Do Emojis Matter? Exploring the Role of Emojis in Sentiment Classification with Small and Large Language Models

Arbel Askayo, Nir Manor

Abstract

Emojis are ubiquitous in social media communication, yet many NLP pipelines routinely discard them during preprocessing. We investigate whether retaining emojis meaningfully improves multi-label emotion classification across different model architectures. To isolate emoji effects, we construct an English-only corpus of emoji-bearing texts by merging established datasets and harmonizing annotations to a unified seven-label schema. We evaluate three model families under controlled conditions by comparing original text versus text with Unicode emojis removed, using identical splits and hyperparameters: TF-IDF with Logistic Regression, fine-tuned RoBERTa-Twitter, and GPT-5 with zero-/few-shot prompting. Across all models, retaining emojis consistently improves classification performance, with the most pronounced benefits for compact, task-specific models. Fine-tuned RoBERTa achieves the strongest overall results, outperforming even large language models when emojis are preserved. A perturbation-based analysis reveals that individual emojis can substantially shift emotion probabilities with emotion-dependent polarity. The same emoji may strongly promote one emotion while suppressing another. These findings demonstrate that removing emojis discards meaningful discriminative signal rather than noise. Treating emojis as first-class tokens enables lightweight models to compete with much larger systems and offers a simple, effective design choice for affective computing pipelines.

1 Introduction

Emotion classification in social media is a core NLP task with practical impact in monitoring public discourse, product feedback, and wellbeing analytics. Emojis are pervasive in these platforms and often function as affective signals for both humans and models. Contemporary resources and studies continue to treat emojis as first-class meaning carriers and active modeling targets, and the emoji inventory itself keeps expanding, underscoring sustained, high usage in everyday communication.

Despite this prominence, many pipelines still handle emojis inconsistently during preprocessing and tokenization. however, whether retaining emojis during training and inference actually improves multi-label emotion classification performance remains an empirical question. We study this question broadly, asking: *to what extent do emojis contribute to emotion classification quality across small and large models when all other factors are held constant?*

We construct an English-only corpus focused on emoji-bearing messages by merging and filtering established sources: the English subset of EmoEvent, Reddit comments from GoEmotions, and the SemEval-2018 Task 1 E-c and TweetEval emotion track. To enable consistent supervision, we aggregate labels to six basic emotions plus *other* (*anger, disgust, fear, joy, sadness, surprise, other*); we discard *optimism* from TweetEval for compatibility. We split data 80/10/10 into train, validation, and test, and we use identical item IDs across all experimental conditions. We define two aligned conditions: **with emojis** (original text) and **without emojis** (only Unicode emoji code points removed; ASCII emoticons are preserved).

Our model suite spans small, classical, and large settings. For a small model, we fine-tune a Twitter-adapted RoBERTa classifier with a sigmoid multi-label head. As a classical baseline, we use TF–IDF with one-vs-rest Logistic Regression that naturally treats emojis as tokens, which yields interpretable features and fast training. For a large-model reference, we evaluate a zero-shot GPT-5 classifier using the same seven-label schema on the same test items under both with/without-emoji conditions. Because exact match is stringent in multi-label settings, we emphasize Jaccard similarity (intersection-over-union) alongside Accuracy, Macro-/Micro-F1, and per-class F1.

We also provide a lightweight interpretability view via frequency co-occurrence: the classifier predicts labels on sampled tweets, we count which emojis appear in those tweets, and we report how often each emoji co-occurs with each predicted class. This yields a compact inventory of class-associated emojis.

Our contributions are:

- A controlled, English-only, emoji-bearing corpus assembled from EmoEvent, GoEmotions, and SemEval-2018 E-c/TweetEval with a unified seven-label schema and consistent splits.
- A paired evaluation that isolates the effect of emojis by training and testing models on text with and without emojis, holding data, splits, and hyperparameters fixed; models include a RoBERTa-Twitter classifier, a TF–IDF + Logistic Regression baseline, and a GPT-5 zero-shot reference.
- Evidence that retaining emojis affects multi-label performance. We additionally report Emoji contribution analysis.

