Deep Learning Challenge: Text Emotion Classification

Problem Statement

Understanding human emotions from text is a fundamental challenge in Natural Language Processing. In this project, your task is to build a deep learning model that classifies short text messages into one of six emotion categories.

Your goal is to: - Build an LSTM/GRU-based neural network for multi-class text classification - Demonstrate understanding of word embeddings, sequence modeling, and training loops - Achieve competitive accuracy while maintaining clean, modular code

Dataset Information

Dataset Specifications

- **Location:** `dataset/`
- **Files:** `train.txt`, `val.txt`, `test.txt`
- **Total Samples:** 20,000 English text messages (Twitter-sourced)
- **Format:** Text file with `message; label` per line

Dataset Description

The dataset comprises 20,000 English text messages, each labeled with one of six emotion categories.

Emotion Classes

Label	Emotion	Example
0	Sadness	"I feel so alone and empty inside"
1	Joy	"This is the best day of my life!"
2	Love	"I adore spending time with you"
3	Anger	"I can't believe they did this to me!"
4	Fear	"I'm terrified about the results"

Label	Emotion	Example
5	Surprise	“Wow I never expected this to happen!”

Data Format

Each file contains one sample per line with format: `text;label`

```
i didnt feel humiliated;sadness
this is the best day of my life;joy
i am feeling grouchy;anger
```

Data Split

- Training set: 16,000 samples (`train.txt`)
- Validation set: 2,000 samples (`val.txt`)
- Test set: 2,000 samples (`test.txt`)

Tasks to Complete

1. Data Preprocessing

- Load and explore the dataset
- Clean text (remove special characters, lowercase)
- Tokenize text and build vocabulary
- Convert text to sequences with padding
- Create PyTorch/TensorFlow data loaders

2. Training Pipeline

Implement a complete training loop

3. Evaluation

Evaluate your model using: - Test set accuracy - Per-class F1 scores - Confusion matrix visualization

Technical Requirements

Allowed Frameworks

- PyTorch (recommended)
- TensorFlow/Keras
- Custom implementations welcome
- Pre-trained transformers (BERT, RoBERTa, etc.)

Deliverables

GitHub repository containing: - `model/emotion_classifier.py` - Model architecture - `train.py` or `notebook.ipynb` - Training code - `README.md` - Setup instructions and approach - `requirements.txt` - Dependencies

Example Repository Structure

```
.
├── README.md
├── requirements.txt
├── data/
│   ├── train.txt
│   ├── val.txt
│   └── test.txt
├── model/
│   ├── emotion_classifier.py
│   └── data_loader.py
├── utils/
│   ├── preprocessing.py
│   └── metrics.py
├── train.py
└── evaluate.py
```

Time Limit

Suggested time to complete: Maximum 2 hours

Focus on building a functional model with clean code. Perfect accuracy is not expected.

Evaluation Criteria

Criteria	Weightage
Model Accuracy (test set > 85%)	30%
Code Quality and Modularity	25%
Architecture Design and Justification	20%
Documentation and Comments	15%
(Bonus) Advanced Techniques	10%

Baseline Performance

- Random baseline: ~17% (1/6 classes)
- Simple model: ~80-85%
- Good submission: ~88-92%
- Excellent submission: >92%