SoftwareX 34 (2026) 102590

Contents lists available at ScienceDirect

SoftwareX

journal homepage: www.elsevier.com/locate/softx

TALL: Text analysis for all–an interactive R-shiny application for exploring, modeling, and visualizing textual data

Massimo Aria[a,d,*], Maria Spano[a,d], Luca D'Aniello[a,d], Corrado Cuccurullo[b,d], Michelangelo Misuraca[c,d]

[a] Department of Economics and Statistics, University of Naples Federico II, Via Cintia, Monte S. Angelo, 80126 Naples, Italy
[b] Department of Economics, University of Campania Luigi Vanvitelli, Corso Gran Priorato di Malta, 81043 Capua (CE), Italy
[c] Department of Management and Innovation Systems, University of Salerno, Via Giovanni Paolo II, 132, 84084 Fisciano (SA), Italy
[d] K-Synth srl, Via Cintia, Monte S. Angelo, 80126 Naples, Italy, 80134 Naples, Italy

HIGHLIGHTS

- TALL democratizes text analysis through a code-free, interactive R-Shiny interface.
- Supports 56 languages via 87 pre-trained models based on Universal Dependencies.
- Integrates an AI assistant for natural-language interpretation of analytical results.
- Unifies data import, NLP pre-processing, analysis, and visualization in one tool.
- Open-source, FAIR-compliant, and reproducible.

ARTICLE INFO

Keywords:
R-shiny app
NLP
Text mining
Data visualization
TALL

ABSTRACT

Analyzing unstructured textual data is essential across disciplines, but often requires programming skills that many researchers lack. TALL (*Text Analysis for All*) is an interactive R-Shiny application that unifies data import, cleaning, pre-processing, statistical analysis, and visualization into a single tool. It supports tokenization, lemmatization and Part-of-Speech (PoS) tagging for analyses in multiple languages. It includes topic modeling, correspondence and cluster analysis, co-occurrence networks, polarity detection, word embedding and text summarization. Designed for accessibility and reproducibility, TALL enables non-programmers to perform advanced text analysis efficiently and effectively. This article outlines the architecture and functionalities, demonstrating its use through illustrative examples.

Metadata

Current code version	v0.5.0
Permanent link to code/repository used for this code version	https://github.com/massimoaria/tall
Permanent link to Reproducible Capsule	*Not Applicable*
Legal Code License	*MIT License*
Code versioning system used	*Git*
Software code languages, tools, and services used	*R, R Shiny, Google Gemini*
Compilation requirements, operating environments & dependencies	*R* (≥ 3.5.0), *shiny*; *Dependencies*: *ca, igraph, pdftools, plotly, textrank, topicmodels, udpipe, umap, visNetwork, word2vec*
If available, link to developer documentation/manual	https://www.tall-app.com
Support email for questions	aria@unina.it

* Corresponding author at: Department of Economics and Statistics, University of Naples Federico II, Via Cintia, Monte S. Angelo, 80126 Naples, Italy.
Email addresses: aria@unina.it (M. Aria), maria.spano@unina.it (M. Spano), luca.daniello@unina.it (L. D'Aniello), corrado.cuccurullo@unicampania.it (C. Cuccurullo), mmisuraca@unisa.it (M. Misuraca).

https://doi.org/10.1016/j.softx.2026.102590
Received 24 October 2025; Received in revised form 6 February 2026; Accepted 1 March 2026
Available online 6 March 2026

1. Motivation and significance

The exponential growth of digital text has transformed how knowledge is generated, disseminated, and interpreted across scientific and professional domains [19]. From scholarly publications and policy documents to online reviews and social media posts, textual data represent a vast empirical resource for understanding complex social, cultural, and technological phenomena. Extracting structured and reproducible insights from these sources, however, requires sophisticated *Natural Language Processing* (NLP) and text mining techniques that demand programming expertise, advanced computational environments, and the ability to manage heterogeneous datasets. Consequently, many researchers—particularly in the social sciences, humanities, and applied disciplines—face technical barriers that limit systematic and replicable text analysis. This methodological divide constrains discovery and exacerbates reproducibility challenges in data-intensive research.

The TALL (*Text Analysis for All*) application was designed to bridge this gap by providing a transparent, interactive, and code-free platform for textual analysis. Developed entirely in R with the Shiny environment, TALL integrates data import, pre-processing, analysis, and visualization into a single workflow. It supports tokenization, PoS tagging, and lemmatization in multiple languages, and includes quantitative techniques such as co-word analysis [4,15], topic modeling [21], and sentiment detection [30].

Its main contribution lies in transforming qualitative text into quantitative, interpretable structures that can be visually explored and statistically examined, reducing the cognitive and technical burden of text mining. This integration supports both exploratory and confirmatory research. While commercial platforms are often proprietary and costly, and open-source tools such as *IRaMuTeQ* [24], *KH Coder* [18], and *AntConc* [2] remain limited by legacy interfaces or rigid data formats, TALL extends the open-source ecosystem by combining the flexibility of R packages with Shiny's interactivity. Each analytical step is documented and reproducible, ensuring methodological transparency. Table 1 offers a comparison of the main functionalities and features of TALL and other alternative software and libraries.

