SSL

Ensemble lemmatization with the Classical Language Toolkit

Patrick J. Burns

Abstract

Because of the less-resourced nature of historical languages, non-standard solutions are often required for natural language processing tasks. This article introduces one such solution for historical-language lemmatization, that is the Ensemble lemmatizer for the Classical Language Toolkit, an open-source Python package that supports NLP research for historical languages. Ensemble lemmatization is the most recent development at *CLTK* in the repurposing and refactoring of an existing method designed for one task, specifically the backoff method as used for part-of-speech tagging, for use in a different task, namely lemmatization. This article argues for the benefits of ensemble lemmatization, specifically, flexible tool construction and the use of all available information to reach tagging decisions, and presents two use cases.

Keywords: lemmatization, natural language processing, Latin, Classical Language Toolkit.

1. *Introduction*

Because of the 'less-resourced' nature of historical languages, specifically due to what is often a paucity of extant text, limited availability of corpora and annotated data, as well as the incompatibility of tools that are available, non-standard solutions are often required for core natural language processing (NLP) tasks[1]. This article offers one such approach to the task of lemmatization, or «the process of transforming any word form into a corresponding, conventionally defined 'base' form» (Sprugnoli *et al.*, 2020: 105) developed for the Classical Language Toolkit (*CLTK*), an open-source Python package that supports NLP research for historical languages with text-analysis pipeline components, including lemmatizers

[1] For a definition of 'less-resourced' with reference to historical languages, see Piotrowski (2012: 85). In keeping with the theme of this special issue, this article will focus on Latin lemmatization, though the tools described here are under development, or can be adapted for use, for other languages.

Received: January 2020
Accepted: April 2020

(Johnson, 2020)[2]. I discuss here an 'ensemble' lemmatization method for Latin developed for *CLTK*, arguing for the benefits of this approach. By ensemble, I mean that the lemmatizer described is in fact a series of sub-lemmatizers that are deployed in unison with a selection mechanism included to limit the output to a single probable lemma or single group of probable lemmas. Following the example of combining results from more than one classifier in machine-learning setups, this version is called the Ensemble lemmatizer[3].

2. *Background*

Because of the lexicographical tradition for many historical languages, including Latin, lemmatization is of primary importance for NLP work on these languages; it is the 'fundamental annotation step' that allows related word forms, often forms with extensive morphological variation, to be grouped under a single identifier[4]. With respect to historical languages, Latin is well-served by off-the-shelf lemmatization tools, interfaces, web services, and desktop applications, including Collatinus, *LatMor*, *Lemlat*, Morpheus, and Whitaker's Words, among others; tools such as Stanza and TreeTagger can be also be included as language-independent tools that support Latin[5].

[2] For a description of text-analysis pipelines and components for historical languages, see Burns (2019). For related material on basic language resource kits, including material pertaining specifically to Latin, see Krauwer (2003); Passarotti (2010: 29); McGillivray (2014: 19-30). *CLTK* currently supports lemmatization for Ancient Greek, Latin, Old English and Old French; tool coverage for different *CLTK* languages can be found in the project's documentation: *http://docs.cltk.org/en/latest/*.

[3] See, for example, Dietterich (2000: 13): «Ensembles are well-established as a method for obtaining highly accurate classifiers by combining less accurate ones». For other examples of an ensemble approach used for Latin lemmatization and part-of-speech tagging, see Stoeckel *et al.* (2020) and Wu and Nicolai (2020).

[4] Mambrini and Passarotti (2019: 73); this article offers an excellent discussion of the lemma as an organizing principle for language tasks and the challenges therein. See Eger *et al.* (2015; 2016) and Gleim *et al.* (2019) for recent surveys of approaches to Latin lemmatization. Heslin (2019) contains a discussion of the challenges of automated Latin lemmatization in a literary critical context. Lemmatization, including specifically the disambiguation of homonymous word forms, has a significantly longer pre-computational tradition dating back to antiquity; see, for example, Dickey (2010: 193-201).

[5] Collatinus: Ouvrard (2010); *LatMor*: Springmann *et al.* (2016); *Lemlat*: Passarotti *et al.* (2017); Morpheus: Crane (1991), originally developed for Greek and later adapted for Latin; Words: Whitaker (1993); Stanza: Qi *et al.* (2020); TreeTagger: Schmid (1994). These lemmatizers are described in more detail in Burns (2019: 166-167).

Yet for the most part these lemmatizers are contextless taggers. That is, they provide lemma information based solely on the value of an isolated token, making no attempt to disambiguate returned tags using information such as the preceding or following words[6]. Accordingly, these tools can perform poorly on lemmatization tasks that would pose little challenge to a competent reader of Latin, as for example with the disambiguation of *ius* ("law") and *ius* ("broth, soup")[7].

Methods used in recent research on historical-language lemmatization include lexicon-assisted tagging and transformation rule induction, joint lemmatization and part-of-speech (PoS) tagging, as well as lemmatization as a neural-network-assisted string-transduction task[8]. With respect to the latter, research in historical-language lemmatization, following larger trends in NLP research generally, has taken a turn toward neural networks and deep learning approaches. These approaches, using either word- or character-level embeddings, often in conjunction with PoS tagging and dependency parsing, represent or near state-of-the-art performance for many languages[9]. Furthermore, neural-network approaches that take advantage of sentence-level context are proving to be especially effective, especially with respect to disambiguation (Bergmanis and Goldwater, 2018; Kestemont *et al.*, 2017; Manjavacas *et al.*, 2019)[10]. Another direction that has emerged in lemmatization for historical languages is their inclusion in recent large multilingual lemmatization studies due to their presence in the Universal Dependency treebanks (Nivre *et al.*, 2018)[11].

[6] See, for example, the notice in Passarotti *et al.* (2017: 25) on word form analysis using the *Lemlat* lemmatization tool: «Given an input word form that is recognised by *Lemlat*, the tool produces in output the corresponding lemma(s) [...] No contextual disambiguation is performed».

