Context-Sensitive Electronic Dictionaries

Gábor PRÓSZÉKY
MorphoLogic
Késmárki u. 8.
1118 Budapest, Hungary
proszeky@morphologic.hu

Balázs KIS
MorphoLogic
Késmárki u. 8.
1118 Budapest, Hungary
kis@morphologic.hu

Abstract

This paper introduces a context-sensitive electronic dictionary that provides translations for any piece of text displayed on a computer screen, without requiring user interaction. This is achieved through a process of three phases: text acquisition from the screen, morpho-syntactic analysis of the context of the selected word, and the dictionary lookup. As with other similar tools available, this program usually works with dictionaries adapted from one or more printed dictionaries. To implement context sensitive features, however, traditional dictionary entries need to be restructured. By splitting up entries into smaller pieces and indexing them in a special way, the program is able to display a restricted set of information that is relevant to the context. Based on the information in the dictionaries, the program is able to recognize—even discontinuous—multiword expressions on the screen.

The program has three major features which we believe make it unique for the time being, and which the development focused on: linguistic flexibility (stemming, morphological analysis and shallow parsing), open architecture (three major architectural blocks, all replaceable along public documented APIs), and flexible user interface (replaceable dictionaries, direct user feedback).

In this paper, we assess the functional requirements of a context-sensitive dictionary as a start; then we explain the program's three phases of operation, focusing on the implementation of the lexicons and the context-sensitive features. We conclude the paper by comparing our tool to other similar publicly available products, and summarize plans for future development.

1 Introduction

With several instant comprehension tools publicly available, we need not justify the usefulness of the type of device we are developing. The main idea behind the program is to help computer users understand the large number of foreign language texts they encounter. In most situations of computer usage, users do not need translations, nor do they have to *provide* translations. A dictionary in such cases must not be another application but a background process providing help when necessary.

This help must be context-sensitive in two aspects: first, it should appear in the context where the need for translation occurred, the user must not be forced to switch to another context of a separate application; second, the output—the translation—should contain only information relevant to the textual context for which the translation is required. An entire dictionary entry should almost never be displayed since it contains multiword examples irrelevant to the context of translation. Adapting a bi-lingual dictionary to foreign language comprehension takes the recompilation of any dictionary to some extent before it is incorporated in the system (Feldweg and Breidt 1996).

We define the context-sensitive electronic dictionary we devise here as a *context-sensitive instant comprehension tool*. It is more than a dictionary lookup engine as it tailors dictionary entries to the context of the translation point. It is less than a translation engine, however, as it performs no syntactic processing of the source text, only series of dictionary lookups.

It is not only the textual context that our tool is sensitive to—like all major instant dictionaries: in a graphical computing environment, it reads text from anywhere on the computer screen, performs its linguistic analysis in the background,

and then uses one or more dictionaries to find the translations. The output is displayed in a bubble, in front of the existing screen contents, leaving it otherwise intact. The program is activated without a mouse click, simply by leaving the mouse pointer over the translation point for one second. There are several aspects of user interface design affecting the decision to use this mechanism. The obvious advantage of using no mouse clicks is that this never interferes with the extisting user interfaces of any other programs.

2 Requirements of a comprehension assistant

An instant comprehension assistant is completely left alone in the sense that it cannot ask for user interaction: it cannot require the user to choose from a list of ambiguous linguistic analyses, and, at the same time, it should keep the proportion of semantic ambiguities as low as possible. So such an application can only rely on its own linguistic knowledge.

When the user leaves the mouse pointer over a word, it means that he needs information about that word and its context. The boundaries of the context are not precisely specified: it could be the entire sentence (or even a larger passage) which includes the selected word, or—more often—a smaller context such as a multi-word expression around it. It is therefore the task of the program to determine the largest possible context, analyze it, and provide as much information of it as possible—based on the dictionaries behind the system. The minimum requirement is that the program should recognize all obvious multi-word expressions and idioms, and provide appropriate translations. All possible forms of the multi-word expressions should be identified—even if word-forms are inflected or the word order is different from the basic form. This is the matter of the quality of the linguistic parsing components and the dictionaries. If no multi-word expressions are recognized in the context, the comprehension assistant should display a simple dictionary entry for the selected word only, listing all possible translations found in the active dictionaries.