Beyond methodological innovation, TALL addresses ethical and epistemological issues in digital research. As NLP increasingly informs decisions in healthcare, education, and policy [e.g., [8,12,13]], auditable analytical environments are essential for accountability. TALL's design emphasizes openness and clarity: all intermediate results are visible, reproducible, and exportable, enabling users to trace analytical choices. The application also functions as a pedagogical resource for teaching computational text analysis, bridging the gap between qualitative interpretation and quantitative modeling.

Unlike tools that rely on external, often heterogeneous taggers (e.g., KH Coder) or fixed lexical lists (e.g., IRaMuTeQ), TALL provides a self-contained analytical environment. It features a unified, end-to-end pipeline that transforms raw text into structured data using a harmonized Universal Dependencies formalism. Notably, TALL does not merely redistribute existing tools but utilizes 87 independently trained models for 56 languages, ensuring they reflect the latest linguistic releases. To further reduce technical fragmentation, the framework provides immediate access to specialized resources without requiring external searches. These include ad-hoc databases for keyness analysis—derived from tokenizing the OpenSubtitles corpus (https://github.com/massimoaria/tall.language.models) to capture spoken language frequency distributions—and comprehensive sentiment lexicons for polarity detection across all supported languages. This architecture ensures methodological transparency and reproducibility while streamlining the transition from linguistic annotation to quantitative analysis.

2. Software description

TALL is accessible via R and can be executed locally using any standard Internet browser. Alternatively, it could be deployed remotely on a Shiny Server or shinyapps.io. This flexibility allows users to run TALL as a standalone desktop tool or as a shared institutional resource supporting collaborative analysis. The web-based GUI and statistical computations are developed using multiple R packages, as listed in the Supplementary Material. The development of TALL is consistent with the open-science principles and the *Findable, Accessible, Interoperable, Reusable* (FAIR) paradigm [36].

Table 1
Comparison of Interactive Text Analysis Applications.

Feature	TALL	IRaMuTeQ	KH Coder	quanteda	AntConc
Platform Type					
Web-based (R-Shiny)	✓	*X*	*X*	*X*	*X*
Cross-platform	✓	✓	✓	✓	✓
No programming required	✓	✓	✓	*X*	✓
Multilingual Support					
Number of languages	56	~10	13	NA	NA
Pre-trained models	87	None	None	None	None
PoS tagging	✓	✓	✓	*X*	*X*
Lemmatization	✓	✓	✓	*X*	*X*
Special-Entities Tagging	✓	*X*	*X*	*X*	*X*
Multiword Detection	✓	*X*	*X*	✓	*X*
Analytical Methods					
Topic modeling (LDA)	✓	*X*	*X*	✓	*X*
Correspondence analysis	✓	✓	✓	*X*	*X*
Reinert clustering	✓	✓	*X*	*X*	*X*
Network analysis	✓	✓	✓	✓	*X*
Word embeddings	✓	*X*	*X*	*X*	*X*
Sentiment analysis	✓	*X*	*X*	*X*	*X*
Keyness	✓	*X*	*X*	Partial	*X*
Concordances/KWIC	✓	✓	✓	✓	✓
AI Integration					
AI-assisted interpretation	✓	*X*	*X*	*X*	*X*
Natural language explanations	✓	*X*	*X*	*X*	*X*
Reproducibility					
FAIR-compliant	✓	Partial	Partial	✓	*X*
Exportable workflows	✓	X	X	✓	✓
Open source	✓	✓	Partial	✓	*X*
Maintenance & Distribution					
CRAN/Official repository	✓	*X*	*X*	✓	NA
Active development (2024-25)	✓	*X*	*X*	✓	✓
Last major update	2025	2020	2017	2025	2024
Documentation					
Comprehensive manual	✓	Not official	✓	✓	✓
Interactive tutorials	✓	Limited	Limited	✓	Limited

Key Features and Contributions

1. AI-assisted interpretation through Google Gemini integration (*TALL AI*) – providing context-aware natural language explanations.
2. 87 pre-trained multilingual models for 56 languages with standardized access through a dedicated repository.
3. Unified end-to-end workflow combining preprocessing, multiple analysis methods, and visualization in a single code-free interface.
4. Reproducibility by design with all steps documented and exportable.
5. FAIR-compliant architecture (see Supplementary Table S2).

2.1. Software architecture

The application architecture follows a client-server model that separates the user interface from the computational backend, ensuring responsiveness and modular extensibility. This modular design facilitates the integration of new analytical functions without altering the core framework. The overall structure mirrors that of other research-oriented Shiny applications – such as the biblioshiny GUI in the bibliometrix package [3] – with distinct modules dedicated to data management, pre-processing and analysis.

The *frontend layer* provides a dynamic, intuitive interface organized into tabs that guide users through each stage of the workflow, from data upload to pre-processing, exploration, and visualization. The layout utilizes a Bootstrap-based grid system for responsive rendering, ensuring optimal usability across various devices. The *backend layer* manages computational logic and data transformations, integrating several R packages and custom functions. Server components ensure that changes to inputs automatically trigger recalculation and plot regeneration.

