Journal of Behavioral Data Science, 2026, *6* (1), 152–164.
DOI: https://doi.org/10.35566/jbds/tonghl

EmojiSentR: An R Package for Integrated Text and Emoji Sentiment Analysis

Logan Hanson[0009−0008−4669−0162], Elias Lahrim, and Xin Tong*[0000−0003−3050−1554]

Department of Psychology, University of Virginia, Charlottesville, VA 22904, USA
loganh11702@gmail.com; elahrim@gmail.com; xt8b@virginia.edu

Abstract.

Although emojis are used by 92% of the world's online population, appear in over one quarter of social media posts, and carry affect beyond words, most software packages for sentiment analysis ignore emoji semantics. Our newly developed R package, `EmojiSentR`, addresses this gap by merging emoji valence with word-level sentiment in a tidy R workflow. The package bundles an internal lexicon derived from the Novak-1200 dataset (1,200 glyphs × 8 emotions) and a pipeline function, `sentiment_analysis()`, that (1) extracts emojis, (2) cleans residual text, (3) scores each channel separately, (4) leverages VADER's (Valence Aware Dictionary and sEntiment Reasoner) simple negation, and (5) returns a weighted composite score. Furthermore, by adding Unicode-cleanup and Poppler-based PDF-to-text utilities, the package also broadens data-source coverage. In an illustrative example, on a corpus of 17,996 English tweets, `EmojiSentR` changed the predicted polarity in 20.7% of messages while adding only 769.4 ms of computation time per 1,000 posts. For instance, polarity reversals appeared in sarcastic laughter ("Great… 😂") and uncertainty cues ("Ok 😬"). The `EmojiSentR` package makes it simple and convenient to treat emojis as first-class sentiment carriers within R. Its modular design supports user-tunable weights, transparent lexicon updates, and forthcoming multilingual extensions, offering researchers a reproducible tool for analyzing emoji-rich text as well as PDF-based data sources.

Keywords: Emoji · Sentiment analysis · Text mining · R package · PDF extraction

1 Introduction

Digitally mediated communication has made text one of the most abundant forms of data in research (Aggarwal, 2015). News articles, social media posts, policy documents, customer reviews, electronic health records, psychotherapy

transcripts, scientific literature, and open-ended survey responses capture attitudes, behaviors, and events at scale and in near real time. Harnessing these sources enables researchers to track public opinion, monitor health and education outcomes, study misinformation, evaluate programs, and generate hypotheses that complement traditional quantitative measures (e.g., Humphreys & Wang, 2018; Khaiser et al., 2023; Kozik et al., 2022; Machová et al., 2023; Thakur, 2023; Westgate et al., 2015).

Text analysis, or text mining, is the process of converting unstructured texts into structured information (e.g., numerical data) so it can be searched, summarized, modeled, and used for inference or prediction. Text analysis methods range from simple lexicons to modern deep learning (e.g., Hotho et al., 2005; Minaee et al., 2021; Wankhade et al., 2022). As an important category of text analysis, sentiment and emotion mining has been broadly applied across many fields to infer affective tone and evaluative stance expressed in text (Itani et al., 2017; Liu et al., 2025). Sentiment typically measures polarity and intensity (positive, negative, or neutral), while emotion targets specific feelings (e.g., happiness, anger, sadness). Despite this difference, one key facet of sentiment and emotion mining is polarity detection, which employs deep learning-based, machine learning-based, or lexicon-based approaches to discern the polarities in a text, as highlighted in Nandwani and Verma (2021). Other categories of text analysis include topic modeling, which can be used to detect latent topics in text data (Blei & Lafferty, 2007; Blei et al., 2003); LLM-based classification, which uses a prompted large language model to label or categorize texts without task-specific training (Hassani et al., 2025); etc.

