EvaLatin 2024

-

Guidelines

Version 1.2

February 11, 2024

Rachele Sprugnoli[2], Federica Iurescia[1], Marco Passarotti[1]
1. CIRCSE Research Centre, Università Cattolica del Sacro Cuore
Largo Agostino Gemelli 1, 20123 Milan, Italy
federica.iurescia[AT]unicatt.it
marco.passarotti[AT]unicatt.it
2. Università di Parma
Viale D'Azeglio 85, Italy
rachele.sprugnoli[AT]unipr.it

Contents

Chapter 1

Introduction

EvaLatin 2024 is the third edition of the evaluation campaign of Natural Language Processing (NLP) tools for the Latin language. The campaign is designed with the aim of answering two main questions:

- How can we promote the development of resources and language technologies for the Latin language?
- How can we foster collaboration among scholars working on Latin and attract researchers from different disciplines?

EvaLatin 2020 [7] was organized around 2 tasks (i. e. Lemmatization and PoS Tagging); EvaLatin 2022 [8] was organized around 3 tasks (i. e. Lemmatization, PoS Tagging, and Features Identification). For EvaLatin 2024, we propose 2 tasks (i. e. Dependency Parsing and Emotion Polarity Detection). Shared data and a scorer are provided to the participants. Participants can choose to participate in either one or both tasks. The organizers rely on the honesty of all participants who might have some prior knowledge of part of the data used for evaluation not to unfairly use such knowledge.

EvaLatin 2024 is organized within the "Third Workshop of Language Technologies for Historical and Ancient Languages" (LT4HALA 2024), colocated at LREC-COLING 2024.[1] The workshop will be held in Turin, Italy, on 25th May, 2024. EvaLatin 2024 is organized by the CIRCSE research centre at the Università Cattolica del Sacro Cuore in Milan and the Università degli studi di Parma, Italy.

For any update, please check the LT4HALA 2024 website: `https://circse.github.io/LT4HALA/2024/`.

[1] `https://lrec-coling-2024.org/`

Chapter 2

Data

Dependency parsing is based on the Universal Dependencies (UD) framework.[1] No specific training data are released but participants are free to make use of any (kind of) resource they consider useful for the task, including the Latin treebanks already available in the UD collection. In this regard, one of the challenges of this task is to understand which treebank (or combination of treebanks) is the most suitable to deal with new test data.[2] Test data are both prose and poetic texts from different time periods. Some of the datasets include punctuation, some do not, as this is the actual state of the art for the Latin treebanks.
Also for the Emotion Polarity Detection task, no training data are released but the organizers provide an annotation sample, a manually created polarity lexicon and annotation guidelines (see Section 3.1.2). Also in this task, participants are free to pursue the approach they prefer, including unsupervised and/or cross-language ones (which promise to be the most efficient, given the lack of training data for Latin for this task). Test data are poetic texts from different time periods. Some of the datasets include punctuation, some do not, as for the data related to the task of Dependency Parsing. Both tasks aim to improve a state of the art that is not optimal. With regard to Dependency Parsing, UD treebanks currently show different degrees of harmonization, and Latin is not an exception in this respect. With regard to Emotion Polarity Detection, there are no available data for Latin yet, as this is unexplored territory for this language. The diversity of the data currently available for both tasks is an issue we are aware of, and that needs to be addressed. This evaluation campaigns aims at addressing this issue, and among the desired outcomes there are strategies to deal with it successfully.

2.1 Format

This section gives details on the format of test data for both the 2024 shared tasks.

[1] `https://universaldependencies.org/`
[2] See [4] for further details.

2.1.1 Dependency Parsing

Test data for the task of Dependency Parsing are distributed in the CoNLL-U format.[3] Following such format, the annotations are plain text files having the `.conllu` extension and encoded in UTF-8 containing:

- 2 comment lines starting with a hashtag (`#`): one line specifies the sentence unique code (`# sent_id`), while the other line reports the sentence string (`# text`).
- Lines composed of 10 tab-separated fields and containing either the UD-style annotation of a single-word token, or denoting a multi-word token. When values for a given field are not determined, an underscore (`_`) is used instead (there are no empty fields). The 10 fields of the CoNLL-U format are as follows:
  1. `ID`: numerical index of the word in the sentence. For single-word tokens, a progressive integer starting from 1 for each new sentence; for multi-word tokens, it is a range in the form of $i-(i+n)$, meaning that the token in the sentence string is split into $n+1$ words, with indices from i to $i+n$, each with its own analysis;
  2. `FORM`: word form as occurring in the sentence string. For multi-word tokens, the form of a single word is normalized, whereas it might be contracted, or otherwise subject to changes, in the univerbation (e. g. *imprimis* would be decomposed as *in* and *primis*);
  3. `LEMMA`: form conventionally representing the lexeme related to the word form;
  4. `UPOS`: universal PoS tag of the word;[4]
  5. `XPOS`: language-specific PoS of the word (as used e. g. in the same treebank prior to conversion to the UD annotation style);
  6. `FEATS`: list of morpholexical features of the word, either "universal" or "language-specific";[5]
  7. `HEAD`: index of the syntactic head of the word (possibly `0` if the word is the sentence root);
  8. `DEPREL`: universal syntactic dependency relation with respect to the `HEAD`;[6]
  9. `DEPS`: syntactic relations in the enhanced dependency graph;
  10. `MISC`: any other annotation.
- Blank lines marking boundaries between sentences and the end of the document.

