JKSCI

한국컴퓨터정보학회논문지
Journal of The Korea Society of Computer and Information
Vol. 25 No. 10, pp. 87-94, October 2020
https://doi.org/10.9708/jksci.2020.25.10.087

Efficient Keyword Extraction from Social Big Data Based on Cohesion Scoring

Hyeon Gyu Kim*

*Associate Professor, Div. of Computer Science and Engineering, Sahmyook University, Seoul, Korea

[Abstract]

Social reviews such as SNS feeds and blog articles have been widely used to extract keywords reflecting opinions and complaints from users' perspective, and often include proper nouns or new words reflecting recent trends. In general, these words are not included in a dictionary, so conventional morphological analyzers may not detect and extract those words from the reviews properly. In addition, due to their high processing time, it is inadequate to provide analysis results in a timely manner. This paper presents a method for efficient keyword extraction from social reviews based on the notion of cohesion scoring. Cohesion scores can be calculated based on word frequencies, so keyword extraction can be performed without a dictionary when using it. On the other hand, their accuracy can be degraded when input data with poor spacing is given. Regarding this, an algorithm is presented which improves the existing cohesion scoring mechanism using the structure of a word tree. Our experiment results show that it took only 0.008 seconds to extract keywords from 1,000 reviews in the proposed method while resulting in 15.5% error ratio which is better than the existing morphological analyzers.

▸ **Key words**: Big data, Social reviews, Keyword extraction, Cohesion score, Morphological analysis

[요 약]

블로그나 SNS 피드 등의 소셜 리뷰는 고객 관점의 의견이나 불만 사항을 반영한 키워드를 추출하기 위한 목적으로 광범위하게 활용되고 있으며, 최근 트렌드를 반영한 신조어나 고유명사를 포함하는 경우가 많다. 이들 단어는 사전에 포함되어 있지 않아 기존 형태소 분석기가 잘 인지하지 못하는 경우가 많으며, 동시에 상당한 처리 시간이 소요되어 키워드 분석 결과를 실시간으로 제공하는데 어려움이 있다. 본 논문에서는 응집도 점수 개념을 기반으로 소셜 리뷰로부터 키워드를 효율적으로 추출하기 위한 방법을 제안한다. 응집도 점수는 단어의 빈도수를 기반으로 계산되어 별도의 사전이 필요없다는 장점이 있으나, 띄어쓰기가 되지 않은 입력 데이터에 대해서는 정확도가 떨어질 수 있다. 이와 관련하여 본 논문에서는 단어 트리 구조를 이용하여 기존의 응집도 점수 계산 방법을 개선한 알고리즘을 제시한다. 또한 실험을 통해 제안하는 방법이 15.5%의 오류율을 보이는 동시에, 1,000개의 리뷰를 처리하는데 0.008초 정도 소요됨을 확인하였다.

▸ **주제어**: 빅데이터, 소셜 리뷰, 형태소 분석, 노이즈 리뷰 필터링, 소상공인

• First Author: Hyeon Gyu Kim, Corresponding Author: Hyeon Gyu Kim
*Hyeon Gyu Kim (hgkim@syu.ac.kr), Div. of Computer Science and Engineering, Sahmyook University
• Received: 2020. 08. 19, Revised: 2020. 10. 11, Accepted: 2020. 10. 12.

http://www.ksci.re.kr pISSN:1598-849X | eISSN:2383-9945

I. Introduction

빅데이터는 3V(Volume, Velocity, Variety) 속성을 지니는 데이터로 정의될 수 있으며, 신용카드 트랜잭션, 휴대폰 전화 및 문자 내역 등 다양한 형태의 데이터가 포함될 수 있다[1-3]. 이 중, SNS 피드 및 블로그 리뷰 등을 포함한 소셜 빅데이터는 고객 관점의 의견이나 불만 사항을 추출하기 위해 가장 대중적으로 이용되고 있다. 소셜 빅데이터는 온라인 포털 업체 등이 제공하는 오픈 API[4, 5] 등을 통해 무료로 획득 가능하며, 텍스트 형태로 구성되어 (신용카드, 휴대폰 내역 등 수치로 이루어진 데이터에 비해) 의견이나 불만 사항의 직접적인 원인을 바로 알아낼 수 있다는 점에서 선호된다.

소셜 빅데이터 분석에서는 형태소 분석을 통한 키워드 추출이 필수적으로 요구된다. 예를 들어, “화덕피자로 엄청 유명해진 그라파피자리아, 인싸들의 핫플!”이라는 리뷰를 가정해보자. 키워드 추출을 위해 먼저 형태소 분석 작업이 수행된다. 해당 단계에서는 “유명해진” 등의 형용사나 동사에 대한 활용 처리를 수행하며, 명사로부터 “로”, “들의” 등의 조사를 분리해낸다. 이후 키워드 선정 과정에서 의미가 크지 않은 조사나 “엄청” 등의 부사를 제거하며, 최종적으로 {“화덕피자”, “유명한”, “그라파피자리아”, “인싸”, “핫플”} 등이 결과값으로 반환된다.