[7] It should be noted that intentional ambiguity is a nuance that lies outside the scope of computer-assisted approaches to lemmatization, at least as it is conceived of as an NLP task. For an overview of intentional ambiguity in Latin literature, see Fontaine *et al.* (2018), and pages xi-xii in particular on wordplay involving the ambiguity of *ius*.

[8] See, for example, Eger *et al.* (2015) and related work in Juršič *et al.* (2010), Bary *et al.* (2017), and Manjavacas *et al.* (2019), respectively.

[9] See, for example, Kondratyuk *et al.* (2018), Malaviya *et al.* (2019), Straka *et al.* (2019a), Straka and Straková (2020), and Celano (2020).

[10] See also, Chrupała (2006) on the usefulness of continuous text for the lemmatization of out-of-vocabulary words.

[11] Historical languages other than Latin, such as Ancient Greek, Coptic, Old French, and Old Church Slavonic, are also represented in version 2.3 of Universal Dependencies. For examples of recent multilingual shared task studies including Latin results, see Zeman *et al.* (2018) and Straka *et al.* (2019b).

In summary, despite advances, significant challenges still remain in historical-language lemmatization, in particular concerning the disambiguation of homonyms and the handling of unseen vocabulary, that is words that appear neither in a lexicon or in the training data used by the lemmatizer[12]. Moreover, there remains the question of whether 'lemma' is a stable enough category to be treated in a truly language-independent way and, for that reason, whether a lemmatizer should be designed to allow for a more flexible definition of the term[13]. Ensemble lemmatization works to address these challenges through flexibility of construction and the ability to combine results derived from a wide range of data sources, including lexicons, sentence-level training data, lists of regular expression patterns, and the output of other lemmatizers, among other sources[14].

3. *Lemmatizer construction with the Classical Language Toolkit*

Most approaches to historical-language lemmatization involve (i) taking an input, either a single token out of context or a token with its adjacent characters or words, (ii) performing a lookup of this token in a lexicon or otherwise analyzing this token, and (iii) returning a lemma or list of potential lemmas. Such approaches to lemmatization tend to share a certain fixity in design; that is, they tend to rely on a specific lexical data source or apply a specific set of rule-based transformations, and so on. Accordingly, the inter-

[12] Rosa and Žabokrtský (2019), for example, report 'deteriorations' on error reduction in unsupervised lemmatization for Latin. All the same, it is worth acknowledging how much progress has been made in this area since Ireland (1976: 46): «The present author knows of no system that as yet offers complete automatic lemmatization». If anything, this present author knows of several systems offering 'complete' automatic lemmatization; the focus of the current work is instead boosting accuracy, improving disambiguation, addressing a wider range of language domains, and handling the longest of long-tail vocabulary.

[13] See Knowles and Don (2004) on the difficulty of generalizing the idea of lemmatization across different languages, in particular English, Latin, Arabic, and Malay.

[14] The combination of multiple lemmatization strategies has something in common with the 'hybrid approach' described in Boudchiche and Mazroui (2019) which uses a two-pass lemmatization strategy: the first pass lemmatizes words out-of-context, a second pass uses a statistical method to disambiguate lemmas in context. Sychev and Penskoy (2019) describe a process for algorithmically «selecting different lemmatizers for different words» in English. For an early example of a staged approach to computer-assisted lemmatization, see Krause and Willée (1981). See also Romero (2019) for an example of 'modular design' in the construction of lemmatizers for Spanish and other languages.

nals of the lemmatization process are not exposed to the user[15]. The *CLTK*, on the other hand, offers options for lemmatization that specifically expose the lemmatizer construction process to the user, allowing for all intents and purposes an unlimited number of lexicons, rule definitions, or other tagging strategies to be combined and coordinated to reach a decision about the optimal choice of lemma for a given token.

3.1. *Backoff lemmatization*

Flexible lemmatizer construction was first introduced to the *CLTK* with the Backoff lemmatizer[16]. The main innovation of the Backoff lemmatizer was the repurposing of an existing method designed for one NLP task, specifically the backoff method as used for PoS tagging, for use in a different task, namely lemmatization[17]. In its original definition in the Natural Language Toolkit, sequential backoff tagging allows users to construct a PoS tagger from a set of sub-taggers (Bird *et al.*, 2015)[18]. A base tagger, called SequentialBackoffTagger, defines the backoff logic as follows: the first sub-tagger in the sequence attempts to tag a given token and, if it is unable to do so, the next sub-tagger in the sequence (that is, the 'backoff' tagger) is tried and so on, until either a token is successfully tagged or the sequence ends. Various sub-taggers make use of different tagging strategies, including the use of frequency data from annotated sentences, custom lexicons, or lists of regular expressions patterns, among other resources, to assign tags. The effectiveness of Sequential Backoff Tagger resides not in any specific sub-tagger but in their combined deployment, since subsequent taggers compensate for the gaps in coverage of previous ones.

[15] It is true that most of the available tools offer some degree of customization with respect to the lemmatization process, even if they lack the flexibility of construction and choice of parameters offered by the *CLTK* lemmatizers. For example, *Lemlat* and Collatinus have parameters available for choosing the lexical basis for analyzing tokens; see *https://github.com/CIRCSE/LEMLAT3/wiki/2.-Use* and *https://outils.biblissima.fr/en/collatinus-web/* respectively.

[16] The Backoff lemmatizer for Latin was developed as part of a 2016 Google Summer of Code project; see the project description here: *https://summerofcode.withgoogle.com/archive/2016/projects/6499722319626240*. The source code can be found in the Lemmatize module at *https://github.com/cltk/cltk/tree/master/cltk/lemmatize*.

[17] The basic design of the Backoff lemmatizer is given in Burns (2016) with additional description in the section 'Lemmatization as reading' in Burns *et al.* (2019). The discussion here of the Backoff lemmatizer is meant to provide context for understanding the motivation for the development of the Ensemble lemmatizer.