An example of the CoNLL-U format is given in Figure 1.

In our test dataset, the fields `HEAD` and `DEPREL` are filled in with underscores. This format is used for the test data and participants are expected to produce the same format for the final evaluation. An example of the test data format is given in Figure 2.

[3] `https://universaldependencies.org/format.html`
[4] `https://universaldependencies.org/u/pos/all.html`
[5] `https://universaldependencies.org/ext-feat-index.html`
[6] `https://universaldependencies.org/ext-dep-index.html`

```
# sent_id = CaesBG4-A-01-607
# text = neque multum frumento sed maximam partem lacte atque pecore uiuunt multumque sunt in uenationibus
1   neque   neque   CCONJ   S   Polarity=Neg    2   cc  _   LiLaflcat=i
2   multum  multum  ADV M   Degree=Pos  10  advmod  _   LASLAVariant=2|LiLaflcat=i
3   frumento    frumentum   NOUN    A2  Case=Abl|Gender=Neut|InflClass=IndEurO|Number=Sing  10  obl _   LiLaflcat=n2
4   sed sed CCONJ   S   _   6   cc  _   LiLaflcat=i
5   maximam magnus  ADJ C1  Case=Acc|Degree=Abs|Gender=Fem|InflClass=IndEurA|Number=Sing    6   amod    _   LiLaflcat=n6
6   partem  pars    NOUN    A3  Case=Acc|Gender=Fem|InflClass=IndEurI|Number=Sing   10  obj _   LiLaflcat=n3
7   lacte   lac NOUN    A3  Case=Abl|Gender=Masc|InflClass=IndEurI|Number=Sing  10  obl _   LiLaflcat=n3
8   atque   atque   CCONJ   S   _   9   cc  _   LASLAVariant=1|LiLaflcat=i
9   pecore  pecus   NOUN    A3  Case=Abl|Gender=Neut|InflClass=IndEurX|Number=Sing  7   conj    _   LASLAVariant=1|LiLaflcat=n3
10  uiuunt  uiuo    VERB    B3  Aspect=Imp|InflClass=LatX|Mood=Ind|Number=Plur|Person=3|Tense=Pres|VerbForm=Fin|Voice=Act   0   root    _   LiLaflcat=v3
11-12   multumque   _   _   _   _   _   _   _   _
11  multum  multum  ADV M   Degree=Pos  15  advmod  _   LASLAVariant=2|LiLaflcat=i
12  que que CCONJ   S   _   15  cc  _   LiLaflcat=i
13  sunt    sum AUX B6  Aspect=Imp|InflClass=LatAnom|Mood=Ind|Number=Plur|Person=3|Tense=Pres|VerbForm=Fin  15  cop _   LASLAVariant=1|LiLaflcat=v6
14  in  in  ADP R   AdpType=Prep    15  case    _   LiLaflcat=i
15  uenationibus    uenatio NOUN    A3  Case=Abl|Gender=Fem|InflClass=IndEurX|Number=Plur   10  conj    _   LiLaflcat=n3
```

Figure 1: Example of CoNLL-U file.

```
# sent_id = CaesBG4-A-01-607
# text = neque multum frumento sed maximam partem lacte atque pecore uiuunt multumque sunt in uenationibus
1   neque   neque   CCONJ   S   Polarity=Neg    _   _   _   LiLaflcat=i
2   multum  multum  ADV M   Degree=Pos  _   _   _   LASLAVariant=2|LiLaflcat=i
3   frumento    frumentum   NOUN    A2  Case=Abl|Gender=Neut|InflClass=IndEurO|Number=Sing  _   _   _   LiLaflcat=n2
4   sed sed CCONJ   S   _   _   _   _   LiLaflcat=i
5   maximam magnus  ADJ C1  Case=Acc|Degree=Abs|Gender=Fem|InflClass=IndEurA|Number=Sing    _   _   _   LiLaflcat=n6
6   partem  pars    NOUN    A3  Case=Acc|Gender=Fem|InflClass=IndEurI|Number=Sing   _   _   _   LiLaflcat=n3
7   lacte   lac NOUN    A3  Case=Abl|Gender=Masc|InflClass=IndEurI|Number=Sing  _   _   _   LiLaflcat=n3
8   atque   atque   CCONJ   S   _   _   _   _   LASLAVariant=1|LiLaflcat=i
9   pecore  pecus   NOUN    A3  Case=Abl|Gender=Neut|InflClass=IndEurX|Number=Sing  _   _   _   LASLAVariant=1|LiLaflcat=n3
10  uiuunt  uiuo    VERB    B3  Aspect=Imp|InflClass=LatX|Mood=Ind|Number=Plur|Person=3|Tense=Pres|VerbForm=Fin|Voice=Act   _   _   _   LiLaflcat=v3
11-12   multumque   _   _   _   _   _   _   _   _
11  multum  multum  ADV M   Degree=Pos  _   _   _   LASLAVariant=2|LiLaflcat=i
12  que que CCONJ   S   _   _   _   _   LiLaflcat=i
13  sunt    sum AUX B6  Aspect=Imp|InflClass=LatAnom|Mood=Ind|Number=Plur|Person=3|Tense=Pres|VerbForm=Fin  _   _   _   LASLAVariant=1|LiLaflcat=v6
14  in  in  ADP R   AdpType=Prep    _   _   _   LiLaflcat=i
15  uenationibus    uenatio NOUN    A3  Case=Abl|Gender=Fem|InflClass=IndEurX|Number=Plur   _   _   _   LiLaflcat=n3
```

Figure 2: Example of the test data format.

