Sentiment Classifier Report

Andreas Pashiardis

November 15, 2025

1 Introduction

This report presents the systematic optimisation of a sentiment classification analysis model, set out to improve pre-processing, feature extraction, and overall model performance for tweets.

Building upon a foundational layer of a Linear SVC and guided by targeted error analysis Section 2, two approaches were implemented and compared , quantifying trade-offs between accuracy, robustness (to noisy social text) and computational cost.

Performance improvements were validated using a 10-fold cross-validation and all scores reported below are to 3 decimal places.

2 Error Analysis

The first fold of the Cross validation (CV) dataset derived from the baseline model was analysed and contained:

- **True positives** : 1654
- **True negatives**: 563
- **False positives**: 255
- **False negatives**: 212

The classifier appears to be slightly biased towards predicting *positive*. Some of the typical failure modes identified suggest:

- Over-reliance on isolated sentiment words (e.g. *love*, *Sunday*, *thank*)
- Weak handling of negation, contrast and sarcasm
- Poor adaptation to domain-specific topics

2.1 Quantitative Error Analysis

(1) Positive words inside complaints (≈ 30% of False Positives)

Many negative tweets are complaints or criticism but contain locally positive or neutral tokens, e.g. *love USC*, *Sunday*, *ice cream*, *prayer*. This is expected as a bag-of-words model overweights these, flipping the overall prediction to positive.

(2) Negation/ obligation misunderstood (≈ 25% of False Positives)

Negations and obligation phrases (“have to”, “need to”) are poorly captured. The model fails to detect aversion to unwanted tasks.

> *“I really wish* ***didn’t have to take the ACT tomorrow.****”*

(3) Positive intent with negative words (≈ 25% of False Negatives)

Tweets that contain negative words but express positive intent are often misclassified as negative.

> *“I can and will* ***fight for #BlackLivesMatter.****”*

The words *fight* and *anxiety* are lexically negative, but the sentiment context is clearly positive.

(4) Sarcasm and irony (≈ 25% of False Negatives)

Sarcasm and political commentary often invert polarity, but the classifier reads literal word sentiment.

> *“A Monday without teen wolf is a with less stress and anxiety”*

2.2 Recommendations

Preprocessing should explicitly handle contractions and negations to correct errors where negative tweets containing phrases like “don’t like” were misclassified as positive. Hashtags, emojis, and elongated words should be preserved as sentiment cues, while contrastive connectives such as *but*, *however*, and *though* should be marked to capture polarity shifts that previously caused false positives. Normalising slang and informal expressions (e.g., “lol”) would also reduce confusion.

3 Two Modelling Approaches

For feature extraction and modelling, this report explored two complementary approaches. The first employed a **rich preprocessing pipeline combined with TF–IDF** weighting and targeted hyperparameter tuning, ending in the baseline LinearSVC. This configuration emphasised discriminative terms via IDF scaling, integrated n-gram features to capture contextual expressions (e.g., “can’t wait to” or “fight for”), and incorporated domain-specific sentiment lexicons, hashtag sentiment mapping, and emoji polarity cues to improve tone and topic sensitivity. —see Section 4

Building on the insights and limitations identified in this first approach, a second, more contextual framework was developed using the **BERTweet Hybrid model**. This model leveraged deep contextual embeddings from BERTweet in combination with subword-level TF–IDF features with a final layer of the baseline LinearSVC, enabling richer representation of semantics. Together, these two approaches demonstrate a progressive shift from traditional bag-of-words methods toward contextualised transformer-based modelling for robust sentiment detection in tweets. —see Section 5

4 Baseline Model (Rich pre-processing + TF-IDF + LinearSVC)

4.1 Tokenisation and Normalisation

Pre-processing was extended beyond tokenisation to handle noisy social text: The tweets were Unicode normalised, lowercased, removed URLs and digits, split hashtags into words, converted emojis to text, replaced common slang, and expanded contractions (e.g., “don't” → “do not”). These steps improved token uniformity and reduced sparsity, resulting in an F1 score of **0.842**.

4.2 Lemmatization vs. Stemming

Lemmatization (WordNet) preserved meaningful forms (e.g., “better” → “good”) and produced cleaner features than Porter stemming, resulting in an F1 score of **0.851**.

4.3 Additional features

Only semantically neutral stopwords were removed while retaining negations (“not”, “never”, “no”), and preserved “!” and “?” because they convey sentiment.

4.3.1 N-gram

Extending the feature space to include bigrams, captured multi-word expressions such as “not good” and “very happy” tackling the misclassified Negation phrases issue and achieving an F1 score of **0.867**.

4.3.2 Opinion Lexicon

External sentiment lexicon from Bing Liu's *Opinion Lexicon*, were incorporated. For each tweet, counts of positive and negative words were computed and appended as numeric features resulting in F1 = **0.878**.

4.4 Model Architecture

To exploit both lexical and subword information, word (pre-processed) and character-level TF–IDF features were combined in a single `pipeline` using a `FeatureUnion`, feeding into a downstream `LinearSVC` classifier:

4.4.1 Character n-grams

Character-level n-grams (3–5) were used to capture subword structures, and stylistic variations typical in tweets (e.g., “happyyyyy”, “luv”) increasing F1 score to **0.886**.