[18] The source code for SequentialBackoffTagger and its subclasses can be found at *https://www.nltk.org/_modules/nltk/tag/sequential.html*; see also Perkins (2014: 92-93).

The repurposing of SequentialBackoffTagger for lemmatization makes sense because at its heart lemmatization is a tagging task (Gesmundo and Samardžić, 2012). That said, as opposed to the well-bounded task of PoS tagging, lemmatization is an infinite tagging task. There are 17 tags in the Universal PoS tagset and 36 in the Penn Treebank PoS tagset[19]. Even with largely fixed-corpus languages such as Ancient Greek and Latin, there are a nearly infinite number of word forms that could be mapped to a lemma, something made clear, for example, by the hundreds of 'new' words published in the supplements to Liddell and Scott's *Greek-English Lexicon*[20]. Accordingly, from a tagging perspective, the performance of a lexicon-based approach can only be improved by expanding lexicon coverage, and even at that, the direction and degree of this expansion would be difficult, if not impossible, to predict. So, for example, the Latin coinage *telecommunicationis* ("of telecommunication") as found in the Latin Wikipedia article about the telephone will not be tagged by any off-the-shelf Latin lemmatizer[21]. Still, this word would likely be lemmatized correctly if a regular-expression-based lemmatizer is included in the backoff chain, since its genitive singular word ending (*-ationis*) can be mapped predictably to the nominative singular form that is traditionally used for reporting Latin noun lemmas[22]. It is this combination of data-driven and rules-based strategies that makes backoff tagging an effective approach to lemmatization.

That said, backoff tagging has a major disadvantage. SequentialBackoffTagger takes a binary approach to tagging; that is, at any given point in the backoff chain, a tagger either assigns a tag or it does not. If a tag is assigned, the sequence is terminated and the tagger moves onto the next token. Foreshortening the backoff chain in this way improves processing speed, but at the cost of loss of information from the unused taggers. Moreover, the arrangement of sub-lemmatizers in the backoff chain can have a hard to predict effect on the results.

[19] For the Universal PoS tagset, see *https://universaldependencies.org/u/pos/*; for the Penn tagset, see Santorini (1995).

[20] See Glare and Thompson (1996), itself a revision of an earlier version from 1968. Eger *et al.* (2016: 1507 n. 2) also notes that the lexicons «cannot store an infinite number of words».

[21] See *https://la.wikipedia.org/wiki/Telephonum*: *Telephonum* [...] *est instrumentum telecommunicationis quo homines per longa spatia inter se loqui possunt* "The telephone is an instrument of telecommunication with which people are able to speak to each other over long distances".

[22] See Diederich (1939: 21-30) for a statistical evaluation of the use of Latin word endings to determine lemmas.

3.2. *Ensemble lemmatization*

In order to avoid the loss of potentially useful information from sub-lemmatizers further down the backoff chain, the Backoff lemmatizer has been refactored so as not to terminate upon the first successful tagging. The resulting tool is the Ensemble lemmatizer[23]. With this setup, all tokens are tagged by all sub-lemmatizers. No tagging information is lost. At the completion of the tagging operation, a list of potential lemmas is returned, and, if requested, a selection mechanism can be used to limit this output to a single probable lemma.

The advantage of complete multiple-pass tagging is that all available information provided by sub-lemmatizers in the sequence is retained and, as such, can be used to make a final determination. Here is a simple example based on Cicero's *De domo suo* 39: *Infirmas igitur tu acta C. Caesaris?*, "Are you therefore weakening Gaius Caesar's decrees?".

We can construct an Ensemble lemmatizer using two sub-lemmatizers, namely a lexicon-based lemmatizer (EnsembleDictLemmatizer) with a lexicon mapping the token *infirmas* to the lemma *infirmus* and regular-expression-based lemmatizer (EnsembleRegexpLemmatizer) with a pattern that replaces tokens ending in *-as* (and other present active endings for first conjugation Latin verbs), in that order (reading from the bottom up):

(1) regexp_ensemble_lemmatizer = EnsembleRegexpLemmatizer(patterns= [('(.)a(s|t|mus|tis|nt)$', '\1o')], backoff = None)

dict_ensemble_lemmatizer = EnsembleDictLemmatizer(dictionary = {'infirmas': 'infirmus'}, backoff = regexp_ensemble_lemmatizer)

As opposed to the backoff setup, the fact that the lexicon-based lemmatizer tags *infirmas* (incorrectly) as a form of the adjective *infirmus* ("weak") on the first pass does not prevent it from also tagging the token (correctly) as the verb *infirmo* ("to weaken") on the second pass. Some selection mechanism needs to be used to perform the disambiguation, whether frequency distributions from training data, probabilities assigned to word-ending patterns, contextual semantics, confidence scores based on dependency parsing[24], and so on. Again, this is a trivial example designed to explain how the

[23] The source code can be found in the Lemmatize module at *https://github.com/cltk/cltk/tree/master/cltk/lemmatize*.

[24] This sentence provides an excellent example of how dependency tree information could be combined with traditional approaches to reading Latin to assist with lemma disambiguation as

Ensemble lemmatizer works and in particular how it works differently than the Backoff lemmatizer. An example showing the clear advantage of the ensemble setup is offered in Section 4.

A final point on the Ensemble lemmatizer. While the example above shows only two main types of sub-lemmatizers, that is lexicon-based and regular-expression-based lemmatizers, the 'building block'-style design of this lemmatizer allows for the development of any number of sub-lemmatizers. By subclassing SequentialBackoffTagger and overriding the 'tag' method with a different method of determining a lemma from a token, any lemmatization algorithm can be incorporated into the Ensemble lemmatizer. As long as a subclass of one of the lemmatizers (i) accepts a list of tokens as its input and (ii) provides a list of lemmas as its output, it can be added to the lemmatization chains.

