# Corpus-Based Learning of Generalized Parse Tree Rules for Translation*

H. Altay Güvenir and Ayşegül Tunç

Department of Computer Engineering and Information Sciences
Bilkent University, Ankara 06533, Turkey
e-mail: {guvenir,aytunc}@cs.bilkent.edu.tr

**Abstract.** This paper proposes a learning mechanism to acquire structural correspondences between two languages from a corpus of translated sentence pairs. The proposed mechanism uses analogical reasoning between two translations. Given a pair of translations, the similar parts of the sentences in the source language must correspond the similar parts of the sentences in the target language. Similarly, the different parts should correspond to the respective parts in the translated sentences. The correspondences between the similarities, and also differences are learned in the form of rewrite rules. The system is tested on a small training dataset and produced promising results for further investigation.

## 1 Introduction

Traditional approaches to machine translation (MT) suffer from tractability, scalability and performance problems due to the necessary extensive knowledge of both the source and the target languages. Corpus-based machine translation is one of the alternative directions that have been proposed to overcome the difficulties of traditional systems. Two different approaches in corpus-based MT have been used. These are *statistical* and *example-based* machine translation (EBMT) or *memory-based* machine translation (MBMT). Both approaches assume the existence of an already translated corpus to derive a translation for an input. While statistical MT uses statistical metrics to choose the most probable words in the target language, EBMT uses pattern matching techniques to translate subparts of the given input [1]. Also, Siskind has dealt with the question of what matches what in the examples in some detail for the case of lexical matching [12].

EBMT has been proposed by Nagao [7] as *Translation by Analogy* which is in parallel with memory based reasoning [13], case-based reasoning [9] and derivational analogy [2]. The example-based translation relies on the use of past translation examples to derive a translation for a given input. The input and the example translations are first compared analogically to retrieve the *closest* examples to the input. Then, the fragments of the retrieved examples are translated and recombined in the target language. Prior to the translation of an input

* This research has been supported in part by NATO Science for Stability Program Grant TU-LANGUAGE.

sentence, the correspondences between the source and target languages should be available to the system; however this issue has not been given enough consideration by the current EBMT systems [3, 5, 10, 11, 14]. Kitano has adopted the manual encoding of the translation rules, however this is a difficult and an error-prone task for a large corpus. In this paper, we formulate this acquisition problem as a machine learning task in order to automate the process.

We use example-based learning techniques to derive the correspondences between two languages, using a corpus of translation cases. The basic idea in example-based learning is to use past experiences or cases to understand, plan, or learn from novel situations [4, 6, 8]. A case is a pair of corresponding sentences in the source and target languages. Our corpus consists of translation examples that represent Turkish translations of English sentences.

We use a heuristic to learn the correspondences between the source and target languages. The heuristic is based on the following idea. Given two translation pairs, if the sentences of the source language exhibit some similarities, then the corresponding sentences in the target language must have similar parts, and they must be translations of the similar parts of the sentences in the source language.

Further, the remaining different parts source sentences should also match the corresponding target sentences. However, if the sentences do not exhibit any similarity, then no correspondences are inferred. Consider the following translation pair to illustrate the heuristic:

`I gave the book to Mary` → `Mary+A kitap+I ver+DI+Im`
`I gave the pencil to Mary` → `Mary+A kalem+I ver+DI+Im`

Similarities between the translation examples are shown as underlined. The remaining parts are the differences between the sentences. We represent the similarities in the source language as {`I gave the` $\square^S$ `to Mary`}, and the corresponding similarities in the target language as {`Mary+A` $\square^T$`+I ver+DI+Im`}. According to our heuristic, these similarities should correspond each other. Here, $\square^S$ denotes a component that can be replaced by *any* appropriate structure in the source language and $\square^T$ refers to its translation in the target language. This notation represents an *abstraction* of the differences {`book` vs. `pencil`} and {`kitap` vs. `kalem`} in the source and target languages, respectively. Using the heuristic further, we infer that `book` should correspond to `kitap` and `pencil` should correspond to `kalem`; hence learning further correspondences between the examples.

