뉴스 기사의 문장 요약

최동현, 신지애, 최기선
한국과학기술원 전산학과
시멘틱웹첨단연구센터

Sentence Summarization of News Articles

DongHyun Choi, Ji-Ae Shin, Key-Sun Choi
cdh4696@gmail.com, jiae@icu.ac.kr, keysun.choi@gmail.com
Computer Science Division
SWRC, Korea Advanced Institute of Science and Technology

요 약

텔레비전 뉴스에서 부제목을 만들거나, 문장을 PDA나 휴대폰과 같은 작은 화면에 출력하고 싶은 경우, 가능한 방법은 두 가지가 있다. 첫번째는 사람에 의해 직접 만드는 방식이다. 두번째는 자동화된 문장 요약 시스템을 사용하는 방법이다. 따라서 문장 요약 알고리즘은 그 중요성이 계속해서 커지고 있다. 본 논문에서는 구문 트리의 서브 트리가 변화할 수 있는 규칙을 제시하는 방법에 (1)공기 정보와 (2) 문법적으로 올바른 구조를 유지하기 위해 핵심적인 부분(주요 문법 구조) 및 같이 요약되어야 할 절을 표시하는 휴리스틱, (3)주어진 문장이 포함된 글의 제목 정보를 추가로 사용하여 문장 요약을 실행하였다. 본 시스템의 결과와 기존의 요약 방식을 비교하는 실험을 분야 전공자들에 의한 주관적 평가로 수행한 결과, 본 시스템의 알고리즘이 기존에 사용되던 구문서브트리 변환 방법보다 중요한 부분 및 문법적으로 올바른 부분을 많이 유지하는 요약임을 확인하였다.

1. 서론

"문장 요약"은 문장의 핵심 의미를 보존하면서 길이가 짧은 다른 문장으로 변형하는 문장 단위의 축약 작업이다. 요약된 문장은 독자가 빠르고 쉽게 의미를 이해할 수 있다. 그러므로 문장 요약 방법은 여러 방면으로 쓰일 수 있다. 예를 들어, 텔레비전 뉴스에서 부제목을 만들거나, 또는 문장을 PDA나 휴대폰과 같은 작은 화면에 출력할 수 있는 형태로 만들 수 있다[4]. 긴 글을 요약할 때 "글 요약"을 통하여 요약된 글의 각각의 문장을 다시 금 요약하여 좀 더 명료하게 뜻을 표현하도록 만들 수도 있다[1]. 최근 들어 이러한 여러 가지 활용 가능성들이 인지되어 문장 요약에 관한 연구가 매우 활발히 진행되고 있다.

지금까지 대부분의 연구는 하나의 문장을 개별적으로 요약하는 방식을 취하였다. 대표적인 것이 노이즈-채널을 이용한 것[2][5], 결정 트리를 이용하여 요약될 단어를 선택하는 방법[2], SVM을 이용하는 방법[3] 등이 있다. 최근에는 문장의 주변 문장도 살펴서 요약을 하려는 시도[6]도 있었다.

개별 문장만을 이용하여 요약하는 방식은 문장이 포함되어 있는 글의 내용을 상당 부분 반영하지 못한다[6]. 본 논문의 시스템은 주로 뉴스 기사의 요약을 목표로 하고 있다. 뉴스 기사의 제목은 기사의 가장 중요한 내용만으로 이루어져 있다는 가정에서 출발한다.

본 논문에서는 결정 트리를 이용하여 구문 트리의 서브 트리 변형 규칙을 활용하는 요약 방법[2]를 기반으로 하며, (1) 공기 정보(예: 동사 'expect'는 to-부정사를 필요로 한다.)와 (2)주요 문법 요소 및 같이 요약되어야 하는 절을 표시하는 휴리스틱, (3)주어진 문장이 포함된 글의 제목 정보를 사용하여 좀 더 효과적인 문장 요약을 실행하는 알고리즘을 제시한다. 본 논문은 다음과 같이 구성되어 있다: 2절에서는 문장 요약과 관련된 연구를 제시한다. 3절에서는 본 시스템에 대한 자세한 설명이 이루어진다. 4절에서는 본 시스템의 성능과 [3]의 시스템의 성능을 비교하며, 제목 정보를 이용하여 요약한 예시를 제시한다. 5절에서는 결론 및 향후에 이루어져야 할 연구에 대해서 말한다.

