🎓 COMPLETE TRAINING GUIDE

How to Train Models, Dataset Requirements, and Supported Formats

📋 TABLE OF CONTENTS

⚡ QUICK START TRAINING

Option 1: Train with Existing Sample (FASTEST - 1 minute)

```
# Uses included sample dataset (6 samples - for testing only)
python train_depression_classifier.py --model distilbert-base-uncased --epochs
1 --data-path data/dreaddit_sample.csv
```

Option 2: Train with Full Dataset (RECOMMENDED - 15-30 minutes)

```
# Uses merged_real_dataset.csv (22,357 samples)
python train_depression_classifier.py --model roberta-base --epochs 3 --data-
path data/merged_real_dataset.csv
```

Option 3: Train with Your Own Data

```
# Use your custom CSV file
python train_depression_classifier.py --model roberta-base --epochs 3 --data-
path data/YOUR_DATA.csv
```

📊 DATASET FORMAT REQUIREMENTS

☑ REQUIRED FORMAT: CSV with 3 columns

Your CSV file **MUST** have these exact columns:

Column	Type	Values	Description
`text`	string	Any text	The social media post or text to analyze
`label`	string/int	`depressed` or `control` OR `0` or `1`	Classification label
`source`	string	Any identifier	Dataset source (e.g., "dreaddit", "reddit")

☑ EXAMPLE CSV FORMAT:

```
text,label,source
I feel empty and tired of life. can't sleep anymore.,depressed,reddit
Today was fine. Went for a run and met friends.,control,reddit
I hate myself and I am a failure at everything.,depressed,twitter
No appetite these days and I can't focus on work.,depressed,dreaddit
I am so restless and pacing all night nothing helps.,depressed,reddit
```

☑ LABEL FORMATS ACCEPTED:

Option 1: String labels

- `depressed` → Depression class
- `control` → Control/Normal class

Option 2: Numeric labels

- `0` → Control/Normal
- `1` → Depression

Option 3: Alternative strings (automatically converted)

- `depression` → Converted to `depressed`
- `normal` → Converted to `control`
- `healthy` → Converted to `control`

❌ WHAT'S NOT SUPPORTED:

- ❌ Excel files (.xlsx, .xls) - Must convert to CSV first
- ❌ JSON files - Must convert to CSV first
- ❌ Text files with one sample per line - Must convert to CSV
- ❌ Missing required columns (`text`, `label`, `source`)
- ❌ Empty or null values in required columns
- ❌ More than 2 classes (this is binary classification)

📏 DATASET SIZE GUIDELINES

Minimum Sizes (By Purpose)

Purpose	Min Samples	Recommended	Expected F1 Score
Quick Test	10+	50+	50-60% (overfits)
Demo/Prototype	100+	500+	60-70%
Research/Thesis	1,000+	3,000-5,000	75-85%
Production	5,000+	10,000-50,000	85-92%
Publication	10,000+	20,000+	90%+

Current Available Datasets

Dataset	Size	Location	Quality
dreaddit_sample.csv	6	`data/`	☑ Testing only
merged_real_dataset.csv	22,357	`data/`	☑ **BEST - Use this!**
Dreaddit (full)	3,553	Download required	☑ Research-grade
RSDD	9,000+	Requires access	☑ Production
CLPsych	5,000+	Request access	☑ Research

⚠ IMPORTANT: Dataset Balance

ALWAYS check class balance:

```
python -c "import pandas as pd; df = pd.read_csv('data/YOUR_DATA.csv');
print(df['label'].value_counts())"
```

Good balance:

```
depressed    11,000  (50%)
control      11,000  (50%)
```

Bad balance (will cause issues):

```
depressed    20,000  (95%)  ❌ Too imbalanced!
control       1,000   (5%)
```

Solution for imbalanced data: Use class weights or undersample majority class.