Our learning algorithm based on this heuristic is called **PA**rse **T**ree **R**ul**E** **L**earning **A**lgorithm (PATRELA). Given a corpus of translation cases, PATRELA infers the correspondences between the source and target languages in the form of *Generalized Parse Tree* (GPT) or *Ordinary Parse Tree* (OPT) rules. An OPT is a parse tree corresponding to the grammatical structure of a complete or a partial sentence. We represent the differences between the sentences by OPTs. On the other hand, A GPT is an OPT with possible variable nodes. We form a GPT by replacing the differences between two sentences by variable nodes. This yields a similarity pattern between the sentences. Fig. 1 shows the

GPT rule which is learned from the translation pairs given above. A variable node can replaced by a subtree of the same category during the translation. By replacing all the variables on a GPT, we obtain a complete and grammatical sentence. This is analogous to the derivation of a sentence in a CFG (Context Free Grammar) by replacing the nonterminals. For example, consider the similarity pattern between the English sentences, given by the left-hand side of the rule in Fig. 1. This pattern yields various different sentences e.g., "I gave the *apple* to Mary", "I gave the *pencil* to Mary", "I gave the *book* to Mary", etc., when the variable is replaced by *apple*, *pencil* or *book*, respectively.

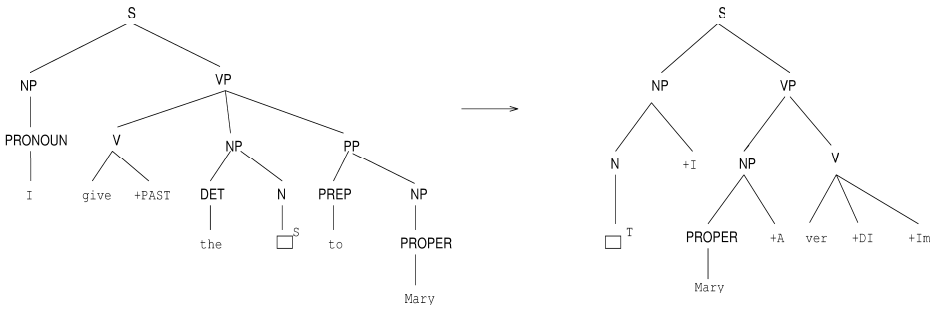


**Fig. 1.** GPT representation for "I gave the $\square^S$ to Mary $\rightarrow$ $\square^T$+I ver+DI+Im".

The rest of the paper is organized as follows. Section 2 describes the underlying mechanisms of PATRELA, along with sample rule derivations. Section 3 illustrates the example-based translation. Section 4 concludes the paper.

## 2 Learning

The PATRELA algorithm infers translation rules using the similarities and differences between a pair of translation examples ($E_i$, $E_j$) from a corpus. A translation example, $E_i$, consists of a source sentence $E_i^S$ and its translation $E_i^T$ in the target language, $E_i = (E_i^S \rightarrow E_i^T)$. Sentences are represented by their parse trees in the form of nested lists. For example, the sentence "I read the book" is represented as

$$[S, [NP, [PRO, I]], [VP, [V, read, +PAST], [NP, [DET, the], [N, book]]]]$$

where the first element of a list represents the root and the remaining elements represent the children of the tree.

Given a pair of translation examples ($E_A$, $E_B$), PATRELA first searches for the similarities between the sentences in the source language ($E_A^S$ and $E_B^S$) and the similarities between the sentences in the target language ($E_A^T$ and $E_B^T$). For example, the following translation examples,

I read+PAST the book $\rightarrow$ Kitap+I oku+DI+Im
I catch+PAST the bus $\rightarrow$ Otobüs+I yakala+DI+Im

exhibit some similarities. The comparison of English sentences yields the similarity $S^S$ = {I $\square_V^S$ +PAST the $\square_N^S$}, and the comparison of Turkish sentences yields $S^T$ = {$\square_N^T$+I $\square_V^T$+DI+Im}. The following GPT rule is formed using the similarities $S^S$ and $S^T$:

I $\square_V^S$+PAST the $\square_N^S$ $\rightarrow$ $\square_N^T$+I $\square_V^T$+DI+Im

Note that each variable in a GPT rule has an associated category. The categories of the variables are used to *constrain* the selection of the target phrases during translation in order to maintain the grammaticality of the output. For example, the pattern, I $\square_V^S$+PAST the $\square_N^S$ has two variables: a verb and a noun. Using the category information, we are not allowed to replace the "noun" variable of this pattern by a phrase with any category other than a noun.

