MetaMorpho TM: a linguistically enriched translation memory

Gábor Hodász and **Gábor Pohl**
Pázmány Péter Catholic University
Department of Information Technology
Práter utca 50/a.
Budapest 1083, Hungary
{hodasz, pohl}@morphologic.hu

Abstract

This paper describes the new linguistic approaches of the MetaMorphoTM system, a linguistically enriched translation memory. Our aim was to develop an improved TM system that uses linguistic analysis in both source and destination languages to yield more exact matches to the source sentence. The MetaMorphoTM stores and retrieve sub-sentential segments and uses a linguistically based measure to determine similarity between two source-language segments, and attempts to assemble a sensible translation using translations of source-language chunks if the entire source segment was not found.

1 Introduction

Commercially available translation memories do not involve linguistic knowledge. Instead, they determine similarity between translation units on the basis of pure mathematical distance calculations, with the majority of the algorithms using fuzzy indexes. Although most such systems recognize and handle non-translatable passages (such as dates, numbers, some abbreviations), and incorporate terminology management modules as well, existing translation memory systems do not systematically treat and recognize morpho-syntactic similarities, if the translation units themselves are too different in terms of characters.

The other new approach of the system is the use of NPs and sentence skeletons (sub-sentential segments) in both source and destination languages. The translation is assembled from stored translations of noun phrases and the morpho-syntactic skeleton of the source unit. The morpho-syntactic skeleton is a sequence of lemmas and morpho-syntactic parses of the words in the source unit, with a symbolic NP slot at the place of each noun phrase. In order to store and retrieve skeletons and NPs, the system uses an automatic NP alignment method.

This scheme may result ambiguities – an undesired phenomenon in CAT systems –, but the best translation will be a far closer approximation of the desired one than in existing systems.

This paper presents a proposed system addressing these particular problems, one that attempts to assemble translations from sub-sentence units stored in its database, both in the source and the target languages.

Section 2 provides details on the linguistic matching of source and target segments with the mathematical definition of linguistic similarity.

Section 3 discusses the method of the NP alignment.

Section 4 provides implementation-specific details of the development project.

Section 5 concludes the paper by describing a proposed evaluation scheme for the translation memory, and provides some early result compared with an “ordinary” TM system.

2 Similarity between linguistic patterns

In this section we give an overview of the linguistic based similarity lookup, including the expectations, and give a mathematical definition for a similarity measure that suits our purpose.

2.1 Similar studies

Though the theory of the example-based translation came out in the mid 80’s (Nagao, 1984), the consideration of linguistic similarity between segments is rather new. There are several practical applications however. In most cases the methods are based on selecting identical phrases in the memory using semantic distances. These methods use a thesaurus or onthology.

The deficiencies of these approaches are:

- they presume an additional particular linguistic knowledge (e.g. equivalence classes or semantic knowledge), which are language-dependent and can be ambiguous

- they never define a similarity measure therefore the ranking of the results are quite difficult
- they mostly search for entire sentences, whereas we need sub-sentential processing.

However even these methods are not used the leading commercial systems. They use the simple character-based similarity methods (called fuzzy matching). (Mandreoli, 2003)

2.2 Levels of similarity

The linguistic analyses of a segment can be considered as different levels of the words in the segment. In this approach an *m* long *S* segment is not only a sequence of words in surface form, but every word has *L* layers correspond to the *L* steps of analyses. If every word has only one result at every level (a disambiguated case), the *S* segment will contains *m* tokens at every level.

The base of our similarity measure is the well-known Levenshtein distance, which was further developed to handle the above described multi-level segments.

In our MetaMorphoTM system 3 levels of similarity were defined:

- surface form of the word (L1)
- lemma of the word (L2)
- class of the word (L3)

Naturally these linguistic analyses often affect ambiguity at each level. There are two strategies to manage this: one is to use disambiguation algorithms to select one result at each level, the other is to define a similarity searching method that manage the different results.

Although we use these 3 similarity levels in MetaMorphoTM, it is possible to use others, even non-linguistic levels, e.g. formatting information.

2.3 The definition of the Multi-Level Linguistic Similarity

(1) Definition: *(Edit distance between sequences of terms)* Let S_1 and S_2 two segments represented by the following sequence of terms: $\sigma(S_2) = t_1^2 ... t_m^2$ and $\sigma(S_1) = t_1^1 ... t_n^1$

The edit distance *ed(σ(S₁),σ(S₂))* between $\sigma(S_1)$ and $\sigma(S_2)$ is the minimal number of operations (insertion, deletion or substitution) that transforms $\sigma(S_1)$ to $\sigma(S_2)$.