2.1.2 Emotion Polarity Detection

Test data for the task of Emotion Polarity Detection are distributed in .tsv format: the first column contains a sentence ID and the second the text to be tagged. An example is given in Figure 3.

```
100 uelox amoenum saepe Lucretilem mutat Lycaeo Faunus
et igneam defendit aestatem capellis usque meis pluuios
que uentos
101 inpune tutum per nemus arbutos quaerunt latentis et
thyma deuiae olentis uxores mariti nec uiridis metuunt
colubras nec Martialis haediliae lupos utcumque dulci
Tyndari fistula ualles et Vsticae cubantis leuia
personuere saxa
102 di me tuentur dis pietas mea et musa cordi est
103 hic tibi copia manabit ad plenum benigno ruris
honorum opulenta cornu
104 hic in reducta ualle caniculae uitabis aestus et
fide Teia dices laborantis in uno Penelopen uitream que
Circen
105 hic innocentis pocula Lesbii duces sub umbra nec
Semeleius cum Marte confundet Thyoneus proelia nec
metues proteruum suspecta Cyrum ne male dispari
incontinentis iniciat manus et scindat haerentem coronam
crinibus inmeritam que uestem
```

Figure 3: Example of the data format for the Emotion Polarity Detection task.

2.2 Test Data

2.2.1 Dependency Parsing

Texts provided as test data for the Dependency Parsing task are by 2 Classical authors (Seneca and Tacitus) for a total of more than 13,000 tokens. Each author is taken as specimen of one specific text genre: Seneca for poetry, more specifically for tragedy, with *Hercules Furens* (more than 7,000 tokens), composed in 1st century AD; Tacitus for prose, more specifically historical and ethnographic treatise, with *Germania* (nearly 6,000 tokens), written in 1st century AD. Data are taken from the LASLA corpus [3], a linguistic resource manually annotated since 1961 by the Laboratoire d'Analyse Statistique des Langues Anciennes (LASLA) at the University of Liège[7], Belgium. Original data were converted into the annotation formalism of the UD project and manually annotated for dependency relations.

AUTHOR	TEXT	TOKENS
Seneca	Hercules Furens	7,711

Table 1: Test data for poetry.

AUTHOR	TEXT	TOKENS
Tacitus	Germania	5,669

Table 2: Test data for prose.

2.2.2 Emotion Polarity Detection

Texts provided as test data for the Emotion Polarity Detection task are by 3 authors for a total of 297 sentences (around 100 sentences for each author):

- Seneca, with the final part (lines 1,175-1,344)[8] of the tragedy *Hercules Furens*, composed in 1st century AD;
- Horace, with 16 odes (4 for each book that makes up *Carmina*), composed in 1st century AD;
- Giovanni Pontano, with 12 poems taken from the work *Neniae*, composed in the 15th century.

Data by Seneca and Horace are taken from the LASLA corpus, while texts by Pontano are taken from the *Poeti d'Italia in lingua latina* website.[9] For this reason, Pontano's text has punctuation while those of Seneca and Horace do not.

[7] `http://web.philo.ulg.ac.be/lasla/textes-latins-traites/`

[8] Line numbers according to the following edition: Fitch, J.G. (2018). *Seneca. Tragedies, Volume I: Hercules. Trojan Women. Phoenician Women. Medea. Phaedra.* Cambridge (MA): Harvard University Press.

[9] `https://www.poetiditalia.it/public/`

AUTHOR	**TEXT**	**SENTENCES**
Seneca	Hercules Furens (lines 1,175-1,344)	103

Table 3: Test data by Seneca.

AUTHOR	**ODE (BOOK_POEM)**	**# SENTENCES**
Horace	I_2	7
Horace	I_14	8
Horace	I_28	9
Horace	I_38	2
Horace	II_3	6
Horace	II_11	7
Horace	II_14	3
Horace	II_16	10
Horace	III_2	5
Horace	III_10	4
Horace	III_18	2
Horace	III_24	7
Horace	IV_1	11
Horace	IV_10	1
Horace	IV_12	8
Horace	IV_13	6
TOTAL		**96**

Table 4: Test data by Horace.

AUTHOR	**NENIAE**	**SENTENCES**
Pontano	I	8
Pontano	II	11
Pontano	III	9
Pontano	IV	14
Pontano	V	6
Pontano	VI	7
Pontano	VII	11
Pontano	VIII	5
Pontano	IX	4
Pontano	X	9
Pontano	XI	8
Pontano	XII	6
TOTAL		**98**

Table 5: Test data by Pontano.

