SoSe 2006 Projekt „Maschinelle Übersetzung“ Teil II

Example-Based Machine Translation by mean of Pattern Extraction

Cristina Vertan, Walther v. Hahn
{vertan,vhahn}@informatik.uni-hamburg.de

Approach in „Machine Translation -part I“

- Translation Database was composed of some hundred sentences in SL and TL
- The input in SL was compared with the SL part of the entries in the database and a best candidate translation is extracted according to metrics like.
  - Edit distance
  - Angle of similarity
  - Semantic distance
- The longest common sequence between the input and the best candidate entry is computed. If this sequence is smaller than the input then the procedure is repeated with the rest part of input.
- For the retrieved SL fragments their translation equivalents are retrieved, using the marking hypothesis.
- The fragments in the TL are recombined using some heuristics for grammar and morphology.

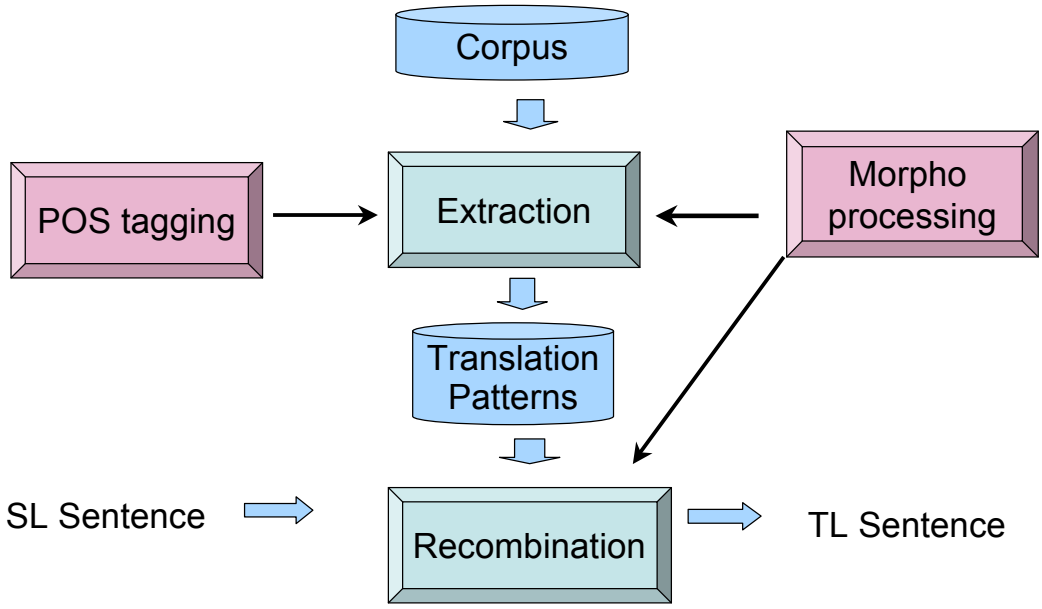
Conclusions of „Machine Translation - part I“

- Comparison between input and SL entries in the database was done only at string level (word or lemma)
- Metrics are quite sensible to :
  - Length difference between input and database entry (ED and partially also the semantic distance)
  - Difference in the word order in the input and database entry (semantic distance)
- The size of lexical resources influences strong the quality of results
- The retrieval of translation equivalents is possible almost only using the marking hypothesis. This require strong manual intervention
- Recombination is also strong dependent of the lexicon size

Approach in „Machine Translation -part II“

- Based on the observation that sequences of strings (also discontinous) appear in several entries in the database
- Identify patterns in the SL and TL and align them
- Retrieve the most similar patterns which matches the input.
- Recombine target patterns
- Advantage: good results also without:
  - big lexical ressources
  - A lot of morphological rules in the recombination phse

System architecture



Contents

Following slides present the approach described in

Kevin McTait „Translation Pattern Extraction and Recombination for Example -Based Machine Translation“, PH.D Thesis UMIST, 2001

- Definition of Translation Patterns
- Pattern Extraction
  - Monolingual Phase

19.04

  - Bilingual Phase
  - Alignment of Text Fragments and Variables
    - Sequence Comparison Algorithm
    - Bilingual Similarity Metric

10.05

- Recombination
  - Pattern retrieval

24.05

  - Core Recombination Method
  - Translation Confidence Score

14.06

Terminology

- „Recombination“ -phase subsums here :
  - the „matching pahse“ in the string based approach . Here this is called pattern retrieval
  - The recombination phase in the string based approach