TALL supports multilingual text analysis by incorporating 87 language-specific models specifically trained on the *Universal Dependencies Treebanks* [11](v 2.15). These models, each tailored to capture the morphological and syntactic characteristics of a given language, enable TALL to process and analyze *corpora* across 56 different languages with native linguistic accuracy. All models are also available via the tall.language.models repository [5] for external use, thereby promoting reproducibility and extensibility.

A distinctive feature of TALL is its integration with the *Google Gemini API*, called *TALL AI*, which provides a context-aware language-based assistant throughout the analytical workflow. This design enables an *interactive and explainable process* that links statistical computation with AI-assisted interpretation within a unified R-based framework. Through TALL AI, the app enhances interpretability by automatically generating contextualized comments and explanations for each analytical step. To ensure data privacy and computational efficiency, information transmitted to the external API is limited to metadata and aggregated results. The model's textual responses are rendered dynamically within the interface, complementing quantitative results with natural-language commentary. The current version implements several hybrid reasoning models from Google's Gemini family. Users can select among different versions and sizes via a drop-down menu, choosing between *Medium* (16,384 tokens) and *Large* (32,768 tokens) configurations, as well as alternative variants of Gemini models. This flexibility allows dynamic adaptation of reasoning depth and processing capacity. Default prompts are preconfigured but can be customized according to specific analytical goals and preferred narrative detail. TALL AI implements strict data minimization: only metadata and aggregated statistical results are transmitted for interpretation generation; raw textual content never leaves the user's computational environment. Users can disable the AI assistant entirely while retaining all other functionality. For institutional deployments that handle sensitive data, TALL can be configured to run on private servers with restricted API access or without external connectivity.

To ensure scalability and responsiveness for large *corpora*, TALL implements several optimization strategies at both architectural and algorithmic levels. Computationally intensive operations, such as topic modeling and word embedding training, are parallelized through the `future` [6] and `promises` [9] packages, enabling asynchronous execution that maintains interface responsiveness during long-running analyses. Pre-trained language models are downloaded on-demand and cached locally, eliminating redundant network transfers across sessions. For performance-critical functions where R's interpreted nature would impose bottlenecks, TALL employs compiled C++ implementations via the `Rcpp` interface. Specifically, the Reinert hierarchical descending classification algorithm, n-gram generation procedures, and statistical collocation measures – including the Pointwise Mutual Information (PMI) and the Morrone's IS score used for multi-word expression identification – are implemented in C++, yielding substantial performance improvements for large-scale text processing. These optimizations allow TALL to handle corpora of tens of thousands of documents on standard desktop hardware while maintaining the accessibility of a code-free interface.

Benchmarking on synthetic corpora (ranging from 78 to 5.3M tokens) demonstrates that TALL scales efficiently on standard hardware. Preprocessing exhibits linear complexity, achieving approximately 14,500 tokens/sec on an Apple M4 Pro, while Rcpp-optimized modules maintain high performance even at scale. Although reference vocabulary-intensive tasks can require up to 4.4 GB of RAM for large datasets, the framework remains highly accessible: small-to-medium corpora ($< 10^5$ tokens) operate comfortably within 1–2 GB of RAM. These results confirm that TALL is suitable for standard research laptops and supports institutional server deployment for exceptionally large-scale projects exceeding 1M tokens.

TALL implements comprehensive error management throughout the analytical pipeline to ensure data integrity and system stability. Input validation at each stage verifies file formats, automatically detects character encodings, and checks data type compatibility before processing begins. The system employs graceful degradation for optional features – for example, core functionalities remain fully operational without internet connectivity when the AI assistant is disabled. Real-time progress indicators and informative error messages guide users toward resolution with specific, actionable suggestions, minimizing frustration and preventing data loss through automatic session state persistence.

2.2. Software functionalities

TALL integrates an end-to-end text mining workflow into a unified and interactive graphical environment. The application is designed to guide the user through the entire process of textual data analysis, from data import to result interpretation, within an intuitive, tab-oriented R-Shiny interface. The overall structure is organized into three main layers: *Data Import and Management*, *Text Pre-processing*, and *Analysis*. These layers correspond to the logical stages of a typical computational text analysis pipeline, schematically depicted in Fig. 1. This modular structure enables users to explore each stage while maintaining the continuity of the data flow.

Data import and management. The first layer handles the acquisition, inspection, and preparation of the textual corpus. TALL accepts a broad range of input formats – including plain text (.txt), comma-separated files (.csv), spreadsheets (.xlsx), and Portable Document Format (.pdf) – and automatically detects character encoding to prevent loss of diacritical or non-ASCII characters. Metadata can be uploaded in parallel with the textual content, allowing users to associate attributes such as author, publication source, thematic label, or date with each text unit. The metadata can later serve as filters or grouping factors in downstream analyses. Moreover, TALL is integrated with the bibliometrix package to analyze scientific publication collections, accepting as input biblioshiny files as well. Within the same interface, a *Data Editor* panel provides a reactive table view that supports row filtering, text search, and random or stratified sampling. The segmentation utility enables the partitioning of large *corpora* into smaller textual units, such as paragraphs, records, or user-defined delimiters. These functionalities enable users to verify data integrity and structure prior to text preprocessing. The interface also allows dynamic saving of edited *corpora*, ensuring that early-stage manipulations are retained for subsequent sessions.

