영한 기계번역에서 효율적인 분석을 위한 긴 문장의 분할[1)]

김 유 섭, 한림대학교 정보통신공학부, yskim01@hallym.ac.kr

A Long Sentence Segmentation for the Efficient Analysis in English-Korean Machine Translation

Yu-Seop Kim, Division of Information Engineering and Telecommunications, Hallym University, yskim01@hallym.ac.kr

요 약

본 연구에서는 영한 기계 번역에서 20단어 이상의 긴 문장을 보다 정확히 분석하기 위하여 문장을 복수개의 의미 있는 절로 분할하고자 한다. 긴 문장은 구문 분석을 시도할 때, 시간적으로 또는 공간적으로 급격히 증가하는 자원을 소모시킨다. 이러한 문제를 해결하기 위하여, 본 연구에서는 긴 문장에서 분할 가능한 지점을 인식하여 이러한 지점을 중심으로 여러 개의 절을 생성한 후, 이 절을 개별적으로 분석하고자 하였다. 문장을 분할하기 위해서 일단 문장 내부에 존재하고 있는 분할이 가능한 지점을 선택하고, 선택된 지점을 중심으로 문맥 정보를 표현하는 입력 벡터를 생성하였다. 그리고 Support Vector Machine (SVM) 을 이용하여 이러한 후보 지점의 특성을 학습하여 향후 긴 문장이 입력되었을 때 보다 정확하게 분할점을 찾고자 하였다. 본 논문에서는 SVM의 보다 좋은 학습과 분류를 위하여 내부 커널로써 다항 커널 (polynomial kernel)을 사용하였다. 그리고 실험을 통하여 약 0.97의 f-measure 값을 얻을 수 있었다.

1) 이 논문은 2004년도 한국학술진흥재단 지역대학우수과학자 지원연구 (R05-2004-000-10376-0) 에 의하여 연구되었음

1. 서 론

영한 기계 번역에서, 번역하고자 하는 문장의 길이는 번역 성능에 상당한 영향을 미치는 요소이다. 왜냐하면 문장의 길이기 길어질수록 이를 처리하는데 필요로 하는 시간과 공간이 기하급수적으로 증가하기 때문이다. 물론 현재의 컴퓨팅 능력은 지속적으로 증가하여 이러한 문제의 상당 부분을 해결할 수 있게 되었지만 아직까지도 대다수의 상용 번역 시스템에서는 여전히 이러한 문제가 존재하고 있다. [1]에서는 이러한 문제를 해결하기 위하여 복잡한 분석 규칙 대신에 문장의 패턴 정보를 활용하는 방법을 제안하였다. 또한 긴 문장을 청크로 분리하고 다양한 조건 검사를 이에 대하여 수행함으로써 문장을 절로 분할하여 분석하고자 하였다. 그러나 이 방법은 긴 문장에서 분할점을 인식하는 보다 정확한 방법을 제시하지는 못하였다.

과거 [2][3][4]의 연구는 긴 문장을 보다 정확하게 번역하기 위하여 여러 개의 의미 있는 절로 분할하고자 하였다. 그러나 이 연구들에서 제시한 방법론들은 기본적으로 사람이 분할에 필요한 조건을 생성하는 것을 전제로 하였기 때문에 생성된 분할 조건의 적용 범위가 지극히 제한적이었다. 반면 [5]의 연구는 이러한 제한적인 문제를 해결하기 위하여 개념 학습, 최대 엔트로피 모델, 유전자 알고리즘과 같은 기계 학습적인 방법론을 적용시켜 긴 문장을 분할하고자 하였다. 그러나 이 연구에서는 여러 단계에서 서로 다른 학습 알고리즘을 적용시켰는데, 이에 대한 명확한 이유나 원칙은 찾아보기 힘들고 이러한 다단계 적용이 문제를 더 복잡하게 만드는 경향이 있었다. 그리고 여기서는 분할을 구 단위까지 그리고 심지어는 단어 수준까지 세분하여 실행하고 있으며 컴마를 무조건적으로 분할의 대상으로 인정을 하고 있다. 이는 불필요한 처리로 인한 자원의 낭비와, 지나치게 자세한 분할로 인한 정확도의 하락을 초래하였다.