Dedicated toolkits have made the text analysis pipeline virtually turnkey. In R software, packages such as `tm`, `quanteda`, and `tidytext` provide tools for corpus management, tokenization, and document-feature matrix construction (Gagolewski, 2024; Wickham, 2024); `sentimentr`, `syuzhet`, and `VADER` implement lexicon- and rule-based polarity scoring (Hutto & Gilbert, 2014; Jockers, 2020; Rinker, 2022); and topic-model frameworks such as `stm` (Roberts et al., 2019) and `text2vec` (Selivanov et al., 2024) enable estimation of latent thematic structures. More recently, R interfaces to large language model APIs have added embedding extraction and generative capabilities. In short, analysts can move from messy text to predictive or explanatory models with only a few lines of code.

Yet these workflows still treat emojis (now a core element of online speech) as either noise to be stripped or crude word-like tokens. They are anything but fringe: Emojipedia reported that more than 26% of all tweets in 2023 contained at least one emoji (Broni, 2023) and Adobe's global survey found that 92% of internet users rely on emojis to communicate across language barriers (Adobe Inc., 2022). Despite this prevalence, most text analysis toolkits (e.g., R sentiment libraries) ignore emoji semantics or rely on small, static lookup tables that lack polarity scores and context handling. Analyses that omit pictographic affect therefore risk under-estimating emotion, misclassifying polarity, and missing subtle cues such as sarcasm conveyed by 🤨 or the tonal shift from 😊 to 😬.

To fill this gap, we focus on the lexicon-based sentiment analysis, which is simple but commonly used in practice, develop and introduce `EmojiSentR`. `EmojiSentR` is an R package that fuses a purpose-built emoji-sentiment lexicon with conventional text-sentiment engines, yielding a single weighted score that captures both modalities. The package (i) expands the publicly available Novak-1200 ratings into an internal dataset; (ii) offers pipe-friendly functions for extracting, scoring, and context-adjusting emojis; (iii) provides Unicode-cleanup and Poppler-based PDF-to-text utilities; and (iv) blends emoji and textual sentiment through user-tunable weights, enabling richer analyses of social-media and other text-form communication.

The remainder of this article is organized as follows. We first review existing work in the literature and discuss the rationale for `EmojiSentR`. The next section explains how the emoji lexicon was derived, how text- and emoji-level polarity are combined, and how the package is installed and invoked from R. Then we present a brief demonstration on a small tweet corpus, reporting the proportion of polarity flips and the additional run-time required. At the end, we interpret these findings, outline practical applications, recognize current limitations, and sketch directions for future expansion of the lexicon and software.

2 Related Work and Rationale for `EmojiSentR`

This section explains why emojis matter for sentiment analysis, outlines the limits of text-only approaches, summarizes prior emoji research, surveys relevant R tooling, and closes with the specific gap that our package fills.

2.1 Emojis

An emoji is a standardized pictographic character (e.g., 😂, 😍, or 🚀) encoded in Unicode and displayed on digital devices to convey emotion, objects, concepts, or actions within text. Originating from Japanese mobile messaging in the late 1990s, emojis are now supported across smartphones, computers, and web platforms, and have become a routine part of digital communication, allowing writers to convey tone and visual nuance (Broni, 2023). Key benefits of using emojis include simplifying communication by conveying emotion quickly, solving miscommunication caused by ambiguous messaging, and fostering a greater sense of connection in digital interactions. For example, although "This is fine" is technically an affirmative response, it can also serve as a signal of the sender's dissatisfaction. An added emoji can easily clarify the intended tone, e.g., "This is fine 😊" conveys genuine approval, whereas "This is fine 🤨" reveals underlying frustration.

Therefore, when sentiment analyses ignore emojis, polarity estimates can be inaccurate or biased, e.g., missing sarcasm flagged by an emoji or positive reinforcement added via emoji intensifiers. Because downstream tasks (e.g., monitoring attitudes, auditing content) depend on accurate affect estimation, incorporating emoji signals is practically important rather than decorative.

