The Defi Matcher

Archibal MICHIELS

Introduction

This paper provides a broad description of the workings of the DEFI matcher. The full Prolog program embodying the mwu (multi-word unit) matcher is to be found in "The DEFI Matcher: Prolog program".[1]

The DEFI project consists in developing an NLP tool which would act as a filter between dictionary and user, rejecting the non-relevant translations and ranking the ones it retains, on the basis of the information it can retrieve from the textual environment of the word the user has requested the translation of. The information the system is to make use of ranges from POS (which is nothing more than a convenient encapsulation of key distributional properties) to SUBJECT LABEL information and COLLOCATIONS (where we are looking at words as *thesauric heads* rather than as lemmas or wordforms—hence the need for thesauric resources such as WordNet).

The dictionary/text matcher is at the heart of the DEFI project (as a matter of fact the remainder of the project boils down to preparing data for the matcher). This central component is responsible for attempting to match the selected word in the textual chunk as

– part of a terminological item,

[1] http://engdep1.philo.ulg.ac.be/michiels/pmatch.htm.

✉ A. Michiels, English Dpt, University of Liège, 3, pl. Cockerill, B-4000 Liège, Belgium
amichiels@ulg.ac.be
http://engdep1.philo.ulg.ac.be/michiels

– part of a general-language multi-word unit,
– or as an individual lexical item for which it tries to choose the most appropriate translation(s).

This three-pronged approach is meant to cash in on the densest/longest match first principle on the one hand, and on the other on the fact that a specialised terminology has to use—to a large extent—the building blocks provided by the general language.

The selected word is first regarded as a member of a terminological item included in the relevant specialised data base (in our test bed, the db covering the field of Mycenaean archaeology). If the match succeeds with a weight that reaches a certain threshold, the terminological item with its translation is returned to the user and the matching process stops.

The second phase attempts to match the selected word as an element of a multi-word unit belonging to the general language and entered in the Prolog data base corresponding to the multi-word entries in our dictionary (this db includes all the examples the dictionary contains). The direction of the match is from dictionary to text. On the basis of the lemma of the user-selected word, candidate mwu's are retrieved from the Prolog data base and each of their elements is matched against the textual chunk. A global weight is assigned to the match, and the heaviest matches are regarded as providing the best fit. Again, a threshold needs to be reached for the match to be regarded as successful.

The third stage consists in matching the selected item in the textual chunk (both as wordform and as lemma) as against its entry or entries in the dictionary, trying to select the right target translation on the basis of the item's environment.

In this paper we concentrate on step two. The matcher for terminological units is still under development, but is not expected to present problems that the mwu matcher does not encounter. If anything, it is likely to be simpler in design and less fraught with difficulties in operation.

On the other hand, the matcher for single word lexical items, although simpler in design than the mwu matcher (of which it can be regarded as the limiting case), has a number of specific aspects that we will touch on briefly.

What the matcher works on and what it works with

These two issues are dealt with in other DEFI papers available on our Web site (see "From SGML tape to Dic clauses"[1] and "From the Oxford Hachette SGML tape to DEFI dictionaries"[2]), and we can only sum up here.

What it works on

The user's text is divided into textual chunks according to major punctuation. The chunks are submitted to the LingSoft *engcg* surface parser, whose results are reformatted and enhanced by an awk program, *tagtext*. Tagtext works out the voice and polarity of the textual chunk, to enable the matcher to carry out clausal controls, discussed below. It produces a list of Nps in the textual chunk (used *inter alia* in the matching of phrasal and prepositional verbs) and a list of syntactic relations holding in the textual chunk (such a list is necessary for the treatment of collocates, as the latter are assigned to syntactic positions of the arg bearer). Tagtext assigns weights to morphological and syntactic features set by the parser, to enable the matcher to assess the quality of the match between dictionary and text with a sufficient degree of delicacy.

The textual chunk file is read in at run time, the user being prompted for the filename. Examples of textual clauses are given in "The DEFI Matcher: Preliminary Results".[3]

What it works with

The dictionary (a merge of OH and RC, the basis for the merge being the identity of the lemma-translation pair) is divided into two parts:

[1] http://engdep1.philo.ulg.ac.be/michiels/sgml2dic.htm.

[2] http://engdep1.philo.ulg.ac.be/michiels/oh2defi.htm.

[3] http://engdep1.philo.ulg.ac.be/michiels/result.htm.

mwu's on the one hand (including all examples) and single-word units on the other.