A specific kind of sub-lemmatizer is ideal for development under this 'building block' logic, namely wrappers, that is classes or functions that allow external code to be used locally, written for existing lemmatization tools[25]. As noted above, there are several off-the-shelf options available for lemmatizing Latin texts, but at present their results cannot be effectively collated and evaluated without some sort of ad hoc post-processing. Moreover, these tools can be incompatible with each other or otherwise not customizable or extensible[26]. This is because each tool is envisioned as a self-sufficient solution for the task. Ensemble lemmatization reconceives them as part of a coordinated lemmatization solution, the combined results of which can be easily and directly incorporated into a tagging workflow. So, rather than having to choose TreeTagger or *Lemlat*, wrapper-based sub-lemmatizers can be chained together so that both are used, leveraging the strengths of each[27].

infirmas ("you weaken") is the only eligible verb in this sentence, not to mention that the explicit (and unambiguous) subject *tu* ("you") confirms the requirement of a second-person singular verb in the sentence. Ensemble lemmatizer development following these kinds of traditional reading approaches is ongoing; see McCaffrey (2006), for example, on disambiguation in reading Latin as well as the discussion of 'philological method' in Section 5 below.

[25] For a general discussion of wrappers in the *CLTK* pipeline, see Burns (2019: 171-172). On wrappers as a best practice when working with third-party software, see Martin (2009: 109).

[26] Addressing interoperability is a primary objective of the Linking Latin (*LiLa*) project; see the *LiLa* objectives here *https://lila-erc.eu/about* as well as in Mambrini and Passarotti (2019).

[27] An example of a chained-together wrappers is given below in Section 4.

4. *Use cases*

While high accuracy is obviously a goal of any NLP tool, the more important contribution of ensemble lemmatization comes with the coordination of results made possible by its modular, flexible construction which allows for a greater degree of customization depending on the language being processed (and the availability of supporting resources for this language) as well as the domain being studied, the research question under consideration, and so on[28]. To illustrate the benefits of this coordination, modularity, and flexibility, I offer two uses cases: (i) the lemmatization of a text likely to pose a significant challenge to existing tools, namely a Latin translation of Lewis Carroll's 'Jabberwocky' and (ii) the use of the Ensemble lemmatizer to combine effectively existing tools.

4.1. *Lemmatizing 'Jabberwocky' with the Ensemble lemmatizer*

The handling of unseen vocabulary is a challenge for lemmatizers. For historical languages, this challenge is particularly acute because, not only are they often less-resourced in general, but their resources can be especially limited for variations of dialect, period, and so on[29]. The example here illustrates this with an extreme case, namely lemmatizing a Latin translation of Lewis Carroll's nonsense poem, 'Jabberwocky', by C. H. Carruthers[30]. Here are the opening lines: *Est brilgum: tovi slimici / in vabo tererotitant* "'Twas brillig, and the slithy toves / did gyre and gimble in the wabe". Some words here would present no difficulty to any Latin lemmatizer: *est* and *in*. The remaining words however will understandably not appear in any Latin lexicon and for this reason off-the-shelf solutions will be unlikely to yield results. At the same time, a competent reader of Latin can lemmatize this text with minimal difficulty through additional interpretative strategies. *Tererotitant* can only be lemmatized as *tererotito*; the Latin reader knows this because the -(*t*)*ant* ending, a marker of the third-person active plural, can be meaningfully transformed to

[28] For a look into the current state of evaluation for Latin lemmatization methods, see Sprugnoli *et al.* (2020) and the participating papers in the *EvaLatin* 2020 campaign.

[29] See Kestemont and De Gussem (2017) for using a neural-network approach to handle historical-language variation, and in particular, Medieval Latin orthography.

[30] This translation appears as *Jabberwocky: An Alternative Version* in Carroll (1966: 132-133). For background on these translations and others, see Imholtz (1987); Van Dam (1982). For an example of NLP methods used on this poem, see Feldman (1999).

the first-person present indicative active forms traditionally used for verb lemmas[31]. Accordingly, if we set up a backoff sequence that reflects the processes of a competent reader, we can make meaningful inroads in lemmatizing this text:

(2) regexp_lemmatizer = EnsembleRegexpLemmatizer(patterns = [('(.)a(s|t|mus|tis|nt)$', '\1o')], backoff = None)
dict_lemmatizer = EnsembleDictLemmatizer(dictionary = {'est': 'sum', 'in': 'in'}, backoff = regexp_lemmatizer)

Additional patterns could be written for other nonsense words in the poem: *vorpalem* to *vorpalis*, *Unguimanu* to *Unguimanus*, *gaudiferum* to *gaudifer*, *praehilare* to *praehilaris*, and so on[32]. Admittedly, the lemmatization of Latin nonsense poetry is a low-priority problem. Nevertheless, the issues raised by this problem, most especially dealing with unknown word forms and transforming them in a consistent, philologically sound manner, will surface whenever NLP tools are used on 'underserved domains' and an ensemble approach is well-equipped to handle this situation[33].

4.2. *Combining lexicons with the Ensemble lemmatizer*

As noted above, off-the-shelf Latin lemmatizers are generally envisioned as self-sufficient solutions for the task and as a result there is often no direct way to combine efficiently and aggregate the results of multiple tools. The Ensemble lemmatizer using wrappers written for existing tools can solve this problem. Here is an example based on the beginning of Book 12 of Ovid's *Metamorphoses*: *Nescius adsumptis Priamus pater Aesacon alis / vivere lugebat* "Father Priam was mourning for Aesacus, not realizing that he had assumed wings and was alive". If we set up a backoff chain with wrappers for Latin lemmatizers mentioned in Section 2 as follows:

[31] Sequence-modeling approaches could also be used to address this, though there would perhaps be a concern of adding unbounded noise to the textual noise inherent in nonsense poetry. See Kestemont and De Gussem (2017) for a discussion of 'computational hypercorrection' and the generation of 'unrecognisable form[s]'. Using a list of regular-expression-based replacement patterns that reflect traditional expectations about the morphological information found in word endings goes some way in mitigating this concern; see below on 'philological method' in Section 5. Moreover, a sequence-modeling-based wrapper could always be written for the Ensemble lemmatizer and could substitute for (or complement) the regular-expression-based lemmatizer in this sequence.

