Automatic Rule Learning for Resource-Limited MT

Jaime Carbonell, Katharina Probst, Erik Peterson, Christian Monson, Alon Lavie, Ralf Brown, and Lori Levin

Language Technologies Institute
Carnegie Mellon University
Pittsburgh, PA 15213
{jgc, kathrin, eepeter, cmonson, alavie, ralf, lsl}@cs.cmu.edu

Abstract. Machine Translation of minority languages presents unique challenges, including the paucity of bilingual training data and the unavailability of linguistically-trained speakers. This paper focuses on a machine learning approach to transfer-based MT, where data in the form of translations and lexical alignments are elicited from bilingual speakers, and a seeded version-space learning algorithm formulates and refines transfer rules. A rule-generalization lattice is defined based on LFG-style f-structures, permitting generalization operators in the search for the most general rules consistent with the elicited data. The paper presents these methods and illustrates examples.

1 Introduction

Machine Translation (MT) for minority languages offers significant potential benefits, but also presents serious scientific and economic challenges. Among the benefits are: communication between isolated, often economically disadvantaged groups (i.e. indigenous groups in Latin America) and the speakers of majority languages, and the potential preservation of endangered languages – over half of the 6,000 presently existing languages worldwide. The primary scientific challenge is the creation of MT systems for languages of little economic importance at very low cost per language, including the acquisition of linguistic information with minimal pre-existing bilingual corpora and little or no previous linguistic analysis of the minority language.

In order to address these needs, we are investigating omnivorous MT systems, including statistical and example-based MT when some parallel training corpora can be acquired, and machine learning of transfer-based MT rules when access to a native non-linguist informant permits partial elicitation of linguistic information, such as translations of model sentences and lexical-level bilingual alignments. This paper focuses on this last objective of our project: supervised learning of transfer rules with the aid of an elicitation interface to a bilingual native speaker without any assumptions regarding his or her linguistic sophistication. While our technology is eventually aimed at low-density languages, it is intended to be target language independent. Hence, we are developing the system using examples from various languages, such as Chinese, German, Mapudungun[1], and Swahili. For illustration purposes, we present examples in these languages throughout the paper.

2 Elicitation Corpus

Since there are usually little or no bilingual translated data available for minority languages, we elicit the minimal amount of information required from a bilingual informant. The informant translates a set of constituents or sentences constructed prior to learning – the same set for all languages – designed to elicit all the basic grammatical constructions, based on the principles of field linguistics. The informant supplies lexical alignments as well as translations from the source language (SL) into the target language (TL). Currently, the source language is English or Spanish.

The elicitation corpus contains sentences and sub-sentential constituents carefully constructed to obtain information about typologically prevalent features (such as tense and number) and their possible values in the target language (such as singular, dual, plural). To this end, we rely on research done in field linguistics, and use lists that were designed for this task, such as [2] and [1]. The corpus exploits compositionality, starting with smaller constituents and recycling them in higher level ones, paralleling the compositional machine learning for version spaces discussed below.

In designing the corpus, we strive to maximize coverage while minimizing size by targeting specific constructions. For instance, in order to infer rules for relative clauses, we developed a set of sentences that exhibit different types of relative clauses, such as subject and object relative clauses. If an uncontrolled corpus was used, it would need to be several orders of magnitude larger to cover a comparable range of linguistic variability, and therefore it would impose a much larger burden on the bilingual informant. Further, a controlled corpus allows us to plan a process where the informant only translates part of the corpus, and leaves other parts that have been determined to be irrelevant for the given target language untranslated. For more detail on the elicitation process, please refer to [7] and [8].

3 Translation Engine

Rule-based translation is done through a custom-built translation engine using an integrated transfer approach with analysis, transfer, and generation stages. In the engine, as in the early METAL system [4], each analysis rule is also coupled with a corresponding transfer action, as enforced by the transfer rule formalism. Table 1 provides examples: Comments are indicated by one or more semicolons. The first line contains the SL and TL category, followed by the constituent sequences.

[1] Mapudungun, spoken in Chile, is one of the minority languages we focus on.

The formalism is able to handle a variety of common translation divergences, including head-switching, changes in grammatical relations such as an object in the source language being expressed as a subject in the target language, structural changes such as having an NP become a PP in another language, and lexical gaps where one target word replaces an entire source phrase [9].