2.2 Limits of text-only sentiment analysis

Previous studies have shown that emojis encode sentiment and can improve affect modeling when integrated appropriately (Boia et al., 2013; Kralj Novak et al., 2015). However, commonly used lexicon-based tools such as `VADER`, `syuzhet`, and `sentimentr` principally score tokens from plain text (Hutto & Gilbert, 2014; Jockers, 2020; Rinker, 2022). While these methods provide fast, interpretable baselines, they may struggle when emojis flip or amplify sentiment, especially when negation scopes across text-emoji mixes (e.g., "not bad 😍"), or when tone is conveyed primarily by pictographs. In such cases, the text-only pipelines can under- or over-estimate polarity. These open issues motivate practical tooling that treats emojis as first-class affective signals rather than post-hoc decorations or annotations.

Within the R software, a robust suite of libraries already supports many core text-mining tasks. Widely used R packages, such as `stringi`, `stringr`, `textclean`, `utf8`, and `text2vec` support tokenization, string handling, and sentiment scoring (Gagolewski, 2024; Rinker, 2023; Selivanov et al., 2024; Urbanek, 2023; Wickham, 2024). Topic modeling and downstream exploration are often implemented with `stm` (Roberts et al., 2019). For emojis, packages such as `emoji`, `emo`, and `emojifont` expose Unicode metadata, aliases, and glyph support (Rudis & Robinson, 2024; Wickham et al., 2022; Yu, 2022). Yet, despite this rich ecosystem, there is still no tidy, end-to-end workflow that simultaneously extracts emojis, scores them with a dedicated emoji lexicon, and fuses text and emoji sentiment with transparent weighting.

2.3 Rationale for `EmojiSentR`

In this study, we propose that implementing sentiment analysis in R based on the following sequence: (i) prepare and normalize text (tokenization, string handling, cleaning), (ii) extract emojis from the same text, (iii) score words with a standard text-lexicon method, (iv) score emojis with an emoji-specific lexicon, and (v) combine both channels with a transparent weighting scheme calibrated to the task. This "text-plus-emoji" storyline is the through line of our work, which is operationalized in the next section.

The developed `EmojiSentR` package bundles the above sequence into a tidy, end-to-end workflow that extracts emojis, scores text and emojis in dedicated channels and returns a tunable weighted composite. The package delivers a lexicon of 1,200 emojis (derived from the Novak-1200 ratings and aligned with Unicode 15) and provides a pipe-friendly function leveraging `VADER's` negation returning a tidy tibble ready for widely used R packages such as `dplyr`, `ggplot2`, or `tidymodels`. By integrating emoji semantics alongside text, rather than treating them as noise, the approach reduces polarity errors in emoji-heavy content while remaining simple enough for routine analysis.

In addition, text data for sentiment analysis are usually ingested into R as plain-text files (.txt or .csv) because these formats are straightforward to

parse and tokenize. But much of the material that researchers use, such as policy briefs, historical documents, court opinions, and annual reports, circulates only as PDFs. Extracting clean text from PDFs in R typically requires juggling separate utilities such as `Poppler`, post-processing errant line breaks, and troubleshooting encoding issues, all of which create friction and discourage analysts from incorporating valuable sources. Our `EmojiSentR` package treats PDF input as first-class data, with which users can directly import PDF files, automatically handle Unicode cleanup, and receive a tidy tibble ready for sentiment analysis. The package makes sentiment workflows as seamless for PDFs as they are for traditional text files, broadening the empirical reach of text mining in R.

3 The `EmojiSentR` package

In this section, we detail how the emoji lexicon is constructed and how the pipeline implements proposed steps with R code.

3.1 Installation, package overview, and intended uses

Installation. The `EmojiSentR` package is hosted on GitHub and can be installed by typing the following code into R:

```
install.packages("devtools")
devtools::install_github("eliaslah/EmojiSentR", subdir
    =  "unicode")
library(EmojiSentR)
```

Users who plan to extract PDFs must also have Poppler (`pdftotext`) installed (Windows: Oschwartz build; macOS: brew install poppler; Linux: apt-get install poppler-utils).

