Architectures of „toy“ systems for teaching Machine Translation

Walther v. Hahn, Cristina Vertan

University of Hamburg, Faculty of CS, Natural Language Systems Dept.
vhahn, cri@nats.informatik.uni-hamburg.de

Abstract

This paper addresses the advantages of practical academic teaching of machine translation by implementations of „toy“ systems. This is the result of experience from several semesters with different types of courses and different categories of students. In addition to describing two possible architectures for such educational toy systems, we will also discuss how to overcome missconceptions about MT and the evaluation both of the achieved systems and the learning success.

1. Introduction

Machine Translation (MT) is an important sub-field both of Computational Linguistics and Natural Language Processing. Therefore academic education in MT addresses students in linguistics and computer science. Usually, according to the background of the students, courses are given separately to these groups and with different methods: theoretical aspects and demonstration of tools for the linguists (Somers, 2000) on one hand, implementation of clear defined algorithms for the computer scientists on the other hand.

Both ways are successful when we aim at illustrating specific methods only. The disadvantage of this limited aim is that students will not be able to evaluate, configure and design MT system. Accordingly, they cannot answer questions like “How will the system react with huge amounts of data or maximal throughput?”, “How much are supported changes of the domain or changes of languages?”, “What about maintenance by users?” Or, more complicated: “What type of architecture is optimal for my required functionality?”, “What kind of grammar is more suitable?”

The alternative, however, to implement a real MT-system in a course is not feasible, due to lack of time and missing background knowledge of the students. Very often they are facing the field for the first time. A solution in between may be the implementation of “toy systems”, with limited language resources and limited functionality.

In this paper we will two discuss examples of such toy systems, implemented by our students in two different types of courses. In section 2 we will describe the context of the courses: The course structures at Hamburg University, the language background of the students and the preparatory material for students. Section 3 presents the architecture and functionality of the two systems. In section 4 we will give details of the students’ evaluation of their systems and learning success, followed by a discussion of the students’ misconceptions about MT at the beginning of the course.

2. Organisational aspects

2. 1. Type of courses and participants

To understand the following paragraphs, it is necessary to give some details about the structure of courses at Hamburg University (HHU). We offer three types of courses: lectures (accompanied by weekly exercises) seminars and projects. Lectures normally address central issues of

Computer Science, like Object Oriented Programming, Computer Architectures, Formal Languages etc. Specific fields are often taught in the frame of "seminars" and projects. Usually, seminars, after an introduction by the professor(s), consist of student presentations of a particular topic, according to given references. Projects aim to familiarise students with real applications of specific computer science fields; usually the students are required to implement "toy" systems. It is assumed that they have already the theoretical background of the respective field.

According to the above-mentioned division, Machine Translation is taught in Computer Science Faculty at Hamburg University in seminars and projects at the undergraduate level.

Another feature of CS studies at HHU is that students have to choose a minor field as application perspective (economy, law, physics, e.g., but also linguistics or philology). In linguistics, on the other hand, it is quite common that students choose computer science as minor. Therefore in a Machine Translation seminar in CS you will typically find a mixture of three types of students: students from CS with no linguistic background, students from linguistics, with no CS background, and students with reasonable knowledge in both fields. Let us call this mixture MXG group.

The other type, the project, was a two weeks "hands-on seminar" on Machine Translation.[1] It was held with a group of 15 foreign students, with strong background in theoretical computer science and mathematical but no linguistic experience. In the followings we will call them group CSR.

In both, seminar and project, the students were working in small groups of 3-4 students, each responsible for a subtask, as syntax, semantics, lexicon, etc. In the MXG course we tried to have at least one linguist and one CS student in each group. The working groups had to report weekly about the results of their architecture discussions. In the end, the groups had to evaluate their results along criteria of coverage and software quality.

