Home-Made Demonstration Software for Teaching Computational Linguistics

Zhang, Xiaoheng

Department of Chinese and Bilingual Studies, Hong Kong Polytechnic University

E-mail: ctxzhang@polyu.edu.hk

Abstract: Commercial software packages for natural language processing are normally large, complicated, expensive, and black box-like to users. Hence it seems worthwhile to create our own demonstration software for university teaching and learning. Such home-made software developed by the subject teachers is more open and transparent. Students are allowed to look inside the software either dynamically or statically and to contribute further improvement, which will in turn help them to gain a better understanding of the subject under study and to cultivate their creativities. The present paper discusses the design and application of a number of home-made demonstration software tools for computational linguistics lessons in the Department of Chinese and Bilingual Studies, Hong Kong Polytechnic University.

Keywords: home-made demonstration software, computational linguistics, Chinese and English teaching.

1 Introduction

Computers are playing an increasingly important role in education. Various software and facilities have been created or employed for CALT (Computer-Assisted Learning and Teaching). In the recent years, there has been a boom for the employment of multimedia technology and the WWW. Even email and blog services have been effectively used for teaching and learning (Gu, 2006; Xie, 2006). Yet little attention has been given to classroom demonstration software, i.e., the software used by instructors for demonstration in class, as a working model of the knowledge being learned. The categories of CALT Software, according to Vockell and Schearts (1992), are given as drill and practice, games, simulations, tutorials, tools, hypermedia, and tests. And none of them are specially created for classroom demonstration, in spite of the fact that demonstration is an important component of quality teaching and learning for many subjects, especially artificial intelligence and computational linguistics.

It is possible to use commercial application software for classroom demonstration. For example, in the University of Exeter, a marketed machine translation program called Power Translator was used for demonstration and performance analysis as part of a course in Natural Language Processing Application (Lewis, 1997). Yet this may not be sufficiently effective. Normally commercial software systems are not created for teaching purposes, they are likely to be too complicated and prevent the teachers and students from looking into the black boxes, which will in turn hinder comprehensive and deep understanding of the knowledge being taught. As for computational linguistic lessons, it seems even less worthwhile to utilize software of that kind as CALL (Computer-Assisted Language Learning) demonstration software, because commercial software packages for natural language processing are normally large, complicated, expensive, rare and immature.

CALL demonstration software is different from other demonstration software for language processing as well. The latter is mostly built for the purposes of product advertising, theory proving or fund raising, while the former puts emphasis on effective teaching and learning. A good CALL demonstration program for university teaching is expected to be student-centered and to provide a clear and easy-to-understand presentation of the subject knowledge to be learned. In addition, it should encourage brave exploration, active discovery, creative thinking and even further development. The present paper discusses the design, implementation and application of several demonstration software prototypes developed and applied in a integrated mode (Lun, 2005) by the author for his lessons in language information processing, including an English parser, a concordance program and an electronic dictionary.

2 An English Parser

A parser is a program for syntactical analysis of a natural language. Our demonstration program is an English chart parser (Allen 1995), where analysis is done in both top-down and bottom-up directions. The software was written in Prolog programming language and consists of three basic components: the user interface, the syntax analyzer and the knowledge-base, which includes a dictionary and a linguistics rule-base, as illustrated in Figure 1.