If installation errors occur, users may alternatively download the package source from the GitHub repository, place it in a local directory, and install from source using the appropriate `install.packages()` path. Troubleshooting steps can be performed in parallel with any preferred large language model to identify and resolve dependency or configuration issues. Be sure to check the functions within the scripts on GitHub to leverage `?function_name` for all package applications.

Package overview. Loading the library exposes two public objects:

- `emoji_lexicon`, a tibble that maps emoji code points to sentiment scores, and
- `sentiment_analysis()`, an end-to-end helper that extracts emojis, scores text and emojis separately, and returns a single weighted sentiment index.

The NAMESPACE also exports four utility functions, `remove_unicode`, `mutate_unicode_cols`, `clean_with_poppler`, and `save_pdf_as_text`, which support Unicode cleaning and PDF ingestion. These aspects are pipe-friendly and have minimal additional dependencies.

A quick start example is provided below.

```
txt <- c("I love this 😍",
         "Great... 😂",
         "Not impressed 🤨")
sentiment_analysis(txt)
```

Table 1. Example outputs from `sentiment_analysis()` with emoji rendering.

clean_text	emojis	text_sentiment	emoji_sentiment	combined_sentiment
I love this 😍	😍	0.60	0.75	0.63
Great... 😂	😂	0.05	-0.12	-0.03
Not impressed 🤨	🤨	-0.45	-0.18	-0.41

By default, the weights for text sentiment and emoji sentiment are 0.7 and 0.3, respectively, when calculating the combined sentiment. Namely, `sentiment_analysis(txt)` and `sentiment_analysis(txt, text_weight = 0.7, emoji_weight = 0.3)` give the same output. The defaults are retained for backward compatibility. Custom weights could also be specified. For example, for emoji-heavy text, users can assign a higher weight for emojis:

```
sentiment_analysis(txt, text_weight = 0.5, emoji_weight
    = 0.5)
```

Note that negation handling is often challenging in text analysis, as negation ("not happy," "never went") can flip or dampen the polarity and meaning of entire phrases. With emojis, the problem of negation could be reduced. For example, `sentiment_analysis("not bad")` provided a text sentiment score of 0.431 while `sentiment_analysis("not bad 😊")` provides an emoji sentiment of 0.644 and a combined sentiment score of 0.495.

The `EmojiSentR` package also provides optional Unicode and PDF utilities. To normalize or remove Unicode characters from input strings, use `remove_unicode(txt)`. For PDF documents, use `clean_with_poppler("path/to/file.pdf")` to extract UTF-8 text (requires Poppler's `pdftotext`).

Intended uses. The package is designed for researchers and analysts who need to incorporate emojis into sentiment analysis pipelines without wrestling with ad-hoc preprocessing. It ingests plain text or PDF-extracted content, tokenizes words and emojis in a single step, and enables users to quantify affect in sources where emojis carry crucial tonal cues, including social media dashboards, chat transcripts, psychology studies of diary entries, and classroom demonstrations of feature engineering. In short, it turns mixed text-and-emoji streams into ready-to-model data, enabling more accurate, interpretable sentiment estimates in modern, emoji-rich communication.

In the next subsection, we explain, in plain language, how the emoji lexicon is built and stored, then outline the end-to-end pipeline.

3.2 Emoji lexicon construction

The setup script reads the Novak-1200 emoji table, computes one scalar sentiment per emoji as the simple mean of its eight Plutchik emotion ratings and saves the result as an internal package object (`load_novak_table`). This build happens once at development time, so users do not need to regenerate the lexicon; it loads automatically when the package is attached.

The lexicon is built by the script `sentiment_helpers.R` as shown below.

```
load_novak_table <- function() {

  fn <- system.file("extdata",
            "emoji_sentiment_novak_unicode.rds",
             package = utils::packageName())

  if (fn == "")
    stop("Bundled Novak emoji table not found in
       package.")

  readRDS(fn)
}
```

- Source dataset – Novak-1200 with eight Plutchik emotion ratings per glyph.
- Aggregation rule – arithmetic mean of the eight columns; no scaling or polarity weighting.
- Storage – the resulting tibble is saved from an internal data object `emoji_sentiment_novak_unicode.rds` and loads automatically when the package is attached.