[32] A starter set of regular-expression-based replacement patterns for Latin can be found at *https://github.com/cltk/cltk/blob/master/cltk/lemmatize/latin/latin.py*.

[33] See Bamman (2017) on NLP for 'underserved domains'.

(3) lemlat = LemlatLemmatizer(backoff = None)
collatinus = CollatinusLemmatizer(backoff = lemlat)
words = WordsLemmatizer(backoff = collatinus)
morpheus = MorpheusLemmatizer(backoff = words)
latmor = LatmorLemmatizer(backoff = morpheus)
treetagger = TreeTaggerLemmatizer(backoff = latmor)

we get a better sense of how the coverage of each tool complements the others. Several words in this example pose no problem for any of the lemmatizers: *nescius*, *pater*, *vivere*, and *lugebat* are all tagged correctly and unambiguously as *nescius*, *pater*, *vivo*, and *lugeo*, respectively[34]. In other cases, an individual tagger fails to return a lemma, but this gap is covered by one of the other taggers: for example, TreeTagger does not return a lemma for *Priamus*, but Collatinus, *LatMor*, *Lemlat*, Morpheus, and Whitaker's Words all return the correct lemma. A token like *ne* (*Met*. 12.590) presents the opposite problem, as the tools return different sets of lemmas: *ne* (TreeTagger); *ne*, *neo* (Collatinus, *Lemlat*, Morpheus, Words); and *ne*, *nere* (*LatMor*). In this case, even a simple count-based vote would return the correct lemma *ne*, present six times across the results of the six lemmatizers[35]. Still, the more important point here is that the Ensemble lemmatizer provides a direct way of combining the output of multiple taggers and maximizing the amount of information available for determining the best choice.

5. *Conclusion*

5.1. *Ensemble lemmatization as a philological method*

As described above, the Ensemble lemmatizer offers technical advantages to the lemmatization of historical-language text. It is worth noting that this approach to lemmatization offers a theoretical advantage as well to the

[34] *LatMor* with its default settings tags *vivere* not as *vivo* but as the present active infinitive *vivere*. In testing this configuration, the *LatMor* wrapper normalized the output of verbs by re-lemmatizing these infinitives with another tool (here, namely, Collatinus). The normalization that can be built into the Ensemble wrappers can be seen as another benefit of the approach.

[35] Other options exist for resolving similar lists of possible lemmas. Gawley (2019), for example, presents a disambiguation method based on corpus frequencies and Heslin (2019) proposes a novel method for disambiguation that compares the lengths of lexicon entries for respective forms.

primary audience for *CLTK*'s tools, namely historical-language researchers, instructors, and students. I have argued before that backoff lemmatization «can be described as following a philological method» because it reflects the decoding strategies of the philologically trained reader of historical texts (Burns, 2018)[36]. That said, ensemble lemmatization demonstrates this even more clearly since it draws on multiple sources of information and makes use of all of them in arriving at a decision. This reflects, for example, the process of the textual critic who, through both a comprehensive accounting of word use in context and the relative frequency of tokens and their endings, is able to make philologically informed decisions about possible readings[37]. This also reflects the process of a Latin translator for whom working through a text with multiple passes can be an effective decipherment strategy, as one Latinist recommends: «Once you know what all the words can mean, re-read the Latin to [...] clarify what the words in the sentence [...] mean» (Hoyos, 2008). Yet another group of Latin teachers emphasize this progressive clarification as a «dynamic process which involves continual reconsideration of previous decisions and expectations», not unlike the process whereby the Ensemble lemmatizer accumulates potential lemmas before arriving at a decision about the most probable lemma or lemmas (Markus and Ross, 2004: 88). The backoff and the ensemble approaches to lemmatization, and the ensemble approach in particular, reflect established disciplinary practices for disambiguating words and acknowledge that this process often requires coordinated methods.

5.2. *Future directions*

The Ensemble lemmatizer discussed here is available at present for Latin, but is included in the 'Multilingual' section of the *CLTK* documentation, since the sub-lemmatizers can be used with any language for which supporting resources such as token-lemma lexicons, annotated

[36] For a discussion of decoding strategies as applicable to the study of Latin, see McCaffrey (2006; 2009); Russell (2018); see also Burns *et al.* (2019) on the relationship between lemmatization, literacy, and 'classical-language reading patterns'. A reviewer astutely points out that similar decoding strategies may be typical of language users generally in negotiating the meaning of words in context; I limit the discussion here to observations that have been made on this point concerning philological activities such as textual criticism and historical-language pedagogy.

[37] See Tarrant (2016: 57): «When choosing between or among equally well-attested variants, the editor may have recourse to a variety of potentially relevant factors». For an example of a systematic study of word endings in the context of textual criticism, see Håkanson (1982).

sentences, or regular expression patterns can be provided. A good next step would be the development of default sequences for the full range of languages covered by *CLTK* for which lemmatization is a core task. Another good step would be the development of more wrappers that can be used with the Ensemble lemmatizer, not only for off-the-shelf tools as discussed above, but also for the state-of-the-art methods discussed in Section 2. In the spirit of the 'all available information' approach of the Ensemble lemmatizer, it is not hard to see the benefit to *CLTK* users in being able to include these methods in backoff sequences and combine them with other methods.

Acknowledgments

Earlier versions of this article were presented at the First *LiLa* Workshop: Linguistic Resources & NLP Tools for Latin at Università Cattolica del Sacro Cuore in the summer of 2019 as well as at DH2018 in Mexico City and the Digital Classicist London Seminar in the summer of 2018. The author thanks the organizers, participants, and attendees of these events for their feedback. The author would also like to acknowledge the Classical Language Toolkit open-source development community, the Institute for the Study of the Ancient World Library, and the Quantitative Criticism Lab for their support. Lastly, the author thanks the anonymous reviewers of this article for their suggestions.