Chapter 3

Tasks

Participants can choose to participate in either one or both tasks described in this Chapter.

3.1 Tasks

This Section provides details on the two tasks included in EvaLatin 2024.

3.1.1 Dependency Parsing

Dependency Parsing is based on the UD framework. The aim of the task is to provide syntactic analysis of Latin texts. The output .conllu file provided by the participants shall have the indications of the syntactic head and of the dependency relations in the fields 7 (`HEAD`) and 8 (`DEPREL`) respectively. The set of labels for dependency relations adopted by the UD framework is as follows (relations are listed in alphabetical order):

- `acl`: a clausal modifier of a noun (`acl`) is a finite or non-finite clause that modifies a nominal.[1]
- `advcl`: an adverbial clause modifier (`advcl`) is a clause that modifies a verb or other predicate (e. g. an adjective) not introducing a core argument.[2]
- `advmod`: an adverbial modifier (`advmod`) is a non-clausal adverb or adverbial phrase that modifies a predicate or a modifier word.[3]
- `amod`: an adjectival modifier of a nominal is a phrase that serves to modify the nominal.[4]

[1] `https://universaldependencies.org/u/dep/acl.html`
[2] `https://universaldependencies.org/u/dep/advcl.html`
[3] `https://universaldependencies.org/u/dep/advmod.html`
[4] `https://universaldependencies.org/u/dep/amod.html`

- `appos`: an appositional modifier (`appos`) is a nominal immediately following the first noun that serves to define, modify, name, or describe that noun.[5]

- `aux`: an auxiliary of a clause (`aux`) is a function word associated with a verbal predicate that expresses categories such as tense, mood, aspect, voice or evidentiality.[6]

- `case`: the relation of case marking (`case`) is used for any case-marking element treated as a separate syntactic word (including prepositions, postpositions, and clitic case markers).[7]

- `cc`: the relation of coordinating conjunction (`cc`) holds between a conjunct and a word that links words or larger constituents without syntactically subordinating one to the other (that is, a word with pos `CCONJ`).[8]

- `ccomp`: a clausal component (`ccomp`) is a dependent clause that functions as the object of a clause.[9]

- `clf`: a classifier (`clf`) is a function word that reflects some kind of conceptual classification of nouns, based principally on features of their referents.[10]

- `compound`: this relation is used to analyze combinations of lexemes that morphosyntactically behave as single words (that is, compounds).[11]

- `conj`: this relation holds between elements that lie syntactically on the same hierarchical level. The conjunct(s) may be connected via coordinating conjunctions or punctuation symbols.[12]

- `cop`: a copula (`cop`) is a function word used to link a subject to a nonverbal predicate.[13]

- `csubj`: a clausal subject (`csubj`) is a clausal syntactic subject of a clause.[14]

- `dep`: this relation is used when a more precise relation is impossible to determine, because of (e.g.) a weird grammatical construction, or a limitation in conversion or parsing software.[15]

[5] `https://universaldependencies.org/u/dep/appos.html`
[6] `https://universaldependencies.org/u/dep/aux_.html`
[7] `https://universaldependencies.org/u/dep/case.html`
[8] `https://universaldependencies.org/u/dep/cc.html`
[9] `https://universaldependencies.org/u/dep/ccomp.html`
[10] `https://universaldependencies.org/u/dep/clf.html`
[11] `https://universaldependencies.org/u/dep/compound.html`
[12] `https://universaldependencies.org/u/dep/conj.html`
[13] `https://universaldependencies.org/u/dep/cop.html`
[14] `https://universaldependencies.org/u/dep/csubj.html`
[15] `https://universaldependencies.org/u/dep/dep.html`

- `det`: a determiner (`det`) is a function word that links a nominal head to a word that modifies nouns or noun phrases and expresses the reference of the noun phrase in context (that is, words with pos `DET`).[16]
- `discourse`: a discourse element is an interjection or any other discourse element (intended as elements that serve to convey expressive functions).[17]
- `expl`: an expletive (`expl`) is a nominal that appears in an argument position of a predicate but does not satisfy any of the semantic roles of the predicate.[18]
- `fixed`: a fixed multiword expression (`fixed`) is used for fixed grammaticized expressions.[19]
- `flat`: a flat expression (`flat`) combines the elements of an expression where none of the immediate components can be identified as the sole head using standard substitution tests.[20]
- `dislocated`: a dislocated element is a fronted or postposed element that does not fulfill the usual core grammatical relations of a sentence.[21]
- `goeswith`: this relation links two or more parts of a word that are separated in texts that are not well edited.[22]
- `iobj`: an indirect object (`iobj`) is a nominal that is a core argument of the verb but is not its subject or (direct) object, typically the recipient of a ditransitive verb of exchange.[23]
- `list`: this relation is used for chains of comparable items.[24]
- `mark`: a marker (`mark`) is a function word marking a clause as subordinate to another clause.[25]
- `nmod`: a nominal modifier (`nmod`) is a nominal dependent of another noun or noun phrase.[26]
- `nummod`: a numeric modifier (`nummod`) of a noun is a number phrase that serves to modify the meaning of the noun with a quantity.[27]

