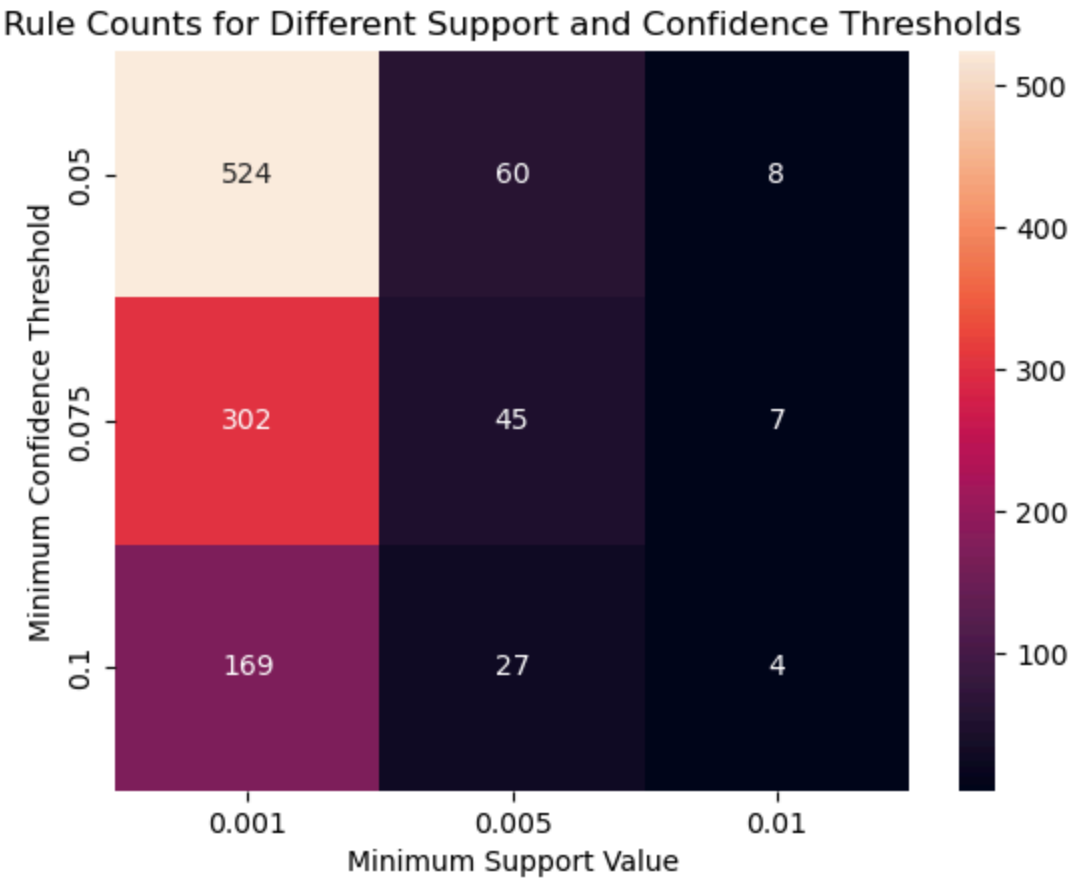
In [66]:

```
import pandas as pd
from mlxtend.preprocessing import TransactionEncoder
from mlxtend.frequent_patterns import apriori, association_rules

# Read the dataset
dataset = pd.read_csv(r'C:\Users\ADMIN\Desktop\nidhi python\dataset\Grocery_

# Prepare the transactions (convert each row into a list of items)
items = []
for i in range(dataset.shape[0]):
    items.append(dataset.iloc[i, :].dropna().tolist())

# Initialize the TransactionEncoder and transform the data into a format su
te = TransactionEncoder()
te_ary = te.fit(items).transform(items)

# Create a DataFrame with the transformed data
df = pd.DataFrame(te_ary, columns=te.columns_)

# Apply the apriori algorithm to find frequent itemsets (min_support is set
frequent_itemsets = apriori(df, min_support=0.01, use_colnames=True)

# Generate association rules based on the frequent itemsets
# Include the 'num_itemsets' argument if required by your version of mlxtenc
rules = association_rules(frequent_itemsets, metric="confidence", min_thresh

# Output the association rules
print(rules)
```

```
          antecedents         consequents  antecedent support  \
0  (other vegetables)        (whole milk)            0.122625
1        (whole milk)  (other vegetables)            0.161500
2        (rolls/buns)        (whole milk)            0.105250
3        (whole milk)        (rolls/buns)            0.161500
4              (soda)        (whole milk)            0.095375
5            (yogurt)        (whole milk)            0.086125

   consequent support   support  confidence      lift  representativity  \
0            0.161500  0.015625    0.127421  0.788985               1.0
1            0.122625  0.015625    0.096749  0.788985               1.0
2            0.161500  0.013250    0.125891  0.779509               1.0
3            0.105250  0.013250    0.082043  0.779509               1.0
4            0.161500  0.011875    0.124509  0.770951               1.0
5            0.161500  0.012625    0.146589  0.907673               1.0

   leverage  conviction  zhangs_metric   jaccard  certainty  kulczynski
0 -0.004179    0.960945      -0.233618  0.058194  -0.040643    0.112085
1 -0.004179    0.971353      -0.241830  0.058194  -0.029492    0.112085
2 -0.003748    0.959262      -0.240197  0.052268  -0.042468    0.103967
3 -0.003748    0.974719      -0.252246  0.052268  -0.025936    0.103967
4 -0.003528    0.957748      -0.247228  0.048469  -0.044116    0.099019
5 -0.001284    0.982528      -0.100156  0.053723  -0.017783    0.112381
```

In [70]:

```
import warnings
import numpy as np  # Import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from mlxtend.frequent_patterns import apriori, association_rules

# Suppress warnings related to deprecated modules
warnings.filterwarnings("ignore", message=".*backend2gui.*")

# Define the values for minimum support and minimum confidence threshold
msv_values = [0.001, 0.005, 0.01]
mct_values = [0.05, 0.075, 0.1]

# Initialize the rule counts matrix
rule_counts = np.zeros((len(mct_values), len(msv_values)))

# Loop over the support values and confidence thresholds to generate rules
for i, msv in enumerate(msv_values):
    frequent_itemsets = apriori(df, min_support=msv, use_colnames=True)
    for j, mct in enumerate(mct_values):
        rules = association_rules(frequent_itemsets, metric="confidence", mi
        rule_counts[j, i] = len(rules)

# Create a heatmap to visualize the rule counts for different combinations c
sns.heatmap(rule_counts, annot=True, xticklabels=msv_values, yticklabels=mct
plt.xlabel('Minimum Support Value')
plt.ylabel('Minimum Confidence Threshold')
plt.title('Rule Counts for Different Support and Confidence Thresholds')
plt.show()
```



In [7]:
```
crop_images_path = r'C:\Users\ADMIN\Desktop\nidhi python\dataset\cropped\ima
```

In [8]:
```
import glob
import cv2
import numpy as np
from tensorflow import keras

# Class names corresponding to the dataset
class_names = ['n02092002-Scottish_deerhound', 'n02093991-Irish_terrier', 'r

# List of base paths for each class (the root folder for each class)
crop_images_paths = [
    r'C:\Users\ADMIN\Desktop\nidhi python\dataset\cropped\images\n02092002-S
    r'C:\Users\ADMIN\Desktop\nidhi python\dataset\cropped\images\n02093991-I
    r'C:\Users\ADMIN\Desktop\nidhi python\dataset\cropped\images\n02097474-T
    r'C:\Users\ADMIN\Desktop\nidhi python\dataset\cropped\images\n02106166-B
]

image_paths = []
classes = []
labels = []
i = 0

# Load the image paths and labels
for class_name, base_path in zip(class_names, crop_images_paths):  # Use zip
    # Construct the full path for the images of this class
    paths = glob.glob(base_path + '/*')  # '*' matches all files in the dire
    print(f"Paths for {class_name}: {paths[:5]}")  # Debug print for paths
```

```
    # Add the paths and corresponding labels to the lists
    image_paths.extend(paths)
    classes.extend([class_name] * len(paths))
    labels.extend([i] * len(paths))

    i += 1

print(f"Total images found: {len(image_paths)}")
print(f"Total labels: {len(labels)}")

# Ensure images are loaded
def load_images(image_paths):
    images = []
    for path in image_paths:
        img = cv2.imread(path, cv2.IMREAD_GRAYSCALE)  # Grayscale for single
        if img is not None:
            images.append(img)
        else:
            print(f"Failed to load image at {path}")
    return images

# Load and resize images
x_train = load_images(image_paths)
print(f"Number of images loaded: {len(x_train)}")  # Debug print

# Resize images to (100, 100)
x_train = np.array([cv2.resize(img, (100, 100)) for img in x_train]) / 255.0

# Reshape for CNN input (add channel dimension)
x_train = x_train.reshape(-1, 100, 100, 1)
print(f"x_train shape: {x_train.shape}")  # Debug print for shape

# Convert labels to categorical
num_classes = 4
y_train = keras.utils.to_categorical(labels, num_classes)

# Verify that x_train and y_train have the correct shapes
print(f"y_train shape: {y_train.shape}")
print(f"x_train shape after reshape: {x_train.shape}")
```

```
Paths for n02092002-Scottish_deerhound: ['C:\\Users\\ADMIN\\Desktop\\nidhi p
ython\\dataset\\cropped\\images\\n02092002-Scottish_deerhound\\n02092002_100
60.jpg', 'C:\\Users\\ADMIN\\Desktop\\nidhi python\\dataset\\cropped\\images
\\n02092002-Scottish_deerhound\\n02092002_1029.jpg', 'C:\\Users\\ADMIN\\Desk
top\\nidhi python\\dataset\\cropped\\images\\n02092002-Scottish_deerhound\\n
02092002_10693.jpg', 'C:\\Users\\ADMIN\\Desktop\\nidhi python\\dataset\\crop
ped\\images\\n02092002-Scottish_deerhound\\n02092002_10699.jpg', 'C:\\Users
\\ADMIN\\Desktop\\nidhi python\\dataset\\cropped\\images\\n02092002-Scottish
_deerhound\\n02092002_1086.jpg']
Paths for n02093991-Irish_terrier: ['C:\\Users\\ADMIN\\Desktop\\nidhi python
\\dataset\\cropped\\images\\n02093991-Irish_terrier\\n02093991_1026.jpg',
'C:\\Users\\ADMIN\\Desktop\\nidhi python\\dataset\\cropped\\images\\n0209399
1-Irish_terrier\\n02093991_1038.jpg', 'C:\\Users\\ADMIN\\Desktop\\nidhi pyth
on\\dataset\\cropped\\images\\n02093991-Irish_terrier\\n02093991_1105.jpg',
'C:\\Users\\ADMIN\\Desktop\\nidhi python\\dataset\\cropped\\images\\n0209399
1-Irish_terrier\\n02093991_114.jpg', 'C:\\Users\\ADMIN\\Desktop\\nidhi pytho
n\\dataset\\cropped\\images\\n02093991-Irish_terrier\\n02093991_1142.jpg']
Paths for n02097474-Tibetan_terrier: ['C:\\Users\\ADMIN\\Desktop\\nidhi pyth
on\\dataset\\cropped\\images\\n02097474-Tibetan_terrier\\n02097474_1023.jp
g', 'C:\\Users\\ADMIN\\Desktop\\nidhi python\\dataset\\cropped\\images\\n020
97474-Tibetan_terrier\\n02097474_1070.jpg', 'C:\\Users\\ADMIN\\Desktop\\nidh
i python\\dataset\\cropped\\images\\n02097474-Tibetan_terrier\\n02097474_109
5.jpg', 'C:\\Users\\ADMIN\\Desktop\\nidhi python\\dataset\\cropped\\images
\\n02097474-Tibetan_terrier\\n02097474_1156.jpg', 'C:\\Users\\ADMIN\\Desktop
\\nidhi python\\dataset\\cropped\\images\\n02097474-Tibetan_terrier\\n020974
74_120.jpg']
Paths for n02106166-Border_collie: ['C:\\Users\\ADMIN\\Desktop\\nidhi python
\\dataset\\cropped\\images\\n02106166-Border_collie\\n02106166_1031.jpg',
'C:\\Users\\ADMIN\\Desktop\\nidhi python\\dataset\\cropped\\images\\n0210616
6-Border_collie\\n02106166_1032.jpg', 'C:\\Users\\ADMIN\\Desktop\\nidhi pyth
on\\dataset\\cropped\\images\\n02106166-Border_collie\\n02106166_1055.jpg',
'C:\\Users\\ADMIN\\Desktop\\nidhi python\\dataset\\cropped\\images\\n0210616
6-Border_collie\\n02106166_1056.jpg', 'C:\\Users\\ADMIN\\Desktop\\nidhi pyth
on\\dataset\\cropped\\images\\n02106166-Border_collie\\n02106166_1059.jpg']
Total images found: 757
Total labels: 757
Number of images loaded: 757
x_train shape: (757, 100, 100, 1)
y_train shape: (757, 4)
x_train shape after reshape: (757, 100, 100, 1)
```