Text pre-processing. The pre-processing layer standardizes and transforms raw text into structured representations suitable for quantitative analysis Fig. 2. Core operations include *tokenization*, *PoS tagging*, and *lemmatization*. These tasks are implemented through the udpipe package [35], accessing the set of 87 *ad hoc* pre-trained language models. The selected model is downloaded and cached locally to enhance performance and facilitate offline processing. For domain-specific adaptation, users may upload custom dictionaries, thesauri, or stopword lists, which can refine lemmatization and lexical normalization. An additional utility allows both the automatic and manual creation of multi-word expressions (*n-grams*). TALL integrates four algorithms for identifying salient collocations and candidate keyphrases based on co-occurrence statistics, including the *Rapid Automatic Keyword Extraction* (RAKE) algorithm [26].

Analysis and visualization modules. The analytical layer provides a comprehensive suite of statistical and graphical modules, each implemented as an independent sub-menu that can be executed separately or in

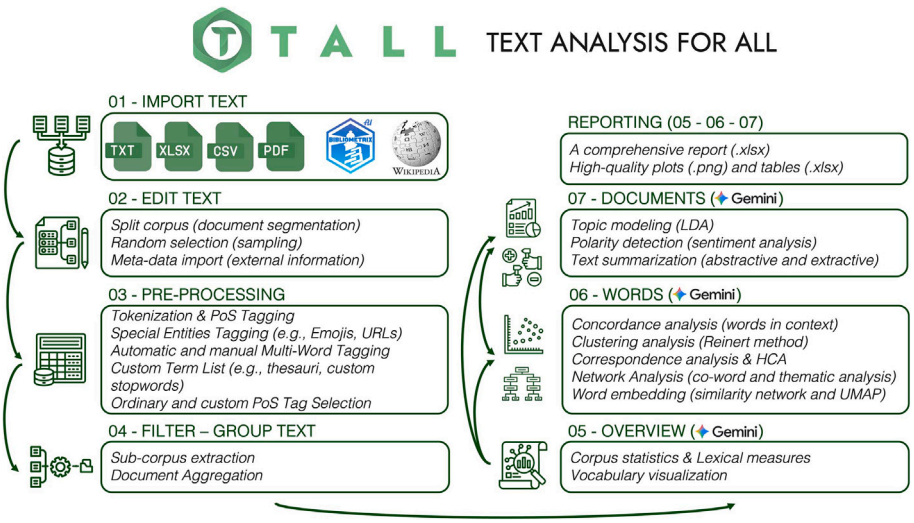

Fig. 1. TALL workflow and main components.

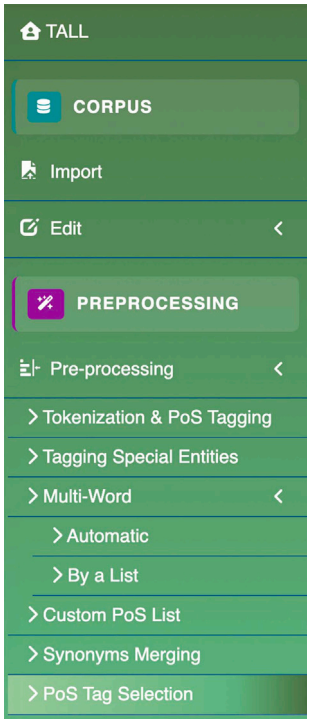

Fig. 2. Navigation sidebar of the TALL app: Import and Pre-processing modules.

sequence. Analyses can be performed on tokens or lemmas. The TALL AI assistant generates natural-language explanations of outputs, contextual comments, and narrative summaries of results. The generated commentary appears in collapsible panels alongside numerical or graphical outputs, providing interpretive context that enhances accessibility for non-specialist users. The *Overview* tab presents descriptive statistics on the analyzed collection, including frequency distributions, lexical richness, and term-weighting measures. The TF-IDF (*term frequency–inverse document frequency*) sub-tab refines term relevance [27], highlighting discriminating words across document sets. Word frequency plots and word clouds, generated using the `plotly` library [29], support zooming and tooltip interactions. The analytical layer distinguishes between word- and document-level analyses, guiding users toward the most suitable approach. Within the *Words* menu, several methods support the exploration of a *corpus* semantic space. *Words in Context* enables the analysis of concordances and co-text environments, facilitating qualitative interpretation of usage patterns [37]; *Reinert Clustering* implements the hierarchical descending clustering proposed by M. Reinert [25], generating lexically homogeneous clusters from co-occurrence patterns; *Correspondence Analysis* – executed via the `ca` package [23] – reveals latent semantic dimensions; the *Network* module performs co-occurrence analysis using the `igraph` [10] and `visNetwork` [1] packages; *Word Embeddings* computes high-dimensional vector representations through the `word2vec` package [34], allowing exploration of semantic relationships among terms. The *Document* menu includes three main modules: *Topic Modeling*, based on *Latent Dirichlet Allocation* (LDA) [7] and implemented with the `topicmodels` package [17], estimates thematic structures across the *corpus*; *Polarity Detection* evaluates emotional valence using sentiment lexicons such as Loughran–McDonald [20] and NRC [22], producing visual dashboards of sentiment distributions across documents and topics; and *Summarization* extracts key sentences via the `textrank` package [33], facilitating concise content overviews. Abstractive summarization can also be performed.

