Machine Translation **17:** 245–270, 2002.
© 2004 *Kluwer Academic Publishers. Printed in the Netherlands.*

MT for Minority Languages Using Elicitation-Based Learning of Syntactic Transfer Rules

KATHARINA PROBST, LORI LEVIN, ERIK PETERSON, ALON LAVIE and JAIME CARBONELL
Language Technologies Institute, Carnegie Mellon University, Pittsburgh, PA, USA

Abstract. The AVENUE project contains a run-time machine translation program that is surrounded by pre- and post-run-time modules. The post-run-time module selects among translation alternatives. The pre-run-time modules are concerned with elicitation of data and automatic learning of transfer rules in order to facilitate the development of machine translation between a language with extensive resources for natural language processing and a language with few resources for natural language processing. This paper describes the run-time transfer-based machine translation system as well as two of the pre-run-time modules: elicitation of data from the minority language and automated learning of transfer rules from the elicited data.

Key words: elicitation, rule learning, syntactic transfer rules, minority languages

1. Introduction

In recent years, much of machine translation (MT) research has focused on two issues: portability to new languages and developing MT systems rapidly. At the same time, there is increasing concern for the promotion of minority languages, as evidenced internationally by a growing number of programs supported by government agencies, non-governmental organizations, indigenous communities, linguistic organizations, etc. (e.g. ILASH, 2002). The current trend is to enable speakers of minority languages to have access to education and health care and to participate in government and the internet without having to give up their languages. An ideal paradigm for portability and rapid development of MT would provide a low-cost way to develop MT for any language, including those with sparse electronic and economic resources. For minority languages, we often encounter both of these problems: little or no data in electronic form, and few financial resources that could benefit the development of an MT system.

The AVENUE MT project[1] addresses the development of MT systems for languages with extremely sparse resources. With respect to MT, languages with "sparse resources" are those that lack a large corpus in electronic form and NLP tools such as parsers. Additionally, MT developers may encounter the problem that the language will have few if any native speakers trained in (computational) linguistics. There may be other difficulties as well, such as spelling and orthographical conventions that are not standardized, and missing vocabulary items due to vocabulary loss.

AVENUE contains a run-time MT program that is surrounded by pre- and post-run-time modules. The pre-run-time modules are concerned with elicitation of data and automatic learning of transfer rules. The post-run-time module uses statistical methods for selection of translation alternatives. As the post-run-time module is still under development, this paper focuses on the run-time transfer-based MT system as well as two of the pre-run-time modules: elicitation of data from the minority language and automated learning of transfer rules.

1.1. NOVEL ASPECTS OF THE AVENUE PROJECT

AVENUE takes a novel approach to rapid, low-cost development of MT systems. Although we are primarily concerned with automatic learning from corpora, the output of the learning mechanism is a set of human-readable rules rather than a set of statistics. The learning mechanism is innovative in several ways. A Compositionality module serves to learn context-free mappings between source and target language structures, in effect inferring a context-free transfer grammar. Seeded Version Space Learning (SVSL) is based on Version Space Learning (Mitchell, 1982) in that it considers transfer rules as residing in a partially ordered space, but differs in that it identifies seed rules before hypothesizing generalizations of the rules. SVSL targets the learning of appropriate unification constraints.

The transfer formalism is unusual in the sense that each transfer rule incorporates parsing, transfer, and generation information. This approach is useful for our learning module: it allows us to produce first guesses at transfer rules directly from the parallel data, incorporating into the rule all that is needed to translate a given sentence pair. This close relationship between the training instances and the produced rules becomes important in the learning process (Section 5).

The AVENUE system is extremely flexible in its design: it is “omnivorous” in that it can use whatever resources are available. If human linguists are available, they can write grammar rules. If bilingual speakers are available, they can produce a word-aligned bilingual corpus for rule learning. If parallel corpora are available, rule learning can be applied to those corpora in addition to any elicited data, and Example-based MT can also be applied.

In addition to the core MT research, our efforts focus on strong collaborative ties with indigenous groups. For instance, we have formed a partnership with the Chilean Ministry of Education to collect data and produce language technologies that support bilingual education in Spanish and one of Chile’s minority languages, Mapudungun (Levin et al., 2002). Under this partnership, spoken conversations in Mapudungun are being recorded and transcribed by native speakers of Mapudungun at the Universidad de la Frontera in Temuco, Chile. The resulting corpus of

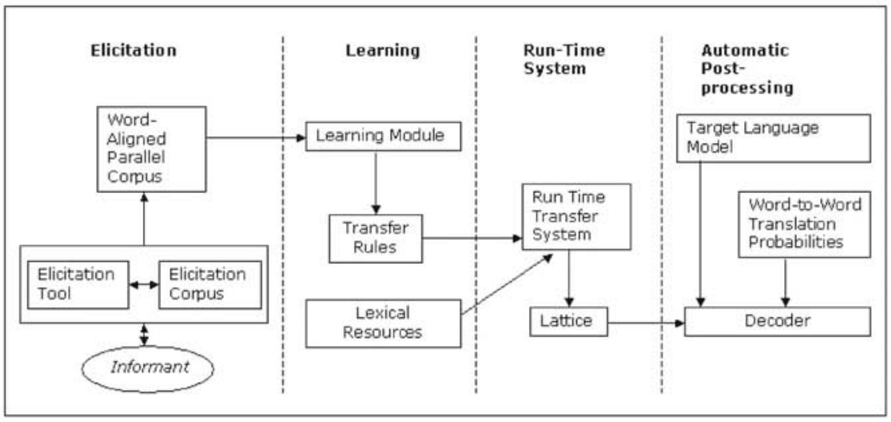


Figure 1. AVENUE's rule learning and translation system architecture.

170 hours of transcribed speech has been translated into Spanish and is being used further as the basis of a web-based dictionary for Mapudungun learners. In future research, it will be used as the basis for a Mapudungun–Spanish translation system.

This article is organized as follows: First, we give some background to our work and an overview of the design of our system. Then we discuss in detail the elicitation module, including issues regarding the informant, the corpus, and a discussion of the advantages and disadvantages of learning from an elicited corpus. We then explain the core MT engine and the rules that it uses. The following section describes how rules can be learned automatically from the corpus. Finally, we give preliminary evaluation results for the rule-learning module and conclude.

2. System Design

The AVENUE system consists of four subsystems: elicitation of a word-aligned paralled corpus; automatic learning of transfer rules; the run-time transfer system; and selection of translation alternatives via a statistical decoder. Figure 1 shows how the four sub-sytems would be used in order to build and run a translation system with a minority language as the source language (SL) and a major language (such as English or Spanish) as the target language (TL).

The purpose of the elicitation system is to collect a high-quality, word-aligned parallel corpus. Because a human linguist may not be available to supervise the elicitation, a user interface presents sentences to informants. The informants must be bilingual and literate in the language of elicitation and the language being elicited, but do not need to have training in linguistics or computational linguistics (Probst et al., 2001; Probst and Levin, 2002). They translate phrases and sentences from the elicitation language into their language and specify word alignments graphically (Section 3.1).

The rule-learning system (Section 5) takes the elicited, word-aligned data as input. Based on this information, it infers syntactic transfer rules using SVSL. The system also learns the composition of simpler rules into more complicated rules, thus reducing their complexity and capturing the compositional makeup of a language (e.g., NP rules can be plugged into sentence-level rules). The output of the rule-learning system is a set of transfer rules that then serve as a transfer grammar in the run-time system.

At run time, the translation module takes as input an SL sentence. The output is a lattice of translation alternatives, where each entry in the lattice represents a partial translation hypothesis. The alternatives arise from syntactic ambiguity, lexical ambiguity, multiple synonymous choices for lexical items in the dictionary, and multiple competing hypotheses from learned rules.