In [9]:

```python
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Ir

# Define the model using the Sequential API with the Input layer
model = Sequential([
    # Input layer (explicit input shape)
    Input(shape=(100, 100, 1)),

    # First Convolutional Layer with 8 3x3 filters
    Conv2D(8, kernel_size=(3, 3), activation='relu'),
    MaxPooling2D(pool_size=(2, 2)),

    # Second Convolutional Layer with 4 3x3 filters
    Conv2D(4, kernel_size=(3, 3), activation='relu'),
    MaxPooling2D(pool_size=(2, 2)),
```

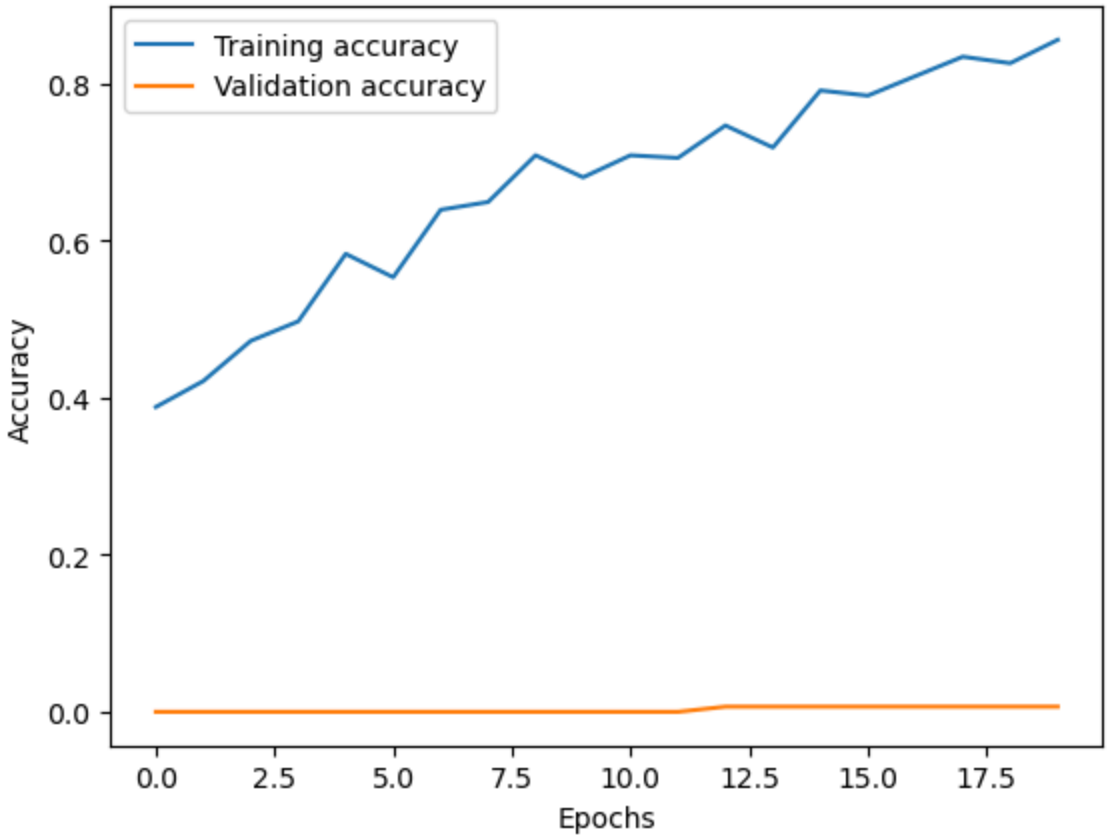
```
    # Flatten the Tensor
    Flatten(),

    # Hidden layer with 8 nodes for fully connected neural network
    Dense(8, activation='relu'),

    # Output layer with 4 nodes using softmax activation
    Dense(num_classes, activation='softmax')
])

# Compile the model
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['a

# Train the model
history = model.fit(x_train, y_train, epochs=20, batch_size=32, validation_s
```

```
Epoch 1/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 4s 43ms/step - accuracy: 0.3643 - loss: 1.2746 -
val_accuracy: 0.0000e+00 - val_loss: 4.7371
Epoch 2/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 22ms/step - accuracy: 0.4358 - loss: 1.0795 -
val_accuracy: 0.0000e+00 - val_loss: 5.6048
Epoch 3/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 23ms/step - accuracy: 0.4841 - loss: 1.0355 -
val_accuracy: 0.0000e+00 - val_loss: 5.8581
Epoch 4/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 22ms/step - accuracy: 0.4955 - loss: 0.9870 -
val_accuracy: 0.0000e+00 - val_loss: 5.8338
Epoch 5/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 22ms/step - accuracy: 0.5887 - loss: 0.9310 -
val_accuracy: 0.0000e+00 - val_loss: 6.3757
Epoch 6/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 23ms/step - accuracy: 0.5586 - loss: 0.9158 -
val_accuracy: 0.0000e+00 - val_loss: 5.8309
Epoch 7/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 20ms/step - accuracy: 0.6410 - loss: 0.8849 -
val_accuracy: 0.0000e+00 - val_loss: 6.6939
Epoch 8/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 21ms/step - accuracy: 0.6450 - loss: 0.8383 -
val_accuracy: 0.0000e+00 - val_loss: 6.8199
Epoch 9/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - accuracy: 0.7159 - loss: 0.7765 -
val_accuracy: 0.0000e+00 - val_loss: 8.7642
Epoch 10/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 1s 25ms/step - accuracy: 0.6794 - loss: 0.7314 -
val_accuracy: 0.0000e+00 - val_loss: 9.1291
Epoch 11/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 1s 25ms/step - accuracy: 0.6846 - loss: 0.7290 -
val_accuracy: 0.0000e+00 - val_loss: 8.6205
Epoch 12/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 1s 27ms/step - accuracy: 0.7061 - loss: 0.6997 -
val_accuracy: 0.0000e+00 - val_loss: 8.0391
Epoch 13/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 1s 28ms/step - accuracy: 0.7604 - loss: 0.6690 -
val_accuracy: 0.0066 - val_loss: 8.1560
Epoch 14/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 1s 28ms/step - accuracy: 0.7475 - loss: 0.6482 -
val_accuracy: 0.0066 - val_loss: 7.4219
Epoch 15/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 1s 32ms/step - accuracy: 0.7850 - loss: 0.5933 -
val_accuracy: 0.0066 - val_loss: 9.7397
Epoch 16/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 1s 28ms/step - accuracy: 0.7798 - loss: 0.5504 -
val_accuracy: 0.0066 - val_loss: 10.7638
Epoch 17/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 1s 28ms/step - accuracy: 0.8317 - loss: 0.5276 -
val_accuracy: 0.0066 - val_loss: 10.9940
Epoch 18/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 1s 33ms/step - accuracy: 0.8121 - loss: 0.5348 -
val_accuracy: 0.0066 - val_loss: 10.5916
Epoch 19/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 1s 27ms/step - accuracy: 0.8403 - loss: 0.4803 -
```

```
val_accuracy: 0.0066 - val_loss: 11.0636
Epoch 20/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 1s 28ms/step - accuracy: 0.8527 - loss: 0.4518 -
val_accuracy: 0.0066 - val_loss: 12.3019
```

In [10]:

```
plt.plot(history.history['accuracy'], label='Training accuracy')
plt.plot(history.history['val_accuracy'], label='Validation accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()
plt.show()
```



In [11]:

```
print('Banner ID is 916489638')
```

```
Banner ID is 916489638
```

In [12]:

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Ir
num_classes = 4

model_4_nodes = Sequential([

    Input(shape=(100, 100, 1)),  # Define the input shape directly with Inpu

    # First convolutional layer with 8 filters of size 3x3
    Conv2D(8, kernel_size=(3, 3), activation='relu'),
    # Max pooling with 2x2 pool size
    MaxPooling2D(pool_size=(2, 2)),

    # Flatten the tensor to prepare for the fully connected layer
```

```
    Flatten(),

    # Fully connected hidden layer with 4 nodes and ReLU activation
    Dense(4, activation='relu'),

    # Output layer with 4 nodes for the classes, using softmax activation
    Dense(num_classes, activation='softmax')
])

# Compile the model
model_4_nodes.compile(loss='categorical_crossentropy', optimizer='adam', met


# Train the model (assuming you have already defined x_train and y_train)
history_4_nodes = model_4_nodes.fit(x_train, y_train, epochs=20, batch_size=
```

```
Epoch 1/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 4s 43ms/step - accuracy: 0.3791 - loss: 1.3108 -
val_accuracy: 0.0000e+00 - val_loss: 2.4279
Epoch 2/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 20ms/step - accuracy: 0.4433 - loss: 1.1810 -
val_accuracy: 0.0066 - val_loss: 2.7097
Epoch 3/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 20ms/step - accuracy: 0.5059 - loss: 1.1247 -
val_accuracy: 0.0000e+00 - val_loss: 2.8032
Epoch 4/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 20ms/step - accuracy: 0.5568 - loss: 1.0312 -
val_accuracy: 0.0000e+00 - val_loss: 2.4274
Epoch 5/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 19ms/step - accuracy: 0.6036 - loss: 0.9843 -
val_accuracy: 0.0066 - val_loss: 2.9348
Epoch 6/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 18ms/step - accuracy: 0.6142 - loss: 0.9311 -
val_accuracy: 0.0000e+00 - val_loss: 3.1929
Epoch 7/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 20ms/step - accuracy: 0.6659 - loss: 0.8549 -
val_accuracy: 0.0000e+00 - val_loss: 3.4933
Epoch 8/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 20ms/step - accuracy: 0.7131 - loss: 0.7791 -
val_accuracy: 0.0066 - val_loss: 3.2218
Epoch 9/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 19ms/step - accuracy: 0.6946 - loss: 0.7554 -
val_accuracy: 0.0132 - val_loss: 3.4291
Epoch 10/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 20ms/step - accuracy: 0.6944 - loss: 0.6995 -
val_accuracy: 0.0066 - val_loss: 3.5046
Epoch 11/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 21ms/step - accuracy: 0.7373 - loss: 0.6098 -
val_accuracy: 0.0132 - val_loss: 3.7695
Epoch 12/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 21ms/step - accuracy: 0.7245 - loss: 0.6170 -
val_accuracy: 0.0132 - val_loss: 4.5179
Epoch 13/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 1s 26ms/step - accuracy: 0.7087 - loss: 0.6145 -
val_accuracy: 0.0000e+00 - val_loss: 4.3598
Epoch 14/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 1s 24ms/step - accuracy: 0.7417 - loss: 0.5318 -
val_accuracy: 0.0132 - val_loss: 4.0693
Epoch 15/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 1s 23ms/step - accuracy: 0.7730 - loss: 0.5128 -
val_accuracy: 0.0132 - val_loss: 4.4244
Epoch 16/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 1s 24ms/step - accuracy: 0.7767 - loss: 0.4958 -
val_accuracy: 0.0066 - val_loss: 4.6162
Epoch 17/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 19ms/step - accuracy: 0.7939 - loss: 0.4550 -
val_accuracy: 0.0132 - val_loss: 4.7295
Epoch 18/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 1s 23ms/step - accuracy: 0.8095 - loss: 0.4588 -
val_accuracy: 0.0066 - val_loss: 4.7341
Epoch 19/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 20ms/step - accuracy: 0.8384 - loss: 0.4219 -
```

```
val_accuracy: 0.0132 - val_loss: 4.4199
Epoch 20/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 21ms/step - accuracy: 0.8647 - loss: 0.4038 -
val_accuracy: 0.0132 - val_loss: 4.8613
```

In [13]:

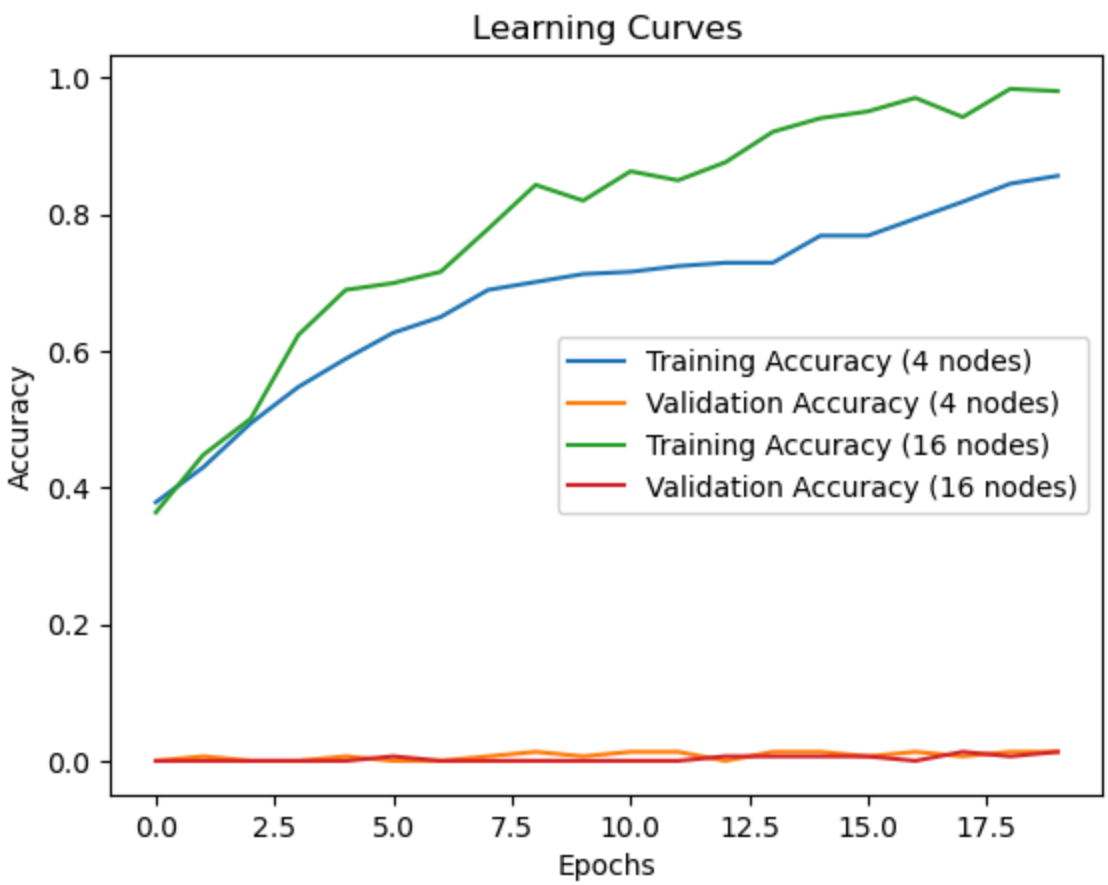
```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Ir
num_classes = 4

model_16_nodes = Sequential([

    Input(shape=(100, 100, 1)),


    Conv2D(8, kernel_size=(3, 3), activation='relu'),
    # Max pooling with 2x2 pool size
    MaxPooling2D(pool_size=(2, 2)),

    # Flatten the tensor to prepare for the fully connected layer
    Flatten(),

    # Fully connected hidden layer with 16 nodes and ReLU activation
    Dense(16, activation='relu'),

    # Output layer with 4 nodes for the classes, using softmax activation
    Dense(num_classes, activation='softmax')
])

# Compile the model
model_16_nodes.compile(loss='categorical_crossentropy', optimizer='adam', me


# Train the model (assuming you have already defined x_train and y_train)
history_16_nodes = model_16_nodes.fit(x_train, y_train, epochs=20, batch_siz
```

```
Epoch 1/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 3s 42ms/step - accuracy: 0.3726 - loss: 1.1814 -
val_accuracy: 0.0000e+00 - val_loss: 8.8213
Epoch 2/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 22ms/step - accuracy: 0.4431 - loss: 1.0304 -
val_accuracy: 0.0000e+00 - val_loss: 8.5865
Epoch 3/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 1s 27ms/step - accuracy: 0.4974 - loss: 1.0128 -
val_accuracy: 0.0000e+00 - val_loss: 6.8587
Epoch 4/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 1s 28ms/step - accuracy: 0.6248 - loss: 0.9488 -
val_accuracy: 0.0000e+00 - val_loss: 7.4464
Epoch 5/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 1s 28ms/step - accuracy: 0.6645 - loss: 0.8735 -
val_accuracy: 0.0000e+00 - val_loss: 9.2984
Epoch 6/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 1s 27ms/step - accuracy: 0.7570 - loss: 0.8057 -
val_accuracy: 0.0066 - val_loss: 10.5846
Epoch 7/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 1s 25ms/step - accuracy: 0.6827 - loss: 0.7994 -
val_accuracy: 0.0000e+00 - val_loss: 10.6239
Epoch 8/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 22ms/step - accuracy: 0.7821 - loss: 0.7153 -
val_accuracy: 0.0000e+00 - val_loss: 11.1922
Epoch 9/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 22ms/step - accuracy: 0.8546 - loss: 0.6356 -
val_accuracy: 0.0000e+00 - val_loss: 12.5476
Epoch 10/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 1s 23ms/step - accuracy: 0.8038 - loss: 0.6180 -
val_accuracy: 0.0000e+00 - val_loss: 12.8867
Epoch 11/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 21ms/step - accuracy: 0.8738 - loss: 0.5454 -
val_accuracy: 0.0000e+00 - val_loss: 12.5765
Epoch 12/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 22ms/step - accuracy: 0.8724 - loss: 0.5193 -
val_accuracy: 0.0000e+00 - val_loss: 12.5144
Epoch 13/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 22ms/step - accuracy: 0.8569 - loss: 0.5208 -
val_accuracy: 0.0066 - val_loss: 12.1059
Epoch 14/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 22ms/step - accuracy: 0.9048 - loss: 0.4307 -
val_accuracy: 0.0066 - val_loss: 12.7926
Epoch 15/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 22ms/step - accuracy: 0.9359 - loss: 0.3943 -
val_accuracy: 0.0066 - val_loss: 13.9716
Epoch 16/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 1s 27ms/step - accuracy: 0.9346 - loss: 0.3491 -
val_accuracy: 0.0066 - val_loss: 14.0519
Epoch 17/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 24ms/step - accuracy: 0.9542 - loss: 0.3317 -
val_accuracy: 0.0000e+00 - val_loss: 15.5679
Epoch 18/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 22ms/step - accuracy: 0.9398 - loss: 0.3019 -
val_accuracy: 0.0132 - val_loss: 14.5962
Epoch 19/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 22ms/step - accuracy: 0.9851 - loss: 0.2479 -
```

