Converting Latin Prose to Poetry

Adante Ratzlaff

Project Goals

The objective of this project was to create a tool which would make it easier for someone with a basic knowledge of Latin to convert Latin prose into poetry without going through the tedious process of trying different word orderings by hand. Specifically, the baseline goals were to create a tool that would take in a Latin phrase that has been *macronized*, or had its naturally-long syllables marked, and look for ways to rearrange to phrase to fit at least one form of Latin meter.

Background

Unlike rhyme-based English poetry, Latin poetry is almost entirely based on meter, or the patterns of long and short syllables in each line. The length of each syllable in a Latin word or phrase is determined by a series of factors, each of which can be determined algorithmically in most cases, or looked up in a dictionary. Furthermore, in Latin and even more so in Latin poetry, the order of words in a sentence usually has little effect on the sentence's meaning due to the amount of information encoded morphologically in nouns, adjectives, and verbs. For example, the sentences "*hominem canis mordet*" and "*mordet hominem canis*" both mean "the dog bites the man", despite the arrangement of their component words. Thus, almost any phrase in Latin can be made into poetry if its words can be rearranged to fit an established meter.

Previously, classical language scholars have made some headway into algorithmically processing Latin meter and poetry. One researcher named Johan Winge created a macronizer, or a tool to automatically mark naturally-long vowels within Latin words, which claims to have up to 99% accuracy on classical texts. Additionally, several tools for *scanning* macronized Latin text, or determining the complete patterns of long and short syllables based on context, and analyzing the meter of Latin poetry have been created and added to the Classical Languages Toolkit, or CLTK, an open-source library of NLP tools for antique languages. However, converting prose to poetry seems to be an unfilled gap in the tools available.

Methodology

The tool for converting Latin prose to poetry, or the *metrifier*, was created using the following major components:

- A tokenizer/pre-processor which breaks a line of macronized Latin text into a list of individual words.
- An algorithm which efficiently searches the space of all possible permutations of a list of words, and returns any permutations which exactly match the meter

- A scanner which determines the meter of each permutation or partially-constructed permutation generated during the course of the above algorithm
- A metricality checker which determines whether the scanner's output matches

Results

A large portion of the time spent on this project was dedicated to an attempt to incorporate Winge's macronizer directly into the tool. Unfortunately, this proved to be operating-system dependent and difficult to set up. Even if it had been successfully implemented in development, including it in the final product would have required users to go through the same gruelling process to set it up on their own environments. Fortunately, the macronizer is available online in an easy-to-use format, so this trial did not greatly affect the final result.

Next, the pre-processor was straightforward to create. It first makes characters lowercase, then removes all characters other than letters and spaces, and then uses NLTK's word tokenizer to split the input into a list of strings.

The algorithm for trying permutations took more work, but it stood up to rigorous testing and is reasonably efficient considering that it needs to find every valid ordering of a list of words instead of merely the first.

The metricality checker uses a similarly smart algorithm, but, for the purposes of this proof-of-concept project, is hard-coded to compare its input to a single line of hendecasyllabic meter. Thus, the tool works well for short phrases, but it cannot convert a long sentence or a paragraph into a multi-line poem.

Finally, the scanner is the tool's one weak link. Imported from CLTK, it seems to use an over-complicated approach that involves tokenizing a sentence by syllables. This approach allows it to perform more-nuanced analysis on the input, but also introduces points of failure where the usual rules for determining syllable length or combining syllables through elision are ignored.

Analysis

Recall

The metrifier's recall was tested using a corpus of 530 lines of hendecasyllabic meter written by the classical poet Catullus. This corpus was macronized using Winge's macronizer rather than by hand, which, with its 98% to 99% accuracy per syllable, theoretically macronized somewhere between 10% and 30% of the roughly ten-to-fifteen-syllable lines in the corpus incorrectly. Each line in the corpus was then fed into the metrifier, and the metrifier was said to accept the line if it output the same word ordering in its list of valid meters. Between the macronizer and the metrifier, the overall recall was 63.0%.

Accuracy

For accuracy, the first “valid” permutation for twenty lines of Catullus from the aforementioned corpus was printed, alongside the reported scansion for the line. Operating under the assumption that the macrons were all correct, the permutations were then compared to their reported scansions to determine the accuracy of the scanner. Each reported scansion was also checked to see if it was valid hendecasyllabic meter. The results are shown in the table below.

Text	**Reported Scansion**	**Scansion is Correct**	**Scansion is Metrical**
cuī dōnō lepidum novum libellum	---uu-u-u-x	Yes	Yes
āridō modo pūmice expolītum	-u-uu-u-u-x	Yes	Yes
cornēlī tibi namque tū solēbās	---uu-u-u-x	Yes	Yes
meās esse aliquid putāre nūgās	u--uu-u-u-x	Yes	Yes
iam tum cum italōrum es ausus ūnus	u--uu-u-u-x	Yes	Yes
omne aevum tribus explicāre chartīs	-u-uu-u-u-x	No	Yes
hoc quārē tibi habē quidquid libellī	---uu-u-u-x	No	Yes
quālecumque quod ō patrōna virgō	-u-uu-u-u-x	No	Yes
plūs ūnō maneat perenne saeclō	---uu-u-u-x	Yes	Yes
passer dēliciae meae puellae	---uu-u-u-x	Yes	Yes
in quīcum lūdere quem sinū tenēre	---uu-u-u-x	No	Yes
et ācrīs solet incitāre morsūs	u--uu-u-u-x	Yes	Yes
cum dēsīderiō meō nitentī	---uu-u-u-x	Yes	Yes
et trīstīs animī levāre cūrās	---uu-u-u-x	Yes	Yes
tam grātum est mihi quam ferunt puellae	---uu-u-u-x	Yes	Yes
pernīcī aureolum fuisse mālum	---uu-u-u-x	Yes	Yes
lūgēte ō venerēs cupīdinēsque	---uu-u-u-x	Yes	Yes
et est quantum hominum venustiōrum	u--uu-u-u-x	Yes	Yes

passer mortuus est meae puellae	---uu-u-u-x	Yes	Yes
quem plūs suīs oculīs amābat illa	---uu-u-u-x	No	Yes
		Total Scansion Accuracy	**Total Scansion Metricality**
		15/20	20/20

In this test set, the scanner was shown to have 75% per-line accuracy, and the metricality checker was shown to have 100% per-line accuracy, for an overall metrifier per-line accuracy of 75%.

Conclusions

In conclusion, it is plausible to generate poetry from Latin prose, but there are two major obstacles: accurate macronization and accurate scansion.

Winge's macronizer is extremely sophisticated, but even one incorrectly-marked syllable in a line can cause a metrifier to fail. In the future, it would be interesting to work with a human-macronized corpus to see how results are improved.

Second, CLTK's scanner leaves much to be desired. It was only shown to be 75% accurate on short lines, and many of the mistakes it made during development looked like they could easily have been caught. Since an accurate scanner is essential for an accurate metrifier, building a new and improved scanner would be the best way to further this line of research.

Finally, it might be useful to accept sequences of words even if their scansions are off by one. This may improve recall significantly at the cost of some accuracy, but even the great Latin poets cheated on pronunciation from time to time to force their work to fit the meter.