2 Methodology

We evaluate the contribution of emojis to multi-label emotion classification by holding data, item IDs, and evaluation protocol fixed while toggling the presence of Unicode emojis in the text. We compare three families of models: (i) a small, fine-tuned RoBERTa classifier, (ii) a classical TF–IDF + Logistic Regression baseline, and (iii) a large-model reference based on GPT-5 via API. we use an identical seven-label schema {*anger, disgust, fear, joy, sadness, surprise, other*} and report multi-label metrics on the same test items under the two text conditions.

2.1 Datasets and Label Harmonization

We merge English subsets from EmoEvent, GoEmotions (Reddit), the SemEval-2018 Task 1 E-c and TweetEval track. We harmonize annotations to six basic emotions plus *other* and drop *optimism* from TweetEval for consistency. We keep only items that contain at least one Unicode emoji for the *emoji-bearing* source pool used in both conditions (with and without emojis), ensuring the same items and IDs in every experiment. We split the data into **80/10/10** train/validation/test partitions once and reuse these splits across all models and conditions.

Emotion	Count	%
joy	1,769	46.1%
other	865	22.6%
sadness	684	17.8%
anger	613	16.0%
disgust	346	9.0%
surprise	189	4.9%
fear	174	4.5%

Table 1: Emotion distribution.

Source File	Rows	%
emoevent_en_only_emojis	1,371	35.8%
goemotions_emoji_aggregated	843	22.0%
2018-E-c-En-train_only_emojis_basic	822	21.4%
TweetEval	798	20.8%
Total	**3,834**	**100%**

Table 2: Data sources merged and their contributions.

Note. The counts above are computed on *emoji-bearing subsets* of each listed dataset: we retained only records that contain at least one emoji. Because the original corpora mostly consist of texts without emojis, those non-emoji records were excluded from our merged dataset.

Two aligned text conditions. We define:

- **With Emojis**: original text.
- **Without Emojis**: we remove only Unicode code points with the Unicode "Emoji" property (UAX #51); ASCII emoticons (e.g., `:-)`, `<3`) are preserved. All remaining tokens, including dataset placeholders (e.g., `HASHTAG`) where present in sources, are left unchanged.

2.2 Task Formulation and Metrics

Each instance take one or more labels from the seven-label set. For learned classifiers, we optimize **binary cross-entropy** with independent sigmoid outputs.

We report metrics standard for multi-label classification: **micro-/macro-F1**, **Jaccard** (intersection-over-union) computed over predicted vs. gold label sets, **Exact Match Ratio** (subset accuracy), **Hamming Loss**, and **per-class F1**. Model selection is based on **macro-F1** on the validation set.

2.3 Small Model: RoBERTa-Twitter Fine-tuning

We fine-tune `cardiffnlp/twitter-roberta-base` with a multi-label classification head (sigmoid). Training uses **AdamW** with a linear schedule and warmup. We run the identical protocol in both text conditions.

Hyperparameters.

- learning_rate: 1×10^{-5}, batch_size: 16, warmup_ratio: 0.1, weight_decay: 0.01, gradient_accumulation_steps: 1.
- fp16: True; dataloader_num_workers: 2; random seed: 42.

Epoch sweep. To quantify data efficiency and stability, we sweep num_epochs $\in \{0, 1, 3, 10\}$. The *0-epoch* setting evaluates the model immediately after initialization (no parameter updates), while the other settings perform task-specific fine-tuning for the specified number of epochs. For every setting, the reported test score comes from the checkpoint with the highest validation macro-F1.

2.4 Classical Baseline: TF–IDF + One-vs-Rest Logistic Regression

We train a sparse linear baseline that naturally treats emojis as tokens and provides interpretable features. The pipeline consists of a TF–IDF vectorizer followed by a one-vs-rest Logistic Regression classifier operating on binarized labels.

Vectorizer.

- max_features: 10,000; ngram_range: $(1, 2)$; stop_words: 'english'.
- lowercase: False (preserve emoji and case distinctions); strip_accents: None.
- tokenizer: custom, emoji-aware (treats each Unicode emoji/emoji sequence as a standalone token); token_pattern: None.
- min_df: 2; max_df: 0.95.
- Text length capped at 256 tokens.