```
val_accuracy: 0.0066 - val_loss: 15.3029
Epoch 20/20
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 23ms/step - accuracy: 0.9761 - loss: 0.2325 -
val_accuracy: 0.0132 - val_loss: 15.4322
```

In [14]:

```
plt.plot(history_4_nodes.history['accuracy'], label='Training Accuracy (4 r
plt.plot(history_4_nodes.history['val_accuracy'], label='Validation Accurac
# Plot for model with 16 nodes
plt.plot(history_16_nodes.history['accuracy'], label='Training Accuracy (16
plt.plot(history_16_nodes.history['val_accuracy'], label='Validation Accura
plt.title('Learning Curves')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()
plt.show()
```



In [15]:

```
print('''
 Model 1: Overfitting, since the training accuracy is significantly higher t
 Model 2: Underfitting, the model has issue with data hence the straight lir
hidden layers.
 Model 3: Right fit, the training and validation accuracies are increasing s
significant difference between them.
 ''')
```

```
Model 1: Overfitting, since the training accuracy is significantly higher th
an the validation accuracy.
Model 2: Underfitting, the model has issue with data hence the straight line
curves, this could be because of less
hidden layers.
Model 3: Right fit, the training and validation accuracies are increasing si
mulatenously, unlike model 1 there is no
significant difference between them.
```

In [16]:
```
!pip uninstall -y accelerate
```

```
Found existing installation: accelerate 0.26.0
Uninstalling accelerate-0.26.0:
  Successfully uninstalled accelerate-0.26.0
```

In [17]:
```
!pip install --no-cache-dir accelerate==0.26.0
```

```
Collecting accelerate==0.26.0
  Downloading accelerate-0.26.0-py3-none-any.whl.metadata (18 kB)
Requirement already satisfied: numpy>=1.17 in c:\users\admin\anaconda3\lib\s
ite-packages (from accelerate==0.26.0) (1.26.4)
Requirement already satisfied: packaging>=20.0 in c:\users\admin\anaconda3\l
ib\site-packages (from accelerate==0.26.0) (23.2)
Requirement already satisfied: psutil in c:\users\admin\anaconda3\lib\site-p
ackages (from accelerate==0.26.0) (5.9.0)
Requirement already satisfied: pyyaml in c:\users\admin\anaconda3\lib\site-p
ackages (from accelerate==0.26.0) (6.0.1)
Requirement already satisfied: torch>=1.10.0 in c:\users\admin\anaconda3\lib
\site-packages (from accelerate==0.26.0) (2.5.1)
Requirement already satisfied: huggingface-hub in c:\users\admin\anaconda3\l
ib\site-packages (from accelerate==0.26.0) (0.26.2)
Requirement already satisfied: safetensors>=0.3.1 in c:\users\admin\anaconda
3\lib\site-packages (from accelerate==0.26.0) (0.4.5)
Requirement already satisfied: filelock in c:\users\admin\anaconda3\lib\site
-packages (from torch>=1.10.0->accelerate==0.26.0) (3.13.1)
Requirement already satisfied: typing-extensions>=4.8.0 in c:\users\admin\an
aconda3\lib\site-packages (from torch>=1.10.0->accelerate==0.26.0) (4.11.0)
Requirement already satisfied: networkx in c:\users\admin\anaconda3\lib\site
-packages (from torch>=1.10.0->accelerate==0.26.0) (3.2.1)
Requirement already satisfied: jinja2 in c:\users\admin\anaconda3\lib\site-p
ackages (from torch>=1.10.0->accelerate==0.26.0) (3.1.4)
Requirement already satisfied: fsspec in c:\users\admin\anaconda3\lib\site-p
ackages (from torch>=1.10.0->accelerate==0.26.0) (2024.3.1)
Requirement already satisfied: setuptools in c:\users\admin\anaconda3\lib\si
te-packages (from torch>=1.10.0->accelerate==0.26.0) (69.5.1)
Requirement already satisfied: sympy==1.13.1 in c:\users\admin\anaconda3\lib
\site-packages (from torch>=1.10.0->accelerate==0.26.0) (1.13.1)
Requirement already satisfied: mpmath<1.4,>=1.1.0 in c:\users\admin\anaconda
3\lib\site-packages (from sympy==1.13.1->torch>=1.10.0->accelerate==0.26.0)
(1.3.0)
Requirement already satisfied: requests in c:\users\admin\anaconda3\lib\site
-packages (from huggingface-hub->accelerate==0.26.0) (2.32.2)
Requirement already satisfied: tqdm>=4.42.1 in c:\users\admin\anaconda3\lib
\site-packages (from huggingface-hub->accelerate==0.26.0) (4.66.4)
Requirement already satisfied: colorama in c:\users\admin\anaconda3\lib\site
-packages (from tqdm>=4.42.1->huggingface-hub->accelerate==0.26.0) (0.4.6)
Requirement already satisfied: MarkupSafe>=2.0 in c:\users\admin\anaconda3\l
ib\site-packages (from jinja2->torch>=1.10.0->accelerate==0.26.0) (2.1.3)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\admin\an
aconda3\lib\site-packages (from requests->huggingface-hub->accelerate==0.26.
0) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in c:\users\admin\anaconda3\lib
\site-packages (from requests->huggingface-hub->accelerate==0.26.0) (3.7)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\users\admin\anaconda
3\lib\site-packages (from requests->huggingface-hub->accelerate==0.26.0) (2.
2.2)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\admin\anaconda
3\lib\site-packages (from requests->huggingface-hub->accelerate==0.26.0) (20
24.8.30)
Downloading accelerate-0.26.0-py3-none-any.whl (270 kB)
   ---------------------------------------- 0.0/270.7 kB ? eta -:--:--
   - -------------------------------------- 10.2/270.7 kB ? eta -:--:--
   -------------------------------------- 10.2/270.7 kB ? eta -:--:--
```

```
   ----- ---------------------------------- 41.0/270.7 kB 326.8 kB/s eta 0:0
0:01
   ------------- ------------------------- 92.2/270.7 kB 581.0 kB/s eta 0:0
0:01
   ---------------------------------------- 270.7/270.7 kB 1.4 MB/s eta 0:0
0:00
Installing collected packages: accelerate
Successfully installed accelerate-0.26.0
```

In [18]:
```
pip install transformers[torch]
```

```
Requirement already satisfied: transformers[torch] in c:\users\admin\anacond
a3\lib\site-packages (4.46.3)
Requirement already satisfied: filelock in c:\users\admin\anaconda3\lib\site
-packages (from transformers[torch]) (3.13.1)
Requirement already satisfied: huggingface-hub<1.0,>=0.23.2 in c:\users\admi
n\anaconda3\lib\site-packages (from transformers[torch]) (0.26.2)
Requirement already satisfied: numpy>=1.17 in c:\users\admin\anaconda3\lib\s
ite-packages (from transformers[torch]) (1.26.4)
Requirement already satisfied: packaging>=20.0 in c:\users\admin\anaconda3\l
ib\site-packages (from transformers[torch]) (23.2)
Requirement already satisfied: pyyaml>=5.1 in c:\users\admin\anaconda3\lib\s
ite-packages (from transformers[torch]) (6.0.1)
Requirement already satisfied: regex!=2019.12.17 in c:\users\admin\anaconda3
\lib\site-packages (from transformers[torch]) (2023.10.3)
Requirement already satisfied: requests in c:\users\admin\anaconda3\lib\site
-packages (from transformers[torch]) (2.32.2)
Requirement already satisfied: tokenizers<0.21,>=0.20 in c:\users\admin\anac
onda3\lib\site-packages (from transformers[torch]) (0.20.3)
Requirement already satisfied: safetensors>=0.4.1 in c:\users\admin\anaconda
3\lib\site-packages (from transformers[torch]) (0.4.5)
Requirement already satisfied: tqdm>=4.27 in c:\users\admin\anaconda3\lib\si
te-packages (from transformers[torch]) (4.66.4)
Requirement already satisfied: torch in c:\users\admin\anaconda3\lib\site-pa
ckages (from transformers[torch]) (2.5.1)
Requirement already satisfied: accelerate>=0.26.0 in c:\users\admin\anaconda
3\lib\site-packages (from transformers[torch]) (0.26.0)
Requirement already satisfied: psutil in c:\users\admin\anaconda3\lib\site-p
ackages (from accelerate>=0.26.0->transformers[torch]) (5.9.0)
Requirement already satisfied: fsspec>=2023.5.0 in c:\users\admin\anaconda3
\lib\site-packages (from huggingface-hub<1.0,>=0.23.2->transformers[torch])
(2024.3.1)
Requirement already satisfied: typing-extensions>=3.7.4.3 in c:\users\admin
\anaconda3\lib\site-packages (from huggingface-hub<1.0,>=0.23.2->transformer
s[torch]) (4.11.0)
Requirement already satisfied: networkx in c:\users\admin\anaconda3\lib\site
-packages (from torch->transformers[torch]) (3.2.1)
Requirement already satisfied: jinja2 in c:\users\admin\anaconda3\lib\site-p
ackages (from torch->transformers[torch]) (3.1.4)
Requirement already satisfied: setuptools in c:\users\admin\anaconda3\lib\si
te-packages (from torch->transformers[torch]) (69.5.1)
Requirement already satisfied: sympy==1.13.1 in c:\users\admin\anaconda3\lib
\site-packages (from torch->transformers[torch]) (1.13.1)
Requirement already satisfied: mpmath<1.4,>=1.1.0 in c:\users\admin\anaconda
3\lib\site-packages (from sympy==1.13.1->torch->transformers[torch]) (1.3.0)
Requirement already satisfied: colorama in c:\users\admin\anaconda3\lib\site
-packages (from tqdm>=4.27->transformers[torch]) (0.4.6)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\admin\an
aconda3\lib\site-packages (from requests->transformers[torch]) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in c:\users\admin\anaconda3\lib
\site-packages (from requests->transformers[torch]) (3.7)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\users\admin\anaconda
3\lib\site-packages (from requests->transformers[torch]) (2.2.2)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\admin\anaconda
3\lib\site-packages (from requests->transformers[torch]) (2024.8.30)
Requirement already satisfied: MarkupSafe>=2.0 in c:\users\admin\anaconda3\l
```