<table>
<tr><th>Rule to handle non-auxiliary verb question transfer from Chinese to English</th><th>Sample lexical rule 1</th><th>Sample lexical rule 2</th></tr>
<tr><td>S::S :</td><td>AUX::AUX |:</td><td>AUX::AUX |:</td></tr>
<tr><td>[NP VP MA] → [V NP VP "?"]</td><td>[zuo4] → [do]</td><td>[zuo4] → [does]</td></tr>
<tr><td>(;;alignments:</td><td>(;;alignments:</td><td>(;;alignments:</td></tr>
<tr><td>(x1::y2)</td><td>((x1::y1)</td><td>(x1::y1)</td></tr>
<tr><td>(x2::y3)</td><td></td><td></td></tr>
<tr><td>;;x-side constraints:</td><td>;;x-side constraints:</td><td>;;x-side constraints:</td></tr>
<tr><td>;; (parsing)</td><td>;; (parsing)</td><td>;; (parsing)</td></tr>
<tr><td>((x0 subj) == x1)</td><td></td><td></td></tr>
<tr><td>((x0 subj case) = nom)</td><td></td><td></td></tr>
<tr><td>(x0 = x2)</td><td></td><td></td></tr>
<tr><td>((x0 act) = *quest)</td><td></td><td></td></tr>
<tr><td>;;xy-constraints:</td><td>;;xy-side constraints:</td><td>;;xy-constraints:</td></tr>
<tr><td>;; (transfer)</td><td>;; (transfer)</td><td>;; (transfer)</td></tr>
<tr><td>(y0 = x0)</td><td></td><td></td></tr>
<tr><td>((y0 act) =c *quest)</td><td></td><td></td></tr>
<tr><td>;;y-side constraints:</td><td>;;y-side constraints:</td><td>;;y-side constraints:</td></tr>
<tr><td>;; (generation)</td><td></td><td></td></tr>
<tr><td>((y1 form) = *do)</td><td>((Y0 form) = *do)</td><td>((y0 form) = *do)</td></tr>
<tr><td>((y1 agr) = (y2 agr))</td><td>((y0 agr) =</td><td></td></tr>
<tr><td></td><td>(*or* *1-sg *2-sg *plu))</td><td>((y0 agr) = *3-sg)</td></tr>
<tr><td>(y2 == (y0 subj))</td><td>((y0 tense) = *pres))</td><td>((y0 tense) = *pres))</td></tr>
<tr><td>(y3 = y0))</td><td></td><td></td></tr>
</table>

Table 1. Sample Transfer Rules

Translation starts with a bottom-up unification-based chart parser that produces an ambiguity-packed representation of the source language input. If at any one point more than one rule can be applied to a structure, the rules are applied in the order in which they appear in the grammar. Analysis builds both a syntactic constituent (or for short c-)structure and an associated feature (f-)structure. For example, in the structural transfer rule in figure 3, which handles the transfer of some types of interrogative sentences from Mandarin to English, the feature structure for the source language (SL) S constituent, represented by x0, is built using the x-side, i.e. parsing, constraints.

Transfer is done from the top down, starting at the top node in the chart created during parsing. Transfer rules explicitly state the alignment of source and target constituents through equations such as (x1::y2) and (x2::y3) in the example rule, indicating that the first source constituent maps to the second target constituent, and the second source constituent maps to the third target constituent. Not all source constituents need to map to the target (such as the Chinese question particle MA at x3 in this rule which is deleted during transfer). Target constituents that are not aligned to a source constituent will be created based upon the feature structure assigned to them in the transfer and generation equations. During transfer, the engine uses these alignments to reorder constituents.

Features may also be passed from source to target using xy-constraints. For example, the (y0 = x0) equation in table 3 copies the entire source sentence feature structure to the target.

During generation, the target side feature structures are built using the y-side constraints. In the example rule, the first constituent V (y1) has its form set to 'do', which is used later to choose the correct verb to insert. To enforce that the form of 'do' agrees with the subject NP at y2, the ((y1 agr) = (y2 agr)) is used.

Once a selected transfer rule is applied, transfer and generation continues by recursively running applicable transfer rules on the sub-constituents of the rule until the lexical level is reached. Lexical transfer rules then apply to select the appropriate translation. In the example, the auxiliary 'do' is inserted at the start of the sentence. The appropriate form of 'do' is selected based upon the agreement constraint between the subject NP and initial V in the sentence rule and the agreement features in the individual lexical rules.