Classifier and tuning. We use one-vs-rest Logistic Regression with class_weight='balanced' and random_state=42. We perform **GridSearchCV** with 3-fold CV on the training set, optimizing f1_macro. The search space is:

$$C \in \{0.1, 1.0, 10.0\},$$
$$\texttt{solver} \in \{\texttt{liblinear}, \texttt{lbfgs}\},$$
$$\texttt{max\_iter} = 1000.$$

We refit on the full training set using the best hyperparameters and apply a threshold τ=0.5 to convert probabilities to labels.

2.5 Large-Model Reference: GPT-5 (API)

We evaluate a strong instruction-following LLM (GPT-5) under two prompting regimes in both text conditions.

Zero-shot. We instruct the model to perform multi-label classification strictly within our seven-label inventory and to return a JSON object:

- The system prompt defines the role, the exact label set, rules for using *other*, and the required JSON schema, emphasizing "return *only* the JSON".
- The user message supplies the batch as a JSON array of {id, text} items. The id field is preserved verbatim to enable alignment with true labels.

Few-shot. We additionally evaluate a template that prepends three labeled examples. We maintain two variants:

- **With Emojis**: examples contain emojis.
- **Without Emojis**: the same examples with emojis removed.

In both cases, the rest of the batch appears unlabeled and the model must output the same JSON schema as in zero-shot. Post-processing enforces the label inventory (unknown labels are dropped) and preserves item ordering and IDs.

2.6 Emoji Contribution Analysis (RoBERTa, 10 epochs)

We quantify the *causal* influence of individual emojis on the model's emotion probabilities using a perturbation-based ablation analysis applied to the best validation checkpoint of our fine-tuned cardiffnlp/twitter-roberta-base trained for 10 epochs (with emojis). The analysis is run on the test set (384 items).

Preliminaries. Let x denote a tokenized input text and $E(x) = \{e_1, \ldots, e_m\}$ the multiset of emoji *occurrences* in x (multi-codepoint emoji sequences are treated as single units). Let $p_c(x) \in [0, 1]$ be the model's sigmoid probability for emotion $c \in \{$*anger, disgust, fear, joy, other, sadness, surprise*$\}$. Emoji extraction and removal are implemented with an emoji-aware parser that preserves text structure when deleting an individual emoji occurrence.

Ablation procedure. For each example x and each emoji occurrence $e \in E(x)$, we construct an ablated variant $x\backslash e$ by removing exactly that occurrence and leaving all other content unchanged. The *local contribution* of e to emotion c on x is

$$\Delta_c(e; x) \;=\; p_c(x) \;-\; p_c(x\backslash e).$$

Positive values indicate that e *supports* emotion c (probability increases when present), while negative values indicate a *suppressive* effect. We aggregate local contributions corpus-wide to obtain, for each (emoji, emotion) pair, the mean $\mu_c(e)$, standard deviation $\sigma_c(e)$, and occurrence count $n_c(e)$. We also compute emoji-level summaries by averaging across emotions and produce top-k lists per emotion; the corresponding leaderboards and descriptive statistics are reported in §3.

Implementation. The analysis is implemented in an `EmojiContributionAnalyzer` that loads the 10-epoch checkpoint, caches baseline forward passes, performs per-emoji ablations, and exports (i) per-example JSON records and (ii) global emoji–emotion tables for visualization.

Outputs. We report: (a) corpus-level tables of $\mu_c(e)$ with $n_c(e)$; (b) per-emotion top-k contributors; and (c) distributional summaries of positive vs. negative contributions to characterize how often emojis increase vs. decrease class probabilities. Full result tables for the test set (381 emoji-bearing items) appear in §3.

Limitations. Ablations assume local independence (removing one emoji at a time may miss emoji–emoji interactions); estimates may vary with architecture and dataset composition; and small-n emojis can yield high-variance statistics. We therefore accompany leaderboards with counts ($n_c(e)$) and confidence descriptors and recommend interpreting low-frequency effects with caution.

2.7 Implementation and Reproducibility

Fine-tuning uses the HuggingFace Transformers training loop (AdamW with linear warmup/decay) in mixed precision (`fp16`). We evaluate and checkpoint once per epoch and load the best validation macro-F1 model before final test evaluation. All experiments use the same random seed (42), identical item IDs and splits, and identical pre-/post-processing across the two text conditions to isolate the effect of emoji removal.

