Sarcasm Detection in Tweets

Christa Sparks

CSPB 4830-001-750

Spring 2025

Abstract

Detecting sarcasm on Twitter is tricky because it depends on subtle cues; tone, context, and wordplay. In this project, I built two pipelines: a technical TF–IDF plus hand-engineered feature model with logistic regression, and a fine-tuned BERT transformer. I ran both quick and extended hyperparameter searches, using RandomizedSearchCV for the Feature-Driven Model pipeline and Optuna for BERT, under practical time constraints. Our optimized ML model hits an F1 of ~0.64, while BERT reaches ~0.73 before tuning and ~0.726 after prolonged tuning (3 epochs). Error analysis shows they fail differently: each catches some cases the other misses. We weigh runtime vs. accuracy, complexity vs. simplicity, and suggest combining engineered features with transformer embeddings for a next-gen sarcasm detector.

---

1. Introduction

Sarcasm flips literal meaning, making sentiment analysis on social media prone to mistakes. A reliable sarcasm detector could boost tools for brand monitoring, customer support, and chatbots. We compare two approaches:

1. A technical TF–IDF + hand-crafted feature pipeline with logistic regression.
2. A BERT transformer fine-tuned for sarcasm.

We push both through hyperparameter optimization to see how far we can stretch them under realistic time budgets.

2. Related Work

Castro et al. (2019)

Summary: Introduced the MUStARD multimodal sarcasm dataset and showed that combining text with visual and audio cues uncovers sarcastic mismatches that text-only models miss.

Relevance: Even though I only have text, I borrow the idea of spotting "incongruences"—for example, a positive word appearing in a negative context—and use F1 benchmarks from their work to gauge my own model's performance.

Bamman & Smith (2015)

Summary: Demonstrated that adding conversational context and simple user metadata (who's talking to whom, user history) significantly improves Twitter sarcasm detection.

Relevance: While I stuck to single tweets, this suggests that even rudimentary user-level features (average sentiment, posting style) could boost my pipeline.

Van Hee, Lefever & Hoste (2018)

Summary: Released the SemEval-2018 irony dataset with subcategories of irony, and found that combining word-level and character-level n-grams yields strong baseline results.

Relevance: I follow their preprocessing guidelines and standard precision/recall/F1 evaluation, and I mix word and char TF–IDF just like they did.

Onan & Toçoğlu (2021)

Summary: Proposed stacking BiLSTM layers on term-weighted embeddings (e.g., TF–IDF or inverse gravity moment) to highlight key words in sarcastic tweets.

Relevance: Their term-weighting tricks inspire how I might weight or augment my BERT embeddings or hand-crafted features to better catch subtle sarcasm cues.

Rajadesingan, Zafarani & Liu (2015)

Summary: Modeled users' tweeting behavior—sudden sentiment swings, habitual snark—to detect sarcasm, showing that behavioral patterns add predictive power.

Relevance: Even on isolated tweets, I can engineer features (like sentiment change tags or repeated sarcasm markers) that capture those broader behavioral signals.

3. Data

We use the SemEval-2018 Task 3 irony dataset (~4,000 tweets). Each instance has an ID, a binary label (ironic vs. non-ironic), and the tweet text. Preprocessing steps:

1. Expand contractions, lowercase.
2. Strip URLs, mentions, and hashtag symbols.
3. Simple negation tagging (not good → not good_NEG until punctuation).
4. Split into 80/20 train/test.

4. Methodology

4.1 Pipeline A: Feature-Driven Model

- **Features**:

    - Word TF–IDF (unigrams, bigrams)
    - Character TF–IDF (3–5 char n-grams)
    - Hand-crafted: VADER sentiment scores, punctuation counts, adjective ratio, emoticon counts

- **Classifier**: LogisticRegression (max_iter=1000)

- **HPO**: RandomizedSearchCV (5 candidates, 3-fold CV) tuning n-gram range, regularization strength C, and class_weight

4.2 Pipeline B: BERT Transformer

- **Model**: bert-base-uncased fine-tuned for binary classification
  Tokenization: max length 256, padding/truncation
- **Trainer API**: evaluate each epoch
- **HPO**: Optuna (3 trials with early stopping) over learning rate [1e-5,5e-5], epochs up to 3, and weight decay [0,0.3]

---

5. Results

5.1 No Tuning Baseline

- **ML**: F1 ~ 0.61.
- **BERT**: F1 ~ 0.73 (limited-time run).

5.2 Quick HPO Run

- **ML**: best params → C=2.54, class_weight=balanced, unigrams; F1 = 0.64.
- **BERT**: (2 Optuna trials) lr~1.90e-5, 1 epoch, wd~0.27; final F1 ~ 0.655.

5.3 Extended HPO Run (3 Trials, 3 Epochs)

- **Best hyperparameters**: learning_rate = 1.277×10^{-5}, num_train_epochs = 3, weight_decay = 0.0726.

- **Test set evaluation**:
  - eval_loss = 0.6629
  - accuracy = 0.6963
  - precision = 0.7106
  - recall = 0.7415
  - **F1 = 0.7258**

This extended tuning run took over 3 hours (~1 hour per epoch) and delivered a modest F1 boost over the quick HPO. For comparison, the shorter ML + BERT quick run finished in just over 1 hour, highlighting the trade-off between compute time and marginal performance gains.

---

6. Error Analysis

Error Type	ML Errors	BERT Errors	Overlap Errors
False Positives	106	105	42
False Negatives	163	161	100

Both pipelines share many misses but also catch unique cases, hinting that a simple ensemble could boost overall coverage.

---

7. Discussion

- **Compute vs. Gains**: Extended tuning costs hours (~1 hour/epoch) for a small bump in F1.
- **Complexity**: BERT shines but needs more engineering. Our feature-driven model is lighter and still respectable.
- **Complementarity**: Hand-crafted features cover some quirks BERT misses.

---

8. Conclusion & Future Work

We built two sarcasm detectors: a technical TF–IDF + features model and a transformer-based model. BERT leads in F1 but demands resources. Next steps:

- Ensemble both approaches

  Incorporate conversation threads
- Explore multimodal signals

References

Castro, S., Hazarika, D., Pérez-Rosas, V., Zimmermann, R., Mihalcea, R., & Poria, S. (2019). Towards
multimodal sarcasm detection (an _obviously_ perfect paper). *arXiv preprint arXiv:1906.01815*

Bamman, D., & Smith, N. (2015). Contextualized sarcasm detection on twitter. In *proceedings of the
international AAAI conference on web and social media* (Vol. 9, No. 1, pp. 574-577).

Van Hee, C., Lefever, E., & Hoste, V. (2018, June). Semeval-2018 task 3: Irony detection in english
tweets. In *Proceedings of the 12th international workshop on semantic evaluation* (pp. 39-50).

Onan, A., & Toçoğlu, M. A. (2021). A term weighted neural language model and stacked bidirectional
LSTM based framework for sarcasm identification. *Ieee Access*, *9*, 7701-7722

Rajadesingan, A., Zafarani, R., & Liu, H. (2015, February). Sarcasm detection on twitter: A behavioral
modeling approach. In *Proceedings of the eighth ACM international conference on web search
and data mining* (pp. 97-106).