Sarcasm Detection Baseline Experiments

Technical Report

1. Overview

This report documents the baseline experiments conducted for sarcasm detection in sitcom dialogue. The purpose of these experiments is to establish baseline performance before fine-tuning, identify model failure patterns, and understand what linguistic and contextual features affect classification accuracy.

Key takeaway for the paper: These baseline results establish performance floors that any fine-tuned model must exceed. The error analysis reveals systematic biases (character name reliance, surface marker confusion) that should inform fine-tuning strategies.

2. Experimental Setup

2.1 Dataset Configuration

Dataset	Size	Purpose
Main Dataset (Orange)	690 examples	Training and evaluation pool
Gold Dataset	28 examples	Held-out test set (professionally annotated)
Orange-Clean	662 examples	Orange after removing gold examples

Why use both datasets?

- The gold dataset (28 examples) is too small for training but serves as a high-quality held-out evaluation set
- The orange dataset provides sufficient examples for training and creating validation/test splits
- Gold examples were removed from orange to prevent data leakage

2.2 Data Splits

The orange-clean dataset (662 examples) was split using stratified sampling to preserve label distribution:

Split	Size	Percentage	Label 0 (Literal)	Label 1 (Sarcastic)
Train	529	79.9%	266	263
Validation	66	10.0%	33	33
Test	67	10.1%	34	33
Gold (held-out)	28	—	12	16

Note for paper: The dataset is nearly perfectly balanced (~50/50 split between literal and sarcastic examples).

2.3 Experimental Conditions

Three independent variables were tested:

1. Input Type

- `text_only`: Only the target utterance to classify
- `context_text`: Dialogue context combined with target utterance
- `context_only`: Only the dialogue context (ablation)

2. Name Normalization

- `original`: Character names preserved (e.g., "CHANDLER:", "SHELDON:")
- `normalized`: All names replaced with "SPEAKER:"

3. Pooling Method (Tier 3 only)

- `cls`: Use CLS token embedding
- `mean`: Mean pooling across all tokens

2.4 Models Tested

Tier 1: Sanity Check Baselines

- Random baseline (predicts based on training class distribution)
- Majority baseline (always predicts majority class)

Tier 2: Classical Machine Learning

- TF-IDF + Logistic Regression
- TF-IDF + Support Vector Machine (SVM)

Tier 3: Frozen Transformer Embeddings

- XLM-RoBERTa (frozen) + Logistic Regression

Note for paper: All Tier 2 and Tier 3 models use frozen representations without fine-tuning. This establishes what performance is achievable from general-purpose features alone.

---

3. Results

3.1 Overall Performance Summary

Best Results on Gold Set (28 examples)

Rank	Model	Input Type	Names	Pooling	Accuracy	F1
1	XLM-R + LogReg	text_only	original	mean	**82.14%**	0.816
2	XLM-R + LogReg	text_only	normalized	mean	78.57%	0.781
3	TF-IDF + LogReg	text_only	original	—	75.00%	0.747

Rank	Model	Input Type	Names	Pooling	Accuracy	F1
4	XLM-R + LogReg	context_text	original	cls	75.00%	0.742
5	XLM-R + LogReg	text_only	original	cls	71.43%	0.708

Best Results on Test Set (67 examples)

Rank	Model	Input Type	Names	Pooling	Accuracy	F1
1	TF-IDF + SVM	context_text	original	—	68.66%	0.684
1	XLM-R + LogReg	text_only	original	mean	68.66%	0.686
1	XLM-R + LogReg	context_text	original	cls	68.66%	0.686
1	XLM-R + LogReg	context_text	original	mean	68.66%	0.684
5	TF-IDF + LogReg	text_only	original	—	68.66%	0.686

Tier 1 Baselines (Sanity Check)

Model	Test Accuracy	Gold Accuracy
Random	50.75%	53.57%
Majority	50.75%	42.86%

Note for paper: All trained models substantially outperform random baselines, confirming the models are learning meaningful patterns.

3.2 Effect of Name Normalization

Replacing character names with generic "SPEAKER:" tags:

Metric	Value
Mean accuracy change	**-4.10%**
Conditions where normalization helps	3
Conditions where normalization hurts	12
Conditions with no change	1

Largest Effects

Model	Input	Dataset	Accuracy Change
TF-IDF + LogReg	context_text	gold	**-14.29%**
TF-IDF + SVM	context_text	gold	**-10.71%**
TF-IDF + SVM	text_only	test	**-10.45%**

Model	Input	Dataset	Accuracy Change
XLM-R + LogReg (cls)	text_only	gold	+5.36%
XLM-R + LogReg (mean)	text_only	gold	+3.57%

Finding: Normalization hurts TF-IDF models significantly more than XLM-R models. TF-IDF directly uses character names as features, while XLM-R's semantic representations are more robust.

