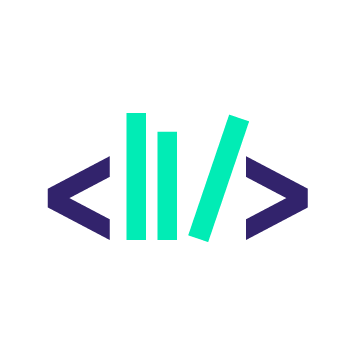


D8.2

REPORT AND PROTOTYPES   
FOR ANNOTATION AS ENRICHMENT Authors: Julie M. Birkholz, Silvie Cinková, Matthieu Decorde, Victor Diego Fresno Fernandez, Serge Heiden, Maarten Janssen, Michal Křen, Alvaro Perez Pozo, and Salvador Ros.

Date: February 28, 2024



This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 101004984

Project Acronym: CLS INFRA

Project Full Title: Computational Literary Studies Infrastructure

Grant Agreement No.: 101004984

Deliverable/Document Information

Deliverable No.: D8.2

Deliverable Title: REPORT AND PROTOTYPES FOR ANNOTATION AS ENRICHMENT Authors: Julie M. Birkholz, Silvie Cinková, Matthieu Decorde, Victor Diego Fresno Fernandez, Serge Heiden, Maarten Janssen, Michal Křen, Alvaro Perez Pozo, and Salvador Ros.

Dissemination Level: PUBLIC

Document History

Version/Date Changes/Approval Author/Approved by

V.1 28/02/2024 Julie M. Birkholz

## The following report documents the ongoing work of Work Package 8 in the CLS Infrastructure Project. The general goals of this work package are to increase the ease of access and application to NLP tools, including for less-well-resourced languages, as well as their standardization. To achieve this goal we have identified a number of necessary tools and services that are needed in literary studies and related reports to document them for users. For this deliverable specifically we have developed a number of tools to support the enrichment and or annotation of text, and specifically to support multiple languages. These tools integrate ongoing work in both WP 6 and 7 which facilitates integration of the pipeline from data preparation, programmable corpora, to analysis and back.

The annotation of texts is an age old practice that in its most simplistic form is a note or comment in a written text (e.g. a novel, technical document, letter). This also could include the practice of a non-textual response for example by highlighting or underlining something in a text as well as including a footnote. See D8.1 Report of the Tools for the Basic Natural Language Processing (NLP) Tasks in the CLS Context for further examples of annotation for natural language and textual sources.

Annotations (or textual encoding) can serve many purposes depending on the task at hand. In literary studies specifically, annotation is implemented for a number of reasons: 1) for reference for a scholar’s own use or for others (i.e. scholarly edition), 2) to share notes and ideas with others, 3) linguistic markups and so forth. These annotations are then used in many different ways including used as a source for scholarly editions to qualitative and quantitative pattern tracing that could include a number of methods from discourse or theme identification to the use of NLP such as identification of named entities.

As described in Deliverable 3.1 in DH and CLS, the accepted standard for textual encoding are the Guidelines of the Text Encoding Initiative (TEI). TEI represents a pivotal framework for encoding and annotating textual materials, fostering standardized methods for digital representation. The Text Encoding Initiative and related standardization of encoding was established in 1987, is a consortium of scholars and institutions committed to developing and maintaining guidelines for the digital representation of texts in the humanities and social sciences. TEI thus provides a shared standardized framework that facilitates the creation, dissemination, and analysis of textual data in digital environments. This enables broader accessibility to diverse textual resources, fostering scholarly inquiry and cultural exchange.

TEI promotes the preservation of textual materials by facilitating their conversion into digital formats. Consequently, these guidelines ensure interoperability across different platforms and systems, allowing for integration and exchange of encoded texts. It allows a nuanced approach to textual encoding, allowing for the representation of complex textual features such as structure, semantics, and linguistic elements. This granularity enables scholars to conduct intricate analyses and interpretations of textual materials. These guidelines are adaptable to diverse textual genres, languages, and disciplinary contexts.

TEI employs a markup language, typically XML (eXtensible Markup Language), to encode textual materials. This markup facilitates the annotation of various textual features, including structure, typography, linguistic elements, and metadata. It implements a hierarchical structure for encoding texts, wherein elements are nested within one another to represent their relationships and dependencies. This hierarchical approach enables the representation of complex textual structures, such as paragraphs, chapters, and sections, with precision and clarity. In addition, there is a modular architecture addresses specific aspects of textual encoding. These modules encompass diverse domains, including transcription, linguistics, manuscript description, and scholarly editing, allowing users to selectively incorporate relevant components into their encoding projects.

In this report we report a number of annotation tools and a large share of them also implement TEI-encoding. A number of other deliverables within CLS also have mentioned the use of annotation and TEI specifically. This include: D3.1 Series of five short survey papers on methodological concerns and D6.1 Inventory of existing data sources and formats, which we have highlighted here in boxed quotes for reference, as well as the extend list of tools from D8.1 Report of the Tools for the Basic Natural Language Processing (NLP) Tasks in the CLS Context.