3 experiments and Results

3.1 Classical Baseline: TF–IDF + Logistic Regression

Setup. We train a one-vs-rest Logistic Regression on TF–IDF features with an *emoji-aware* tokenizer that preserves Unicode emojis as standalone tokens (10k uni/bi-grams; `lowercase=False`; `min_df=2`; `max_df=0.95`). We tune $C \in \{0.1, 1, 10\}$ and `solver`$\in \{$`liblinear`, `lbfgs`$\}$ with 3-fold CV optimizing macro-F1, then refit on train and evaluate on the fixed test split. No post-hoc fallback is applied to predictions.

Metric (Test)	With Emojis	No Emojis
Exact Match Ratio ↑	0.3125	0.2969
Hamming Loss ↓	0.1611	0.1741
Jaccard (IoU) ↑	0.3677	0.3297
$F1_{micro}$ ↑	0.5994	0.5385
$F1_{macro}$ ↑	0.5173	0.4816

Table 3: Logistic Regression baseline (emoji-aware TF–IDF). Scores on the fixed test set; emojis improve all aggregate metrics.

Overall results. Retaining emojis yields consistent gains across multi-label metrics (Table 3): Jaccard $+0.0380$ (11.5% rel.), micro-F1 $+0.0609$ (11.3%), macro-F1 $+0.0357$ (7.4%), EMR $+0.0156$ (5.3%), and Hamming Loss -0.0130 (better). Best hyperparameters differ by condition (**with emojis**: C=1.0, `lbfgs`; **without**: C=0.1, `liblinear`).

Takeaway. Even with a sparse linear model, explicitly modeling emojis as features provides measurable gains in multi-label quality. This establishes a text+emoji baseline that our RoBERTa and GPT-5 results will build upon.

3.2 RoBERTa-Twitter Fine-tuning: Effect of Emojis and Training Duration

We evaluate `cardiffnlp/twitter-roberta-base` under two aligned text conditions (*with emojis* vs. *without emojis*) and three training budgets (1, 3, 10 epochs). We report test-set metrics with Jaccard as the primary score (multi-label IoU), alongside Micro-/Macro-F1, Exact Match Ratio (EMR), and Hamming Loss. All runs use the same splits and the same label decision rule with the no-empty fallback (see §2).

Setup. We adopt a three-stage workflow (training → validation/selection → test) implemented with `transformers`. The classifier (`cardiffnlp/twitter-roberta-base`) is fine-tuned on the training split with AdamW and a linear warmup/decay schedule in mixed precision (`fp16`), using dynamic padding and `max_length`$=256$. At the end of each epoch we evaluate on the validation split and checkpoint; the checkpoint with the highest validation macro-F1 is automatically

restored (`load_best_model_at_end=True`). To keep the search lightweight, all hyperparameters are fixed (see Appendix), and we sweep only the number of epochs $\{0, 1, 3, 10\}$, yielding model selection by validation macro-F1 without any test-set reuse. For reproducibility, the model and tokenizer are pinned to the Hugging Face revision `cbb417e9647b51504caf68cbe1af6bbf56da06b7`.

Test evaluation. After selection, we run a single evaluation on the held-out test split and report Exact Match Ratio (EMR), Jaccard (IoU), micro-/macro-F1, and Hamming Loss. Sigmoid outputs are thresholded at 0.5 per label; if the predicted set is empty, we assign the highest-probability label (or *other* when all confidences are very low). The identical pipeline is used for both text conditions (with and without emojis).

Headline findings. (1) Emojis provide clear gains across training budgets, with the largest improvement when data/time are limited (1 epoch). (2) Additional training improves performance for both conditions; gains taper from 3 to 10 epochs. (3) At 10 epochs, keeping emojis yields the strongest test Jaccard and Micro-F1 overall.

Effect of emojis (paired by epoch). At **1 epoch**, emojis deliver large gains: Jaccard improves by $+0.1172$ ($+28.1\%$ rel.), Micro-F1 by $+0.1333$ ($+32.6\%$), EMR by $+0.0964$ ($+25.5\%$), and Hamming Loss decreases by 0.0431 (-22.5%). At **3 epochs**, gains persist but are smaller (Jaccard $+0.0171$; Micro-F1 $+0.0167$). At **10 epochs**, the emoji advantage is still measurable (Jaccard $+0.0278$; Micro-F1 $+0.0311$) and yields the strongest overall scores (Table 4).