References

Bamman, D. (2017), *Natural Language Processing for the long tail*, in Lewis, R., Raynor, C., Forest, D., Sinatra, M. and Sinclair, S. (2017, eds.), *Digital Humanities 2017, DH 2017, Conference Abstracts, McGill University & Université de Montréal, Montréal, Canada, August 8-11, 2017*, Alliance of Digital Humanities Organizations (ADHO).

Bary, C., Berck, P. and Hendrickx, I. (2017), *A memory-based lemmatizer for Ancient Greek*, in *Proceedings of the 2nd International Conference on Digital Access to Textual Cultural Heritage - DATeCH2017*, Association for Computing Machinery, New York, pp. 91-95.

Bergmanis, T. and Goldwater, S. (2018), *Context sensitive neural lemmatization with Lematus*, in Walker, M., Ji, H. and Stent, A. (2018, eds.), *Proceedings of the 2018 Conference of the North American Chapter of the Asso-*

ciation for Computational Linguistics: Human Language Technologies. Vol. 1: *Long Papers*, Association for Computational Linguistics, Stroudsburg, PA, pp. 1391-1400.

Bird, S., Klein, E. and Loper, E. (2015), *Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit* [available online at *https://www.nltk.org/book*, accessed on 21.05.2020].

Boudchiche, M. and Mazroui, A. (2019), *A hybrid approach for Arabic lemmatization*, in «International Journal of Speech Technology», 22, pp. 563-573.

Burns, P.J. (2016), *Wrapping up Google Summer of Code* [available online at *https://disiectamembra.wordpress.com/2016/08/23/wrapping-up-google-summer-of-code/*, accessed on 21.05.2020].

Burns, P.J. (2018), *Backoff lemmatization as a philological method*, in Girón Palau, J. and Russell, I.G. (2018, eds.), *Digital Humanities 2018, DH 2018, Book of Abstracts, El Colegio de México, UNAM, and RedHD, Mexico City, Mexico, June 26-29, 2018*, Red de Humanidades Digitales.

Burns, P.J. (2019), *Building a text analysis pipeline for Classical languages*, in Berti, M. (2019, ed.), *Digital Classical Philology: Ancient Greek and Latin in the Digital Revolution*, De Gruyter, Berlin, pp. 159-176.

Burns, P.J., Hollis, L. and Johnson, K.P. (2019), *The future of ancient literacy: Classical Language Toolkit and Google Summer of Code*, in «Classics@», 17.

Carroll, L. (1966), *Ludovici Carroll fabella lepida in qua aliud Aliciae sominium narravit: Aliciae per speculum transitus (quaeque ibi invenit)*, Macmillan, London.

Celano, G.G.A. (2020), *A gradient boosting-Seq2Seq system for Latin PoS tagging and lemmatization*, in Sprugnoli, R. and Passarotti, M. (2020, eds.), *Proceedings of 1st Workshop on Language Technologies for Historical and Ancient Languages*, European Language Resources Association (ELRA), Paris, pp. 119-123.

Celano, G.G.A. *et al.* (2014), *The Ancient Greek and Latin Dependency Treebank v.2.1* [available online at *https://perseusdl.github.io/treebank_data*, accessed on 21.05.2020].

Chrupała, G. (2006), *Simple data-driven context-sensitive lemmatization*, in «Procesamiento del Lenguaje Natural», 37, pp. 121-127.

Crane, G. (1991), *Generating and parsing Classical Greek*, in «Literary and Linguistic Computing», 6, pp. 243-245.

Dickey, E. (2010), *The creation of Latin teaching materials in antiquity: A re-interpretation of P.sorb. Inv. 2069*, in «Zeitschrift für Papyrologie und Epigraphik», 175, pp. 188-208.

Diederich, P.B. (1939), *The Frequency of Latin Words and Their Endings*, The University of Chicago Press, Chicago.

Dietterich, T.G. (2000), *Ensemble methods in machine learning*, in Kittler, J. and Roli, F. (2000, eds.), *Multiple Classifier Systems. Proceedings of the First International Workshop, MCS 2000 (Cagliari, Italy, June 21-23, 2000)*, Springer, Berlin, pp. 1-15.

Eger, S., Gleim, R. and Mehler, A. (2016), *Lemmatization and morphological tagging in German and Latin: A comparison and a survey of the state-of-the-art*, in Calzolari, N., Choukri, K., Declerck, T., Goggi, S., Grobelnik, M., Maegaard, B., Mariani, J., Mazo, J., Moreno, A., Odijk, J. and Piperidis, S. (2016, eds.), *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, European Language Resources Association (ELRA), Paris, pp. 1507-1513.

Eger, S., vor der Brück, T. and Mehler, A. (2015), *Lexicon-assisted tagging and lemmatization in Latin: A comparison of six taggers and two lemmatization methods*, in Zervanou, K., van Erp, M. and Alex, B. (2015, eds.), *Proceedings of the 9th SIGHUM Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities (LaTeCH)*, Association for Computational Linguistics, Stroudsburg, PA, pp. 105-113.

Feldman, S. (1999), *NLP meets the Jabberwocky: Natural Language Processing in information retrieval*, in «ONLINE», 23.

Fontaine, M., McNamara, C.J. and Short, W.M. (2018), *Introduction*, in Fontaine, M., McNamara, C.J. and Short, W.M. (2018, eds.), Quasi labor intus*: Ambiguity in Latin Literature: Papers in Honor of Reginald Thomas Foster, OCD*, Paideia Institute for Humanistic Study, Middletown, DE, pp. ix-xxxi.