위 예제에서도 볼 수 있듯이, 소셜 리뷰는 고유명사(“그라파피자리아”)나 트렌드를 반영한 신조어(“인싸”, “핫플”)를 포함하는 경우가 많다. 문제는 KAIST의 한나눔[6]이나 서울대의 꼬꼬마[7] 등을 포함한 기존 형태소 분석기들이 이를 잘 처리하지 못한다는 점이다. 그 이유는 형태소 분석을 위해 기존 분석기들이 사전을 이용하나, 해당 사전에는 고유명사나 신조어가 포함되지 않아 관련 명사 추출이 어렵다는 점에 기인한다. 예를 들어 꼬꼬마의 경우, 위 리뷰에 대해 {“화덕”, “피자”, “그”, “라파”, “피자”, “리아”, ...} 등의 형태로 사전에 포함된 단어 위주로 결과를 전달한다. 실제 리뷰에서 중요한 의미들은 명사에 부여된다는 점을 고려할 때, 고유명사 및 신조어를 효과적으로 추출하기 위한 방법은 필수적이라 할 수 있다.

기존 형태소 분석기를 활용할 때 제기될 수 있는 또다른 이슈는 형태소 분석에 상당한 시간이 소요된다는 점이다. 예를 들어 100개의 소셜 리뷰를 처리하는데 한나눔은 3.87초, 꼬꼬마는 135초가 소요된다(4장 참조). 만약 전국 10만개의 업체를 대상으로 업체별 평균 1,000개 정도의 리뷰를 처리해야 한다고 가정할 경우, 한나눔에서 리뷰에 대한 형태소 분석 시간으로 약 33일이 소요된다. 형태소 분석 외에 요약정보 추출이나 결과 시각화 시간까지 합칠 경우 그 시간이 더 늘어날 수 있으며, 이 경우 월 단위로 최신 정보를 유지하거나, 실시간으로 분석 결과를 제공하고자 할 때 어려움이 있을 수 있다[8].

위 문제를 해결하기 위해, 본 논문에서는 [9, 10]에서 제안한 응집도 점수(Cohesion Score)를 활용한 키워드 추출 방법을 제안한다. 응집도 점수는 별도의 사전 없이 단어의 빈도수만을 이용하여 계산 가능하며, 여러 유사한 활용 형태의 단어 중 빈도수가 높게 나타나는 단어를 원형으로 채택할 확률이 높다. 따라서 응집도 점수를 이용할 경우, 사용자에 의해 리뷰에서 자주 언급된 단어의 형태와 가까운 키워드가 추천되는 경우가 많으며, 고유명사나 신조어 처리에 있어 사전을 기반으로 단어를 너무 세분화하여 오류를 발생시키는 기존 형태소 분석기의 문제점을 효과적으로 개선할 수 있다(<표 5> 참조).

이에 반해, 온라인으로부터 수집된 실제 리뷰에 대해 [9, 10]에서 정의한 식을 그대로 적용할 경우 정확도가 현저히 낮아질 수 있는 예외 사항들을 발견하였으며, 본 논문에서는 해당 문제점과 이를 해결한 알고리즘을 함께 소개한다. 그리고 실험을 통해 기존 형태소 분석기와 키워드 추출 속도 및 정확도를 비교함으로써 제안하는 알고리즘의 효용성을 입증한다.

II. Related Work

소셜 리뷰 및 고객의 상품평을 효과적으로 분석하기 위한 많은 연구가 진행되었다. 소셜 리뷰로부터 잠재적인 광고 키워드를 추출하거나[11], 소셜 빅데이터를 이용한 영화 흥행 요인 분석[12, 13], 온라인 쇼핑몰의 상품평 분류를 통한 감성 분석 [14] 등의 주제가 소개된 바 있다. 이들 대부분은 한나눔 등의 기존 형태소 분석기를 활용하여 소셜 리뷰로부터 키워드 집합을 얻는다고 가정한 후 논의를 진행한다는 점에서 공통점이 있다.

형태소 분석의 효율을 높이기 위한 방법 관련해서도 다양한 연구가 진행되었으며, 크게 3가지 접근 방법으로 분류될 수 있다. 첫째는 [15]에서 소개된 방법으로, 형태소 분석 알고리즘을 개선하여 사전 탐색 회수를 줄이고자 하였다. 둘째는 [16]에서 논의한 방법으로 사전 구조를 개선하여 효율을 높이고자 하였으며, 셋째는 고빈도 어절에 대한 기분석 사전을 구축하여 코드 변환, 원형 복원 등의 절차적인 분석 단계를 생략하는 방법으로 [17]에서 소개되었다. 이들 방법은 모두 사전을 기반으로 형태소 분석을 수행한다는데 공통점이 있다.

한편, 소셜 빅데이터 분석에 대한 관심이 높아지면서 이들에 포함된 신조어나 고유 명사 등을 보다 정확하게 처리하기 위한 기법들이 제안되었다. 이들은 단어 추출에 있어 사전이 아닌, 단어의 빈도수 등을 바탕으로 한 통계적 기법을 이용하며, 브랜칭 엔트로피(Branching Entropy)[18] 등이 대표적이다. 해당 방법은 단어 내부에서는 불확실성, 즉 엔트로피가 줄어들고, 단어의 경계에서는 불확실성이 증가한다는 점을 이용한다. 예를 들어, 글자가 "natur"까지 등장했다면, 그 다음에 나올 수 있는 글자는 "nature" 이거나 "natural" 두 가지 경우에 국한되므로 불확실성이 낮아진다. 이에 반해 "nature"까지 등장할 경우, 다음 글자에 대한 불확실성이 다시 높아지므로, "nature"를 단어의 경계로 설정한다.