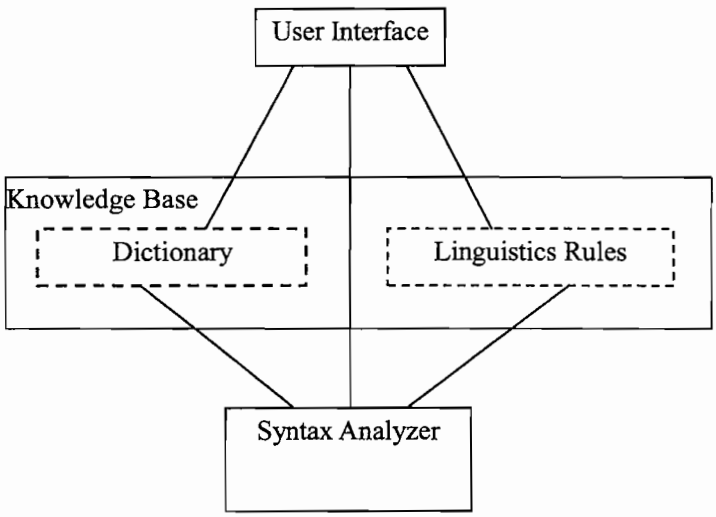

Figure 1 Structure of the educational demonstration parser

Acceptable input includes individual English sentences, noun phrases, verb phrases and prepositional phrases. When started, the parser displays a menu for the user, with options of all the acceptable input categories. The user can make a choice and input a corresponding phrase or sentence to be analyzed, or alternatively, ask the parser to draw an existing sample from the system's example database of sentences and phrases. The user can also trace the reasoning and analyzing procedures of the program. When the analysis is done, a Success or Failure message is

reported, depending on whether the input text is regarded as grammatically correct or not according to the system's linguistic knowledge. In the case of a success, a syntax tree is generated, which can be displayed if required. A sampled user-parser interaction screen output is as follows:

```
?- parse.
Welcome to use our small Mixed-Mode Parser!
What are you going to process?
s: sentence
np: noun phrase
vp: verb phrase
pp: prepositional phrase
q: exit.
Please input an element of [s,np,vp,pp,q]
pp.
trace? (y/n)
Please input an element of [y,n]
n.
Pick up an existing sample? (y/n)
Please input an element of [y,n]
y.
Sample List Form:
    [in,a,room]
SUCCESS
Print out the Analysis Result?
Please input an element of [y,n]
y.
         |pp
            |prep
            |   |in
            |pobj
               |np
                  |num
                  |   |s3
                  |art
                  |   |num
                  |   |   |s3
                  |   |head
                  |   |   |a
                  |noun
```

```
|num
|   |s3
|head
    |room
```

Where `prep` stands for preposition, `pobj` prepositional object, `num` number, `s3` singular number and 3rd person, `head` head word (of a phrase).

The form of the grammar tree is adjustable. Users can define the depth (or maximum number of layers) of the tree. The user can also directly browse and edit the linguistic rules and dictionary entries and test the effect.

3 A Concordance Program

Concordance programs, also called concordancers, are vital to corpus linguistics, i.e., linguistic studies based on large corpora of texts (McEnery, et. al., 2006). PUCON (standing for PolyU CONcordancer) is a practical program written in C, capable of retrieving from its target corpus (either Chinese or English) useful contextual information for keywords or keyword patterns given by the user. When running on our 5,000,000-characters Chinese corpus with sample newspaper texts from Hong Kong, Taiwan and the Mainland of China, the concordancer can also group output concordance items into three sub-concordances, corresponding to the three sources of geographical regions.

PUCON is in a modular and hierarchical structure. Lower-level components are independent of any specific language or corpus. The major components of PUCON include the input component, the information retrieving component, the sorting and formatting component, and the output component, as shown in Figure 2. More details about the system are given in a separate paper (Zhang and Cheung 1995).

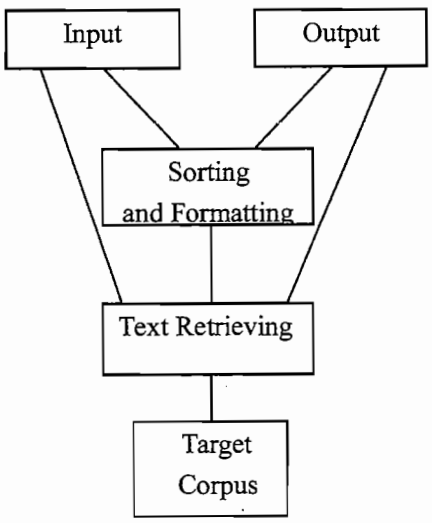

Figure 2 System Design of PUCON

PUCON has been successfully used both as a demonstration program in class and as a useful tool for information retrieval to support linguistic studies. In fact, quite a few of undergraduate and postgraduate students in the Department of Chinese and Bilingual Studies of the Hong Kong Polytechnic University have benefited from PUCON for linguistics analysis in their final year dissertation projects.