🚀 TRAINING COMMANDS

1. **DistilBERT** (Fastest - 4GB GPU)

Best for: Quick experiments, limited GPU memory

```
python train_depression_classifier.py `
  --model distilbert-base-uncased `
  --data-path data/merged_real_dataset.csv `
  --epochs 3 `
  --batch-size 16 `
  --lr 2e-5 `
  --max-length 256
```

Training time: ~10-20 minutes
Expected F1: 79-84%
GPU Memory: 4GB

---

2. **BERT-Base** (Baseline - 6-8GB GPU)

Best for: Standard experiments, balanced accuracy vs speed

```
python train_depression_classifier.py `
  --model bert-base-uncased `
  --data-path data/merged_real_dataset.csv `
  --epochs 3 `
  --batch-size 16 `
  --lr 2e-5 `
  --max-length 256
```

Training time: ~15-25 minutes
Expected F1: 81-87%
GPU Memory: 6-8GB

---

3. **RoBERTa-Base** (Best Accuracy - 8-10GB GPU)

Best for: Production, highest accuracy

```
python train_depression_classifier.py `
  --model roberta-base `
  --data-path data/merged_real_dataset.csv `
  --epochs 3 `
  --batch-size 16 `
  --lr 2e-5 `
  --max-length 256
```

Training time: ~20-30 minutes
Expected F1: 84-89%
GPU Memory: 8-10GB

---

4. **Small Dataset Training** (100-1,000 samples)

```
python train_depression_classifier.py `
  --model distilbert-base-uncased `
  --data-path data/YOUR_SMALL_DATA.csv `
  --epochs 5 `
  --batch-size 8 `
  --lr 3e-5 `
  --max-length 128
```

Tips for small datasets:

- Use more epochs (5-10)
- Smaller batch size (8)
- Higher learning rate (3e-5)
- Shorter sequences (128 tokens)

---

5. **Large Dataset Training** (10,000+ samples)

```
python train_depression_classifier.py `
  --model roberta-base `
  --data-path data/YOUR_LARGE_DATA.csv `
  --epochs 2 `
  --batch-size 32 `
  --lr 2e-5 `
  --max-length 256
```

Tips for large datasets:

- Fewer epochs needed (2-3)
- Larger batch size (32-64)
- Standard learning rate (2e-5)

---

6. **CPU Training** (No GPU available)

```
python train_depression_classifier.py `
  --model distilbert-base-uncased `
  --data-path data/merged_real_dataset.csv `
  --epochs 1 `
```

```
    --batch-size 4 `
    --no-cuda
```

⚠ **Warning:** CPU training is 10-50x slower!
Recommended: Use Google Colab (free GPU) or reduce dataset size

---

ADVANCED CONFIGURATION

Full Command with All Options

```
python train_depression_classifier.py `
  --model roberta-base `
  --data-path data/merged_real_dataset.csv `
  --test-size 0.15 `
  --epochs 3 `
  --batch-size 16 `
  --lr 2e-5 `
  --weight-decay 0.01 `
  --max-length 256 `
  --output-dir models/trained `
  --run-name my_experiment `
  --seed 42
```

Parameter Explanations

Parameter	Default	Description	When to Change
`--model`	roberta-base	Model architecture	DistilBERT (faster), BERT (baseline), RoBERTa (best)
`--data-path`	data/dreaddit-train.csv	Training data CSV	Use your own dataset
`--test-size`	0.2	Test set percentage	0.15 for large datasets, 0.25 for small
`--epochs`	3	Training epochs	5-10 for small data, 2-3 for large
`--batch-size`	16	Samples per batch	8 (small GPU), 32 (large GPU)
`--lr`	2e-5	Learning rate	3e-5 (small data), 1e-5 (fine-tuning)
`--weight-decay`	0.01	L2 regularization	0.1 (prevent overfitting)
`--max-length`	256	Max tokens	128 (short texts), 512 (long texts)

Parameter	Default	Description	When to Change
`--output-dir`	models/trained	Save location	Custom path for organization
`--run-name`	auto	Experiment name	Use for tracking different runs
`--seed`	42	Random seed	Change for different random splits
`--no-cuda`	False	Disable GPU	Use for CPU-only training