본 논문에서는 이러한 배경을 기반으로 하여 보다 단순한 모델링에 기초한 방법론을 찾게 되었다. 그래서 본 연구에서는 Support Vector Machines (SVM) [6] 에 기반하여 문제 영역을 단순화하여 분할을 시도하였다. 본 논문에서는 문장 분할에 필요한 문맥 정보를 품사 정보로 정의하였다. 이 품사 정보를 근간으로 입력 벡터를 생성하여 SVM을 통하여 학습과 분류를 시도하였다. 또한 SVM의 커널 함수로는 3의 정도 (degree)를 가진 다항 커널 함수 (Polynomial Kernel Function) 을 사용하였다. 이러한 방법론을 적용한 결과, recall에서의 제약을 가정하고 약 0.97의 f-measure 값을 얻을 수 있었다.

2장에서는 SVM의 기본 개념과 SVM에 사용되는 커널 함수에 대하여 간단하게 살펴보고, 3장에서는 전체 분할 시스템의 개요를 살펴본 후, 예제를 이용하여 전체 흐름을 설명한다. 4장에서는 실험 결과와 이에 대한 평가를 하고 마지막으로 결론을 5장에서 설명한다.

2. Support Vector Machine

2.1 Support Vector Machine 의 개요

Support Vector Machine(SVM)은 최소의 일반화 에러로 나타나게 하는 최적의 분류 평면 (Separating Hyperplane)을 결정하는 기법이라고 볼 수 있다. 일반적으로 선형적으로 분류 가능한 문제의 분류식은

$$f_{w,b} = sign(w \cdot x + b)$$

와 같이 나타낼 수 있다.

SVM에서 최적의 분류 평면은 서로 다른 클래스들을 구분하는 최대 마진(margin) 사이에 존재한다고 본다. 입력벡터 x_i에 대한 클래스 레이블

(label)이 y_i라고 할 때, 최적의 분류 평면은 다음의 제약조건 최소화를 만족해야 한다.

$$Min : \frac{1}{2} w^t w \; where \; y_i (w \cdot x_i + b) \geq 1$$

선형적으로 구분이 불가능한 경우, 위의 최소화 조건은 오분류 데이터를 허용하기 위해 수정되어야 한다. 수정된 식에서 soft margin 분류기가 어느 정도의 에러를 허용하는 대신 제약조건의 위반의 측정치로 새로운 변수인 c를 포함한다. 그리고 a_i가 라그랑지(Largrangian) 계수일 때,

$$Min : L(W) = \frac{1}{2} \cdot < w.w > - \sum a_i y_i [(< w, \varphi(x_i) > + b) - 1]$$

$$0 \leq a_i \leq C$$

$$\frac{\partial L}{\partial b} = 0$$

$$\frac{\partial L}{\partial w} = 0$$

이다. 여기서, C는 ζ_i의 가중치이며, $\phi(\cdot)$는 입력 공간을 보다 고차원의 공간으로 매핑하는 비선형 함수이다. 이 때, 위 식의 첫 번째 항을 최소화하는 것은 VC 차원을 최소화 하는 것과 같은 효과이다. 위 식을 풀기 위해서 라그랑지 방법을 이용하여 다음과 같이 변형한다.

$$Max : W(a) = \sum a_i - \frac{1}{2} \cdot \sum a_i a_j y_i y_j K(x_i, x_j)$$

$$0 \leq a_i \leq C$$

$$\sum a_i y_i = 0$$

이 때 $K(x_i, x_j) = \langle \phi(x_i), \phi(x_j) \rangle$ 인 커널 함수이다.

두 클래스(binary class) 분류에서의 임의의 입력 벡터 X에 대한 분류 함수는 다음 식과 같다.

$$f(X) = sign(\sum_{i=1}^{l} y_i a_i (x_i \cdot X) + b)$$

2.2 커널 함수

SVM에서는 고차원의 기본 함수들을 구축하는 것이 매우 중요하다. 이 때 구축되는 고차원의 공간을 자질 공간 (feature space) 이라고 하고 이전의 공간을 입력 공간 (input space) 이라고 한다[7]. 만일 입력 공간을 그대로 자질 공간으로 사용하여 최적의 hyperplane을 찾고자 한다면, 이 때의 커널 함수 $K(\cdot)$는 입력 벡터의 단순 내적이 되는데 이 때의 커널 함수를 선형 커널이라고 한다. 선형 커널은 여러 커널 함수 중에서 가장 단순한 형태의 함수이다.