In the final stage the best path through the lattice is selected by a statistical decoder (Lavie et al., forthcoming). The decoder selects lexical items in the TL using probabilities of source-word to target-word translations (based on the elicited data or on a larger parallel corpus if it is available). The language model is used by the decoder to determine the most probable sequence of TL words in the lattice. The decoder can also perform a limited amount of re-ordering of words in the lattice to obtain a better fit to the TL model (Lavie et al., forthcoming).

Figure 1 represents one scenario for use of the AVENUE system – translation into a major language from a sparse resource language with no pre-existing parallel corpora. However, AVENUE is adaptive to other scenarios as well. For example, if a parallel corpus can be obtained and aligned by some means other than elicitation, the elicitation process can be bypassed. In this case, the learning module contains an extra step for bootstrapping the translation lexicon by automatically learning parts of speech and other syntactic features (Probst, 2003) that would otherwise be learned through elicitation. Furthermore, if corpora are available, other corpus-based methods such as Example-Based MT and Statistical MT can be combined in a multi-engine system along with the transfer rule system (Lavie et al., forthcoming). If translation is from the major language into the sparse resource language where no large language model can be trained, the statistical decoder would be replaced by general heuristics for disambiguation and lexical selection.

At this point in the AVENUE project we have conducted experiments using major languages with restricted amounts of data to simulate a sparse resource situation. Working with major languages allows us to focus on a subset of the research issues while using existing resources to fill in other components of the system. For example, we have worked on elicitation, rule learning, a run-time transfer system, and statistical decoding for Hindi (Lavie et al., forthcoming) without having to worry about how to acquire morpholgical rules because an existing morphological analyzer[2] was available. Other experiments were conducted on languages spoken by project members (English, German (this paper), French (Probst, 2003), and Chinese) so that algorithms could be designed without having to hire consultants for sparse resource languages. We are currently in the process of developing lexical resources for Mapudungun in preparation for conducting MT experiments. In order to make our system more readily adaptable to other languages, we are also developing an unsupervised morphology learning module, taking an approach similar to Goldsmith (2001).

3. Elicitation Module

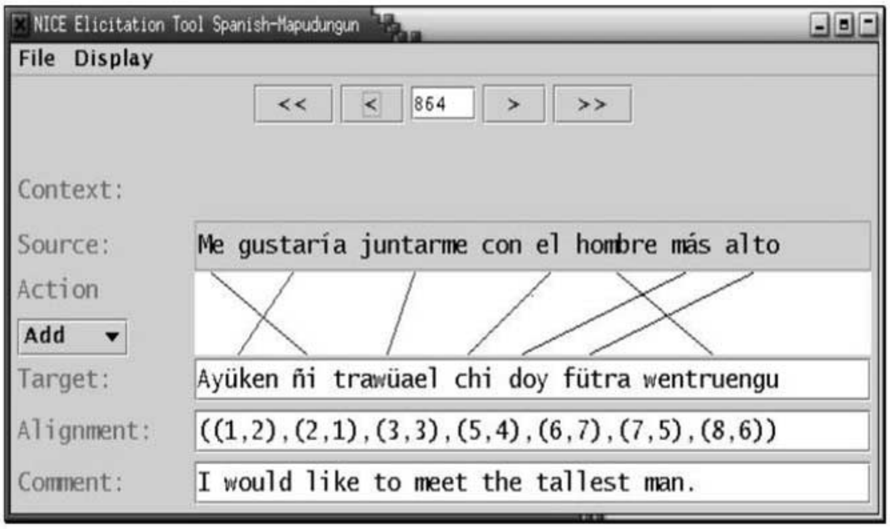
Since human expertise on minority languages may be in short supply, and handcrafted development can be a time-intensive process, automated learning of statistics or rules has been critical to both language portability and rapid development. Automated methods have typically been trained on large, uncontrolled parallel corpora (Brown et al., 1990, 1993; Brown, 1997; Papineni et al., 1998; Och and Ney, 2002; Yamada and Knight, 2001; Vogel and Tribble, 2002). However, a minority of projects (Nirenburg, 1998; Jones and Havrilla, 1998) have addressed automated learning of translation rules from a controlled corpus of carefully elicited sentences. The AVENUE learning module also works from a controlled corpus.

The objective of the AVENUE elicitation module is to obtain a parallel corpus of high-quality word-aligned phrases and sentences that can be input to the rule learning module. The elicitation module consists of a corpus and a graphical interface. The corpus is a list of sentences or phrases in a major language such as English or Spanish. The interface presents each sentence or phrase to a bilingual informant and allows them to translate and identify word alignments. These components are described below.

3.1. ELICITATION INTERFACE

Figure 3.1 illustrates the AVENUE elicitation interface. In order to obtain a corpus of word-aligned text, a bilingual informant is presented with sentences or phrases in English or Spanish (or another major language) and is then asked to translate the sentence to the language of interest using the elicitation interface. After translating, the informant then indicates for each word in the elicited sentence the corresponding word(s) in the elicitation language. In some cases, the word may not have a corresponding word in the matching sentence, or may correspond to multiple words.

The informant identifies word alignments graphically by using the mouse to click on a word in one sentence and then the corresponding word in the other sentence. A link is automatically added, which is visually indicated with a line between the two words and color-highlighting of the newly aligned words. Deletion of an existing alignment pair is equally straightforward: the informant clicks on the first word of the existing pair and then clicks on the second word. The link is then deleted. In the case where two or more (possibly non-contiguous) words act as a unit, the informant can hold down the control key while clicking on the words with the mouse. In this way, one-to-many and many-to-many word alignments are



Context:

Source: Me gustaría juntarme con el hombre más alto

Target: Ayüken ñi trawüael chi doy fütra wentruengu

Alignment: ((1,2),(2,1),(3,3),(5,4),(6,7),(7,5),(8,6))

Comment: I would like to meet the tallest man.

Figure 2. Elicitation interface.

possible. The alignments are shown as lines connecting SL and TL words and also as pairs of indices such as `(1,2)` indicating that the first Spanish word is aligned with the second Mapudungun word.

The language in the menus and buttons of the interface, as well as the help file, can be easily changed, and has been localized for English and Spanish.

Aside from the basic function of translating and word-aligning sentences, the elicitation interface also has several support tools. From the main menu the user can display a bilingual glossary built from existing word alignments. Users can also easily search for words in the eliciting or elicited sentences. The search results include the sentence the word was found in, along with any aligned words. Clicking on a result links directly to the sentence. This makes it easier for users to be consistent in their alignment work.

3.2. GENERAL DESIGN OF THE ELICITATION CORPUS

The elicitation corpus is a list of sentences in a major language such as English or Spanish. It is designed to cover major linguistic phenomena in typologically diverse languages. The elicitation process emulates a field linguist in systematically identifying characteristics of a language. In designing the elicitation corpus, we have drawn heavily on guides for field linguists such as Bouquiaux and Thomas (1992) and Comrie and Smith (1977). We assume that no information is known about the target language at the beginning of the elicitation process.

The elicitation corpus has two organizational properties: it is organized into minimal pairs; and it is compositional, i.e., starting with smaller phrases and combining them into larger phrases. Compositionality in rule learning is addressed in Section 5.2. Here we will address the use of minimal pairs in the elicitation corpus.

A minimal pair can be defined as two sentences that differ only in one linguistic feature. Each sentence can be associated with a feature vector. Consider, for instance, the three sentences (1)–(3).

(1) The man saw the young girl.
(2) The men saw the young girl.
(3) The woman saw the young girl.

Sentence (1) and (2) differ merely in one feature, namely the number of the subject. Thus, they represent a minimal pair. However, sentences (1) and (3) also form a minimal pair. They differ only in the gender of the subject.