The mwu's and examples are submitted to the *engcg* parser. As in the case of the textual chunks, the parser's results are reformatted and enhanced. In addition to the enhancements described above, a structural hypothesis is worked out for each mwu, giving the nature of the phrase that the mwu functions as (*np*, *vp*, *pp*, whole *s*, ...). Such a structural hypothesis is used by the matcher, viz. to set the global weight on the basis of the partial weights computed on the various elements of the mwu.

The single word lexical items are stored in a binary tree which is part of the internal Prolog data base (.idb) that the matcher restores on loading and executing. The retrieval key is the lemma.

The mwu's give rise to two binary trees, the first one (*hl*) acting as a tree of keys for the second (*dico*). The *hl* tree stores all the mwu's that a given lexical item is part of. By going from the user-selected word to the relevant *hl* keys, and from these keys to the relevant mwu's in the *dico* tree, we ensure that none of the candidate mwu's is neglected. A similar procedure will be used to store and retrieve the terminological items belonging to our test bed domain.

The matching process also makes use of some of the *WordNet* Prolog data bases to compute the distance from a dictionary-specified collocate to the item we have in text in the relevant syntactic slot. For the same purpose we also use data bases (*colloc*) made up of all the metalinguistic information contained in both OH and RC, exploring the hypothesis put forward by MONTEMAGNI *et al.*: 1996. On the use of both the *WordNet* and the *colloc* dbs see "Target Selection and Word Sense Discrimination in DEFI".[1]

What the matcher produces

For mwu matches, the output is a list of pairs *Weight-Structure*, in increasing order of *Weight*, and therefore of relevance. The *Structure*

[1] http://engdep1.philo.ulg.ac.be/michiels/wdts.htm.

records the dictionary *Idnum* (for debugging purposes), the dictionary lemma, the selected translation, the word the user selected and the whole textual chunk. The results are both displayed on screen and stored in a file bearing a *.lst* extension. Preliminary results are to be found in "The DEFI Matcher: Preliminary Results".[1]

The mwu matcher

For multi-word units we believe a very precise grammar of mwu's (recording the exact amount of syntactic manipulation and lexical variation that they admit of while retaining their interpretation as mwu's) may not be the way to go despite various claims to the contrary. We argue for such a position on the following grounds:

- if the grammar is delicate enough, how do we make sure that the information it needs is available on the textual side? In DEFI we have opted for a robust surface parser (LingSoft's *engcg*) to apply to both dictionary and text. It is important that the same parser be applied, not just a similar one in terms of delicacy—idiosyncrasies, even some errors, do not matter too much if they occur on both sides (dictionary and text) under similar conditions;
- the match is not an *either/or* decision, but a cline—therefore heuristics need to be admitted by the back door even if a very delicate grammar is relied on;
- a very delicate grammar is bound to be hard to agree on, costly, and hard to apply without error; data are likely to be hard to come by. It may very well be that such a grammatical description is not to be found in dictionaries simply because it is not specifiable. The borders of admissible syntactic manipulation and lexical variation are not hard and fast. Lexicographers may have proved wise in deciding to illustrate (in examples), rather than specify (in rules), such variation.
- we are not aiming DEFI at linguist-produced counterexamples, but at real text—real text is not supposed to explore the limits of a

[1] http://engdep1.philo.ulg.ac.be/michiels/result.htm.

system which, like DEFI, errs on the side of caution in admitting textual chunks as realisations of dictionary-recorded mwu's;

- even if the grammar tells us the textual chunk we are confronted with can't be an exponent of the phrase we are considering, it may be useful to present that phrase to the user, because there is probably some kind of allusion that the reader ought to be aware of. If *till the cows come home* is manipulated to yield **till the cow comes home**, this nonce phrase to be applied to a not very nice lady, we still need to hear an echo of the idiom, even if the literal meaning of *come home* is appropriate in the nonce phrase, combined with the metaphorical reading of *cow* as *woman*;
- two references are worth pondering over here:

1. John Sinclair in SINCLAIR: 1987, Chapter 8, *The Nature of the Evidence*, shows how important it is to look at words not only as lemmas, but as morphological variants, wordforms whose distribution cannot be predicted on the basis of general properties associated with the lemma. In DEFI we certainly should not look for matches restricted to lemmatised forms. In our matcher, the nearer we get to the very textual form the mwu is given in the dictionary (remember that mwu's include examples), the higher marks we give. And we go further than a wordform match: we also take into account the syntactic features associated with the wordform on the basis of the context it occurs in, since we collect marks for identity of both morphological and syntactic features.
2. Sue Atkins in ATKINS: 1994 shows that the amount of syntactic manipulation and lexical variation that idiomatic phrases admit of is much higher than one would think at first blush, because this is one of the areas where linguistic creativity is rife. We need to be very flexible if what we are interested in is to determine whether the phrase is still felt as recognisable as such, and relevant to a good understanding or appreciation of the text it is embedded in (and I would argue that that is precisely what the DEFI user is interested in).

