French/English Translation

by Sharon Ulery
TJHSST Computer Systems Lab 2006-2007

Abstract

This project uses hard-coding techniques in the field of computational linguistics to translate French to English and English to French well enough to be understandable to someone who knows only the output language.

Introduction

Starting with a word-for-word translation from French to English and vice versa, I gave the program grammar rules so that it correctly translates increasingly complex grammar structures. The translation mechanism uses only a small number of grammar rules because encoding extensive rules amounts to data entry, which is not pedagogically useful. Using the same technique, however, the program could simply be 'taught' to translate all grammatically correct, non-idiomatic sentences in both languages with correct agreement of number and gender, assuming a perfect knowledge of each language's grammar.

Current, commercial translation programs usually use a more sophisticated, statistical technique further discussed under “Background.”

Purpose

This project uses computational linguistics to serve students of French or English as a non-native language as well as those who know only one of these languages. The output should be understandable even to someone who knows only the output language. Even a less than perfect translation is useful for surfing the web, reading texts (especially instructions) in a foreign language, and communication with someone with whom one has no language in common. It can also be used for students to check their writing in the target language by translating back to their native tongue.

Results

There are two methods of language processing that can be used in translation. Statistical analysis is used in most web-based and commercially available translation programs. Using that method, the translation program "learns" grammar rules by using a few basic rules and going over a large corpus of written material in both languages. This can be contrasted with the method I used, in which grammar rules are hard-coded into a program. The biggest obstacle in using hard-coded grammar rules is that to create a sufficient number of rules and a large dictionary requires a lot of data entry. As this wasn't beneficial in a senior tech lab project, the translation program I wrote is very primitive. However, by expanding the program I created in a straightforward way, further grammar rules can simply be coded in, and the small dictionary I used can be easily expanded or replaced. The program currently fits its specifications exactly, although those specifications are not extensive. If the program were extended through additional data entry, however, it would continue to exactly fulfill specifications as they grew.

Procedures and Methodology

Input: The user works through a GUI to first specify the original language of the text. Then, still through a GUI, the user inputs the phrase to be translated.

Output: Output will be in the terminal. For input using only words in the dictionary and acceptable forms thereof, output is the literal equivalent of the root words in the order they appear in the non-input language.

Requirements: The program is written in Java. It uses a small bilingual dictionary, but a large bilingual dictionary would greatly extend its capabilities A technique for inputing and outputting accent marks would also be valuable.

Presentation of results is relatively simple. A list of a variety of sample inputs and corresponding outputs should allow clear evaluation of the results.

Testing is also relatively simple. As I increased capability, I used specific structural and functional testing of the new grammatical structures the program includes. From time to time, including at the end of the project, I used dynamic testing to make sure that there were no hidden bugs.

Background

The field of computational linguistics originally used mostly word-for-word translation. As the field has developed, new techniques for parsing and translating natural language have been introduced. Eventually, various techniques relying on statistics have become the favored methods for natural language processing (NLP).

The advantage in using statistical techniques in NLP is that it is more robust: the program can grow and change as the material it works from grows and changes, and it doesn't require a division of language into "grammatical" and "ungrammatical" statements, which in English is an artificial boundary. Instead of having "grammatical" and "ungrammatical" statements, there are just syntactic structures that people are more or less likely to write.

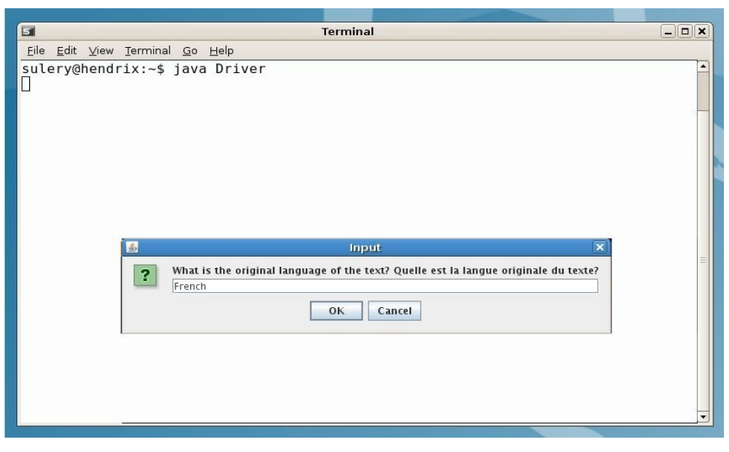
Statistical translation techniques rely on a large bilingual corpus, which is a large text in both languages that was created using a human translator. The program then aligns sentences that say similar things in both languages and check to see how often a given word of the first language occurs when a given word of the second language appears in the matching sentence. The more often it happens, the more likely they are to translate each other.

Statistical techniques can be used at a variety of levels, from creating a dictionary as described above to learning about syntactic and even semantic organization at the level of paragraphs or even larger chunks of text. The program compares the frequency of various structures occuring in the translation language given various structures in the originial language to find how different grammar forms are best translated.

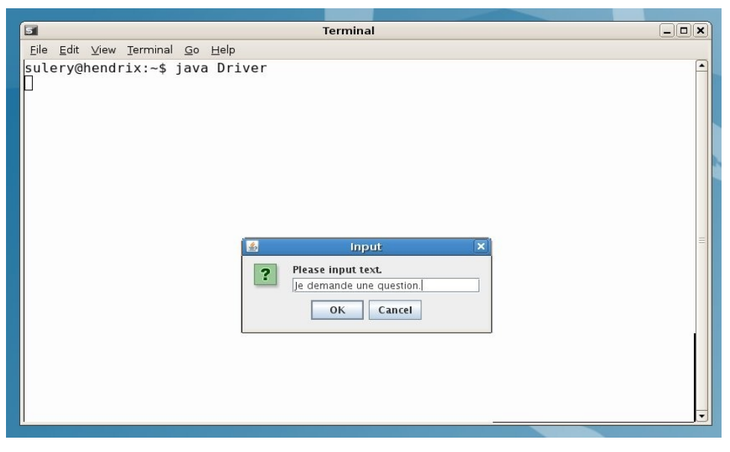
Unfortunately, using the full statistical techniques recommended by these sources is beyond the scope of a year-long, high school project. I have, therefore, created a translation program using word-for-word translation.

Screen Shots:
Three stages of the Program

1)



2)



3)