The organization of the corpus in minimal pairs is specific to our use of the elicited data. An ideal informant would translate a minimal pair in the elicitation language into a minimal pair in the elicited language. (The issue of non-ideal informants is addressed below.) Each minimal pair is intended to test for the presence or absence of a grammatical feature in the elicited language. For instance, if we want to find out whether target-language verbs inflect depending on the number of the subject, we can use sentences (1) and (2) above: they differ only in the number of their subject, while all other features are kept constant. An indication of whether the target-language verbs agree in number with their subjects can then be found by comparing the verbs in the elicited versions of sentences (1) and (2). Verb agreement with the gender of the subject can be tested by comparing the elicited versions of sentences (1) and (3). The elicitation corpus contains many redundant minimal pairs for each feature so that the system will not be misled by isolated exceptions to the general rules of the language.

A feature-detection module is contained in the AVENUE learning module for the purpose of comparing minimal pairs and noting the presence or absence of features (Probst et al., 2001). The detection module is coupled with a navigation module that chooses a different course through the elicitation corpus depending on what features have been detected. This idea is based on linguistic universals that identify clusters of co-occurring or non-co-occurring features: research on implicational universals has found that certain features are guaranteed not to exist in a language if certain other features are not present. If a language does not mark plural, then it will also not mark dual or paucal (on implicational universals see for example Comrie (1981) and Greenberg (1966)). The AVENUE navigation module should therefore try to elicit dual and paucal nouns only if the feature-detection module has detected plural nouns. In other cases, tests cannot be performed properly without knowing the results of other tests. For example, the form of adjectives often depends on the noun class of their head noun. If we have not yet performed tests to determine the noun classes in a language, it would be difficult to learn agreement patterns of adjectives. Ordering of the elicitation process is therefore an important issue.

The current elicitation corpus is a pilot study with about 2,000 sentences (in English, some of which have been translated into Spanish), though we expect it to grow to at least the number of sentences in Bouquiaux and Thomas (1992), which includes around 6,000 to 10,000 sentences, and ultimately to cover most of the phenomena in the Comrie and Smith (1977) checklist for descriptive grammars. The phenomena covered in the pilot corpus include number, person, and definiteness of noun phrases; case marking of subject and object; agreement with subjects and objects; differences in sentence structure due to animacy or definiteness of subjects and objects; possessive noun phrases; and some basic verb subcategorization classes.

The elicitation corpus has been translated and aligned in English–Chinese, English–Hindi, and Spanish–Mapudungun. Most informants can translate and align around 1,000 phrases and sentences per week. With a small amount of feedback, most of our informants have been able to provide fairly consistent translations and alignments. However, elicitation is never an easy process even when conducted by human linguists. We now turn to some of the potential pitfalls of computer-based elicitation.

3.3. CHALLENGES FOR COMPUTER-BASED ELICITATION

Word alignment: As has been pointed out in the literature (Melamed, 1998), human-specified alignment is not noise-free. There is some degree of disagreement between people who align the same sentences, and any given person will also not be completely consistent. Informants can be given intuitive instructions for alignment (Melamed, 1998). Yet, the learning process must still be tolerant to noise in alignments, and we must also be prepared to accept non-optimal rules that are learned from noisy data. It should be noted, however, that while human alignments are not error-free, automatic alignment will suffer from more severe noise problems. Therefore, tolerance to noise in word alignments must be dealt with in any automatically learned MT system.

Orthographic and segmentation issues: Many of the languages AVENUE is designed for, such as Mapudugun, do not have standardized orthographies. Orthographic variations include dialectal differences, characters from competing alphabets, and inconsistent word segmentation. Naturally, if the same word is written with two or three different spellings throughout the elicitation corpus, a computational system will not take them to be the same word, and will create different alignment pairs as well as different dictionary entries. For this reason, an important part of the AVENUE project has been automatic tools for normalizing orthogaphic variants, as well as the adoption of a standardized orthography for the project's databases (Levin et al., 2002).

Morphology: In addition to learning which inflectional features are present in a language, we must learn which sequences of characters or perturbations of stems are associated with each inflectional feature. We are currently investigating learning morphology from an untagged, monolingual training corpus, as has been done by other groups, e.g., Gaussier (1999) and Goldsmith (2001). Of course segmented or aligned corpora (such as our elicitation corpus) are useful, if they are available, for learning morphemes.

Elicitation of non-compositional data: In certain cases, we can expect *a priori* that the source and target language sentences will differ widely in their grammatical construction. Many fixed expressions and special constructions contribute to "translation divergences" (Trujillo, 1999; Habash and Dorr, 2002), meanings that are expressed with different syntax in the source and target languages. A typical example of a special construction is *Why not* `VP-infinitive` as in (4) as a way of performing the speech act of suggesting in English. This construction poses a problem for our compositionally learned transfer rules in two ways: first, it does not follow normal English syntax in that it does not contain a tensed verb or a subject. Second, it does not translate literally into other languages using the translation equivalents of *why*, *not*, and infinitive. For example, conventional ways of saying the same thing in Japanese uses a conditional or gerundive form for the main verb (5).

(4) Why not go to the conference?

(5) a. *Gakkai ni ittara, doo?*
CONFERENCE TO GO-conditional, HOW

b. *Gakkai ni ittemitara, doo?*
CONFERENCE TO GO-gerund SEE-TRY-conditional, HOW.

In contrast to the literature on MT mismatches, which seems to assume that mismatches are exceptions to an otherwise compositional process, we believe that conventional, non-compositional constructions are basic and on an equal footing with compositional sentences (as in Construction Grammar, Fillmore et al., 1988). In fact, in our other projects, we have capitalized on the inherent non-compositionality of some types of sentences, such as those found in task-oriented dialogues (Levin et al., 1998).

Our plans for non-compositional constructions are based on the observation that some types of meaning are more often a source of mismatches than others. Specific sections of the elicitation corpus will be devoted to eliciting these meanings including modalities such as obligation and eventuality; evidentiality; comparatives; potentials, etc. An interesting field of investigation will be to exploit compositional subcomponents of non-compositional constructions such as the `VP` structure of *go to the conference* in (4).

Verb Phrases: Verb subcategorization patterns are another source of MT mismatches as verbs do not agree in their subcategorizations across languages (see e.g., Trujillo, 1999: 124). For example, in the English construction in (6a), the verb *declare* subcategorizes for two noun phrases, an indirect object (*him*), and a direct object (*Prime Minister*). In German, the verb *ernennen* 'declare' subcategorizes for a direct object (*ihn*, 'him'), as well as a prepositional phrase introduced by the preposition *zu* 'to'. Subcategorization mismatches are common, and may be random or systematic according to verb classes (Dorr,1992). Thus our aim will be lexical transfer rules for individual verbs or verb classes as opposed to lexically independent verb phrase rules.

(6) a. He declared him Prime Minister.
b. *Er ernannte ihn zum Premierminister.*
HE DECLARED HIM TO-THE PRIME-MINISTER

Limits of the elicitation language: A problem always faced by field workers is that elicitation may be biased by the wording of the elicitation language, or by the lack of discourse context. A language that has word order determined by the structure of given and new information may be difficult to elicit using only isolated English sentences. Furthermore there are many categories such as evidentiality (whether the speaker has direct or indirect evidence for a proposition) which can be expressed in English (e.g., *Apparently, it is raining* or *It must be raining* are expressions of indirect evidence) but, unlike in other languages, they are not an inherent part of the grammar. In order to elicit phenomena that are not found in the elicitation language, it may be necessary to include discourse scenarios in addition to isolated elicitation sentences. The AVENUE elicitation interface therefore has a context field which can contain additional information about the sentence (such as the gender of one of the participants) that may influence the translation. In the future we may use the context field to provide more extensive discourse context in order to be sure that the word order in elicited sentences is influenced by the discourse context rather than by the word order in the elicitation language.