아래 표1에 제목 정보를 사용하여 요약한 문장과 제목 정보를 사용하지 않고 요약한 문장을 비교하였다.

표 1. 문장요약 결과: 제목사용과 비사용 비교

제목 : Execution, body dumping site found south of Baghdad **원문 :** ***The attacks*** also ***left five people hurt. The U.S. military*** also ***announced the Wednesday arrest in northern Arab Jabour of a man described as a foreign fighter*** who heads a car-bombing cell network. ***The man,*** who was not named, ***is an Egyptian-born militant*** who **came to Iraq in the early 1980s and** later joined al Qaeda fighters, ***the military said.*** **제목비사용:** *The attacks left five people hurt. The U.S. military announced the Wednesday arrest in northern Arab Jabour of a man described as a foreign fighter .The man is an Egyptian-born militant the military said .* **제목사용:** *The attacks left five people hurt. The U.S. military announced the Wednesday arrest in northern Arab Jabour of a man described as a foreign fighter .The man is an Egyptian-born militant* **came to Iraq in the early 1980s** *and the military said .*

2. 관련 연구

문장 요약과 관련된 연구로는 비주요 문법 요소에 대해서만 단어의 빈도수에 근거하여 축약하는 방법[1]과 구문 트리의 서브 트리가 변형할 수 있는 방법을 이용한 요약 방법[2]이 있다. 또한 [2]의 방법에 개체명 (named entity) 정보와 하위범주화 (subcategorization) 정보를 추가한 시도가 있다[3]. 이 외에도 노이즈-채널 모델을 적용하는 요약 방법[2][5], 최대 복잡도 모델(maximum entropy model)을 사용한 방법[7]이 있다. 본 시스템에서는 [2]의 방법을 확장하여 비주요 문법 요소를 축약하는 데 사용하였다.

3. 문장 요약 시스템

3.1 전체 구조

본 시스템의 전체 구조는 다음과 같다.

먼저 원문 요약-비교 코퍼스를 사용하여 요약에 사용될 구문 트리 변환 규칙을 찾는다. 코퍼스에서 원문에 대한 일정 구문 트리의 형태에 대하여 어떤 규칙이 적용되어 요약문장으로 변형을 이루었는지를 찾는다. 이 정보를 이용하여 구문 트리의 형태에 따라 규칙을 정하는 결정 트리를 생성한다. 생성된 결정 트리와 제목 정보, 공기 정보와 휴리스틱을 이용하여 대상이 되는 문장을 요약한다.

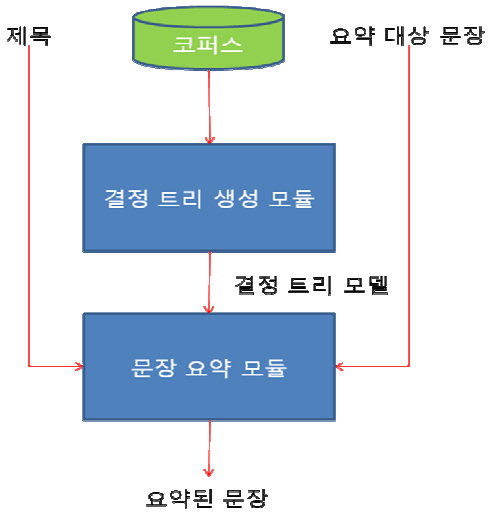

그림 1. 문장요약 시스템 제안도

3.2 결정 트리 생성 모듈

결정 트리를 생성하기 위하여 원문-요약문 코퍼스를 사용하여 요약에 사용될 구문 트리 변환 규칙을 찾는다[2]. 이 코퍼스는 각 문장과 그것의 요약된 문장으로 구성되어 있다. 원래 문장의 구문 트리와 요약된 문장의 구문 트리를 비교하여, 임의의 구문 트리의 형태에서 요약된 구문 트리의 형태를 만들기 위하여 필요한 규칙 또는 규칙의 배열을 찾는다.

규칙들은 다음 세 가지 변환으로 나타낸다.

REDUCE: 요약된 내용에 포함된 단어들 또는 부분 구문 분석 트리를 묶어 새로운 구문 분석 트리를 만든다.

SHIFT: 해당 단어가 요약된 내용에 포함된다.

DROP: 해당 단어가 요약된 내용에 포함되지 않는다.