Effect of training duration (within each condition). From $1\rightarrow3$ epochs, performance jumps for both conditions; from $3\rightarrow10$ epochs, gains taper: with emojis, Jaccard $+0.0976$ then $+0.0142$; without emojis, Jaccard $+0.1977$ then $+0.0035$. Micro-F1 and Macro-F1 follow the same pattern. This suggests most learning occurs by epoch 3, with smaller refinements by epoch 10.

Takeaways. Keeping emojis yields consistent improvements in multi-label quality (Jaccard, Micro-F1, Hamming), especially under small training budgets; gains remain at 10 epochs and are particularly visible for *sadness*, *joy*, *other*, and *disgust*.

3.3 GPT-5: Zero-shot vs. Few-shot and Emoji Impact

Setup. We evaluate GPT-5 with the prompts in §2.5 under four configurations: zero-shot/few-shot $\times$ with/without emojis. We report Exact Match (EMR), Jaccard (IoU), Micro-F1, and Macro-F1 on the same test split and label schema as our other models.

Overall results. *Zero-shot with emojis* is the strongest configuration across EMR, Jaccard, and Micro-F1 (Table 7). Relative to zero-shot without emojis, emojis yield +0.0254 Jaccard (+4.47% rel.), +0.0282 Micro-F1 (+4.78%), and +0.0104 EMR (+2.22%); Macro-F1 is comparable (+0.0046). In few-shot, emojis also help Micro-F1 and Jaccard (+0.024 and +0.0218), though Macro-F1 is slightly lower (-0.0129). Differences between zero-shot and few-shot are small (e.g., with emojis: Micro-F1 0.6172 vs. 0.6093), and **none of the pairwise comparisons are statistically significant** (all $p>0.05$; negligible effect sizes). Taken together, GPT-5 performs robustly across setups, with a modest, consistent benefit from retaining emojis and a slight edge for the simpler zero-shot regime.

3.4 Emoji Contribution Analysis: Quantifying Individual Emoji Influence

Setup. We apply the perturbation-based ablation analysis described in §2.6 to our best-performing RoBERTa model (10-epoch checkpoint, with emojis). The analysis covers 381 emoji-bearing test samples (99.2% of test set), systematically removing individual emoji occurrences and measuring the resulting probability shifts across all seven emotion classes. We aggregate local contributions to derive corpus-wide statistics and identify emotion-specific emoji influences.

Overall emoji influence patterns. Table 5 presents the ten most influential emojis by average absolute impact across all emotions. Musical note shows the highest average influence (0.1724), though with limited occurrences (n=7). More frequent emojis like unamused face (n=28) and weary face (n=21) demonstrate consistent moderate influence (0.1009 and 0.1211 respectively). The influence distribution is nearly balanced: 622 positive contributions vs. 694 negative contributions (ratio 0.90:1), indicating emojis both amplify and suppress different emotions rather than uniformly boosting predictions.

Condition	Epochs	EMR	Hamm.	Jacc.	$F1_{micro}$	$F1_{macro}$
With emojis	1	0.4740	0.1481	0.5343	0.5425	0.2666
With emojis	3	0.5573	0.1164	0.6319	0.6423	0.3856
With emojis	10	**0.5651**	**0.1105**	**0.6461**	**0.6629**	**0.4239**
No emojis	1	0.3776	0.1912	0.4171	0.4092	0.0905
No emojis	3	0.5417	0.1220	0.6148	0.6256	0.3755
No emojis	10	0.5443	0.1205	0.6183	0.6318	0.3947

Table 4: Test results for RoBERTa-Twitter across epochs and text conditions. Best scores per column are in bold.

