Elicitation-Based Machine Translation for Resource-Poor Languages

Katharina Probst, Lori Levin, Erik Peterson, Alon Lavie, Jaime Carbonell, Ariadna Font Llitjós
Language Technologies Institute
Carnegie Mellon University
Pittsburgh, PA, U.S.A.

January 9, 2003

Abstract. Elicitation-based Machine Translation is an approach to MT that can be applied to language pairs where large bilingual corpora as well as other NLP resources are not available. In our approach, a bilingual speaker translates and aligns a small elicitation corpus that is fed as training data into a rule learning system. We assume that the elicitation language is resource-rich, meaning that resources such as a grammar and a parser are available. The rule learning system then uses the syntactic analysis of the sentence in the resource-rich language, as well as any information given in the other language, to infer *seed* transfer rules. The seed rules are generalized in order to increase their coverage of unseen examples, using the bilingual elicitation corpus for validation. The learning mechanism is based on Version Space Learning with .... to insure convergence.....

Keywords: elicitation, machine learning, syntactic transfer rules

1. Introduction and Motivation

In recent years, much of machine translation research has focused on two issues: portability to new languages, and developing machine translation systems rapidly. Since human expertise on rare languages may be in short supply, and human development time can be lengthy, 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 (**?**; **?**; **?**; **?**). However, a minority of projects (**?**; **?**; **?**) have addressed automated learning of translation rules from a controlled corpus of carefully elicited sentences.

This article presents an elicitation-based approach to learning transfer rules for machine translation. The input to the rule learning mechanism is a parallel corpus of elicitation sentences, where each sentences is word-aligned (allowing multiple and zero alignments). The learning mechanism is a new machine learning technique called Seeded Version Space Learning (SVSL). SVSL is based on Version Space Learning (VSL) (**?**), which involves focusing in on a hypothesis in a space between specific and general hypotheses. In the case of translation rules, the specific hypotheses are rules about specific words or specific features (singular, past tense, etc.). The general hypotheses are about parts of speech (like N), phrase level categories (like NP) and agreements between elements (like adjectives and nouns). To make the VSL more tractable, we use a seed rule as a starting point. A seed rule is created automatically from a bilingual word-aligned phrase or sentence and a lexicon. Generalizations are hypothesized by comparing seed rules and are confirmed by checking the translations of the elicitation sentences.

The work described here is part of the AVENUE project at Carnegie Mellon University.[1] The overall goal of AVENUE is to research methods for machine translation that are suitable for languages with scarce electronic resources, and possibly also a scarcity of humans trained in linguistic analysis. AVENUE aims to develop "omnivorous" machine translation techniques that can take advantage of whatever resources are available. This includes using large unconstrained corpora when they are available, and resorting to methods such as elitiation-based SVSL when corpora are not available.

This article is divided into three main parts: First we present the transfer rule formalism and the run time system. The second part discusses how the elicited data is used to learn syntactic transfer rules. It also describes how the learning process is facilitated in each of its steps by the elicitation corpus, and includes a preliminary evaluation of the rule learning approach. . . . The third part discusses the strengths of our rule learning system in comparision to other corpus-based approaches to machine translation.

2. The Transfer Rule Formalism and Interpreter

The output of the SVSL system is a set of unification-based transfer rules for machine translation. This section describes the transfer rule formalism and transfer rule engine. Transfer engines typically have separate rules for the three traditional stages of MT: analysis, transfer, and generation. The AVENUE rule formalism combines the information for all three into a single rule, but uses it in several stages. The AVENUE engine consequently is closer to the 'modified' transfer approach used in the early METAL system as described by Hutchins (**?**).

Like most modern translations sytems, the AVENUE transfer engine maintains a strict separation between the transfer algorithms and linguistic data. The design of the transfer rule formalism itself was guided

[1] NSF grant number

by the consideration that the rules must be simple enough to be learned by an automatic process, but also powerful enough to allow manually-crafted rule additions and changes to improve the automatically learned rules.

The following list summarizes the components of a transfer rule. The reader is reminded that in the current state of the system, the source language is the resource-rich language (such as English), whereas the target language is the resource-poor language[2].

- **Type information:** This component identifies the type of the current transfer rule. Sentence rules are of type `S`, noun phrase rules of type `NP`, etc. The formalism also allows for type information on the target language side.

- **Part-of speech/constituent information:** These are the context-free production rules. The elements of the right hand side can be lexical categories, lexical items, and/or phrasal categories.

- **Alignments:** During elicitation, the informant provides an alignment of source and target language words. 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 target language. At run-time, y-side constraints serve to generate the TL sentence.

- **XY-constraints:** The xy-constraints provide information about what feature values will transfer from the source into the target language. Specific TL words can obtain feature values from the source language sentence.

For illustration purposes, Figure 1 shows an example of a transfer rule.

In the example rule, the production rules (constituent sequences) are `S::S : [NP PP VP] -> [VP NP PP]`. The first `S` refers to the left-hand side of the source production rule. The first `S` also represents the type of the rule. After the `::`, the second `S` refers to the type of target

[2] We have begun investigating the reversal of translation direction with some success. The results will be reported in a future publication.

```
S::S : [NP PP VP] -> [VP NP PP]
; Source and target production rules
; (‘constituent sequences’)
(
  ; Constituent alignments
(X1::Y2)
(X2::Y3)
(X3::Y1)

  ; Analysis unification equations (‘x-side constraints’)
  ((x0 subj) == x1)
  (x0 = x3)

  ; Transfer Equations (‘xy-constraints’)
  (y0 = x0)

  ; Feature completion and Generation Constraints
  ; (‘y-side constraints’)
  (y2 == (y0 subj))
  (y1 = y0)
  ((y0 agr) = (y1 agr))
  ((y1 agr) = (y2 agr))
)
```

Figure 1. Sample Transfer Rule

constituent the source constituent will be mapped to during transfer. Informally, this rule states that a source language `S` will be translated as a target language `S`. The sequence of source language constituents `NP PP VP` will be translated into the target language sequence ....

