Language Identification Programming Project

Faculty Project Description

Introduction

In this project, students write a computer program, in Java or C, which computes a histogram showing the frequencies of individual letters in a text. Students then modify their code to use the histogram to identify the language of the text. The resulting code should be able to differentiate between, for example, texts in English, Italian, and German. In the next section, the history of using machines to identify languages and the related problem of machine translation are discussed. In the following section, the skills and techniques used in completing this project are discussed. Trade-offs are discussed in the following section. In the next section, resources that are needed to complete this project are discussed. Variations, or tracks, of this project are discussed in the following section. The last section concludes.

Background on the Problem

People describe the world, tell stories, share ideas, and communicate in over 6900 languages [1]. Information on the internet may seem limitless. However, a person who speaks one language can access only a fraction of the available information. For example, the website Wikipedia contains articles in 262 languages, and as of July 2007, only 23% of the articles were in English [2]. WorldPress [3] is a website that describes itself as "a nonpartisan magazine whose mission is to foster the international exchange of perspectives and information." It contains approximately seven million blogs, and only around 36% are in English [4]. Throughout history, Electrical and Computer engineers have built tools, including the telegraph, telephone, and internet router, which have helped people communicate. Computer software which can translate between languages is one such tool. The first step of translating a text is to identify its language. In this project, students write a computer program, a tool, which accomplishes this first step.

The history of the problems of translation and language identification are closely related because both aid communication, and their solutions require similar skills. The idea of using computers to translate text was first proposed in a letter from Warren Weaver to Norbert Wiener in March 1947 when computers were still in their infancy [5]. Warren Weaver is best known for coauthoring, along with Claude Shannon, the book "The Mathematical Theory of Communication" [6] and for translating "Alice in Wonderland" into multiple languages [7]. Norbert Wiener is known for founding the field of cybernetics and for his work on filtering [8]. In the letter, Weaver emphasizes the importance of aiding communication [5]. "A most serious problem, for UNESCO and for the constructive and peaceful future of the planet, is the problem of translation, as it unavoidably affects the communication between peoples." In 1949, Weaver wrote a memorandum expanding his ideas on computer translation and introducing the topic to a wider audience [5]. His memorandum begins [5], "There is no need to do more than mention the obvious fact that a multiplicity of languages impedes cultural interchanges between the peoples of the earth, and is a serious deterrent to international understanding." Improving communication is still a high priority for the world community today. In 2000, the UN hosted a Millennium Summit described as "the largest gathering of world leaders in history" [9]. One of the agreed upon goals was, "in cooperation with the private sector, make available the benefits of new technologies, especially information and communications technologies."

The first attempts at computer translation began in the late 1940s and used a dictionary approach where words or short phrases were translated individually [5]. Over the next few decades, the next generations of machine translation systems were built for research purposes, government use, and commercial use [10]. By the mid 1960s, two of the most advanced systems were Systran and the Mark II. Both of these systems went beyond a dictionary approach to translation incorporating frequency domain and statistical ideas. The Mark II was developed by researchers at IBM and Georgetown and was demonstrated at the 1964 New York World Fair [10]. The Systran system was developed by Toma and others [10], and in 1974, it was used by NASA for translating between Russian and English during the Apollo-Soyuz project [10]. Machine translation became widely available with the blossoming of the internet. In the 1990s, Altavista's Babelfish translation service, based on the Systran system, was introduced online [11]. Google began working on a translation tool around 2001 [12], and today Google's system can translate amongst 47 languages [13]. The "Detect Language" feature was added to Google Translate quite recently, in May 2008 [14]. In this project,

students write their own version of this feature.

Background on Skills and Techniques

This project introduces students to a number of ideas which are encountered by Electrical and Computer engineers on a regular basis. Specifically, students are introduced to the concepts of frequency analysis, the statistical nature of a signal, and computer programming.

The first step of this project is to compute a histogram which shows the letter frequencies. Frequency analysis is one of the most fundamental techniques for obtaining information about signals. Fourier's work on the topic has been called "One of the greatest advances in the history of mathematics" [15]. It is common for electrical and computer engineering curricula to devote at least a semester to the topic. Most introductions to the topic, however, involve Fourier transforms and require a knowledge of calculus. In this project, students are introduced to the concept of frequency analysis in a concrete way that does not require integrals, infinite sums, or other higher mathematics.