From D3.2: “This XML-based format allows for the inclusion of detailed metadata, encoding of textual structure as well as token-based linguistic annotation (see (Burnard 2014)). Using a formal language like XML to represent analytic annotation in a text has a significant advantage in automatic validation. This means that the annotation used in a document can be checked to see if it conforms to a previously defined model of which types of annotation are permitted and in which contexts (Burnard 2014). Examples of recent literary corpora that use XML-TEI include the Deutsches Textarchiv, the Diachronic Spanish Sonnet Corpus, the European Literary Text Collection, DraCor, TextGrid’s Digitale Bibliothek, Théâtre classique and many more. Corpora are often prepared using XML. TEI is a frequently adopted de facto standard. However, there are simplified suggestions, such as Hardie (2014), which do not impose strict rules. Simplified versions of TEI might be a good choice for smaller projects, but departing from TEI does come with considerable downsides with respect to interoperability. To make a corpus ready for query systems such as CWB and Manatee (Rychlý 2007), we often need to provide the corpus formatted in a vertical format (vrt). This means each token is represented in a line and its respective tags are added to subsequent columns (Stephanie Evert 2022). Other corpus query and analysis systems, however, are able to ingest annotated XMLTEI, for example TXM or TEITOK. The decision of how to format the corpus is normally made according to how the analysis will be performed. The chapter “General Issues in Data Analysis” (Chapter 3) deals with that.”

From D6.1: “It is well known that one of the most established formats, TEI-XML, is so flexible that it does not ensure full interoperability of different data sets. Hence, a really helpful description of a corpus format is solely a full formal definition (such as a DTD) to which its document pertain. This definition cannot be further summarized and requires good XML proficiency from the reader. D 6.1 has so far left this issue open. Our makeshift approach is giving the reader at least a raw estimate whether a tool can work with any TEI-XML at all, based on several test cases, and a basic presentation of the tool’s assumptions about the data format and structure beyond TEI-XML. Besides, we provide a somewhat more narrative section about how to use a given tool and what for.”

In this deliverable we report on the development of a number of tools that aim to fill the gap between TEI & NLP tool use. This includes TEITOK, TXM, PoetryLab API, Rantanplan, and Alberti-stanzas. We have listed here the use of these tools and included below more detailed information of the specific tools and their functionalities.

TEITOK can be used to create corpora and keep a wide range of different types of information (facsimile alignment, audio alignment, dependency relations, pos tagging, geolocation information, IGT, morphological data, parallel alignment, witness alignment). It then provides various interfaces to exploit all these types of information, and in most cases provides an editing environment as well to provide automatic annotations and manually correct or create annotations. It also provides searches through the corpus, either using CQL for simple annotated files, or Grew or PML-TQ for dependency parsed corpora. It also provides specific search functionality for parallel corpora (be it based on translation or on witnesses) as well as for spoken and facsimile aligned data.

TXM can be used to create and analyze four main types of textual corpora: written texts (possibly aligned with fac-simile), speech transcriptions (possibly synchronized with media files at word level), multilingual/parallel corpora, and corpora encoded in a table (e.g. answers to a survey, tweets). TXM can automatically annotate texts with part of speech and lemma with the help of TreeTagger (or other NLP tools) or use linguistic annotations already encoded into the sources.

Specifically for poetry annotation the following tools have been developed: PoetryLab API, Rantanplan, and Alberti-stanzas. The PoetryLab API is a REST API that provides an analysis endpoint to extract information about the poems. It can annotate Spanish poetry with rhyme, metric, enjambment, and part of speech. Only the method POST is allowed at the /analysis endpoint. A playground interface documenting the API can be found at the /ui endpoint. Rantanplan is an extension of this Service and can annotate Spanish poetry with rhyme, meter, and part of speech. Alberti-stanzas is a language model trained with poetic texts that returns word embeddings. It has been trained for the stanzas detection task port Spanish poetry. It can return the stanza type of a Spanish poem.

# Tool name

TEITOK

# Version

3.2.781

# Version Date

2023-12-29

# Which Operation Systems does it work on?

* Linux
* MacOs

# License

* Fully free (GNU GPLv3)

# ~~Di~~stribution

* Local server-client installation

# Where to download, how to install, documentation, user guides

* https://gitlab.com/maartenes/TEITOK
* Project URL: http://www.teitok.org/
* Help Pages: http://www.teitok.org/index.php?action=help

# User interface

* Web GUI
* REST-API

# Docker instance

* There is a docker version of an earlier version TEITOK: https://github.com/rahonalab/TEITOK-docker

# How does the tool process your text?

* Convert from several input formats
* Run annotation tools on the server for automatic annotation
* Manually correct or annotate in the GUI
* Use the API to run processing or annotation tools remotely.

# Does the tool allow you to export the results?

* Yes

# Is this tool a statistical model (or a set thereof)?

* No

# Does this tool include statistical models?

* Yes

# Can you easily train and use your own statistical model with this tool?

There is a native POS tagger (NeoTag) that can be easily trained in the GUI on the manually corrected data. But NeoTag is gradually being replaced by UDPIPE, which currently can only be trained in the GUI using the older 1.0 version

# XML-TEI compatibility

* The whole tool is TEI/XML based, with minimal deviation from the standard to make it possible to edit and process/display, but all such deviation can be converted to standard TEI in the export.

