WNUT-2020 Task 2 Text Classification

This project implements text classification for the WNUT-2020 Task 2 dataset (binary classification: INFORMATIVE vs UNINFORMATIVE) using ALBERT models with data augmentation and self-supervised learning techniques.

Features

- Base model: ALBERT-base-v2
- Synonym augmentation using WordNet
- Multiple self-supervised learning strategies
- Weighted loss function for class imbalance
- Confidence-based pseudo-labeling

Environment Setup

This project is designed to run in the NCSA OpenCE 1.5.1 environment:

```bash
# Activate the environment
conda activate opence-v1.5.1

# Install additional requirements
pip install transformers nlpaug
```

Data Preparation

1. Create the dataset directory:

```bash
mkdir WNUT-2020-Task-2-Dataset
```

2. Download and place the WNUT-2020 Task 2 dataset files:
   - `train.tsv` (6,936 samples)
   - `valid.tsv` (999 samples)
   - `test.tsv` (2,000 samples)
   - `unlabeled_test_with_noise.tsv` (for self-supervised learning)

Running the Code

Approach 1: Basic Data Augmentation Training

Train the model using only labeled data with synonym augmentation.

Steps:

1. Run **Cell 1** or **Cell 6**
2. Model saves to `./results/`

python

```
# Cell 1: Complete augmented training with weighted loss
# OR
# Cell 6: Simplified augmented training
```

Approach 2: Data Augmentation + High-Confidence Pseudo-labels

Train with augmented data, then add high-confidence (≥0.7) pseudo-labels for further training.

Steps:

1. First complete Approach 1 (run Cell 1 or Cell 6)
2. Run **Cell 3**
3. Final model saves to `./final_model/`

python

```
# Step 1: Run Cell 1 or Cell 6
# Step 2: Run Cell 3
```

Approach 3: Data Augmentation + Top 50% Pseudo-labels

Train with augmented data, then use the top 50% highest-confidence predictions as pseudo-labels.

Steps:

1. First complete Approach 1 (run Cell 1 or Cell 6)
2. Run **Cell 9**
3. Final model saves to `./final_model/`

python

```
# Step 1: Run Cell 1 or Cell 6
# Step 2: Run Cell 9
```

Model Evaluation

After training, evaluate model performance on the test set:

```python
# Run Cell 2 or Cell 10
evaluate_on_test()
```

Outputs:

- Overall accuracy
- Precision, recall, F1-score per class
- Detailed classification report

Code Structure

Notebook Cells

Cell	Description
Cell 1	Complete augmented training with weighted loss
Cell 2	Test set evaluation function
Cell 3	Self-supervised learning (confidence threshold 0.7)
Cell 6	Simplified augmented training
Cell 9	Self-supervised learning (top 50% pseudo-labels)
Cell 10	Test set evaluation function (duplicate)

Key Components

- **Data Augmentation**: WordNet-based synonym replacement doubles training data
- **Custom Dataset**: ALBERT tokenization with max length 128
- **Weighted Loss**: Class weights of 0.7 (UNINFORMATIVE) and 1.7 (INFORMATIVE)
- **Pseudo-labeling**: Two strategies for utilizing unlabeled data

Expected Performance

Approach	F1 Score	Training Time	Description
Approach 1	~0.888	10-15 min	Baseline with data augmentation
Approach 2	~0.888+	15-25 min	High-quality pseudo-labels
Approach 3	Best	30-40 min	Maximum data utilization

Troubleshooting

Common Issues

1. **NLTK Download Errors**:

   python

   ```
   import nltk
   nltk.download('wordnet')
   nltk.download('averaged_perceptron_tagger')
   ```

2. **GPU Memory Issues**:
   - Reduce batch size in training arguments
   - Monitor CUDA memory usage

3. **Training Interruption**:
   - Model checkpoints are saved in `./results/checkpoint-*`
   - Training can resume from last checkpoint

Directory Structure

```
.
├── WNUT-2020-Task-2-Dataset/
│   ├── train.tsv
│   ├── valid.tsv
│   ├── test.tsv
│   └── unlabeled_test_with_noise.tsv
├── results/                 # Base model outputs
├── final_model/            # Final self-supervised model
├── logs/                   # TensorBoard logs
└── zyx1 (1) (1).ipynb     # Main notebook
```

Quick Start

To run the best-performing approach (Approach 3):

bash

```
# 1. Prepare the environment
conda activate opence-v1.5.1

# 2. Run basic training
# Execute Cell 6 in the notebook

# 3. Run self-supervised learning with top 50% pseudo-labels
# Execute Cell 9 in the notebook

# 4. Evaluate the model
# Execute Cell 10 in the notebook
```

License

This project is for academic research purposes only.