🔍 TRAINING OUTPUT EXPLAINED

What Happens During Training

```
2025-11-26 22:00:00 - INFO - Loading data from data/merged_real_dataset.csv
2025-11-26 22:00:01 - INFO - Loaded 22357 samples
2025-11-26 22:00:01 - INFO - Label distribution:
2025-11-26 22:00:01 - INFO -   depressed: 11179 (50.0%)
2025-11-26 22:00:01 - INFO -   control: 11178 (50.0%)
2025-11-26 22:00:02 - INFO - Train/Test split: 17885/4472
2025-11-26 22:00:10 - INFO - Loaded model: roberta-base
2025-11-26 22:00:10 - INFO - Starting training...

Epoch 1/3:
  Step 100: loss=0.45, accuracy=0.78
  Step 200: loss=0.32, accuracy=0.85
  ...
  Validation: loss=0.28, accuracy=0.87, f1=0.86

Epoch 2/3:
  Step 100: loss=0.25, accuracy=0.89
  ...

2025-11-26 22:25:00 - INFO - Training complete!
2025-11-26 22:25:01 - INFO - Test Results:
  Accuracy: 0.8745
  Precision: 0.8912
  Recall: 0.8567
  F1 Score: 0.8736

2025-11-26 22:25:02 - INFO - Model saved to:
models/trained/roberta_20251126_220000
```

Output Files

After training, you'll find:

```
models/trained/roberta_20251126_220000/
├── pytorch_model.bin          # Model weights (499 MB)
├── model.safetensors          # Alternative format (499 MB)
├── config.json                # Model configuration
├── tokenizer_config.json      # Tokenizer settings
├── vocab.json                 # Vocabulary
├── merges.txt                 # BPE merges
├── special_tokens_map.json    # Special tokens
└── training_report.json       # Training metrics
```

🆘 TROUBLESHOOTING

Problem 1: "Dataset not found"

```
ERROR - Dataset not found: data/dreaddit-train.csv
```

Solution:

```
# Check what datasets you have
ls data/*.csv

# Use the correct path
python train_depression_classifier.py --data-path data/merged_real_dataset.csv
```

Problem 2: "Out of memory"

```
RuntimeError: CUDA out of memory
```

Solution 1: Reduce batch size

```
python train_depression_classifier.py --batch-size 8  # Instead of 16
```

Solution 2: Use smaller model

```
python train_depression_classifier.py --model distilbert-base-uncased
```

Solution 3: Reduce sequence length

```
python train_depression_classifier.py --max-length 128  # Instead of 256
```

Solution 4: Use CPU (slow)

```
python train_depression_classifier.py --no-cuda
```

Problem 3: "KeyError: 'text'" or "KeyError: 'label'"

```
KeyError: 'text'
```

Solution: Your CSV doesn't have the required columns

Fix your CSV:

```
import pandas as pd

# Load your data
df = pd.read_csv('your_data.csv')

# Rename columns to match requirements
df = df.rename(columns={
    'post': 'text',        # Your text column
    'category': 'label',   # Your label column
    'dataset': 'source'    # Your source column
})

# Save
df.to_csv('data/fixed_data.csv', index=False)
```

Problem 4: "Too few samples"

```
WARNING - Only 50 samples found. Minimum 100 recommended.
```

Solution:

Option 1: Download more data

```
# Use the included 22K dataset
python train_depression_classifier.py --data-path data/merged_real_dataset.csv
```

Option 2: Create synthetic data (for testing only)

```
import pandas as pd

# Create mock data
data = []
for i in range(500):
    if i % 2 == 0:
        data.append({
            'text': f"I feel depressed and hopeless sample {i}",
            'label': 'depressed',
            'source': 'synthetic'
        })
    else:
        data.append({
            'text': f"I am happy and excited sample {i}",
            'label': 'control',
            'source': 'synthetic'
        })

df = pd.DataFrame(data)
df.to_csv('data/synthetic_500.csv', index=False)
```

Problem 5: "Imbalanced classes"

```
WARNING - Class imbalance detected: 90% depressed, 10% control
```

Solution: Use class weights (automatic in our script)

The script automatically handles imbalanced data using class weights.