After the production rules, 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 ((**?**)). The `x` and `y` variables used in the alignments and constraints correspond to the feature structures of the elements of the production rules. `x0` is the feature structure of the source language parent node `S`. `y0` is the feature structure of the target language parent node `S`. The source constituents are referred to with x followed by the constituent index, starting at one. So `NP` is referenced as `x1`, `PP` as `x2`, and `VP` is referenced as `x3`. On the target side the production rule order is `VP PP NP`, 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 to always insert this exact word into the translation in relative position to the rule's other target constituents.

After the constituent alignments for the rule are 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 source language constituents and are used during analysis; xy-constraints, used during transfer to selectively move feature structures from the source to the target language structures; and y-side constraints, used during generation to distribute target language 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 $((\Phi_i\gamma_m\ \alpha_j) = \beta_{kj})$, while agreement constraints are of the format $((\Phi_i\gamma_m\ \alpha_j) = (\Phi_i\gamma_n\ \alpha_j))$, where

- $\Phi_i \in \{X, Y\}$
- $\gamma_i \in \begin{cases} \{0 \ldots number\ of\ SL\ words\} & if \Phi_i = X \\ \{0 \ldots number\ of\ TL\ words\} & if \Phi_i = Y \end{cases}$
- $\alpha_i \in \{linguistic\ features\ such\ as\ number,$ $definiteness,\ agreement,\ tense,\ \ldots\},$
- $\beta_{ki} \in \{x | x\ is\ a\ valid\ value\ of\ feature\ \alpha_i\}$

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

It is important to note that the learned transfer rules are comprehensive in the sense that they incorporate parsing, transfer, and generation information all in the rules. In other words, the set of rules that results from learning, represents a complete transfer grammar. The transfer engine uses it exclusively to translate, and no additional parsing and/or generation grammars are available. Hence, the transfer rules contain a myriad of information, including feature information about its constituents.

In many systems, transfer rules contain merely the mapping, i.e., alignment, between the source and target language 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. The first reason is that the rules are learned from bilingual sentence pairs, so that it makes intuitive sense to build a transfer rule for each sentence pair. The second reason is the discrepancy between the amount of information available on the source language and the target language sides. The less information is available in the TL, the more assumptions we make about a given sentence, using the SL information. For instance, if no number information is available for the TL, we assume that it is the same as in the SL. 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 achieved. 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 rules contains parsing, transfer, and generation constraints as if there was no difference between them, the transfer engine does distinguish between them very strictly.

3. Elicitation Interface

AVENUE's machine learning mechanism takes as input pairs of word-aligned sentences in the source and target languages. The Elicitation Interface collects the word and phrase alignment information necessary for automatic rule learning An informant using this interface is presented with a sentence in English or Spanish (or other resource rich languages) and is then asked to translate the sentence to the language of interest. In the interface is a context field, where additional information about the sentence (such as the gender of one of the participants) that may influence the translation can be included. After translating the sentence, for each word in the elicited sentence the informant then indicates corresponding word(s) in the eliciting language. In some cases, the word may not have a corresponding word in the matching sentence, or may corrspond to multiple words.

If the informant wants to include some information or clarifications about the translation or the alignment of the sentence pair, the interface provides comment field in which to write these comments. Once the informant has finished with one sentence pair, clicking on the "Next"

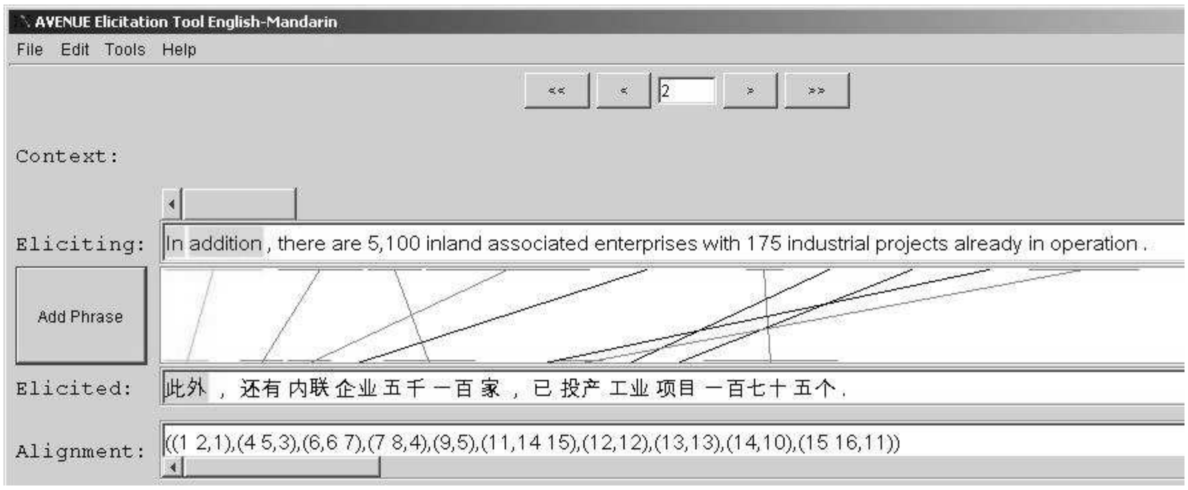


Figure 2. Elicitation Interface

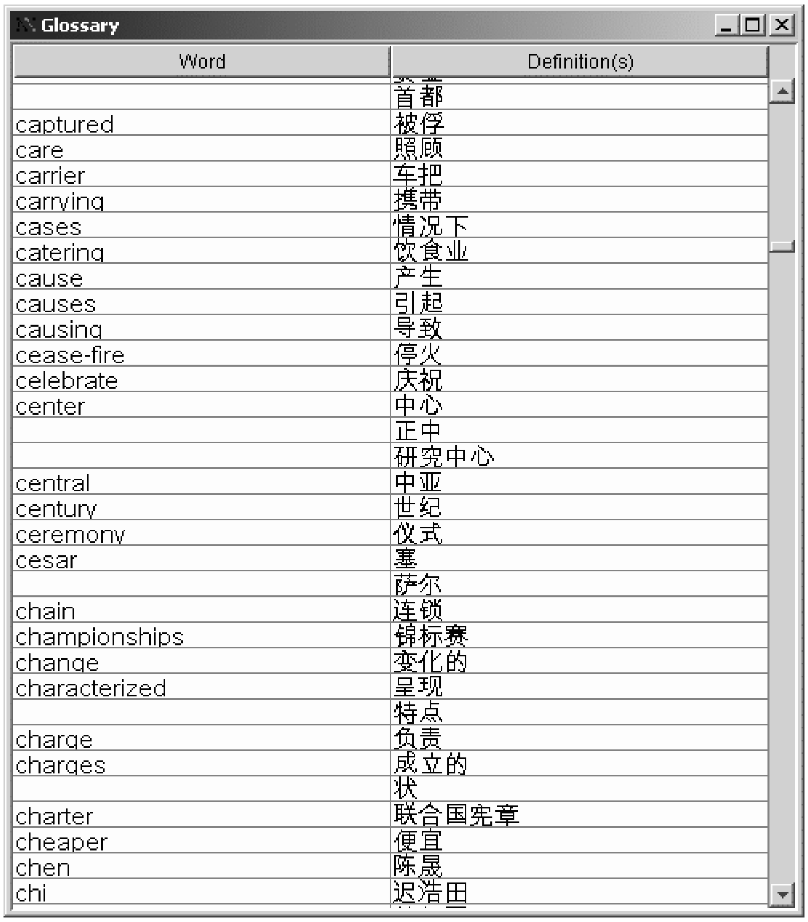
button will bring up the next sentence to translate and align. Users can also go back to change previous sentences or type in the index number for a particular sentence to jump directly to it.