그러나 만일 큰 집합의 사상 함수가 사용이 된다면 최적화 문제는 기본 함수들로 정의된 자질 공간에서의 새로운 내적이 필요하다. 그리고 내적 커널 $K(\cdot)$는 기본 함수 $\phi_l(\boldsymbol{x})$, $l = 1,...,m$, 를 가지고 다음과 같이 표현할 수 있다. 여기서 m은 무한할 수 있다.

$$K(x_i, x_j) = \sum_{l=1}^{m} \Phi_l(x_i) \Phi_l(x_j)$$

일반적으로 고차원 자질 공간에서의 두 자질 벡터간의 내적을 계산하는 것은 입력 공간에서의 벡터들의 support vector들 간의 커널 K 의 계산을 통하여 간접적으로 이루어진다. 커널 함수의 일반적인 형식은 다음과 같다.

$$K(x_i, x_j) = (z_i, z_j)$$

여기서 벡터 $\mathbf{z}_i$와 $\mathbf{z}_j$는 입력 벡터들의 자질 공간에서의 사상이다. 그리고 커널 함수는 자신의 비선형적인 형태를 다음 식을 이용하여 자질 공간을 생성하기 위하여 확장시킨다.

$$K(x_i, x_j) = [(x_i \cdot x_j) + 1]^q$$

여기서 q는 다항의 정도를 말한다. 이러한 형식을 가지고 있는 커널 함수를 다항 커널 함수라고 한다.

지금까지 많은 연구자들이 다양한 형태의 커널 함수를 그들의 응용에 적용시키기 위하여 개발하여 왔다. 그러나 선형 커널과 다항 커널이 그 중에서 가장 광범위하고 다양한 응용에 사용된 커널이다. 따라서 본 연구에서는 다항 커널을 SVM에 적용시켜 문제를 해결하고자 하였다.

3. 문장 분할 시스템

3장에서는 먼저 문장 분할 시스템의 전체 구조에 대하여 간략하게 알아본 후에 실세 분할 과정을 예제를 통하여 설명한다.

3.1 분할 시스템의 전체 개요

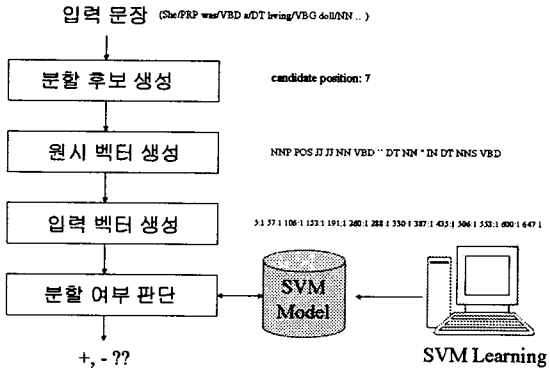

그림 1 문장 분할 시스템의 전체 구조

그림 1은 문장 분할 시스템의 전체 과정을 잘 보여주고 있다. 분할 시스템은 크게 4개의 부분으로 구성되는데, 그 구성은 분할 후보 생성, 벡터 생성, SVM 분류 그리고 학습이다.

이 시스템은 먼저 형태소 분석의 결과를 입력으로 취한다. 이 때 입력 정보로는 문장을 구성하는 각 단어의 품사 정보가 사용된다. 입력된 문장 중에서 분할이 가능한 품사를 가진 단어를 분할 후보로 선정하고 이 후보를 기준으로 하여 일정 크기의 윈도우를 설정하여 이들을 원시 벡터로 생성한다. 원시 벡터는 분류를 위한 입력 벡터로 변환이 되고 변환된 입력 벡터를 입력으로 하여 Support Vector Machines (SVM) 을 사용하여 분할 여부를 결정한다. SVM은 미리 학습 예제로 학습하여 가중치 값이 결정되며 이러한 가중치 값은 SVM model이라는 파일 형태로 저장되어 분할 여부를 판단할 때 사용된다.

3.2 예제를 이용한 분할 과정의 상세 설명

3.2.1 입력 문장

먼저 다음 문장을 보자.

Their cars weren't small enough, they didn't have the power, they were old-fashioned.

이 문장을 형태소 분석을 통하여 품사를 결정하게 되면 다음과 같은 결과가 나온다.

Their/PRP$ cars/NNS were/VBD n't/RB small/JJ enough/RB ,/, they/PRP did/VBD n't/RB have/VB the/DT power/NN ,/, they/PRP were/VBD old-fashioned/JJ ./.

여기서 사용된 품사 정보는 Penn Treebank에서 정의된 품사정보이다. 이러한 형태로 품사가 결정된 문장이 입력으로 사용된다.