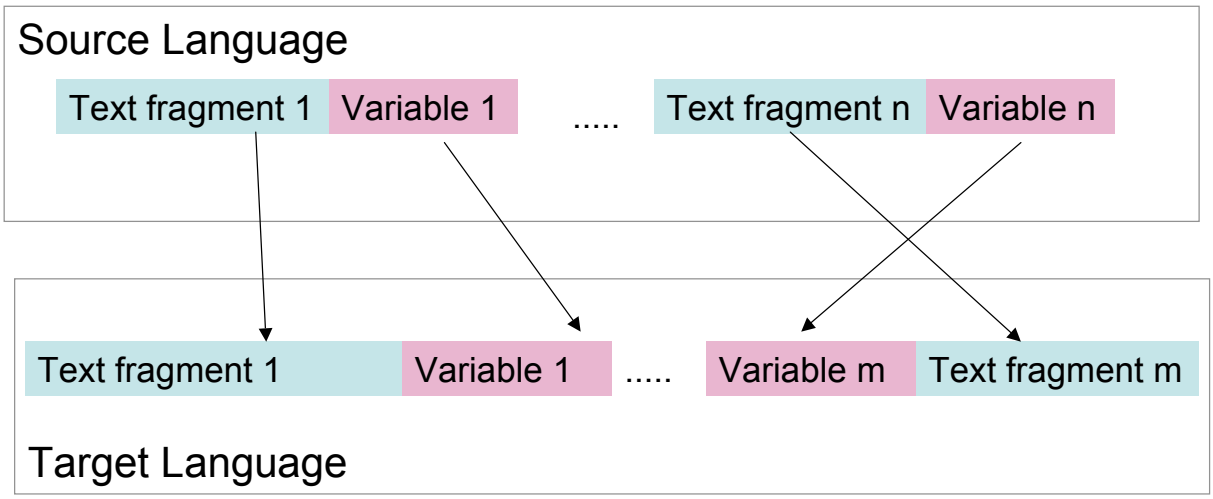
What are Translation Patterns

- **Translation patterns** – generalizations of sentences that are translations of each other in that sequences of one or more words are replaced by variables, possibly with alignments between the resulting word sequences and/or variables made explicit
- EBMT systems that extract translation patterns or templates from bilingual texts:
  - Kaji et al.(1992); Güvenir & Cicekli (1998), Brown (1999), Carl (1999), McTait &Trujilo (1999).

How to obtain Translation Patterns

- Extracted:
  - from a bilingual corpus aligned at the level of sentence
  - by language neutral recursive machine learning algorithm based on principle of similar distributions of strings: SL and TL strings that co-occur in two (or more) sentence pairs of a bilingual corpus are likely to be translations of each other
- Formed from lexical items that occur in a minimum only twice in the corpus. ADVANTAGE: the algorithm is useful even in case of sparse data

Alignement of patterns in SL and TL



Formal definition of a translation pattern -1-

- A tuple {S,T,A,Af,Av}, where:
  - S- sequence of SL subsentential text fragments, separated by SL variables;
  - T- sequence of TL subsentential text fragments, separated by TL variables;
  - Af-the global alignment of text fragments between S and T;
  - Av-the global alignment of variables;

Formal definition of a translation pattern -2-

- in S there can be any number p (p>0) of SL text fragments with p, p+1 or p-1 SL variables
- in T there can be any number q (q>0) of TL text fragments with q, q+1 or q-1 TL variables
- Possible configuration of S and T

$$F^S_{1,}V^S_{1,}F^S_{2,}V^S_{2}\ldots F^S_{p},V^S_{p}\_\_F^T_{1,}V^T_{1,}F^T_{2,}V^T_{2}\ldots F^T_{q},V^T_{q}$$

Pattern Storage

- Each text fragment F is represented as a pointer to the position in the corpus from which it is extracted, namely a data structure with the following fields:
  - pointer to the start position of F in the sentence from which it was retrieved;
  - pointer to the end position of F in the sentence from which it was retrieved;
  - pointer to the index of the translation example (sentence pair) containing F;
  - Pointer to the side of the corpus – SL or TL -containing F

Example of Storage as pointers

- Let F={4,8,43,0}
- It means:
  - text fragment F spans 5 words in length,
  - starts at position 4
  - terminates at position 8 in the SL sentence of
  - translation example 43 in the corpus.
  - The SL side of the corpus is denoted by 0 and the TL side by 1.

Translation Pattern Extraction

- Input - bilingual corpus aligned at the sentence level;
- Output - a set of translation patterns;
- Algorithm: language-neutral, operates on simple principles of string co-occurrence and frequency thresholds;
- divided into 3 stages:
  - monolingual
  - bilingual
  - alignment

Monolingual Phase

Phases

1. SL / TL Tokenisation;
2. SL / TL Word List Creation;
3. SL / TL Collocation Tree Formation

2. Word List Creation

- lexical items occurring in two or more SL and TL sentences are collected.