Soynlp[19]에서 이용되는 응집도 점수 역시 유사한 아이디어로부터 산정식을 산출하며, 엔트로피 값이 아닌 조건부 확률을 이용한다는 점에서 차이가 있다(3장 참조). 이외에도 [20]에서는 비지도 기계학습과 코퍼스의 단어를 이용한 형태소 분석 방법에 대해 논의하였다.

III. Proposed Method

제안하는 방법은 [9, 10]에서 소개된 응집도 점수를 기반으로 키워드를 추출한다. 먼저 3.1절에서는 기존에 소개된 응집도 점수의 동작 원리에 대해 설명하고, 실제 데이터 적용 시 생길 수 있는 문제점에 대해 논의한다. 3.2절에서는 해당 문제점을 해결하기 위해 본 논문에서 제안하는 알고리즘에 대해 설명한다.

1. Cohesion Score

응집도 점수는 단어를 구성하는 글자들만의 정보를 이용하여 단어의 경계를 판단하는 산정 방법이다. 한국어 어절의 경우, 의미를 지니는 명사, 동사, 형용사 등의 단어에 해당하는 L 파트와, 문법 기능을 위한 조사나 어미에 해당하는 R 파트로 구성되어 있다. 목표는 L + R 형태의 어절에서 L을 분리해내는 것이며, 응집도 점수는 이를 위한 식을 제공한다.

예를 들어, 입력 데이터에 네 개의 어절과 각 어절에 대한 빈도가 아래와 같이 주어졌다고 가정해 보자.

노 (200)
노란 (100)
노란색 (90)
노란색을 (9)

일반적으로, 주어진 어절이 의미가 있는 L 파트에 해당할 경우 빈도수가 높게 나타나며, 의미가 없는 R 파트를 포함할 경우 빈도수가 떨어진다. 따라서 위 예의 경우 네 어절 중 "노란색"이 키워드로 선정될 수 있도록 응집도 점수가 정의된다. "노란"의 경우도 키워드로 선정될 수 있으나, 빈도가 비슷하게 나타날 경우 일반적으로 긴 단어가 더 풍부한 의미를 지닌다는 사실로부터 "노란색"이 선정될 수 있도록 응집도 점수를 정의한다. 아래 식에서 n은 어절의 최대 길이에 해당한다.

$$f(c_{0:n}) = \left(\prod_{i=1}^{n} P(c_{0:i+1}|c_{0:i})\right)^{\frac{1}{n-1}} \quad (1)$$

위 식으로부터 응집도 점수를 계산하기 위해서는 먼저 부분문자열 x 이후 y가 나타날 조건부 확률인 P(xy|x) 값을 계산해야 한다.

P(노란|노) = 100/200 = 0.5
P(노란색|노란) = 90/100 = 0.9
P(노란색을|노란색) = 9/90 = 0.1

위 확률로부터 식 (1)을 이용하여 "노란색"에 대한 응집도 점수를 다음과 같이 계산할 수 있다.

$$f(\text{노란색}) = (P(\text{노란}|\text{노}) \times P(\text{노란색}|\text{노란}))^{0.5}$$
$$= (0.5 \times 0.9)^{0.5} \fallingdotseq 0.67$$

P(xy|x)는 1 이하의 확률값이므로, 길이가 길수록 누적곱의 값이 줄어들게 된다. 따라서 빈도가 비슷할 경우 더 긴 글자를 선택할 수 있도록, 식(1)에서는 P(xy|x)의 곱에 1/(n-1) 승을 취한다.

f(노란)과 f(노란색을) 값 역시 식 (1)을 통해 각각 0.5와 0.36으로 계산되며, 비교 결과 "노란색"의 점수가 가장 높게 나타난다. 해당 결과로부터 어절이 "노란색" + "을"의 형태로 구성된 것으로 간주하고, "노란색"을 L 값으로 반환한다.

식 (1)은 아래와 같이 간소화될 수 있다.

$$f(c_{0:n}) = \left(\frac{c_{0:2}}{c_{0:1}} \times \frac{c_{0:3}}{c_{0:2}} \times \ldots \times \frac{c_{0:n}}{c_{0:n-1}}\right)^{\frac{1}{n-1}}$$
$$= \left(\frac{c_{0:n}}{c_{0:1}}\right)^{\frac{1}{n-1}} \quad (2)$$

위 식에서 볼 수 있듯이, 응집도 점수 값을 가장 크게 좌우하는 파라미터는 단어의 길이 n임을 알 수 있다. 예를 들어, n이 일정 수준보다 커지면 (예를 들어, n > 15), 단어의 빈도수에 상관없이 응집도 점수가 항상 최대값이 되는 경우가 발생할 수 있다. 아래 표는 "그라파피자리아"를 검색 키워드로 주었을 때 온라인에서 수집된 소셜 리뷰를 대상으로 계산된 응집도 점수 예를 보여준다. 아래 표에서 freq는 리뷰에서 단어(word)의 빈도수, n은 단어의 길이,

exp2는 식 (2)에 단어의 freq와 n 값을 대입한 수식이며, score는 exp2를 통해 계산된 응집도 점수를 나타낸다.