[16] `https://universaldependencies.org/u/dep/det.html`
[17] `https://universaldependencies.org/u/dep/discourse.html`
[18] `https://universaldependencies.org/u/dep/expl.html`
[19] `https://universaldependencies.org/u/dep/fixed.html`
[20] `https://universaldependencies.org/u/dep/flat.html`
[21] `https://universaldependencies.org/u/dep/dislocated.html`
[22] `https://universaldependencies.org/u/dep/goeswith.html`
[23] `https://universaldependencies.org/u/dep/iobj.html`
[24] `https://universaldependencies.org/u/dep/list.html`
[25] `https://universaldependencies.org/u/dep/mark.html`
[26] `https://universaldependencies.org/u/dep/nmod.html`
[27] `https://universaldependencies.org/u/dep/nummod.html`

- `nsubj`: a nominal subject (`nsubj`) is the first core argument of a verb. It is a nominal (noun, pronoun, noun phrase) that is the syntactic subject of a clause.[28]
- `obj`: an object (`obj`) is the second most core argument of a verb after the subject. It is a nominal that denotes the entity acted upon or which undergoes a change of state or motion.[29]
- `orphan`: this relation is used to mark cases of head ellipsis where simple promotion (standard treatment of ellipsis) would result in an unnatural and misleading dependency relation.[30]
- `parataxis`: this relation is used to mark a clause placed side by side without any explicit relation with the head word.[31]
- `punct`: this relation is used for any piece of punctuation in a clause.[32]
- `reparandum`: this relation indicates disfluencies overridden in a speech repair.[33]
- `root`: this relation is used to mark the root node of a sentence.[34]
- `vocative`: a vocative is the term of address used to address the interlocutor.[35]
- `xcomp`: an open clausal component (`xcomp`) is a dependent clause that a) functions as a core argument of the verb; b) does not have an own subject; c) has an external argument that determines its subject.[36]

Please refer to UD guidelines for further details on the relations subtypes.[37]

3.1.2 Emotion Polarity Detection

The aim of the task is to identify the polarity conveyed by each sentence in the text, taking into consideration both the vocabulary used by the author and the images that are evoked in the text [6]. More specifically, the question to be answered is: which of the following classes best describes how are the emotions conveyed by the poet in the sentence under analysis?

- `positive`: the only emotions that are conveyed in the text are positive, or positive emotions are clearly prevalent;

[28] `https://universaldependencies.org/u/dep/nsubj.html`
[29] `https://universaldependencies.org/u/dep/obj.html`
[30] `https://universaldependencies.org/u/dep/orphan.html`
[31] `https://universaldependencies.org/u/dep/parataxis.html`
[32] `https://universaldependencies.org/u/dep/punct.html`
[33] `https://universaldependencies.org/u/dep/reparandum.html`
[34] `https://universaldependencies.org/u/dep/root.html`
[35] `https://universaldependencies.org/it/dep/vocative.html`
[36] `https://universaldependencies.org/u/dep/xcomp.html`
[37] `https://universaldependencies.org/ext-dep-index.html`

- `negative`: the only emotions that are conveyed in the text are negative, or negative emotions are clearly prevalent;
- `neutral`: there are no emotions conveyed by the text;
- `mixed`: lexicon and evoked images produce opposite emotions; it is not possible to find a clearly prevailing emotion polarity.

Sentences are provided in their original order in the source text.

Figure 4 shows an Ode of Horace (the same in Figure 3) annotated with the previously described labels in .tsv format. With respect to the test data format, the output provided by the participants should have an additional column with a label per sentence.

```
100 uelox amoenum saepe Lucretilem mutat Lycaeo Faunus
et igneam defendit aestatem capellis usque meis pluuios
que uentos   positive
101 inpune tutum per nemus arbutos quaerunt latentis et
thyma deuiae olentis uxores mariti nec uiridis metuunt
colubras nec Martialis haediliae lupos utcumque dulci
Tyndari fistula ualles et Vsticae cubantis leuia
personuere saxa   positive
102 di me tuentur dis pietas mea et musa cordi est
positive
103 hic tibi copia manabit ad plenum benigno ruris
honorum opulenta cornu   positive
104 hic in reducta ualle caniculae uitabis aestus et
fide Teia dices laborantis in uno Penelopen uitream que
Circen positive
105 hic innocentis pocula Lesbii duces sub umbra nec
Semeleius cum Marte confundet Thyoneus proelia nec
metues proteruum suspecta Cyrum ne male dispari
incontinentis iniciat manus et scindat haerentem coronam
crinibus inmeritam que uestem  mixed
```

Figure 4: Example of a text enhanced with Emotion Polarity annotation.

A sample annotation is provided: `https://github.com/CIRCSE/LT4HALA/blob/master/2024/GoldStandardv1-Horace.tsv`.

Chapter 4

Evaluation

Unlabeled test data are released according to the deadlines provided in the EvaLatin 2024 web page: `https://circse.github.io/LT4HALA/2024/EvaLatin`. After the assessment, the labels for the test data will be released to the participants.

4.0.1 Scorer

Two different scorers are used for the two shared tasks proposed at EvaLatin 2024.