Emoji	Avg. Impact	Count
musical note	0.1724	7
anguished face	0.1508	14
skull	0.1462	7
frowning face	0.1269	21
disappointed face	0.1214	14
weary face	0.1211	21
unamused face	0.1009	28
grinning cat	0.0898	7
winking face	0.0788	28
index pointing up	0.0725	7

Table 5: Top 10 most influential emojis by average absolute contribution across all emotion classes. Impact measures mean absolute probability change when emoji is removed. We couldn't insert real emojis to LaTeX

Emotion-specific emoji contributions. Emojis exhibit strong class-specific effects (Table 6). For *sadness*, frowning face dramatically increases probability by $+0.4434$ (strongest single effect observed), while weary face adds $+0.2821$. Conversely, for *joy*, the same frowning face *decreases* probability by -0.3048, demonstrating opposing polarity across emotions. winking face boosts *joy* by $+0.2657$ but suppresses *other* by -0.1614. For *anger*, unamused face (+0.1943) and pouting face (+0.1320) serve as strong positive indicators. The *other* class shows predominantly negative contributions from emotion-specific emojis, suggesting it captures neutral/ambiguous cases as intended.

Magnitude of emoji effects. Emoji contributions vary substantially by emotion class: *joy* shows the highest average influence magnitude (0.0414), followed by *other* (0.0302) and *sadness* (0.0282), while *surprise* (0.0046) and *fear* (0.0071) exhibit minimal emoji sensitivity. Individual emoji effects often exceed 0.1 probability shift, which represent a substantial effect given the multi-label setting. The strongest observed contribution is (frowning face) increasing *sadness* probability by +0.4434, while several emojis show effects in the 0.2-0.3 range (e.g., boosting *joy* by +0.2657, increasing *sadness* by +0.2821). Even moderate contributors like (unamused face) demonstrate meaningful impact (+0.1943 for *anger*).

Takeaways. (i) Individual emojis causally influence emotion predictions with effect sizes often exceeding 0.1 probability; (ii) emoji polarity is emotion-dependent. the same emoji can strongly promote one emotion while suppressing another; (iii) certain emotion classes (*joy*, *sadness*) show substantially higher emoji sensitivity than others (*fear*, *surprise*).

4 Conclusion

We investigated whether emojis improve multi-label emotion classification across classical, small, and large language models. Our controlled experiments, using identical data splits and preprocessing conditions (with/without emojis), provide clear evidence that emojis consistently enhance classification performance across all model architectures.

For our classical baseline, TF-IDF + Logistic Regression improved from Jaccard 0.3297 to 0.3677 (+11.5%) and Micro-F1 0.5385 to 0.5994 (+11.3%) when emojis were retained. The fine-tuned RoBERTa model achieved our strongest results with emojis, reaching Jaccard 0.6461 and Micro-F1 0.6629 while surpassing its no-emoji counterpart (0.6183/0.6318) and even outperforming GPT-5. This demonstrates that smaller, task-specific models can remain highly competitive when emoji signals are preserved.

GPT-5's zero-shot configuration with emojis yielded its best performance (Jaccard 0.5940, Micro-F1 0.6172), with few-shot learning providing minimal additional benefit or slight degradation. The near-identical performance between zero-shot and few-shot (no statistically significant differences) suggests that for well-defined emotion taxonomies, LLMs' pretrained knowledge already encodes sufficient structure, making demonstration-based learning less impactful than expected.

Our emoji contribution analysis revealed the mechanisms behind these improvements. Individual emojis can dramatically shift emotion proba-

Emotion	Top Positive Contributors	Contrib.	Top Negative Contributors	Contrib.	n
anger	unamused face	+0.1943	weary face	−0.0829	4/6
	pouting face	+0.1320	confused face	−0.0790	8/5
	angry face	+0.1222			3
disgust	tired face	+0.1057	tears of joy	−0.0035	3/47
	unamused face	+0.0941	smiling eyes	−0.0044	4/16
	see-no-evil monkey	+0.0334			5
fear	tired face	+0.0865	beaming face	−0.0304	3/3
	confused face	+0.0458	smiling eyes	−0.0290	5/16
	see-no-evil monkey	+0.0450			5
joy	winking face	+0.2657	frowning face	−0.3048	4/3
	grinning with sweat	+0.1946	tired face	−0.2919	8/3
			unamused face	−0.2347	/4
sadness	frowning face	+0.4434	tears of joy	−0.0585	3/47
	tired face	+0.2821	beaming face	−0.0547	3/3
	crying face	+0.1984			7
	see-no-evil monkey	+0.1952			5
	weary face	+0.1892			6
surprise	tired face	+0.0256	angry face	−0.0198	3/3
	eyes	+0.0228	tears of joy	−0.0035	7/47
	face screaming	+0.0215			7
	flushed face	+0.0190			8
other			angry face	−0.1692	3
			winking face	−0.1614	4
			tears of joy	−0.1212	47
			pouting face	−0.1095	8
			face screaming	−0.1081	7