```
ib\site-packages (from jinja2->torch->transformers[torch]) (2.1.3)
Note: you may need to restart the kernel to use updated packages.
```

In [19]:
```
pip install --upgrade transformers accelerate
```

```
Requirement already satisfied: transformers in c:\users\admin\anaconda3\lib
\site-packages (4.46.3)
Requirement already satisfied: accelerate in c:\users\admin\anaconda3\lib\si
te-packages (0.26.0)
Collecting accelerate
  Using cached accelerate-1.1.1-py3-none-any.whl.metadata (19 kB)
Requirement already satisfied: filelock in c:\users\admin\anaconda3\lib\site
-packages (from transformers) (3.13.1)
Requirement already satisfied: huggingface-hub<1.0,>=0.23.2 in c:\users\admi
n\anaconda3\lib\site-packages (from transformers) (0.26.2)
Requirement already satisfied: numpy>=1.17 in c:\users\admin\anaconda3\lib\s
ite-packages (from transformers) (1.26.4)
Requirement already satisfied: packaging>=20.0 in c:\users\admin\anaconda3\l
ib\site-packages (from transformers) (23.2)
Requirement already satisfied: pyyaml>=5.1 in c:\users\admin\anaconda3\lib\s
ite-packages (from transformers) (6.0.1)
Requirement already satisfied: regex!=2019.12.17 in c:\users\admin\anaconda3
\lib\site-packages (from transformers) (2023.10.3)
Requirement already satisfied: requests in c:\users\admin\anaconda3\lib\site
-packages (from transformers) (2.32.2)
Requirement already satisfied: tokenizers<0.21,>=0.20 in c:\users\admin\anac
onda3\lib\site-packages (from transformers) (0.20.3)
Requirement already satisfied: safetensors>=0.4.1 in c:\users\admin\anaconda
3\lib\site-packages (from transformers) (0.4.5)
Requirement already satisfied: tqdm>=4.27 in c:\users\admin\anaconda3\lib\si
te-packages (from transformers) (4.66.4)
Requirement already satisfied: psutil in c:\users\admin\anaconda3\lib\site-p
ackages (from accelerate) (5.9.0)
Requirement already satisfied: torch>=1.10.0 in c:\users\admin\anaconda3\lib
\site-packages (from accelerate) (2.5.1)
Requirement already satisfied: fsspec>=2023.5.0 in c:\users\admin\anaconda3
\lib\site-packages (from huggingface-hub<1.0,>=0.23.2->transformers) (2024.
3.1)
Requirement already satisfied: typing-extensions>=3.7.4.3 in c:\users\admin
\anaconda3\lib\site-packages (from huggingface-hub<1.0,>=0.23.2->transformer
s) (4.11.0)
Requirement already satisfied: networkx in c:\users\admin\anaconda3\lib\site
-packages (from torch>=1.10.0->accelerate) (3.2.1)
Requirement already satisfied: jinja2 in c:\users\admin\anaconda3\lib\site-p
ackages (from torch>=1.10.0->accelerate) (3.1.4)
Requirement already satisfied: setuptools in c:\users\admin\anaconda3\lib\si
te-packages (from torch>=1.10.0->accelerate) (69.5.1)
Requirement already satisfied: sympy==1.13.1 in c:\users\admin\anaconda3\lib
\site-packages (from torch>=1.10.0->accelerate) (1.13.1)
Requirement already satisfied: mpmath<1.4,>=1.1.0 in c:\users\admin\anaconda
3\lib\site-packages (from sympy==1.13.1->torch>=1.10.0->accelerate) (1.3.0)
Requirement already satisfied: colorama in c:\users\admin\anaconda3\lib\site
-packages (from tqdm>=4.27->transformers) (0.4.6)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\admin\an
aconda3\lib\site-packages (from requests->transformers) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in c:\users\admin\anaconda3\lib
\site-packages (from requests->transformers) (3.7)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\users\admin\anaconda
3\lib\site-packages (from requests->transformers) (2.2.2)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\admin\anaconda
3\lib\site-packages (from requests->transformers) (2024.8.30)
```

```
Requirement already satisfied: MarkupSafe>=2.0 in c:\users\admin\anaconda3\l
ib\site-packages (from jinja2->torch>=1.10.0->accelerate) (2.1.3)
Using cached accelerate-1.1.1-py3-none-any.whl (333 kB)
Installing collected packages: accelerate
  Attempting uninstall: accelerate
    Found existing installation: accelerate 0.26.0
    Uninstalling accelerate-0.26.0:
      Successfully uninstalled accelerate-0.26.0
Successfully installed accelerate-1.1.1
Note: you may need to restart the kernel to use updated packages.
```

In [20]:
```
pip install --upgrade datasets
```

```
Requirement already satisfied: datasets in c:\users\admin\anaconda3\lib\site
-packages (3.1.0)
Requirement already satisfied: filelock in c:\users\admin\anaconda3\lib\site
-packages (from datasets) (3.13.1)
Requirement already satisfied: numpy>=1.17 in c:\users\admin\anaconda3\lib\s
ite-packages (from datasets) (1.26.4)
Requirement already satisfied: pyarrow>=15.0.0 in c:\users\admin\anaconda3\l
ib\site-packages (from datasets) (18.0.0)
Requirement already satisfied: dill<0.3.9,>=0.3.0 in c:\users\admin\anaconda
3\lib\site-packages (from datasets) (0.3.8)
Requirement already satisfied: pandas in c:\users\admin\anaconda3\lib\site-p
ackages (from datasets) (2.2.2)
Requirement already satisfied: requests>=2.32.2 in c:\users\admin\anaconda3
\lib\site-packages (from datasets) (2.32.2)
Requirement already satisfied: tqdm>=4.66.3 in c:\users\admin\anaconda3\lib
\site-packages (from datasets) (4.66.4)
Requirement already satisfied: xxhash in c:\users\admin\anaconda3\lib\site-p
ackages (from datasets) (3.5.0)
Requirement already satisfied: multiprocess<0.70.17 in c:\users\admin\anacon
da3\lib\site-packages (from datasets) (0.70.16)
Requirement already satisfied: fsspec<=2024.9.0,>=2023.1.0 in c:\users\admin
\anaconda3\lib\site-packages (from fsspec[http]<=2024.9.0,>=2023.1.0->datase
ts) (2024.3.1)
Requirement already satisfied: aiohttp in c:\users\admin\anaconda3\lib\site-
packages (from datasets) (3.9.5)
Requirement already satisfied: huggingface-hub>=0.23.0 in c:\users\admin\ana
conda3\lib\site-packages (from datasets) (0.26.2)
Requirement already satisfied: packaging in c:\users\admin\anaconda3\lib\sit
e-packages (from datasets) (23.2)
Requirement already satisfied: pyyaml>=5.1 in c:\users\admin\anaconda3\lib\s
ite-packages (from datasets) (6.0.1)
Requirement already satisfied: aiosignal>=1.1.2 in c:\users\admin\anaconda3
\lib\site-packages (from aiohttp->datasets) (1.2.0)
Requirement already satisfied: attrs>=17.3.0 in c:\users\admin\anaconda3\lib
\site-packages (from aiohttp->datasets) (23.1.0)
Requirement already satisfied: frozenlist>=1.1.1 in c:\users\admin\anaconda3
\lib\site-packages (from aiohttp->datasets) (1.4.0)
Requirement already satisfied: multidict<7.0,>=4.5 in c:\users\admin\anacond
a3\lib\site-packages (from aiohttp->datasets) (6.0.4)
Requirement already satisfied: yarl<2.0,>=1.0 in c:\users\admin\anaconda3\li
b\site-packages (from aiohttp->datasets) (1.9.3)
Requirement already satisfied: typing-extensions>=3.7.4.3 in c:\users\admin
\anaconda3\lib\site-packages (from huggingface-hub>=0.23.0->datasets) (4.11.
0)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\admin\an
aconda3\lib\site-packages (from requests>=2.32.2->datasets) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in c:\users\admin\anaconda3\lib
\site-packages (from requests>=2.32.2->datasets) (3.7)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\users\admin\anaconda
3\lib\site-packages (from requests>=2.32.2->datasets) (2.2.2)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\admin\anaconda
3\lib\site-packages (from requests>=2.32.2->datasets) (2024.8.30)
Requirement already satisfied: colorama in c:\users\admin\anaconda3\lib\site
-packages (from tqdm>=4.66.3->datasets) (0.4.6)
Requirement already satisfied: python-dateutil>=2.8.2 in c:\users\admin\anac
onda3\lib\site-packages (from pandas->datasets) (2.9.0.post0)
```

```
Requirement already satisfied: pytz>=2020.1 in c:\users\admin\anaconda3\lib
\site-packages (from pandas->datasets) (2024.1)
Requirement already satisfied: tzdata>=2022.7 in c:\users\admin\anaconda3\li
b\site-packages (from pandas->datasets) (2023.3)
Requirement already satisfied: six>=1.5 in c:\users\admin\anaconda3\lib\site
-packages (from python-dateutil>=2.8.2->pandas->datasets) (1.16.0)
Note: you may need to restart the kernel to use updated packages.
```

In [21]:
```
!pip install torch
```