4.4.2 Stylistic and Lexical Features (not retained)

Additional features were engineered to capture stylistic and lexical tendencies, but were not included in the final model as they did not improve validation performance:

- **Sentence-length statistics.** For each tweet, the minimum, maximum, and mean sentence length were computed. These values were discretised by dividing by 5 (simulating categorical variable behavior) and encoded as special tokens
- removing user mentions from the text
- handling negations by joining them with the following token (e.g. `not_good`)
- marking tokens in the scope of contrastive connectives such as “but” or “however”
- expanding contractions (e.g. `can't` → `cannot`)

4.5 Model Hyperparameters

To ensure stable convergence during training, the maximum number of iterations in the `LinearSVC` classifier was increased from its default value to `max_iter = 3000`.

Additionally, a custom class-weight dictionary was tested to compensate for potential label imbalance. However, this led the model to slightly overfit. Therefore, the final model retained the default balanced weighting scheme with `max_iter = 3000` .

System Variation	Description	F1 Score (3 d.p.)
Baseline	Unigrams	0.827
Improved Pre-processing	Normalisation + Lemmatization	0.851
N-grams	Add (1,2)-grams + Binary	0.867
Opinion Lexicon	External lexicon features	0.878
TF–IDF Weighting	Word + Char (3–5)-grams	0.886

Table 1: Effect of different preprocessing and feature extraction techniques on F1 score.

5 BERTweet Hybrid Model

A custom model was developed combining contextual embeddings from **BERTweet** with character-level TF–IDF features within a unified `pipeline`. The transformer `BertweetEmbeddings` produces dense sentence representations from the pretrained `bertweet-base`. Each tweet is only lightly normalised (Unicode, URL, emoji handling) and then tokenised with the BERTweet tokenizer, after which the `[CLS]` embedding is extracted. Preprocessing was kept minimal since BERTweet's tokenization and contextual pretraining already capture morphology, misspellings and word order. Aggressive steps like lemmatisation, stemming, or handcrafted n-grams can discard useful cues and reduce alignment with pretrained distributions.

The resulting embeddings are fused with 3–5 character n-gram TF–IDF vectors via a `FeatureUnion` and fed to a downstream `LinearSVC` (`max_iter`=3000, class_weight = balanced).

Performance & Latency. This hybrid approach effectively leverages semantic context *and* subword cues (handling misspellings and informal patterns), but at a computational cost:

- **Single fold (GPU):** training on 26,832 tweets with 6,708 test instances took ∼10 minutes.
- **Baseline TF–IDF+LinearSVC (CPU):** ∼2 minutes for an equivalent single fold.
- **Cross-validation:** Baseline (with CPU) completes in ∼10 minutes, whereas the BERTweet hybrid (with GPU) requires approximately ∼1 hour.

The hybrid improves robustness to noisy social text, trading off increased latency and compute.

Approach	Precision	Recall	F_1 Score
Rich preprocess + TF–IDF + SVC	0.886	0.887	0.886
BERTweet + TF–IDF + SVC	0.912	0.911	0.912

Table 2: Effect of different pipeline approaches on Cross validation performance.

References

[Bhagat and Mane, 2020] Bhagat, C. and Mane, D. (2020). Text categorization using sentiment analysis. In *Proceedings of the International Conference on Computational Science and Applications*, pages 361–368, Saint Petersburg, Russia. Springer.

[Chang et al., 2023] Chang, H.-S., Sun, R.-Y., Ricci, K., and McCallum, A. (2023). Multi-cls bert: An efficient alternative to traditional ensembling. *arXiv preprint arXiv:2210.05043*.

[Jayakody and Kumara, 2021] Jayakody, J. P. U. S. D. and Kumara, B. T. G. S. (2021). Sentiment analysis on product reviews on twitter using machine learning approaches. In *Proceedings of the 2021 International Conference on Decision Aid Sciences and Application (DASA)*, pages 1056–1061, Sakheer, Bahrain. IEEE.

[Leung et al., 2022] Leung, M. F., Wang, J., and Li, D. (2022). Decentralized robust portfolio optimization based on cooperative-competitive multiagent systems. *IEEE Transactions on Cybernetics*, 52:12785–12794.

[Liu, 2004] Liu, B. (2004). Opinion lexicon. https://www.cs.uic.edu/~liub/FBS/sentiment-analysis.html#lexicon. Accessed: 2025-11-10.

[Nguyen et al., 2020] Nguyen, D. Q., Vu, T., and Tuan Nguyen, A. (2020). Bertweet: A pre-trained language model for english tweets. https://huggingface.co/vinai/bertweet-base. Accessed: 2025-11-10.

[Suhasini and Srinivasu, 2020] Suhasini, M. and Srinivasu, B. (2020). Emotion detection framework for twitter data using supervised classifiers. In *Data Engineering and Communication Technology*, pages 565–576. Springer, Singapore.

[Vateekul and Koomsubha, 2016] Vateekul, P. and Koomsubha, T. (2016). A study of sentiment analysis using deep learning techniques on thai twitter data. In *Proceedings of the 2016 13th International Joint Conference on Computer Science and Software Engineering (JCSSE)*, Khon Kaen, Thailand. IEEE.