After the similarities are determined, the differences are formed by removing these similar parts. Differences between the sentences are denoted by the sets $D^S$ and $D^T$ for the source and the target languages, respectively. A difference is a pair of parts, one from the first sentence, and the other from the second sentence. For example, the differences for the examples give above are denoted by the sets, $D^S$ = {read : catch, book : bus} and $D^T$ = {kitap : otobüs, oku : yakala}.

Next, the similarities and differences between the translation examples are matched according to the heuristic described above. This heuristics is similar to the way humans actually learn languages from examples. Whether it is a native or a second language acquisition, humans derive generalized sentence patterns from the example sentences that they are exposed to in the target language.

### 2.1 Matching

The question we address here is, among various ways of matching multiple differences, which one is the *correct* matching? To illustrate the problem, consider the following cases.

I go+PAST to school $\rightarrow$ Okul+A git+DI+Im
I come+PAST to home $\rightarrow$ Ev+A gel+DI+Im

with differences $D^S$ = {go : come, school : home} and $D^T$ = {okul : ev, git : gel}. The matching question is "does go correspond to okul or git in Turkish?" Similarly "does school correspond to ev or gel?" The answer is, given no other information, there is no way to guarantee the correct matching among more than one different parts. In these situations, only the similarities are matched; no other rules are learned from the differences.

The problem may be attacked more effectively, if some of the differences are translated using the previous rules. Since those differences will not yield new rules, even if they are matched, we eliminate those from the difference sets, This is called the *reduction*. If, after reduction, there remains a *singleton* difference between the sentences, its parts are matched to each other. Otherwise, no rules are learned from the differences. For example, consider again the translation pairs given above. If we know that go corresponds to git in Turkish, we may

**PATRELA**$(E_A, E_B)$
**begin**
  ALIGN $E_A : (E_A^S \rightarrow E_A^T)$ and $E_B : E_B^S \rightarrow E_B^T$
  **if** $S^S$ or $S^T$ is null **then** exit {*no learning*}
  **else if**$(|D^S| \neq |D^T|$ **then** exit {*no learning*}
  **else if** $|D^S| = |D^T| = 1$ **then**
    produce GPT : $S^S \rightarrow S^T$ and
    produce OPTs : $d_{A_1}^S \rightarrow d_{A_1}^T$ and $d_{B_1}^S \rightarrow d_{B_1}^T$
  **else if** for some $i$ and $j$ {*reduction*}
      $d_{A_i}^S \rightarrow d_{A_j}^T$ (or $d_{B_i}^S \rightarrow d_{B_j}^T$) has already been learned **then**
    produce OPT: $d_{B_i}^S \rightarrow d_{B_j}^T$ (or $d_{A_i}^S \rightarrow d_{A_j}^T$)
    recurse PATRELA$(E_A - d_{A_i}, E_B - d_{B_i})$
**end**

**Fig. 2.** The PATRELA algorithm. Here, $E_A - d_{A_i} = E_A - (d_{A_i}^S, d_{A_i}^T)$ denotes the operation of removing $d_{A_i}^S$ from $E_A^S$ and $d_{A_i}^T$ from $E_A^T$.

immediately infer that `school` corresponds to `okul`. However, if we do not how to translate any of these differences, the problem remains unsolved.

Another matching problem occurs when the difference sets for the source and target languages have different sizes. The following cases illustrate this situation.

`I win+PAST the prize` → `Ödül+I ben kazan+DI+Im`
`We win+PAST the prize` → `Ödül+I biz kazan+DI+Ik`

Here, $D^S$ = {`I : we`} and $D^T$ = {`ben : biz, +Im : +Ik`}, however, the elements of $D^S$ and $D^T$ cannot be matched one-to-one to each other, since one element of $D^T$ will remain unmatched in any way. Therefore, we do not allow learning from the differences when $|D^S| \neq |D^T|$.

Therefore, the necessary condition for learning can be stated as $S^S \neq \emptyset$ and $S^T \neq \emptyset$. Further, if $|D^S| = |D^T| = 1$ is also satisfied, then learning also takes place among the differences. Otherwise, only a rule from the similarities is generated.