The idea of using frequency analysis to find out information about a text is a technique which has been successfully used many times. The idea can be traced at least as far back as Weaver's original memorandum [5]. In reference [16] for example, the frequency of small words like 'by', 'to', and 'upon' was used to determine whether some of the Federalist Papers were written by Alexander Hamilton, James Madison, or John Jay. In reference [17], the study of word frequencies was used to distinguish whether letters were written by Jane Austen or by other authors of the same time period. In reference [18], it was used to determine whether or not a poem was written by Shakespeare. A similar technique was used in reference [19] to determine whether or not letters written under the name 'Quintus Curtius Snodgrass' were written by Mark Twain. Frequency analysis also successfully has been used to find out information about other types of works. In reference [20], it was used to help identify the artist of a painting, and in reference [21], it was used to help identify the programmer of a piece of computer code. It has also been used to determine if works, including audio files [22] and pictures [23], have been tampered with or illicitly altered.

Real signals almost always contain some amount of noise or randomness. It is often difficult for students to understand that signals may be stochastic and that solutions to a problem may contain some uncertainty because textbook problems typically deal exclusively with deterministic signals. For example, reference [24], one of the most commonly used texts for Signals and Systems courses, limits its discussion to deterministic signals. In this project, students plot letter frequency versus length of the text analyzed. This signal is not constant, but it obeys the Central Limit Theorem by converging for large lengths of text. The algorithm that students write to determine the language of the text does not need to be complicated. However, it must deal with the fact that the letter frequencies are not constants, and it must deal with the fact that the precise values of the letter frequencies might be different if, for example, 5000 or 10,000 characters of the text are analyzed.

This project is also intended to teach students about programming. Programming is a skill that is useful for virtually all engineers. One programming concept that is difficult to convey to students, yet vitally important in almost all real world programming situations, is that programming is not an individual job. A programmer typically writes code for a user other than himself or herself. Users may have very different backgrounds, skill levels, and expectations than the programmer, so it is the job of the programmer to try to understand the user. Because this project involves analyzing texts of different languages, students should consider that the user might speak a different language than the programmer. Relatedly, students work on this project in pairs or groups of three, so they must communicate with their partners. During the design, implementation, and testing phases of the project students gain experience working collaboratively.

Another concept related to programming which is difficult to convey to students is that writing code is only a small fraction of the job of a computer programmer. When working on a project, the programmers must design an algorithm, design the program's user interface, write the code, test the code, and debug it. Quoting Brian Kernighan [25], one original authors of the UNIX operating system, "Everyone knows that debugging is twice as hard as writing a program in the first place." In this project, students encounter all of these steps, not just algorithm development.

Trade-offs Encountered

Engineers always face trade-offs when designing products whether the products involve, hardware, software, or both. The concept of trade-offs is often difficult for students because, like randomness, it is rarely encountered in textbook problems. In this project, students will encounter a trade-off involving the length of text analyzed. As the length of text analyzed increases, the letter frequency converges and the accuracy of the detection algorithm likely improves. However, the time needed for computation also increases. Students plot both letter frequency and computation time versus length of text analyzed to illustrate this trade-off.

Students may also a encounter trade-off involving the number of languages considered. A simple algorithm may be used to determine if a text is written in English or Italian. If the normalized percent frequency of the letter 'a' is over 9.2, the text is identified as Italian, and otherwise it is identified as English. An algorithm, that works in most cases, to differentiate between English, Italian, and German is to identify texts as Italian if the normalized frequency of 'a' is over 9.2, identify the text as English if the normalized frequency of 'a' is less than 9.2 and the normalized frequency of 'o' is more than 5.0, and to identify the text as German in other cases. Many other algorithms are possible. To differentiate between more languages, a more complicated algorithm is needed. As the complexity of the algorithm goes up, the computation time may also increase. Additionally, complex algorithms are more difficult to test and debug. The accuracy of the language identification is also likely to decrease when more languages are considered.