There is another implication of the fact that the comprehension assistant is not allowed to ask for user interaction. The program has to acquire pieces of text from the screen regardless of the application that displayed them without relying on user input, clipboard or file contents, or special application properties. As there is no direct access to the text, the program sees pieces of text as sequences of characters without formatting or other document-specific information, including the language of the source text. This requires implementing a *language identification* algorithm, too. So far, it is clear that a well-behaved comprehension assistant is a rather special combination of different techniques, involving language technology in almost every bit of operation.

3 Phases of context-sensitive instant comprehension

Phase 1: Text acquisition. When the user leaves the mouse pointer unmoved for one second, the text acquisition phase is activated. This is a task all instant dictionaries must face. Current implementations rely on operating system (or graphical user interface) resources to acquire text displayed on the screen. Our implementation performs a combination of an OCR-like procedure on the screen contents and application-specific acquisition procedures. The former works with all applications, but is less accurate with nonstandard character sizes, while the latter communicates with known programs—this is very accurate, but limited to a closed set of programs. Depending on the version, text is acquired either one line or one paragraph at a time (when applicable).

Phase 2: Linguistic analysis and dictionary lookups. Linguistic analysis is used to identify the word that was pointed at, and perform a morpho-syntactic analysis of its context to determine what to look up in the dictionaries. Linguistic analysis consists of several steps essential for proper dictionary lookup, because there is no initial information about the text other than the text itself—with a single word highlighted indicating the position of the mouse pointer and thus the initial point of analysis. One must take into account that the initial data are often results of an OCR-like process whose errors require correction during subsequent linguistic analy-

sis—similarly to the procedure in common OCR programs.[1]

The linguistic analyzer module performs morpho-syntactic analysis for the selected word in context—by means of the HUMOR engine (Prószéky and Kis 1999). At this point, morphological analysis has three main purposes: (a) linguistic stemming for accurate dictionary lookups, (b) spelling correction and (c) preparation of shallow parsing of the context to identify candidates for multi-word expressions.

If linguistic analysis fails to recognize any multi-word expressions, words from the context are still passed on to the dictionary lookup phase as the dictionaries may contain idiomatic phrases that cannot be recognized on a linguistic basis.

The dictionary lookup module receives lexical stems in the context of the translation point, and matches them against the installed dictionaries. The program uses the same robust dictionary engine as the one we use in our terminology management system. It is capable of handling multiple dictionaries at the same time (Prószéky 1998).

Dictionaries are compiled to facilitate the filtering of multiword lexemes. This means two things: first, in addition to headwords, all lexemes (subheadwords, examples) within entries are indexed. Second, entries are split into smaller parts to retrieve only relevant information.

The engine is capable of finding all multi-word lexemes which include one or more words with a single lookup. In some cases, this could be a rather lengthy list which must be filtered using the other words in the context. More precisely, (translations for) multi-word expressions will be displayed if and only if they include some significant words of the context (and do not contain other significant words). By 'significant word', we mean that there are also 'nonsignificant' words (or stop-words) that are skipped when forming a query expression for the dictionary engine.

[1] According to our experience, however, recognition errors are very rare because there is a closed set of shapes (glyphs in the currently installed system fonts) that may occur in any text displayed by applications (except for pieces of text within bitmap images). Recognition errors are usually results of applications using nonstandard techniques (e.g. dynamically altering character spacing) to display text.

The ambiguity of the output is reduced 'only' by this filtering process. If an entry is considered as relevant by the filtering procedure, it is displayed. In current implementations, different *meanings* of a single word or a multiword lexeme are *not* filtered out based on the context.

Phase 3: Rendering and graphic output. The output of the program is displayed in a bubble-shaped pop-up window on the screen that disappears if the user moves the mouse cursor again. The bubble contains formatted text: current implementations use either a proprietary XML-to-RTF conversion procedure, or XSLT formatting, depending on the version.