The above definition can be used at any level of similarity. Let $\phi(S)$ the form of *S* after an arbitrary step of analyses. Following the above $ed(\phi(S_1), \phi(S_2))$ can be calculated easily.

(2) Definition: *(Multilayer Similarity Measure between sequences of terms)* Let S_1 and S_2 two segments represented by the following sequence of terms:

$\sigma(S_1) = t_1^1 ... t_n^1$ and $\sigma(S_2) = t_1^2 ... t_m^2$

According to the *i* steps of analysis $\phi_1, ..., \phi_i$, S_1 and S_2 contains *i* layer of similarity layer: $L_1, ... L_i$. The multilayer similarity vector between $\sigma(S_1)$ and $\sigma(S_2)$:

$$ED(\sigma(S_1), \sigma(S_2)) =$$
$$[ed_{Li}(\sigma(S_{1,Li}), \sigma(S_{2,Li})), ed_{Li-1}(\sigma(S_{1,Li-1}), \sigma(S_{2,Li-1})), ...,$$
$$ed_{L1}(\sigma(S_{1,L1}), \sigma(S_{2,L1}))]$$

The above defined similarity vector is feasible for linguistic similarity matching between any kind of segments and it accomplishes the above described requirements.

For our purposes we use a normalized form: the edit distance divided by the length of the segment to be translated:

$$d(\sigma(S_1), \sigma(S_2)) = \frac{ed(\sigma(S_1), \sigma(S_2))}{|\sigma(S_1)|}$$

therefore the normalized similarity vector is:

$$D(\sigma(S_1), \sigma(S_2)) = \left[\frac{ed_{Li}(\sigma(S_{1,Li}), \sigma(S_{2,Li}))}{|\sigma(S_{1,Li})|}, ..., \frac{ed_{L1}(\sigma(S_{1,L1}), \sigma(S_{2,L1}))}{|\sigma(S_{1,L1})|} \right]$$

2.4 Handling ambiguity

The calculation of the similarity vector is more complex, when the terms of the sequence have several results at the similarity levels (generally they have). In this case two strategies are possible: to use a disambiguition strategy at each level that designates a winner from the candidate segments (e.g. POS-tagger) or to compute multiple similarity vectors and choose the smallest one. To improve the latter method a simplification is advisable: if there is a result of the *i*th word of the segment at layer *l* that matches with one of the results of the candidate segment *i*th word at layer *l*, we can accept it as a concord.

In the MetaMorpho TM a POS-tagger is applied to handle ambiguity, although in the future the latter method will be implemented and tested.

3 NP alignment

To be able to look up NP (and sentence skeleton) translations in the translation memory, NPs of the stored sentence pair have to be aligned either by the translator or by an automatic means. Leaving the tedious task of NP alignment to the translator would decrease productivity hence an automatic means of NP alignment was developed for our TM.

Previous and related works include corpus-based statistical phrase alignment methods, and parse tree alignment techniques for EBMT systems. Recent advances in tree alignment (e.g. Groves, 2004) are promising but English and Hungarian parsers are too different for such methods depending on the internal structure of NP trees. Corpus-based statistical methods, like the one developed by Kupiec (1993), are more robust, but they require reprocessing of the whole translation memory after a new sentence pair is stored. In a TM product a new sentence pair should be stored in less than one second, so in our online NP aligner we substituted the time consuming statistical data collection for dictionary usage.

In our first experiment NPs of the stored sentence pair were extracted by the MetaMorpho English and Hungarian parser. The Hungarian MetaMorpho grammar is in its early development stage, so later on we developed another means of Hungarian NP extraction, which aims to find Hungarian counterparts of English NPs without using a Hungarian parser.

3.1.1 Dictionary-based NP alignment

Our heuristic NP aligner algorithm calculates a matching score for all possible English-Hungarian NP pairs, extracts the best-matching pairs, and applies a threshold to filter ambiguous alignment links. The matching score is based on tokenized dictionary matching, cognate matching and POS matching. The alignment method allows insertion or deletion of function words (prepositions, pronouns, determinants).