규칙의 배열을 찾기 위하여 입력 목록(Input List)을 정의한다. 입력 목록은 요약되기 전 문장의 단어들을 문장을 이루고 있던 순서대로 가지고 있다.

요약될 내용에 포함될 단어들을 이루고 있는 서브 구문 트리를 스택에 저장한다면, 위의 세 가지 변환은 다음과 같이 재정의할 수 있다.

REDUCE: 스택에서 하나 이상의 서브 구문 트리를 꺼내어 그것들을 자식으로 하는 루트를 가진 새로운 구문 트리를 만들고, 다시 스택에 넣는다.

SHIFT : 입력 목록의 첫번째 단어를 스택에 넣어 하나의 자식을 가진 루트를 가지는 새로운 부분 구문 트리를 만든다.

DROP : 입력 목록에서 차례대로 하나 이상의 단어를 제거한다.

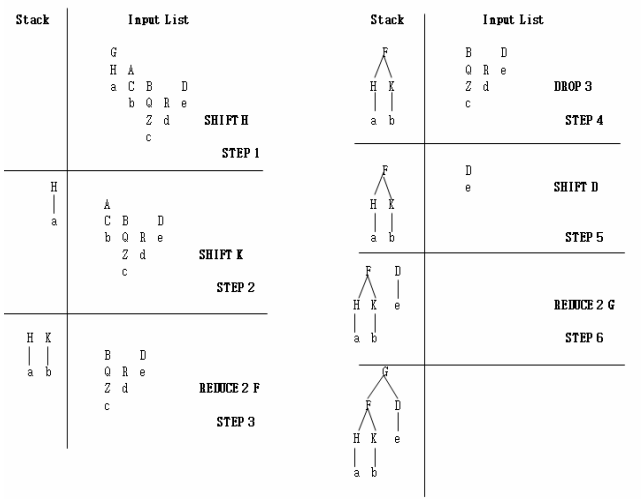

그림 2. REDUCE-SHIFT-DROP과 스택의 예

스택에 있는 부분 구문 트리와 요약된 문장의 구문 트리를 비교하여 변환의 배열을 찾는다. 이 변환의 배열은 원래 문장의 구문 트리에 순서대로 적용되었을 때 요약된 문장의 구문 트리를 만든다.

변환의 배열을 찾는 방법은 다음과 같다. 먼저, 스택의 하나 이상의 서브 구문 트리를 묶어 만든 구문 트리가 요약된 문장의 구문 트리의 일부분과 일치하면 그 부분과 같아지도록 REDUCE한다. REDUCE가 불가능할 경우. 입력 목록의 첫번째 단어가 요약된 문장의 구문 트리에 존재하지 않으면 DROP하고, 존재하면 SHIFT한다.

알고리즘은 다음과 같다.

알고리즘 1. 변환 배열을 찾는 알고리즘

입력: nowStatus는 처음에 원 문장의 구문 트리를 가지고 있다.
결과: 결정 트리를 만들기 위해 사용될 훈련 데이터

```
Queue statusQueue; // 각 과정 status들의 정보를 가지고 있다.
Status nowStatus; // 현재 status의 정보를 가지고 있다.
while(statusQueue is not empty){
    if(nowStatus' inputlist is empty){
        if(nowStatus의 스택의 원소가 요약된 문장과 같을 경우){
            status가 형성된 과정을 따라서 훈련 데이터를 만든다.
            break;
        }else{
            이 경우 요약된 문장의 단어 출력 순서가 원 문장과 다른 경우이다. 이 경우에 대해서는 생각하지 않는다.
            break;
        }
    }
    if(스택의 하나 이상의 구문 트리에 REDUCE 적용 가능한 경우){
        nowStatus에서 하나 이상의 스택의 구문 트리를 REDUCE하여 새로운 status를 만든다.
        만든 status를 statusQueue에 넣는다.
    }else if(nowStatus의 inputlist가 가지는 첫 번째 단어가 요약된 문장에 존재할 경우){
        nowStatus의 inputList에서 하나의 단어를 SHIFT하여 새로운 Status를 만든다.
        새로운 status를 statusQueue에 넣는다.
    }else if(nowStatus의 inputlist가 가지는 첫 번째 단어가 요약된 문장에 존재하지 않는 경우){
        nowStatus의 inputlist에서 가능한 최장의 단어의 목록을 DROP하여 새로운 status를 만든다.
        만든 status를 statusQueue에 넣는다.
    }
    nowStatus = statusQueue.nextStatus;
}
```