Resources Needed

Since this project primarily involves programming, no specialized tools or hardware are needed. Students need access to computers and compilers. Students also need access to the internet to obtain texts to analyze. Students need texts in multiple natural languages to analyze. Project Gutenberg [26] is an online database, dating back to 1971, which contains over 30,000 books. All of the books are public domain and are available online in plain text which makes them ideal for student analysis. Project Gutenberg contains books in 46 languages, and it contains more than 50 books in 13 languages. Most of the texts from Project Gutenberg have a license written in English included at the beginning or the end. The license may need to be deleted manually before the texts can be analyzed.

This project is intended to take one week of class time. In cases where teachers need to introduce additional background information before students can begin the project, two weeks should be devoted to it to provide time for additional introductory lectures.

The instructor should provide some computer code to the students. In this way, students will be able to focus on writing the parts of the code most fundamental to the project. Example code is provided in the files App3_JavaCode and App4_CCode in Java and C respectively. These computer languages were chosen because they are the most popular based on the number of queries in online search engines [27]. Also, compilers for these computer languages are free and widely available [28] [29]. However, this project may be readily adapted to other programming languages. Comments within the files App3_JavaCode and App4_CCode separate the code into three blocks. Block one demonstrates reading characters in from a file and using a 'for' loop. Block two calculates the histogram, and block three identifies the language. The code should be modified as described below before it is given to the students.

Variations of the Project

Many variations of this project are possible. Two tracks will be discussed. Track one is intended for students in an introductory programming course. Track two is intended for students with less experience. Modifications for students with more experience are also discussed below.

The first track is intended for students who have had approximately half a semester of an introductory programming course. Students should be familiar with how to use a text editor or integrated development environment and a compiler. Students should also be familiar with flow control statements such as 'for' loops and 'if' statements along with conditionals including 'less than', 'greater than', && (and), || (or), and 'equal to'. Students should also be able to get input from the console and produce output to the console using commands such as 'printf'. Additionally, students should be able to take the data produced by their code and plot it using a spreadsheet or other external program. It may be helpful, but is not necessary, for students to be familiar with the concept of arrays. The concept of pointers is often presented in introductory

C courses but is not needed for this project. The concept of objects is often presented in introductory Java courses. It is not necessary for students to have a solid understanding of objects to complete this project. However, it is difficult to accomplish anything in Java without having a cursory understanding of objects.

For track one, students should be provided with a modified version of the file App3_JavaCode or App4CCode. More specifically, the instructor should delete blocks two and three and provide the students with the remaining code. Students are then responsible for writing these blocks themselves. The remaining provided code reads in thirty characters from a file called intext.txt and prints them to the screen. It demonstrates the conversion between ASCII characters and integers. It also demonstrates how to measure the time of computation using the currentTimeMillis method (for Java) or the functions in time.h (for C). Since examples of all of these topics will be provided, students do not need to know them before the project. However, it may be helpful for the teacher to introduce these topics at the beginning of the project. Students following track one should use the version of the student project assignment which is found in the file M2_Assign.pdf. This track involves three steps: Calculating the histogram, analyzing example texts, and identifying the language.

The second track is intended for students with either no programming experience or with less time to devote to the project. Students on this track should be familiar with how to plot data using a spreadsheet or other external program. A teacher should explain to the students at the beginning of the project how to use a text editor or integrated development environment and how to use a compiler. The teacher should also explain flow control statements such as the 'if' statement and how to display to the console with commands such as 'printf'. Also, information should be presented to the students on conditionals such as 'less than', 'greater than', and 'equal to'.

For track two, students should be provided with a modified version of the file App3_JavaCode or App4CCode where only block three is deleted. Students on track two will follow the alternate version of the student assignment in the file App5_AltAssign.pdf. This version of the assignment involves only two steps: analyzing example texts and identifying the language. Students do not need to write any code to analyze the example texts because they can use the provided code. Students should design an algorithm for identifying the language. Students should implement their algorithm with help from the teacher as needed.

Other tracks or variations of this project are possible for students with more programming experience. More advanced students could complete the assignment without any starting code provided. Additionally, advanced students can modify their code so it can identify more than three languages. The code can also be modified to identify the language of a website instead of a text file. Advanced students may also want to improve the speed of their code. These ideas are listed in the student project assignment.