4.0.2 Dependency Parsing

The scorer employed for the evaluation of the Dependency Parsing task is the one developed for the *CoNLL18 Shared Task on Multilingual Parsing from Raw Text to Universal Dependencies*[1] and available on the EvaLatin 2024 web page: `https://github.com/CIRCSE/LT4HALA/blob/master/2024/conll18_ud_eval.py`.
The command to run the script is the following (Python 3 required):

- `conll18_ud_eval.py -v gold_conllu_file system_conllu_file`

An example of the scorer's output is given in Figure 5. The evaluation starts by aligning the system-produced tokens to the gold standard one; given that we provide test data already sentence-splitted and annotated with morpho-grammatical information, the alignment for tokens, sentences, words, UPOS, UFeats and lemmas should be perfect (i. e. 100.00). Then, CLAS[2] and LAS[3] are evaluated.[4] Each metric is evaluated in terms

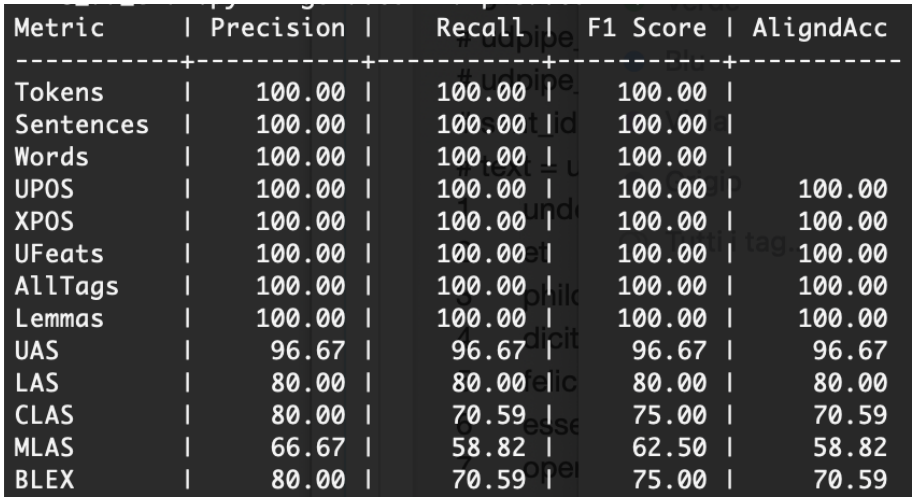
[1] `https://universaldependencies.org/conll18/evaluation.html`

[2] Content-Word Labeled Attachment Score (CLAS) is the labeled F1- score over all relations except relations for function words (`aux`, `case`, `cc`, `clf`, `cop`, `det`, `mark`) and for punctuation (`punct`). For greater details, see [5].

[3] Labeled Attachment Score (LAS) is the percentage of tokens assigned both the correct DEPREL and HEAD. For further details, see [2].

[4] As shown in Figure 5, the scorer computes also the Unlabeled Attachment Score (UAS), that is the percentage of tokens assigned the correct HEAD; the Morphology-aware Labeled

of Precision, Recall, F1 and Aligned Accuracy.[5]



Metric	Precision	Recall	F1 Score	AligndAcc
Tokens	100.00	100.00	100.00	
Sentences	100.00	100.00	100.00	
Words	100.00	100.00	100.00	
UPOS	100.00	100.00	100.00	100.00
XPOS	100.00	100.00	100.00	100.00
UFeats	100.00	100.00	100.00	100.00
AllTags	100.00	100.00	100.00	100.00
Lemmas	100.00	100.00	100.00	100.00
UAS	96.67	96.67	96.67	96.67
LAS	80.00	80.00	80.00	80.00
CLAS	80.00	70.59	75.00	70.59
MLAS	66.67	58.82	62.50	58.82
BLEX	80.00	70.59	75.00	70.59

Figure 5: Example of the scorer's output for the Dependency Parsing task.

As a baseline, we provide the scores obtained on the test data using UDPipe 2 [9] with the model trained on the Perseus Universal Dependencies Latin Treebank[6] [1], as it is available from the tool's web interface[7]. Baseline results are given in Tables 6 (for poetry) and 7 (for prose).

Metric	Precision	Recall	F1 Score	AligndAcc
CLAS	48.52	48.50	48.51	48.50
LAS	50.36	50.36	50.36	50.36

Table 6: Baseline results on poetry.

Metric	Precision	Recall	F1 Score	AligndAcc
CLAS	52.17	51.44	51.81	51.44
LAS	56.73	56.73	56.73	56.73

Table 7: Baseline results on prose.

Attachment Score (MLAS), that is CLAS extended with evaluation of POS tags and morphological features; the Bi-Lexical dependency score (BLEX) that combines content-word relations with lemmatization, but not with POS tags and features. These 3 metrics are not taken into account for this shared task.