Gawley, J.O. (2019), *An unsupervised lemmatization model for Classical languages* [available online at *https://dev.clariah.nl/files/dh2019/boa/1007.html*, accessed on 21.05.2020].

Gesmundo, A. and Samardžić, T. (2012), *Lemmatisation as a tagging task*, in Li, H., Lin, C-Y., Osborne, M., Geunbae Lee, G. and Park, J.C. (2012, eds.), *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*. Vol. 1: *Short Papers*, Association for Computational Linguistics, Stroudsburg, PA, pp. 368-372.

Glare, P.G.W. and Thompson, A.A. (1996, eds.), *Greek-English Lexicon. Revised Supplement*, Clarendon Press, Oxford.

Gleim, R., Eger, S., Mehler, A., Uslu, T., Hemati, W., Lücking, A., Henlein, A., Kahlsdorf, S. and Hoenen, A. (2019), *A practitioner's view: A survey and comparison of lemmatization and morphological tagging in German and Latin*, in «Journal of Language Modelling», 7, pp. 1-52.

Håkanson, L. (1982), Homoeoteleuton *in Latin dactylic poetry*, in «Harvard Studies in Classical Philology», 86, pp. 87-115.

Heslin, P. (2019), *Lemmatizing Latin and quantifying the Achilleid*, in Coffee, N., Forstall, C., Milić, L.G. and Nelis, D. (2019, eds.), *Intertextuality in Flavian Epic Poetry*, De Gruyter, Berlin, pp. 389-408.

Hoyos, D. (2008), *The ten basic rules for reading Latin* [available online at *http://www.latinteach.com/Site/ARTICLES/Entries/2008/10/15_Dexter_Hoyos_-_The_Ten_Basic_Rules_for_Reading_Latin_files/Reading%20%26Translating%20Rules.pdf*, accessed on 21.05.2020].

Imholtz, A.A. (1987), *Latin and Greek versions of «Jabberwocky». Exercises in laughing and grief*, in «Rocky Mountain Review of Language and Literature», 41, pp. 211-228.

Ireland, S. (1976), *The computer and its role in classical research*, in «Greece & Rome», 23, pp. 40-54.

Johnson, K.P. (2020), *CLTK: The Classical Language Toolkit* [available online at *https://github.com/cltk/cltk*, accessed on 21.05.2020].

Juršič, M., Mozetič, I., Erjavec, T. and Lavrač, N. (2010), *LemmaGen: Multilingual lemmatisation with induced ripple-down rules*, in «Journal of Universal Computer Science», 16, pp. 1190-1214.

Kestemont, M. and De Gussem, J. (2017), *Integrated sequence tagging for medieval Latin using deep representation learning*, in Büchler, M. and Mellerin, L. (2017, eds.), *Special Issue on Computer-Aided Processing of Intertextuality in Ancient Languages*, «Journal of Data Mining & Digital Humanities» (special issue).

Kestemont, M., de Pauw, G., van Nie, R. and Daelemans, W. (2017), *Lemmatization for variation-rich languages using deep learning*, in «Digital Scholarship in the Humanities», 32, pp. 797-815.

Knowles, G. and Don, Z.M. (2004), *The notion of a "lemma": Headwords, roots and lexical sets*, in «International Journal of Corpus Linguistics», 9, pp. 69-81.

KONDRATYUK, D., GAVENČIAK, T., STRAKA, M. and HAJIČ, J. (2018), *LemmaTag: Jointly tagging and lemmatizing for morphologically-rich languages with BRNNs* [preprint available online at *https://arxiv.org/abs/1808.03703*].

KRAUSE, W. and WILLÉE, G. (1981), *Lemmatizing German newspaper texts with the aid of an algorithm*, in «Computers and the Humanities», 15, pp. 101-113.

KRAUWER, S. (2003), *The Basic Language Resource Kit (BLARK) as the first milestone for the Language Resources Roadmap*, in *Proceedings of the International Workshop Speech and Computer, Moscow, Russia*, pp. 8-15.

MALAVIYA, C., WU, S. and COTTERELL, R. (2019), *A simple joint model for improved contextual neural lemmatization* [preprint available online at *https://arxiv.org/abs/1904.02306*].

MAMBRINI, F. and PASSAROTTI, M. (2019), *Harmonizing different lemmatization strategies for building a knowledge base of linguistic resources for Latin*, in FRIEDRICH, A., ZEYREK, D., and HOEK, J. (2019, eds.), *Proceedings of the 13th Linguistic Annotation Workshop*, Association for Computational Linguistics, Stroudsburg, PA, pp. 71-80.

MANJAVACAS, E., KÁDÁR, A. and KESTEMONT, M. (2019), *Improving lemmatization of non-standard languages with joint learning* [preprint available online at *https://arxiv.org/abs/1903.06939*].

MARKUS, D.D. and ROSS, D.P. (2004), *Reading proficiency in Latin through expectations and visualization*, in «Classical World», 98, pp. 79-93.

MARTIN, R.C. (2009), *Clean Code: A Handbook of Agile Software Craftsmanship*, Prentice Hall, Upper Saddle River, NJ.

MCCAFFREY, D. (2006), *Reading Latin efficiently and the need for cognitive strategies*, in GRUBER-MILLER, J. (2006, ed.), *When Dead Tongues Speak: Teaching Beginning Greek and Latin*, Oxford University Press, New York, pp. 113-133.

MCCAFFREY, D. (2009), *When reading Latin, read as the Romans did*, in «Classical Outlook», 86, pp. 62-66.

MCGILLIVRAY, B. (2014), *Methods in Latin Computational Linguistics*, Brill, Leiden.

NIVRE, J., ABRAMS, M. and AGIĆ, Ž. (2018), *Universal Dependencies v.2.3* [available online at *http://hdl.handle.net/11234/1-2895*, accessed on 21.05.2020].

OUVRARD, Y. (2010), *Collatinus, lemmatiseur et analyseur morphologique de la langue latine*, in «ÉLA. Études de linguistique appliquée», 158, pp. 223-230.