```
Requirement already satisfied: torch in c:\users\admin\anaconda3\lib\site-pa
ckages (2.5.1)
Requirement already satisfied: filelock in c:\users\admin\anaconda3\lib\site
-packages (from torch) (3.13.1)
Requirement already satisfied: typing-extensions>=4.8.0 in c:\users\admin\an
aconda3\lib\site-packages (from torch) (4.11.0)
Requirement already satisfied: networkx in c:\users\admin\anaconda3\lib\site
-packages (from torch) (3.2.1)
Requirement already satisfied: jinja2 in c:\users\admin\anaconda3\lib\site-p
ackages (from torch) (3.1.4)
Requirement already satisfied: fsspec in c:\users\admin\anaconda3\lib\site-p
ackages (from torch) (2024.3.1)
Requirement already satisfied: setuptools in c:\users\admin\anaconda3\lib\si
te-packages (from torch) (69.5.1)
Requirement already satisfied: sympy==1.13.1 in c:\users\admin\anaconda3\lib
\site-packages (from torch) (1.13.1)
Requirement already satisfied: mpmath<1.4,>=1.1.0 in c:\users\admin\anaconda
3\lib\site-packages (from sympy==1.13.1->torch) (1.3.0)
Requirement already satisfied: MarkupSafe>=2.0 in c:\users\admin\anaconda3\l
ib\site-packages (from jinja2->torch) (2.1.3)
```

In [22]:
```
import torch
from transformers import BertTokenizer, BertForSequenceClassification, AdamW
from torch.utils.data import DataLoader, TensorDataset
import json
import numpy as np
from sklearn.metrics import accuracy_score, f1_score
import matplotlib.pyplot as plt
import torch
from transformers import BertTokenizer
import os

# Function to normalize paths
def normalize_path(filepath):
    return os.path.normpath(filepath)

# Load JSON files
def load_json_file(filepath):
    filepath = normalize_path(filepath)
    with open(filepath, 'r', encoding='utf-8') as f:
        data = [json.loads(line.strip()) for line in f]
    return data

# Corrected file paths
train_data = load_json_file(r'C:\Users\ADMIN\Desktop\nidhi python\dataset\tr
test_data = load_json_file(r'C:\Users\ADMIN\Desktop\nidhi python\dataset\tes
```

```
val_data = load_json_file(r'C:\Users\ADMIN\Desktop\nidhi python\dataset\vali

# Print the structure of the first item in train_data
print("Structure of first item in train_data:")
print(json.dumps(train_data[0], indent=2))

# Define label classes
all_labels = ['anger', 'anticipation', 'disgust', 'fear', 'joy',
              'love', 'optimism', 'pessimism', 'sadness', 'surprise', 'trust

# Convert labels to one-hot encoded lists
def convert_labels_to_list(data, label_classes):
    print("Keys in data item:", list(data[0].keys()))
    for item in data:
        item['labels'] = [float(item[label]) for label in label_classes]
    return data

train_data = convert_labels_to_list(train_data, all_labels)
val_data = convert_labels_to_list(val_data, all_labels)
test_data = convert_labels_to_list(test_data, all_labels)

# Initialize tokenizer
tokenizer = BertTokenizer.from_pretrained('bert-base-uncased')

# Tokenize and encode the texts
def encode_texts(data):
    # Assuming 'tweet' is the key for the text content. Adjust if necessary.
    texts = [item['tweet'] for item in data]
    return tokenizer(texts, padding=True, truncation=True, max_length=128, r

# Try to encode texts and catch any errors
try:
    train_encodings = encode_texts(train_data)
    test_encodings = encode_texts(test_data)
    val_encodings = encode_texts(val_data)
except KeyError as e:
    print(f"KeyError: {e}. The key for text content might be incorrect.")
    print("Available keys:", list(train_data[0].keys()))

# Convert labels to tensors
train_labels = torch.tensor([item['labels'] for item in train_data], dtype=t
test_labels = torch.tensor([item['labels'] for item in test_data], dtype=tor
val_labels = torch.tensor([item['labels'] for item in val_data], dtype=torch

# Print shapes to verify
print("Train labels shape:", train_labels.shape)
print("Test labels shape:", test_labels.shape)
print("Validation labels shape:", val_labels.shape)

# If encodings were successful, print their shapes too
if 'train_encodings' in locals():
    print("Train encodings shape:", train_encodings['input_ids'].shape)
    print("Test encodings shape:", test_encodings['input_ids'].shape)
    print("Validation encodings shape:", val_encodings['input_ids'].shape)
```

```
Structure of first item in train_data:
{
  "ID": "2017-En-10065",
  "Tweet": "In 2016, Black people are STILL fighting to be recognized as hum
an beings. #cantsleep #angry",
  "anger": true,
  "anticipation": false,
  "disgust": true,
  "fear": false,
  "joy": false,
  "love": false,
  "optimism": false,
  "pessimism": false,
  "sadness": false,
  "surprise": false,
  "trust": false
}
Keys in data item: ['ID', 'Tweet', 'anger', 'anticipation', 'disgust', 'fea
r', 'joy', 'love', 'optimism', 'pessimism', 'sadness', 'surprise', 'trust']
Keys in data item: ['ID', 'Tweet', 'anger', 'anticipation', 'disgust', 'fea
r', 'joy', 'love', 'optimism', 'pessimism', 'sadness', 'surprise', 'trust']
Keys in data item: ['ID', 'Tweet', 'anger', 'anticipation', 'disgust', 'fea
r', 'joy', 'love', 'optimism', 'pessimism', 'sadness', 'surprise', 'trust']
KeyError: 'tweet'. The key for text content might be incorrect.
Available keys: ['ID', 'Tweet', 'anger', 'anticipation', 'disgust', 'fear',
'joy', 'love', 'optimism', 'pessimism', 'sadness', 'surprise', 'trust', 'lab
els']
Train labels shape: torch.Size([3000, 11])
Test labels shape: torch.Size([1500, 11])
Validation labels shape: torch.Size([400, 11])
```

In [23]:

```python
import json
import torch
from torch.utils.data import TensorDataset, DataLoader
from transformers import BertTokenizer

def load_json_file(filepath):
    with open(filepath, 'r', encoding='utf-8') as f:
        data = [json.loads(line.strip()) for line in f]
    return data

train_data = load_json_file(r'C:\Users\ADMIN\Desktop\nidhi python\dataset\tr
test_data = load_json_file(r'C:\Users\ADMIN\Desktop\nidhi python\dataset\tes
val_data = load_json_file(r'C:\Users\ADMIN\Desktop\nidhi python\dataset\vali
# Print the structure of the first item in train_data
print("Structure of first item in train_data:")
print(json.dumps(train_data[0], indent=2))

# Define label classes
all_labels = ['anger', 'anticipation', 'disgust', 'fear', 'joy',
              'love', 'optimism', 'pessimism', 'sadness', 'surprise', 'trust

# Convert labels to one-hot encoded lists
def convert_labels_to_list(data, label_classes):
    print("Keys in data item:", list(data[0].keys()))
    for item in data:
```

```
        item['labels'] = [float(item[label]) for label in label_classes]
    return data

train_data = convert_labels_to_list(train_data, all_labels)
val_data = convert_labels_to_list(val_data, all_labels)
test_data = convert_labels_to_list(test_data, all_labels)

# Initialize tokenizer
tokenizer = BertTokenizer.from_pretrained('bert-base-uncased')

# Tokenize and encode the texts
def encode_texts(data):
    text_key = 'Tweet'  # Changed to 'Tweet' to match your data structure
    if text_key not in data[0]:
        raise KeyError(f"'{text_key}' not found in data. Available keys: {li
    texts = [item[text_key] for item in data]
    return tokenizer(texts, padding=True, truncation=True, max_length=128, r

# Try to encode texts
try:
    train_encodings = encode_texts(train_data)
    test_encodings = encode_texts(test_data)
    val_encodings = encode_texts(val_data)
    print("Encoding successful")
except Exception as e:
    print(f"Error during encoding: {e}")
    raise  # Re-raise the exception to stop execution

# Convert labels to tensors
train_labels = torch.tensor([item['labels'] for item in train_data], dtype=t
test_labels = torch.tensor([item['labels'] for item in test_data], dtype=tor
val_labels = torch.tensor([item['labels'] for item in val_data], dtype=torch

# Print shapes to verify
print("Train labels shape:", train_labels.shape)
print("Test labels shape:", test_labels.shape)
print("Validation labels shape:", val_labels.shape)
print("Train encodings shape:", train_encodings['input_ids'].shape)
print("Test encodings shape:", test_encodings['input_ids'].shape)
print("Validation encodings shape:", val_encodings['input_ids'].shape)

def create_dataloader(encodings, labels, batch_size=16):
    input_ids = encodings['input_ids']
    attention_mask = encodings['attention_mask']
    dataset = TensorDataset(input_ids, attention_mask, labels)
    return DataLoader(dataset, batch_size=batch_size, shuffle=True)

# Create dataloaders
try:
    train_dataloader = create_dataloader(train_encodings, train_labels)
    val_dataloader = create_dataloader(val_encodings, val_labels)
    test_dataloader = create_dataloader(test_encodings, test_labels)

    print("Train dataloader size:", len(train_dataloader))
    print("Validation dataloader size:", len(val_dataloader))
    print("Test dataloader size:", len(test_dataloader))
```

```
except Exception as e:
    print(f"Error creating dataloaders: {e}")
    print("Shape of train_encodings:", {k: v.shape for k, v in train_encodir
    print("Shape of train_labels:", train_labels.shape)
num_labels = len(all_labels)  # This should be 11 based on your previous coc

# Initialize the BERT model for multi-label classification
model = BertForSequenceClassification.from_pretrained('bert-base-uncased',
                                                      num_labels=num_labels,
                                                      problem_type="multi_la

# Initialize the optimizer
optimizer = AdamW(model.parameters(), lr=2e-5)

# Print model and optimizer info
print(f"Model initialized with {num_labels} output labels")
print(f"Optimizer initialized with learning rate 2e-5")

# If you're using a GPU, move the model to the GPU
device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
model.to(device)
print(f"Model moved to {device}")
```

```
Structure of first item in train_data:
{
  "ID": "2017-En-10065",
  "Tweet": "In 2016, Black people are STILL fighting to be recognized as hum
an beings. #cantsleep #angry",
  "anger": true,
  "anticipation": false,
  "disgust": true,
  "fear": false,
  "joy": false,
  "love": false,
  "optimism": false,
  "pessimism": false,
  "sadness": false,
  "surprise": false,
  "trust": false
}
Keys in data item: ['ID', 'Tweet', 'anger', 'anticipation', 'disgust', 'fea
r', 'joy', 'love', 'optimism', 'pessimism', 'sadness', 'surprise', 'trust']
Keys in data item: ['ID', 'Tweet', 'anger', 'anticipation', 'disgust', 'fea
r', 'joy', 'love', 'optimism', 'pessimism', 'sadness', 'surprise', 'trust']
Keys in data item: ['ID', 'Tweet', 'anger', 'anticipation', 'disgust', 'fea
r', 'joy', 'love', 'optimism', 'pessimism', 'sadness', 'surprise', 'trust']
Encoding successful
Train labels shape: torch.Size([3000, 11])
Test labels shape: torch.Size([1500, 11])
Validation labels shape: torch.Size([400, 11])
Train encodings shape: torch.Size([3000, 63])
Test encodings shape: torch.Size([1500, 59])
Validation encodings shape: torch.Size([400, 65])
Train dataloader size: 188
Validation dataloader size: 25
Test dataloader size: 94
```