The numerical and graphical results are exportable in multiple formats. All the tables and statistics can be downloaded as `.csv` or `.xlsx` files, whereas visualizations can be downloaded as `.png` outputs. Moreover, it is possible to assemble the various outcomes of the analytical layer into a comprehensive report (Fig. 3) that can be exported as a single `.xlsx` file.

3. Illustrative examples

To demonstrate the capabilities of TALL, two examples based on publicly available datasets included in the application are presented. These examples illustrate how users can perform complete analytical workflows – from data import to interpretation – without writing any code.

3.1. Example 1: thematic analysis on BBC news

The first example uses a set of 386 short news articles from the *BBC News corpus*, a collection of English-language stories retrieved from the BBC website [16]. After loading the *corpus*, standard pre-processing steps are applied through the *Pre-processing* menu, including tokenization and lemmatization via the `EN GUM` model. Adjectives, nouns, proper nouns, and verbs are retained by default. In the *Overview* menu

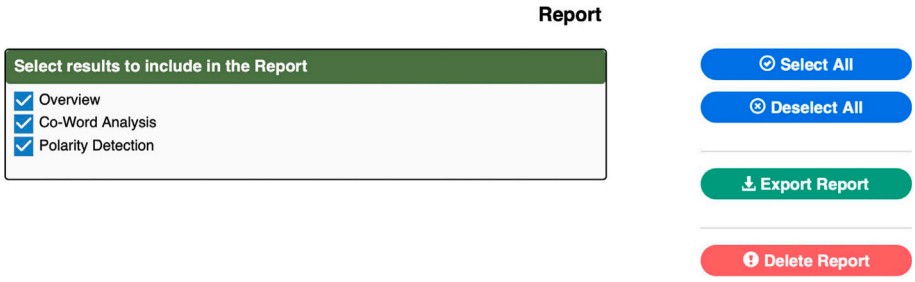

Fig. 3. Interface for report customization showing selected outcomes and options.

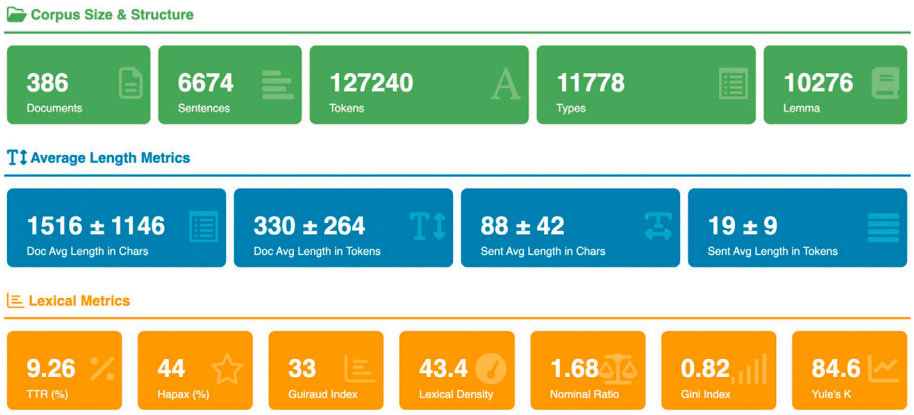

Fig. 4. Descriptive statistics of the BBC News entertainment sub-*corpus*.