3.4. COMPARISON OF CONTROLLED AND NATURALLY EXISTING CORPORA AS A BASIS FOR LEARNING TRANSLATION RULES

Although the AVENUE learning module could be applied to any word-aligned corpus, we initially focused our efforts on learning from the elicitation corpus, which is a tightly controlled corpus. As an alternative, or in addition, if a naturally occurring parallel corpus is available, it can be word-aligned automatically or by humans, and can then be used as input to the learning module.

The issues in choosing a corpus are reminiscent of the contrast between using uncontrolled corpora or controlled data for research in linguistic theory. Any unstructured training corpus will lack examples of various linguistic phenomena, with the problem becoming more prevalent the smaller the corpus is. A controlled corpus can cover more phenomena, but may treat all phenomena as equally salient and not reflect the frequency or importance of the phenomena in real text. A controlled corpus can be designed once and can then be applied to all languages. A naturally occurring corpus, on the other hand, would be different for each new language, but it would not require the development effort that goes into hand-crafting a controlled corpus.

Recent experiments conducted by the AVENUE team (Lavie et al., forthcoming) have led to a compromise between using controlled and naturally occurring corpora. It may be the case that a naturally occurring parallel corpus does not exist for a pair of major and minority languages, but naturally occurring monolingual corpora certainly exist for the major language. Using robust parsers or treebanks for this language, it is possible to obtain a set of phrase structure trees. The trees can be broken into smaller phrases, and the smaller phrases can be sorted according to their daughter nodes (e.g., `NPs` containing just `N`, `NPs` containing `Det` and `N`, etc.). Features of the phrases such as number, case, and tense can also be extracted by a parser for the major language. The phrases can then be ordered into an elicitation corpus that can be translated and aligned by informants.

The resulting elicitation corpus is natural in that it reflects the frequency of naturally occurring grammatical phenomena, but it also retains the advantage of a controlled corpus in being ordered compositionally. The natural elicitation corpus will, however, be limited to phenomena that occur in the elicitation language. Therefore, our future efforts will involve combinations of controlled and naturally occurring elicitation corpora.

4. The Transfer Rule Formalism

We now turn to a description of the formalism for the AVENUE transfer rules. The transfer rules can be written by hand, or can be output automatically by the rule-learning mechansim. The transfer rules are comprehensive in the sense that they include all information that is necessary for parsing, transfer, and generation, similar to the "modified" transfer approach used in the early METAL system (Hutchins and Somers, 1992). In this regard, they differ from "traditional" transfer rules that exclude parsing and generation information. Despite this difference, we will continue to refer to them as transfer rules.

The following list summarizes the components of a transfer rule. In general, the *x*-side of a transfer rule refers to the source language (SL), whereas the *y*-side refers to the target language (TL).

– **Type information:** This identifies the type of the transfer rule and in most cases corresponds to a syntactic constituent type. Sentence rules are of type `S`, noun-phrase rules of type `NP`, etc. The formalism also allows for SL and TL type information to be different.

– **Part-of-speech/constituent information:** For both SL and TL, we list a linear sequence of components that constitute an instance of the rule type. These can be viewed as the "right-hand sides" of context-free grammar rules for both SL and TL grammars. The elements of the list can be lexical categories, lexical items, and/or phrasal categories.

```
{S,2}  ; Rule identifier
S::S : [NP VP MA] -> [AUX NP VP]
(
   (x1::y2) ; set constituent alignments
   (x2::y3)

   ; Build Chinese sentence f-structure
   ((x0 subj) = x1) ; NP supplies subj. f-structure
   ((x0 subj case) = nom)
   ((x0 act) = quest) ; speech act is question
   (x0 = x2)

   ((y1 form) = do) ; base form of AUX is "do"

   ((y3 vform) =c inf) ; verb must be infinitive
   ((y1 agr) = (y2 agr)) ; "do" must agree with subj.
)
```

Figure 3. Sample transfer rule.

- **Alignments:** Explicit annotations in the rule describe how the set of SL components in the rule align and transfer to the set of TL components. Zero alignments and many-to-many alignments are allowed.
- ***X*-side constraints:** The *x*-side constraints provide information about features and their values in the SL sentence. These constraints are used at run-time to determine whether a transfer rule applies to a sentence.
- ***Y*-side constraints:** The *y*-side constraints are similar in concept to the *x*-side constraints, but they pertain to the TL. At run-time, *y*-side constraints serve to guide and constrain the generation of the TL sentence.
- ***XY*-constraints:** The *xy*-constraints provide information about which feature values transfer from the SL into the TL. Specific TL words can obtain feature values from the SL sentence.

For illustration purposes, Figure 3 shows an example of a transfer rule for translating Chinese *yes/no*-questions into English. Chinese *yes/no*-questions use the same word order as declarative sentences, but end in the particle *ma*. The transfer rule shown in Figure 3 adds the auxiliary verb *do*, makes it agree with the subject, and constrains the `VP` to be infinitive.

In Figure 3, the notation `S::S` indicates that a sentence (dominated by an `S` node) on the source side translates into another sentence in the TL. The constituent sequences are `[NP VP MA] -> [AUX NP VP]`. The remaining portion of the transfer rule specifies alignments and constraints. Following a basic unification-based approach, we assume that each constituent structure node may have a corresponding feature structure. The feature structure holds syntactic features (such as number, person, and tense) so that they can be checked for agreement and other constraints. The feature structure may also specify the grammatical relations that hold between the nodes.

The feature unification equations used in the rules follow the formalism from the Generalized LR Parser/Compiler (Tomita, 1988). The *x* and *y* variables used in the alignments and constraints correspond to the feature structures of the elements of the constituent sequences. `x0` is the feature structure of the SL parent node (`S` in Figure 3). `y0` is the feature structure of the TL parent node (`S` in Figure 3). The source constituents are referred to with `x` followed by the constituent index, starting at `1`. Thus, `NP` is referenced as `x1`, `VP` as `x2`, and `MA` is referenced as `x3`. On the target side the constituent sequence order is `AUX NP VP`, referred to with `y1`, `y2`, and `y3`, respectively.

The body of the rule begins with one or more constituent alignments that describe which constituents on the source side should transfer to which constituents on the target side. Not all constituents need to have an alignment to the other side. A double-quote-enclosed string may also be used on the target side, directing the transfer engine always to insert this string into the translation in relative position to the rule's other target constituents.

The constituent alignments are followed by a set of feature unification constraints, to be used in each of the analysis, transfer, and generation stages. Transfer unification equations are generally divided into three types: *x*-side constraints, which refer to SL constituents and are used during analysis; *xy*-constraints, used selectively during transfer to move feature structures from the SL to the TL structures; and *y*-side constraints, used during generation to distribute TL feature structures and to check agreement constraints between target constituents.

In a further distinction, *x*-side and *y*-side constraints can be either "value constraints" or "agreement constraints". Value constraints are of the format (X_n *attribute*) = *value*)) or (Y_n *attribute*) = *value*)), while agreement constraints are of the format ((X_n *attribute*) = (Y_m *attribute*)). Agreement constraints can also be between two *x*-indices (*xx*-constraints) or two *y*-indices (*yy*-constraints).

An example of a value constraint is `((X2 AGR) = *3PLU)`, while an agreement constraint example is `((X2 AGR) = (X3 AGR))`. This distinction is used during learning. Initial seed hypotheses contain value constraints, which are then generalized to agreement constraints where possible.