First dictionary matching is done then cognates are searched among the unmatched words. At last POS matching is calculated among the previously unmatched words. If any function word remains unmatched it is discarded with a small penalty in the matching score. Dictionary entries matched in an English NP and in the Hungarian sentence but not in the examined Hungarian NP also imply some penalty. The Matching Score is calculated by the following formula:

$$MS = \frac{a \cdot DMW - b \cdot DNMW + c \cdot CMW + d \cdot PMW - e \cdot RFW}{W - RFW}$$

where DMW is the number of dictionary matched words, DNMW is the number of words found in some other part of the translation sentence but not in the translation NP, CMW is the number of matched cognate words (not matched before), PMW is the number of words (not matched before) with matching POS tags, RFW is the number of remaining function words, and W is the number of words; all values calculated considering both the English and the Hungarian NPs (or sentence, in case of DNMW). The constant coefficients (a=1, b=0.9, c=0.5, d=0.25, e=0.1) are guessed values, later on they are going to be trained on an NP-aligned parallel corpus.

3.1.2 Dictionary matching

All possible stems of words in an English NP are searched in the dictionary using a stem index. The dictionary index also contains references to phrases (or multi-word lexemes). A maximum of 200 dictionary entries are searched for each NP stem to reduce search field (without using a stop-word list). The resulting dictionary entries are filtered by the criteria that they should only contain source side stems found in the English NP and target side stems found in the Hungarian sentence. The words of the English NPs are matched to dictionary entries; each NP position should belong to only one dictionary entry. Longer matches are preferred so matching is started from the longest dictionary entries. This way if "*hard disk drive*" is in an English NP and in the dictionary too, the shorter matching dictionary entries, "*hard disk*", "*disk drive*", "*hard*", "*disk*", "*drive*" are not matched to words in the NP.

A dictionary entry found in an English NP is considered to be found in a Hungarian NP if at least one stem of all tokens of the entry can be matched to an unmatched token of the Hungarian NP.

3.1.3 Cognate matching

Cognates (Simard 1992) are searched among the words of the English and Hungarian NPs in order to find named entities that are not in the dictionary. In our implementation two words are cognates if they are both longer than one character; both contain at least one capital letter, number, or other special character; and they are exactly the same word or at least their first 4 characters are the same.

3.1.4 POS matching

By matching the part of speech tags of words not found in the dictionary the recall of the aligner can be increased. In our experiment we used only basic POS categories like noun, verb, adjective, preposition, determinant, pronoun, etc. The POS matching score should be much lower than the dictionary or cognate matching score, otherwise alignment precision would be low.

3.1.5 Unmatched function words

Elimination of unmatched function words is important because there are differences in grammar words of English and Hungarian. In Hungarian different cases are used instead of prepositions of English, and English possessive pronouns correspond to a determinant and some case marking on the possessed Hungarian noun. Handling unmatched words without any distinction would result in lower recall, e.g. the following NP pair could not be aligned if we had given the same penalty for all unmatched words:

- my book [PRON N]
- a könyvem (“the book-my”) [DET N]

3.2 Hungarian NP extraction guided by English NPs

At the time of our experiment the Hungarian grammar developed for the MetaMorpho parser was only partially implemented ad had low recall and precision. Therefore we developed a simple heuristic means of extracting Hungarian NP candidates by mapping the words of the English NPs to the words of the Hungarian sentence. Each word of an English NP is mapped to all possible word positions in the Hungarian sentence using stemmed dictionary matching and cognate matching. The word mappings producing the smallest number of words between the mapped Hungarian words are selected. After selecting the matched words, the Hungarian NP may be expanded to the left (preferably) or to the right if the English NP contains unmatched words, and there is no unmatched word with matching POS category in between the matched Hungarian words. Determinants on the left side of the NP are considered part of the Hungarian NP even if they had no counterpart in the English NP.

We calculate English-Hungarian NP matching scores the same way we do with parsed Hungarian NPs.

4 Implementation

In this section we discuss some of the important implementation details.

The underlying database is a relational database, storing whole sentences and noun phrases in each supported language (English and Hungarian at present). The linguistic similarity calculation and the NP alignment implemented in C++ and the system has a fully functioning graphical user interface, designed for professional translators. So far, these modules have been implemented.

Success of the language-aware TM depends on the quality of the NLP modules implemented for both the source and the target language. The proposed TM scheme is also designed for robustness: should the linguistics-based process fail, it can fall back to a ‘traditional’ TM procedure. To this end, a traditional fuzzy index will also be created. The fall-back protocol is the following:

(1) Attempt exact match;

(2) Try to assemble a composite translation using the TM database and the grammar of the MT engine;

(3) Attempt fuzzy match (with a rather high threshold);

(4) Attempt fully automatic translation.