Alignments are added 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 straight-forward: 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 possible.

Figure 3 shows an ..... (include alignments) ...

The interface was written in Java for portability and ease of language localization. The language used by the menus and buttons of the interface, as well as the help file, can be easily changed without recompiling the entire program. Currently, the interface has been localized for English and Spanish. Using Java makes displaying languages as varied as Arabic and Chinese as easy as having an appropriate font installed.

Beside the basic function of translating and word-aligning sentences, the elicitation interface also has several helpful 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



Glossary

Word	Definition(s)
	首都
captured	被俘
care	照顾
carrier	车把
carrying	携带
cases	情况下
catering	饮食业
cause	产生
causes	引起
causing	导致
cease-fire	停火
celebrate	庆祝
center	中心
	正中
	研究中心
central	中亚
century	世纪
ceremony	仪式
cesar	塞
	萨尔
chain	连锁
championships	锦标赛
change	变化的
characterized	呈现
	特点
charge	负责
charges	成立的
	状
charter	联合国宪章
cheaper	便宜
chen	陈晟
chi	迟浩田

Figure 3. Elicitation Interface Glossary

or elicited sentences. The search results include the sentence the word was found in, along with any aligned words. Clicking on a result goes directly to the sentence. This makes it easier for users to be consistent in their alignment work.

4. The Rule Learning System

In this section we introduce the rule learning system for learning syntactic transfer rules from elicitation examples. Commonly, rule-based systems are developed by human experts writing grammar rules that map structures from the source onto the target language. Our goal is to automate this process.

The system architecture (figure 4) is divided into a learning module and a run-time module. Within the learning module, data is collected and rules are learned from the bilingual data. The elicitation phase presents a bilingual user with a set of sentences (the elicitation corpus), who translates the corpus and aligns the words in each sentence pair (as described in Section **??**). The rule learning system takes as input the elicited and hand-aligned data, and infers transfer rules. During the learning process, each proposed rule is immediately validated, meaning that the transfer engine translates using the new rule. At the end of the learning process stands the learned transfer grammar, which is made available to the transfer engine for run-time translation. This means that, when presented with a sentence, the engine uses the grammar to produce the corresponding TL sentence.

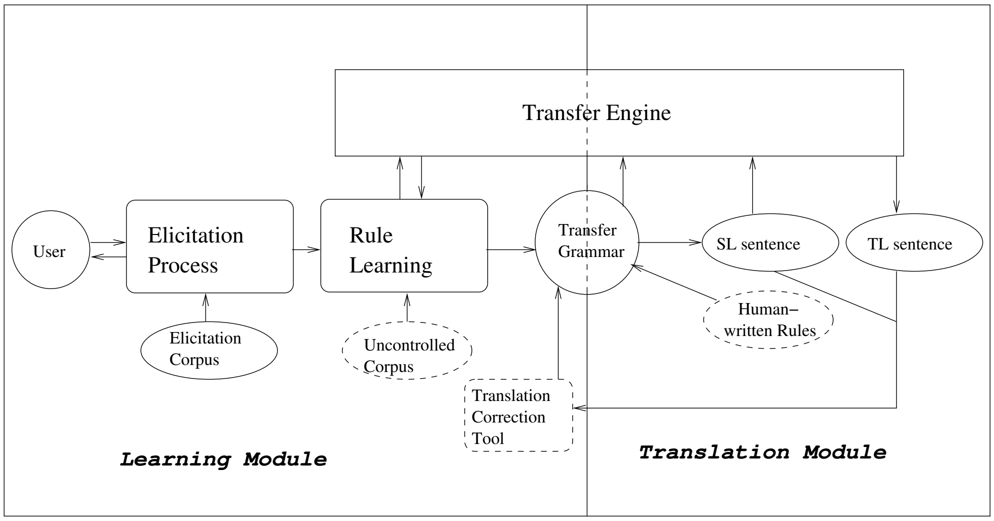


Figure 4. AVENUE's rule learning and translation system architecture

Figure 4 further shows two optional sources of information. During the learning phase, the rule learning system can also use an uncontrolled corpus. Further, the learned transfer grammar can at any point be supplemented by human-written rules. In the future, they will be refined using a tranlation correction tool (TCTool).

Learning from elicited data proceeds in three main stages: the first phase, Seed Generation, produces initial 'guesses' at transfer rules. The rules that result from Seed Generation are 'flat' - they transfer part of speech sequences. This is clearly not desirable for syntactic transfer rules. 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, Seeded Version Space Learning, adjusts the generality of the rules. In particular, the rules that are learned during Seed Generation and structuralized during Compositionality Learning are very specific to the sentence pair that they were derived from. In order to make the rules applicable to a wider variety of unseen examples at run-time, Seeded Version Space Learning makes the rules more general. The following sections will explain all three stages in detail. Furthermore, in each section we will detail how elicited and hand-aligned training data facilitates the learning process.

4.1. Seed Generation

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

The seed generation algorithm can proceed in different settings depending on how much information is available about the target language. In one extreme case, we assume that we have a fully inflected target language 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 into the target language. In such a case we assume, for example, that a plural noun will be translated as a plural noun.