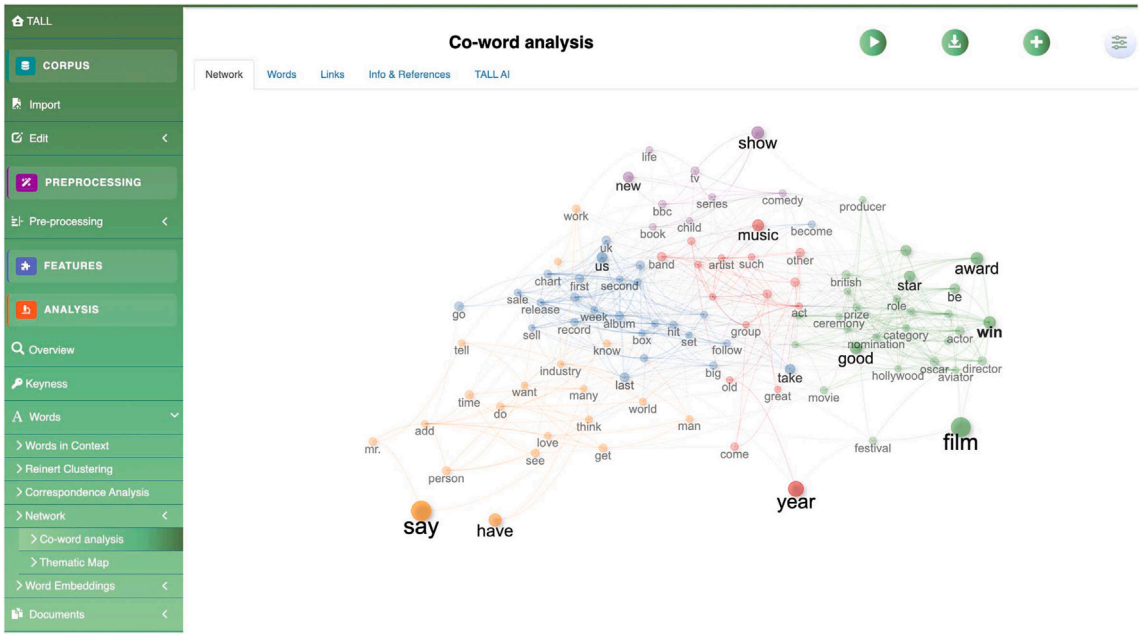

Fig. 5. Community detection on the BBC News entertainment sub-*corpus*.

Fig. 4, *corpus* statistics and lexical diversity measures provide an initial characterization of the collection.

The *Network* module highlights the main co-occurrence relationships among terms. After segmenting each document into sentences, a community detection algorithm is applied to the adjacency matrix of the top 100 lemmas' co-occurrences, identifying five dense sub-graphs (Fig. 5).

With the help of TALL AI, we interpret the five sub-graphs as topical clusters: *Awards/Movies* (green cluster), *Music and Performance* (red cluster), *Music Industry and Sales* (blue cluster), *Media and Entertainment Coverage* (purple cluster), *General News* (orange cluster). We can also gain some insights via TALL AI (e.g., *the entertainment news focuses heavily on film awards and the movie industry*; *music-related news is a significant portion of the content*).

Fig. 6. Top word distributions by document polarity in the US Airline Tweets *corpus*.

3.2. Example 2: sentiment detection on US airline tweets

The second example employs the *US Airline Tweets corpus*, which contains 14,640 tweets directed at six major American airlines in February 2015 [14].

After loading the *corpus*, standard pre-processing steps are applied through the *Pre-processing* sub-menu, including tokenization and lemmatization via the EN EWT model. Hashtags detected in the *Tagging Special Entities* sub-menu are retained, along with adjectives, nouns, and proper nouns. Within the *Polarity Detection* module, sentiment analysis is performed using the NRC Emotion Lexicon. The resulting dashboards display the distribution of sentiment across airlines, showing that negative tweets are predominantly associated with service disruptions and delays (Fig. 6).

In TALL AI comments, it is possible to find *Key Takeaways & Implications* derived from the outcomes of the analysis: e.g, *the dominance of 'delay' and 'cancel' in negative tweets suggests airlines should focus on improving their processes for handling flight disruptions (…)*; *both positive and negative tweets highlight the importance of customer service (…)*. By selecting sub-sets of positive or negative documents, users can further examine common lexical patterns using, for example, *Words in Context* or *Topic Modeling* modules. A step-by-step workflow example on the US Airline Tweets dataset is available in the supplementary material.

4. Impact and conclusions

TALL advances reproducible and transparent analysis of textual data by integrating NLP and text mining methods into a unified, open-source environment. The application operationalizes the FAIR data principles through explicit documentation of each analytical step, persistent repositories, and interoperable export formats. Each module is accompanied by metadata that describes the algorithms and parameters, ensuring that both human and machine agents can reproduce the analytical process. The combination of classical methods – such as correspondence analysis and topic modeling – with recent advances in word embeddings and network analytics enables multimodal language exploration. The embedded TALL AI assistant introduces AI-assisted interpretation of results, generating textual explanations and fostering interaction between statistical evidence and linguistic reasoning. This hybrid human–machine approach stimulates new research directions in semantic modeling, interpretability, and evaluation of generative language models within controlled analytical workflows.

TALL enhances the investigation of research questions in fields in which unstructured text is a primary empirical source. In social and political sciences, it supports large-scale thematic and sentiment analyses of institutional reports or online debates. In science mapping, the integration with the bibliometrix package allows the combination of semantic and citation-based analyses. In digital humanities, the multilingual architecture based on our *ad hoc* pre-trained models enables cross-linguistic *corpus* studies and cultural analyses. By lowering technical barriers, TALL extends the reach of NLP to communities that have traditionally lacked computational resources, promoting methodological inclusivity and reproducibility in qualitative inquiry. Evidence of its early impact is already visible: its role as an application for non-programmers has been emphasized in [28,31], and it has been applied in empirical studies using textual data [32]. TALL is currently employed in postgraduate and doctoral programs at several Italian universities, supporting the teaching of data-driven methods in social sciences and computational linguistics. The package has been officially published on the *Comprehensive R Archive Network* (CRAN) (https://cran.r-project.org/package=tall), ensuring long-term accessibility and compliance with R standards. According to CRANlogs, TALL has exceeded 4000 downloads since its release (February, 2025), denoting growing interest within the research community. This diffusion underscores the app's utility and reliability for text mining and NLP tasks. CRAN publication also guarantees version control and continuous integration within the R ecosystem, enhancing interoperability with other open-source resources. By coupling algorithmic transparency with accessibility, TALL consolidates the methodological foundations of text analytics as a reproducible scientific practice, positioning itself as an extensible infrastructure for interdisciplinary NLP research.

