Machine Learning, Language Rules, and Statistical Translation

Andrew Runge

October 29, 2009

Abstract

Development of language translators, spoken or written, has most often used either rule-based or statistical strategies. In addition, machine learning is becoming one of the most efficient and effective methods for interpreting and deciphering text. Through the use of machine learning, the less common rule-based strategies may be implemented to greater effect. This project aims to use machine learning strategies to combine these two strategies to create an effective and efficient Latin translator. The project will be tested on several samples of Latin, including original Latin prose. The results will be studied for correct grammar, as well as compared to human translation of the same lines. The program will be done using python and the IDLE interface.

Keywords: Machine Learning, Statistical Translation, N-Gram

1 Introduction

The field of machine translation has been growing significantly over the past few years as a way to advance artificial intelligence and to make computers more capable of operating on their own in the real world. One such example of machine translation is through language translation software. Language translators have often been developed using two different strategies. Rule-based strategies are use to translate words properly so that their purpose in the sentence can be accurately defined. In addition, rule-based strategies have been used to try to properly put words in an intelligent order so that the resulting sentence makes sense. However, rule-based strategies are often not proficient in assigning word order. Another problem with rule-based strategies is that they are relatively inefficient in the code, and take up a lot of time to simply discern the role of a single word in a sentence. Through the use of machine translation strategies, such as n-gram generators, it can become quite simple and quick to generate information on each word in a sentence.

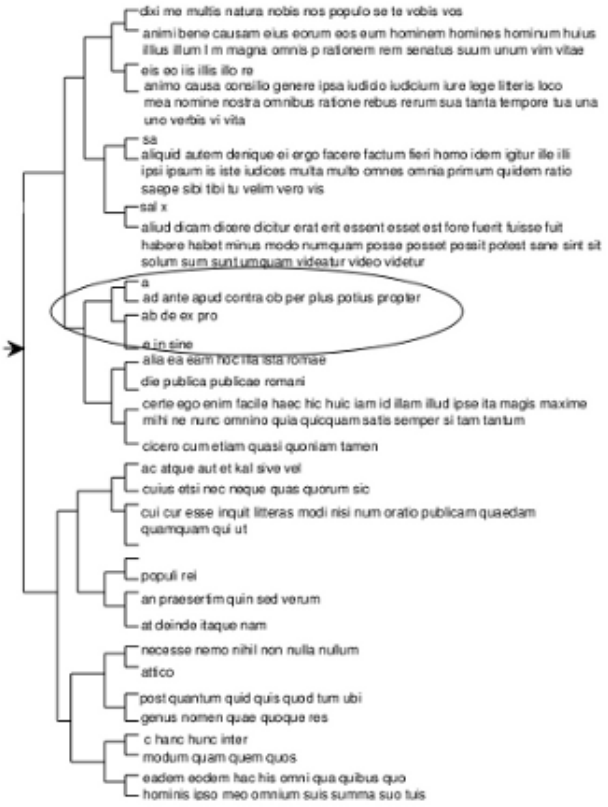

Figure 1: Tree of words sorted by sentence role from the assorted works of Cicero generated by the methods of McMahon and Smith.

N-grams are sets of words in a sentence, n words long, which can be used to identify word context, and from this its role in the sentence. In addition, n-grams are often used in the process of tagging words for things such as part of speech. Using n-grams and machine learning, it is possible to quickly and easily translate a sentence, but there still remains the problem of getting the word order in the sentence correct. This is where the second method, statistical analysis, comes into play. Statistical methods also make use of n-grams, but to a different effect. They use the n-gram and find the most common contexts of that word in various other situations to then generate partial n-grams of the given words in order to improve word order and subsequently sentence interpretation.

$$h_{IBM}(f_1^J, e_1^I) = \log\left(\frac{1}{(I+1)^J}\prod_{j=1}^{J}\sum_{i=0}^{I} p(f_j|e_i)\right)$$

Figure 2: An equation used to determine the accuracy of the hypotheses generated by Chen et al.

2 Background

The project tested in Two-Stage Hypotheses Generation for Spoken Language Translation used n-grams to generate multiple possible translations for a given sentence and then statistically selected the best one based on a series of tests to "score" each hypothesis.

McMahon and Smith also demonstrated use of n-grams in word tagging for part of speech purpose. They generated trees of the most common words within a lexicon and sorted them based on their context to determine what their part of speech was. They applied their method not just to English, but also to the collected works of Cicero in Latin. I plan to attempt to implement a basic version of what they did in order to identify important characteristics of the words, such as case, tense, person, etc.

3 Design and Procedures

The first thing that my program must be able to do is correctly identify all the important characteristics about each word. For nouns, this would mean case, number and gender. For verbs, this would mean person, number, tense. After that, the next step will be ensuring that it can translate rudimentary sentences. To do this, I will attempt to implement machine learning strategies via n-gram generators which can be used to identify im-

portant things about the context of words. From there, the next step will be applying statistical translation strategies on more complicated sentences in order to properly sort out the word order. For this stage, I will use sections of Latin prose from famouse authors. These sentences tend to have complicated word order and will provide a good test for my statistical alogrithm. The final step will be attempting to teach the to make assumptions about words in the sentence in an effort to have the sentence make sense. For this stage, I will test it on Latin poetry, which often has words left out, or uses some words to mean other things.

4 Expected Results and Discussion

I expect that my program will be able to properly translate Latin sentences using the methods detailed in my introduction and design procedures. I will test the program on several sections of Latin, ranging from very basic sentences whose words are already in the correct English order, all the way up to original Latin prose. As it stands, my program currently can translate individual words, but cannot yet translate full sentences in an intelligent manner. Further research in this area can be done to improve the use of n-grams and their efficiency. In addition, further research can be done in improving statistical methods used to generate the hypotheses for translation.

References

[1] Boxing Chen, Min Zhang, and AI TI AW., "Two-Stage Hypotheses Generation for Spoken Language Translation", *ACM Trans. Asian Lang. Inform. Proccess. 8, 1*, Article 4, 22 pages March 2009

[2] J. McMahon and F.J. Smith "Structural Tags, Annealing and Automatic Word Classification", *Sruct. Tags, Word Class.*, Queen's University of Belfast, May 1994