With the information available, how can we construct a transfer rule of the format described in Section **??**? All of the x-side information can be gathered directly from the English f- and c-structures. This includes part of speech information, x-side constraints, and type information. The word alignment is given directly by the user. However, the system also needs to construct constraints for the target language, the y-side. By default, the target language 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 target language, 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 a Bantu language that classifies it as a manmade artifact, 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, and disambiguating and unfying if TL words have more than one entry in the target language dictionary. 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 Seeded Version Space Learning. From a procedural point of view, y-side constraints are more specific than the equivalent xy-constraints when input into the learning algorithm. 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) = (*OR* sg pl))` does not have any impact on the value of `y2`'s number, but in the second case it does. As Seeded Version Space Learning manipulates constraints in such a fashion, there is actually a difference between the two forms. As the goal is to start Seeded Version Space Learning at a specific level, it makes sense to produce y-side constraints rather than xy-constraints. Examples of seed rules can be found in the left-most and middle colums in table 4.3.

4.2. Seed Generation - Advantages of a Controlled Corpus

In the discussion of the seed generation algorithm, we have already touched on some advantages that a controlled corpus entails. In this section, we will make these advantages more explicit. As was said above, the seed generation algorithm relies heavily on the source language f-structure. This information is obtained from a parser and a parsing grammar in our system. It is helpful for the system to have access to a completely unambiguous and correct parse. Since in our case we can completely predict what source language sentences will need to be parsed (as well as disambiguated and verified), we can go through this process once during the creation of the corpus, then rely on this information for any target language. This makes the learning problem slightly easier, because no noise is introduced in the parsing stage.

A related advantage arising from a controlled corpus is that together with the unambiguous and verified parse, we can further store in the corpus the type information of each training example (e.g. NP, PP, S). Again, this can be done once and can then be reused for any target language, and it ensures that the type information given to the system is noise-free.

4.3. Compositionality

In order to scale to more complex examples, the system learns compositional rules: when producing a rule e.g. for a sentence, we can make use of sub-constituent transfer rules already 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 source language subtree. Consider the following example:

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

This c-structure represents the parse tree for the `NP` *the highly qualified applicant*. The system traverses the c-structure from the top in a depth-first fashion. For each node, such as `NP`, `NBAR`, etc., it extracts the part of the phrase that is rooted at this node. For example, the node `ADJP` covers the chunk *highly qualified*. The question is now if there already exists a learned transfer rule that can correctly translate this chunk. To determine what the reference translation should be, the system consults the user-specified alignments and the given target language translation of the training example. In this way, it extracts a bilingual sentence chunk. In our example the bilingual sentence chunk is *highly qualified – äußerst qualifizierte*. It also obtains its SL category, in this example `ADJP`.

The next step is to call the transfer engine to translate the chunk *highly qualified*. The transfer engine returns zero or more translations together with the f-structures that are associated with each translation. 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 chunk's subtree. After this step is completed, the original flat transfer rule is modified to reflect any compositionality that was found.

After a compositional sub-constituent is found, the initial flat seed rule must be revised. The rightmost column in 4.3 presents an example of how the flat rule is modified to reflect the existence of an `ADJP` rule that can translate the chunk *highly qualified*. The flat, uncompositional rule can be found in the middle column, whereas the lower-level `ADJP` rule can be found in the leftmost column. 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 `ADJP`. 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 4.3. Note that for the rules in the two rightmost colums no specific sentences can be given since 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 sub-constituent that was detected, the transfer engine may return multiple translations, some of which are correct and some of which are wrong. The compositionality module compares the f-structures 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.

Table I. Example of Compositionality

Lower-level rule	**Uncompositional 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))		

Before simply inserting the constraint, however, the indices need to be adjusted to the higher-level constituent sequence. Again, an example of this is in table 4.3.

4.4. Compositionality - Advantages of a Controlled Corpus

Like Seed Generation, Compositionality is facilitated by the existence of a controlled corpus that has been parsed and disambiguated off-line. As was shown in the algorithm above, the Compositionality procedure makes heavy use of the source language c-structure. Again, if this c-structure is available and noise-free, learning is more reliable.

4.5. Seeded Version Space Learning

Seeded Version Space Learning 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.

4.5.1. *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 1) type information, 2) constituent sequences, and 3) alignments. The rules in a group may have different feature constraints, for example some may have singular nouns and some may have plural nouns. Merges 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.

4.5.2. *Exploring the search space*

Our Seeded Version Space Learning algorithm is based loosely on (**?**). 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 goal of the Seeded Version Space Learning 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.

At the moment, we view the rules that the seed generation algorithm produced 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 following examples:

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

Here, the second tranfer 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
   $((\Phi_i \gamma_m\ \alpha_j) = \beta_{kj}) \rightarrow NULL$
2. Deletion of agreement constraint
   $((\Phi_i \gamma_m\ \alpha_j) = (\Phi_i \gamma_n\ \alpha_j)) \rightarrow NULL$
3. Merging of two value constraints into one agreement constraint. Two value constraints can be merged if they are of the following format:
   $((\Phi_i\ \alpha_k) = \beta_{kl})$
   $((\Phi_j\ \alpha_k) = \beta_{kl})$
   $\rightarrow ((\Phi_i\ \alpha_k) = (\Phi_j\ \alpha_k))$
   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)).

4.5.3. *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 section 4.5.2 above) that are possible.
3. Repeat step 1.

After performing the merge, 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 tranlate all of the items in both CSets. In the example in table **??**, the system can verify that this is the case.[3]

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 again to the example in table **??**, the result of a merging operation can be seen. The first two columns contain the unmerged rules, the last column the merged rule.

4.5.4. *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 merges 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 merges can be found and the rule set is as general as possible. Note again that a merge 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 merges are found, the final rule set is output. Cross-validation as an alternative evaluation metric is currently under investigation.

[3] Note that in the example, the source and target language sentences are not given with the rules. The reason again is that the rules are compositional, so that they must have been learned from at least two training examples. However, the example is given in reference to the previous example, where are rule was learned from *The highly qualified applicant did not accept the offer*, which translates into *Der äußerst qualifizierte Bewerber nahm das Angebot nicht an.*.

