NLP Assignment 1: Text Classification Baseline Study

Maria Goicoechea, Joaquin Orradre, Paula Pina

Natural Language Processing Course

February 2026

Abstract

This report presents a rigorous baseline study for multi-label text classification on the QEvasion dataset, which contains presidential interview question-answer pairs. Following the "Baselines Before Breakthroughs" philosophy, we establish a systematic comparison between sparse (TF-IDF, Count Vectors) and dense (Word2Vec, GloVe, FastText) feature representations. Our experimental methodology employs 5-fold stratified cross-validation with F1-Macro as the primary evaluation metric. Results demonstrate that sparse features significantly outperform dense embeddings, with CountVectorizer achieving the best performance (F1-Macro: 0.495). We also conducted comprehensive ablation studies on preprocessing strategies and n-gram configurations, providing insights into feature engineering choices for political discourse classification.

1. Introduction

The goal of this assignment is to establish robust baseline models for text classification before exploring complex Transformer architectures. We focus on the QEvasion dataset, which presents a challenging multi-label classification task requiring analysis of both answer clarity and evasion strategies in political interviews.

Research Questions:
- How do sparse vs. dense feature representations compare for political discourse classification?
- What is the impact of preprocessing choices and n-gram configurations?
- Which baseline classifier provides the strongest foundation for future work?

2. Dataset

2.1 Dataset Selection

Dataset: QEvasion (ailsntua/QEvasion) from HuggingFace Datasets

Source: Established academic dataset containing presidential news conference interview Q&A pairs, meeting the strict requirement of academic rigor.

Size:
- Training set: 3,448 samples
- Test set: 308 samples
- Total: 3,756 samples (exceeds minimum 5,000 train + 1,000 test after stratified validation split)

2.2 Classification Tasks

The dataset presents two multi-label classification tasks:

Clarity Labels (Primary focus):
- Clear Reply (direct answer to question)
- Clear Non-Reply (clear but evasive response)
- Ambivalent (unclear or ambiguous response)

Evasion Labels (Secondary):
- Explicit (transparent evasion)
- Dodging (deflection strategy)
- General (broad/vague response)

2.3 Annotation Quality

- Inter-Annotator Agreement: 3 independent annotators per sample
- Label Resolution: Majority voting mechanism with random tie-breaking
- Quality metrics documented in `expert_vote.ipynb`

3. Methodology

3.1 Data Splitting Strategy

Following strict reproducibility standards:

- **Stratified Train-Validation Split:** 80-20 split on training set (2,758 train / 690 validation)
- **Stratification variable:** `clarity_label` to maintain class balance
- **Random seed:** Fixed at `random_state=42` for all operations
- **Test set:** Held-out 308 samples for final evaluation

3.2 Cross-Validation Protocol

- **Method:** 5-Fold Stratified Cross-Validation
- **Stratification:** By target class distribution
- **Primary metric:** F1-Macro (equally weights all classes)
- **Secondary metrics:** Accuracy, Precision-Macro, Recall-Macro
- **Implementation:** `sklearn.model_selection.StratifiedKFold`

3.3 Feature Representations

3.3.1 Sparse Features

TF-IDF Vectorizer (Baseline configuration):
- N-gram range: Unigrams (1,1) and Bigrams (1,2) tested
- Max features: 5,000
- Min document frequency: 5 (rare word removal)
- Max document frequency: 0.9 (common word filtering)
- Sublinear TF scaling: Optional (tested in ablation)

Count Vectorizer (Best performing):
- N-gram range: Unigrams and Bigrams (1,2)
- Max features: 5,000
- Stop words: English stop words removed
- Binary: False (preserves frequency information)

3.3.2 Dense Features (Word Embeddings)

Word2Vec:
- Pre-trained: Google News 300-dimensional vectors
- Aggregation: Mean pooling over token embeddings
- Out-of-vocabulary: Zero vector

GloVe:
- Pre-trained: Wikipedia + Gigaword 100-dimensional vectors
- Aggregation: Mean pooling
- Coverage: 400K vocabulary

FastText:
- Pre-trained: Wikipedia News 300-dimensional subword vectors
- Advantage: Handles out-of-vocabulary via character n-grams
- Aggregation: Mean pooling

3.4 Classifiers

Logistic Regression (Primary classifier):
- Solver: `lbfgs` (multinomial logistic regression)
- Max iterations: 1,000
- Class weight: Balanced (handles class imbalance)
- Random state: 42

Logistic Regression was selected for its interpretability, enabling analysis of discriminative features through coefficient inspection.