3.2.2 분할 후보 생성

분할 후보란 분할이 가능한 단어를 말한다. 분할 후보는 문장에 나타난 단어들을 대상으로 결정하게 되는데 본 연구에서는 단어의 품사 정보를 가지고 후보를 선정하였다. 먼저 Penn Treebank에서 이미 구문 태그된 말뭉치 정보에서 문장의 시작을 의미하는 태그를 찾아 이들 단어들의 품사들을 수집하였다. 시작점이 되는 품사들을 빈도순으로 나열하여 상위 9개의 품사를 선정하였다 선정된 품사는 CONJ(접속사), ,(컴마), "(인용부호), WP(의문대명사), WRB(의문부사), WDT(의문한정사), IN(전치사), :(세미콜런), CC (등위접속사) 이다.

위의 예제를 본다면 7번째 단어 컴마와 14번째 단어 컴마가 분할 후보로 선정되어 향후 이 부분에서 분할이 가능한지 여부를 판단하게 된다.

3.2.3 벡터 생성

1항 단계에서 생성된 분할 후보 단어들에 대하여 각각 주변 정보를 개별 자질로 하는 하나의 벡터가 생성된다. 이 벡터를 구성하고 있는 성분들은 매우 다양한 형태를 가질 수 있다. 본 연구에서는 다음과 같은 형태로 이 벡터를 생성하였다.

$$\langle p_{i-k}, p_{i-(k-1)}, p_{i-(k-2)}, \ldots, p_{i-1}, p_i, p_{i+1}, \ldots, p_{i+(k-2)}, p_{i+(k-1)} \rangle$$

여기서 p_i는 분할 후보인 i번째 단어 w_i의 품사를 의미하며 k는 단어 w_i 주위의 문맥 정보의 범위를 뜻하는 윈도우 크기를 결정할 때 사용된다. 본 연구에서는 최적의 성능을 보이는 k를 실험적으로 찾고자 하였다.

k를 7이라 하였을 때, 위에 제시된 예문에서 생성되는 후보 입력 벡터는 다음과 같다.

candidate position 7 :
<PRP$, NNS, VBD, RB, JJ, RB, ,, PRP,
VBD, RB, VB, DT, NN, ,>
candidate position 14 :
<PRP, VBD, RB, VB, DT, NN, ,, PRP,
VBD, JJ, ., -RRB-, -RRB->

여기서 '-RRB-'는 문장이 끝나 더 이상 단어가 없을 때 벡터를 채워주기 위하여 사용되는 기호이며 비슷한 이유로 '-LRB-'가 사용되는데 이는 문장이 시작되기 이전을 표시하는 것이다.

Support Vector Machine의 입력이 되는 벡터는 수치 데이터만을 자질로 가질 수 있다. 본 연구에서는 벡터의 자질로써 품사를 사용하였는데 이 품사를 수치화하는 것이 매우 중요한 문제가 되었다. 그래서 본 연구에서는 품사를 수치 데이터로 변환하여 SVM의 입력 벡터를 생성하기 위하여 다음과 같은 방법을 사용하였다.

먼저 하나의 자질을 전체 39개의 자질을 가지고 있는 하나의 소벡터로 표현하였는데 이 소벡터는 0 또는 1의 값을 가지는 자질로 구성된다. 또한 여기서 39란 본 연구에서 사용된 품사의 가짓수를 의미한다. 그래서 각 품사마다 39개 중에 하나의 위치를 가지는 것으로 가정하였으며 해당하는 품사는 39개의 위치 중에서 자신의 위치에 1을 마크하고 나머지 자질은 0으로 하였다. 예를 들어 'NN'을 표현하기 위해서는 39개의 성분으로 구성된 소벡터에서 'NN'의 위치인 '4'번째 자질만 '1'로 표시하고 나머지는 '0'으로 표시한다.

이러한 방식으로 후보 자질 벡터는 39개의 자질을 가지는 소벡터를 $2*k$ 개 가지는 총 39*(2*k) 개의 자질을 가지는 벡터로 변환된다. 이러한 벡터는 SVM의 입력을 위하여 다음과 같이 표현되는데 이는 벡터의 희소성을 해결하기 위한 방법이라 볼 수 있다.

candidate position 7 : <34:1 69:1 97:1 142:1

190:1 200:1 259:1 310:1 331:1 386:1 411:1 455:1 489:1 525:1>
candidate position 14 : <1:1 68:1 99:1 158:1 189:1 210:1 259:1 310:1 331:1 388:1 404:1 448:1 480:1 519:1>

이 표현을 보면, 첫 번째 예의 경우 전체 546개의 자질 중에서 34, 69, 97, 142, 190, 200, 259, 310, 331, 386, 411, 455, 489, 525 번째 자질 14개만 1로 표현되고 나머지 자질은 0으로 표현되는 벡터를 보여주고 있다.