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)	((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)))

4.6. Preliminary Experimental Results

4.7. Seeded Version Space Learning - Advantages of a Controlled Corpus

The elicitation corpus is in many ways optimized for the Seeded Version Space Learning algorithm. It contains sets of 'similar' sentences, i.e. sentences of the same structure, but of different feature-value pairs. This ensures that the sentences can be grouped into version spaces, where each space will be sufficiently populated to produce and test hypotheses.

As the focus of this paper is elicitation-based machine translation, it is important to discuss the advantages and disadvantages of such an approach. Many data-driven approaches to machine learning have been proposed and explored, and since the perfect solution is as yet elusive, it makes sense to examine the strengths and weaknesses of the elicitation approach, albeit uncommon, thereby hopefully providing new insights to researchers in the field.

Learning from controlled elicitation is particularly useful in a resource-rich/resource-poor situation: for one of the languages in the translation pair, the resource-rich language, linguistic and language-technological tools are easily available. Such tools include unilingual dictionaries, part-of-speech taggers, morphological analyzers, parsers and grammars. For the other language, such resources are not available, or only to a limited extent. This situation is not uncommon: while extensive language technologies are available for languages such as English, Spanish, and Japanese, other languages such as Croatian, Bulgarian, or indigenous languages have been neglected. Thus, when translating between a resource-rich language such as English and a resource-poor language such as Mapudungun (an indigenous language spoken in Southern Chile and Argentina), dealing with the imbalance between the information available on each language side becomes the highest priority.

Most corpus-based MT methods require large bilingual training corpora. The corpora must be large because the programs are learning statistical distributions, and not linguistic generalizations. Statistical methods could use corpora on the order of ten million words in length. Recent experiments with one hundred thousand word training corpora (**?**) have shown that statistical methods may be possible with smaller corpora, but results are considerably worse than when larger corpora are used.

Our elicitation approach does not require the existence of large bilingual training corpora. The elicitation corpus is a monolingual corpus in the resource-rich language. (We currently have elicitation corpora in English and Spanish.) The corpus is organized linguistically with small phrases building up compositionally into larger phrases, and sentences designed to detect various kinds of inflections, agreements, and constraints in the resource-poor language. Ideally, all that would be required from the resource-poor language is a person who is literate in both the resource-rich and resource-poor languages. (Section **??** describes the current state of our research, which requires a literate bilingual and a lexicon with inflectional features in the resource-poor language.) By using carefully chosen phrases that cover basic sentence structures and syntactic constraints our system learns rules from less noisy data and we avoid the need for a corpus that is large enough to calculate statistical distributions. Also, because our corpus is organized according to linguistic principles, it learns transfer rules that are extendable by human system developers.

Our rule learning approach is equally applicable to situations where large amounts of bilingual training data are available. Optimally, the system will combine results from training on both an unstructured corpus if it is available and from a structured corpus. In this paper, we will focus merely on learning from a structured corpus, and explain why this elicitation-based approach is advantageous to our system.

One appeal of a large, unstructured training corpus is that no elicitation corpus needs to be designed. The design of a corpus that covers major typological phenomena is a non-trivial and time-consuming task. However, once the elicitation corpus is designed, it can easily be applied to new language pairs.

Learning from a large, unstructured training corpus has the further advantage that quantitative statements can be made. The elicitation corpus aims at covering all kinds of linguistic phenomena, while keeping the corpus small. Therefore, each single phenomenon does not receive attention beyond a few sentences. If one of the sentences provides an exception to a rule, the system will learn two different transfer rules, but it has no way of knowing which rule should be the default. In other words, the design of the elicitation corpus inherently violates the distribution of rules in a given language pair. If learning from a large corpus, the distribution is preserved, and can possibly be extracted and used by the learning algorithm.

However, there are also strong arguments in favor of a small, controlled corpus. Below, we will describe how transfer rules are learned in our system. It will also become clear that the availability of a structured corpus facilitates the learning problem in a variety of ways. The more is known about the structure and make-up of the training data, the more this information can be incorporated into rule learning. Roughly speaking, the difference between learning from a structured and an unstructured corpus corresponds to the difference between supervised and unsupervised learning. The more is known about the training data, the simpler the learning problem.

A big advantage of eliciting a small training corpus is the obvious time savings in the data collection process. A bilingual speaker will translate several thousand sentences which are carefully designed to provide a variety of linguistic phenomena. Therefore, the ratio between the size of the training corpus and the linguistic coverage of the system can be much smaller when learning from a structured training corpus.

Further, the elicitation corpus is designed to cover 'major' linguistic features and phenomena first, and to then move on to less common features. This will discussed in more detail below, but to give a brief example here, the corpus covers noun phrases and simple intransitive and transitive sentences first, whereas relative clauses are not elicited until later. This design implies that the time of the bilingual informant can be used most effectively: even if he or she only translates part of the corpus, the resulting learned rules cover common linguistic features and constructions. The more data is translated, the more sophisitcated the resulting transfer grammar will become.

Another way in which a controlled corpus can be useful is that it uses knowledge of language typology to determine what needs to be learned and what does not apply to a given language. For example, once we determine that a language does not mark plural number on nouns, we do not need to check whether it marks dual number. Conversely, if we know that a language encodes plural differently than dual, the system will not attempt to learn a transfer rule that fits both dual and plural. It will not consider it noise if the predicted transfer rule for a dual noun phrase does not match a transfer rule for plural that has already been learned. This process is called 'feature detection'.

To summarize, both learning from an unstructured and a structured corpus have their advantages. Our system will adapt to a given situation, depending simply on the availability of large amounts of sentence-aligned bitext versus the availability of a native informant who can translate the elicitation corpus. Ideally, of course, both types of training data can be combined in order to get the 'best of both worlds'. Currently, we have not yet developed sophisticated techniques to apply our rule learning algorithm to unstructured text. However, future versions of our system will provide this technology.

5. Principles of Elicitation

5.1. General Design of the Elicitation Corpus

As was mentioned above, the elicitation corpus is designed to cover major linguistic phenomena in typologically diverse languages. Currently, the source language is a major language such as English or Spanish. The target language can in principle be any language. In the early development stages, we have performed most of our experiments on German, but we are currently moving to other languages (Mapudungun, Chinese).

