구문 단위 예문을 이용한 한-영 변환

김 중혁* 서 정연 김 길창
한국과학기술원 전산학과

Transferring Dependency Structure from Korean to English Using Phrase-Unit Examples

JongHyeog Kim Jungyun Seo Gil Chang Kim
Department of Computer Science
Korea Advanced Institute of Science & Technology

E-mail: {lucifer | seo | gckim}@csking.kaist.ac.kr

요 약

변환 방식의 기계 번역이란 변환 규칙을 사용하여 원시 언어의 중간 단계 표현으로부터 목적 언어의 중간 단계 표현을 구하는 것이다. 변환 방식에는 규칙을 사용하는 변환과 예문을 기반으로 하는 변환이 있다.

규칙을 사용하는 변환 시스템은 시스템의 확장성에 문제가 있는 반면에 예문을 기반으로 하는 변환 시스템은 견고하고 시스템의 확장성이 좋으며, 특정 영역에 대한 번역에 매우 적합하다는 등의 장점을 가진다.

예문을 기반으로 하는 변환 시스템에서 예문의 표현은 문장 단위로 할 수도 있고, 혹은 구문 단위로 할 수도 있다. 예문을 문장 단위로 정의할 경우에는 매우 다양한 형태의 문장들을 포함해야 하므로 예문의 갯수가 매우 많아지며, 그다지 견고하지 못하게 된다. 반면에 구문 단위를 예문으로 하는 경우는 융통성이 좋고 견고하지만, 부분적으로 번역된 여러 구문들을 결합해야 하는 문제가 있다.

이러한 결합 문제를 해결하기 위한 한-영 예문의 대응 함수를 정의하고, 이 결합 함수를 사용하여 정의한 예문을 통하여 호텔 영역에서의 한-영 변환을 하는 시스템을 구현하였다.

1. 서론

기계 번역에 있어서 변환 방식이란 원시 언어의 중간 단계 표현으로부터 변환 규칙을 사용하여 목적 언어의 중간 단계 표현을 얻어내는 것을 말한다. 중간 단계 표현이란 원시 언어에서는 해석중의 중간 결과이고, 목적 언어에서는 생성중의 중간 결과를 말한다. 일반적으로 해석의 과정이 복잡해 지면 변환이 간단해 지고, 반면에 해석을 간단히 하면 변환의 과정이 복잡해 진다[1]. 한 예로, 돌면 한국어-일본어 변환에는 형태소 수준의 변환 방식을 사용할 수 있다. 즉 한국어의 형태소 해석 결과로부터 일본어의 형태소 해석 결과를 적절한 변환 규칙을 사용하여 얻어낼 수 있다. 그러나 한국어-영어의 변환에서는 한국어-일본어의 변환과는 달리 단어의 순서가 심하게 변하기 때문에 형태소 수준의 변환을 할 수는 없고 따라서 구문 분석이 된 트리 구조에 대해서 변환을 행해야 한다.

변환에서는 크게 두가지 일을 해야 하는데, 첫째로 원시 언어로부터 목적 언어의 대역어를 선택해야 하고, 둘째로, 원시 언어의 구조를 목적 언어의 구조로 변환해야 한다. 규칙을 사용하는 변환 시스템들은 규칙을 사용해서 원시 언어의 중간 단계 표현을 목적 언어의 중간 단계 표현으로 바꾼다. 규칙을 적용함에 따라 원시 언어의 중간 단계 표현이 단계적으로 목적 언어의 중간 단계 표현으로 바뀌게 되므로 이 변환 과정중에 사용되는 규칙들은 서로간에 적용 순서가 존재하게 된다. 이렇게 규칙의 적용에 순서가 있다는 것은 규칙들이 서로 독립적이지 못하다는 것이고 따라서 새로운 규칙을 만들기가 매우 어려워지며 변환 문법을 기술하기 또한 매우 어렵다[3, 5, 14].

이러한 규칙을 이용한 변환에서의 문제점을 해결하기 위한 여러 시도들이 있어 왔고 이들 중 하나가 최근 일본어-영어의 번역에서 시도되고 있는 예문에 의한 변환이다 [5, 9, 7, 15].