Since the transfer rules are comprehensive (in the sense that they each incorporate parsing, transfer, and generation information), the transfer engine uses them exclusively to translate; there are no additional parsing and generation grammars. Hence, the transfer rules contain large amounts of information, including feature information about the SL and TL constituents.

In many systems, transfer rules contain only the mapping (what we call the alignment) between the SL and TL constituents. This is the obvious choice if an additional parsing and an additional generation grammar are available. Our system, however, learns the parsing, transfer, and generation grammars at once, encoding all the necessary information in each rule. There are two main reasons for this. First, the rules are learned from bilingual sentence pairs from which all of the information is derivable (SL constituents and constraints, TL constituents and constraints, source–target alignments, and source–target feature transfers). The second reason is the discrepancy between the amount of information available for the major and minority languages. If less information is available for one language, more assumptions are made using the other language. For instance, if no number information is available for the minority language, we assume that it is the same as in the major language. Our transfer rules are thus specifically designed for the circumstances as well as for the learning task at hand.

The fact that each transfer rule contains parsing, transfer, and generation information should however not be confused with the way translation is implemented. The transfer engine does indeed first parse the sentence, then transfer and generation are achieved in an integrated pass. In other words, while each transfer rule jointly represents parsing, transfer, and generation constraints, the transfer engine keeps a strict distinction between them (Peterson, 2002).

5. Rule-Learning Module

In this section we introduce the rule-learning system for learning syntactic transfer rules from elicited examples. Commonly, rule-based systems are developed by human experts. Our goal is to automate this process. It is important to distinguish between the training stage and the run-time translation system. Training is always done with the major language (such as English) being the SL (*x*-side) and the minority language being the TL (*y*-side). Run-time translation can then occur in either direction. In this paper we report results translating from English into the minority language. As was mentioned above, this means that no statistical decoder is used. However, the rules can be reversed and then be used to translate in the other direction, as reported in Lavie et al. (forthcoming).

Learning from elicited data proceeds in three stages: the first phase, "seed generation", produces initial versions of transfer rules. The rules that result from "seed generation" are "flat" – they transfer linear sequences of parts of speech and do not contain any non-terminal daughter nodes. For this purpose, the second phase "compositionality learning", adds structure using previously learned rules. For instance, in order to learn a rule for a `PP`, it is useful to recognize the `NP` that is contained in it, and to reuse `NP` rules that were learned previously. The third stage of learning, SVSL, attempts to increase the generality of the rules. The rules that are learned during seed generation and structuralized during compositionality learning are specific to the sentence pair from which they were derived. In order to make the rules applicable to a wider variety of unseen examples at run-time, SVSL attempts to generalize them automatically. The following sections explain all three stages in detail.

5.1. SEED GENERATION

The first part of the automated learning process is seed generation – the production of preliminary transfer rules that will be refined during SVSL. The seed generation algorithm takes as input a pair of parallel, word-aligned sentences. The SL (major language) sentences are parsed and disambiguated in advance, so that the system has access to a correct parse in the form of a feature structure (f-structure) and a phrase or constituent (c-)structure for each source sentence.

The seed generation algorithm can proceed in different settings depending on how much information is available about the TL. In one extreme case, we assume that we have a fully inflected TL dictionary, complete with feature-value annotations. The other extreme is the absence of any information, in which case the system makes assumptions about what linguistic feature values can be assumed to transfer from the SL to the TL. In such a case we assume, for example, that a plural noun will be translated as a plural noun.

With the information available, we can construct a transfer rule of the format described in Section 4. All of the *x*-side (major language) information can be gathered directly from the English (or other major language) parse. This includes part-of-speech information, *x*-side constraints, and type information. However, the system also needs to construct constraints for the minority TL, the *y*-side. By default, the TL part-of-speech sequence is determined by assuming that English words transfer into words of the same part of speech on the target side. If more information is available about the TL, e.g., a dictionary with part-of-speech information, this information overrides the assumption that parts of speech transfer directly.

For *y*-side constraints, we make a similar assumption as with parts of speech: in absence of any information, we assume that linguistic features and their values transfer from SL words to the corresponding TL words. However, we are more selective about this type of projection. Some features, like gender (e.g., masculine, feminine, neuter) and noun class (e.g., Bantu nominal classifiers or Japanese numerical classifiers), will not generally have the same values in different languages. The Japanese classification of a window as a sheet, like paper, will not be relevant to Swahili, which puts it in class 5, or to French, which classifies it as feminine. Other features may tend to have the same values across languages. For example, if a language has plural nouns in common usage, it will probably use one in the translation of *The children were playing*, because using a singular noun (*child*) would change the meaning. The decision of which features to transfer is necessarily based on heuristics that are less than perfect. For example, we will transfer past tense for a sentence like *The rock fell* because it is part of the meaning, even though we are aware that tense systems vary widely and that English past tense will not always correspond to past tense in another language.

If an inflected TL dictionary is available, the *y*-side constraints are determined by extracting all the information about the given TL words from the dictionary. If there is more than one entry for a word in the target dictionary, it is disambiguated during transfer or generation when *xy*- or *y*-side constraints are evaluated. For example, the definite article *der* in German can be either masculine nominative or else feminine dative, so that the appropriate entry can only be determined by considering the other words in the sentence.

Lastly, no *xy*-constraints are constructed during seed generation. This arises from the fact that the seed rules are quite specific to the sentence pair that they are produced from. In some cases, a *y*-side constraint could equivalently also be expressed as an *xy*-constraint. The difference does, however, become apparent during SVSL. From a procedural point of view, *y*-side constraints are more specific than the equivalent *xy*-constraints. For instance, the constraints `((x1 num) = pl)` and `((y2 num) = pl)` are, on the surface, equivalent to `((x1 num) = pl) ((y2 num) = (x1 num))`. There is, however, a difference: in the first case, changing `((x1 num) = pl))` to `((x1 num) = (*OR* sg pl))` does not have any impact on the value of `y2`'s number, but in the second case it does. As SVSL manipulates constraints in such a fashion, there is actually a difference between the two forms. As the goal is to start SVSL at a very specific level, we produce *y*-side constraints rather than *xy*-constraints. An example of a seed rule can be found in the leftmost column in Table I below.

After a seed rule is produced, it is added to a tentative translation grammar. Using this rule, the transfer engine then tries to translate the bilingual sentence pair that the rule was derived from. Only if the rule can be used to translate the pair correctly (if the rule "covers" the sentence pair) is the rule acceptable and is passed on to further learning steps.

5.2. COMPOSITIONALITY

In order to scale to more complex examples, the system next learns compositional rules with non-terminal daughter constituents. For example, when producing a rule for a sentence, we can make use of noun phrase and verb phrase subconstituent transfer rules that have already been learned.

Compositional rules are learned by first producing a flat rule in seed generation. Then the system traverses the c-structure parse of the SL sentence. Each node in the parse is annotated with a label such as `NP`, which is the root of a subtree. The system then checks if it has learned a lower-level transfer rule that can be used to correctly translate the words in the SL subtree.

The left-most column in Table I contains the flat seed rule for the bilingual sentence pair (7).

(7) The highly qualified applicant did not accept the offer.
Der äußerst qualifizierte Bewerber nahm das Angebot nicht an.
THE HIGHEST QUALIFIED APPLICANT TOOK THE OFFER NOT ON

In order to add structure to the flat rule, the system performs a top-down traversal of the English parse tree. For each node, such as `S`, `NP`, `NBAR`, etc., it extracts the part

Table I. Example of compositionality.