# How is an input document to be structured for the tool?

The input should ideally be in TEI/XML, but there are various conversion tools that convert from a collection of OCR, Spoken, and annotated formats. But all internal files are TEI/MXL

# Methods and features

|  |  |  |  |
| --- | --- | --- | --- |
| **Quick orientation for contributors** | **Task (what you want to achieve with that method)** | **Method (usually more generic than task)** | **Features/ Metric/ Formalism, Tagset** |
| **NLP** | Recognize borders of tokens and sentences | Tokenization & Sentence splitting | Features: Tokens; Sentence borders |
| **Corpus manager** | Search in a text corpus | Linear text search with Corpus Query Processor and its flavors | Features: Text snippets matching a query;  Formalism: CQL |
| **Corpus manager** | Search in a parsed text corpus (treebank) | Syntactic tree search | Features: Syntactic trees matching a query.  Formalism: PMLTQ; Grew |
| **Corpus manager** | Tell which words typically co-occur with a given word | Collocations analysis | Metric: Statistics expressing the mutual attraction of two words or short text segments (MI, Log-Likelihood Ratio) |
| **Corpus manager -Textometry** | Give summary statistics of a corpus or selected texts | Various formulas | Metric: Token frequencies, Lexical richness/diversity; Key words, Thematic concentration, Descriptivity/activity, Verb distance, Entropy, Hapaxes |
| **Annotation tool** | Manually annotate your texts, e. g. with named entities | Manual text annotation/annotation editing | Features: Labeled text spans or tokens or their mutual relations |
| **Information extraction** | Relational extraction from metadata (DraCor etc.) | Network analysis | Features: Network graph with characters as nodes and interactions as edges; Metrics: Degree centrality, Eigenvector centrality, Weight |
| **Information extraction** | Relational extraction from unstructured text |  |  |
| **Information extraction** | Event extraction |  |  |

# Which human languages does the tool support (as of February 2023)?

The tool is fully language independent, and has support for RTL languages, as well as non-alphabetic scripts. TEITOK has been often used for NLP on LRL languages such as Ladino or Sardinian.

# Which other tools from this list does it integrate?

* UDPipe
* Corpus WorkBench
* DraCor
* NameTag
* QuitaUp (under development)
* INCePTION
* GrewMatch
* NeoTag

# What can this tool do for you?

TEITOK can be used to create corpora and keep a wide range of different types of information (facsimile alignment, audio alignment, dependency relations, pos tagging, geolocation information, IGT, morphological data, parallel alignment, witness alignment). It then provides various interfaces to exploit all these types of information, and in most cases provides an editing environment as well to provide automatic annotations and manually correct or create annotations. It also provides searches through the corpus, either using CQL for simple annotated files, or Grew or PML-TQ for dependency parsed corpora. It also provides specific search functionality for parallel corpora (be it based on translation or on witnesses) as well as for spoken and facsimile aligned data.

# Papers

Arrabal Rodríguez, P. 2022. TEITOK, a visual solution for XML/TEI encoding: editing, annotating and hosting linguistic corpora. RIDE – A review journal for digital editions and resources

Janssen, M. 2021. A Corpus with Wavesurfer and TEI: Speech and Video in TEITOK. International Conference on Text, Speech, and Dialogue. LNAI 12848: 261–268

Janssen, M. 2021. Integrating TEITOK and Kontext/PMLTQ at LINDAT. Selected papers from the CLARIN Annual Conference 2020. Linköping Electronic Conference Proceedings 180: 180 104–110.104

Janssen, M. 2018. TEITOK as a tool for Dependency Grammar. In: Procesamiento del Lenguaje Natural, vol. 61.

Janssen, M. 2018. Adding Words to Manuscripts: From PagesXML to TEITOK. In: Méndez E., Crestani F., Ribeiro C., David G., Lopes J. (eds) Digital Libraries for Open Knowledge. TPDL 2018. Lecture Notes in Computer Science, vol 11057. Springer, Cham.

Janssen, M. and Vaamonde, G. 2017. From traditional historical corpora to TEI-based corpora. Benefits and challenges in joining digital edition and corpus annotation. In: Calen barbas, falen cartas. A escrita en galego na Idade Moderna. Santiago de Compostela.

Janssen, M. 2016. TEITOK: Text-Faithful Annotated Corpora. Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016), Portorož, Slovenia.

Janssen, M. 2015. Multi-level manuscript transcription: TEITOK. Congresso de Humanidades Digitais em Portugal, Lisboa. Portugal

# Tool name

TXM desktop

# Nickname

TXM

# Version

0.8.3

# Version Date

2023-11-07

# Which Operation Systems does it work on?

* Windows
* MacOs
* Linux

# License

* Fully open (GNU GPLv3)

# Price

* Free

# Distribution

* Local application
  + installed by OS dependent installation software

# Where to download, how to install, documentation, user guides

* Download: <https://txm.gitpages.huma-num.fr/textometrie/files/software/TXM/0.8.2/en/>
* Project URL: <https://txm.gitpages.huma-num.fr/textometrie/en/>
* Online User Manual (French): <https://txm.gitpages.huma-num.fr/txm-manual>
* PDF User manual (old 0.7): <https://txm.gitpages.huma-num.fr/textometrie/files/documentation/TXM%20Manual%200.7.pdf>
* All the docs: <https://txm.gitpages.huma-num.fr/textometrie/en/Documentation>
* User Wiki (French) : <https://groupes.renater.fr/wiki/txm-users/index> (includes a FAQ)
* Leaflet: <http://sourceforge.net/projects/txm/files/documentation/TXM%20Leaftlet%20EN.pdf/download>