예문에 의한 변환이란 예문 데이타 베이스로부터 입력문과 가장 근사한 예문을 선택하고, 선택된 예문에 표현된 변환 정

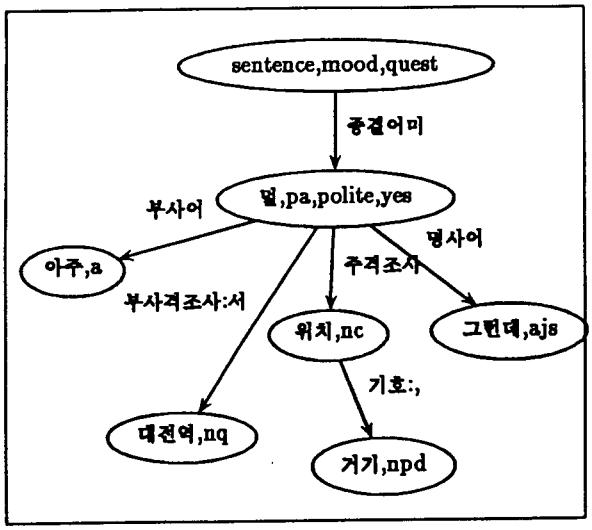

그림 1: 입력 문장 : '그런데, 거기 위치가 대전역에서 아주 멉니까?' †

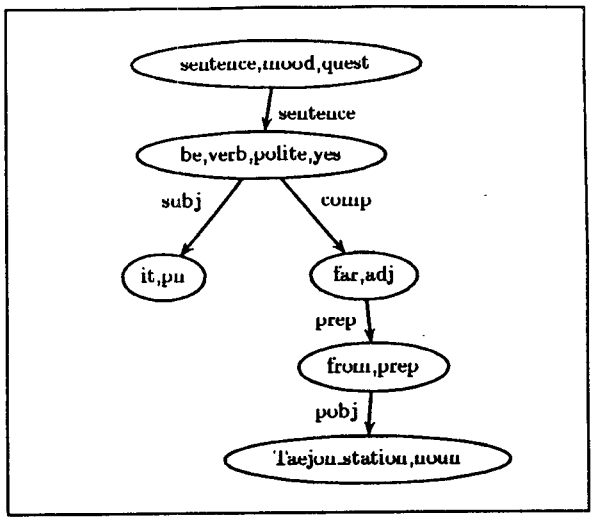

그림 2: 번역 문장 : 'Is it far from the Taejon station?'

보를 입력문에 적용하는 것이다. 즉, 일반화된 규칙을 기술하여 이를 이용한 변환 시스템을 구성하지 않고 원시 언어-목적 언어간에 대응된 예문을 규칙으로 사용하여 변환 시스템을 구성하는 것이다.

본 논문은 한국어-영어의 변환을 대상으로 하고 있으며, 변환 시스템의 입력 구조는 한국어의 의존 트리이고, 변환 결과로써 영어의 의존 트리를 생성한다. 한국어의 의존 트리는 실질형태소(체언, 수식언, 감탄사, 용언 어간)를 노드로 하고, 형식형태소(조사, 어미)를 에지로 하며, 영어의 의존 트리는 영어 단어를 노드로 하고, 단어간의 문법적인 관계를 에지로 한다. 한국어와 영어의 의존 트리의 예는 각각 그림 1, 2와 같다. 변환 시스템의 적용 영역은 호텔 예약 영역으로 하였다.

변환에 사용할 예문을 어떻게 표현할 것인지를 3.장에서 설명하고, 구문 단위 예문을 사용한 변환 시스템이 어떻게 구성되고 변환의 과정이 어떻게 되는지를 5.장에서 설명하고 4.장에서 결론적으로 예문을 사용한 변환 시스템의 장· 단점이 무엇인지를 설명하겠다.

2. 예문의 단위

예문은 변환 규칙을 대신하는 것으로써 한국어-영어의 문법적인 차이와 대역어에 관한 정보가 포함되어 있어야 한다.

예문을 결정할 때 고려해야 할 점은 예문을 문장 단위로 표현할 것인가 아니면 구문 단위로 표현할 것인가 하는 것이다. 예문을 문장 단위로 표현하는 경우는 다양한 문장 형태에 대한 번역예를 모두 가지고 있어야 하기 때문에 매우 많은 수의 예문이 필요하게 된다. 실제로 모든 다양한 문장 형태에 대한 예문 데이타 베이스를 구성하기는 현실적으로 불가능하다.