[5]Precision (P) is the number of the correct values divided by the number of the values produced by the system. Recall (R) is the number of correct values divided by the number of the values of the gold standard. F1 score is the harmonic mean of P and R. Aligned Accuracy is the Accuracy computed on the data aligned with the values of the gold standard; for greater details on this, see `http://universaldependencies.org/conll17/evaluation.html`

[6]`https://github.com/UniversalDependencies/UD_Latin-Perseus/`

[7]`http://lindat.mff.cuni.cz/services/udpipe/`

4.0.3 Emotion Polarity Detection

The scorer for the Emotion Polarity Detection task is a Python script that calculates precision, recall and F1 measure for each class assigned at sentence level but also accuracy, macro-average and weighted average. The scorer is available on the EvaLatin web page: `https://github.com/CIRCSE/LT4HALA/blob/master/2024/scorer-emotion.py`. It requires Python 3, pandas and scikit-learn and it takes as input a tab-separated file with two columns: one column contains the predicted labels, the other the gold labels. Columns must have a header: P for predictions and GS for gold labels. Figure 6 shows an example of input file.

```
P           GS
positive    positive
positive    positive
positive    positive
positive    positive
positive    positive
mixed       positive
positive    neutral
negative    negative
```

Figure 6: Example of the input file for the scorer of the Emotion Polarity Detection task.

The command to run the script is the following:

- `python scorer-emotion.py -i input_file.tsv`

An example of the scorer's output is given in Figure 7.

	precision	recall	f1-score	support
mixed	0.22	1.00	0.36	2
negative	0.75	0.56	0.64	16
neutral	1.00	0.57	0.73	7
positive	0.63	0.63	0.63	19
accuracy			0.61	44
macro avg	0.65	0.69	0.59	44
weighted avg	0.71	0.61	0.64	44

Figure 7: Example of the scorer's output for the Emotion Polarity Detection task.

As a baseline, we provide the macro-average F1 scores obtained on the test data with a lexicon-based approach using LatinAffectus v.4, a prior polarity sentiment lexicon for Latin[8] (for details about the lexicon-based approach, see Section 4.1 of [6]).

[8] `https://github.com/CIRCSE/Latin_Sentiment_Lexicons/blob/master/LatinAffectusv4.tsv`

AUTHOR	TEXT	MACRO-AVERAGE F1
Seneca	Hercules Furens (lines 1,175-1,344)	0.25

Table 8: Baseline results on the text by Seneca.

AUTHOR	ODE (BOOK_POEM)	MACRO-AVERAGE F1
Horace	I_2	0.66
Horace	I_14	0.64
Horace	I_28	0.35
Horace	I_38	0.00
Horace	II_3	0.25
Horace	II_11	0.26
Horace	II_14	1.00
Horace	II_16	0.12
Horace	III_2	0.13
Horace	III_10	0.33
Horace	III_18	1.00
Horace	III_24	0.79
Horace	IV_1	0.33
Horace	IV_10	0.00
Horace	IV_12	0.33
Horace	IV_13	0.22
MACRO-AVERAGE		**0.40**

Table 9: Baseline results on the poems by Horace.

AUTHOR	NENIAE	MACRO-AVERAGE F1
Pontano	I	0.22
Pontano	II	0.19
Pontano	III	0.06
Pontano	IV	0.20
Pontano	V	0.14
Pontano	VI	0.17
Pontano	VII	0.22
Pontano	VIII	0.44
Pontano	IX	0.67
Pontano	X	0.24
Pontano	XI	0.72
Pontano	XII	0.24
MACRO-AVERAGE		**0.29**

Table 10: Baseline results on the poems by Pontano.

Chapter 5

How to Participate

Participants are required to submit their runs and to provide a technical report describing their systems.

5.1 Submitting Runs

Each participant can submit a maximum of 2 runs for each task. Participants are allowed to use any approach (e.g. from traditional machine learning algorithms to Large Language Models) and any resource (annotated and non-annotated data, embeddings): all approaches and resources are expected to be described in the system's report.

Once the system has produced the results for the task over the test set, participants have to follow these instructions to complete their submissions:

- name the runs with the following filename format: `task_docName_teamName_runID.conllu`.
  For example: `parsing_XXX_unicatt_1.conllu` would be the first run of a team called *unicatt* for the dependency parsing task on the `XXX.conllu` document.
  `emotion_XXX_unicatt_2.tsv` would be the second run of a team called *unicatt* for the emotion polarity detection task for the `XXX.tsv` document.

- send the file to the following email address: `rachele.sprugnoli[AT]unipr.it`, using the subject "EvaLatin Submission: task - teamName", where the "task" is either *Parsing* or *Emotion*.

5.2 Writing the Technical Report

Technical reports will be included in the proceedings of the Third Workshop on Language Technologies for Historical and Ancient Languages (LT4HALA 2024) as short papers and will be published along the LREC-COLING 2024 proceedings. Reports must be submitted through the START platform (`https://softconf.com/lrec-coling2024/lt4hala2024/`). All the reports must meet the following requirements:

- they must be written in English;
- they must be formatted according to the LREC-COLING 2024 conference style[1];
- the maximum length is 4 pages (excluding references);
- they should contain (at least) the following sections: description of the system, results, discussion, references.

Reports will receive a light review: we will check for the correctness of the format, the exactness of results and ranking, and overall exposition. If needed, we will contact the authors asking for corrections.