# User interface

* GUI in three languages (FR, EN & RU)
* script in Groovy
* script in R

# Docker instance

* No

# How does the tool process your text?

* Import a set of documents as a corpus
* Convert from several input formats to XML-TEI TXM internal pivot format, available input formats:
  + .docx, .odt, .rtf
  + .txt
  + .xml
  + .tei
  + .tmx
  + .xlsx, .ods
* Run annotation tools for automatic annotation:
  + TreeTagger (production)
  + UDPipe, Stanford NLP, Talismane... (prototypes)
* Manually annotate or correct automatic annotations through the GUI
* Extract various textual observables based on lexical patterns for searching and counting (for statistical models)
* Build various sub-corpora and partitions
* Build various visualisations and data tables from statistical analysis of observables inside sub-corpora/partitions

# Does the tool allow you to export the results?

* As spreadsheets [results data]
* As R data frames [results data]
* As images (vector or bitmap) [data visualisations]
* As XML-TEI [texts and annotations]

# Is this tool a statistical model (or a set thereof)?

* No

# Does this tool include statistical models?

* Yes

# Can you easily train and use your own statistical model with this tool?

Yes, TreeTagger can be trained from the GUI to build new language models from a corpus.

# XML-TEI compatibility

* All input formats are converted to XML-TEI TXM pivot format internaly, including any XML-TEI as input
* The XML-TEI TXM format is a specific TEI extension that can be normalized to TEI on export.

# How is an input document to be structured for the tool?

The tool imports a set of documents as a corpus.  
Metadata can be associated with each document through a spreadsheet file.

The structure of documents depends on their type:

* Written texts [the structure model is TEI format] : any XML-TEI structure encoding, links to fac-simile pages images, hyperlinks, etc.
* Speech transcriptions (eg interview recording) [the structure model is Transcriber software file format]: link to media file, episodes, speech turns and word level timing with media files
* Multilingual/parallel [the structure model is TMX format]: alignments on any structure (div, p...)
* As a table (eg answers to a survey, tweets set) [the structure model is a spreadsheet]: some columns for metadata and some columns for textual content

# Methods and features

|  |  |  |  |
| --- | --- | --- | --- |
| **Quick orientation for contributors** | **Task (what you want to achieve with that method)** | **Method (usually more generic than task)** | **Features/ Metric/ Formalism, Tagset** |
| **NLP** | Recognize borders of tokens and sentences | Tokenization & Sentence splitting | Features: Tokens; Sentence borders |
| **NLP** | Give the basic  dictionary  form of a  token | Lemmatization | Features: Lemmas, aka dictionary forms  of tokens |
| **NLP** | Determine  part of speech  and relevant morphological  categories | Morphological tagging | Features: Parts of speech;  Morphological categories; Tagset:  TreeTagger |
| **Corpus management** | Build a subcorpus | Build a corpus configuration by selecting a set of texts or a set of text parts  Corpus configuration | Features: selection of texts by a combination of their properties ; selection of text structures by a combination of their properties  Formalism: CQL |
| **Corpus management** | Build a partition | Build a contrastive corpus configuration by selecting a set of texts or a set of text parts  Corpus configuration | Features: selection of texts by a combination of their properties ; selection of text structures by a combination of their properties ; the sum of selections is always the whole corpus (resp.) the whole text  Formalism: CQL |
| **Corpus extraction** | Search in a text corpus | Linear text search with Corpus Query Processor  Quantification | Features: Lexical pattern observables extractions ; Text snippets matching a query ; Word lists matching a query  Formalism: CQL |
| **Corpus extraction** | Search in a parsed text corpus (treebank) | Syntactic tree search  Quantification | Features: Syntactic trees matching a query ; Text snippets matching a query ; Word lists matching a query ; Lexical pattern observables extractions  Formalism: TIGER Search, CQL |
| **Corpus extraction** | Search in an annotated text corpus (co-reference) | Co-reference chain search  Quantification | Features: Co-reference chains Units, Relations or Schema matching a query  Formalism: URSQL |
| **Corpus analysis** | Tabulate the contexts of occurrences of word patterns | Concordance analysis  Qualitative analysis | Features: Tabulate word pattern occurrences with reference, left context, occurrence and right context columns ; Bring similar contexts together by multiple sort on any column ; Link to the text edition (to read) with occurrence highligthed ; Link to audio/video stream (to play) synchronized with occurrence ;  Formalism: CQL, TIGER Search, URSQL |
| **Corpus analysis** | Show word patterns occurrences density within the corpus | Progression analysis Analytical views production and interaction  Syntagmatic analysis  Qualitative analysis | Features: Visualisation (interactive graphic) showing a density curve for each word pattern searched ; link to concordances  Formalism: CQL, TIGER Search, URSQL |
| **Corpus analysis** | Tabulate word properties of occurrences of word patterns | Hierarchical Index analysis  Paradigmatic analysis  Quantitative analysis | Features: Tabulate word pattern properties values (eg combine word form, pos or lemma) of occurrences with their frequency  Formalism: CQL, TIGER Search, URSQL |
| **Corpus analysis** | Tell which words typically co-occur with a given word | Collocations analysis  Syntagmatico-Paradigmatic analysis  Quantitative analysis | Metric: Statistics expressing the mutual attraction of two words or short text segments (specificity statistical measure)  Formalism: CQL |
| **Corpus analysis** | Tell which words typically occur within a sub-corpus or inside parts of a partition | Keywords analysis of a partition or subcorpus  Paradigmatic analysis  Quantitative analysis | Metric: Statistics expressing the attraction of a word or a short text segment with a sub-corpus of the whole corpus or with parts of a partition of the corpus (specificity statistical measure)  Visualisation: Bar plot showing statistic value for each word pattern searched  Formalism: CQL |
| **Corpus analysis** | Show similarities between texts or structures based on word pattern frequencies | Correspondence factor analysis of a partition  Analytical views production and interaction  Paradigmatic analysis  Quantitative analysis | Metric: khi2 Features: factorial coordinates, contributions, quality of representation  Visualisation: factorial planes  Formalism: CQL |
| **Corpus analysis** | Show hierarchy of similarities between texts or structures based on word pattern frequencies | Hierarchical Cluster analysis of a partition  Analytical views production and interaction  Paradigmatic analysis  Quantitative analysis | Metric: khi2 Features: clusters  Visualisation: clusters hierarchy, factorial planes  Formalism: CQL |
| **Corpus analysis** | Give summary statistics of a corpus or selected texts | Various countings | Features: Token properties (form, pos, lemma) list and frequencies ; Structures properties list and frequencies |
| **Annotation tool** | Manually annotate your tokens, e. g. with pos | Manual token annotation/annotation editing  Annotation through concordance  Qualitative analysis | Features: Labeled token spans or tokens Formalism: CQL |
| **Annotation tool** | Semi-automatically annotate your tokens, e. g. with pos | Semi-automatical token annotation/annotation editing  Annotation through scripts  Qualitative analysis | Features: Labeled token spans or tokens Formalism: CQL |
| **Annotation tool** | Manually annotate your texts, e. g. with co-reference chain units | Manual token spans annotation/annotation editing  Annotation through text reading  Qualitative analysis | Features: Labeled token spans or tokens  Formalism: Unit-Relation-Schema (URS) |