Uncompositional rule	Lower-level rule	Compositional rule
;;SL: *The highly qualified applicant did not accept the offer.*		
;;TL: *Der äußerst qualifizierte Bewerber nahm das Angebot nicht an.*		
S::S	NP::NP	S::S
[det adv adj n aux neg v det n]	[det ADJP n]	[NP aux neg v det n]
[det adv adj n v det n neg vpart]	[det ADJP n]	[NP v det n neg vpart]
;;alignments:	;;alignments:	;;alignments:
(x1::y1)	(x1::y1)	(x1::y1)
(x2::y2)	(x2::y2)	(x3::y5)
(x3::y3)	(x3::y3)	(x4::y2)
(x4::y4)	(x4::y4)	(x4::y6)
(x6::y8)		(x5::y3)
(x7::y5)		(x6::y4)
(x7::y9)		
(x8::y6)		
(x9::y7)		
;;x-side constraints:	;;x-side constraints:	;;x-side constraints:
((x1 def) = *+)	((x3 agr) = *3-sing)	((x2 tense) = *past)
((x4 agr) = *3-sing)		
((x5 tense) = *past)		
;;y-side constraints:	;;y-side constraints:	;;y-side constraints:
((y1 def) = *+)	((y3 agr) = *3-sing)	((y1 def) = *+)
((y3 case) = *nom)		((y1 case) = *nom)
((y4 agr) = *3-sing)		

of the sentence that is rooted at this node. For example, in the part of a c-structure tree in (8), the top node `NP` covers the chunk *the highly qualified applicant*.

(8)
```
(<NP> (<DET> ((ROOT *THE)) THE)
(<NBAR>
(<ADJP> (<ADVP> (<ADV> ((ROOT *HIGHLY)) HIGHLY))
(<ADJP> (<ADJ> ((ROOT *QUALIFIED)) QUALIFIED)))
(<NBAR> (<N> ((ROOT *APPLICANT)) APPLICANT))))
```

At this point, the question is if there already exists a learned transfer rule that can correctly translate this chunk. Before this can be determined, the system first must do the following.

(a) Find the TL reference translation that corresponds to the SL chunk. For this information, the system consults the user-specified alignments and the given TL translation.

(b) Find the category of the rule. This is obtained simply from the label of the top node of the sentence chunk. In example (8) the category would be `NP`. Compositionality is only added to the rule if a previously learned rule *of the desired type* (here `NP`) can account for the bilingual chunk. This is done in order to preserve the integrity of the original c-structure.

The next step is to call the transfer engine to translate the chunk *the highly qualified applicant*. The transfer engine returns zero or more translations together with the f-structures that are associated with each translation. The f-structures are stored as constraints of the same format as the *y*-side constraints in the transfer rules. In case there exists a transfer rule that can translate the sentence chunk in question, the system takes note of this and traverses the rest of the c-structure, excluding the subtree that corresponds to the chunk. The goal of traversing the c-structure is to find the *maximal* subtrees that can be translated by lower-level rules, hence the top-down traversal.

After traversal is completed and one or more compositional subconstituents have been found, the initial flat seed rule must be revised. Table I presents an example of how the flat rule is modified to reflect the existence of an `NP` rule that can translate the chunk *the highly qualified applicant*. The flat transfer rule (leftmost column) and the lower-level rule (middle column) are combined to form the structured rule that can be seen in the rightmost column. Several adjustments are made to the flat seed rule: First, the part-of-speech sequence from the flat rule is turned into a constituent sequence on both the SL and the TL sides, where those chunks that are translatable by lower-level rules are represented by the category information of the lower-level rule, in this case `NP`. The alignments are adjusted to the new sequences. Lastly, the constraints must be changed. The *x*-side constraints are mostly retained (with the indices adjusted to the new sequences). However, those constraints that pertain to the sentence/phrase part that is accounted for by the lower-level rule are eliminated, as can be seen in Table I. Note that for the rules in the two rightmost colums no specific sentences can be given as the rules are compositional and thus must have been learned from a combination of at least two training examples.

Finally, the *y*-side constraints are adjusted. For each new subconstituent that was detected, the transfer engine may return multiple translations, some of which are correct and some of which are wrong. (Correctness is checked against the original sentence pair.) The compositionality module compares the f-structures (constraints) of the correct translation and the incorrect translations. This is done so as to determine what constraints need to be added to the higher-level rule in order to produce the correct translation in context. For each constraint in the correct translation, the system checks if this constraint appears in *all* other translations. If this is not the case, a new constraint is constructed and inserted into the compositional rule. Before simply inserting the constraint, however, the indices need to be adjusted to the higher-level constituent sequence.

5.3. SEEDED VERSION SPACE LEARNING (SVSL)

SVSL follows the construction of compositional rules. The learning goal is to transform the set of seed rules into more general transfer rules. This transformation is done by merging rules, which in effect represents a generalization. Each rule is associated with a set of training sentences that it "covers", i.e., is able to translate correctly. This is called the CSet of a rule. After seed generation, each transfer rule's CSet contains exactly one element, namely the index of the sentence that it was produced from. When two rules are merged into one, their CSets are also merged.

The algorithm can be divided into two steps: the first step groups sets of similar seed rules, where each group effectively defines a separate version space. The second step is then run on each version space separately. It involves exploring the search space that is delimited by a set of rules.

5.3.1. *Grouping Similar Rules*

In order to minimize the search space, the learning algorithm should only merge two rules if there is a chance that the resulting rule will correctly cover the CSets of both unmerged rules. We group together rules that have the same (a) type information, (b) constituent sequences, and (c) alignments. The rules in a group may have different feature constraints, for example some may have singular nouns and some may have plural nouns. Mergers are only attempted within a group, thus simplifying the merging process.

From a theoretical standpoint, each such group of rules effectively defines a version space. Each version space has a boundary of specificness (the specific boundary) marked by the seed rules, and a boundary of generality (the general boundary), marked by a virtual rule with the same part-of-speech sequences, alignments, and type information as the other rules in the group, but no constraints. The goal of version space learning is to pinpoint the right level of generality between the specific boundary and the general boundary.

5.3.2. *Exploring the search space*

Our SVSL algorithm is based loosely on Mitchell (1982). We view the space of transfer rules as organized in a partial ordering, where some rules are strictly more general than others, but not all rules can be compared directly. The partial order of rules is based on the partial order of f-structures that satisfy the constraints in the rule (Shieber, 1986). The goal of the SVSL is to adjust the generality of rules. If the process is successful, the resulting rules should be much like those a human grammar-writer would design.

```
Transfer Rule 1:        Transfer Rule 2:
...                     ...
((X1 AGR) = *3SING)     ((X1 AGR) = *3SING)
((X2 DEF) = *DEF)
...                     ...
```

Figure 4. Example of comparable transfer rules.

At the moment, we view the rules that the seed generation algorithm produces as the most specific possible rules. However, the seed rules already present a generalization over sentences, generalizing to the part of speech or constituent level. It can happen that words that are of the same part of speech behave differently. For instance, some French adjectives appear before, others after the noun they modify. In these cases, the algorithm will have to explore the space "below" the seed rules. The issue is left for future investigation.

The major goal of the seeded version space algorithm is then to generalize over specific feature values if this is possible. Consider the examples in Figure 4.

Here, the second transfer rule is more general than the first, as the first has more restrictive constraints. We shall now formalize the notion of partial ordering of transfer rules in our framework.

Definition. A transfer rule tr_1 is strictly more general than another transfer rule tr_2 if all f-structures that are satisfied by tr_2 are also satisfied by tr_1. The two rules are equivalent if and only if all f-structures that are satisfied by tr_1 are also satisfied by tr_2.