표 2. 변환 규칙의 예시

```
IF(입력 목록의 구문트리 깊이 > 0) &&
  (입력 목록의 첫 단어의 품사가 CC) &&
  (이전 1단계에 적용된 변환이 DROP) &&
  (이전 2단계에 적용된 변환이 REDUCE) &&
  (입력 순열 구문 트리의 루트의 2번째 자식의 품사가 VP)
  THEN DROP 1

IF(입력 목록의 구문트리 깊이 > 0) &&
  (입력 목록의 첫 단어의 품사가 NNS)
  THEN SHIFT NNS
```

위 알고리즘을 사용하여 찾아낸 변환의 배열을 이용하여 입력 목록과 스택의 상태가 주어졌을 때 적용될 변환을 알려주는 규칙을 찾아낸다. 이 때 규칙들이 참고하는 내용은 다음과 같이 크게 두 가지로 나뉜다.

실행적 특징 (Operational feature):이전에 적용되었던 규칙의 종류(본 논문에서는 이전 5단계까지의 규칙이 새로이 결정될 규칙에 어느 정도 영향을 미친다고 생각하였다), 스택 및 입력 목록에 있는 구문 트리의 상태를 나타낸다. 예를 들어, 스택에

있는 구문 트리의 개수, 이전 1단계에 적용된 변환의 종류 등이 있다.

원 구문 트리 특징 (Original-tree-specific feature): 구문 트리의 문법적 구성 요소를 나타낸다. 예를 들어, 입력 목록의 첫번째 단어의 품사가 CC인지 등이 있다.

위 두 가지 종류의 특징으로부터 157개의 세부적인 특징들을 찾아내고, 이들을 참고하여 다음에 이루어질 변환을 결정한다.

표 3. 결정 트리의 일부분

```
inputListTreeHeight <= 0 :
|   numberStackTrees <= 2 :
|   |   numberStackTrees <= 1 : REDUCE_1_S (10.0/1.3)
|   |   numberStackTrees > 1 :
|   |   |   wasPrevious1DROP = yes: REDUCE_1_NP (2.0/1.8)
|   |   |   wasPrevious1DROP = no: REDUCE_2_S (1016.0/11.9)
|   numberStackTrees > 2 :
                    :

inputListTreeHeight > 0 :
|   POSLevel1InputList = S: SHIFT_NN (0.0)
|   POSLevel1InputList = SBAR: SHIFT_NN (0.0)
|   POSLevel1InputList = SBARQ: SHIFT_NN (0.0)
|   POSLevel1InputList = SINV: SHIFT_NN (0.0)
|   POSLevel1InputList = SQ: SHIFT_NN (0.0)
|   POSLevel1InputList = ADJP: SHIFT_NN (0.0)
|   POSLevel1InputList = ADVP: SHIFT_NN (0.0)
|   POSLevel1InputList = CONJP: SHIFT_NN (0.0)
|   POSLevel1InputList = FRAG: SHIFT_NN (0.0)
|   POSLevel1InputList = INTJ: SHIFT_NN (0.0)
|   POSLevel1InputList = LST: SHIFT_NN (0.0)
|   POSLevel1InputList = NAC: SHIFT_NN (0.0)
|   POSLevel1InputList = NP: SHIFT_NN (0.0)
|   POSLevel1InputList = NX: SHIFT_NN (0.0)
|   POSLevel1InputList = PP: SHIFT_NN (0.0)
|   POSLevel1InputList = PRN: SHIFT_NN (0.0)
|   POSLevel1InputList = PRT: SHIFT_NN (0.0)
|   POSLevel1InputList = QP: SHIFT_NN (0.0)
|   POSLevel1InputList = RRC: SHIFT_NN (0.0)
                    :
```

이 때 만들어진 규칙들은 다음과 같은 한계점을 가지고 있다.

- 동사의 공기 정보를 포함하고 있지 않다. 예를 들어 'expect'는 주로 to-부정사와 사용되지만, 일반적으로 동사는 이를 필요치 않는다. 따라서 문법적으로 맞지 않는 변환들이 만들어지게 된다.
- 코퍼스를 이용한 방법의 한계로서, 이 규칙들이 모든 가능한 요약 방식을 찾을 수 없다.