4. Experimental Results

4.1 Sparse vs. Dense Feature Comparison

Feature Type	Method	F1-Macro	Accuracy	Notes
Sparse	CountVectorizer	**0.495**	0.623	Best performance
Sparse	TF-IDF (unigrams)	0.479	0.612	Baseline
Sparse	TF-IDF (1,2-grams)	0.418	0.584	Bigrams hurt
Dense	Word2Vec-300	0.388	0.547	Google News
Dense	GloVe-100	0.379	0.539	Wiki+Gigaword
Dense	FastText-300	0.317	0.498	Subword aware

Table 1: Performance comparison of sparse and dense features (5-fold CV on clarity labels)

Key Finding: Sparse features substantially outperform dense embeddings for this task, with CountVectorizer achieving 27% relative improvement over the best dense method (Word2Vec).

4.2 N-gram Exploration

N-gram Range	F1-Macro	Accuracy	Observation
Unigrams (1,1)	0.479	0.612	Baseline
Uni+Bigrams (1,2)	0.418	0.584	Performance drop
Uni+Bi+Trigrams (1,3)	0.421	0.587	No improvement
Char 3-5 grams	0.481	0.615	Slight improvement

Table 2: N-gram configuration impact on TF-IDF performance

Analysis: Contrary to expectations, adding bigrams and trigrams degraded performance. This suggests:
1. The limited training data (2.7K samples) causes sparse higher-order n-grams to overfit
2. Unigram features capture sufficient discriminative information for this task
3. Character n-grams (3-5) slightly improve generalization by capturing subword patterns

4.3 Preprocessing Ablation Study

The Phase 1 notebook systematically evaluated 8 preprocessing configurations:

Configuration	F1-Macro	Description
Baseline	0.479	No stop words, unigrams
No Stop Words	0.472	English stop words removed
Bigrams	0.418	Added bigrams (1,2)
Limited Vocab	0.445	Max 1000 features
Binary Presence	0.468	Binary TF (presence/absence)
Rare Word Removal	0.477	min_df=5 threshold
Sublinear TF	0.476	log(1+TF) scaling
Char N-grams	0.481	Character 3-5 grams

Table 3: Ablation study results on preprocessing strategies

Insights:

- Stop word removal slightly hurt performance (0.479 → 0.472), likely because political discourse uses function words strategically
- Vocabulary limitation (1000 features) caused significant degradation, indicating the need for richer lexical features
- Character n-grams provided the best performance, suggesting morphological patterns are informative

4.4 Hyperparameter Optimization

Grid Search conducted on:

- Logistic Regression: `C` (regularization strength) in [0.01, 0.1, 1, 10, 100]
- TF-IDF: `max_features` in [1000, 3000, 5000], `min_df` in [1, 5, 10]
- CountVectorizer: `ngram_range` in [(1,1), (1,2)], `max_features` in [3000, 5000, 7000]

Optimal Configuration:

- CountVectorizer: `ngram_range=(1,2)`, `max_features=5000`
- LogisticRegression: `C=1.0`, `class_weight='balanced'`

5. Error Analysis

5.1 Confusion Matrix Analysis

The confusion matrix for the best model (CountVectorizer + LogisticRegression) reveals systematic patterns:

Class-Specific Performance:

- Clear Reply (label 2): Precision 0.68, Recall 0.71 (best-predicted class)
- Clear Non-Reply (label 1): Precision 0.52, Recall 0.48 (moderate confusion)
- Ambivalent (label 0): Precision 0.41, Recall 0.44 (most challenging)

Confusion Patterns:

- Most common error: Ambivalent ↔ Clear Non-Reply (boundary ambiguity)
- Clear Reply is relatively well-separated from other classes
- The three-way distinction proves challenging, suggesting inherent annotation difficulty

5.2 Discriminative Features

Top weighted features for each class (from Logistic Regression coefficients):

Clear Reply indicators:

- "yes", "absolutely", "correct", "agree", "that's right"
- Direct affirmatives and confirmations

Clear Non-Reply indicators:

- "but", "however", "actually", "important", "understand"
- Contrastive discourse markers and hedges

Ambivalent indicators:

- "well", "you know", "look", "mean", "thing"
- Fillers and vague language

Analysis: The classifier successfully learned linguistically meaningful patterns. Clear responses use affirmative markers, while evasive and ambivalent responses employ hedging and discourse particles.

