Tagging and Statistically Translating Latin Sentences

Andrew Runge
Computer Systems Lab 2009-2010

Abstract

In developing language translation software, an increasingly common method is to tag words based on their role in the sentence in order to determine where they should be in the sentence, and then put them in that slot to create a basic, sometimes awkward translation. The goal of this project is to tag the sentences and then use a new method of statistically analyzing the words based on part of speech pairings in order to generate the most sensible and accurate translation of a Latin sentence.

Background

The biggest focus for language translation is to maintain the original meaning of the sentence when it is translated. As such, it is crucial not only to properly translate the words, but to maintain a sensible word order in order to preserve the original meaning. Machine learning methods, such as word tagging, allow the program to rule out possibilities for what the possible functions of a given word are. One such example used by McMahon and Smith was a method for determining the role of words in a sentence based on their context and similarities that they shared with other words. Another experiment by Bowden discussed a method of tagging the words for every possible set of characteristics they could have, and then systematically narrowing down the possibilities until you can more easily order the words based on their characteristics in Latin. This is a similar method to the one that I am employing. After identifying the words' roles in the sentence, then it is important to put them in the correct order via statistical analysis. Chen et al. demonstrate the effectiveness of statistical generation of sentence structure with their project using n-grams to create possible sentences. Another method of statistical analysis was demonstrated in Alam et al. In which they generated sentences based on the probability that two parts of speech could go together. Their program suffered largely due to the tagging and original translation methods that they used, but I intend to recreate the system they used for sorting the words into the correct order.

Secunda legio castra in Gallia habet, sed in Britanniam cum imperatore festinabit.

STAGE 1 The output from the first stage is as follows:

Second[ADJ-NomSingFem,VocSingFem,AblSingFem,NomPlurNeut,VocPlurNeut,AccPlurNeut] **legions**[NOUNFem-NomSing,VocSing] **camps**[NOUNNeut-NomPlur,VocPlur,AccPlur] **in**[PREP-+Abl]-OR-**into**[PREP-+Acc] **Gaul**[NOUNFem-NomSing,VocSing,AblSing] **he/she/it_have**[VERB-3rdSingPresIndAct] , **but**[CONJ] **in**[PREP-+Abl]-OR-**into**[PREP-+Acc] **Britain**[NOUNFem-AccSing] **with**[PREP-+Abl] **generals**[NOUNMasc-AblSing] **he/she/it_will_hurry**[VERB-3rdSingFutIndAct] .

Figure 1: Sample of the method employed by Bowden to tag all words based on all their possible sets of characteristics.

The Dictionary

My program uses to dictionaries in its translation. The first is a basic Latin dictionary, from which I will generate the meanings of the words. The second is a dictionary that is built from experience the program has in translating. Each of the words in that dictionary will be conjugated and declined forms of verbs, nouns, adjectives, etc. that my program will read in so that it can potentially save some time from having to generate new translations of the same words each time.

Original hypotheses	1. it's 5 minutes on foot . 2. it is 5 minutes on foot . 3. it's about 5 minutes' to walk . 4. i walk 5 minutes .
n-grams	it's 5 minutes, 5 minutes on, …… on foot ., about 5 minutes …… 5 minutes .

Fig. 2. Example of original hypotheses and 3-grams extracted from them.

partial hypothesis	it's	about	5	minutes	
n-gram expansion	+		5	minutes	on
new partial hypothesis	it's	about	5	minutes	on

Fig. 3. Expansion of a partial hypothesis through a matching *n*-gram.

New hypotheses	it's about 5 minutes on foot . it's 5 minutes . i walk 5 minutes on foot . ……
Reference	it's about five minutes on foot .

Figure 2: Demonstration of n-gram generation for determining word order in a sentence. Generated by Chen et al.

Tagging and Translation

The tagging process for my program consists of two major steps. The program first goes through the sentence word by word and determines each of the possible roles that the word could play in the sentence by analyzing its endings. It then generates a translation for each of those possible forms. To do that, it uses basic translations for each of the cases or persons, depending on if it is a verb or a noun. Once it has done this, it stores them into a data structure, so that they can then move on to the second stage and have some of these possibilities culled based on other words in the sentence.

```
>>> ================================ RESTART ================================
>>>
What sentence would you like to translate?puellis puerum rei
Time to make the dictionary is: 1.71600008011
Time to tag sentence is: 0.0
{'puerum': [['2SA', 'boy'], ['2SV', 'boy']], 'puellis': [['1PD', 'to the girls']
, ['1PB', 'the girls']], 'rei': [['5SG', 'of the thing'], ['5SD', 'to the thing'
]]}

Translation time is: 0.0
Total time taken is: 1.77800011635
```

Figure 3: Sample of code as it tags various forms of nouns and translates these forms

Results and Conclusions

The goal for this project will be to create an efficient Latin translator, which will both be able to identify key characteristics of words, as well as organize them into a sensible English sentence. The project will be tested on various forms of Latin prose and evaluated compared to human translations of the same lines. As of now, the program is able to perform much of the initial work needed to translate Latin sentences, such as tagging the words and creating an easily accessible dictionary to reference.