반면 예문을 구문 단위로 표현하는 경우는 문장 단위로 표현할 때와는 달리 문장의 요소들을 예문으로 선택하므로 필요로 하는 예문의 갯수가 문장 단위로 표현할 때보다는 적다. 그러나 구문 단위로 예문을 표현하는 경우는 입력 문장을 구문 단위로 나누어 예문을 찾아야 하며 선택된 여러개의 구문 단위 예문의 정보를 결합해야 하는 문제가 생긴다.

3. 번역 예문

번역 예문은 구문 단위이며 한국어의 의존 트리와 영어의 의존 트리, 그리고 한국어 의존 트리와 영어 의존 트리간의 대응으로 표현된다. 예문의 표현 형식은 다음과 같다.

$$\text{번역 예문} = [G_m, M, G_e]$$

여기서 G_m은 한국어의 의존 트리이고, G_e는 영어의 의존 트리이며, M은 한국어 의존 트리 노드에서 영어 의존 트리 노드로의 대응 함수이다.

한국어 문장의 집합 $\mathcal{K}$가 있고, 서브 트리 집합을 구하는 함수 $\mathcal{S}$가 있다면 입력문의 의존구조 G_m은 다음과 같다. $\mathrm{Dep}(\mathcal{K})$는 $\mathcal{K}$의 의존 트리를 나타낸다.

$$G_m \in \mathcal{S}(\mathrm{Dep}(\mathcal{K}))$$

대응 함수 M은 한국어 의존 트리와 영어 의존 트리 상에서의 단어 대응을 나타내야 하고, 구문 단위 예문을 사용하므로 예문의 결합을 위한 정보로도 사용되어야 한다. 구조상의 변환 정보는 G_m과 G_e의 트리 구조로써 표현된다.

대응 함수 M은 (m, l, f, lf)의 네가지 종류가 있는데 이들은 각각 변환에서의 한국어-영어 간의 서로 다른 대역어 정보와 결합 정보를 가지고 있다. 각각의 의미는 다음과 같다.

- m(결합 대응:mapping) : 단어간의 대응 관계는 있고 단

†mood : 평서문인지 의문문인지의 정보, pa : 형용사, a : 부사, nq : 고유 명사, nc : 보통 명사, ajs : 문장 접속 부사, npd : 지시 대명사

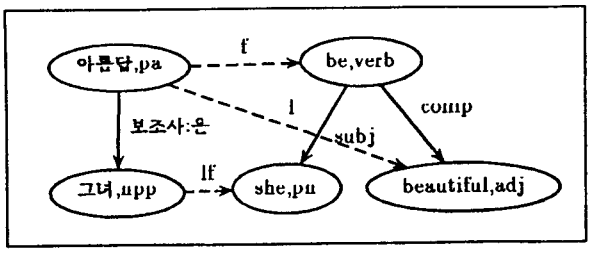

그림 3: 번역 예문 : '그녀는 아름답다.' ⇒ 'She is beautiful.'

순히 결합 정보만을 나타낸다.

- l(어휘 대응:lexical mapping) : 한국어-영어의 의존 트리 상에서 어휘적 대응이 있는 한국어 의존 트리 노드에서 영어 의존 트리 노드로의 대응이다.
- f(자질 대응:feature mapping) : 한국어-영어의 의존 트리 상에서 자질의 전이가 일어나는 한국어 의존 트리 노드에서 영어 의존 트리 노드로의 대응이다.
- lf(어휘,자질 대응:lexical and feature mapping) : 한국어-영어의 의존 트리 상에서 어휘적 대응과 자질의 전이가 동시에 일어나는 한국어 의존 트리 노드에서 영어 의존 트리 노드로의 대응이다.

lf 대응은 일상적인 한국어-영어 간의 대응이다. 그림 3에서 한국어 '그녀'와 영어 'she'간의 대응이다.

l, f대응은 그림 3에서와 같이, '아름답다'가 'be beautiful'로 번역될 때 'be'는 동사적인 자질들을 '아름답다'로부터 얻어내고, '아름답다'의 영어 대역어는 'beautiful'이 된다. 따라서 '아름답다'에서 'be'로는 f 대응을, 'beautiful'로는 l 대응을 만들어 준다.