3.3 문장 요약 모듈

본 모듈에서는 결정 트리 생성 모듈에서 찾아낸 규칙들을 이용하여 문장을 요약한다. 그러나 찾아낸 규칙들만으로 문장을 요약하면 3.2에서 언급된 한계점들로 인해 요약이 제대로 이루어지지 않는다. 따라서 규칙을 적용할 때 위의 한계점들을 해결하기 위한 추가적인 방법들을 사용한다.

3.4 휴리스틱

먼저 주요 문법 요소를 표시한다. (본 논문에서는 여러 가지 예를 관찰하여 사람이 직접 주요 문법 요소의 목록을 제작하였다). 주요 문법 요소를 표시하기 위해 먼저 구문 분석 트리의 루트를 주요 문법 요소로 표시한 다음, 주요 문법 요소의 목록에 따라 위에서 아래로(Top-Down 방식) 표시한다. 또한 어떤 동사가 요약된 문장에 포함되면(SHIFT), 그 동사의 공기 정보를 이용하여 필요한 부분을 표시한다. 표시된 부분들에 대해 DROP이 실행된다면, 해당 단어를 강제로 SHIFT시킨다. 그리고 육하 원칙 및 관사의 품사를 가진 단어가 DROP되면 해당 구는 요약에 포함시키지 않는다. 또한, 전체 문맥에서 핵심 부분이 없어지는 것을 막기 위하여 제목이 포함하고 있는 문맥에 관한 정보를 이용한다. 제목에 포함된 대명사를 제외한 명사와, 그 명사들의 반의어 (antonym), 유의어 (synonym), 전체어 (holonym), 부분어 (meronym)인 단어들은 주요 문법 요소로 표시한다. 상위어와 하위어 정보는 기존 단어의 의미와 상당히 동떨어진 경우가 대다수였기 때문에 사용하지 않았다.

표 4. 주요 문법 요소의 예

- 구문 트리에서 S의 POS를 가진 노드가 표시되어 있고, 자식이 차례로 NP와 VP의 POS를 가지고 있을 경우, NP와 VP를 표시하고, 그 자식들에 대하여 계속한다.
- 구문 트리에서 NP의 POS를 가진 노드가 표시되어 있고, 자식이 차례로 DT와 NN의 POS를 가지고 있을 경우, DT와 NN를 표시하고, 그 자식들에 대하여 계속한다.
- …

휴리스틱을 사용할 경우와 사용하지 않을 경우의 차이는 아래의 예에서 확연히 드러난다.

표 5. 휴리스틱이 필요한 요약문의 예

원 문 : ***Many debugging features,*** including user-defined break points and variable-watching and message-watching windows, **have been added**.

휴리스틱 사용: **Many debugging features have been added .**

휴리스틱 사용안함: *Many debugging features.*

알고리즘은 다음과 같다.

알고리즘 2. 문장 요약 알고리즘

입력: 제목과 관련된 명사의 목록
적용될 규칙을 결정하는 결정 트리
nowStatus는 요약 대상 문장의 구문 트리를 가지고 있다.
주요 문법 요소 목록
결과: 요약된 문장의 구문 트리

```
nowStatus의 구문 트리의 주요 문법 요소를 표시한다.
While(nowStatus의 inputlist가 단어를 가지고 있을 때){
    결정 트리를 이용하여 다음에 적용될 규칙을 찾는다.
    If(다음 규칙이 SHIFT이면){
        공기 정보 표시;
    }else if(다음 규칙이 REDUCE이면){
        REDUCE;
    }else if(다음 규칙이 DROP이면){
        If(DROP대상 단어가 표시되어 있으면){
            SHIFT;
        }else if(DROP되는 단어의 품사가 육하원칙 또는 관사){
            해당 절을 모두 DROP
        }else{
            DROP;
        }
    }
}
요약된 문장 출력
```

다음은 이 알고리즘을 이용하여 실제로 문장 요약을 수행한 결과이다. “Beyond the basic level, the operations of the three products vary widely.”가 “the operations of the three products vary.”로 요약될 때 입력 목록 및 스택의 상태를 보여주고 있다.