Table 1. Frequency and cohesion score for each word in social reviews for "Grappa Pizzaria"

word	freq	n	exp2	score
그	746	1		-
그라파	82	3	$(82/746)^{1/2}$	0.334
그라파가	4	4	$(4/746)^{1/3}$	0.139
그라파피자	35	5	$(35/746)^{1/4}$	0.465
그라파피자를	2	6	$(2/746)^{1/5}$	0.306
그라파피자리아	25	7	$(25/746)^{1/6}$	0.571
그라파피자리아앤펍	2	9	$(2/746)^{1/8}$	0.477
그라파피자리아라는 동구핫플레이스	1	16	$(1/746)^{1/15}$	0.661

위 예에서 "그라파피자리아라는동구핫플레이스" 단어의 경우, 빈도수가 1임에도 불구하고 응집도 점수가 가장 높게 나타나며, 결과적으로 키워드로 추천된다. 이는 띄어쓰기가 잘 이루어지지 않은 입력 데이터에 산정식을 적용할 경우 문제가 발생할 수 있음을 보여준다. 문제는 인터넷으로부터 수집된 소셜 리뷰들을 살펴보면 띄어쓰기를 제대로 하지 않는 경우가 자주 발견된다는 점이다. 이는 키워드 추출의 정확도를 높이기 위한 산정식의 보완이 필요함을 보여준다.

2. Proposed Algorithm

제안하는 방법에서는 문제 해결을 위해 파라미터 n을 다르게 정의한다. n의 산정에 있어, 글자 수를 나타내는 절대적인 값이 아닌, 단어 간의 관계를 의미하는 상대적인 크기로 정의한다. 이를 위해 먼저 주어진 단어 집합에 대해 문자열 포함 관계를 기반으로 트리를 구성한다. 아래 그림은 <표 1>의 단어 집합에 대해 트리를 구성한 예를 보여준다.

트리에서 각각의 단어는 노드로 구성되며, 단어 A가 B에 포함될 경우, A는 B의 부모 노드가 된다. 트리 구성이 완료되면 각 단어에 대해 트리 상의 깊이(depth) 값을 결정할 수 있으며, 제안하는 방법에서는 해당 값을 n으로 이용한다. 그림에서 각 노드의 깊이에 대해 옆에 숫자로 표기하였다.

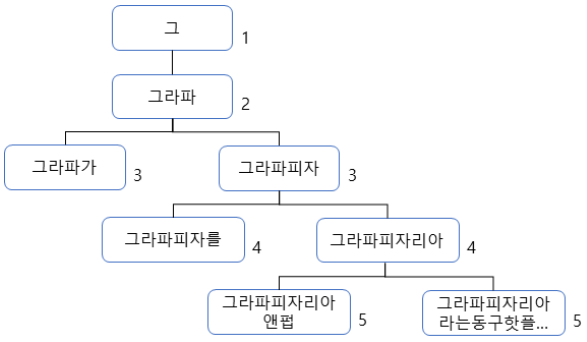

Fig. 1. Tree constructed from word bags in Table 1

<표 2>는 제안하는 방법을 통해 계산된 n값과 응집도 점수를 보여주며, 해당 방법에서는 "그라파피자리아"가 가장 높은 점수를 얻어 키워드로 채택된다. 이는 띄어쓰기가 제대로 되어있지 않은 단어가 주어지더라도 키워드 추출의 정확도가 크게 영향을 받지 않음을 보여준다.

Table 2. Frequency and cohesion score for each word in social reviews for "Grappa Pizzaria" (in our proposed method with the parameter *n* as a depth of a tree node)

word	freq	n	exp2	score
그	746	1		-
그라파	82	2	$(82/746)^{1/1}$	0.110
그라파가	4	3	$(4/746)^{1/2}$	0.073
그라파피자	35	3	$(35/746)^{1/2}$	0.216
그라파피자를	2	4	$(2/746)^{1/3}$	0.139
그라파피자리아	25	4	$(25/746)^{1/3}$	0.322
그라파피자리아앤펍	2	5	$(2/746)^{1/4}$	0.228
그라파피자리아라는 동구핫플레이스	1	5	$(1/746)^{1/4}$	0.191

위 방법을 구현하기 위한 알고리즘은 크게 두 부분으로 구성된다. 첫째는 단어 집합으로부터 각 단어에 대한 출현 빈도수를 계산하기 위한 알고리즘으로, 아래는 자바스크립트로 구현된 코드를 제시하고 있다.

```
function getWordList(wdset) {
    var list = [];
    for (var i=0; i<wdset.length; i++) {
        var found = false;
        for (var a=0; a<list.length; a++) {
            if (wdset[i] == list[a].key) {
                list[a].cnt++;
                list[a].freq++;
                found = true;
                break;
            }
        }
        if (found) continue;

        list.push({key: wdset[i], cnt: 1, freq: 1});
    }
    return list;
}
```

Fig. 2. JavaScript function to build a list of words with their counts of exact matching

입력 데이터로 주어진 wdset은 수집된 소셜 리뷰를 공백으로 분리한 토큰 집합에 해당하며, 위 함수는 wdset의 각 단어별로 출현 빈도를 계산한 배열인 list를 생성하여 결과로 반환한다. list의 각 요소는 아래 속성을 포함한다.

- key: 단어 (리뷰에서 공백을 기준으로 분리된 토큰)
- cnt: key와 동일한 단어의 출현 빈도수
- freq: key가 포함된 (key로 시작하는) 단어의 출현 빈도수

위 코드의 6번째 라인 if 문의 구현에서 볼 수 있듯이 getWordList() 함수는 주어진 단어와 정확히 일치하는 단어들의 출현 빈도수만 계산한다. 따라서 현재까지는 각 단어의 cnt와 freq 값이 동일하게 나타난다. 아래는 <표 2>의 단어 집합에 대한 getWordList()의 결과값이다.