# Which human languages does the tool support (as of February 2023)?

The tool is fully language independent:

* any Unicode script
* any character collation

Automatic linguistic annotations depend on NLP tools used or annotations imported from source.

For example, TreeTagger can lemmatize English, German, Italian, Spanish, Russian, Latin, Greek, Ancien Greek, Written Modern French, Spoken Modern French, Written Old French, Written Middle French, Written Early Modern French, etc. (see TreeTagger software description for all available languages)

# Which other tools from this list does it integrate?

* TreeTagger
* UDPipe (prototype)
* CQP (Corpus Query Processor)
  + CQP as a library version (<https://gitlab.huma-num.fr/txm/txm-cqp>)
* R
  + textometry package (<https://cran.r-project.org/web/packages/textometry/index.html>)
  + FactoMineR package (<https://cran.r-project.org/web/packages/FactoMineR/index.html>)

# What can this tool do for you?

TXM can be used to create and analyze four main types of textual corpora:

* written texts (possibly aligned with fac-simile)
* speech transcriptions (possibly synchronized with media files at word level)
* multilingual/parallel corpora
* and corpora encoded in a table (e.g. answers to a survey, tweets)

TXM can automatically annotate texts with part of speech and lemma with the help of TreeTagger (or other NLP tools) or use linguistic annotations already encoded into the sources.

One specificity of TXM is that it can help analyse any kind of TEI encoded corpus by possibly adapting (with the help of an XSLT transformation stylesheets library) the texts encodings to its internal XML-TEI TXM format and data model if needed.

Another specificity of TXM is that it can help the user to process progressively his corpus from plain text to richly structured XML-TEI encoded text. It offers a continuous range of import modules covering the most frequently used standard formats:

* TXT: for any plain text coming from word processors, PDFs, websites, etc.;
* XML: for slightly structured texts (only sentences or paragraphs for example) or linguistically enriched (with XML tags that encode certain words with lexical properties);
* TEI: for texts encoded according to the recommendations of the TEI consortium and which are intended to be capitalized for long-term projects, shared with other initiatives or compatible with archiving systems.

A project can apply TXM to its encoded data progressively, from the simplest way (and most limited to use) to the most complex way (and richest). TXM therefore makes it possible to adapt the costs of corpus encoding according to the real needs of the study, especially when these needs are discovered as the corpus is being analyzed. Under these conditions, TXM assists both the encoding activity and the exploitation of corpora.

When using TEI, TXM helps to build and host online rich HTML 5 + CSS 3 + Javascript based text editions, with possible fac-simile synoptic views (when page image links are simply encoded in <pb/> tags in the documents).

TXM supports various flavors of TEI P4/P5 encoding practices: Perseus, TextGrid, Base de Français Médiéval (BFM), BVH Epistemon, etc. TEI sources are preprocessed by several XSL stylesheets that are delivered with TXM. Other stylesheets are available in the TXM XSL stylesheets library online: http://sourceforge.net/projects/txm/files/library/xsl.