Note for paper: This indicates models are using character identity as a classification shortcut rather than learning generalizable sarcasm patterns.

3.3 Effect of Including Context

Adding dialogue context to the target utterance:

Metric	Value
Mean accuracy change	**+0.32%**
Conditions where context helps	8
Conditions where context hurts	5

Finding: Context provides minimal benefit in baseline models. The frozen representations do not effectively leverage pragmatic relationships between context and response.

Note for paper: Fine-tuned models may show larger context effects if they learn to detect incongruity between context and response.

3.4 CLS vs Mean Pooling (Tier 3)

For XLM-R embeddings:

Pooling	Best Gold Accuracy	Observation
Mean	**82.14%**	Consistently better on gold
CLS	75.00%	Lower performance

Finding: Mean pooling outperforms CLS pooling, suggesting that aggregating information across all tokens captures more relevant signal than relying on the CLS token alone.

---

4. Error Analysis

4.1 Quantitative Error Breakdown

Total unique errors analyzed: 78 examples

Prediction Flips by Normalization

Direction	Count

Direction	Count
Normalization helps (wrong→correct)	66
Normalization hurts (correct→wrong)	91
Total flips	**157**

Prediction Flips by Input Type

Direction	Count
Context helps (wrong→correct)	83
Context hurts (correct→wrong)	58
Total flips	**141**

Model Disagreements

92 unique examples where different models disagree on the prediction.

4.2 Linguistic Feature Analysis

Features were extracted from all predictions and compared between correct and incorrect predictions.

Features More Common in INCORRECT Predictions

Feature	Difference	Interpretation
all_caps_count	+1440%	ALL CAPS text confuses model
ellipsis_count	+166%	"..." patterns confuse model
sentiment_polarity	+114%	Strong positive sentiment confuses model
sarcasm_marker_count	+64%	Words like "oh", "great", "sure" confuse model
starts_with_interjection	+61%	Starting with "Oh", "Well" confuses model
positive_word_count	+57%	Positive words confuse model
intensifier_count	+37%	Words like "very", "really" confuse model

Features More Common in CORRECT Predictions

Feature	Difference	Interpretation
negative_word_count	-38%	Negative words help model
has_contrast	-38%	"but", "however" help model
has_rhetorical_pattern	-26%	Rhetorical questions help model
negation_count	-25%	Negation words help model

Note for paper: This reveals a critical limitation — the baseline models fail on examples with surface sarcasm markers. The markers that humans associate with sarcasm (exclamations, "oh", "great") actually make classification HARDER, not easier. This suggests models cannot distinguish genuine vs. ironic use of these markers.

4.3 False Positive vs False Negative Comparison

Error Type	Count	Description
False Positives (FP)	365	Literal examples predicted as sarcastic
False Negatives (FN)	452	Sarcastic examples predicted as literal

Features Distinguishing FN from FP

(Positive percentage = higher in False Negatives)

Feature	Difference
intensifier_count	+583%
has_intensifier	+465%
exclamation_count	+121%
sarcasm_marker_count	+117%
has_exclamation	+116%
starts_with_interjection	+106%

Finding: The model misses sarcasm (FN) in examples that have MORE sarcasm markers, not fewer. This is counterintuitive but explained by the fact that obvious sarcasm may be "too obvious" — heavy marker use in sarcastic examples looks similar to genuine enthusiasm in literal examples.

---

5. Clustering Analysis

Two clustering approaches were applied to group error examples and identify patterns.

5.1 TF-IDF Clustering (Lexical Similarity)

Metric	Value
Number of clusters	12
Silhouette score	0.0153
FN-dominant clusters	6
FP-dominant clusters	5

5.2 Semantic Clustering (Meaning Similarity)

Metric	Value
Number of clusters	8
Silhouette score	0.0860
FN-dominant clusters	6
FP-dominant clusters	2

The higher silhouette score for semantic clustering indicates more coherent groupings based on meaning rather than shared vocabulary.

5.3 Character-Based Error Patterns

Both clustering methods identified consistent character associations:

Character	Error Mentions	Primary Error Type
Chandler	21	FN (missed sarcasm)
Howard	10	FN (missed sarcasm)
Leonard	9	FP (false sarcasm)
Person (generic)	7	Mixed
Sheldon	6	FP (false sarcasm)

Finding: Chandler dominates the errors — 21 out of 78 unique error examples mention Chandler. These are predominantly False Negatives, meaning the model fails to detect Chandler's sarcasm despite likely having learned a "Chandler = sarcastic" association.

Note for paper: This "Chandler Effect" suggests the model has learned a spurious correlation (character identity predicts sarcasm) but fails when that character's sarcasm is subtle or marker-free.