Table 3. List generated from getWordList() for the same input data of Table 2

Word	freq	n	score
그	644	-	-
그라파	43	-	-
그라파가	4	-	-
그라파피자	8	-	-
그라파피자를	2	-	-
그라파피자리아	22	-	-
그라파피자리아앤펍	2	-	-
그라파피자리아라는동구핫플레이스	1	-	-

여기서 주의할 점은, 위 함수의 결과값이 $C_{0:1}$에 해당하는 단어 카운트를 포함하지 않을 수 있다는 점이다. 예를 들어, “그라파”에 대한 P(xy|x)를 계산하려면 “그라”와 “그”에 대한 빈도수가 집계되어야 하나, wdset에서 해당 단어들이 나타나지 않을 수 있다. 따라서 $C_{0:1}$의 빈도수는 위 알고리즘과 별도로 계산되어야 하며, 이는 wdset의 모든 단어에 대해 $C_{0:1}$으로 시작하는지 체크하기 위한 별도의 스캔 작업을 포함한다.

두 번째 알고리즘은 getWordList()에서 반환된 list 정보를 바탕으로 단어별 응집도 점수를 계산하는 부분을 포함하며, 아래에서 주석으로 표기된 바와 같이 크게 세 부분으로 구성된다.

```
function getScoredList(list) {
    // [Step.1] initialize info for tree construction
    list.sort(function(a, b) {
        return a.key.length < b.key.length ? -1 : 1;
    });
    for (var i=0; i<list.length; i++) {
        list[i].id = i;
        list[i].pid = -1;
        list[i].dep = 1;
    }

    // [Step.2] calculate a freq. and a tree depth
    // for each key
    for (var i=0; i<list.length; i++) {
        for (var j=i+1; j<list.length; j++) {
            if (list[a].key.startsWith(list[i].key)) {
                list[j].pid = list[i].id;
                list[j].dep = list[i].dep + 1;
                list[i].freq += list[j].cnt;
            }
        }
    }
    // [Step.3] calculate a cohesion score for each key
    for (var i=0; i<list.length; i++) {
        list[i].score = Math.pow(list[i].freq/list[0].freq,
                        1/(list[i].dep + 1));
    }
    return list;
}
```

Fig. 3. JavaScript function to calculate a cohesion score for each key in the list generated from getWordList()

[Step.1]은 주어진 list에 대해 key 길이를 기준으로 오름차순으로 정렬하며, 그 결과 길이가 짧은 단어가 list의 앞에 위치하게 된다. 이는 [Step.2]에서 길이가 짧은 단어를 포함한 부모 노드가 긴 단어를 포함한 자식 노드보다 우선해서 나오도록 강제함으로써, 알고리즘의 효율성을 높이는데 기여한다. 정렬 이후, list내 단어별로 트리 구성에 필요한 아래의 정보를 추가한다.

- id: 노드 아이디
- pid: 부모 노드의 아이디
- dep: 트리에서 깊이

[Step.2]는 list로부터 트리를 구성하고 freq 값을 계산한다. list의 첫 번째 요소는 가장 길이가 짧은 단어이며, 트리의 루트 노드가 된다. 따라서 루트 노드부터 순차적으로 단어의 포함 관계를 검사하여 트리에서 단어 간의 부모/자식 관계를 설정한다. 예를 들어 i번째 단어에 대해, 자신 이후에 나오는 j번째 단어가 자신을 포함하면 if 조건이 참이 된다. 이 경우 j번째 단어의 pid에 자신의 id를 할당함으로써 부모/자식 관계를 설정하며, 동시에 j번째 단어 노드의 깊이 값은 자신의 깊이 값에 1을 더하면 된다. 그리고 j번째 단어의 cnt를 자신의 freq 값에 추가한다. 이로부터 freq는 자식 노드의 카운트 수를 모두 더한 값이 되며, 결과적으로 i번째 단어가 포함된 모든 단어의 출현 빈도수를 의미하게 된다. if 조건에서 포함 관계를 알아내기 위해 startsWith() 함수가 이용되었으며, 단어 A가 B에 포함될 경우 B.startsWith(A)는 true를 반환한다.

[Step.3]에서는 이전 단계에서 얻어진 freq 및 dep 값을 이용하여 응집도 점수를 계산한다. 해당 구현에서는 식 (2)가 이용되었으며, 지수승을 계산을 위해 자바스크립트의 Math.pow() 함수를 이용하였다. <표 3>의 데이터에 대해 getScoredList()를 통해 최종적으로 계산된 freq, dep, score 값은 <표 2>와 동일하다. 해당 표에서는 dep 대신 n으로 표기되어 있으며, 이는 앞서 설명한 바와 같이 제안하는 방법이 n 값으로 dep를 이용한다는 점에 기인한다.

IV. Experimental Results

제안한 알고리즘을 검증하기 위해 온라인에서 수집된 실제 데이터를 바탕으로 키워드 추출의 정확도 및 효율성을 실험하였다. 실험 비교 대상으로는 현재 소셜 빅데이터의 키워드 추출을 위해 대중적으로 이용되고 있는 KAIST 한나눔 형태소 분석기[6]와 서울대의 꼬꼬마 분석기[7]를 선정하였다. 한나눔의 버전은 0.8.4이며, 꼬꼬마는 2.1 버전이 이용되었다. 실험은 Intel-core i5 CPU, 16G 메모리 상에서 윈도우 운영 체제를 기반으로 수행되었다.