Based on this definition, we can define operations that will turn a transfer rule tr_1 into a strictly more general transfer rule tr_2. More precisely, we have defined three operations:

1. Deletion of value constraint
   $((X_n\ \mathit{attribute}) = \mathit{value}))$ or $((Y_n\ \mathit{attribute}) = \mathit{value})) \rightarrow \mathit{NULL}$
2. Deletion of agreement constraint
   $((X_n\ \mathit{attribute}) = (Y_m\ \mathit{attribute})) \rightarrow \mathit{NULL}$
   (similarly for an *xx*-agreement constraint or a *yy*-agreement constraint.)
3. Merging of two value constraints into one agreement constraint. Two value constraints can be merged if they are of the following format:
   $(X_n\ \mathit{attribute}) = \mathit{value}1))$
   $(X_m\ \mathit{attribute}) = \mathit{value}2))$
   $((X_n\ \mathit{attribute}) = (X_m\ \mathit{attribute}))$
   (similarly for an *x*- and a *y*-constraint or two *y*-constraints.)
   For example, the two value constraints `((X1 AGR) = *3PLU)` and `((X2 AGR) = *3PLU)` can be generalized to an agreement constraint of the form `((X1 AGR) = (X2 AGR))`.

Table II. Example of version space generalization.

Seed rule 1	Seed rule 2	Generalized rule
S::S	S::S	S::S
[NP aux neg v det n] →	[NP aux neg v det n] →	[NP aux neg v det n] →
[NP v det n neg vpart]	[NP v det n neg vpart]	[NP n det n neg vpart]
;;alignments:	;;alignments:	;;alignments:
(x1::y1)	(x1::y1)	(x1::y1)
(x3::y5)	(x3::y5)	(x3::y5)
(x4::y2)	(x4::y2)	(x4::y2)
(x4::y6)	(x4::y6)	(x4::y6)
(x5::y3)	(x5::y3)	(x5::y3)
(x6::y4)	(x6::y4)	(x6::y4)
;;constraints:	;;constraints:	;;constraints:
...	...	...
((x2 tense) = *past)	((x2 tense) = *past)	((x2 tense) = *past)
((y1 def) = *+)	((y1 def) = *+)	((y1 def) = *+)
((y1 case) = *nom)	((y1 case) = *nom)	((y1 case) = *nom)
...	...	...
((y4 agr) = *3-sing)	((y4 agr) = *3-plu)	((y4 agr) = (y3 agr))

5.3.3. *Merging Two Transfer Rules*

At the heart of the learning algorithm is the merging of two rules. Generalization is achieved by merging, which in turn is done based on the three generalization operations defined above. Suppose we wish to merge two rules tr_1 and tr_2 to produce the most specific generalization of the two, stored in tr_3. The algorithm proceeds in three steps:

1. Insert all constraints that appear in both tr_1 and tr_2 into tr_3 and subsequently eliminate them from tr_1 and tr_2.
2. Consider tr_1 and tr_2 separately. Perform all instances of operation 3 (see previous section) that are possible.
3. Repeat step 1.

After performing the merger, the system uses the new transfer rule, tr_3, to translate the CSets of tr_1 and tr_2. The merged rule, tr_3, is accepted if it can correctly translate all of the items in both CSets. In the example in Table II, the system can verify that this is the case. Given the way the partial order is defined, this sequence of steps ensures that tr_3 does not have any constraints that would violate this partial order, and that it lacks no constraints. To return to the example in Table II, the result of a merging operation can be seen. The first two columns contain the unmerged rules, the last column the merged rule.

5.3.4. *The Learning Algorithm and Evaluation of Merged Rules*

The ultimate goal of the learning algorithm is to maximize coverage of a rule set with respect to the language pair, meaning that a rule set can correctly translate as many unseen sentences as possible. Currently, we estimate the coverage of a rule set by the generality of the rules. Since mergers always provide a generalization of the two unmerged rules, we assume that they will increase coverage with respect to the language pair. Therefore, the learning algorithm seeks to minimize the size of the rule set. It iteratively merges two rules, until no more acceptable mergers can be found and the rule set is as general as possible. Note again that a merger is only acceptable if there is no loss in coverage over the more specific rules, and if it produces no incorrect translations. When no more mergers are found, the final rule set is output.

6. Preliminary Experimental Results

The above algorithms have been implemented and run on a corpus of 141 noun phrases and sentences. The training set consists of structurally relatively simple noun phrases and short sentences. Two examples from the training set are *the very tall man* and *The family ate dinner*. The training corpus was translated into German and word-aligned by hand. A fully inflected TL dictionary as described above was used. No other TL information was given. This section presents preliminary results that indicate that seed generation, compositionality, and SVSL can indeed effectively be used to infer transfer rules.

We report results for both training set accuracy (i.e., the system was trained on all 141 examples and evaluated on all of them), and for each split of a 10-fold cross-validation. To evaluate translation accuracy we compared the resulting translations against the reference translation. If the grammar could be used to produce the reference translation exactly, the sentence counted as correctly translated. Since these results are used for purposes of future research and development, no partial matching was done. Note that this is a very harsh evaluation metric – it is well known that a sentence can be correctly translated into many different surface forms in the TL. For this reason, MT evaluation generally involves user studies or matches partial outputs to reference translations. In this test, our system was graded on a harder scale, while in future evaluations, we will consider other metrics as well.

Further, our system considers a sentence as correctly translated if one of the possible translations matches the reference translation. This set accuracy measure should be judged in conjunction with the number of possible translations per sentence (see Table III). For instance, if the grammar can produce the reference translation, but only as one of 100 possible translations, then the system may still perform poorly, as it may not pick as its top translation the perfect translation. The current version of the transfer engine does not rank outputs when no statistical decoder is used. Therefore, it is important to report the average number of possible

Table III. Translation set accuracy and number of possible translations for the training set as well as 10 cross-validation sets. Translation accuracy is defined as the number of output sentences that are strictly correct divided by the number of input sentences. Average number of translations is defined as the number of proposed output sentences divided by the number of input sentences.

Evaluation set	Translation accuracy	Average number of possible translations
Training	0.716	4.936
TestAve	0.625	4.460
CVSplit1	0.400	2.533
CVSplit2	0.714	7.000
CVSplit3	0.714	3.286
CVSplit4	0.643	6.643
CVSplit5	0.714	2.357
CVSplit6	0.786	4.857
CVSplit7	0.786	3.214
CVSplit8	0.500	5.643
CVSplit9	0.429	2.929
CVSplit10	0.571	6.143

translations per rule in order to put into perspective the translation (set) accuracy. The results of this evaluation can be seen in Table III.

Because of the simplicity and small size of the training set, it is clear that much work remains to be done, both in refining the algorithms and making them scale to more complex examples and to bigger sets of data. The initial results, however, are promising and suggest that future exploration will very much be worthwhile.

7. Conclusions and Future Work

We have presented a novel approach to learning syntactic transfer rules for MT. We emphasize again that the learning algorithms are designed to work with naturally occurring or controlled corpora. However, learning is facilitated in a variety of ways when the translated and aligned elicitation corpus is available, for example because we can create relatively noise-free data on the major language side.

Our long-term plan is to expand the existing elicitation corpus to have reasonable coverage of linguistic phenomena that occur across languages and language families. In practice, we first have to focus on the more common features, so that the informant's time is used as efficiently as possible and a preliminary rough translation can be produced. It will be interesting to examine the effectiveness of an elicitation corpus for languages that are quite different from the elicitation languages English and Spanish. Aside from core MT research, we expect to gain insights on the approach of emulating a field linguist with a carefully controlled corpus.

The elicitation corpus as well as the learning system are continually being expanded to handle more structures and more structurally complex sentences. Our current research includes navigation and control of the elicitation process, automatic learning of morphological rules, and refining rules semi-automatically through interaction with informants.

Acknowledgements

The authors would like to thank Ralf Brown, Ariadna Font Llitjós, and Christian Monson for their support and their useful comments.