The elicitation corpus is a list of sentences in a major language like English or Spanish. The current pilot elicitation corpus has about 2000 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 (**?**), which includes around 6,000 to 10,000 sentences, and to ultimately cover most of the phenomena in the Comrie and Smith (**?**) checklist for descriptive grammars.

The elicitation corpus has three organizational properties: it is organized into minimal pairs; it is dynamic (the elicitation takes a different path depending on what has been found so far); and it is compositional (starting with smaller phrases and combining them into larger phrases).

A minimal pair can be defined as two sentences that differ only in one feature. Each sentence can be associated with a feature vector. Consider, for instance, the following three sentences:

The man saw the young girl.
The men saw the young girl.
The woman saw the young girl.

The first and the second sentence differ merely in one feature, namely the number of the subject. Thus, they represent a minimal pair. However, sentence two and three also form minimal pair. They differ only in the gender of the subject.

The reason that we decided to organize the corpus in minimal pairs is specific to our use of the elicited data. One component of the transfer rule learning is feature detection. If we define linguistic phenomena (such as number, gender, subject-verb agreement, etc.) as a set of features, we want to determine what subset of these features is instantiated in a given language. This is achieved by defining the features such that they can be tested using a minimal pair of sentences. For instance, if we want to find out whether target language verbs inflect depending on the number of the subject, we can use the two sentences above: sentence one and sentence two only differ in the number of their subject, while all other features are kept constant. Determining whether the target language verbs agree in number with their subjects is then reduced to comparing the verbs in the elicited target language versions of sentences 1 and 2.

However, we have to careful to make the opposite conjecture. If the two verbs do not differ, then this could be an indication that target language verbs do not agree with their subjects in number. However, it could also mean that the verb used for comparison represents an exception to a more common rule. We approach this problem in two ways. First, we never rule out the existence of a feature based on only one example (i.e., one minimal pair). Second, the results from feature detection are interpreted not as absolutes, but as tendencies, which guide our search through the space of possible transfer rules and give more preference to those rules that use features that were detected over those that use features that were not detected. However, the latter type of rule is not considered impossible in our algorithm, accounting for the imperfect results of feature detection. A more detailed discussion of our feature detection module can be found in (**?**).

5.2. Challenge Number 1: The bilingual informant

Before we can design a controlled corpus, we need to define clearly what the characteristics of a typical user of the system would be. Of high priority in our work is not to expect the bilingual informants to know and understand linguistic terminology. However, we must expect that the bilingual speaker has a good intuition about the complexity of language and knows that the elicited data will be analyzed automatically rather than by a human. For instance, if we present a user with a minimal pair of sentences, ideally the translation will again be a minimal pair. If no minimal pair translation is possible, this should be pointed out by the user or else detected automatically by the system.

In our system, an ideal user is furthermore able to make reliable grammaticality judgments via the Translation Correction Tool. The internal machine learning program that analyzes the elicited data strives to acquire transfer rules automatically. In general, machine learning programs yield better performance if they have the option of learning actively, 'asking questions'. In our system, 'asking questions' amounts to the system proposing a translation using a transfer rule that is not completely confirmed. The translation is then presented to the user for a grammaticality judgment. If the user can act as an oracle for the system, and, even more ideally, correct the translation to the closest possible fit, automated learning will be much facilitated.

In sum, we expect two main characteristics from the ideal users of the system: 1) that they be aware of the complexity of the learning problem and maximize consistency without compromising accuracy. If this is the case, the system will achieve higher accuracy in the learned transfer rules. 2) that they be able to make grammaticality judgments (given the correct English translation) and ideally to correct the proposed translation to the closest grammatical match.

5.3. Challenge number 2: Learning morphology

There are two paradigms by which lexical acquisition can be achieved, and it is possible to use them in conjunction. In the first paradigm, the lexicon is elicited in a separate step from the user. Lexical items are presented to the user in isolation and in simple sentences where we expect the lexical item to appear in its default (lexicon) form: for instance, in order to elicit the translation of a noun, the user is presented with it in the subject of a sentence. The second paradigm allows the system to acquire the lexicon 'on the fly'. Using the user-specified alignments, it stores the translation of each word when it appears in the context of a sentence. The advantage of the second approach is immediately obvious, as it saves user time. The problem, however, is that the word may not appear in its default form in the context of a sentence. For example, it may have case markings, proximity markings, etc. If we want to collect a fully inflected dictionary, this problem does not exist. However, if we want the lexicon entries to be in their default form, and then apply a morphology module, we want to elicit only the default forms.

In our system, we combine both approaches. This is achieved by eliciting words in isolation or in expected default position, but then also storing the translations of all new words that appear later in the corpus. This often results in having multiple translations for one word, where the translations are morphological variants of each other. It is a challenge for the system to determine when to use which form in generation. We are currently building a morphology learning system that will alleviate the problem of lexicon acquisition. Morphology is learned in an unsupervised manner, as has been done by other groups, e.g. (**?**) and (**?**).

In addition to using user elicited lexical items, we also use any word lists or lexicons that are already available in the target language. Sometimes, a linguist will have already created a partial lexicon for the target language in question, as is the case with Siona (**?**). Naturally, we make use of the pre-existing lexicons in our system.

5.4. Challenge number 3: Learning Grammatical features

There are two main challenges associated with eliciting grammatical features. The first is to achieve reasonably large coverage. It is clear that the number of grammatical features across languages is very large. Not only do we need to compile a list of features and their possible values across languages. We also need to find sentences that exhibit each of these features. The big advantage of compiling a target-language independent corpus is that while the task is time-intensive, it will not have to be redone when applied to a new language. We are growing the elicitation corpus based on research on typology and universals (e.g. (**?**; **?**)).

The second big challenge concerning grammatical features is that the source language (so far, English) may not exhibit a grammatical feature that does exist in the target language. If we want examine the grammatical feature in the target language by merely asking for translations of well-chosen sentences, we are faced with the challenge of finding sentences (or minimal pairs of sentences) that achieve that. It is quite possible to elicit features that do not exist in the source language, but may exist in the target language. For example, consider the feature dual. English does not exhibit dual. However, if we want to know whether another language has dual, we present the following minimal pair of sentences:

Two men ran across the street.

Three men ran across the street.