### 2.2 PATRELA

For a given pair of translation examples, $E_A = (E_A^S, E_A^T)$ and $E_B = (E_B^S, E_B^T)$, the outline of the learning algorithm is as follows:

1. **Alignment:** Determine the similarity and difference sets of the source and target sentences. Call them $S^S$, $S^T$ and $D^S$, $D^T$, where $S$ stands for the similarity, $D$ stands for the difference set, and the indices denote whether the alignment is in the source(S) or the target(T) language.
2. **Reduction:** Using the rules learned earlier, if a difference $(d_{A_i}^S \in D^S)$ can be matched to another difference $(d_{A_j}^T \in D^T)$, then remove $d_{A_i}^S$ from $D^S$

| | | Source (English) | | Target (Turkish) |
|---|---|---|---|---|
| | $E_A^S$ | I go+PAST to school | $E_A^T$ | Okul+A git+DI+Im |
| | $E_B^S$ | I come+PAST to my office | $E_B^T$ | Ofis+Im+A gel+DI+Im |
| | $D_E$ | {go : come, school : my office} | $D_T$ | {git : gel, okul : ofis} |
| Reduction: | $D_E^r$ | {school : my office} | $D_T^r$ | {okul : ofis+Im} |
| Rule-1: | | I $\square_V^S$+PAST to $\square_{NP}^S$ | $\rightarrow$ | $\square_{NP}^T$+A $\square_V^T$+DI+Im |
| Rule-2: | | school | $\rightarrow$ | okul |
| Rule-3: | | my office | $\rightarrow$ | ofis+Im |

**Table 1.** An example trace of PATRELA.

and its translation $d_{A_i}^T$ from $D^T$, then recurse with $E_A - (d_{A_i}^S, d_{A_j}^T)$ and $E_B - (d_{B_i}^S, d_{B_j}^T)$.

3. **Learning:** If $S^S = \emptyset$ or $S^T = \emptyset$, exit without learning any rules. Otherwise, produce a GPT rule by matching the similarities in $S^S$ and $S^T$. Further if $D^S$ and $D^T$ have singleton elements then produce two OPT rules $d_{A_1}^S \rightarrow d_{A_1}^T$ and $d_{B_1}^S \rightarrow d_{B_1}^T$.

The PATRELA algorithm is given in Fig 2. An example trace of the algorithm is shown in Table 1. Prior to the execution, the rules `go` $\rightarrow$ `git` and `come` $\rightarrow$ `gel` are assumed to exist in the memory, so that the reduction operation can eliminate some of the differences.

### 2.3 Example-Base

An example-base is composed of a set of example translations which are decomposed into their morphological constituents before they are represented as parse trees. The morphological information is embedded in the parse tree representation where the morphemes are considered as separate children. For example, we represent the Turkish sentence "Kitabı okudum" by [$S$, [$NP$, [$N$, kitap], [$C$, +I]], [$VP$, [$V$, oku], [$T$, +DI], [$AGR$, +Im]]] in the example-base.

The morphological information accounts for handling the lexical mismatches between the source and target languages. A lexical mismatch occurs when the number of words in a source phrase differs from the number of words in its target translation, as illustrated by "`at the garden`" vs. "`evde`". To highlight the utility of morphological information, consider the following examples which are represented at the string level, rather than decomposed into morphological constituents:

`I saw you at the garden` $\rightarrow$ `Seni bahçede gördüm`
`I saw you at the party` $\rightarrow$ `Seni partide gördüm`

An alignment of these examples would produce the differences $D^S$ = {`garden : party`} vs. $D^T$ = {`bahçede : partide`}. However, these differences yield incorrect rules, when matched as follows:

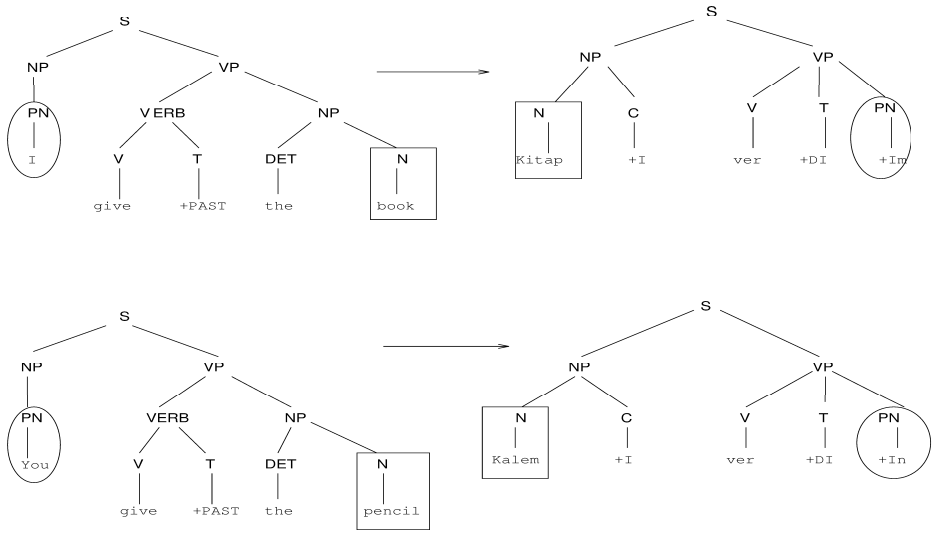


**Fig. 3.** Parse tree representation for Example 1.

`garden` → `bahçede`
`party` → `partide`

The problem is that `garden` actually corresponds to `bahçe` and `party` corresponds to `parti`. To infer the correct rules, we use morphological information to align the translations, as follows:

`I see+PAST you at the garden` → `Sen+I bahçe+DA gör+DI+Im`
`I see+PAST you at the party` → `Sen+I parti+DA gör+DI+Im`

where $D^S$ = {`garden : party`} vs. $D^T$ = {`bahçe : parti`}. In this alignment, it is detected that the morpheme `+DA` which corresponds to the preposition `at` is similar to both of the Turkish statements.

### 2.4 Sample Derivations of Translation Rules

In this section, sample derivations of translation rules are given. Translation examples are taken from a sample example-base used for training.

**Example 1.** Given the following translation pairs:

`I give+PAST the book` → `Kitap+I ver+DI+Im`
`You give+PAST the pencil` → `Kalem+I ver+DI+In`

the corresponding parse trees are shown in Fig. 3. The differences between the sentences are shown in circles and rectangles, which respectively correspond to each other. The similarities are matched, yielding the following GPT rule:

$\square^S_{PN}$ `give+PAST the` $\square^S_N \rightarrow \square^T_N$`+I ver+DI+`$\square^T_{PN}$

Note that the categories associated with the variables are the same as the root of the corresponding subtrees. After determining the similarities, the remaining parts constitute the differences, $D^S$ = {`I : you, book : pencil`} and $D^T$ ={`+Im : +In, kitap : kalem`}. Assuming that the rules `I` → `+Im` and `you` → `+In` have been learned previously, the reduction operation eliminates the first difference. Since there remains a singleton difference after reduction, two OPT rules are inferred as below:

`book` → `kitap`
`pencil` → `kalem`

**Example 2.** From the examples,

`Does Mary study+PRES` → `Mary çalış+Iyor mI`
`Does he like+PRES this song` → `Bu parça+I sev+Iyor mI`

the following rules are learned:

`Does` $\square^S_{NP}$ $\square^S_{VP}$ → $\square^T_{NP}$ $\square^T_{VP}$ `mI`
`Mary study+PRES` → `Mary çalış+Iyor`
`He like+PRES this song` → `Bu parça+I sev+Iyor`

The GPT rule describes an interrogative sentence pattern. Using the difference sets $D_S$ = {`Mary study+PRES : he like+PRES this song`} and $D_T$ = {`Mary çalış+Iyor : Bu parça+I sev+Iyor`}, the remaining singleton differences are matched at the sentence level.

**Example 3.** From the examples,

`I go+PAST to my school` → `Okul+Im+A git+DI+Im`
`I go+PAST to school` → `Okul+A git+DI+Im`

the following rules are learned:

`I go+PAST to` $\square^S_{NP}$ → $\square^T_{NP}$`+A git+DI+Im`
`my school` → `okul+Im`
`school` → `okul`

In this example, a dangled minimal difference is generalized during the Alignment. For two examples $E_A$ and $E_B$, either in the source or the target language, a *minimal* difference is an element of the difference set, $d_i = d_{A_i} : d_{B_i}$, where $d_{A_i}$ and $d_{B_i}$ are disjoint. For example, the differences $D^S$ = {`my school : school`} are *not* minimal, because `school` is common to both sentences. Similarly, $D^T$ = {`okul+Im : okul`} is not minimal. A *dangled* difference, on the other hand, is an element where either $d_{A_i}$ or $d_{B_i}$ is null, i.e. one of the differences does not correspond to any element in the other sentence.

Since dangled differences may yield null translation rules which may lead to incorrect translations, we avoid the production of dangled differences during the Alignment. For the translation examples given above, the alignment process first produces minimal dangled differences {`my : NULL`} and {`+Im : NULL`}.