표 6. 요약 입력목록 및 스택 상태의 예

STACK	INPUT LIST
	S PP NP IN NP NP PP VP Beyond DT JJ NN DT NNS IN NP VBP ADVP The basic level the operations of DT CD NNS vary RB The three products widely **DROP 2 STEP 1**
	S NP NP PP VP DT NNS IN NP VBP ADVP the operations of DT CD NNS vary RB The three products widely **SHIFT DT STEP 2**
DT the	NNS PP VP operations IN NP VBP ADVP of DT CD NNS vary RB the three products widely **SHIFT NNS STEP 3**
DT NNS the operations	PP VP IN NP VBP ADVP of DT CD NNS vary RB the three products widely **REDUCE 1 NP STEP 4**
DT NP the NNS operations	PP VP IN NP VBP ADVP of DT CD NNS vary RB the three products widely **SHIFT IN STEP 5**
DT NP IN the NNS of operations	PP VP NP VBP ADVP DT CD NNS vary RB the three products widely **SHIFT DT STEP 6**
DT NP IN DT the NNS of the operations	PP VP NP VBP ADVP CD NNS vary RB three products widely **SHIFT CD STEP 7**
DT NP IN DT CD the NNS of the three operations	PP VP NP VBP ADVP NNS vary RB products widely **SHIFT NNS STEP 8**
DT NP IN DT CD NNS the NNS of the three products operations	VP VBP ADVP vary RB widely **SHIFT VBP STEP 9**
DT NP IN DT CD NNS VBP the NNS of the three products vary operations	VP ADVP RB widely **DROP 2 STEP 10**
DT NP IN DT CD NNS VBP the NNS of the three products vary operations	**FINISHED**

4. 구현 및 평가

변화 규칙 결정 과정을 표현하기 위하여 결정 트리를 사용하였다. 훈련 코퍼스로 Ziff-Davis 코퍼스 [1] 를 사용하였다. 이 코퍼스는 1067개의 요약된 문장과 원 문장의 쌍을 가지고 있다. 구문 분석 트리를 만들기 위해 스탠포드 구문 분석기[2]가 사용되었다. 동사의 공기 정보 및 제목의 명사와

[1] [2] 및 [3]에서 사용되었다.

[2] http://www-nlp.stanford.edu/downloads/lex-parser.shtml

관련된 단어들을 얻어내기 위하여 워드넷(WordNet)[3]을 사용하였다.

표 7. Ziff-Davis 코퍼스 예시

Each has strengths and weaknesses . Each has strengths and weaknesses , just like the third generation languages . Documentation is good . The MultiModem V32 's documentation is good and includes such nice touches as a listing on the bottom of the case that enumerates the possible switch settings and their meanings . Documentation is excellent . The documentation is typical of Epson quality ; excellent .

[3]의 저자의 도움을 받아 [3]의 결과와 본 논문에서 사용된 방법의 결과를 비교해 보았다. 실험을 위해 http://www.benton.org/의 2003년 기사에서 무작위로 47개의 문장을 추출하였다. [3]의 방법과 본 논문의 방법, 그리고 사람이 직접 요약하는 방법을 사용하여 세 가지의 각각 다른 요약된 문장을 얻어내었다. 사람이 직접 요약하는 방식은 중급 정도의 영어 능력을 가진 비원어민에 의해 이루어졌다. 자연언어 처리 연구에 종사하고 있는 4명의 사람들에게 두 가지 항목에 대해서 1점부터 10점까지 점수를 주도록 하였다. 조사된 항목은 다음과 같다 : 첫째, 요약된 문장이 중요한 부분을 많이 포함하고 있는가(중요도). 둘째, 요약된 문장이 문법적으로 얼마나 올바른가(문법성). 설문에 참여한 사람들에게는 세 가지 요약된 문장이 모두 컴퓨터에 의해 생성된 문장이라고 설명되었다. 이 실험에서는 제목 정보는 포함되지 않았다.

표 8. 실험결과 비교: 압축율, 중요도, 문법성

	압축율	중요도	문법성
방법1	55.03±23.58%	5.16±1.99	6.35±1.92
방법2	73.13±27.65%	6.86±2.69	7.41±2.24
사람	62.91±19.65%	6.94±1.22	8.21±0.94

방법1은 [3]의 결과이고, 방법 2는 본 시스템의 결과이다.