4 Some implementation details

Dictionaries. Dictionaries in our system are represented as lexical databases where the structure of each dictionary is strictly preserved. This is achieved through using XML as the *single* dictionary format. Dictionaries are either originally written in XML or transformed from a printed or another electronic format by means of automatic and semi-automatic tools.

All dictionaries are bi-lingual. Currently available dictionaries use language pairs such as English-Hungarian, German-Hungarian. However, there are experimental dictionaries for other languages such as Spanish, Polish, and even Japanese.

The largest dictionary currently available is an adaptation of Hungary's newest academic English-Hungarian dictionary, which contains over 400,000 entries in the electronic version. (Note that for the reasons mentioned earlier, original entries are split into multiple parts for filtering multiword lexemes.)

We have mentioned earlier that a language identification module might be required for efficient operation of an instant dictionary. One could notice, however, that we have not implemented such a module. Although we have developed a language identifier called LangWitch, we use a much simpler approach in the instant comprehension tool: all dictionaries are looked up in both their languages. If a word is there in a dictionary in any language, there is a hit. Therefore, if there is a word on the screen that is included in any of the installed dictionaries in any

language handled by them, it will be recognized and translated.

Filtering. By using a heuristic procedure, the program is able to recognize continuous and discontinuous multiword lexemes. The size of the analysis window if configurable, but basically it is determined by the longest multiword example in the dictionary.

Text acquisition accuracy. Most versions of our instant comprehension assistant use the OCR-based text acquisition technique mentioned earlier. This procedure is capable of recognizing text written in fonts installed on the computer. If a piece of text is written in an installed font and in a standard size between 8 and 16 points, the recognition accuracy is near 100 percent. With nonstandard text sizes (zoomed display, too small or too large character spacing), however, the accuracy radically declines. Some applications—like Microsoft Word or Adobe Acrobat Reader—display text in a nonstandard way. For these applications, we use alternative acquisition methods that communicate with the particular application using an application-specific protocol, which provides accurate text recognition.

Processing user feedback. Our team does not regularly develop dictionary contents. Some dictionaries, however, have been developed by us, and these dictionaries are continuously reviewed and updated. The update process is rather unique because it is built largely on user feedback. From the aspect of dictionary development (and even linguistic research), the comprehension assistant is an ideal source of linguistic information because it reaches a potentially large number of users (since it is not a special application but a utility that has its place in every computing environment). Based on this insight, we have implemented an instant feedback feature, which comprises of two processes:

(1) Logging: the program continuously logs words and multiword expressions it was unable to analyze or failed to find in the dictionaries.

(2) Contacting the developers: the program automatically sends e-mails containing the current logs to the developer lab.[2]

[2] This requires permission from the user which the program asks for during installation.

Logs are gathered and analysed by further automatic tools at the development site. Having been filtered to exclude obvious noise entries, the list is then sent to lexicographers for further analysis. This process effectively reveals errors and deficiencies in the dictionaries and the morphological lexicons, and, at the same time, it helps defining directions of further improvements.

5 Comparison to other systems

There are two categories where our context-sensitive instant comprehension tool—the brand name is MoBiMouse—might be compared to other systems: functionality and linguistic accuracy. There are a few pop-up dictionaries on the market: the most well-known are Babylon, WordPoint, CleverLearn, iFinger, Langenscheidt's Pop-up Dictionary and Techocraft's RoboWord, but none of them have as many language technology features as MoBiMouse. There are some 'glossing' programs in research laboratories (RXCE, see Feldweg and Breidt 1996; or Sharp, see Poznanski et al, 1998) that access dictionaries with a context-sensitive look-up procedure. However, they present the information to the user through their own graphical interface, and none of them have the basic featuere of MoBiMouse, namely, being a context-sensitive instant comprehension tool for *any* running application. The above systems do not have access to more than one dictionary at the same time, unlike MoBiMouse. On the other hand, the treatment of multiword units in the IDAREX formalism (Segond and Breidt 1996) is more sophisticated than in MoBiMouse. Another project with instant understanding is GLOSSER, whose prototype (Nerbonne et al. 1997) performs morphological analysis of the sentence containing the selected word in a similar manner. In GLOSSER—unlike in MoBiMouse—there is a stochastic disambiguation step but everything is shown in a.separate window.