Passarotti, M. (2010), *Leaving behind the less-resourced status. The case of Latin through the experience of the Index Thomisticus Treebank*, in Sarasola, K., Tyers, F.M. and Forcada, M.L. (2010, eds.), *Proceedings of the 7th SaLTMiL Workshop on Creation and Use of Basic Lexical Resources for Less-Resourced Languages, LREC 2010*, European Language Resources Association, La Valletta, pp. 27-32.

Passarotti, M., Budassi, M., Litta, E. and Ruffolo, P. (2017), *The* Lemlat *3.0 package for morphological analysis of Latin*, in Bouma, G. and Adesam, Y. (2017, eds.), *Proceedings of the NoDaLiDa 2017 Workshop on Processing Historical Language*, Linköping University Electronic Press, Linköping, pp. 24-31.

Perkins, J. (2014), *Python 3 Text Processing with NLTK 3 Cookbook*, Packt Publishing Ltd., Birmingham, U.K.

Piotrowski, M. (2012), *Natural Language Processing for Historical Texts*, Morgan & Claypool Publishers, San Rafael, CA.

Qi, P., Zhang, Y., Zhang, Y., Bolton, J. and Manning, C.D. (2020), *Stanza: A Python natural language processing toolkit for many human languages* [preprint available online at *https://arxiv.org/abs/2003.07082*].

Romero, G. (2019), *Rethinking rule-based lemmatization* [available online at *https://www.youtube.com/watch?v=88zcQODyuko*, accessed on 21.05.2020].

Rosa, R. and Žabokrtský, Z. (2019), *Unsupervised lemmatization as embeddings-based word clustering* [preprint available online at *https://arxiv.org/abs/1908.08528*].

Russell, K. (2018), *Read like a Roman: Teaching students to read in Latin word order*, in «Journal of Classics Teaching», 19, pp. 17-29.

Santorini, B. (1995), *Part-of-speech tagging guidelines for the Penn Treebank Project* (3rd revision, 2nd printing) [available online at *https://catalog.ldc.upenn.edu/docs/LDC99T42/tagguid1.pdf*, accessed on 14.07.2020].

Schmid, H. (1994), *Probabilistic part-of-speech tagging using decision trees*, in *Proceedings of International Conference on New Methods in Language Processing, Manchester, UK*, pp. 44-49.

Springmann, U., Schmid, H. and Najock, D. (2016), *LatMor: A Latin finite-state morphology encoding vowel quantity*, in «Open Linguistics», 2, pp. 386-392.

Sprugnoli, R., Passarotti, M., Cecchini, F.M. and Pellegrini, M. (2020), *Overview of the EvaLatin 2020 evaluation campaign*, in Sprugnoli, R. and Passarotti, M. (2020, eds.), *Proceedings of 1st Workshop on Language Technologies for Historical and Ancient Languages*, European Language Resources Association (ELRA), Paris, pp. 105-110.

Stoeckel, M., Henlein, A., Hemati, W. and Mehler, A. (2020), *Voting for PoS tagging of Latin texts: Using the flair of FLAIR to better ensemble classifiers by example of Latin*, in Sprugnoli, R. and Passarotti, M. (2020, eds.), *Proceedings of 1st Workshop on Language Technologies for Historical and Ancient Languages*, European Language Resources Association (ELRA), Paris, pp. 130-135.

Straka, M., Straková, J. and Hajič, J. (2019a), *Czech text processing with contextual embeddings: PoS tagging, lemmatization, parsing and NER*, in Ekštein, K. (2019, ed.), *Text, Speech, and Dialogue. TSD 2019* (Lecture Notes in Computer Science, vol. 11697), Springer, Cham, pp. 137-150.

Straka, M., Straková, J. and Hajič, J. (2019b), *Evaluating contextualized embeddings on 54 languages in PoS tagging, lemmatization and dependency parsing* [preprint available online at *https://arxiv.org/abs/1908.07448*].

Straka, M. and Strakova, J. (2020), *UDPipe at EvaLatin 2020: Contextualized embeddings and treebank embeddings*, in Sprugnoli, R. and Passarotti, M. (2020, eds.), *Proceedings of 1st Workshop on Language Technologies for Historical and Ancient Languages*, European Language Resources Association (ELRA), Paris, pp. 124-129.

Sychev, O. and Penskoy, N.A. (2019), *Method of lemmatizer selections in multiplexing lemmatization*, in «IOP Conference Series Materials Science and Engineering», 483, pp. 1-6.

Tarrant, R. (2016), *Texts, Editors, and Readers: Methods and Problems in Latin Textual Criticism*, Cambridge University Press, Cambridge.

Van Dam, H.-J. (1982), *A laughing Jabberwocky*, in «Wauwelwok: The Magazine of Het Nederlands Lewis Carroll Genootschap», 5, pp. 6-13.

Whitaker, W. (1993), *Words v.1.97F* [available online at *http://archives.nd.edu/whitaker/wordsdoc.htm*, accessed on 21.05.2020].

Wu, W. and Nicolai, G. (2020), *JHUBC's submission to LT4HALA EvaLatin 2020*, in Sprugnoli, R. and Passarotti, M. (2020, eds.), *Proceedings of 1st Workshop on Language Technologies for Historical and Ancient Languages*, European Language Resources Association (ELRA), Paris, pp. 114-118.

Zeman, D., Hajič, J., Popel, M., Potthast, M., Straka, M., Ginter, F., Nivre, J. and Petrov, S. (2018), *CoNLL 2018 shared task: Multilingual parsing from raw text to universal dependencies*, in Hajič, J. and Zeman, D. (2018, eds.), *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, Association for Computational Linguistics, Stroudsburg, PA, pp. 1-21.

Patrick J. Burns
Department of Classics
University of Texas at Austin
WAG 123
Austin, TX 78712 (United States)
patrick.burns@austin.utexas.edu