본 시스템의 출력 결과는 압축율은 높지 않으나, 기존 알고리즘에 비해 문법적으로 올바르다. [3]의 방법에서도 공기 정보를 사용하였으나 [3]의 방법보다 문법적으로 더 올바른 문장을 만들어낸 이유는 (1) [3]에서는 변환 규칙을 찾아내기 위하여 공기 정보를 사용하였지만, 본 시스템에서는 변환 규칙을 적용할 때 공기 정보를 사용하여 규칙을 찾아내는 과정에서 발생한 과훈련 (overfitting) 및 잡음 (Noise) 문제가 많이 줄었고, (2) 추가적인 휴리스틱으로 인한 성능 향상을 들 수 있다. 실제 사용자는 문법적으로 올바른 문장을 더 쉽게 이해할 수 있다. 따라서 기존 알고리즘에 비해 문장 요약이 실용화될 수 있는 가능성을 조금 더 높였다고 평가할 수 있다.

표1의 요약의 예를 다시 살펴보자.

제목 정보를 사용하지 않을 경우, 남자의 신상 정보에서 이라크와 관련된 부분은 언급되지 않는다. 하지만 'Iraq'가 'Baghdad'의 전체어(holonym)이기 때문에, 제목 정보를 사용할 경우 남자의 신상 정보 중 이라크와 관련된 부분도 요약에 포함되게 된다.

반면에 제목 정보를 삽입하여 요약되어야 할 문장이 요약되지 않는 경우가 있는데, 아래와 같은 경우가 한 예이다.

표 9. 제목 정보가 요약을 방해하는 예

제 목: Execution, body dumping site found south of Baghdad **원 문:** ***Carts, motorcycles and bicycles were banned from Baghdad streets Saturday evening*** **to protect Shiite pilgrims trekking south** to Karbala **for a pilgrimage.** **제목비사용:** *Carts motorcycles and bicycles were banned from Baghdad streets Saturday evening .* **제목사용:** *Carts motorcycles and bicycles were banned from Baghdad streets Saturday evening* **to protect Shiite pilgrims trekking south for a pilgrimage .**

이 경우, 제목의 'south'에 의해 기사와는 크게 관련이 없는 'Shiite pilgrims'에 관한 이야기가 삽입되게 되었다.

5. 결론 및 향후 연구

본 시스템은 기존 시스템에 비해 압축율은 낮으나, 문법적으로 더 올바른 문장을 생성한다. 실용적인 시스템을 만들기 위해서는 문법적으로 올바른 문장을 생성하는 것이 필수적이다. 본 시스템은 실

[3] http://wordnet.princeton.edu/obtain

용적인 문장 요약 시스템에 기존 알고리즘보다 더 가까이 접근하였다.

변환 규칙을 찾아내기 위해 사용되는 구문 트리의 형질이 바뀔 때 본 시스템의 성능이 상당히 큰 영향을 받음을 알 수 있었다. 가장 효과적인 문장 요약을 가능케 하는 변환 규칙을 찾아내기 위해, 사용되는 구문 트리의 형질에 관하여 추가 연구를 진행하고 있다.

감사의 글

본 논문은 정통부 및 정보통신연구진흥원의 정보통신선도기반기술개발사업의 연구결과로 수행되었습니다.

참고 문헌

[1] Hongyan Jing, *Sentence Reduction for automatic text summarization*: Proceedings of the sixth conference on Applied Natural Language processing, Seattle, 2000, pp. 310-315

[2] Kevin Knight, Daniel Marcu, *Summarization beyond sentence extraction: A probabilistic approach to sentence compression*: Artificial Intelligence, Volume 139(1), 2002, pp. 91-107

[3] Minh Le Nguyen, Akira Shimazu, Susumu Horiguchi, *Probabilistic Sentence Reduction Using Support Vector Machines*: The 20th Int. Conf. on Computational Linguistics COLING 2004, Geneva, 2004, pp.23-27

[4] Simon Corston-Oliver, *Text compaction for display on very small screens*: In Proceedings of the Workshop on Automatic Summarization, NAACL 2001, 2001, pp.89-98

[5] Jenine Turner, Eugene Charniak, *Supervised and unsupervised learning for sentence compression*: Annual Meeting of the ACL Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics, 2005, pp. 290-297

[6] James Clarke, Mirella Lapata, Modelling *Compression with Discourse Constraints*: Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, 2007, pp. 1-11

[7] Stefan Riezler, Tracy H. King, Richard Crouch, Annie Zaenen, *Statistical sentence condensation using ambiguity packing and stochastic disambiguation methods for Lexical-Functional Grammar*: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology, Volume 1, 2003, pp. 118-125