TXM provides four types of tools to analyze a corpus:

* 1) Corpus configuration tools to build specific subcorpora or corpus partitions, based on sets of texts or sets of text structures (inside a text)
* 2) Semi-manual annotation tools at word or word sequence level: through concordance or text reading
* 3) Qualitative analytic tools like word pattern lists, concordances or progression views
* 4) Quantitative analytic tools like word pattern cooccurrents analysis, keyword analysis, and contrastive tools like correspondence factorial analysis or cluster analysis

Every analytic tool uses word patterns expressed with search queries on annotations with the help of different search engines (Corpus Query Processor, TIGER Search or URSQL). For example, word patterns expressed with the CQL query language:

* "aiming": to simply search for the word 'aiming'
* [pos="VERB" & word=".\*ing"]: to search for verb forms ending in ".ing" (where Part of Speech annotation is present)
* [lemma="group"] []{0,3} [pos="VERB" & word=".\*ing"]: to search for the collocation <group lemma> followed by a <verb with progressive aspect> with at most 3 words in between

TXM makes an analysis working session more fluid by interconnecting the tools as much as possible through hypertextual links located in their result views. For example: a double-click from a density curve of a word pattern in a text (general view of Progression) to the Concordance of this pattern (focused contextual reading), then from a line of this concordance to an Edition page where the occurrence is highlighted (precise reading of the text), then from this page to the Playback of the video corresponding to the transcription at the time the occurrence is pronounced (verification in the primary source).

All statistical models used by the quantitative tools are implemented and documented in R.

Results can be exported as spreadsheets (for data) or images (for visualisations). Texts and annotations can be exported in XML-TEI format. Corpora can be exported as a .txm file to be loaded in another TXM or uploaded to a TXM portal server version for online access and analysis.

TXM tools can be driven by Groovy scripts (a Java based scripting language) or by R scripts.

# Recommended tutorials

|  |  |  |
| --- | --- | --- |
| **Title** | **Provider/Author** | **URL** |
| Explore, play, analyse your corpus with TXM: A short introduction of TXM | José Calvo and Silvia Gutiérrez, April 17 2014 | <http://dhd-blog.org/?p=3384> |
| TXM-Kurzreferenz | Christof Schöch, July 4 2014 | <https://zenodo.org/record/10769> |
| Videocast of TXM 0.6 initiation Workshop (in French) | Bénédicte Pincemin, September 27 2012 | <https://txm.gitpages.huma-num.fr/textometrie/html/enregistrement_atelier_initiation_TXM_fr.html> |
| Videocast of importing corpora into TXM 0.8.1 training Workshop (in French) | Alexey Lavrentev, March 2020 | <https://txm.gitpages.huma-num.fr/textometrie/Documentation#2-atelier-de-formation-import-de-corpus-dans-txm-08-dalexey-lavrentev-en-mars-2020> |

# Papers

Béranger, M., Heiden, S., & Lavrentiev, A. (2015). Reengineering Akkadian Tablets with TEI and TXM for Linguistic Analysis. *TEI Conference and Members’ Meeting*, 36. https://halshs.archives-ouvertes.fr/halshs-01318713

Grobol, L., Landragin, F., & Heiden, S. (2018, October). XML-TEI-URS: using a TEI format for annotated linguistic resources. *CLARIN Annual Conference 2018*. https://hal.archives-ouvertes.fr/hal-01827563

Heiden, S. (2010). The TXM Platform: Building Open-Source Textual Analysis Software Compatible with the TEI Encoding Scheme. In R. Otoguro, K. Yoshimoto, K. Ishikawa, H. Umemoto, & Y. Harada (Eds.), *24th Pacific Asia Conference on Language, Information and Computation* (pp. 389–398). Institute for Digital Enhancement of Cognitive Development, Waseda University. https://halshs.archives-ouvertes.fr/halshs-00549764

Heiden, S. (2018). Annotation-based Digital Text Corpora Analysis within the TXM Platform. *Fourteenth International Conference on the Statistical Analysis of Textual Data*, *1*, 367–374. https://hal.archives-ouvertes.fr/hal-02015898

Heiden, S. (2019). Coping With The Complexity Of The TXM Platform Annotation Services With A Unified TEI Encoding Framework [Data set]. In *Abstracts of the Digital Humanities Conference*. Digital Humanities Conference, Utrecht. DataverseNL. https://doi.org/https://doi.org/10.34894/YYKDMM

Lavrentiev, A., & Heiden, S. (2012). The TXM Portal Software giving access to Old French Manuscripts Online. *7th International Conference on Language Resources and Evaluation (LREC)*, 29–35. https://halshs.archives-ouvertes.fr/halshs-00759361

Pincemin, B., Heiden, S., & Decorde, M. (2020). Textometry on Audiovisual Corpora. In P. M. & P. RATINAUD (Ed.), *15th International Conference on Statistical Analysis of Textual Data JADT 2020*. University of Toulouse. https://halshs.archives-ouvertes.fr/halshs-02779055

Pincemin, B., Mayaffre, D., Heiden, S., & Weyl, P. (2016). Génétique mémorielle. Shoah, mémoire et ADT. In D. Mayaffre, C. Poudat, L. Vanni, V. Magri, & P. Follette (Eds.), *JADT 2016 - Statistical Analysis of Textual Data: Vol. volume 2* (pp. 495–506). Presses de FacImprimeur. https://hal.archives-ouvertes.fr/hal-01361988

Quignard, M., Heiden, S., Landragin, F., & Decorde, M. (2018). Textometric Exploitation of Coreference-annotated Corpora with TXM: Methodological Choices and First Outcomes. In M. M. Domenica Fioredistella IEZZI Livia CELARDO (Ed.), *Fourteenth International Conference on the Statistical Analysis of Textual Data* (pp. 610–615). UniversItalia. https://hal.archives-ouvertes.fr/hal-01814858

# Tool name

Poetrylab-API

# Version

Unknown

# Version Date

2021-09-17

# Which Operating Systems does it work on?

* Windows
* Unix
* MacOs
* Linux

# License

* Apache 2.0

# Distribution

* REST API
* Library for Python

# Where to download, how to install, documentation, user guides

* <https://github.com/linhd-postdata/poetrylab-api>

# User interface

* API-REST

# Docker instance

* Yes

# How does the tool process your text?

* Automatically adds some labels (annotation) to your texts.