한국어 의존 트리-영어 의존 트리의 대응에서 영어 의존 트리의 어휘를 결정할 때 모든 영어 의존 트리 노드는 오직 하나의 한국어 의존 트리 노드를 참조해야 하며, 한국어 의존 트리 노드로부터 영어 의존 트리 노드로 자질을 넘겨줄 때, 모든 한국어 의존 트리 노드는 오직 하나의 영어 의존 트리 노드로 자질을 넘겨줘야 한다.

따라서 M은 다음과 같은 조건을 만족해야 한다. root(G)는 의존 트리 G의 루트 노드를 가리킨다.

1. root(G_c) = M(root(G_m)) 이다. 이는 의존 트리에서 임의의 노드에 대한 지배소가 하나만 존재하기 때문에 이런 제약을 만족시키기 위한 대응이다.
2. G_c의 임의의 노드 a_c에 대하여 $M(a_c)$가 존재하고, $M = m$ 이거나 $M = lf$이면 M은 a_c에 대한 유일한 대응 함수이다.
3. G_c의 임의의 노드 a_c에 대하여 $M(a_c)$가 존재하고, $M = l$(혹은 $M = f$)이면 $\hat{M} = f$(혹은 $\hat{M} = l$)인 $\hat{M}(a_c)$가 존재할 수 있다.

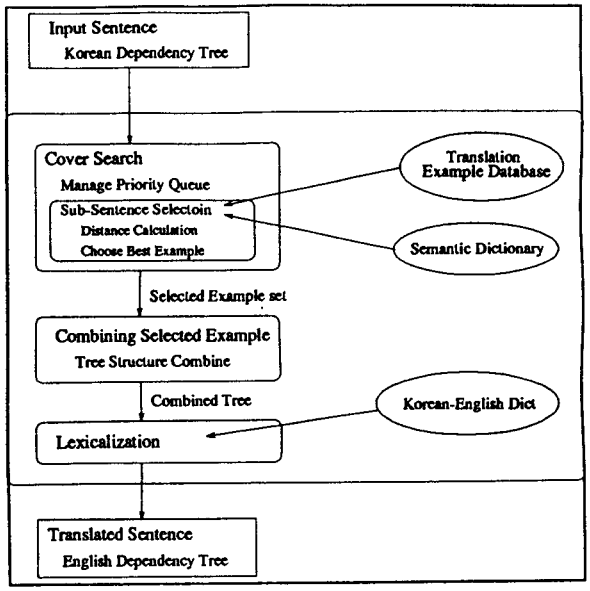

그림 4: 전체 시스템 구성도

4. G_c의 임의의 노드 a_c, b_c에 대하여 $M = l$ 이거나 $M = lf$이고 $M(a_c) = M(b_c)$ 이면, $a_c = b_c$이다.

4. 변환

변환 시스템은 그림 4와 같이 예문 찾기(Cover Search), 결합(Combining), 대역어 선정(Lexicalization)의 단계로 구성된다.

예문 찾기 단계에서는 주어진 한국어 입력 문장에 대한 변환 정보를 가지고 있는 예문들의 집합을 찾고, 결합 단계에서 이들 예문 집합을 결합하며, 대역어 선정 단계에서 한-영 대역어를 결정한다.

4.1 예문 찾기(Cover Search)

Definition 1 (대응 예문) T를 한국어의 의존 트리라 하고, $P = [G_m, M, G_c]$를 번역 예문이라 할 때, 다음의 조건을 만족하는 G_m의 에지에서 T의 에지로의 일대일 함수 $\mathcal{F}$가 존재한다면 예문 P를 입력문 T에 대한 대응 예문이라고 하고, $\mathcal{F}$를 P에 대한 예문 대응 함수라 한다.

1. G_m의 두 에지 a, b의 시작 노드가 같다면 $\mathcal{F}(a)$, $\mathcal{F}(b)$의 시작 노드도 같다.
2. P의 두 에지 a, b에 대하여 a의 시작 노드와 b의 끝 노드가 같다면 예문 $\mathcal{F}(a)$의 시작 노드와 $\mathcal{F}(b)$의 끝 노드가 같다.