With the lexicon in place, the pipeline can combine emoji-level and text-level scores for any input string.

3.3 Sentiment pipeline

The `sentiment_analysis()` function executes a five-step workflow:

(i) Emoji extraction - We first extract all emoji code-points from each message (e.g., via a Unicode-aware regex (regular expression)) and store them as a per-message list. The regex targets the primary pictograph blocks and is conservative by design; it can be extended if a dataset uses additional ranges. For example,

```
[\U0001F600-\U0001F64F\U0001F300-\U0001F5FF\U0001F680-\
   U0001F6FF]
```

isolates all pictographs in each string.

(ii) Text cleansing – We remove the extracted emojis from the text and apply light normalization (e.g., lowercasing, punctuation/HTML cleanup). Removing

emojis before text scoring prevents double-counting the same affective cue across channels.

(iii) Word-level sentiment – We compute a baseline text polarity using a standard lexicon-based method (with simple handling of negators and intensifiers; e.g., the VADER negation detector ("not", "no", "never") downweights the score). This yields a text-only polarity that we will merge with the emoji channel.

(iv) Emoji-level sentiment – each extracted emoji is matched in emoji_lexicon and averaged.

(v) Weighted fusion – the final score is a convex combination of the two channels:

```
FinalScore = (text_weight*TextSentiment) + (emoji_
    weight*EmojiSentiment)
```

For example, with default weights text_weight = 0.7 and emoji_weight = 0.3, FinalScore = (0.7 * TextSentiment) + (0.3 * EmojiSentiment). If defaults are not desired, weights are user-settable, constrained to [0,1], and must sum to 1. Edge cases are handled gracefully: if a message contains no emojis, the text score is returned; if it is emoji-only, the emoji score is returned; if neither is present, the result is NA.

With the weight tuning parameters, researchers may perform a sensitivity analysis, using weight pairs (e.g., 0.9/0.1, 0.7/0.3, 0.5/0.5, 0.3/0.7, and 0.1/0.9) on a small labeled set, checking the robustness of the analysis, and picking the best-performing pair.

3.4 Unicode-cleaning and PDF-ingestion utilities

These utilities normalize Unicode and preserve emoji code-points so downstream tokenization and scoring behave consistently across platforms. The functions `remove_unicode()` and `mutate_unicode_cols()` strip emojis, pictographs, and control characters from individual strings or entire data-frame columns, enabling analysts to create parallel text-only datasets. The functions `clean_with_poppler()` and `save_pdf_as_text()` wrap the Poppler tool `pdftotext`, converting PDF survey responses or social-media screenshots to UTF-8 text for downstream sentiment analysis.

3.5 Scope, current status, and limitations

Before turning to the results, we briefly note the scope and limitations of the current release. The `EmojiSentR` package uses base-emoji entries; skin-tone and gender modifiers do not change the base emoji's score, and platform-specific renderings are not modeled. The emoji-extraction regex is conservative but extensible if a dataset requires additional Unicode ranges. Negation handling is

intentionally simple for transparency, and the default weights reflect typical text-first sentiment but are user-tunable. Finally, the implementation assumes English-centric tokenization; multilingual or domain-specific slang may require custom tweaks.

The package compiles and passes R CMD check on all major platforms. The core pipeline is fully functional. Planned enhancements (e.g., variant collapsing, and expanded benchmarks) are described in Section 5.3.

4 Applications

This section provides an illustrative example, including a simple weight override to show how tuning the weight affects the final polarity. See the runnable code in section 3.1 for defaults, weight overrides, negation, and Unicode/PDF helpers, for implementation.

We tested `EmojiSentR` on a convenient sample of 17,996 English tweets collected from X/Twitter, of which 15,337 contained at least one emoji. Scoring the entire set required only an additional 769.4 ms of computation time per 1,000 posts (Apple M2, 8 GB RAM). Compared with the text-only baseline (using `sentimentr`), `EmojiSentR` changed the predicted polarity in 20.7% of cases.