Table 6: Emotion-specific emoji contributions. Values represent mean probability change when emoji is removed. Positive values indicate the emoji increases that emotion's probability; negative values indicate suppression. n shows occurrence count in labeled samples.

bilities by up to +0.4434 for sadness (frowning face emoji), exhibiting strong emotion-dependent polarity. The same emoji often promotes one emotion while suppressing another (e.g., frowning face: +0.4434 for sadness, -0.3048 for joy), demonstrating that emojis serve as discriminative features rather than simple sentiment amplifiers.

These findings have immediate practical implications: the common preprocessing step of removing emojis can discard signal comparable to deleting key content words. Our results show that a 125M-parameter RoBERTa with emoji features can match or exceed models orders of magnitude larger. For practitioners and researchers in affective computing, this work provides quantitative evidence that emojis should be treated as first-class features in emotion analysis pipelines, not as noise to be filtered out.

Configuration	EMR	Jacc.	$F1_{micro}$	$F1_{macro}$
Zero-shot + Emojis	0.4792	**0.5940**	**0.6172**	0.5101
Few-shot + Emojis	0.4714	0.5866	0.6093	0.5043
Zero-shot (No Emojis)	0.4688	0.5686	0.5890	0.5077
Few-shot (No Emojis)	0.4661	0.5648	0.5858	**0.5173**

Table 7: GPT-5 performance on the fixed test set under zero-/few-shot and with/without emojis.

Method	Prompting / Fit	Emojis	EMR ↑	Jaccard ↑	$F1_{micro}$ ↑	$F1_{macro}$ ↑	Hamm. ↓
Logistic Regression	—	With	0.3125	0.3677	0.5994	**0.5173**	0.1611
Logistic Regression	—	Without	0.2969	0.3297	0.5385	0.4816	0.1741
RoBERTa-Twitter	10-epoch fine-tune	With	**0.5651**	**0.6461**	**0.6629**	0.4239	**0.1105**
RoBERTa-Twitter	10-epoch fine-tune	Without	0.5443	0.6183	0.6318	0.3947	0.1205
GPT-5	Zero-shot	With	0.4792	0.5940	0.6172	0.5101	0.1343
GPT-5	Zero-shot	Without	0.4688	0.5686	0.5890	0.5077	0.1417
GPT-5	Few-shot	With	0.4714	0.5866	0.6093	0.5043	0.1369
GPT-5	Few-shot	Without	0.4661	0.5648	0.5858	**0.5173**	0.1436

Table 8: Best values per column are in **bold**. Lower is better for Hamming Loss.

A Appendix

A.1 RoBERTa Training Configuration (Exact Values)

Listing 1: RoBERTa fine-tuning configuration used in all runs (with/without emojis).


```
{
  "huggingface_revision": "cbb417e9647b51504caf6
      8cbe1af6bbf56da06b7",
  "model_name": "cardiffnlp/twitter-roberta-base
      ",
  "data_dir": "FinalData/split with emoji",
  "output_dir": "results/
      emoji_roberta_emotion_local",
  "max_length": 256,
  "learning_rate": 1e-5,
  "batch_size": 16,
  "num_epochs_candidates": [0, 1, 3, 10],
  "warmup_ratio": 0.1,
  "weight_decay": 0.01,
  "save_strategy": "epoch",
  "evaluation_strategy": "epoch",
  "save_total_limit": 3,
  "load_best_model_at_end": true,
  "metric_for_best_model": "f1_macro",
  "greater_is_better": true,
  "logging_steps": 10,
  "seed": 42,
  "gradient_accumulation_steps": 1,
  "fp16": true,
  "dataloader_num_workers": 2,
  "loss_fn": "BCEWithLogits",
  "threshold": 0.5,
  "no_empty_fallback": {
    "policy": "argmax_or_other",
    "other_label_index": 4
  },
  "notes": "Same configuration used for both
      with-emoji and no-emoji conditions; only
      the input text differs."
}
```