The text acquisition techniques used in MoBiMouse are independent from both the language and the writing system. Hence it is rather different from most known applications that work with English characters only. Most other pop-up dictionary applications start by pressing a button or clicking the mouse. MoBiMouse is activated without mouse clicks (like RoboWord), therefore

it can be used to acquire any text from the screen without affecting other running applications. MoBiMouse is even able to access user interface elements such as menus and buttons because it works from the graphical content of the entire screen, while others such as RoboWord access only the window contents displayed by applications.

The speed of the text acquisition module is 1000 character/s, stemming takes 0,002 s/word-form, an average dictionary lookup 0,02 s. MoBiMouse, unlike Babylon, can be used in both language directions of the dictionary due to its writing independence and liguistic components for many languages.

6 Future development plans

Most our development plans focus on improving the program's user interface. Before MoBiMouse, we have developed an electronic dictionary/terminology management program called MoBiDic. The new versions of both programs are integrated into a single package, where the full MoBiDic user interface is callable through the MoBiMouse technology..

As for the linguistic capabilities, we plan to exploit MoBiMouse's open architecture and integrate the 'traditional' dictionary lookup module with a parser/translator engine capable of analysing and often translating an entire sentence or at least a part of it. The parser/translator engine (called MetaMorpho) is still under development.

Conclusion

MoBiMouse is a context-sensitive instant comprehension tool that offers translations for words and expressions displayed on computer screens. The program is activated without a mouse click when the user leaves the mouse pointer over the word in question. The translation is displayed in a tooltip-like bubble. If the mouse is moved again, the translation disappears promptly, so the user's work will not be disrupted by another program requiring a whole window.

Although there are many similar programs publicly available, we believe MoBiMouse is quite unique thanks to many of its features: (1) the combined text acquisition procedure (using an application-independent and an application-specific module), which makes it work in any application, (2) the rich linguistic processing with the linguistic stemming and the context-sensitive filtering module, which makes the program the most linguisticly sophisticated of its kind, (3) and the open architecture which makes any major architectural element replaceable, providing for an easy development of any kind of instant information acquisition application.

Acknowledgements

The authors would like to thank András Földes, development project leader for MoBiMouse, and László Tihanyi, chief content developer at MorphoLogic.

References

Feldweg, H. and E. Breidt (1996) COMPASS—An Intelligent Dictionary System for Reading Text in a Foreign Language. *Papers in Computational Lexicography (COMPLEX 96),* Linguistics Institute, HAS, Budapest, pp. 53–62.

Nerbonne, L. Karttunen, E. Paskaleva, G. Pröszéky and T. Roosmaa (1997) Reading More into Foreign Languages. *Proceedings of the 5th Conference on Applied Natural Language Processing (ANLP–97),* Washington, pp. 135–138.

Poznanski, V., P. Whitelock, J. Udens and S. Corley (1998) Practical Glossing by Prioritised Tiling. *Proceedings of the COLING-98,* Montreal, pp. 1060–1066.

Prószéky, G. (1998) An Intelligent Multi-Dictionary Environment. *Proceedings of the COLING-98,* Montreal, pp. 1067–1071.

Prószéky, G. and B. Kis (1999) A Unification-based Approach to Morpho-Syntactic Parsing of Agglutinative and Other (Hughly) Inflectional Languages. *Proceedings of the37th Annual Meeting of ACL,* College Park, pp. 261–268.

Segond, F. and E. Breidt (1996) IDAREX: description formelle des expression à mots multiples en français et en allemand. In: A. Clas, Ph. Thoiron and H. Béjoint (eds.) *Lexicomatique et dictionnairiques,* Montreal, Aupelf-Uref