To further examine the package's PDF workflow, we compared polarity classifications obtained from plain-text inputs with those obtained after converting the same content from PDF using `clean_with_poppler()`. Across a sampled set of tweets, PDF-derived polarity matched the plain-text result in 63.3% of cases and differed in 36.7% of cases. These findings suggest that the PDF helper is useful for document-based text ingestion, while also indicating that PDF-to-text extraction artifacts can affect downstream sentiment scoring in a subset of cases. Table 2 lists illustrative examples.

Table 2. Illustrative polarity shifts introduced by the emoji channel in plain text and PDF-derived text.

Text / File (excerpt)	text-only polarity	EmojiSentR polarity	effect
Dose 2! (Moderna)... 😊	neutral	positive	flip
@ctraywick ... 😍	negative	positive	flip

On a 300-tweet subset manually labeled as positive, neutral, or negative, the text-only baseline achieved a macro-F1 of 0.427, whereas `EmojiSentR` achieved 0.437. This result indicates that incorporating EmojiSentR can improve classification performance on emoji-rich social media text. In particular, the emoji channel helps capture affective emphasis, stance, and tone cues that are not always reflected in text alone, as illustrated by the examples in Table 2.

5 Discussion

In sum, on a pilot set of 17,996 English tweets, `EmojiSentR` shifted predicted sentiment in 20.7 % of messages, most often correcting cases of sarcasm or emoji-heavy enthusiasm (e.g., Table 1). On a 300-tweet manually labeled subset, the text-only baseline achieved a macro-F1 of 0.427 versus 0.437 for `EmojiSentR`, demonstrating a better performance of `EmojiSentR`. The additional processing time was modest, 769.4 ms per 1,000 posts, making the approach practical for real-time analysis.

5.1 Implications for sentiment analysis

Methodologically, the results confirm that pictographic affect can be integrated into classic lexicon pipelines without resorting to deep-learning infrastructure. The findings also support theories that view emojis as affective amplifiers. For quantitative psychological work, such as monitoring social media expressions, quantifying public stigma, and tracking shifts in sentiment, `EmojiSentR` offers a reproducible way to capture emotional nuance that would otherwise be lost.

Because `EmojiSentR` outputs tidy tibbles, analysts can drop sentiment scores directly into social-media dashboards, mixed-method studies that triangulate survey data with textual affect, or classroom demonstrations of feature engineering in R.

5.2 Limitations

The current lexicon is English-centric and may not capture cultural or temporal drift in emoji meaning. The negation module handles only simple cues ("not", "no", "never") and misses sarcasm or multimodal context (GIFs, images). Validation to date is limited to 17,996 English tweets collected from X/Twitter, so generalizability to long-form or multilingual corpora is uncertain. Finally, the lexicon will need periodic updates as Unicode expands, and a preliminary PDF test revealed a 36.7% of score mismatches, indicating that further refinement of the Poppler wrapper is necessary for full cross-format consistency. In conclusion, common evaluation metrics may under-reward the nuanced effect that emojis contribute, suggesting the need for richer annotation schemes in future benchmarks.

5.3 Future Work

Planned extensions include crowd-sourcing multilingual emoji ratings, handling skin-tone and gender variants, exploring a lightweight embedding back-end, and releasing a Shiny GUI for drag-and-drop sentiment scoring.

In summary, `EmojiSentR` fills a long-standing methodological gap by treating emojis as first-class sentiment carriers within tidy R workflows. Preliminary tests show that adding an emoji channel alters polarity predictions in a substantial

subset of social-media posts, demonstrating the analytical cost of ignoring pictographic affect. The package's lightweight, modular design, coupled with fully open-source code, enables straightforward replication, extension, and integration into dashboards or experimental pipelines. Although the present release relies on an English-only lexicon and exhibits occasional PDF-processing mismatches, the architecture is engineered for rapid Unicode updates and forthcoming multilingual tables. Collectively, `EmojiSentR` offers behavioral researchers a reproducible, practical tool for capturing the full emotional nuance of contemporary digital text while laying a clear path for future accuracy benchmarks and feature growth.