Definition 2 (대응 예문 집합(Cover)) T를 한국어의 의존 트리라고 하고, $C = \{(P_1, \mathcal{F}_1), (P_2, \mathcal{F}_2), \ldots, (P_k, \mathcal{F}_k)\}$ 이라고 하자.

```
input: T(Input Sentence);
output: C(Cover);

Q : Queue;
while do
        Q ← [T, NULL, 0];
        repeat
              [T_rest, Cover_tmp, Cost_tmp]=delete(Q);
              if T_rest = NULL then
                Cover ← Cover_tmp;
                exit while;
              for Sub∈sub_tree(T_rest) do
                [E, Cost] ← best_example(Sub);
                add(Q, [T_rest−Sub,
                        Cover_tmp+E,
                        Cost_tmp+Cost])
        until S = NULL;
end do
```

그림 5: 대응 예문 찾기 알고리즘

$1 \le i \le k$인 i에 대하여 $P_i = [G_m^i, M^i, G_c^i]$가 T에 대한 대응 예문이고 $\mathcal{F}_i$가 P_i에 대한 예문 대응 함수일때, 다음과 같은 조건을 만족하면 C를 대응 예문 집합이라 한다.

1. $1 \le i,j \le k$ 이고, $([G_m^i, M^i, G_c^i], \mathcal{F}^i)$, $([G_m^j, M^j, G_c^j], \mathcal{F}^j)$에서 a가 G_m^i의 에지이고 b가 G_m^j의 에지일때, $\mathcal{F}^i(a) = \mathcal{F}^j(b)$이면 $i = j$이고, $a = b$이다.
2. T에 속하는 임의의 에지 a에 대하여 $a = \mathcal{F}^i(x)$이면 $([G_m^i, M^i, G_c^i], \mathcal{F}^i)$, $1 \le i \le k$ 이고, x가 G_m^i의 에지인 i가 존재한다.

정의 1은 대응 예문이란 한국어 입력문의 의존 트리 T의 구문 단위와 대응되는 예문을 말하는 것이고, 정의 2는 T의 대응 예문 집합은 T의 한 에지에 대응되는 대응 예문의 에지가 존재하며, 오직 하나만 되도록 대응 예문들을 선택하여 구성한 것이다.

한국어 입력 문장에 대한 변환 정보를 가지고 있는 예문들의 집합을 찾기 위해서는 대응 예문 집합을 찾으면 된다. 대응 예문 집합을 찾는 알고리즘은 우선 순위 큐를 사용하여 최적의 대응 예문 집합을 찾도록 한다. 알고리즘은 그림 5와 같다.

대응 예문들은 다음에 설명하는 거리에 의해 우선 순위를 부여받게 된다.

입력문 T의 임의의 서브 트리 S_T에 대한 최적의 대응 예문은 S_T의 대응 예문 $P_i = [G_m^i, M^i, G_c^i]$중 S_T와 가장 근사한 대응 예문이다. S_T와 P_i와의 근사도는 S_T와 G_m^i의 대응하는 노드들의 거리의 합에 반비례한다.

4.1.1 대응 거리

입력문 T의 임의의 서브 트리 S_T의 노드 a와 a에 대응하는 G_m^i의 노드 b에 대한 노드 거리D는 두 노드 a, b의 의미 거리를 D_s, 자질 거리를 D_f라고 할 때, 다음과 같이 정의된다.

$$D = D_s + D_f \times \alpha$$

α는 거리의 계산에 있어서 의미 거리와 자질 거리의 중요도를 나타내는 factor이다.

다시 의미 거리와 자질 거리는 각각 다음과 같이 정의된다.

- 자질 거리 D_f는

$$D_f(a,b) = \sum_{\text{feature}\in b} d(\text{feature}, b)$$

으로 정의 되며 다시 $d(\text{feature}, b)$은 다음과 같이 정의된다.

$$d(\text{feature}, b) = \begin{cases} 0 & \text{feature} \in b \\ \text{weight(feature)} & \text{feature} \notin b \end{cases}$$

weight(feature)는 변환에 있어서 자질 feature의 중요도를 나타낸다. 이러한 자질 거리는 a과 b의 대응 함수 M이 m인 경우에는 0이 되어야 한다.