4 Seeded Version Spaces for Transfer Rule Learning

Transfer rule learning consists of three steps. The first step, feature detection, determines what lingustic features are present in the target language. For instance, does the language have a number-agreement feature? If so, is the distinction between singular and plural, or between singular, dual and plural? The details of feature detection are beyond the scope of this paper; the interested reader should refer to [7]. The second step is seed-rule generation, where preliminary transfer rules are generated based on lexically-aligned translated constituents. Seed generation produces transfer rules that are very specific to the training examples, essentially defining the operational *S-boundary* (specific boundary) for Version-Space learning [6] [5] [3]. The third step, Seeded Version Space Learning, generalizes the seed rules in order to produce transfer rules that can be used to translate larger classes of unseen examples.

4.1 Seed Generation

Version Space learning suffers worst-case exponential behavior as a function of the number of generalization (or specialization) steps required from initial instances. But if the initial instances are already generalized and the number of additional generalization steps k-bounded, or limited to a greedy-search, the process is at worst a k-degree polynomial. Hence, to ensure computational tractability, we first build generalized rule-instances called 'seed rules,' to the extent that we can build first-level, error-free deterministic generalization. Future work will relax the error-free assumption by substituting a retractable S-boundary.

Seed rules represent generalizations over the lexical level, but are still very specific to the lexically-aligned sentence pair from which they are derived. The major-language (English) sentences in the training corpus have been pre-parsed and disambiguated in advance, so that the system has access to a correct f-structure and a correct phrase or c-structure for each sentence. The seed generation algorithm can proceed in different settings based 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. The other extreme is the absence of any information, in which case the system makes assumptions about what linguistic feature values transfer to the target language. For example, we assume that number and definiteness transfer across languages, but not gender or case. This information is either defaulted or extracted in the earlier feature detection phase.

During seed generation, all transfer rule components are produced from the sentence pair, with the exception of xy-constraints.[2]. Table 2 below summarizes how each part of the seed rule is produced.

SL part-of-speech sequence	English c-structure
TL part-of-speech sequence	1) from the TL dictionary 2) from the English POS sequence
Alignments	from the user
x-side constraints	from the English f-structure
y-side constraints	1) from the TL dictionary 2) from the corresponding English words
xy-constraints	Not produced during seed generation

Table 2. Summary of Seed Generation

To make these concepts more concrete, consider an example of a seed rule as produced by our algorithm for simple NPs in English-German (for expositionary simplicity), as illustrated in table 4 in section 4.3: The English part-of-speech sequence was obtained from the English parse. The German part-of-speech sequence was obtained from a combination of the target language dictionary and the assumption that in the absence of unambiguous target language information, parts of speech transfer into the target language for aligned words. The lexical alignments were given by the bilingual informant. The x-side constraints were read off the English c- and f-structures. The y-side constraints were again obtained by a combination of target language information and 'safe' transfer

[2] xy-constraints are inferred during the Seeded Version Space Learning.

heuristics. If feature information is given in the target language, it is used. If no such information is available, features such as agreement and count project their values onto the target language. This means that target language words obtain their feature values from their corresponding words in the source language.

4.2 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. Then, the new seed rule (the higher-level rule) is no longer a flat rule, but consists of previously learned transfer constituents (lower-level rules) such as NPs or ADJPs.

Compositional rules are learned by first producing a flat rule as was described above from the bilingual constituent (phrase or sentence). In the following, the system traverses the c-structure parse of the English constituent. Each node in the parse is annotated with a label such as NP, and roots a subtree, which itself covers part of the constituent. The system then checks if there exists a lower-level rule that can be used to correctly translate the words in this subtree. Consider the following example:

(<NP> (<DET> ((ROOT ∗THE)) THE)
(<NBAR>
(<ADJP> (<ADVP> (<ADV> ((ROOT *VERY)) VERY))
(<ADJP> (<ADJ> ((ROOT *TALL)) TALL)))
(<NBAR> (<N> ((ROOT *WOMAN)) WOMAN))))

This c-structure represents the parse tree for the NP ‘the very tall woman’. 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 constituent that is rooted at this node. For example, the node ADJP covers the chunk ‘very tall’. 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 gains access to a bilingual sentence chunk. In our example the bilingual sentence chunk is ‘very tall’ - ‘sehr grosse’. It also obtains its SL category, in this example ADJP.