Author Notes

Correspondence should be sent to Xin Tong, Department of Psychology, University of Virginia, Charlottesville, VA 22904; Email: `xt8b@virginia.edu`

Author Contributions

Conceptualization, L.H. (Logan Hanson) and E.L. (Elias Lahrim); Methodology, L.H. and E.L.; Software, E.L.; Validation, L.H. and E.L.; Data curation, E.L. and L.H.; Writing—original draft, L.H.; Writing—review & editing, L.H. and X.T. (Xin Tong); Visualization, L.H. and E.L.; Project administration, X.T.; Supervision, X.T. L.H. and E.L. contributed equally to this work.

Data Availability Statement

All code and data are openly hosted at `https://github.com/eliaslah/EmojiSentR`, including the package source, the 17,996 tweet evaluation corpus, the 300 tweet labeled subset, the internal Novak 1200 emoji lexicon, and scripts to reproduce the benchmarks.

References

Adobe Inc. (2022). *Future of creativity: 2022 global emoji trend report.* (Retrieved from https://www.adobe.com/)

Aggarwal, C. C. (2015). *Data mining: The textbook*. Cham: Springer. doi: https://doi.org/10.1007/978-3-319-14142-8

Blei, D. M., & Lafferty, J. D. (2007). A correlated topic model of science. *The Annals of Applied Statistics*, 17–35. doi: https://doi.org/10.1214/07-aoas114

Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent dirichlet allocation. *Journal of Machine Learning Research*, *3*(Jan), 993–1022. doi: https://doi.org/10.7551/mitpress/1120.003.0082

Boia, M., Faltings, B., Musat, C., & Pu, P. (2013). A study of user assessment of the emoji sentiment lexicon. *Proceedings of the International Conference on Social Informatics*, 24–33.

Broni, K. (2023). *What's new on the 10th annual world emoji day.* (Retrieved from https://blog.emojipedia.org/)

Gagolewski, M. (2024). stringi: Fast and portable character string processing in r [Computer software manual]. Retrieved from https://cran.r-project.org/package=stringi (R package version 1.8.4)

Hassani, S., Sabetzadeh, M., & Amyot, D. (2025). An empirical study on llm-based classification of requirements-related provisions in food-safety regulations. *Empirical Software Engineering*, *30*(3), 72. doi: https://doi.org/10.1007/s10664-025-10619-z

Hotho, A., Nürnberger, A., & Paaß, G. (2005). A brief survey of text mining. *LDV Forum*, *20*(1), 19–62. doi: https://doi.org/10.21248/jlcl.20.2005.68

Humphreys, A., & Wang, R. J. H. (2018). Automated text analysis for consumer research. *Journal of Consumer Research*, *44*(6), 1274–1306. doi: https://doi.org/10.1093/jcr/ucx104

Hutto, C. J., & Gilbert, E. (2014). Vader: A parsimonious rule-based model for sentiment analysis of social media text. In *Proceedings of the international aaai conference on web and social media* (Vol. 8, pp. 216–225). doi: https://doi.org/10.1609/icwsm.v8i1.14550

Itani, M., Roast, C., & Al-Kjayatt, S. (2017). Developing resources for sentiment analysis of informal arabic text in social media. *Procedia Computer Science*, *117*, 129–136. doi: https://doi.org/10.1016/j.procs.2017.10.101

Jockers, M. L. (2020). syuzhet: Extract sentiment and plot arcs in novels [Computer software manual]. Retrieved from https://cran.r-project.org/package=syuzhet (R package version 1.0.6)

Khaiser, F. K., Saad, A., & Mason, C. (2023). Sentiment analysis of students' feedback using text-based classification and nlp. *Journal of Language and Communication*, *10*(1), 101–111. doi: https://doi.org/10.47836/jlc.10.01.06