Production of dangled differences is avoided as follows. Given two parse trees

$P_1$ and $P_2$ with a matching internal node $N$ at the same relative positions in the two trees. Suppose N has different number of children in the two subtrees. Then some children will remain dangled in the subtree that has more children. In this situation, we form a single difference rooted at N, including all children. This difference may not be minimal (i.e. some of the children may exhibit similarities), but it is ensured to be non-dangled.

**Example 4.** From the examples,

```
if a pencil is drop+PP then it fall+PRES
      → kalem bırak+Il+Ir ise düş+Ar
if he study+PRES, then his mark will be higher
      → Çalış+Ir ise not+I daha yüksek ol+Ir
```

an `if-then` pattern is learned as

$$\texttt{if } \square^S_{cond} \texttt{ then } \square^S_{concl} \rightarrow \square^T_{cond} \texttt{ ise } \square^T_{concl}$$

The variables of the source and target patterns are *uniquely* renamed in the GPT rule in order to be distinguished from each other uniquely. If the variables are not distinguished from each other, they may be translated into unmatching positions on the GPT pattern. For example, the rule

$$\texttt{if } \square^S_S \texttt{ then } \square^S_S \rightarrow \square^T_S \texttt{ ise } \square^T_S$$

may yield an incorrect translation when the variables are translated in reverse order. To avoid this situation, we match the variables belonging to the *same* category through unique renaming.

**Example 5.** From the examples,

```
I would like to look at it → O+A bak+mAk iste+Ir+Im
Don't look at it           → O+A bak+mA
```

Although `look at it` is similar in both sentences, no rules are learned from these examples. An implicit condition to learn a GPT rule is that the sentences must have a common sentence structure. In this example, the first sentence is affirmative while the second sentence is imperative. Therefore, the sentences do not exhibit a common structure. Consequently we do not infer any rules, although `look at it` and `O+A bak` are similar in the source and the target statements, respectively.

**Example 6.** The problem in the previous example is further illustrated by the following translations:

```
It is a book → O bir kitap+DIr
Is it a book → O bir kitap mI+DIr
```

Although the two translations exhibit similarities, they differ in their sentence structures. The first translation is *affirmative* while the second translation is *interrogative*. Therefore, the existing similarities are not considered and no rules are learned.

**Example 7.**

```
It is rain+PRES outside
     → Dışarı+DA yağmur yağ+IyOr
Singing in the rain is interesting
     → Yağmur+DA şarkı söyle+mAk ilginç+DIr
```

The similarity in `rain` is used in two different categories in the two examples, i.e. a *noun* in the first sentence and a *verb* in the second sentence. Therefore this similarity does not yield a common sentence pattern between the sentences. The target sentences also do not show any structural similarity. The surface similarity in `yağmur` is a constituent of the verb phrase in the first sentence, while it is a constituent of the noun phrase in the second sentence. Therefore, the learning condition is not met and no rules are learned from these examples.

## 3 Translation

The translation rules learned by the PATRELA algorithm can be used in the translation. The outline of the translation process is given below:

1. First, the parse tree for the source sentence to be translated is derived.
2. The parse tree of the source sentence is compared to the GPT rules of the same sentence structure in the memory in search for the most specific GPT rules. Any GPT rule that differs in any terminal (non-variable) node with the source sentence is discarded. Specifity of a GPT rule is measured in terms of the number of terminal nodes.
3. For each selected (most specific) GPT rule, its variables are tried to be instantiated with the OPT rules for the corresponding values in the source sentence. If all the variables are instantiated, then these OPT rules are applied to the GPT rule to generate the translation in the target language.

As an example for translation, consider the input sentence "I gave the pencil." Its list representation is

[S,[NP,[PN,I]],[VP,[VERB,[V,give],[T,+PAST]],[NP,[DET,the],[N,pencil]]]].

Among the example rules derived above, the following GPT rules have the same sentences structure and the terminals nodes match:

$$\texttt{I } \square^S_V\texttt{+PAST the } \square^S_N \rightarrow \square^T_N\texttt{+I } \square^T_V\texttt{+DI+Im}$$
$$\square^S_{PN}\texttt{ give+PAST the } \square^S_N \rightarrow \square^T_N\texttt{+I ver+DI+}\square^T_{PN}$$

Since both of these rules have three terminal nodes, both of them are selected as the most specific GPT rules matching the source sentence. For the first GPT rule the variables $\square^S_V$ match `give` and $\square^S_N$ match pencil, respectively. In order to use that GPT rule OPT rules for *verb* `give` and *noun* `pencil` are sought. Having found the rules

```
give → ver
pencil → kalem
```

the sentence `Kalem+I ver+DI+Im` is generated. The surface level representation of this sentence is "Kalemi verdim." When the second GPT rules is selected the same sentence will be produced. If the source sentence is ambiguous, then one translation of each interpretation is generated.