1. Experimental Setup

실험을 위해 온라인으로부터 실제 소셜 리뷰를 수집하였으며, 리뷰 수집을 위해 네이버가 제공하는 오픈 API[4]를 이용하였다. 리뷰 수집 대상으로 울산 동구의 유명 레스토랑 10군데를 선정하였으며, 상호명 별로 수집된 소셜 리뷰를 각각의 입력 데이터 집합으로 구성하고 실험을 수행하였다.

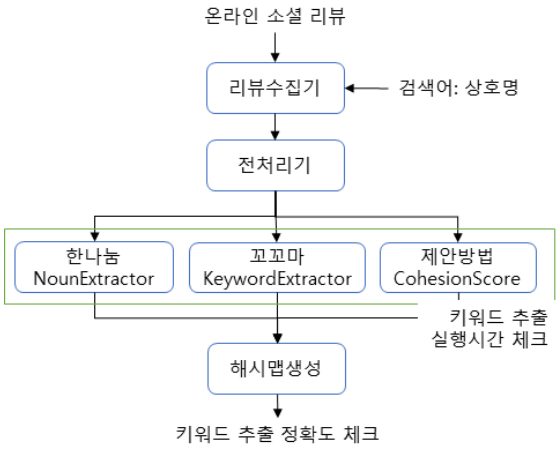

Fig. 4. Experimental setup for measuring accuracy and execution time of keyword extraction

수집된 리뷰 집합은 효율적인 형태소 분석을 위해 전처리 과정을 거치며, 해당 과정에서는 특수 기호를 제거하여 문자 및 숫자 중심의 단어만 남을 수 있도록 구성하였다. 이후 정제된 리뷰 집합을 대상으로 키워드 추출 과정이 수행되며, 한나눔의 WorkflowNounExtractor, 꼬꼬마의 KeywordExtractor 모듈을 키워드 추출기로 이용하였다. 이들 모듈은 기본적으로 사전을 이용하여 형태소 분석 단계를 수행하며, 이를 통해 얻어진 단어와 각 단어의 빈도수를 결과로 제공한다. 각각의 모듈은 모두 .jar 확장명을 지닌 Java Archive 파일 형태로 Eclipse 개발 환경에 포함될 수 있으며, 실행 모듈 역시 Java로 구현되었다.

이후 추출된 키워드들의 정확도를 비교하기 위해 각각의 형태소 분석기로부터 전달된 결과를 해시맵(HashMap)에 저장하고 정렬하였다. 해시맵의 키는 추출된 단어와 연결하였으며, 해당 단어의 빈도수를 값으로 구성하였다.

2. Experimental Results

먼저 키워드 추출 실행속도와 관련하여, 세 방법을 비교한 결과를 <표 4>에서 제시하였다. 아래 표의 숫자는 밀리초(Millisecond) 단위로 표시되었으며, 10개 리뷰 집합에 대한 실행 속도의 평균값에 해당한다. 실험 결과, 실행 속도 차이가 너무 커 그래프로 표현하기 어려웠다. 예를 들어 리뷰수가 500개일 경우, 꼬꼬마는 1,200초, 한나눔은 10초, 제안 방법은 0.0039초가 소요되었다. 특히 형태소 분석 과정에서 동적 프로그래밍 기법을 쓰는 것으로 알려진 꼬꼬마의 경우 리뷰 수가 700개 이상 주어졌을 때, 메모리 에러가 발생하여 실행 속도 측정 자체가 불가능하였다.

각각의 방법에 대해 전체 실험 케이스에 대한 평균값을 비교할 경우, 꼬꼬마에 비해 한나눔이 75배 이상, 그리고 한나눔에 비해 제안 방법이 3,000배 이상 정도 빠른 것으로 조사되었다. 특히 제안 방법은 0.008초 내에 1000개의 리뷰에 대한 키워드 추출을 완료하였으며, 이는 실시간 처리가 가능한 수준임을 보여준다.

Table 4. Comparison of execution time of keyword extraction (in milliseconds): Kokoma, Hannanum, and Proposed method

# of reviews	Kokoma	Hannanum	Proposed method
100	135,537	3,871	0.786
300	587,761	7,326	2.457
500	1,241,678	10,875	3.912
700	Memory error	14,269	5.532
1,000	Memory error	20,375	7.857

다음으로 키워드 추출의 정확도와 관련한 실험을 수행하였다. 위 실험과 동일한 10개의 리뷰 집합을 대상으로 분석기별로 추출된 키워드 수 대비 오류 수에 대한 비율을 계산하였으며, 10개 집합에 대한 꼬꼬마의 평균 오류율은 0.247, 한나눔은 0.213, 제안방법은 0.155로 나타났다. 이로부터 제안 방법의 키워드 추출 정확도가 꼬꼬마와 한나눔에 비해 각각 59%, 37% 정도 더욱 우수한 것으로 조사되었다.