We believe a lot depends on the quality of the heuristics which are used to compute the degree of quality of the match; this is an area for time-consuming investigation, but we cannot think of any useful shortcut.

We treat examples as mwus; we believe it would be a pity not to use them—they store information that the lexicographer was not able to store in more formalised information slots, and besides the border between phrase and example is not a hard and fast one. We use the same matching strategy, but assign different weighting to the global match.

The mwu matching algorithm opens with what could be called a pre-processing stage. The lemma of the user-selected word provides a key to the *hl* binary tree, and we retrieve all the handles for the candidate mwu's. The pre-processing stage consists in a first rough evaluation of the chances the candidates possess of providing a successful match when confronted with the textual chunk. We compute the intersection of the handle list (storing the lemmas in the mwu, discounting tool words) and the lemma list for the textual chunk, to work out whether the shared word list is sufficient in length; the evaluation takes into account the number of words in the handle list, and the nature, phrasal or sentential, of the mwu.

At this pre-processing stage, we also operate clausal controls on polarity and voice. The rationale here is that lexicographers give priority to canonical clausal format (positive polarity, active voice) unless they have good reasons to deviate from that format (passive much more frequent than active—*we've been had*—, inherent negativity—*no earthly reason*—). Polarity and voice need to be computed on the basis of the whole mwu on the one hand and on the whole textual clause on the other. The pre-processing referred to here checks that the values are compatible: an unmarked value is compatible with a marked one, but the marked value (neg, pass) is only compatible with itself. This compatibility test is probably too severe for voice (we could change it into a weight penalty) but appropriate for polarity.

Once the pre-processing stage has eliminated the mwu candidates that couldn't have made it, the real matching process begins. The directionality of the match needs to be emphasised again: from mwu to text. The matching process progresses in the mwu, attempting to match each of its elements as against a word or several words in the textual chunk. Each element of the mwu needs to be matched (this does not mean that it needs to appear as the same wordform in the textual chunk), but the progression in the textual chunk allows elements to be jumped.

The jumpable elements can be specified in terms of POS (for words) or constituency (for phrases). For instance, adverbs and prepositional phrases are jumpable. Note that the negation is jumpable too: its contribution to the match is through the computation of clausal polarity and the pre-processing compatibility checks.

The match of a given mwu element with the textual chunk is either **one word—one word** (standard case), **two words—one word** (*do so*, *doing so* on the dictionary side matched against any infinitive or gerund on the textual chunk side), or **one word—a phrase** (for placeholders such as *someone* or *something* on the dictionary side we look for a whole np on the textual chunk side).

Let us take the simplest case, that of a oneword-oneword match. The mwu element can be matched in four ways, in order of decreasing match weight:

– wordform match and lemma match (here the type of lemma, full word vs. tool word, bears on the weight assigned to the partial match);
– lemma only;
– depleted lemma (i.e. semantically depleted lemmas; intransitive and transitive classes; vi: *be*, *become*, *come*, *get*, ...; vtr: *do*, *get*, *give*, *have*, ...);
– placeholder (*one's*, *someone*, ...).

Besides, for each word match, in case of feature match we collect the marks attributed to the morphological and syntactic features assigned by the *engcg* parser (the weight of each feature is assigned by *tagtxt* and *tagdic*, the two awk programs that work on the output of the *engcg* parser). The rationale for collecting the weights assigned to the matched features is again that the lexicographer selects and edits the examples so that they display a high degree of typicality, and extracts and canonises phrases in a similar fashion. The nearer the text is to the example or phrase, from all possible points of view, the heavier the weight.

Once we have collected all partial weights (i.e. for each element of the mwu), we add them up and compute the global weight on the basis of this accumulated weight and the nature of the mwu. If the mwu is sentential (in most cases this means that we have to do with an example), the global weight is less than the accumulated weight. For phrasal mwu's (true phrases, be they idioms or not) the global weight and the accumulated weight are identical.