TALL is designed as a desktop application for standalone personal computers, with computational constraints inherent in the user's hardware specifications (processor speed, available RAM). While we have implemented multiple optimization strategies to enhance computational efficiency (see Section 2.1), very large *corpora* may require server deployment for practical processing times. A key advantage of TALL's web-based architecture is scalability: the same application can be seamlessly deployed on institutional Shiny Server infrastructure, enabling analysis of large-scale *corpora* without modifying the codebase

or user interface. This architectural flexibility allows TALL to serve both individual researchers working with modest datasets on personal laptops and research groups requiring centralized, high-performance text analysis infrastructure. TALL's AI integration requires privacy evaluation for confidential research. Despite only aggregated statistics are transmitted, these may reveal sensitive patterns in small *corpora*. For regulated data (e.g., EU-GDPR, US-HIPAA), it is recommended to: (1) disable TALL AI; (2) isolated server deployment; or (3) anonymize data. In any case, it would be necessary to consult institutional review boards for human subjects research. Future versions may support local language models to enhance privacy management.

The development team maintains TALL through regular updates on GitHub and CRAN, ensuring compatibility with future R releases and the incorporation of emerging NLP and visualization libraries. Future versions will enhance interoperability through API endpoints for external datasets and connectors to cloud-based repositories (e.g., Kaggle), thereby improving scalability and facilitating collaborative research.

CRediT authorship contribution statement

Massimo Aria: Writing – review & editing, Validation, Software, Resources, Methodology, Data curation, Conceptualization. **Maria Spano:** Writing – review & editing, Validation, Software, Resources, Methodology, Data curation, Conceptualization. **Luca D'Aniello:** Writing – review & editing, Validation, Software, Resources, Methodology, Data curation, Conceptualization. **Corrado Cuccurullo:** Writing – review & editing, Visualization, Conceptualization. **Michelangelo Misuraca:** Writing – review & editing, Writing – original draft, Validation, Methodology, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

This research has been financed by the following research projects: Italian Ministry of University and Research (Ministero dell'Università e della Ricerca, MUR) Project PRIN-2022 (Project Code:2022825Y5E; CUP: E53D2300611006) – *SCIK-HEALTH*.

Appendix A. Supplementary data

Supplementary data to this article can be found online at doi:10.1016/j.softx.2026.102590.

Data availability

The code and data used in this paper can be retrieved from the repositories at https://github.com/massimoaria.