- e.g: From the sample corpus entries:
  - (1)The commission **gave** the plan **up** = La commission **abandonna** le plan;
  - (2)Our government **gave** all laws **up** = Notre government **abandonna** toutes les lois

  - Integers denote the sentences from which they were retrieved:

(gave)[1,2], (up) [1,2] and (abandonna)[1,2]

3. Collocation Tree Formation -1-

- Collocation – a data structure representing (possibly discontinuous) strings that co-occur in two or more sentences.
- Lexical items combine recursively to form a tree-like data structure of collocations
- Each lexical item is tested to see whether it can combine with any daughters of the root node, and if so, recursively with each subsequent daughter(and the daughters of that node and so on), as long as there is an intersection of at least two sentence IDs.

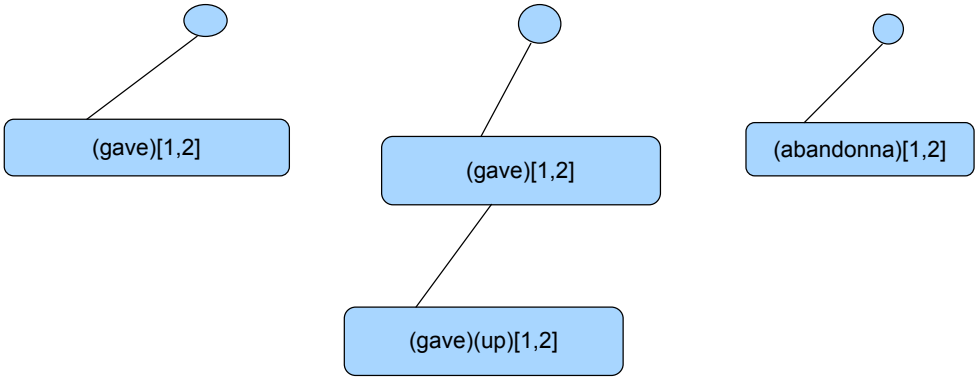
3. Collocation Tree Formation -2-

- The combination process is constrained only by:
  - the integer IDs of the sentences from which the lexical items were retrieved and the frequency threshold (a minimum of 2).
  - The intersection constraint enforces string co-occurrence in two or more sentences.
- If the node to be added cannot combine with any daughter of the root(or parent) node, it is added as a new daughter of the root (or parent) node.

Collocation Tree Formation (Result)

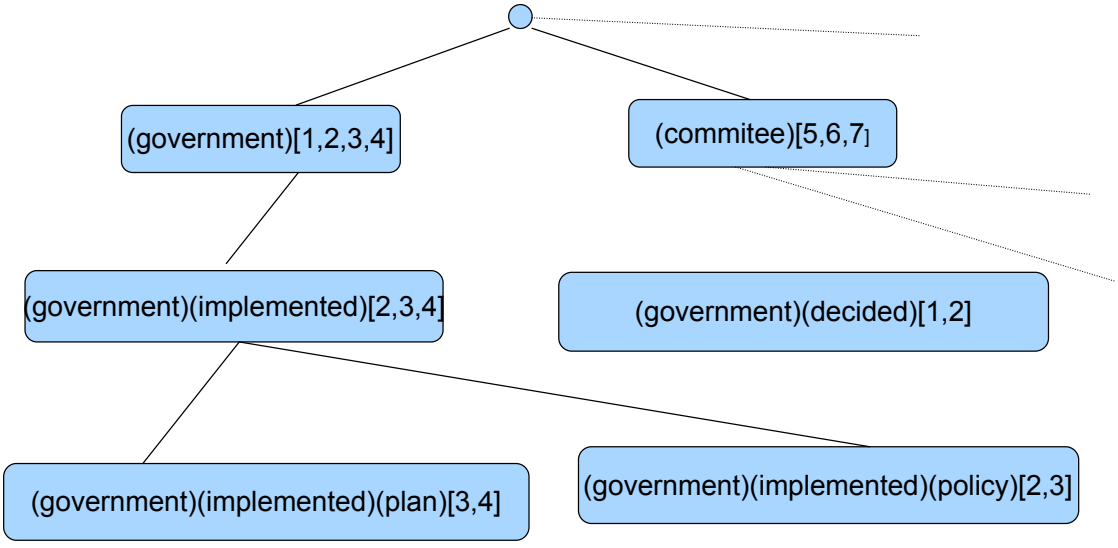
- Tree of collocations of increasing length, but decreasing frequency as the tree is descended from the root node to the leaves.
- The leaves become the most informative parts of the tree and are collected at the end of this phase.
- The leaf-collocations are filtered so, that only the longest are selected.
- Collocations that are subsumed by other collocations with the same sentence IDs are removed.

Collocation Tree (Example)



Complex collocation example

{(government)[1,2,3,4], (implemented)[2,3,4], (plan)[3,4], (policy)[2,3], (decided)[1,2], (committee)[5,6,7]},



Next Task - 10.05

–Bilingual Phase
–Alignment of Text Fragments and Variables