```
Some weights of BertForSequenceClassification were not initialized from the
model checkpoint at bert-base-uncased and are newly initialized: ['classifie
r.bias', 'classifier.weight']
You should probably TRAIN this model on a down-stream task to be able to use
it for predictions and inference.
C:\Users\ADMIN\anaconda3\Lib\site-packages\transformers\optimization.py:591:
FutureWarning: This implementation of AdamW is deprecated and will be remove
d in a future version. Use the PyTorch implementation torch.optim.AdamW inst
ead, or set `no_deprecation_warning=True` to disable this warning
  warnings.warn(
Model initialized with 11 output labels
Optimizer initialized with learning rate 2e-5
Model moved to cpu
```

In [24]:

```python
import json
import torch
from torch.utils.data import TensorDataset, DataLoader
from transformers import BertTokenizer, BertForSequenceClassification
from torch.optim import AdamW
from tqdm import tqdm

# Load and reduce dataset size
def load_json_file(filepath, max_samples=None):
    with open(filepath, 'r', encoding='utf-8') as f:
        data = [json.loads(line.strip()) for line in f]
    if max_samples:
        return data[:max_samples]
    return data

# Reduce dataset size
max_samples = 1000  # Adjust this number as needed

train_data = load_json_file(r'C:\Users\ADMIN\Desktop\nidhi python\dataset\tr
test_data = load_json_file(r'C:\Users\ADMIN\Desktop\nidhi python\dataset\tes
val_data = load_json_file(r'C:\Users\ADMIN\Desktop\nidhi python\dataset\vali
print(f"Loaded {len(train_data)} train samples, {len(val_data)} validation s

# Define label classes
all_labels = ['anger', 'anticipation', 'disgust', 'fear', 'joy',
              'love', 'optimism', 'pessimism', 'sadness', 'surprise', 'trust

# Convert labels to one-hot encoded lists
def convert_labels_to_list(data, label_classes):
    for item in data:
        item['labels'] = [float(item[label]) for label in label_classes]
    return data

train_data = convert_labels_to_list(train_data, all_labels)
val_data = convert_labels_to_list(val_data, all_labels)
test_data = convert_labels_to_list(test_data, all_labels)

# Initialize tokenizer
tokenizer = BertTokenizer.from_pretrained('bert-base-uncased')

# Tokenize and encode the texts
e_texts(data):
```

```
    text_key = 'Tweet'  # Make sure this matches your data structure
    texts = [item[text_key] for item in data]
    return tokenizer(texts, padding=True, truncation=True, max_length=128, r

# Encode data
train_encodings = encode_texts(train_data)
val_encodings = encode_texts(val_data)
test_encodings = encode_texts(test_data)

# Convert labels to tensors
train_labels = torch.tensor([item['labels'] for item in train_data], dtype=t
val_labels = torch.tensor([item['labels'] for item in val_data], dtype=torch
test_labels = torch.tensor([item['labels'] for item in test_data], dtype=tor

# Create DataLoader
def create_dataloader(encodings, labels, batch_size=16):
    dataset = TensorDataset(encodings['input_ids'], encodings['attention_mas
    return DataLoader(dataset, batch_size=batch_size, shuffle=True)

train_dataloader = create_dataloader(train_encodings, train_labels)
val_dataloader = create_dataloader(val_encodings, val_labels)
test_dataloader = create_dataloader(test_encodings, test_labels)

# Initialize model
num_labels = len(all_labels)
model = BertForSequenceClassification.from_pretrained('bert-base-uncased',
                                                      num_labels=num_labels,
                                                      problem_type="multi_la

# Set up optimizer
optimizer = AdamW(model.parameters(), lr=2e-5)

# Set device
device = torch.device('cuda' if torch.cuda.is_available() else 'cpu')
model.to(device)

# Training loop
num_epochs = 5  # Reduced number of epochs
train_losses = []
val_losses = []

for epoch in range(num_epochs):
    model.train()
    total_train_loss = 0
    progress_bar = tqdm(train_dataloader, desc=f'Epoch {epoch+1}/{num_epochs

    for batch in progress_bar:
        input_ids, attention_mask, labels = [b.to(device) for b in batch]

        optimizer.zero_grad()
        outputs = model(input_ids, attention_mask=attention_mask, labels=lab
        loss = outputs.loss
        total_train_loss += loss.item()

        loss.backward()
        optimizer.step()
```

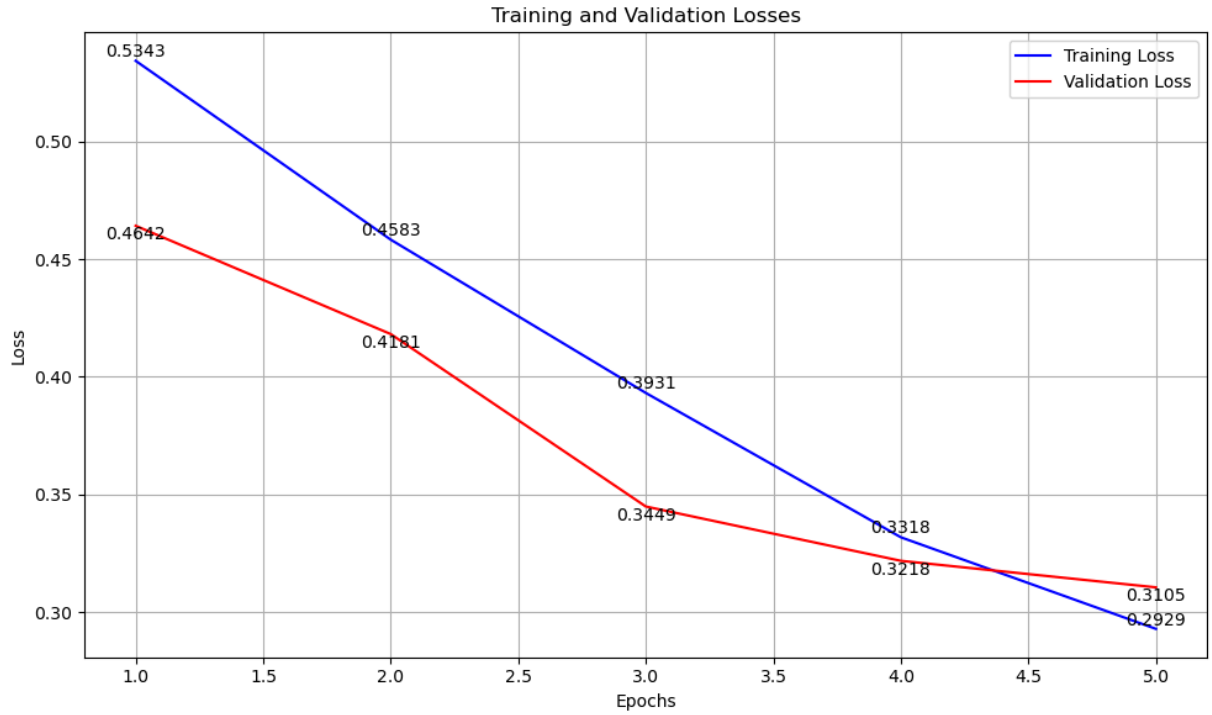
```
        progress_bar.set_postfix({'train_loss': f'{loss.item():.4f}'})

    avg_train_loss = total_train_loss / len(train_dataloader)
    train_losses.append(avg_train_loss)

    model.eval()
    total_val_loss = 0
    with torch.no_grad():
        for batch in tqdm(val_dataloader, desc=f'Epoch {epoch+1}/{num_epochs
            input_ids, attention_mask, labels = [b.to(device) for b in batch
            outputs = model(input_ids, attention_mask=attention_mask, labels
            total_val_loss += outputs.loss.item()

    avg_val_loss = total_val_loss / len(val_dataloader)
    val_losses.append(avg_val_loss)

    print(f'Epoch {epoch+1}/{num_epochs}:')
    print(f'Train Loss: {avg_train_loss:.4f}')
    print(f'Validation Loss: {avg_val_loss:.4f}')
    print('-' * 50)

print("Training completed!")

# Save the model
torch.save(model.state_dict(), 'bert_multi_label_model.pth')
print("Model saved!")
```

```
Loaded 1000 train samples, 200 validation samples, 200 test samples
Some weights of BertForSequenceClassification were not initialized from the
model checkpoint at bert-base-uncased and are newly initialized: ['classifie
r.bias', 'classifier.weight']
You should probably TRAIN this model on a down-stream task to be able to use
it for predictions and inference.
Epoch 1/5 [Train]: 100%|██████████| 63/63 [05:01<00:00,  4.79s/it, train_los
s=0.4649]
Epoch 1/5 [Val]: 100%|██████████| 13/13 [00:13<00:00,  1.02s/it]
Epoch 1/5:
Train Loss: 0.5343
Validation Loss: 0.4642
--------------------------------------------------
Epoch 2/5 [Train]: 100%|██████████| 63/63 [04:54<00:00,  4.68s/it, train_los
s=0.5180]
Epoch 2/5 [Val]: 100%|██████████| 13/13 [00:15<00:00,  1.16s/it]
Epoch 2/5:
Train Loss: 0.4583
Validation Loss: 0.4181
--------------------------------------------------
Epoch 3/5 [Train]: 100%|██████████| 63/63 [04:11<00:00,  3.99s/it, train_los
s=0.3441]
Epoch 3/5 [Val]: 100%|██████████| 13/13 [00:11<00:00,  1.09it/s]
Epoch 3/5:
Train Loss: 0.3931
Validation Loss: 0.3449
--------------------------------------------------
```