2.2 Choice of Target and Source Language

Participants of MT courses can rightly expect that their mother tongue is target or source language of the exercises. For the group MXG we chose German as source language (mother language of nearly all participants) and English as target language (which was known on a reasonable level to all of them). German as source language, additionally, was chosen because of didactic reasons: during the implementation of a machine translation system, a large amount of work must be devoted to the analysis of the source language. German grammar requires proper treatment of some difficulties like:

- high degree of inflexion *("eines guten Glases Weins")*
- separable verbs and their *ambiguities ("er sah ein anderes Vorgehen ein")*
- composite nouns *("Staubecken", Arbeitsbereichsleitersitzung")*

which do not occur in English.

With the group CSR, the source language was German and the target language Romanian (their mother tongue). The choice of Romanian as target language follows the idea that the students should be able to evaluate the quality of the target expressions of their systems. Correspondingly, the selected domain was the syllabus of CS courses of HHU.

2.3 Preparatory material and test corpus.

The domain was defined in advance:

- car repair for the MXG group

[1] This course was financed by DAAD.

- the syllabus of CS courses of HHU for the CSR group

For both groups we provided a corpus containing about 100 test sentences. In order to show the difficulties of real natural language but also encourage students we prepared three groups of sentences:

- Original sentences, taken from technical manuals or university brochures.
  Example 1(a): *"Im Praktikum soll für eine Datenbank eine Schnittstelle entwickelt werden, die die graphische und natürlichsprachlichen Mittel für Anfrage und Darstellung des Ergebnisses kombiniert",*
- slightly simplified sentences, where for example difficult relative clauses were deleted.
  Example 1(b): *"Für eine Datenbank wird eine natürlichsprachlichliche Scnittstelle entwickelt werden"*
- artificial sentences, with basic syntax and semantics
  Example 2: *"Die Lehrveranstaltung gibt eine Einführung in die Computergraphik".*

There was also provided a full-form lexicon of the corpus, initially annotated only with PoS tags. Additional information was provided on request, i.e. when the students realised that they needed additional features for specific task (inflexion features, semantic classes, etc.). To make processing easier, the format of the Lexicon was XML-like:

Example :

```
<entry="kombiniert">
<hom="1">
<PoS="verb">
<tense="part2">
<pers="">
<gen="">
<num="">
<Rom="combinat">
</entry>

<entry="kombiniert">
<hom="2">
<PoS="verb">
<tense="pres">
<pers="3">
<gen="">
<num="sg">
<Rom="combin_">
</entry>

<entry="kombiniert">
<hom="3">
<PoS="verb">
<tense="pres">
<pers="2">
<gen="">
<num="plr">
<Rom="combina_i">
</entry>
```

The basic tasks (i.e. the groups and hence the module structure) have been predefined as

Lexicon
Word-to-word-processing
Morphology
Syntax
Semantics
World Knowledge
Architecture

3. System architecture

As we did not define control structure nor system architectures, the students had to find a system design by discussions among the working groups.
In the following we describe the two types of system architectures chosen by the students., both belonging to the class of transfer systems.
The architecture of System 1 (see Figure 1) included 4 independent modules each providing independently a different translation type:

- pattern matching,
- word-to-word translation,
- syntactic translation and
- semantic translation.

The input of the system is the source text and the translation type. For the same input, sequentially up to four types of translations can be started and the results can be compared.