Kozik, R., Kula, S., Choraś, M., & Woźniak, M. (2022). Technical solution to counter potential crime: Text analysis to detect fake news and disinformation. *Journal of Computational Science*, *60*, 101576. doi: https://doi.org/10.1016/j.jocs.2022.101576

Kralj Novak, P., Smailović, J., Sluban, B., & Mozetič, I. (2015). Sentiment of emojis. *PLOS ONE*, *10*(12), e0144296. doi: https://doi.org/10.1371/journal.pone.0144296

Liu, H., Tsang, S., Wood, A., & Tong, X. (2025). Longitudinal sentiment analysis with conversation textual data. *Fudan Journal of the Humanities and Social Sciences*, *18*(1), 193–214. doi: https://doi.org/10.1007/s40647-024-00417-0

Machová, K., Szabóová, M., Paralič, J., & Mičko, J. (2023). Detection of emotion by text analysis using machine learning. *Frontiers in Psychology*, *14*, 1190326. doi: https://doi.org/10.3389/fpsyg.2023.1190326

Minaee, S., Kalchbrenner, N., Cambria, E., Nikzad, N., Chenaghlu, M., & Gao, J. (2021). Deep learning-based text classification: A comprehensive review. *ACM Computing Surveys*, *54*(3), 1–40. doi: https://doi.org/10.1145/3439726

Nandwani, P., & Verma, R. (2021). A review on sentiment analysis and emotion detection from text. *Social Network Analysis and Mining*, *11*(1), 81. doi: https://doi.org/10.1007/s13278-021-00776-6

Rinker, T. W. (2022). sentimentr: Calculate text polarity sentiment [Computer software manual]. Retrieved from https://cran.r-project.org/package=sentimentr (R package version 2.9.0)

Rinker, T. W. (2023). textclean: Text cleaning tools [Computer software manual]. Retrieved from https://cran.r-project.org/package=textclean (R package version 0.9.3)

Roberts, M. E., Stewart, B. M., & Tingley, D. (2019). stm: An r package for structural topic models. *Journal of Statistical Software*, *91*(2), 1–40. doi: https://doi.org/10.18637/jss.v091.i02

Rudis, B., & Robinson, D. (2024). emoji: Data and functions to work with emojis [Computer software manual]. Retrieved from https://cran.r-project.org/package=emoji (R package version 0.2.0)

Selivanov, D., Wang, Q., & Tang, Y. (2024). text2vec: Modern text mining framework for r [Computer software manual]. Retrieved from https://cran.r-project.org/package=text2vec (R package version 0.6)

Thakur, N. (2023). Sentiment and text analysis of public discourse on twitter about covid-19 and mpox. *Big Data and Cognitive Computing*, *7*(2), 116. doi: https://doi.org/10.3390/bdcc7020116

Urbanek, S. (2023). utf8: Unicode text processing [Computer software manual]. Retrieved from https://cran.r-project.org/package=utf8 (R package version 1.2.3)

Wankhade, M., Rao, A. C. S., & Kulkarni, C. (2022). A survey on sentiment analysis methods, applications, and challenges. *Artificial Intelligence Review*, *55*(7), 5731–5780. doi: https://doi.org/10.1007/s10462-022-10144-1

Westgate, M. J., Barton, P. S., Pierson, J. C., & Lindenmayer, D. B. (2015). Text analysis tools for identification of emerging topics and research gaps in conservation science. *Conservation Biology*, *29*(6), 1606–1614. doi: https://doi.org/10.1111/cobi.12605

Wickham, H. (2024). stringr: Simple, consistent wrappers for common string operations [Computer software manual]. Retrieved from https://cran.r-project.org/package=stringr (R package version 1.6.3)

Wickham, H., Francois, R., & D'Agostino McGowan, L. (2022). emo: Easily insert emojis into r documents [Computer software manual]. Retrieved from https://github.com/hadley/emo (R package version 0.0.0.9000)

Yu, G. (2022). emojifont: Emoji and font awesome in graphics [Computer software manual]. Retrieved from https://cran.r-project.org/package=emojifont (R package version 0.5.6)