Problem 6: "Model not learning" (accuracy stuck at 50%)

```
Epoch 1: accuracy=0.51
Epoch 2: accuracy=0.52
Epoch 3: accuracy=0.51
```

Possible causes:

1. ❌ Too high learning rate
2. ❌ Too small dataset
3. ❌ Corrupted labels

4. ❌ All samples are identical

Solution:

```
# Lower learning rate
python train_depression_classifier.py --lr 1e-5

# More epochs
python train_depression_classifier.py --epochs 10

# Check your data
python -c "import pandas as pd; df = pd.read_csv('data/YOUR_DATA.csv');
print(df.head(20))"
```

📚 DATASET CREATION GUIDE

How to Create Your Own Dataset

Step 1: Collect Text Data

Option 1: From Reddit (using PRAW)

```
import praw
import pandas as pd

reddit = praw.Reddit(
    client_id='YOUR_CLIENT_ID',
    client_secret='YOUR_SECRET',
    user_agent='depression_research'
)

# Collect from depression subreddit
depression_posts = []
for post in reddit.subreddit('depression').hot(limit=500):
    depression_posts.append({
        'text': post.title + ' ' + post.selftext,
        'label': 'depressed',
        'source': 'reddit_depression'
    })

# Collect from happy subreddit
control_posts = []
for post in reddit.subreddit('happy').hot(limit=500):
    control_posts.append({
        'text': post.title + ' ' + post.selftext,
        'label': 'control',
        'source': 'reddit_happy'
    })
```

```
# Combine and save
all_posts = depression_posts + control_posts
df = pd.DataFrame(all_posts)
df.to_csv('data/reddit_dataset.csv', index=False)
```

Step 2: Clean and Validate

```
import pandas as pd

df = pd.read_csv('data/reddit_dataset.csv')

# Remove empty texts
df = df[df['text'].str.len() > 10]

# Remove duplicates
df = df.drop_duplicates(subset=['text'])

# Remove URLs
df['text'] = df['text'].str.replace(r'http\S+', '', regex=True)

# Validate labels
assert df['label'].isin(['depressed', 'control']).all()

# Check balance
print(df['label'].value_counts())

# Save cleaned data
df.to_csv('data/reddit_dataset_cleaned.csv', index=False)
```

Step 3: Train

```
python train_depression_classifier.py --data-path
data/reddit_dataset_cleaned.csv
```

☑ QUICK CHECKLIST

Before training, verify:

- ☐ CSV file exists in `data/` folder
- ☐ CSV has columns: `text`, `label`, `source`
- ☐ Labels are either `depressed`/`control` or `0`/`1`
- ☐ At least 100+ samples (1,000+ recommended)
- ☐ Classes are balanced (40-60% ratio)

- [ ] No empty or null values
- [ ] Text length is reasonable (10-500 words)
- [ ] Dependencies installed: `pip install transformers torch datasets scikit-learn`
- [ ] GPU available (or use `--no-cuda` flag)
- [ ] Enough disk space (2-5 GB for model)

🎯 RECOMMENDED WORKFLOW

For Research/Thesis:

```
# 1. Use merged_real_dataset.csv (22K samples)
python train_depression_classifier.py --model roberta-base --data-path
data/merged_real_dataset.csv

# 2. Test on multiple models
python train_depression_classifier.py --model bert-base-uncased --data-path
data/merged_real_dataset.csv
python train_depression_classifier.py --model distilbert-base-uncased --data-
path data/merged_real_dataset.csv

# 3. Compare results
python compare_models.py --models models/trained/* --test-data
data/merged_real_dataset.csv

# 4. Run inference
python predict_depression.py --model models/trained/roberta_* --text "I feel
hopeless"

# 5. Launch web app for demo
streamlit run src/app/app.py
```

Need help? Check:

- `README.md` - Project overview
- `GET_STARTED.md` - Quick start
- `TRAINING_GUIDE.md` - Original guide
- `MODEL_COMPARISON_GUIDE.md` - Model selection

Have questions? Open the project in VS Code and ask GitHub Copilot!