If the translation of men differs in the target language, we have an indication that the target language marks nouns for dual. Not all grammatical features are as straightforward to elicit. However, research has been conducted on how to elicit information from users without misleading them (**?**). These elicitation methods use media such as pictures, videos, audio clues, etc.

5.5. Challenge number 4: Compositional elicitation

Compositional elicitation is based on the principle that sentences are constructed from smaller units, such as noun phrases, adjective phrases, etc. When constructing a transfer rule for a sentence, it will be helpful to the system if transfer rules for these smaller units have already been learned. For instance, consider the following sentence:

$[_{PP}$*In the winter*$]$, $[_{NP}$*the roads*$]$ $[_{VP}$*freeze over*$]$ *and* $[_{NP}$*driving*$]$ $[_{VP}$*becomes*$]$ $[_{AdjP}$*dangerous*$]$.

If the system has already learned transfer rules for, e.g., noun phrases, then in learning a transfer rules for the entire sentence will not only be easier. Furthermore, the resulting transfer rule will be structural rather than flat, recognizing the parts that make up the sentence. If we can elicit translations for noun phrases, learn their transfer rules, then do the same for verb phrases, we can put the rules together to translate whole sentences.

However, as was hinted at above, compositional elicitation becomes more complex when dealing with a language that marks noun phrases in different functions in the sentence. For instance, German marks direct objects by using accusative case marking, but if a noun phrase is elicited in isolation, the translation would be in the nominative case. This puts an additional challenge on compositional learning. Not only do we need to learn transfer rules for phrases, we also need to learn in what context they appear in what specific form. The result is a kind of ordering paradox: We want to learn the smaller pieces first, but we cannot learn them without the larger context.

In Avenue we have applied a simple bootstrapping loop to get around the paradox. We elicit the simplest noun phrases in the context of very short sentences. We use these short sentences to determine how noun phrases are affected by the sentence (e.g., case marking) and how they affect the rest of the sentence (e.g., subject-verb agreement, alternative constructions for inanimate subjects, etc.)

5.6. Challenge number 5: Elicitation on 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, for example expressions of greeting, polite conversation, etc., cannot easily be captured in syntactic transfer rules. 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 (**?**). The examples below show some examples of fixed expressions that often do not translate literally between two closely related languages, English and German. The problem will be even more aggravated when working with unrelated (or not as closely related) languages.

Hello.
Guten Tag.
Good-ACC-SG day-ACC

How are you?
Wie geht es dir?
How go-3SG-PRES it you-DAT

How much is this? (asking for price)
Wieviel kostet das?
How-much cost-3SG-PRES this-NOM

It is nice to meet you.
Es istnett Sie kennen zu lernen.
It-NOM is nice you-POLITE-ACC meet-INF

In principle, we deal with non-compositional data by eliciting their translations, but we do not try to put them in context with other translations. We also do not try to learn transfer rules for subcomponents of these sentences, nor do we try to apply previously learned (phrase) transfer rules to these sentences. Non-compositional data will be elicited in a stand-alone way, simply recording the translations of the fixed expressions. We are also investigating the possibility of exploiting the small degree of compositionality still present in these problem cases.

For example, we would like to have a transfer rule that covers both *How are you?* and *How is your brother?*.

5.7. Challenge number 6: Verb Phrases

We have identified the elicitation of verb phrases as a particular challenge. The main reason for this is that verbs do not agree in their subcategorizations across languages (see e.g. (**?**), p. 124). For example, in the English construction *He declared him Prime Minister*, the verb *declare* subcategorizes for two noun phrases, an indirect object (*him*), and a direct object (*Prime Minister*). The same construction in German would be:

Er	*ernannte*	*ihn*	*zum*	*Premierminister.*
He	*declare-3SG-PAST*	*him-ACC*	*to+the-DAT-SG*	*prime minister*

In German, the verb 'ernennen' ('to 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 (**?**). Thus our aim will be lexical transfer rules for individual verbs or verb classes as opposed to lexically independent verb phrase rules.

Another reason why verb phrases pose a challange to the elicitation process is that all the resources we have are based on English verb classifications, which emphasize the different phenomena to be found in English verbs, but which do not always correspond to the verb classification in another language.

For instance, many languages have more tenses than English. Without knowing which verb tenses a given target language has, it is a hard task to obtain all the different verb forms with an English-based elicitation. This situation is very likely to arise when eliciting form new languages we know very little about, specially with languages that are very different from the ones that have been widely studied before, such as Mapudungun, a polysynthetic, highly inflected language. The following examples show how to conjugate the verb 'seek' ('kintun'), depending on who the actor is (that is, who is performing the action):

iñche kintun (I seek)
eymi kintuymi (you seek)
fey kintuy (he/she/it seeks)
iñchiñ kintuiñ (we all seek)
inchiu kintuyu (we two seek)
eymün kintuymün (you all seek)
eymu kintuymu (you two seek)
feyengün kintuyngün (they all seek)
feyengu kintuyngu (they two seek)

(slightly adapted from (**?**))

5.8. Challenge number 7: Alignment issues

As has been pointed out in the literature (e.g. (**?**)), human-specified alignment is not noise-free. There is some degree of disagreement between people who align the same sentences, and even one and the same person will sometimes not be completely consistent. This provides a problem for a system that automatically elicits data and relies heavily on informant-specified alignments. Informants can be given intuitive instructions for alignment - for instance, we will ask the informant to align a word to only those words in the target language that actually have the root with the same meaning. Such guidelines have proven useful, as in (**?**). 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 similar noise problems. Therefore, alignment issues must be dealt with whether learning from the elicitation corpus or from an unstructured corpus.

5.9. Challenge number 8: Orthographic and segmentation issues

Many of the languages AVENUE is working with have no pre-established writing system, and thus if we ask more than one bilingual informant to translate the elicitation corpus, we can not only get different ways of saying the same thing or the use of different words for the same object, but also different spellings of the same word. 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.

In addition to that, in the case of polysynthetic languages, different informants might segment words differently, and since the alignments are very sensitive to different word segmentations, the system might be confronted with two very different rules for the same phenomenon.