A.2 Prompt Templates for GPT-5

Zero-shot (System + User)

System

```
You are a careful multi-label emotion classifier.

Classify each input text into zero or more of these labels:
{ anger, disgust, fear, joy, sadness, surprise, other }

Rules:
- A text may receive 1 or several labels.
- Use "other" for emotions not covered by the specific categories.
- Use "other" for neutral or ambiguous emotional content.
- Treat each item independently.
- Preserve every input id exactly.

Return your response as a JSON object with this exact structure:
{
  "items": [
    {"id": "item_id", "labels": ["label1", "label2"]},
    {"id": "item_id", "labels": ["other"]}
  ]
}

IMPORTANT: Return ONLY the JSON object, no additional text or formatting.
```

User

```
Classify the emotions in each text:

{items_json}

Return ONLY a JSON object with the structure shown in the system message.
```

Figure 1: Zero-shot prompt template for GPT-5 (system and user).

Few-shot with emojis (System + Demos + User)

System

```
You are a careful multi-label emotion classifier.

Classify each input text into zero or more of these labels:
{ anger, disgust, fear, joy, sadness, surprise, other }

Rules:
- A text may receive 1 or several labels.
- Use "other" for emotions not covered by the specific categories.
- Use "other" for neutral or ambiguous emotional content.
- Treat each item independently.
- Preserve every input id exactly.

Here are some examples:


Example 1:
Input: {"id": "319", "text": "I remember when Rooney wanted to leave United and the fans threaten to kill the man and bare
      junk... wanna regretting that now? {face-with-tears-of-joy} {face-with-tears-of-joy}"}
Output: {"id": "319", "labels": ["anger", "disgust"]}

Example 2:
Input: {"id": "1557", "text": "Epic battle HASHTAG breathtaking, Arya is awesome! {thumbs-up}"}
Output: {"id": "1557", "labels": ["joy"]}

Example 3:
Input: {"id": "2005", "text": "HASHTAG Glory ,then ashesMen's dreams in a spark transformedBurning hearts unite {rose}{
      folded-hands}"}
Output: {"id": "2005", "labels": ["other"]}


Return your response as a JSON object with this exact structure:
{
  "items": [
    {"id": "item_id", "labels": ["label1", "label2"]},
    {"id": "item_id", "labels": ["other"]}
  ]
}

IMPORTANT: Return ONLY the JSON object, no additional text or formatting.
```

User

```
Classify the emotions in each text:

{items_json}

Return ONLY a JSON object with the structure shown in the system message.
```

Figure 2: Few-shot prompt template with emojis (system, demonstrations, and user). The model got the real emojis but we couldn't insert real emojis to LaTeX

Few-shot without emojis (System + Demos + User)

System

```
You are a careful multi-label emotion classifier.

Classify each input text into zero or more of these labels:
{ anger, disgust, fear, joy, sadness, surprise, other }

Rules:
- A text may receive 1 or several labels.
- Use "other" for emotions not covered by the specific categories.
- Use "other" for neutral or ambiguous emotional content.
- Treat each item independently.
- Preserve every input id exactly.

Here are some examples:


Example 1:
Input: {"id": "319", "text": "I remember when Rooney wanted to leave United and the fans threaten to kill the man and bare
      junk... wanna regretting that now?"}
Output: {"id": "319", "labels": ["anger", "disgust"]}

Example 2:
Input: {"id": "1557", "text": "Epic battle HASHTAG breathtaking, Arya is awesome!"}
Output: {"id": "1557", "labels": ["joy"]}

Example 3:
Input: {"id": "2005", "text": "HASHTAG Glory ,then ashesMen's dreams in a spark transformedBurning hearts unite"}
Output: {"id": "2005", "labels": ["other"]}


Return your response as a JSON object with this exact structure:
{
  "items": [
    {"id": "item_id", "labels": ["label1", "label2"]},
    {"id": "item_id", "labels": ["other"]}
  ]
}

IMPORTANT: Return ONLY the JSON object, no additional text or formatting.
```

User

```
Classify the emotions in each text:

{items_json}

Return ONLY a JSON object with the structure shown in the system message.
```

Figure 3: Few-shot prompt template without emojis (system, demonstrations, and user).