- 의미 거리는 노드 a의 단어를 w_a, 노드 b의 단어를 w_b이라고 할때 다음과 같다.

$$D_s(a,b) = \begin{cases} 0 & w_a = w_b \\ 0.5 & w_a \text{ or } w_b \text{ is unknown} \\ 1 & w_a \text{ and } w_b \text{ are unknown} \\ d & \text{otherwise} \end{cases}$$

d는 의미 사전으로 부터 얻어낸 w_a과 w_b사이의 의미적 거리이다. 의미 사전은 type hiearchy로부터 각 단어에 대하여 의미코드가 주어져 있다. 그리고, a과 b의 대응 함수 M이 m이거나 f이면 0이 되어야 한다.

4.2 결합(Combining)

결합 부분은 예문 찾기 단계에서 구해진 대응 예문 집합을 이용하여 입력문에 해당하는 영어 의존 트리의 구조를 결정한다.

예문 찾기 단계에서 구해낸 대응 예문 집합 C에 대해서, $([G_m^i, M^i, G_c^i], \mathcal{F}^i) \in$ C이고, $a = \mathcal{F}^i(\text{root}(G_m^i))$인 노드 a에 대하여, $a = \mathcal{F}^j(b), b \in G_m^j, i \neq j$인 대응 예문 $([G_m^j, M^j, G_c^j], \mathcal{F}^j)$이 대응 예문 집합 C에 속한다.

위의 성질에 의해 예문 찾기 단계에서 구해진 대응 예문 집합은 구조적으로는 항상 결합 가능하다.

이 두 대응 예문에 대한 결합된 영어 의존 트리는 $ea = M^i(a) \in G_c^i$와 $eb = M^j(b) \in G_c^j$인 ea 노드를 eb 노드에 붙이면 얻을 수 있다.

ea노드와 eb노드를 결합할 때는 a와 ea를 대응 시켜주는 함수 M^i와 b와 eb를 대응시켜 주는 함수 M^j에 따라서 결합될 영어 의존 트리 노드의 영어 단어와 자질을 결정하게 된다.

이런 성질을 이용해서 결합 알고리즘은 그림 6와 같다.

```
input: C(Cover);
output: T(Combined Tree);

begin
    Left_C ← C;
    repeat
        (Sub1, Sub2) ← get_match(left_C);
        Left_C ← remove_match(left_C,
                              Sub1,
                              Sub2);
        if root(Sub1)∈Sub2 then
          Sub ← attach_tree(Sub1, Sub2);
        else
          Sub ← attach_tree(Sub2, Sub1);
        endfi
        Left_C ← Left_C + Sub;
    until one_element(Left_C);
end
```

그림 6: 결합 알고리즘

4.3 대역어 선정(Lexicalization)

결합된 영어 의존 트리에서 영어 대역어가 주어지지 않은 노드를 찾아서 이에 대응하는 한국어 의존 트리 상의 노드를 결정하고, 대역어 사전을 통해서 가장 일반적인 대역어를 준다.

5. 결론

본 논문에서는 호텔 예약 영역에서의 한국어-영어의 변환 시스템을 구현하였다. 예문에 의한 변환 시스템은 다음과 같은 장점을 갖는다.

- 확장성이 좋다.

  새로운 예문을 집어 넣고 싶으면 기존의 예문과는 관계 없이 새로운 규칙을 첨가하면 된다. 규칙 기반 시스템의 경우는 규칙들이 서로 독립적이지 못하므로 새로운 규칙을 집어 넣는 것은 어려운 일이다[3, 5].

- 특정 영역의 번역에 적합하다.

- 변환에 필요한 예문을 선택할 때 최적 탐색을 하므로 견고하다[14].

- 이해하기 쉽다.

  변환 규칙이 번역 예문으로 표현되므로 읽고 이해하기가 쉽다[16].

- 숙어 구문 처리를 별도의 처리 과정 없이 할 수 있다.

  숙어 구문이 구문이 번역 예문에 포함되어 있기 때문에 대응 함수를 이용하여 표현할 수 있다.