[1] https://lrec-coling-2024.org/authors-kit/

Appendix A

Selection of Resources for Latin

- Lemma embeddings: https://embeddings.lila-erc.eu/
- Latin texts and embeddings: http://www.cs.cmu.edu/~dbamman/latin.html
- Latin BERT: https://github.com/dbamman/latin-bert
- Word embeddings: https://lindat.mff.cuni.cz/repository/xmlui/handle/11234/1-1989
- Other embeddings: http://embeddings.texttechnologylab.org/
- CLTK: http://cltk.org/
- UD Latin PROIEL: https://github.com/UniversalDependencies/UD_Latin-PROIEL
- UDLatin ITTB: https://github.com/UniversalDependencies/UD_Latin-ITTB
- UD Latin Perseus: https://github.com/UniversalDependencies/UD_Latin-Perseus
- UD Latin LLCT: https://github.com/UniversalDependencies/UD_Latin-LLCT
- UD UDante: https://github.com/UniversalDependencies/UD_Latin-UDante
- Latin texts: https://github.com/PerseusDL
- Collatinus: https://outils.biblissima.fr/en/collatinus/index.php
- LEMLAT v.3: https://github.com/CIRCSE/LEMLAT3
- Treetagger: https://www.cis.uni-muenchen.de/~schmid/tools/TreeTagger/
- Glossaria: https://glossaria.eu/outils/lemmatisation/#page-content
- Word Formation Latin (WFL) lexicon: http://wfl.marginalia.it/
- Lemmatized corpus with morphological analysis "Corpus Latin antiquité et antiquité tardive lemmatisé": https://doi.org/10.5281/zenodo.4337145

- Latin Lasla Model (Thibault Clérice) for lemmatization and morphological analysis: `https://doi.org/10.5281/zenodo.3773327`
- LatinCy: `https://spacy.io/universe/project/latincy`
- LatinAffectus v4: `https://github.com/CIRCSE/LT4HALA/blob/master/2024/LatinAffectusv4.tsv`
- Deucalion: `https://dh.chartes.psl.eu/deucalion/latin`
- UDPipe Models: `https://ufal.mff.cuni.cz/udpipe/1/models`

Bibliography

[1] David Bamman and Gregory Crane. The ancient Greek and Latin dependency treebanks. In Caroline Sporleder, Antal van den Bosch, and Kalliopi Zervanou, editors, *Language Technology for Cultural Heritage*, Theory and Applications of Natural Language Processing, pages 79–98, Berlin/Heidelberg, Germany, 2011. Springer. Preprint retrievable at `http://www.cs.cmu.edu/~dbamman/pubs/pdf/latech2011.pdf`.

[2] Sabine Buchholz and Erwin Marsi. CoNLL-X shared task on multilingual dependency parsing. In Lluís Màrquez and Dan Klein, editors, *Proceedings of the Tenth Conference on Computational Natural Language Learning (CoNLL-X)*, pages 149–164, New York City, June 2006. Association for Computational Linguistics.

[3] Joseph Denooz. Opera Latina : une base de données sur internet. *Euphrosyne*, 32:79–88, 2004. Retrievable at `https://www.brepolsonline.net/doi/10.1484/J.EUPHR.5.125535`.

[4] Federica Gamba and Daniel Zeman. Universalising Latin Universal Dependencies: a harmonisation of Latin treebanks in UD. In Loïc Grobol and Francis Tyers, editors, *Proceedings of the Sixth Workshop on Universal Dependencies (UDW, GURT/SyntaxFest 2023)*, pages 7–16, Washington, D.C., March 2023. Association for Computational Linguistics.

[5] Joakim Nivre and Chiao-Ting Fang. Universal Dependency evaluation. In Marie-Catherine de Marneffe, Joakim Nivre, and Sebastian Schuster, editors, *Proceedings of the NoDaLiDa 2017 Workshop on Universal Dependencies (UDW 2017)*, pages 86–95, Gothenburg, Sweden, May 2017. Association for Computational Linguistics.

[6] Rachele Sprugnoli, Francesco Mambrini, Marco Passarotti, and Giovanni Moretti. The sentiment of latin poetry. annotation and automatic analysis of the odes of horace. *IJCoL. Italian Journal of Computational Linguistics*, 9(9-1), 2023.

[7] Rachele Sprugnoli, Marco Passarotti, Flavio Massimiliano Cecchini, and Matteo Pellegrini. Overview of the EvaLatin 2020 evaluation campaign. In *Proceedings of LT4HALA 2020 - 1st Workshop on Language Technologies for Historical and Ancient Languages*, pages 105–110, Marseille, France, May 2020. European Language Resources Association (ELRA). Retrievable at `https://www.aclweb.org/anthology/2020.lt4hala-1.16`.

[8] Rachele Sprugnoli, Marco Passarotti, Cecchini Flavio Massimiliano, Margherita Fantoli, and Giovanni Moretti. Overview of the evalatin 2022 evaluation campaign. In *Proceedings of the Second Workshop on Language Technologies for Historical and Ancient Languages (LT4HALA 2022), Language Resources and Evaluation Conference (LREC 2022)*, pages 183–188, 2022.

[9] Milan Straka, Jan Hajič, and Jana Straková. UDPipe: Trainable pipeline for processing CoNLL-U files performing tokenization, morphological analysis, POS tagging and parsing. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 4290–4297, Portorož, Slovenia, May 2016. European Language Resources Association (ELRA). Retrievable at `https://aclanthology.org/L16-1680`.