If any of the above steps is successful, the translation procedure is finished. It is very important to note that human-confirmed patterns always take precedence over those in the grammar of the MT engine.

5 Evaluation

The proposed approach attempts at providing significantly higher quality in a translation memory than achieved by commercially available products. Evaluation must thus provide evidence for the hypothesis that linguistic annotation and sub-sentential alignment and retrieval can provide quality improvement.

The recall and the precision of the translation memory itself must be measured. It presents a challenge as there is little information available on measurements of existing translation memories.

The recall of a translation memory is a vague concept: when compared to the source text, it always depends on the size and contents of the TM database, which in turn depends on the individual user. It is more useful to compare the hits to the contents of the TM database: the recall of a TM system can be measured by assessing how many of the stored translation units have a chance greater than 50% being retrieved, when testing it on a sufficiently large corpus.

If we are measuring recall this way, there is an argument in favour of the language-aware TM engine: by common sense, the shorter the source segment, the more probable it is to be eventually found. It is obvious that the language-aware translation memory stores shorter segments: on the one hand, it stores substrings of the input segment, on the other hand, it stores simplified patterns such as sentence skeleton where the full variability of NPs is collapsed into a single NP gap.

The precision of a translation memory can be measured by the time the user has to spend with correcting translations offered by the system. There were no end user tests conducted as of the time of writing, however, we can again provide an argument. It is inherent to the language-aware TM to provide combined translations where the translations offered are adapted to the source segment instead of offering an entire target segment unchanged.

The process of evaluating the proposed system has only begun. It is being tested on a sample

English-Hungarian parallel corpus of texts on computing, with a size of approx. 1.2 million words per language, 2.5 million words altogether. As an example we present an early result to demonstrate the difference between the traditional and our approach. We compare the answer of an "ordinary" TM System based on bi-gram similarity methods and full-sentence retrieval, and the suggestion of MetaMorphoTM assembled from stored translations of noun phrases and a sentence skeleton.

Sentence to be translated (from the corpus):
Microsoft Windows 2000 makes it possible to configure hard disk drives in a variety of ways.

The sentence suggested by the language-independent TM:
Install the Microsoft Windows 2000 Resource Kit.
In hungarian: *Installálja a Microsoft Windows 2000 Resource Kit-et!*

MetaMorphoTM:

[01: Microsoft Windows 2000] make it possible to configure [02: hard disk drives] in a variety of ways.

In hungarian: *[Microsoft Windows 2000] [sokféle módon] [lehetővé teszi a beállítását] [merevlemez][ek][nek].*

Although the sentence was not in the database, the MetaMorphoTM assembled it from a stored skeleton and NPs.
However, the main drawbacks of the approach are coming from the language-aware methods:

- the handle of the morphological ambiguity
- the differences between parsers for different languages

6 Conclusion

This paper presented a language-aware translation memory scheme with the detailed description of the basic algorithms and processes of the NP chunking and sub-sentential (NP) alignment, and the searching with a linguistic based similarity measure.
We state that the only way to achieve substantial improvement in translation memory quality is the integration of language-aware methods.

7 References

Julian Kupiec (1993): An Algorithm for finding Noun Phrase Correspondences in Bilingual Corpora. In: Proceedings of the 31st annual meeting on Association for Computational Linguistics, pp. 17-22, 1993

Groves, D.; Hearne, M.; Way, A. (2004): Robust Sub-Sentential Alignment of Phrase-Structure Trees. COLING '04, Geneva, Switzerland, 2004

Mandreoli, F., (2002): Martoglia R., Tiberio, P.: 'Searching Similar (Sub)Sentences for Example-Based Machine Translation', Atti del Decimo Convegno Nazionale su Sistemi Evoluti per Basi di Dati (SEBD 2002), Isola d'Elba, Italy.

Nagao, M. (1984): 'A framework of a mechanical translation between Japanese and English by analogy principle', In A. Elithorn and R. Banerji (eds.), Artificial and human intelligence, 173-180. Amsterdam: North-Holland.

Simard, M., Foster, G. & Isabelle, P. (1992): Using Cognates to Align Sentences in Bilingual Corpora. In: Proceedings of the Fourth International Conference on Theoretical and Methodological Issues in Machine translation, (TMI92), Montreal, pp. 67-81, 1992