- 전문가가 필요없다.

  시스템을 확장시키기 위해서는 단지 한국어-영어 번역 예문을 규칙으로 표현하기만 하면 된다[16].

반면 다음과 같은 단점을 갖는다.

- 예문이 많이 필요하며 그에 따라 속도 저하 문제가 있다. 그러나 예문에 의한 변환 시스템은 SIMD등을 이용하거나 예문을 색인하여 속도를 높일 수 있다. SIMD 아키텍처를 사용하여 예문 탐색을 하는 경우 그 효율이 매우 높아지게 된다[16].

- 자동 예문 추출의 과정이 필요하다. 자동 예문 추출 과정이 없다면 모든 번역 예문을 사람이 추출해야 하며 이는 매우 많은 시간을 요구하게 된다[16].

참고문헌

[1] W. John Hutchins & Harold L. Somers, *An Introduction to Machine Translation*, Academic Press, 1992.

[2] Hideo Watanabe, "A Model of a Transfer Process Using Combinations of Translation Rules," *Proc. of Pacific Rim of Int. Conf. on AI '90*, pp. 215-220, 1990.

[3] Hideo Watanabe, "A Similarity-Driven Transfer System," *Proc. of the 13th International Conference on Computational Linguistics*, pp. 770-776, 1992.

[4] Hideo Watanabe, "A Method for Extracting Translation Patterns from Translation Example," *Proc. of 5th Int'l. Conf. on Theoretical and Methodological Issues in Machine Translation*, pp. 292-301, 1993.

[5] Hideo Watanabe & Hiroshi Maruyama, "A Transfer System Using Example-Based Approach," *IEICE Transactions on Information and Systems*, Vol. E77-D, No. 2, pp. 247-257, Feb. 1994.

[6] Furuse & Hitoshi Iida, "Transfer-Driven Machine Translation," *Proc. Int'l. Workshop on Fundamental Research for The Future Generation of Natural Language Processing*, pp. 95-111, 1992.

[7] Furuse & Hitoshi Iida, "An Example-Based Method for Transfer-Driven Machine Translation," *Proc. of 4th Int'l. Con. on Theoretical and Methodological Issues in Machine Translation*, pp. 139-152, 1992.

[8] Hiroyuki Kaji & Yuuko Kida & Yasutsugu Morimoto, "Learning Translation Templates from Bilingual text," *Proc. of the 13th International*

Conference on Computational Linguistics, pp. 672-678, 1992.

[9] Hiroshi Maruyama & Hideo Watanabe, "Tree Cover Search Algorithm for Example-Based Translation," *Proc. of 4th Int. Con. on Theoretical and Methodological Issues in Machine Translation*, pp. 173-184, 1992.

[10] Kazunori Muraki & Shinichi Doi, "Robust Translation and Meaning Interpretation Mechanism Based on Examples in Dictionary," *Proc. of Pacific Rim of Int'l. Conf. on AI '92*, pp. 989-993 1992.

[11] Makoto Nagao, "Some Rationals and Methodologies for Example-based Approach," *Proc. int. Workshop on Fundamental Research for The Future Generation of Natural Language Processing*, pp. 82-94, 1992.

[12] Sato & Makoto Nagao "Toward Memory-based Translation," *Proc. of Coling '90*, 3, pp. 247-252, 1990.

[13] Eiichiro Sumita & Hitoshi Iida & Hideo Kohyama "Translating with Examples: a New Approach to Machine Translation," *Proceedings of The Third International Conference on Theoretical and Methodological Issues in Machine Translation of Naural Languages*, Texas, pp. 203-212, 1990

[14] Eiichiro Sumita, "Experiments and Prospects of Example-Based Machine Translation," *Proc. of the 29th Annual Meeting of the Association for Computational Linguistics*, pp. 185-192, 1991.

[15] Eiichiro Sumita & Hitoshi Iida, "Example-Based Transfer of Japanese Adnominal Particles into English," *IEICE Transactions on Information and Systems*, Vol E75-D, No. 4, pp. 585-594, July, 1992.

[16] Graig Stanfill & David Waltz, "Toward Memory-Based Reasoning," CACM, 1986.