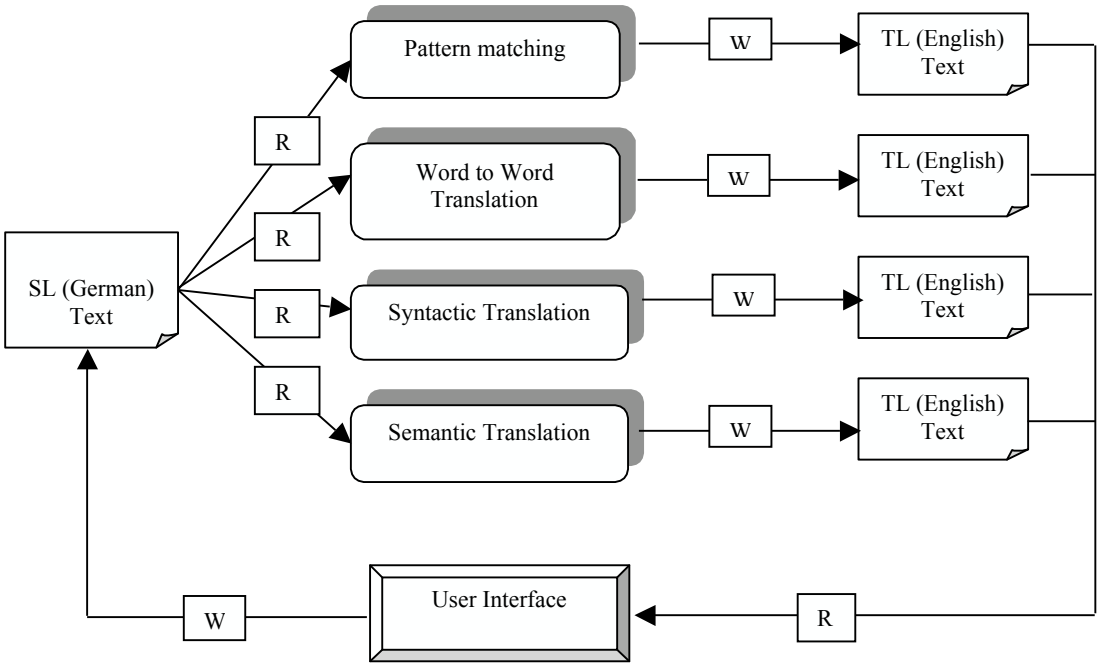

Figure 1. Architecture of System 1

The advantage of this approach is that the students can estimate themselves the limitations of each approach. Out of the provided test set, for each sentence at least one module succeeded to deliver a translation. There was no common language resource or intermediate results memory among the modules. The start-up lexicon was modified by each group according to their internal requirements. The simplicity of the architecture is the result of the restricted time resources in a seminar-type course.

The architecture of system 2 is the result of a ten full days implementation time. There are 4 modules (word-to-word translation, parsing, semantic analysis and domain knowledge) which co-operate to produce a translation. The architecture is strictly sequential; there is one input (German) and one output (Romanian) and the modules work in the order mentioned above (see Figure 2). The communication between modules is realised through a backbone. This backbone contains all possible intermediate results produced by each module; these intermediate results were called *readings.*

The word-to-word translation module takes the source text as input and produces lists of morphological units as output, which represent possible morphological equivalents of the input and their lexical correspondences. The module performs also pre-processing of the input sentence in order to identify possible idioms and to bring together parts of compound or separated verbs. Another pre-processing was necessary for particularities of Romanian morphology, where the definite article is always a suffix of the noun. Therefore in the German morphological analysis a morphological unit will always contain the noun together with the article information.

Example: The input sentence is:

> *"Die Literatur wird in der ersten Sitzung ausgegeben."*

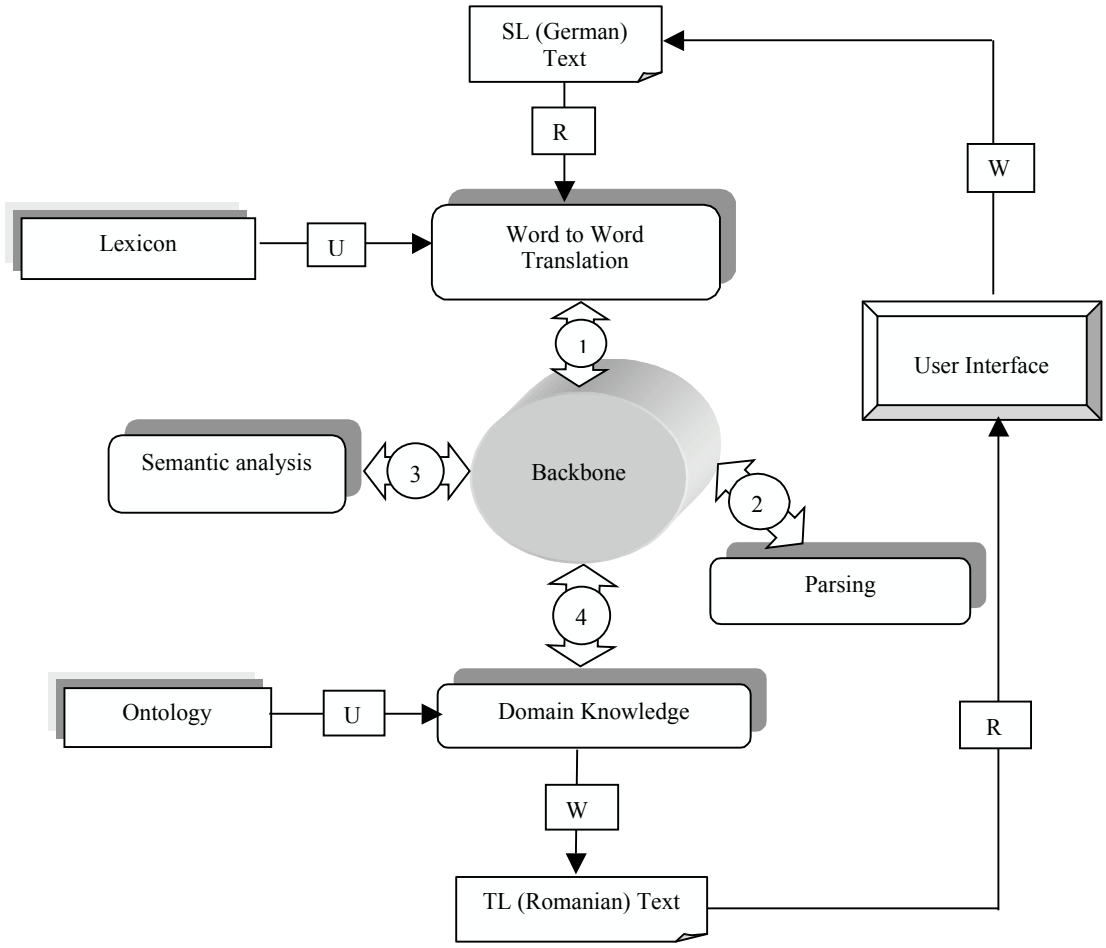

Figure 2. Architecture of System 2

After processing by the word-to-word translation module, the backbone will contain the following "readings":

Die Literatur
Noun, "bibliografie",
Det="det",
gen="fem",
cas="noun",
num="sg"

wird ausgegeben
verb, "este furnizata",
mod="ind",
tense="pres",
num="sg",
pers="3"

in
prep, in
cas="dat"

der Sitzung
Noun, "sedinta",
Det="det", gen="fem",
Num="sg",cas="dat"

ersten
adjective, "prim"
det="det" gen="fem",
num="sg",cas="dat"

If a word has more than one possible translation, several readings will be inserted in the backbone.