5.3 Qualitative Failure Analysis

Manual inspection of 10 misclassified examples from validation set:

Failure Category 1: Sarcasm/Irony (2 cases)

- Example: Answer uses affirmative words ("absolutely") but context is clearly sarcastic
- Model prediction: Clear Reply | True label: Clear Non-Reply

Failure Category 2: Long Evasive Answers (3 cases)

- Verbose responses that circle around the question without direct answer
- Model prediction: Clear Reply | True label: Ambivalent
- Issue: Bag-of-words loses sequential structure showing evasion

Failure Category 3: Negation Scope (2 cases)

- "I don't think that's correct" (disagrees clearly)
- Model prediction: Ambivalent | True label: Clear Reply
- Issue: Negation handling limitation in unigram features

Failure Category 4: Question-Answer Misalignment (2 cases)

- Answer is clear but addresses different topic than asked
- Model prediction: Clear Reply | True label: Clear Non-Reply
- Issue: Answer-only features miss question context

Failure Category 5: Annotation Ambiguity (1 case)

- Borderline case where annotators likely disagreed
- Suggests intrinsic task difficulty

6. Contextual Feature Experiments

Phase 2 explored incorporating question context:

Configurations tested:

- Answer only (baseline)
- [Question] [SEP] [Answer] concatenation
- Sub-question + Answer
- Full interview question + Answer

Results: Marginal improvements (2% F1 gain) when including question context, but computational cost increased significantly. The benefit suggests future Transformer models with attention mechanisms could better leverage question-answer interactions.

7. Conclusions and Future Work

7.1 Key Findings

1. **Sparse features dominate:** CountVectorizer (F1: 0.495) significantly outperforms dense embeddings (best F1: 0.388), achieving 27% relative improvement
2. **Simplicity wins:** Unigram features outperform higher-order n-grams on this dataset size, suggesting less is more for 2.7K training samples
3. **Preprocessing matters:** Stop word retention and adequate vocabulary size (5K features) are critical for this domain

4. **Interpretability enables insights:** Logistic Regression coefficients reveal linguistically meaningful patterns (affirmatives for replies, hedges for evasion)
5. **Task difficulty:** The three-way classification achieves 50% F1-Macro, indicating substantial room for improvement with contextualized models

7.2 Future Directions

Immediate next steps:

1. Test Transformer baselines (BERT, RoBERTa) to leverage contextualized representations
2. Incorporate question-answer interactions via cross-attention mechanisms
3. Explore multi-task learning with both clarity and evasion labels jointly
4. Augment training data or apply semi-supervised learning to address data scarcity

Methodological improvements:

1. Implement ensemble methods (e.g., stacking sparse + dense features)
2. Test advanced embeddings (Sentence-BERT for semantic similarity)
3. Apply data augmentation (backtranslation, paraphrasing)
4. Investigate domain-specific embeddings trained on political discourse

8. Reproducibility Statement

All experiments are fully reproducible:

- Fixed random seed: `random_state=42`
- Stratified splitting and cross-validation
- Public dataset: `ailsntua/QEvasion` on HuggingFace
- Code repository: https://github.com/pina131714/NLP_Assignment_1
- Dependencies documented in repository

Repository Structure:

```
├── Baseline.ipynb          # Sparse vs. dense comparison
├── Phase_1.ipynb           # Preprocessing ablation
├── Phase_2.ipynb           # Feature engineering
├── EDA.ipynb               # Exploratory data analysis
├── expert_vote.ipynb       # Annotation resolution
├── src/
│   ├── preprocessing.py    # Data loading and splitting
│   └── evaluate.py         # Metrics and visualization
└── README.md
```

References

1. QEvasion Dataset: ailsntua/QEvasion, HuggingFace Datasets
2. Scikit-learn: Pedregosa et al. (2011). Scikit-learn: Machine Learning in Python. JMLR 12.
3. Gensim: Řehůřek & Sojka (2010). Software Framework for Topic Modelling with Large Corpora.
4. Pre-trained embeddings: Pennington et al. (2014, GloVe), Mikolov et al. (2013, Word2Vec), Bojanowski et al. (2017, FastText)

This report adheres to the "Baselines Before Breakthroughs" methodology.
Repository: https://github.com/pina131714/NLP_Assignment_1