In the following, it calls the transfer engine with this chunk, and is returned 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. The rightmost column in table 3 presents an example of how the flat rule is modified to reflect the existance of an ADJP rule that can translate the chunk

Lower-level rule	**Uncompositional Rule**	**Compositional Rule**
;;SL: very tall	;;SL: the very tall woman	;;SL: the very tall woman
;;TL: sehr gross	;;TL: die sehr grosse Frau	;;TL: die sehr grosse Frau
ADJP::ADJP	NP::NP	NP::NP
[ADV ADJ]	[DET ADV ADJ N]	[DET ADJP N]
→ [ADV ADJ]	→ [DET ADV ADJ N]	→ [DET ADJP N]
(;;alignments:	(;;alignments:	(;;alignments:
(x1::y1)	(x1::y1)	(x1::y1)
(x2::y2)	(x2::y2)	(x2::y2)
	(x3::y3)	(x3::y3)
	(x4::y4)	
;;x-side constraints:	;;x-side constraints:	;;x-side constraints:
	((x1 agr) = *3-sing)	((x1 agr) = *3-sing)
	((x1 def) = *def)	((x1 def) = *def)
	((x2 agr) = *3-sing)	
	((x2 count) = +)	
	((x3 agr) = *3-sing)	((x3 agr) = *3-sing)
	((x4 agr) = *3-sing)	((x3 count) = +)
	((x4 count) = +)	((x3 count) = +)
(x0 = x2)	(x0 = x4)	(x0 = x3)
;;y-side constraints:	;;y-side constraints:	;;y-side constraints:
	((y1 gender) = *f)	((y1 gender) = *f)
	((y1 case) = (*or* *acc *nom))	((y1 case) = (*or* *acc *nom))
	((y1 agr) = *3-sing)	((y1 agr) = *3-sing)
	((y1 def) = *def)	((y1 def) = *def)
	((y3 gender) = *f)	((y2 gender) = *f)
	((y3 case) = (*or* *acc *nom))	((y2 case) = *nom)
	((y3 def) = *def)	((y2 def) = *def)
	((y3 agr) = *3-sing)	((y2 agr) = *3-sing)
	((y4 gender) = *f)	((y3 gender) = *f)
	((y4 case) = (*or* *acc *nom))	((y3 case) = (*or* *acc *nom))
	((y4 agr) = *3-sing)	((y3 agr) = *3-sing)
	((y4 count) = +)	((y3 count) = +)
(y0 = y2)	(y0 = y4)	(y0 = y3)

Table 3. Example of Compositionality

'very tall'. 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. In the example in table 3, all the x-side constraints on the indices x2 and x3 are removed.

Finally, the y-side constraints are adjusted. For each sentence chunk that was correctly translated by a lower-level rule, the compositionality module compares the f-structures of the correct translation and the incorrect translations as returned by the transfer engine. 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, as can again be seen in the example in table 3.

4.3 Seeded Version Space Learning

The first step in Seeded Version Space Learning is to group the seed rules by their constituent sequences, alignments, and category information. This means that in each group the seed rules differ only in the constraints. The learning algorithm is run on each group separately, as each group corresponds to a target concept (i.e. a target generalized transfer rule), thus it defines a version space. At the heart of the version space learning is the merging of two transfer rules to a more general transfer rule. To this end, it is necessary to clearly define the partial order by which the generality of transfer rules can be assessed (i.e. how the implicit generalization lattice is constructed):

Definition 1 *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 transfer rules are equivalent if and only if all f-structures that are satisfied* 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. In particular, we identified three generalization operations:

1. Deletion of a value constraint
2. Deletion of an 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:
   $((X_i\ feature_k) = value_l)$
   $((X_j\ feature_k) = value_l)$
   $\rightarrow ((X_i\ feature_k) = (X_j\ feature_k))$,
   or similarly for y-side and xy-constraints.

Generalization is achieved by merging transfer rules, which in turn is based on the three generalization operations defined above. Suppose we wish to merge two transfer rules tr_1 and tr_2 to produce the most specific generalization of the two, stored in tr_{merged}. The algorithm proceeds in three steps:

SeedRule1	**SeedRule2**	**Generalized Rule**
;;SL: the man	;;SL: the woman	;;SL:
;;TL: der Mann	;;TL: die Frau	;;TL:
NP::NP	NP::NP	NP::NP
[DET N] → [DET N]	[DET N] → [DET N]	[DET N] → [DET N]
(;;alignments:	(;;alignments:	(;;alignments:
(x1::y1)	(x1::y1)	(x1::y1)
(x2::y2)	(x2::y2)	(x2::y2)
;;x-side constraints:	;;x-side constraints:	;;x-side constraints:
((x1 agr) = *3-sing)	((x1 agr) = *3-sing)	((x1 agr) = *3-sing)
((x1 def) = *def)	((x1 def) = *def)	((x1 def) = *def)
((x2 agr) = *3-sing)	((x2 agr) = *3-sing)	((x2 agr) = *3-sing)
((x2 count) = +)	((x2 count) = +)	((x2 count) = +)
;;y-side constraints:	;;y-side constraints:	;;y-side constraints:
((y1 agr) = *3-sing)	((y1 agr) = *3-sing)	((y1 agr) = *3-sing)
((y1 case) = *nom)	((y1 case) = (*not* *gen *dat))	
((y1 def) = *def)	((y1 def) = *def)	((y1 def) = *def)
((y2 gender) = *m)	((y2 gender) = *f)	((y2 gender) = *f)
((y2 agr) = *3-sing)	((y2 agr) = *3-sing)	((y2 agr) = *3-sing)
((y2 case) = *nom)		
((y2 gender) = *m)	((y2 gender) = *f)	((y2 gender) = (y1 gender))

Table 4. Seed Rules and Generalized Transfer Rule

1. Insert all constraints that appear in tr_1 and tr_2 into tr_{merged} and subsequently eliminate them from tr_1 and tr_2.
2. Consider tr_1 and tr_2 separately. Perform all instances of operation 3 that are possible given the constraints.
3. Repeat step 1.

Figure 4 is an example of a very simple version space, seeded with only two transfer rules, a rule produced by ‘the man’ and one produced by ‘the woman’. In this case, the merged rule can be used to translate both ‘the man’ and ‘the woman’, whereas each of the seed rules can only be used to translate the NP they were produced from.

The Seeded Version Space algorithm itself is the repeated application of merging two transfer rules in a group and checking whether the merged rule is specific enough to translate correctly all those sentences that the unmerged rules could translate. If this is the case, a merge is accepted. Merging continues until no two transfer rules in the cluster can be merged any more. Note that this method is a greedy approach to generalization, without guaranteeing that the optimal (most general) transfer rule will be found. However, the method is sound with respect to allowable generalizations and it is computationally tractable.

5 Conclusions and Future Work

We presented a novel approach to learning in machine translation, a method that we hope will open MT up to a variety of languages in which little training data are available. We realize that our approach presents a large undertaking: it requires a specially adapted transfer engine, as well as a system that infers transfer rules. This paper presents the current state of our system. The focus of the future will be to scale to complex constructions and alignments. Aside from performing a baseline evaluation, we plan to refine the search through the seeded version space by defining a function that determines what merge is the best at any one step, given that there is more than one possible merge. Further, we plan to revisit the generalization operations that have been defined, so as to determine what the optimal step size of generalization should be. Also, currently no retraction is possible from overgeneralization. This issue will be addressed by adding specialization operations. The transfer engine will be extended to output partial translations if no full translation can be given. Also, work is underway to order the rule application by the complexity of the rule and the specificity of its constraints. This will be especially important as our system will be integrated into a multi-engine system, together with statistical and example-based MT methods.

References

1. Luc Bouquiaux and Jacqueline M.C. Thomas: Studying and Describing Unwritten Languages. The Summer Institute of Linguistics. (1992)
2. Bernard Comrie and Norval Smith: Lingua Descriptive Series: Questionnaire. In: Lingua. **42** (1977) 1–72
3. Hirsh Haym: Theoretical Underpinnings of Version Spaces. In: Proceedings of the Twelfth International Joint Conference on Artificial Intelligence (IJCAI91). Morgan Kaufmann Publishers. (1991) 665–670
4. Hutchins, W. John and Somers, Harold L.: An Introduction to Machine Translation. Academic Press, London. (1992)
5. Tom Mitchell: Machine Learning. McGraw Hill (1996)
6. Mitchell, T. M.: Version Spaces: An Approach to Concept Learning. Stanford University. **December** (1978)
7. Katharina Probst and Ralf Brown and Jaime Carbonell and Alon Lavie and Lori Levin and Erik Peterson: Design and Implementation of Controlled Elicitation for Machine Translation of Low-density Languages. Workshop MT2010, Machine Translation Summit 2001. (2001)
8. Katharina Probst and Lori Levin: Challenges in Automated Elicitation of a Controlled Bilingual Corpus. TMI 2002. (2002)
9. Trujillo, A.: Translation Engines: Techniques for Machine Translation. Springer-Verlag London Limited, London. (1999)