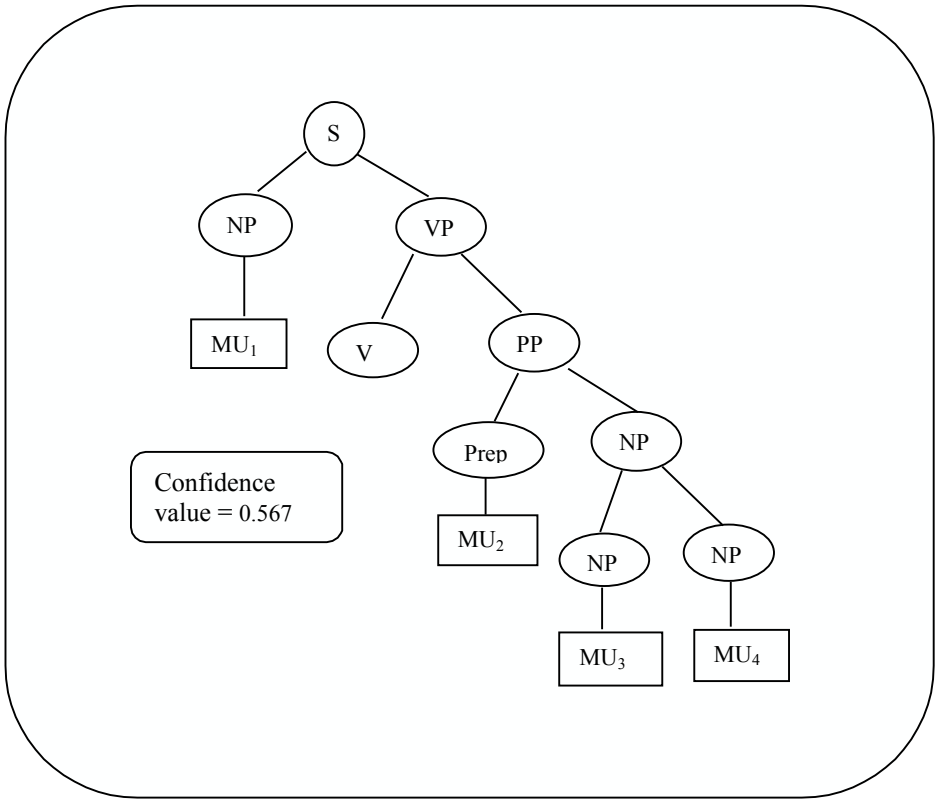

Figure 3: Syntactic reading in System 2

The syntactic parser has the morphological "readings" as input and produces syntactic trees with associated confidence values as output. These values computed from stochastic probabilities associated with the grammar of the source language. These syntactic trees are again placed in the backbone. In figure 3 we present a syntactic reading.

The semantic analysis module takes a syntactic tree reading from the backbone as input and outputs a "dependency tree reading". Additionally, the module performs the target language morphology generation and uses the inflected word forms from the target language lexicon.

The domain knowledge module performs disambiguation based on a domain ontology. Highly unreasonable readings are marked for discarding (the confidence value is decreased).

The output of the whole system is the translation of the reading with the highest confidence value.

4. Evaluation

Two steps of evaluation were carried out by the students: One concerning their own results and another concerning the teaching success.

4.1 Evaluation of the systems

In both groups we asked the students to prepare a theoretical presentation of their subtask at the beginning of the course and at the end of the course a presentation of their work and the evaluation of their

software. The system evaluation criteria were given by us and contained

- 17 criteria for software quality, among them maintainability, conviviality, or efficiency, and
- 7 criteria for the linguistic functionality, like lexical coverage, syntactic coverage, or compatibility with (European) standards and formats.

The aim was to familiarise students with evaluations of projects. For some criteria, however, like maintanability or domain coverage, there was no reasonable answer concerning a "toy" system, but the idea was to face the students early enough with all aspects of software evaluation in general and machine translation in particular.

Obviously both presented systems have a lot of limitations. In the case of System 1 the translation is limited exclusively to the sentences contained in the test corpus. The user has no possibility to enter other sentences.

In the case of System 2 the interaction with the user is free but the translation is limited first by the size of the lexicon and secong, by the parser. The target language generation is still problematic, and long relative clauses are not passing the parsing process.

4.2 Evaluation of the educational success

The second evaluation concerned the instruction process. The students were asked to answer (anonymous) to questionnaires, referring both to quality of the preparatory material and learning success. In table 1 we present the results of this evaluation according to different criteria. The results proved that the students rated the subject very interesting and the result of the course as "learning success". The difference between "good" and "very good" apparently stem from the student's expectation that a "normal" CS course is rather formal and does not address a subject from humanities. Moreover the majority of undergraduate CS students (the MXG group) is primarily interested in business applications.

The criticised points were:

- *the lack of time*, which is in a way related to the fact that an MT system can be always improved (so it will be never enough time to cover all aspects of the languages during a one term course). The expectation apparently was a compact task with only one technical solution, which can be completed in a defined time.
- *the lack of defining* very precise tasks for each student. The students (as they explained during the discussion) expected very clear definitions of the algorithms to be used, and saw their work as simple programming. We pointed out, in contrast, that such a course is not a mere programming exercise but an introductory and experimental way to learn MT techniques by comparing results of chosen methods (and there combinations) to the own linguistic intuition.