The anchoring of the one-word unit or mwu in context

Once the mwu has been matched according to the process just described, or once the single-word unit has been matched as wordform and lemma, or lemma alone, a similar check is performed on the anchoring of the element in context. This check (except for POS) does not lead to failure, but affects the global weight assigned to the match.

For one-word units we start with a POS check. The POS assigned by the surface parser and the ones found in the dictionary entries do not build up identical sets, so that here too the match is not an all-or-none affair.

In measuring the quality of the insertion of the mwu or single-unit word in its context we again try to cash in on the densest match first principle. The tighter the links between the item and its environment, the higher marks the match gets.

We use specifications in the dictionary as to the right-handside environment, embodied in the *envir* structure. Such specifications often include a placeholder as in *by doing*: we match by requiring preposition *by* followed by a gerund, the whole group to be found to the right of the matched governor.

The matching of collocates is a much more delicate process. Collocate lists have three characteristics:

– they are lists: each element in the list needs to be considered, and the one providing the highest match needs to be regarded as winner;

– they are assigned to syntactic positions with respect to their governor; *extended* collocates with [*contract, leave, program, sentence*]: this applies only to configurations where *extended* modifies an element of the collocate list; similarly, we find collocate lists for the subject or object slots of verbs, etc.;
– they are not to be considered wordforms or lemmas, but thesauric heads; to come back to the example just quoted, we need not find the very word *contract* (although a thesauric head matches itself, of course); we can have any hyponym or synonym of *contract*; even antonyms and hypernyms cannot be disregarded.

In short, to match the collocate list against the text, we need to be able to recover the relevant syntactic positions; we need to be able to recover np heads inside nps; and we need to be able to measure the semantic distance separating two lexical items. The first two of these requirements are taken care of by the enhancements brought to the parser by *tagtxt* and *tagdic*; the third one necessitates access to a thesaurus (in our case WordNet) and a good deal of computational power if the WordNet walk is allowed to be fairly extensive. We have also tried to use the sharing of metalinguistic slot as a basis for measuring semantic distance (see "Target Selection and Word Sense Discrimination in DEFI").[1]

We also use the patterns associated in OH with phrasal and prepositional verbs; this matching also requires access to the np list. A lot of subcategorisation info regarding the source language is missing in bilingual dictionaries when compared with present-day, learner-oriented monolinguals such as CIDE or COBUILD. Hence the considerations for future work touched on in the next section.

The matching of single lexical items brings a number of specific problems. One of them concerns typographical case; we first try to match with the case preserved as found in the textual chunk; if no match is obtained, we try an all-lower-case match. Another problem is raised by those items that refer to an obligatory head: *abortion* in OH has an entry where it expects *law* as head, *abortion law* in fact making

[1] http://engdep1.philo.ulg.ac.be/michiels/wdts.htm.

up a compound entry. In such cases, we check that the lemma of the head is found in text directly to the right of the matched item.

There are quite a number of information slots in our dictionary that are not used at present: we have rewritten part of the metalinguistic indicators as semantic features, and they can certainly be made use of. We also plan to make extensive use of the labels, especially the subject field labels, to carry out consistency controls that extend beyond the range of the textual chunk, or, to put it differently, to prefer the readings that enhance such consistency.

Future work

Besides fiddling with the heuristics (we repeat that this is bound to be a time-consuming process) we will look for improvement by trying to hook the information provided by several monolingual dictionaries to the source side of our bilingual ones. It is obvious that there is no hope of a direct match here, since the divisions of the source language in a bilingual dictionary reflect the cutting up of the semantic space enforced by the target. However, linking a source element with its nearest equivalents in a monolingual would enable us to use some of the information that our bilingual dictionaries lack and that the monolinguals (at least for English) are good at (subcategorization info being an obvious example). Linking with LDOCE, COBUILD and CIDE (see HARLEY and GLENNON: 1996) is bound to improve source word sense discrimination.

References[1]

ATKINS (Sue): 1994 (ed), "COMPASS LRE 62-080, Adapting Bilingual Dictionaries for Online Comprehension Assistance, Deliverable 2", *Functional Specification for the Icomprehension Assistant Dictionary.*

[1] See also previous paper.

HARLEY (Andrew) and GLENNON (Dominic): 1996, *Sense Tagging in Action*, paper submitted to the ACL 1997 conference on *Tagging Text with Lexical Semantics: Why, What and How?*