# Does the tool allow you to export the results?

* Yes

# Is this tool a statistical model (or a set thereof)?

* Yes

# Does this tool include statistical models? (Irrelevant, erase if this is a statistical model)

* No

# Can you easily train and use your own statistical model with this tool?

* Yes

# XML-TEI compatibility

* The tool does not support XML-TEI at all.

# How is an input document to be structured for the tool?

Plain text

# Methods and features

|  |  |  |  |
| --- | --- | --- | --- |
| **Quick orientation for contributors** | **Task (what you want to achieve with that method)** | **Method (usually more generic than task)** | **Features/ Metric/ Formalism, Tagset** |
| **NLP** | Automatic annotation of Spanish poetry | Tokenization, syllabification | rhyme, metric, part of speech, stanza type |
| **NLP** | Automatic enjambment detection | Rule-based enjambment detection | enjambment |

# Which human languages does the tool support (as of February 2023)?

|  |  |
| --- | --- |
| Language | Variety (geographical, or temporal if not modern) |
| [Spanish](https://en.wikipedia.org/wiki/Vietnamese_language) |  |

# Which other tools from this list does it integrate?

* Rantanplan

# What can this tool do for you?

The PoetryLab API is a REST API that provides an analysis  endpoint to extract information about the poems. It can annotate Spanish poetry with rhyme, metric, enjambment, and part of speech.

Only the method POST is allowed at the /analysis endpoint. A playground interface documenting the API can be found at the /ui endpoint.

# Recommended tutorials

|  |  |  |
| --- | --- | --- |
| **Title** | **Provider/Author** | **URL** |
| Documentation | Postdata | https://rantanplan.readthedocs.io/en/latest/readme.html |
| Readme | Postdata | https://github.com/linhd-postdata/poetrylab-api |

# Tool name

Rantanplan

# Version

0.8.0

# Version Date

2023-12-19

# Which Operating Systems does it work on?

* Windows
* Unix
* MacOs
* Linux

# License

* Apache Software License 2.0

# Distribution

* Library for Python

# Where to download, how to install, documentation, user guides

* <https://github.com/linhd-postdata/rantanplan>
* Pip install rantanplan

# User interface

* none

# Docker instance

* No

# How does the tool process your text?

* Automatically adds some labels (annotation) to your texts.

# Does the tool allow you to export the results?

* Yes

# Is this tool a statistical model (or a set thereof)?

* No

# Does this tool include statistical models? (Irrelevant, erase if this is a statistical model)

* No

# Can you easily train and use your own statistical model with this tool?

* Yes

# XML-TEI compatibility

* The tool does not support XML-TEI at all.

# How is an input document to be structured for the tool?

Plain text

# Methods and features

|  |  |  |  |
| --- | --- | --- | --- |
| **Quick orientation for contributors** | **Task (what you want to achieve with that method)** | **Method (usually more generic than task)** | **Features/ Metric/ Formalism, Tagset** |
| **NLP** | Automatic annotation of Spanish poetry | Tokenization, syllabification | rhyme, metric, part of speech, stanza type |

# Which human languages does the tool support (as of February 2023)?

|  |  |
| --- | --- |
| Language | Variety (geographical, or temporal if not modern) |
| Spanish |  |

# Which other tools from this list does it integrate?

* None.

# What can this tool do for you?

It can annotate Spanish poetry with rhyme, meter, and part of speech.

# Recommended tutorials

|  |  |  |
| --- | --- | --- |
| **Title** | **Provider/Author** | **URL** |
| Documentation | Javier de la Rosa, Álvaro Pérez, Aitor Díaz | https://rantanplan.readthedocs.io/en/latest/ |

# Papers

Pérez Pozo, Á., de la Rosa, J., Ros, S., González-Blanco, E., Hernández, L., & de Sisto, M. (2022). A bridge too far for artificial intelligence?: Automatic classification of stanzas in Spanish poetry. *Journal of the Association for Information Science and Technology*, *73*(2), 258-267. <https://doi.org/10.1002/asi.24532>

Rosa, J. de la, Pérez, Á., Hernández, L., Ros, S., & González-Blanco, E. (2020). Rantanplan, Fast and Accurate Syllabification and Scansion of Spanish Poetry. *Procesamiento del Lenguaje Natural*, *65*(0), Article 0.

# Tool name

Averell

# Version

1.2.3

# Version Date

2023-12-11

# Which Operating Systems does it work on?

* Windows
* Unix
* MacOs
* Linux

# License

* Apache Software License 2.0

# Distribution

* Local application
* Library for Python

# Where to download, how to install, documentation, user guides

* https://github.com/linhd-postdata/averell

# User interface

* none
* GUI
* Command line

# Docker instance

* Yes

# How does the tool process your text?

* Automatically adds some labels (annotation) to your texts.
* Downloads an annotated corpus and reconcile different TEI entities to provide a unified JSON or TEI output at the desired granularity.