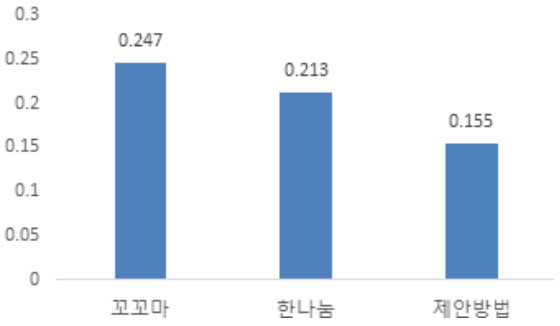

Fig. 5. Comparison of error ratio in keyword extraction: Kokoma, Hannanum, and Proposed method

아래는 각 방법으로부터 "그라파피자리아" 상호명으로 수집된 리뷰 집합으로부터 추출된 상위 10개의 키워드를 나열하였다. 제안하는 방법과 한나눔의 경우 추출된 키워드가 유사한 패턴을 보여주는 반면, 꼬꼬마의 경우 명사를 너무 세분화하여 오류가 증가하였음을 볼 수 있다.

Table 5. Comparison of top-10 keywords extracted from reviews of "Grappa Pizzaria": Kokoma, Hannanum, and Proposed method

Rank	Kokoma	Hannanum	Proposed method
1	그	그라파	그라파
2	집	피자리아	피자리아
3	맛	맛집	맛집
4	피	그라파피자리아	그라파피자리아
5	피자	꽃바위	피자
6	자리	피	화덕피자
7	피자리	일요일	남진길
8	화덕	휴무	꽃바위
9	맛집	위치한	휴무
10	꽃	남진길	피자맛집

위 실험을 통해 기존 형태소 분석기의 경우 고유명사 및 신조어 처리에 있어 명사를 너무 세분화하는 문제점이 파악되었으며, 꼬꼬마의 경우 그 문제점이 더욱 심각하게 나타났다. 이에 반해 제안하는 방법의 경우 고유명사나 신조어 처리에는 강점을 지니나, 형용사 및 동사의 활용처리 기능이 없어 같은 의미를 지닌 형용사가 다른 형태로 여러 번 제시되거나, 명사에 조사가 추가된 형태로 추천되는 문제점을 발견할 수 있었다.

제안 방법의 문제점은 단어별로 빈도수가 낮아지게 될 경우 더욱 심화될 수 있다. 그 이유는 응집도 점수 산정식은 통계적 기법으로, 산정에 필요한 모수가 적어질수록 결과의 정확도가 낮아진다는 점에 기인한다. 예를 들어, <표 3>에서 "그라파"로 시작하는 단어의 수가 많을수록 응집도 점수를 통한 명사와 조사 경계가 명확히 분리되며, 그렇지 않을 경우 조사를 포함한 부정확한 단어가 반환될 수 있다.

따라서 제안하는 알고리즘에서 일정 수준의 정확도를 담보하기 위해서는 키워드 추천에 필요한 단어별 최소 빈도수에 대한 기준을 설정해야 한다. 이와 관련하여, 아래 그래프는 제안 방법에서 단어별 최소 빈도수 기준 조정에 따른 오류율의 추이를 보여준다.

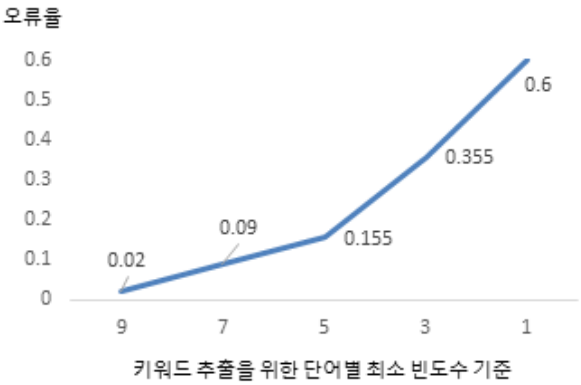

Fig. 6. Error ratio vs. word frequency criteria for keyword extraction in the proposed method

실험 결과, 9개 이상의 빈도수를 가지는 단어만을 대상으로 키워드 추천을 수행할 경우 오류율은 2% 이하로 상당한 정확도를 제공함을 알 수 있다. 이에 반해, 빈도수 기준을 1로 낮추면, 즉 토큰으로 분리된 모든 단어를 대상으로 할 경우, 에러율은 60% 이상으로 급격하게 높아진다.

결과적으로 제안하는 방법의 키워드 추천 정확도는 단어별 최소 빈도수 기준 값에 비례하며, 빈도수 기준을 5 이상으로 설정해야 기존 분석기에 비해 높은 정확도를 얻을 수 있다. 이러한 맥락에서 [그림 5]에서 제시한 제안 방법의 실험 결과 역시 빈도수 기준을 5로 설정하고 얻은 결과값을 소개하였다. 이와는 반대로 빈도수 기준을 4이하로 설정할 경우, 추천된 키워드가 대중들의 의견을 적절히 반영한 것인지에 대한 고민이 필요하다. 그럼에도 불구하고, 리뷰 수가 극단적으로 적어 대다수의 키워드가 4이하의 빈도수를 지니는 경우에도 여전히 키워드 추출에 대한 요구가 있을 수 있다. 이 경우, 제안 방법을 통해 얻어진 키워드의 정확도를 높이기 위한 후처리 과정의 도입이 필수적이다. 이와 관련한 내용은 향후 연구에서 다룰 예정이다.