Notes

[1] The work described here is conducted at Carnegie Mellon University as part of the AVENUE project (NSF grant number IIS-0121-631) and the MilliRADD project (DARPA TIDES program).

[2] http://www.iiit.net/ltrc/morph/index.htm

References

Bouquiaux, Luc and Jacqueline M. C. Thomas: 1992, *Studying and Describing Unwritten Languages*, Dallas, TX: The Summer Institute of Linguistics.

Brown, Peter F., John Cocke, Stephen A. Della Pietra, Vincent J. Della Pietra, Fredrick Jelinek, John D. Lafferty, Robert L. Mercer and Paul S. Roossin: 1990, 'A Statistical Approach to Machine Translation', *Computational Linguistics* **16**, 79–85.

Brown, Peter F., Stephen A. Della Pietra, Vincent J. Della Pietra and Robert L. Mercer: 1991, 'The Mathematics of Statistical Machine Translation: Parameter Estimation', *Computational Linguistics* **19**, 263–311.

Brown, Ralf D.: 1997, 'Automated Dictionary Extraction for "Knowledge-Free" Example-Based Translation', in *Proceedings of the 7th International Conference on Theoretical and Methodological Issues in Machine Translation*, Santa Fe, New Mexico, pp. 111–118.

Comrie, Bernard: 1989, *Language Universals and Linguistic Typology*, 2nd edn, Oxford: Blackwell.

Comrie, Bernard and Noah Smith: 1977, 'Lingua Descriptive Series: Questionnaire', *Lingua* **42**, 1–72.

Dorr, Bonnie Jean: 1992, *Machine Translation: A View from the Lexicon*, Cambridge, MA: MIT Press.

Fillmore, Charles J., Paul Kay and Mary Catherine O'Connor: 1988, 'Regularity and Idiomaticity in Grammatical Constructions: The Case of Let Alone', *Language* **64**, 501–538.

Gaussier, Éric: 1999, 'Unsupervised Learning of Derivational Morphology from Inflectional Lexicons', in *Proceedings of the Workshop on Unsupervised Learning in Natural Language Processing at the 37th Annual Meeting of the Association for Computational Linguistics*, College Park, Maryland, pp. 24–30.

Goldsmith, John: 2001, 'Unsupervised Learning of the Morphology of a Natural Language', *Computational Linguistics* **27**, 153–198.

Greenberg, Joseph H.: 1966, *Universals of Language*, 2nd edn, Cambridge, MA: MIT Press.

Habash, Nizar and Bonnie Dorr: 2002, 'Handling Translation Divergences: Combining Statistical and Symbolic Techniques in Generation-Heavy Machine Translation', in Stephen D. Richardson (ed.) *Machine Translation: From Research to Real Users, 5th Conference of the Association for Machine Translation in the Americas, AMTA 2002*, Berlin: Springer, pp. 84–93.

Hutchins, W. John and Harold L. Somers: 1992, *An Introduction to Machine Translation*, London: Academic Press.

ILASH (Institute for Language, Speech and Hearing): 2002, *Half-day workshop on Minority Languages and Computation*, Sheffield, `www.dcs.shef.ac.uk/research/ilash/Meetings/ML.html`.

Jones, Douglas and Rick Havrilla: 1998, ‘Twisted Pair Grammar: Support for Rapid Development of Machine Translation for Low Density Languages’, in David Farwell, Laurie Gerber and Eduard Hovy (eds) *Machine Translation and the Information Soup: Third Conference of the Association for Machine Translation in the Americas, AMTA’98 . . .* , Berlin: Springer, pp. 318–332.

Lavie, Alon, Stephan Vogel, Erik Peterson, Katharina Probst, Ariadna Font Llitjós, Rachel Reynolds, Jaime Carbonell and Richard Cohen: forthcoming, ‘Experiments with a Hindi-to-English Transfer-based MT System under a Miserly Data Scenario’, to appear in *ACM Transactions on Asian Language Information Processing*.

Levin, Lori, Donna Gates, Alon Lavie and Alex Waibel: 1998, ‘An Interlingua Based on Domain Actions for Machine Translation of Task-Oriented Dialogues’, in *Proceedings of the International Conference on Spoken Language Processing (ICSLP’98)*, Sydney, Australia, pp. 1155–1158.

Levin, Lori, Rodolfo Vega, Jaime Carbonell, Ralf Brown, Alon Lavie, Eliseo Cañulef and Carolina Huenchullan: 2002, ‘Data Collection and Language Technologies for Mapudungun’, in *International Workshop on Resources and Tools in Field Linguistics*, Las Palmas, Spain, pp. 18-1–18-4.

Melamed, I. Dan: 1998, *Manual Annotation of Translational Equivalence: The Blinker Project*, IRCS Technical Report 98-07, Institute for Research in Cognitive Science, University of Pennsylvania, Philadelphia, PA.

Mitchell, Tom: 1982, ‘Generalization as Search’, *Artificial Intelligence* **18**, 203–226.

Nirenburg, Sergei: 1998, ‘Project Boas: A Linguist in the Box as a Multi-Purpose Language’, in *Proceedings of the First International Conference on Language Resources and Evaluation*, Granada, Spain, pp. 739–746.

Och, Franz Josef and Hermann Ney: 2002, ‘Discriminative Training and Maximum Entropy Models for Statistical Machine Translation’, in *40th Anniversary Meeting of the Association for Computational Linguistics*, Philadelphia, PA, pp. 295–302.

Papineni, Kishore, Salim Roukos and Todd Ward: 1998, ‘Maximum Likelihood and Discriminative Training of Direct Translation Models’, in *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP ’98)*, Seattle, WA, pp. 189–192.

Peterson, Erik: 2002, *Adapting a Transfer Engine for Rapid Machine Translation Development*, Masters Research Paper, Georgetown University.

Probst, Katharina: 2003, ‘Using “smart” Bilingual Projection to Feature-tag a Monolingual Dictionary’, in *Seventh Conference on Natural Language Learning*, Edmonton, Canada, pp. 103–110.

Probst, Katharina, Ralf Brown, Jaime Carbonell, Alon Lavie, Lori Levin and Erik Peterson: 2001, ‘Design and Implementation of Controlled Elicitation for Machine Translation of Low-density Languages’, in *MT Summit VIII Workshop: MT 2010 – Towards a Road Map for MT*, Santiago de Compostela, Spain, pp. 44–49.

Probst, Katharina and Lori Levin: 2002, ‘Challenges in Automated Elicitation of a Controlled Bilingual Corpus’, in *Proceedings of the 9th International Conference on Theoretical and Methodological Issues in Machine Translation (TMI-2002)*, Keihanna, Japan, pp. 157–167.

Shieber, Stuart M.: 1986, *An Introduction to Unification-Based Approaches to Grammar*, Stanford, CA: Center for the Study of Language and Information.

Tomita, Masaru (ed.), Teruko Mitamura, Hiroyuki Musha and Marion Kee: 1988, *The Generalized LR Parser/Compiler Version 8.1: User's Guide*, Center for Machine Translation Technical Report, Carnegie Mellon University, Pittsburgh, PA.

Trujillo, Arturo: 1999, *Translation Engines: Techniques for Machine Translation*, Berlin: Springer.

Vogel, Stephan and Alicia Tribble: 2002, 'Improving Statistical Machine Translation for a Speech-to-Speech Translation Task', in *Proceedings of the Workshop on Speech-to-Speech Translation at the 7th International Conference on Spoken Language Processing (ICSLP 2002/Interspeech 2002)*, Boulder, CO, pp. 1901–1904.

Yamada, Kenji and Kevin Knight: 2001, 'A Syntax-based Statistical Translation Model', in *Association for Computational Linguistics 39th Annual Meeting and 10th Conference of the European Chapter*, Toulouse, France, pp. 523–530.