# Does the tool allow you to export the results?

* Yes

# Is this tool a statistical model (or a set thereof)?

* No

# Does this tool include statistical models? (Irrelevant, erase if this is a statistical model)

* Yes
* No

# Can you easily train and use your own statistical model with this tool?

* Yes
* No

# XML-TEI compatibility

* The tool requires or accepts text in XML-TEI, adds its markup in XML-TEI, and ensures valid XML-TEI on the output.
* The tool requires or accepts text in XML-TEI but the original markup gets lost and/or the output will not be integrated into the original XML-TEI format.

# How is an input document to be structured for the tool?

The input format does not matter but the following information must be given as a bare minimum:

* Corpus name
* Annotation type (manual, automatic, or none)
* Author name
* Poem title
* Poem text split into stanzas

# Methods and features

|  |  |  |  |
| --- | --- | --- | --- |
| **Quick orientation for contributors** | **Task (what you want to achieve with that method)** | **Method (usually more generic than task)** | **Features/ Metric/ Formalism, Tagset** |
| **NLP** | Automatic poetic annotation for Spanish poetry (rhyme, stanza type, metric) | Syllabify, tokenization & Sentence splitting, stanza detection. | Features: Tokens, syllables, Sentence borders, stanzas, rhymes |
| **Corpus** | Download one or several corpora and export to a common format | Text parsing |  |

# Which human languages does the tool support (as of February 2023)?

For automatic annotation:

* Spanish
* For automatic integration

|  |  |
| --- | --- |
| Language | Variety (geographical, or temporal if not modern) |
| Any (it is language-agnostic) |  |

# Which other tools from this list does it integrate?

* Rantanplan.

# What can this tool do for you?

Averell is a python library and command line interface that facilitates working with existing repositories of annotated poetry. Averell is able to download an annotated corpus and reconcile different TEI entities to provide a unified JSON output at the desired granularity. That is, for their investigations some researchers might need the entire poem, poems split line by line, or even word by word if that is available. Averell allows to specify the granularity of the final generated dataset, which is a combined JSON with all the entities in the selected corpora. Each corpus in the catalog must specify the parser to produce the expected data format.

# Recommended tutorials

|  |  |  |
| --- | --- | --- |
| **Title** | **Provider/Author** | **URL** |
| Readme, basic usage and installation | Postdata | https://averell.readthedocs.io/en/latest/ |

# Tool name

Alberti-stanzas

# Version

Unknown

# Version Date

2022

# Which Operating Systems does it work on?

* Windows
* Unix
* MacOs
* Linux

# License

* CC-BY 4.0

# Distribution

* Language model (transformers)

# Where to download, how to install, documentation, user guides

* https://huggingface.co/alvp/alberti-stanzas

# User interface

* none
* GUI

# Docker instance

* Yes

# How does the tool process your text?

* Automatic annotation of stanza type for Spanish poems.

# Does the tool allow you to export the results?

* No

# Is this tool a statistical model (or a set thereof)?

* It is a fine-tuned transformers model

# Can you easily train and use your own statistical model with this tool?

* Yes

# XML-TEI compatibility

* The tool does not support XML-TEI at all.

# How is an input document to be structured for the tool?

Plain text.

# Methods and features

We use a tool to solve a task. The tool is approaching this task with methods to provide some features or compute values of some metrics for the given text. Features typically enrich the input text (either in-line or stand-off) with labels or filter the text according to a query. Assigned labels typically belong to a given tagset (e. g. Universal Dependencies or Penn Treebank) or/and pertain to a formalism such as HPSG or minimalist syntax). Filters typically pertain to a query language (also a formalism).

|  |  |  |  |
| --- | --- | --- | --- |
| **Quick orientation for contributors** | **Task (what you want to achieve with that method)** | **Method (usually more generic than task)** | **Features/ Metric/ Formalism, Tagset** |
| **NLP** | Automatic annotation of stanzas | Word embeddings | Stanza type |

# Which human languages does the tool support (as of February 2023)?

|  |  |
| --- | --- |
| Language | Variety (geographical, or temporal if not modern) |
| Spanish |  |

# Which other tools from this list does it integrate?

* None.

# What can this tool do for you?

This is a language model trained with poetic texts that returns word embeddings. It has been trained for the stanzas detection task port Spanish poetry. It can return the stanza type of a Spanish poem.

# Recommended tutorials

|  |  |  |
| --- | --- | --- |
| **Title** | **Provider/Author** | **URL** |
| UI/Docker | Postdata | https://github.com/linhd-postdata/alberti-stanzas-streamlit |
| Metrics and instructions | Postdata | https://github.com/linhd-postdata/alberti-stanzas-api |
| Model | Postdata | https://huggingface.co/alvp/alberti-stanzas |

# Papers

de la Rosa, J., Pozo, Á. P., Ros, S., & González-Blanco, E. (2023). *ALBERTI, a Multilingual Domain Specific Language Model for Poetry Analysis*. <https://doi.org/10.48550/ARXIV.2307.01387>

Pérez Pozo, Á., de la Rosa, J., Ros, S., González-Blanco, E., Hernández, L., & de Sisto, M. (2022). A bridge too far for artificial intelligence?: Automatic classification of stanzas in Spanish poetry. *Journal of the Association for Information Science and Technology*, *73*(2), 258-267. <https://doi.org/10.1002/asi.24532>