4 An Electronic Dictionary

This is a very small English-Chinese electronic dictionary, which also serves as a convenient seed program for students to develop into more complete and valuable software. Dictionary entries are represented as facts in Prolog with several arguments, e.g.,

word (pen, 鋼筆, gang1bi3, n, object, 'to write a letter with a pen')

Where "word" is the predicate name. The first argument in brackets is an English word. Its Chinese counterpart comes second. Pinyin third, followed by part of speech, the semantic category of the word and an example English sentence or phrase. The dictionary has been implemented in Prolog programming languages. It can also be easily built up on a database management system or with JavaScript on the WWW.

Due to its small scale and simplicity, the program can be created anew by the instructor in the classroom, together with the students. Programming and explanation by the instructor will halt when a tiny English-Chinese electronic dictionary is formed and running. And students are encouraged to take it as a seed to develop their own software. As the seed is so simple and open to the students, they soon get a thorough understanding of its working mechanism and the important programming techniques employed, and become ready to contribute further development. As a matter of fact, quite a few of interesting programs have been developed by our students in this way, including an English-Chinese bi-directional dictionary, an information retrieval program for Hong Kong KCR trains time tables at each station along the rail-line, and a simple CALL program for Chinese children to learn English prepositions.

5 Conclusion and Discussion

In the previous sections, we have introduced several of our home-made computational linguistic CALL programs for classroom demonstration, including an English syntax parser, a Chinese concordancer and a bilingual dictionary (The section on machine translation in the draft paper has been deleted due to limit of available space). Though still quite small in size and remaining in the state of prototype, these software tools are capable to demonstrate the essential functions of the typical systems in their families. They are presented to the students as glass boxes instead of black boxes. Special care has been taken to make the working mechanisms easy to understand, for the purpose of an effective demonstration of the knowledge students are supposed to explore, to learn and even to enrich. Students are allowed to study the software tools when they are running, or simply read the program files, under the help of the instructor, who is

the designer and developer of the software. The combination of such dynamic and static studies has proved very effective. The students soon become active co-researchers with their instructor in the studies of computational linguistics, which is by all means helpful to enhance their knowledge of both language and information technology.

Though our discussion has been concentrated on the field of computational linguistics and language information processing, the basic ideas should be equally applicable to other education domains as well. There is a report (Jonassen, 1993) that it facilitates effective and deep learning when students are asked to use expert system shells to build expert systems in the domain of knowledge they are learning. Home-made demonstration software gives even more control power and flexibility to the students, because the software was totally created by the instructor. Sometimes it may be worthwhile to create an education front-end on commercial software instead of building the whole system from scratch (Micarther, 1995). The basic intention of this paper is to draw more and adequate attention of teachers and education researchers to the development and application of home-made demonstration software to facilitate teaching and learning.

References

[1] Allen, J. F. (1995). *Natural Language Understanding*. Redwood City, CA: Benjamin/Cummings.

[2] Gu, P. (顾佩亚 等, 2006). 计算机辅助语言教学理论与实践. 上海：复旦大学.

[3] Jonassen, D. H., at. el. (1993). Constructivist uses of expert systems to support learning, *Journal of Computer-Based Instruction*, Vol. 20, No. 3.

[4] Lewis, D. (1997). Machine translation in a modern language curriculum. *Computer Assisted Language Learning*, Vol. 10, No 3, pp. 255-271.

[5] Lun, S. (2005). *An Integrated Approach to Computer-Assisted Language Learning*. Hong Kong: LangComp.

[6] McEnery, T., Xiao, R. and Tono, Y. (2006). *Corpus-Based Language Studies: An advanced resource book*. Oxford: Routledge.

[7] Micarthur, D. (1995) et. al. ESSCOTS for learning: Transforming commercial software into powerful education tool, *Journal of Artificial Intelligence in Education*, Vol 6. November 1, 1995.

[8] Vockell, E. L. and Schearts., E. M. (1992). *The Computer in the Classroom*. Watsonville, CA: Mitchell McGRAW-HILL.

[9] Xie, T. (2006). Blog, wiki, podcasting and learning Chinese language. In Zhang, P. Xie, T. et. al. eds., *Research and Application of Digitized Chinese Teaching and Learning*. Beijing: Yuwen Press, pp 429-436.

[10] Zhang, X. and Cheung, Y. S. (1995). Development of a flexible concordance program for large corpora, *Proceedings of the Joint Meeting of the Fourth International Conference on Chinese Linguistics and Seventh North American Conference on Chinese Linguistics (ICCL-4/NACCL-7)*, June 1995. (Madison, USA).