4.3 Misconceptions about Machine Translation

During the implementation several well known misconceptions about MT were brought up for discussion by the students.

We think that an implementation of a MT system from scratch (even if it is only a "toy" system) can overcome most of the false ideas about MT. In the following we give a list of such misconceptions and the (counter-) arguments that the students usually found by themselves.

	Interesting (%)		Didactic design (%)		Seminar style (%)		Preparation by profs. (%)		Learning success (%)		Time effort (%)	
	MXG	CSR	MXG	CSR	MXG	CSR	MXG	CSR	MXG	CSR	MXG	CSR
Very good	55	76	27	32	27	69	27	46	18	53	9	7
Good	45	24	63	61	47	24	73	54	64	24	72	16
Well...	0	0	9	0	27	7	0	0	18	23	9	24
Must be improved	0	0	0	7	0	0	0	0	0	0	9	53

Table 1: Evaluation of the educational success

- Processing of natural language is very similar to processing formal languages (v. Hahn, 1999), and the slightly modified versions:
  - natural language processing mainly consists of parsing,
  - constituent grammars are the only option for grammatical description.

  ➢ Counter-experience: The students implemented a phrase structure grammar. Although it was easier for them from the theoretical point of view, in the transfer and generation phase they recognised that a dependency grammar or word-centered processes might be reasonable alternatives. For example: The students started with designing a typical formal language grammar, and it turned out that they need annotations expressing selectional restrictions. Moreover, after parsing all sentences, they found, that not all ambiguities could be resolved at the morpho-syntactic level

- acceptable results for specific tasks/domains can be obtained by word-to-word-translation (Pérez, 2000).

  ➢ Counter-experience: One group in each seminar was responsible for word correspondencies. They presented their output and discussed the inadequacies in detail.

- if the algorithms are clear, it is easy to implement a machine translation system.

  ➢ Counter-experience: When discussing ambiguity, ellipses, pronoun resolution (if necessary), vagueness, and cultural correctness, they immediately saw that MT is a special field, where heuristics, corpus-based statistics and world knowledge are necessary for the correct treatment of many phenomena. This finding was even true for the given domains (see section 2.3).

- For MT, as for all programms, you can give the (or a list of) correct result to be met by the program..

  ➢ Counter-experience: As a result of some design decisions it was extremely difficult to meet exactly the translation example (given in the preparatory material). The achieved translation, however, was correct, fluent and adequate.

5. Conclusions

Many institutions (Balkan, 1997) use market systems for MT teaching. According to our experience, however, hand-on teaching with implementations and much freedom of choice in practical seminars has several advantages:

1. design principles of MT systems become transparent,
2. interaction of modules can be explicitely discussed,
3. the correspondence between design decisions and evaluation is clear for all students
4. students can judge the quality improvement of possible extensions (pre- or postprocessing) or modifications of market systems.
5. evaluation of translation quality can be discussed in relation to system design alternatives.

The demo systems of the student groups are available on request. We intend to continue the series of "toy" system implementations for demonstrating spoken language translation also.

References

(Balkan, 1997) Balkan L., Arnold D., Sadler L., „Tools and Techniques for Machine Translations Teaching: A survey", ELSNET report, 1997

(v. Hahn, 1999) vHahn W., „Natural Language Processing at School ?", in „Restructuring of the retraining of school teachers in computer science", Ed. Libris, Cluj, 1999

(Pérez, 2000) Pérez-Ortziz J., Forcada M., „Discovering Machine Translation Strategies Beyond Word-for-Word Translation: a Laboratory Assignment", in Proceedings of the workshop on „Teaching Machine Translation" related to the WIIth MT Summit, Santiago de Compostella, 2000

(Somers, 2000) Sommers H., „Three Perspectives on MT in the Classroom", in Proceedings of the workshop on „Teaching Machine Translation" realtiod to the WIIth MT Summit, Santiago de Compostella