3.2.4 SVM 학습 및 분류

본 연구에서 사용된 SVM 학습 및 분류를 위한 입력 벡터의 모습은 부류(class) 값의 유무만을 제외하고는 동일한 형태를 가지고 있다. 부류 값은 학습을 위하여 필요한 값이며 이 값은 +1 또는 -1 중에 하나를 가진다. +1은 양의 예제를 의미하는 것으로 후보가 두 개의 독립된 절로 분할하는 지점으로 적합하다는 것을 뜻한다. 다음은 양의 예제 및 음의 예제의 모습을 보여준다.

양의 예제 : <1 4:1 42:1 86:1 118:1 160:1 204:1 244:1 274:1 323:1 354:1 394:1 460:1 499:1 538:1>
음의 예제 : <-1 2:1 41:1 81:1 119:1 157:1 199:1 239:1 275:1 318:1 358:1 397:1 433:1 471:1 515:1>

SVM 을 학습시키면 SVM의 매개 변수들이 최적화 되고 최적화된 분류 모델은 모델 파일에 저장되고 향후 분류 과정에서 사용된다. 분류 함수 결과 값이 0 이상인 경우에는 후보를 분할 대상 단어로 선정하고 0 미만인 경우를 분할 불가 대상 단어로 선정하였다. 분류 함수를 예제에 적용시키면 다음과 같은 결과를 얻을 수 있다.

candidate position: 7, 3.464845
candidate position: 14, 2.353343

위의 결과를 살펴보면 두 후보 모두 0 이상의 그것도 상당히 큰 편이라 할 수 있는 3.4와 2.3 이상의 값이 함수 실행결과 출력되었다. 이러한 결과는 두 후보에서 문장을 분할해야 한다는 것을 의미한다.

4. 실험 및 평가

4.1 실험 환경

본 논문에서는 53,129개의 구문 태그된 영어 문장을 사용하여 제시된 방법론을 평가하였다. 각각의 문장은 Penn Treebank 에 의하여 구문 정보를 부여받았고 이들 문장들은 Wall Stree Journal 말뭉치로부터 추출되었다. 또한 이 말뭉치로부터 62,690개의 분할점을 추출하였다. 3장에서 설명한대로 9개의 품사들을 분할 가능점으로 분류하고 이 9개의 품사를 가진 48,020개의 분할점을 추출하였는데 이는 전체 실제 분할점의 81% 정도가 된다. 이렇게 분할 가능점을 제한하게 되면 결과적으로 전체 시스템의 recall을 최대 81%로 제한하는 결과를 초래한다. 따라서 이러한 제약은 가급적이면 회피되어야 하나, 본 시스템이 기계번역의 최 앞단을 구성하는 것을 피할 수 없기에 보다 높은 정밀도를 요구한다는 점을 고려한다면 불가피한 측면이 있다고 할 수 있다.

SVM 학습과 분류를 위해서, 전체 말뭉치에서 9개 품사에 해당하는 250,303개의 분할 후보점들을 추출하였다. 이들 후보점들은 학습 예제와 검증 예제로 초기 단계부터 분리를 하였다. 전체 예제 중에서 약 17.6% 만이 양의 예제였다. 따라서 SVM이 예제에 종속적인 결과를 낳는다면 17.6%에 가까운 양의 분류를 할 것이다. 검증 예

제는 2,000개로 고정시켰으며 학습 예제의 수는 1,000개부터 50,000개 까지 다양화시켜 최적의 학습 예제 크기를 찾고자 하였다. 또한 예제 벡터의 길이도 4부터 20까지 다양화시켜 최적의 문맥 정보양을 또한 찾고자 하였다.

4.2 실험 결과

그림 2는 다항정도를 3으로 한 다항 커널을 사용하여 학습하고 분류한 결과를 보여준다.