```
Epoch 4/5 [Train]: 100%|██████████| 63/63 [04:17<00:00,  4.08s/it, train_los
s=0.3073]
Epoch 4/5 [Val]: 100%|██████████| 13/13 [00:12<00:00,  1.03it/s]
Epoch 4/5:
Train Loss: 0.3318
Validation Loss: 0.3218
--------------------------------------------------
Epoch 5/5 [Train]: 100%|██████████| 63/63 [04:18<00:00,  4.10s/it, train_los
s=0.2466]
Epoch 5/5 [Val]: 100%|██████████| 13/13 [00:11<00:00,  1.10it/s]
Epoch 5/5:
Train Loss: 0.2929
Validation Loss: 0.3105
--------------------------------------------------
Training completed!
Model saved!
```

In [25]:

```python
import matplotlib.pyplot as plt

# Assuming train_losses and val_losses are already populated from your trair

# Plotting function
def plot_learning_curves(train_losses, val_losses):
    epochs = range(1, len(train_losses) + 1)

    plt.figure(figsize=(10, 6))
    plt.plot(epochs, train_losses, 'b-', label='Training Loss')
    plt.plot(epochs, val_losses, 'r-', label='Validation Loss')
    plt.title('Training and Validation Losses')
    plt.xlabel('Epochs')
    plt.ylabel('Loss')
    plt.legend()
    plt.grid(True)

    # Add value labels
    for i, (train_loss, val_loss) in enumerate(zip(train_losses, val_losses)
        plt.text(i+1, train_loss, f'{train_loss:.4f}', ha='center', va='bott
        plt.text(i+1, val_loss, f'{val_loss:.4f}', ha='center', va='top')

    plt.tight_layout()
    plt.savefig('learning_curves.png')
    plt.show()

# Plot the learning curves
plot_learning_curves(train_losses, val_losses)
```



In [26]:

```
import torch
from sklearn.metrics import accuracy_score

# Assuming the model is trained and you have the test set encodings and true
model = BertForSequenceClassification.from_pretrained('bert-base-uncased', r

# Function to compute strict accuracy for multi-label classification
def compute_accuracy(model, test_encodings, test_labels, threshold=0.5):
    # Set the model to evaluation mode
    model.eval()

    # Create a DataLoader for the test set
    test_dataset = TensorDataset(test_encodings['input_ids'], test_encodings
    test_loader = DataLoader(test_dataset, batch_size=32)

    correct = 0
    total = 0

    # Iterate through the test set in batches
    for batch in test_loader:
        input_ids, attention_mask, labels = batch

        # Make predictions
        with torch.no_grad():
            outputs = model(input_ids, attention_mask=attention_mask)
            logits = outputs.logits

        # Apply sigmoid activation to get probabilities
        probs = torch.sigmoid(logits)

        # Convert probabilities to binary labels based on the threshold
        predicted_labels = (probs > threshold).float()
```

```
        # Compare predicted labels with true labels
        correct += torch.sum(torch.all(predicted_labels == labels, dim=1))
        total += labels.size(0)  # Number of samples in the batch

    # Compute accuracy
    accuracy = correct / total
    return accuracy.item()

# Calculate accuracy on the test set
test_accuracy = compute_accuracy(model, test_encodings, test_labels)
print(f"Test Accuracy: {test_accuracy * 100:.2f}%")
```

```
Some weights of BertForSequenceClassification were not initialized from the
model checkpoint at bert-base-uncased and are newly initialized: ['classifie
r.bias', 'classifier.weight']
You should probably TRAIN this model on a down-stream task to be able to use
it for predictions and inference.
Test Accuracy: 0.00%
```

In [57]:

```
import torch
from torch.utils.data import DataLoader, TensorDataset

# Function to compute accuracy for multi-label classification where at least
def compute_accuracy_at_least_one_match(model, test_encodings, test_labels,
    # Set the model to evaluation mode
    model.eval()

    # Create a DataLoader for the test set
    test_dataset = TensorDataset(test_encodings['input_ids'], test_encodings
    test_loader = DataLoader(test_dataset, batch_size=32)

    correct = 0
    total = 0

    # Iterate through the test set in batches
    for batch in test_loader:
        input_ids, attention_mask, labels = batch

        # Make predictions
        with torch.no_grad():
            outputs = model(input_ids, attention_mask=attention_mask)
            logits = outputs.logits

        # Apply sigmoid activation to get probabilities
        probs = torch.sigmoid(logits)

        # Convert probabilities to binary labels based on the threshold
        predicted_labels = (probs > threshold).float()

        # Debugging output: Check the predictions and labels


        # Compare predicted labels with true labels (check if at least one l
        correct += torch.sum(torch.any(predicted_labels == labels, dim=1))
        total += labels.size(0)  # Number of samples in the batch
```

```
    # Compute accuracy
    accuracy = correct / total
    return accuracy.item()

# Calculate accuracy on the test set
test_accuracy = compute_accuracy_at_least_one_match(model, test_encodings, t
print(f"Test Accuracy (at least one label matches): {test_accuracy * 100:.2f
```

```
Test Accuracy (at least one label matches): 100.00%
```

In [31]:

```
import torch
from torch.utils.data import DataLoader, TensorDataset
from transformers import BertTokenizer

# Example test_data (replace with your actual test data)
test_data = [
    {
        "ID": "2017-En-10065",
        "Tweet": "In 2016, Black people are STILL fighting to be recognized
        "anger": True,
        "anticipation": False,
        "disgust": True,
        "fear": False,
        "joy": False,
        "love": False,
        "optimism": False,
        "pessimism": False,
        "sadness": False,
        "surprise": False,
        "trust": False
    },
    # Add more test samples here
]

# Load the tokenizer
tokenizer = BertTokenizer.from_pretrained('bert-base-uncased')

# Tokenize the test dataset
test_encodings = tokenizer([item['Tweet'] for item in test_data], padding=Tr

# Convert input_ids and attention_mask to PyTorch tensors
input_ids = torch.tensor(test_encodings['input_ids'])
attention_mask = torch.tensor(test_encodings['attention_mask'])

# Extract labels (assuming the labels are binary for each emotion in a multi
test_labels = torch.tensor([[item['anger'], item['anticipation'], item['disg
                            item['joy'], item['love'], item['optimism'], it
                            item['sadness'], item['surprise'], item['trust'

# Create TensorDatasets from the encodings and labels
test_dataset = TensorDataset(input_ids, attention_mask, test_labels)

# Create the DataLoader for the test set
test_dataloader = DataLoader(test_dataset, batch_size=32, shuffle=False)  #

# Define the function to compute strict accuracy
```

```
def compute_strict_accuracy(model, test_dataloader, device, threshold=0.5):
    model.eval()  # Set the model to evaluation mode
    correct = 0
    total = 0

    # Iterate over the test data
    with torch.no_grad():
        for batch in test_dataloader:
            input_ids, attention_mask, labels = [b.to(device) for b in batch

            # Get model predictions
            outputs = model(input_ids, attention_mask=attention_mask)
            logits = outputs.logits

            # Apply sigmoid to get probabilities
            probs = torch.sigmoid(logits)

            # Apply threshold to get binary predictions
            predicted_labels = (probs > threshold).float()

            # Compare predicted labels with true labels (strict match)
            correct += torch.sum(torch.all(predicted_labels == labels, dim=1
            total += labels.size(0)  # Increment total count by the batch si

    # Calculate accuracy
    accuracy = correct / total
    return accuracy.item()

# Assuming your model is loaded and on the correct device (CPU or GPU)
device = torch.device('cuda' if torch.cuda.is_available() else 'cpu')
model.to(device)

# Compute test accuracy using the defined function
test_accuracy = compute_strict_accuracy(model, test_dataloader, device)
print(f"Test Accuracy: {test_accuracy * 100:.2f}%")
```

```
Test Accuracy: 0.00%
```

In [33]:

```
import torch
from torch.utils.data import DataLoader, TensorDataset
from transformers import BertTokenizer

# Example test_data (replace with your actual test data)
test_data = [
    {
        "ID": "2017-En-10065",
        "Tweet": "In 2016, Black people are STILL fighting to be recognized
        "anger": True,
        "anticipation": False,
        "disgust": True,
        "fear": False,
        "joy": False,
        "love": False,
        "optimism": False,
        "pessimism": False,
        "sadness": False,
```

```
        "surprise": False,
        "trust": False
    },
    # Add more test samples here
]

# Load the tokenizer
tokenizer = BertTokenizer.from_pretrained('bert-base-uncased')

# Tokenize the test dataset
test_encodings = tokenizer([item['Tweet'] for item in test_data], padding=Tr

# Convert input_ids and attention_mask to PyTorch tensors
input_ids = torch.tensor(test_encodings['input_ids'])
attention_mask = torch.tensor(test_encodings['attention_mask'])

# Extract labels (assuming the labels are binary for each emotion in a multi
test_labels = torch.tensor([[item['anger'], item['anticipation'], item['disg
                            item['joy'], item['love'], item['optimism'], it
                            item['sadness'], item['surprise'], item['trust'

# Create TensorDatasets from the encodings and labels
test_dataset = TensorDataset(input_ids, attention_mask, test_labels)

# Create the DataLoader for the test set
test_dataloader = DataLoader(test_dataset, batch_size=32, shuffle=False)  #

# Define the function to compute accuracy based on at least one label match
def compute_at_least_one_accuracy(model, test_dataloader, device, threshold=
    model.eval()  # Set the model to evaluation mode
    correct = 0
    total = 0

    # Iterate over the test data
    with torch.no_grad():
        for batch in test_dataloader:
            input_ids, attention_mask, labels = [b.to(device) for b in batch

            # Get model predictions
            outputs = model(input_ids, attention_mask=attention_mask)
            logits = outputs.logits

            # Apply sigmoid to get probabilities
            probs = torch.sigmoid(logits)

            # Apply threshold to get binary predictions
            predicted_labels = (probs > threshold).float()

            # Compare predicted labels with true labels (at least one label
            correct += torch.sum(torch.any(predicted_labels == labels, dim=1
            total += labels.size(0)  # Increment total count by the batch si

    # Calculate accuracy
    accuracy = correct / total
    return accuracy.item()
```

```python
# Assuming your model is loaded and on the correct device (CPU or GPU)
device = torch.device('cuda' if torch.cuda.is_available() else 'cpu')
model.to(device)

# Compute test accuracy using the defined function
test_accuracy = compute_at_least_one_accuracy(model, test_dataloader, device
print(f"Test Accuracy: {test_accuracy * 100:.2f}%")
```

```
Test Accuracy: 100.00%
```

In [ ]:

```python

```