5.10. Elicitation Procedure – Conclusions

We have presented challenges that we face when designing and implementing a tool that elicits a controlled bilingual corpus from an informant. Some of these challenges are common to all systems that elicit linguistic data, for example alignment issues. Others are unique to our system because they are related to the way we use the elicited data. In particular, the elicited data is largely processed automatically. This poses special problems when dealing with noisy data, either because a language behaves differently than the designers anticipated, or because the informant makes mistakes. In our rule learning system, we have to have explicit noise-handling procedures, because we cannot expect to receive noise-free data from the informant.

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.

6. Conclusions and Future Work

We have presented a novel approach to learning syntactic transfer rules for machine translation. We emphasize again that the learning algorithms are not designed to work exclusively for structured data. However, learning is facilitated in a variety of ways when the translated and aligned elicitation corpus is available. The reasons lie deep within the particular algorithms. Suffice it to say that each of the three learning steps profits from knowledge about the structure and make-up of each of the training examples.

It remains to put the elicitation-based rule learning system in context: currently, we learn from structurally rather simple examples. This means that the resulting rule system is not able to translate (by structure), say, newspaper text. It can, however, translate parts of sentences via deeper analysis than for instance EBMT. In this way, the rule learning system can complement well a shallow system such as EBMT (especially when EBMT was trained from a small corpus).

Finally, the elicitation corpus as well as the learning system are continually being expanded to handle more structures and more structurally complex sentences. As the learning algorithms and the learning algorithms are developed in conjunction, scaling them up is also achieved in parallel.

References

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

Brown, Peter and Cocke, John and Della Pietra, Vincent J. and Della Pietra, Stehen A. and Jelinek, Fredrick and Lafferty, John and Mercer, Robert L. and Roossin, Paul S. A statistical approach to Machine Translation. In: Computational Linguistics, Vol. 16, No. 2, pages 79–85, 1990

Brown, Peter and Della Pietra, Stephen A. and Della Pietra, Vincent J., and Mercer, Robert L. The mathematics of statistical Machine Translation: Parameter estimation In: Computational Linguistics, Vol. 19, No. 2, pages 263–311, 1993

Brown, Ralf. Automated Dictionary Extraction for 'Knowledge-Free' Example-Based Translation In: Proceedings of the 7th International Conference on Theoretical and Methodological Issues in Machine Translation (TMI-97), 1997.

Calendino, P. Francisco. Selección de verbos mapuches. Estructura general del verbo mapuche. Instituto Superior Juan XXIII, Bahía Blanca, Argentina, 2001.

Carbonell, Jaime and Probst, Katharina and Peterson, Erik and Monson, Christian and Lavie, Alon and Brown, Ralf and Levin, Lori Automatic Rule Learning for Resource-Limited MT. In: Proceedings of the 5th Biennial Conference of the Association for Machine Translation in the Americas (AMTA-02), 2002.

Comrie, Bernard. Language Universals & Linguistic Typology. The University Chicago Press, Chigaco, IL, 1981, 2nd edition.

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

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

Eskenazi, Maxine. Using Automatic Speech Processing For Foreign Language Pronunciation Tutoring: Some Issues and a Prototype. In: Language Learning & Technology, Vol. 2, No. 2, pages 62–69, 1999.

Gaussier, Eric. 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 Assosication for Computational Linguistics (ACL-99), 1999.

Goldsmith, John. Unsupervised Learning of the Morphology of a Natural Language. In: Computational Linguistics, Vol. 27, Np. 2, pages 153–198, 2001.

Joseph H. Greenberg. Universals of language Cambridge, MA: MIT Press, 1966, 2nd edition.

Han, Benjamin and Lavie, Alon UKernel: A Unification Kernel. Available at http://www-2.cs.cmu.edu/~ benhdj/c_n_s.html. 2001.

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

Jones, Douglas and Havrilla, R. Twisted Pair Grammar: Support for Rapid Development of Machine Translation for Low Density Languages. In: Proceedings of the Third Conference of the Association for Machine Translation in the Americas (AMTA-98), 1998.

Levin, Lori and Gates, Donna and Lavie, Alon and Waibel, Alex. An Interlingua Based on Domain Actions for Machine Translation of Task-Oriented Dialogues. In: Proceedings of the 5th International Conference on Spoken Language Processing (ICSLP-98), Vol. 4, pages 1155–1158, 1998.

Levin, Lori and Vega, Rodolfo and Carbonell, Jaime and Brown, Ralf and Lavie, Alon and Cañulef, Eliseo and Huenchullan, Carolina. Data Collection and Language Technologies for Mapudungun. In: Proceedings of the Third International Conferences on Language Resources and Evaluation (LREC-02), 2002.

Melamed, Dan I. Manual Annotation of Translational Equivalence: The Blinker Project. IRCS Technical Report, Num. 98-07, 1998.

Mitchell, Tom. Generalization as Search In: *Artificial Intelligence*, Vol. 18, pages 203–226, 1982.

Nirenburg, Sergei. Project Boas: A Linguist in the Box as a Multi-Purpose Language. In: Proceedings of the First International Conference on Language Resources and Evaluation (LREC-98), 1998.

Nirenburg, Sergei and Raskin, V. Universal Grammar and Lexis for Quick Ramp-Up of MT Systems. In: Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and the 17th International Conference on Computation Linguistics (COLING-ACL-98), 1998.

Och, Franz Josef and Ney, Hermann. Discriminative Training and Maximum Entropy Models for Statistical Machine Translation. In: Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics (ACL-02), 2002.

Probst, Katharina and Brown, Ralf and Carbonell, Jaime and Lavie, Alon and Levin, Lori and Peterson, Erik Design and Implementation of Controlled Elicitation for Machine Translation of Low-density Languages. Workshop MT2010 at Machine Translation Summit VIII, 2001.

Sherematyeva, Svetlana and Nirenburg, Sergei. Towards a Unversal Tool for NLP Resource Acquisition. In: Proceedings of the 2nd International Conference on Language Resources and Evaluation (LREC-00), 2000.

Tomita, Masaru, Editor and Mitamura, Teruko and Musha, Hiroyuki and Kee, Marion. The Generalized LR Parser/Compiler Version 8.1: User's Guide. CMU Center for Machine Translation Technical Report. April, 1988.

Trujillo, Arturo. *Translation Engines*. Springer Verlag, 1999.

Wheeler, Alvaro. *Dictionary of Siona*. unpublished, 1981?.