## 4 Conclusion

In this paper, we have presented a model for learning translation rules between a source and a target language. We integrated this model with an example-based translation model to translate from English to Turkish. The major contribution of this paper is to eliminate the need for manually encoding the translations, which is a difficult task for a large corpus. However the difficulty with this approach is its need for a bilingual parsed corpus.

We applied machine learning techniques into the domain of natural language processing (NLP) to achieve the task of learning translation rules between languages. Our main motivation was that the underlying inference mechanism is compatible with one of the ways humans learn languages, i.e. learning from examples. We believe that in everyday usage, humans learn general sentence patterns, using the similarities and differences between many different example sentences that they are exposed to. This observation lead us to the idea that a computer can be trained similarly, using analogy within a corpus of example translations.

The accuracy of the translations learned by this approach is quite high with ensured grammaticality. Given that a translation is carried out using the rules learned, the accuracy of the output translation critically depends on the accuracy of the rules learned.

We do not require an extra operation to maintain the grammaticality and the style of the output, as in Kitano's EBMT model [5]. The information necessary to maintain these issues is directly provided by the translation templates.

The model that we have proposed in this paper may be integrated with an intelligent tutoring system (ITS) for second language learning. The parse tree representation in our model provides a level of information that may help in error diagnosis and student modeling tasks of an ITS. The model may also be used in tuning the teaching strategy according to the needs of the student by analyzing the student answers analogically with the closest cases in the corpus. Specific corpora may be designed to concentrate on certain topics that will help in student's acquisition of the target language. The work presented by this paper provides an opportunity to evaluate this possibility as a future work.

## References


1. Arnold D., Balkan L., Humphreys R.Lee, Meijer S. Sadler L.: *Machine Translation*, NCC Blackwell (1994).
2. Carbonell, J.G.: Derivational Analogy: A Theory of Reconstructive Problem Solving and Expertise Acquisition. In Jude W. Shavlik and Thomas G. Dietterich (eds), *Readings in Machine Learning*, Morgan Kaufmann (1990) 636–646.

3. Furuse, O. and Iida, H.: Cooperation between Transfer and Analysis in Example-Based Framework, *Proceedings of COLING-92* (1992).
4. Hammond, K.J.: (Ed.) *Proceedings: Second Case-Based Reasoning Workshop.* Pensacola Beach, FL:Morgan Kaufmann, (1989).
5. Kitano, H.: A Comprehensive and Practical Model of Memory-Based Machine Translation. In Ruzena Bajcsy (Ed.) *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence*, Morgan Kaufmann V.2 (1993) 1276–1282.
6. Kolodner, J.L.: (Ed.) *Proceedings of a Workshop on Case-Based Reasoning.* Clearwater Beach, FL:Morgan Kaufmann (1988).
7. Nagao, M. A.: Framework of a Mechanical Translation between Japanese and English by Analogy Principle (1985).
8. Ram, A.: Indexing, Elaboration and Refinement: Incremental Learning of Explanatory Cases. In Janet L. Kolodner (ed.), *Case-Based Learning*, Kluwer Academic Publishers (1993).
9. Reisbech, C. and Schank, R.: *Inside the Case-Based Reasoning*, Lawrence Elbaum Associates (1990).
10. Sato, S.: *Example-Based Machine Translation*, Ph.D. Thesis, Kyoto University (1991).
11. Sato, S. and Nagao, M.: The Memory-Based Translation, *Proceedings of COLING-90* (1990).
12. Siskind, J.M.: Lexical Acquisition in the presence of Noise and Homonymy, *Proceedings AAAI-94* (1994) 760–766.
13. Stanfill, C. and Waltz, D.: Toward Memory-Based Reasoning. *CACM*, Vol.29, No.12 (1991) 185-192.
14. Sumita, E. and Iida, H.: Experiments and Prospects of Example-Based Machine Translation, *Proceedings of the 29th Annual Meeting of the Association for Computational Linguistics* (1991).

This article was processed using the LaTeX macro package with LLNCS style