Conclusion

The history of Electrical and Computer engineering is filled with inventions that have improved communication. In this project, first year students build a tool that aids communication. More specifically, students write a computer program which identifies the language of a text. The idea for this project dates back at least to the work of Weaver in the late 1940s, but was implemented as a feature in Google only about a year ago.

References

[1] M. P. Lewis, *Ethnologue: Languages of the World*, 16th ed. SIL International, 2009.

[2] "Wikipedias," http://en.wikipedia.org/wiki/Wikipedia, date accessed: June 24, 2009.

[3] "Worldpress website," http://worldpress.org, date accessed: June 24, 2009.

[4] L. Berlin, "A web that speaks your language," *New York Times*, May 2009.

[5] W. Weaver, "Translation," *Machine Translation of Languages: Fourteen Essays*, pp. 15–23, 1949, http://www.mt-archive.info/Weaver-1949.pdf, Date accessed: June 24, 2009.

[6] C. E. Shannon and W. Weaver, *The Mathematical Theory of Communiction.* University of Illinois Press, 1949.

[7] W. Weaver, *Alice in Many Tongues.* University of Wisconsin Press, 1964.

[8] N. Wiener, *Cybernetics.* MIT press, 1965.

[9] "Millenium project," http://www.unmillenniumproject.org/goals/gt.htm, date accessed: June 24, 2009.

[10] W. J. Hutchins, *Early Years in Machine Translation.* John Benjamins Publishing, 2000, http://books.google.com/books?id=CsPYkBRva1gC.

[11] R. Balkin, "Altavista's automatic translation program," *Database*, pp. 56–57, Apr. 1999.

[12] "Google blog July 18, 2008," http://googlebolg.blogspot.com/2008/07/hitting-40-languages.html, date accessed: June 24, 2009.

[13] "Google blog June 9, 2009," http://googleblog.blogspot.com/2009/06/ translating-worlds=information-with.html, date accessed: June 24, 2009.

[14] "Google blog May 15, 2008," http://googleblog.blogspot.com/2008/05/ google-translate-adds-10-new-languages.html, date accessed: June 24, 2009.

[15] E. A. Robinson, "A historical perspective of spectrum estimation," *Proceedings of the IEEE*, vol. 70, no. 9, Sept. 1982.

[16] F. Mosteller and D. L. Wallace, "Inference in an authorship problem," *Journal of the American Statistical Association*, vol. 58, no. 302, pp. 275–309, June 1963.

[17] D. A. Graves, "Vocabulary profiles of letters and novels of Jane Austen and her contemporaries," *Persuasions On-line*, vol. 26, no. 1, 2005.

[18] R. Thisted and B. Efron, "Did Shakespeare write a newly-discovered poem," *Biometrika Trust*, vol. 74, no. 3, pp. 445–455, 1987.

[19] C. S. Brinegar, "Mark Twain and the Quintus Curtius Snodgrass letters, a statistical test of authorship," *Journal of the American Statistical Association*, vol. 58, no. 301, pp. 85–96, Mar. 1963.

[20] S. Lyu, D. Rockmore, and H. Farid, "A digital technique for art authentication," *Proceedings of the National Academy of Science*, vol. 101, no. 49, pp. 17 006–17 010, Dec. 2004.

[21] P. Sallis, A. Aakjaer, and S. MacDonell, "Software forensics: old methods for a new science," *Software Engineering: Education and Practice*, pp. 481–485, 1996.

[22] L. Hunyadi, K. Abari, and E. Toth, "Forensic linguistics, its contribution to humanities computing," *Literary and Linguistic Computing*, vol. 18, no. 1, pp. 49–62, 2003.

[23] H. Farid, "Detecting digital forgeries using bispectral analysis," *Technical Report AIM-1657*, 1999.

[24] B. P. Lathi, *Linear Systems and Signals.* Oxford, 2004.

[25] B. W. Kernighan and P. J. Plauger, *Elements of Programming Style.* McGraw Hill, 1978.

[26] "Project gutenberg," http://www.gutenberg.org, Date accessed, June 24, 2009.

[27] "Tiobe software BV website," http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html, date accessed: June 24, 2009.

[28] "Java compiler," http://developers.sun.com/downloads, date accessed: June 24, 2009.

[29] "GCC compiler," http://gcc.gnu.org, date accessed: June 24, 2009.