V. Conclusion and Future Work

본 논문에서는 소셜 리뷰로부터 키워드 추출을 효과적으로 수행하기 위한 알고리즘을 소개하였다. 제안하는 방법은 단어의 빈도수를 기반으로 산정되는 응집도 점수를 이

용한 통계적 방법에 해당하며, 별도의 사전이나 형태소 분석 과정 없이도 고유명사 및 신조어 등을 효과적으로 처리할 수 있다. 또한 실제 데이터 적용 시 발생할 수 있는 예외사항들을 반영하여 알고리즘의 정확도를 높였다. 제안한 알고리즘의 검증을 위해 온라인에서 수집된 실제 소셜 리뷰들을 대상으로 실험을 수행하였으며, 그 결과 현재 대중적으로 이용되고 있는 한나눔 형태소 분석기에 비해 키워드 추출 속도는 평균 3,000배, 정확도는 평균 37% 정도 우수한 것으로 조사되었다. 특히 제안 방법은 1000개의 소셜 리뷰로부터 키워드 추출하는데 소요되는 시간이 0.008초 미만으로, 실시간 처리가 가능한 수준으로 파악되었다.

현재 연구의 한계점으로는 빈도수가 낮은 단어에 대한 키워드 추출 정확도가 떨어진다는 점이다. 이는 제안 방법이 단어별 빈도수에 기반한 통계적 기법이라는 점에 기인한다. 따라서 빈도수가 낮은 단어에 대한 정확도를 높이기 위한 연구가 필요할 것으로 예측되며, 이는 본 연구에서 제안한 응집도 점수 기반의 추출 결과에 일정 부분 형태소 분석을 후처리의 형태로 적용하는 방안이 예상된다.

REFERENCES

[1] W. L. Kang, H. G. Kim, and Y, J. Lee, "Reducing IO Cost in OLAP Query Processing with MapReduce," IEICE Trans. Inf. & Syst, Vol. E98-D, No. 2, pp. 444-447, Feb. 2015.

[2] K. H. Lee et al., "Parallel Data Processing with MapReduce: a Survey," ACM SIGMOD Record, Vol. 40, No. 4, pp. 11-20, 2012.

[3] IDC Korea, https://www.idc.com/getdoc.jsp?containerId=prAP 459 38720

[4] Naver Open API, https://developers.naver.com/docs/common/ open apiguide/

[5] Google Developer API, https://developers.google.com/

[6] Hannanum, http://semanticweb.kaist.ac.kr/hannanum/index.html

[7] Kokoma, http://kkma.snu.ac.kr/documents/index.jsp

[8] H. G. Kim, "Developing a Big Data Analysis Platform for Small and Medium-Sized Enterprises," Journal of the Korea Society of Computer and Information, Vol. 25, No. 8, Aug. 2020.

[9] H. J. Kim and S. J. Cho, "Cleansing Noisy Text Using Corpus Extraction and String Match," MS. Thesis, Seoul National University, 2013.

[10] Cohesion Score, https://lovit.github.io/nlp/2018/04/09/cohesion _l tokenizer/

[11] H. G. Seo and H. W. Park, "Design and Implementation of Potential Advertisement Keyword Extraction System Using SNS," Journal of the Korea Convergence Society, Vol. 9, No. 7, pp. 14-24, 2018.

[12] O. J. Lee, S. B. Park, D. Chung, and E. S. You, "Movie Box-Office Analysis Using Social Big Data," Journal of the Korea Contents Society, Vol. 14, No. 10, pp. 527-538, 2014.

[13] C. Lee, D. Choi, S. Kim, and J. Kang, "Classification and Analysis of Emotion in Korean Microblog Texts," Journal of KIISE, Vol. 40, No. 3, pp. 159-167, Jun. 2013.

[14] J. Y. Chang, "A Sentiment Analysis Algorithm for Automatic Product Reviews Classification in Online Shop ping Mall," Vol. 14, No. 4, pp. 19-32, 2009.

[15] H. Lim, B. Yoon, and H. Lim, "An Efficient Korean Morphological Analyzer using Exclusive Information," Journal of KIISE, Vol. 22, No. 6, pp. 957-964, 1995.

[16] Y. Kim, M. Park, J. Choi, and H. Kwon, "Improvement of Analysis Speed in Korean Morphologlcal Analyzer Using Ameliorated Dictionary," Proc. of the 11th Hangul and Korean Information Processing, pp. 479-483, 1999.

[17] S. H. Yang and Y. S. Kim, "A High-Speed Korean Morphological Analysis Method based on Pre-Analyzed Partial Words," Journal of KIISE, Vol. 27, No. 3, pp. 290-301, 2000.

[18] Z. Jin and K Tanaka-Ishii, "Unsupervised Segmentatino of Chinese Text by Use of Branching Entropy," The Journal of Korea Navigation Institute, pp. 428-435, Jul. 2006.

[19] Soynlp, https://github.com/lovit/soynlp

[20] E. Kim, "The Unsupervised Learning-based Language Modeling of Word Comprehension in Korean," Journal of the Korea Society of Computer and Information, Vol. 24, No. 11, pp. 41-49, Nov. 2019.

Authors

Hyeon Gyu Kim received the B.S. and M.S. degrees in Computer Science from University of Ulsan, and Ph.D. degree in Computer Science from Korea Advanced Institute of Science and Technology, Korea, in 1997, 2000 and 2010, respectively. Dr. Kim joined the faculty of the Division of Computer Science and Engineering at Sahmyook University, Seoul, Korea, in 2012. He is currently an Associate Professor in the Division of Computer Science and Engineering, Sahmyook University. He is interested in big data processing, data stream processing, and mobile computing.