5.4 Identified Error Clusters

FN Clusters (Missed Sarcasm)

Cluster	Size	Character	Pattern
Cluster 1	34	Chandler (21)	Dry conversational sarcasm, minimal markers
Cluster 5	13	Howard (10)	Cultural references, self-deprecating humor
Cluster 2	6	Amy (3)	Intellectual backhanded compliments
Cluster 4	3	Dorothy (3)	Ultra-short deadpan one-liners

Examples of missed sarcasm:

- "If that doesn't keep kids in school, what will?" (Chandler)
- "It's why my people wandered the desert for 40 years. Took that long to walk it off." (Howard)

- "Hope your hands are steady. It's the width of a single hair. But this is just biology, so I'm sure it's no problem for a genius like you." (Amy)

FP Clusters (False Sarcasm)

Cluster	Size	Character	Pattern
Cluster 3	11	Leonard (9)	Practical suggestions, genuine questions
Cluster 7	3	Sheldon (2)	Technical explanations

Examples of false sarcasm:

- "Come on, let's just start walking. There's got to be a gas station or something nearby." (Leonard)
- "It's white plastic, with a handle, and it fits onto a stroller." (Joey)

---

6. Summary of Key Findings

6.1 Performance Findings

1. **Best baseline accuracy**: 82.14% on gold (XLM-R, mean pooling, text_only, original names)
2. **Best baseline accuracy on test**: 68.66% (multiple conditions tied)
3. **XLM-R outperforms TF-IDF** on the gold set by ~7 percentage points
4. **Mean pooling outperforms CLS pooling** consistently

6.2 Normalization Findings

1. **Normalization hurts performance** on average (-4.1% accuracy)
2. **TF-IDF models are more affected** by normalization than XLM-R
3. **91 examples flip from correct to incorrect** when names are normalized
4. **Models rely on character identity** as a classification shortcut

6.3 Context Findings

1. **Context provides minimal benefit** in frozen baselines (+0.32% average)
2. **83 examples improve with context**, but 58 examples worsen
3. **Frozen representations cannot effectively leverage** pragmatic context

6.4 Linguistic Findings

1. **Surface sarcasm markers correlate with errors**, not correct predictions
2. **Sentiment polarity is 114% higher** in incorrect predictions
3. **Negative words and contrast markers** help classification
4. **False Negatives have more markers** than False Positives

6.5 Character Bias Findings

1. **Chandler dominates errors** (21/78 unique errors, 27%)
2. **Chandler/Howard errors are FN** (model misses their sarcasm)
3. **Leonard/Sheldon errors are FP** (model falsely predicts sarcasm)

4. **Both clustering methods agree** on character patterns

---

7. Implications for Fine-Tuning

Based on these baseline findings:

1. **Character bias must be addressed**: Fine-tuning on original names may reinforce spurious correlations. I will train with normalized names and mixed conditions.
2. **Context usage requires learning**: Frozen models do not benefit from context. Fine-tuned models should be evaluated specifically on whether they learn to use context for incongruity detection.
3. **Surface markers are unreliable**: Models should not rely on words like "oh", "great", "sure" as sarcasm indicators. Fine-tuning should teach discrimination between genuine and ironic use.
4. **Dry sarcasm is the hard case**: The model consistently misses sarcasm that lacks lexical markers. This is the key challenge for improvement.
5. **Mean pooling is preferred**: For transformer-based models, mean pooling provides better representations than CLS token alone.pdf

---

8. Files and Reproducibility

8.1 Pipeline Scripts

Step	Script	Purpose
1	step1_split_data.py	Load data, remove gold from orange, create splits
2	step2_preprocess.py	Apply preprocessing for all conditions
3	step3_run_baselines.py	Train and evaluate Tier 1/2/3 models
4	step4_analysis_quantitative.py	Error breakdown, normalization/context effects
5	step5_analysis_linguistic.py	Linguistic feature extraction and comparison
6	step6_analysis_clustering.py	TF-IDF clustering of errors
6b	step6b_analysis_clustering_semantic.py	Semantic embedding clustering

8.2 Output Files

Directory	Contents
outputs/data_splits/	train.csv, val.csv, test.csv, gold.csv
outputs/preprocessed/	48 preprocessed condition files
outputs/predictions/	48 prediction files with per-example results
outputs/analysis/	Quantitative and linguistic analysis results

Directory	Contents
outputs/clustering_analysis/	TF-IDF clustering results
outputs/clustering_semantic/	Semantic clustering results

8.3 Configuration

- Random seed: 42
- Train/Val/Test split: 80/10/10
- TF-IDF: max_features=5000, ngram_range=(1,2)
- XLM-R: xlm-roberta-base (frozen)
- Clustering: K-Means with auto-tuned k (silhouette score)
- Semantic embeddings: all-MiniLM-L6-v2