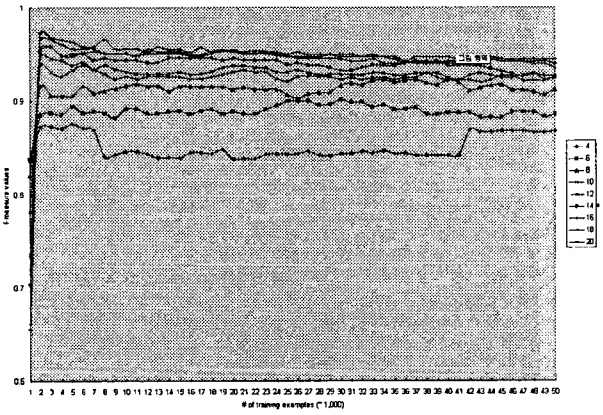

그림 2 실험 결과

실험 결과를 보면 20개의 벡터 크기를 가진 2,000개의 학습 예제를 통하여 학습을 했을 때, 최대 0.973의 f-measure 값을 보여주고 있는데, 이 값은 전체 중에서 약 81%의 분할 가능점에 대한 측정치이다. 본 실험에서는 다항 커널은 매우 작은 수의 학습 예제만을 가지고도 최적의 hyperplane을 찾을 수 있다는 점을 발견할 수 있었다. 불과 2,000개의 예제만으로도 최적의 성능을 찾는 것이 이를 설명한다. 또한 대부분의 벡터 크기의 경우에, 학습 예제 1,000개에서 2,000개 사이에서 매우 급격한 기울기를 보여주고 있다는 것은 이러한 특성이 벡터의 크기와는 큰 상관이 없다는 것을 보여준다.

다항 커널의 경우 선형 커널에 비하여 매우 많은 시간과 컴퓨팅 자원을 소모한다. 하지만 다항 커널은 매우 적은 수의 학습 예제 만으로도 최적의 hyperplane을 찾을 수 있기 때문에 자원 소모가 매우 적다는 선형 커널과 비교해도 매우 효율적이라고 볼 수 있다.

또한 매우 적은 수의 예제로 최적화된 공간을 찾을 수 있기 때문에 예제 수집에 어려움이 많은 응용의 경우에 매우 적합하게 활용될 수 있을 것이다.

5. 결 론

본 연구는 긴 문장의 효율적인 분석을 위하여 긴 문장을 의미 있는 복수개의 절로 분리하는 방법에 관한 연구이다. 긴 문장을 분할하기 위하여 분할과 관련한 정보를 단어의 품사 정보로 단순화하여 이해도를 높일 수 있었으며 Support Vector Machine을 사용하여 0.97 이상의 f-measure 값을 얻을 수 있었다.

그러나 기본적으로 분할 후보 품사를 9개로 제한하여 약 20%의 분할점은 처리의 사각지대에 있는 것은 문제가 있다. 따라서 이들 사각지대를 최소화하기 위하여 보다 다양한 품사들을 분할 후보 품사로 선정하여 작업을 진행해야 할 것이다. 또한 이 결과로 precision이 떨어질 수 있는 것을 방지하기 위하여 보다 정밀한 커널 함수를 도입해야 한다. 또한 벡터를 구성할 때, 단순한 품사 정보만이 아닌 단어의 형태 정보 또는 의미 정보 역시 고려하여 벡터를 구성하면 보다 정교한 분할이 가능할 것이다.

참 고 문 헌

[1] Roh, Y., M. H. Hong, S. Choi, K. Lee, and S. Park, "For the Proper Treatment of Long Sentences in a Sentence Pattern-based English-Korean MT System," *Procs. of*

MT-Summit 2003, 2003.

[2] Li, W., T. Pei, B. Lee, and C. Chiou, "Parsing Long English Sentences with Pattern Rules," *Procs of the 25th Conference on Computational Linguistics*, 1990.

[3] Kim, Y., and T. Ehara, "A Method for Partitioning of Long Japanese Sentences with Subject Resolution in J/E Machine Translation," *Procs. of the International Conference on Computer Processing of Oriental Languages*, 1994.

[4] Kim, S. D. and Y. T. Kim, "Sentence Segmentation for Efficient English Syntactic Analysis," *Journal of Korea Information Science Society*, **Vol.** 24, *No.* 8, 1997.

[5] Kim, S. D., "Intra-Sentenc Segmentation using Maximum Entropy Model for Efficient Parsing of English Sentences," *Journal of Korea Information Science Society*, **Vol.** 32, *No.* 5, 2005

[6] Vapnik, V. N., "The Nature of Statistical Learning Theory," *Springer*, 1995.

[7] Cherkassky, V., and F. Mulier, "Learning from Data - Concepts, Theory, and Methods," John Wiley & Sons, Inc., 1998.