References

[1] Almende BV, Contributors, Thieurmel B. visNetwork: network visualization using 'vis.js'. Library. 2025. https://doi.org/10.32614/CRAN.package.visNetwork, R package version 2.1.4.
[2] Anthony L. AntConc. version 4.3.1 [software]. 2024. https://www.laurenceanthony.net/software/AntConc [29 July 2024].
[3] Aria M, Cuccurullo C. Bibliometrix: an R-tool for comprehensive science mapping analysis. J Informetr 2017;11(4):959–75. https://doi.org/10.1016/j.joi.2017.08.007
[4] Aria M, Cuccurullo C, D'Aniello L, Misuraca M, Spano M. Thematic analysis as a new culturomic tool: the social media coverage on COVID-19 pandemic in Italy. Sustainability 2022;14(6):3643. https://doi.org/10.3390/su14063643
[5] Aria M. Pre-trained models for TALL; 2025. https://github.com/massimoaria/tall.language.models.
[6] Bengtsson H. A unifying framework for parallel and distributed processing in R using futures. R J 2021;13(2):208–27. https://doi.org/10.32614/RJ-2021-048
[7] Blei DM, Ng AY, Jordan MI. Latent dirichlet allocation. J Mach Learn Res 2003;3(Jan):993–1022. https://www.jmlr.org/papers/volume3/blei03a/blei03a.pdf.
[8] Chen H, Fuller SS, Friedman C, Hersh HW Management data mining, and text mining in medical informatics. In Chen H, Fuller SS, Friedman C, Hersh W, editors. Medical informatics. Integrated series in information systems. Boston, MA: Springer; 2005. pp. 3–33. https://doi.org/10.1007/0-387-25739-X_1
[9] Cheng J. Promises: abstractions for promise-based asynchronous programming. 2025. https://doi.org/10.32614/CRAN.package.promises, R package version 1.5.0.
[10] Csárdi G, Nepusz T. The Igraph software package for complex network research. InterJournal Complex Syst 2006; 1695. https://igraph.org.
[11] de Marneffe MC, Manning C, Nivre J, Zeman D. Universal dependencies. Comput Linguist 2021;47(2):255–308. https://doi.org/10.1162/coli_a_00402
[12] DiMaggio P. Adapting computational text analysis to social science (and vice versa). Big Data Soc 2015; 2:2053951715602908. https://doi.org/10.1177/2053951715602908
[13] Feuerriegel S, Maarouf A, Bär D, Geissler D, Schweisthal J, Pröllochs N, et al. Using natural language processing to analyse text data in behavioural science. Nat Rev Psychol 2025;4:96–111. https://doi.org/10.1038/s44159-024-00392-z
[14] Eight F. Twitter US airline sentiment [dataset]. Kaggle Data 2015. https://www.kaggle.com/datasets/crowdflower/twitter-airline-sentiment.
[15] Fortunato S, Hric D. Community detection in networks: a user guide. Phys Rep 2016;659:1–44. https://doi.org/10.1016/j.physrep.2016.09.002
[16] Greene D, Cunningham P. BBC news [dataset]. ML resources UCD; 2005. http://mlg.ucd.ie/datasets/bbc.html.
[17] Grün B, Hornik K. topicmodels: an R package for fitting topic models. J Stat Softw 2011;40(13):1–30. https://doi.org/10.18637/jss.v040.i13
[18] Higuchi K. New quantitative text analytical method and KH coder software. Jpn Sociol Rev 2017;68(3):334–50. https://doi.org/10.4057/jsr.68.334
[19] Ignatow G. Theoretical foundations for digital text analysis. J Theory Soc Behav 2016;46(1):104–20. https://doi.org/10.1111/jtsb.12086
[20] Loughran T, McDonald B. Textual analysis in accounting and finance: a survey. J Account Res 2016;54(4):1187–230. https://doi.org/10.1111/1475-679X.12123
[21] Misuraca M, Spano M. Unsupervised analytic strategies to explore large document collections. In Iezzi DF, Mayaffre D, Misuraca M, editors. Text analytics, advances and challenges. Cham: Springer; 2020. pp. 17–28. https://doi.org/10.1007/978-3-030-52680-1_2
[22] Mohammad SM, Turney DP. Crowdsourcing a word-emotion association lexicon. Comput Intell 2013;29(3):436–65. https://doi.org/10.1111/j.1467-8640.2012.00460.x
[23] Nenadić O, Greenacre M. Correspondence analysis in R, with two- and three-dimensional graphics: the CA package. J Stat Softw 2007;20(3):1–13. https://doi.org/10.18637/jss.v020.i03
[24] Ratinaud P. IRaMuTeQ : interface de r pour LES analyses multidimensionnelles de textes ET de questionnaires. Version 0.8 alpha 7 [software]. 2 Nov 2024. http://www.iramuteq.org.
[25] Reinert M. Une méthode de classification descendante hiérarchique: application à l'analyse lexicale par contexte. Cah Anal Donn 1983;8(2):187–98. https://www.numdam.org/item/CAD_1983__8_2_187_0.
[26] Rose S, Engel D, Cramer N, Cowley W. Automatic keyword extraction from individual documents. In Berry MW, Kogan J, editors. Text mining: applications and theory. Chichester: Wiley; 2010. pp. 1–20. https://doi.org/10.1002/9780470689646.ch1
[27] Salton G, Buckley C. Term-weighting approaches in automatic text retrieval. Inf Process Manage 1988;24(5):513–23. https://doi.org/10.1016/0306-4573(88)90021-0
[28] Santosa FA, Lamba M, George C, Downie JS. Coconut Libtool: bridging textual analysis gaps for Non-Programmers. Proc Assoc Inf Sci Technol 2024;61(1):639–44. https://doi.org/10.1002/pra2.1072
[29] Sievert C. Interactive Web-Based data visualization with r, plotly, and shiny. Boca Raton, FL: Chapman & Hall/CRC; 2020. https://doi.org/10.1201/9780429447273
[30] Taboada M, Brooke J, Tofiloski M, Voll K, Stede M. Lexicon based methods for sentiment analysis. Comput Linguist 2011;37(2):267–307. https://aclanthology.org/J11-2001/.
[31] Tay A. TALL - text analysis for ALL - new R shiny; 2025. https://library.smu.edu.sg/topics-insights/tall-text-analysis-all-new-r-shiny [accessed 17 October 2025].
[32] Valentino C, Aria M, Angelelli M, Ciavolino E In: Boccuzzo G, Bovo E, Manisera M, Salmaso L. Patient satisfaction and healthcare quality perception: an explorative textual analysis Innovation & society – statistics and data science for evaluation and quality. Padua: Cleup; 2025. pp. 552–7.
[33] Wijffels J. TextRank: summarize text by ranking sentences and finding keywords; 2020. https://CRAN.R-project.org/package=textrank, R package version 0.3.1.
[34] Wijffels J, Watanabe K. Word2vec: distributed representations of words; 2023. https://CRAN.R-project.org/package=word2vec, R package version 0.4.0.
[35] Wijffels J. UDPipe: tokenization, parts of speech tagging, lemmatization and dependency parsing with the 'UDPipe' 'NLP, Toolkit. 2025. https://doi.org/10.32614/CRAN.package.udpipe,R package version 0.8.12.
[36] Wilkinson MD, Dumontier M, Aalbersberg IJ, Appleton G, Axton M, Baak A, et al. The FAIR guiding principles for scientific data management and stewardship. Sci Data 2016;3:160018. https://doi.org/10.1038/sdata.2016.18
[37] Wulff S, Baker P. Analyzing concordances. In Paquot M, Gries ST, editors. A practical handbook of corpus linguistics. Cham: Springer; 2020. pp. 161–79. https://doi.org/10.1007/978-3-030-46216-1_8