JKSCI

한국컴퓨터정보학회논문지
Journal of The Korea Society of Computer and Information
Vol. 26 No. 8, pp. 55-63, August 2021
https://doi.org/10.9708/jksci.2021.26.08.055

A Method for Compound Noun Extraction to Improve Accuracy of Keyword Analysis of Social Big Data

Hyeon Gyu Kim*

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

[Abstract]

Since social big data often includes new words or proper nouns, statistical morphological analysis methods have been widely used to process them properly which are based on the frequency of occurrence of each word. However, these methods do not properly recognize compound nouns, and thus have a problem in that the accuracy of keyword extraction is lowered. This paper presents a method to extract compound nouns in keyword analysis of social big data. The proposed method creates a candidate group of compound nouns by combining the words obtained through the morphological analysis step, and extracts compound nouns by examining their frequency of appearance in a given review. Two algorithms have been proposed according to the method of constructing the candidate group, and the performance of each algorithm is expressed and compared with formulas. The comparison result is verified through experiments on real data collected online, where the results also show that the proposed method is suitable for real-time processing.

▸ **Key words**: Big data analysis, Social reviews, Keyword extraction, Morphological analysis, Compound noun extraction

[요 약]

소셜 빅데이터는 신조어나 고유명사를 포함하는 경우가 많으며, 이들을 처리하기 위해 단어별 출현 빈도수를 기반으로 한 통계적인 형태소 분석 방법이 많이 활용되고 있다. 그러나 이들 방법에서는 복합 명사를 제대로 인지하지 못해, 키워드 추출의 정확도가 떨어지는 문제점이 지적되고 있다. 본 논문에서는 소셜 빅데이터의 키워드 분석에 있어 복합 명사를 추출하기 위한 방법을 제안한다. 제안 방법은 형태소 분석 단계를 통해 얻어진 단어를 조합하여 복합 명사 후보군을 만들고, 주어진 리뷰에서 이들의 출현 빈도를 조사하여 얻어진 빈도수를 기반으로 복합 명사를 추출한다. 복합 명사 후보군을 구성하는 방법에 따라 두 가지 알고리즘을 제안하였으며, 각 알고리즘의 성능을 수식으로 표현하고 비교한다. 그리고 온라인에서 수집된 실제 데이터를 대상으로 실험을 통해 비교 결과를 검증하는 동시에, 제안 방법이 실시간 처리에도 적합함을 보여준다.

▸ **주제어**: 빅데이터 분석, 소셜 리뷰, 키워드 추출, 형태소 분석, 복합 명사 추출

• 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: 2021. 06. 30, Revised: 2021. 08. 11, Accepted: 2021. 08. 11.

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 신용카드, 휴대폰 이용 내역 등 수치로 이루어진 다른 형태의 빅데이터에 비해 고객들의 의견이나 불만 사항을 텍스트 형식으로 바로 파악할 수 있다는 점에서 선호되고 있다.

소셜 빅데이터 분석은 텍스트를 다뤄야 한다는 점에서 다른 형태의 데이터에 대해 더욱 세밀한 전처리 과정을 필요로 하게 된다. 이는 수집된 리뷰에 대해 형태소 분석을 통해 표준화하거나, 주어진 검색어와 연관성이 떨어지는 리뷰들을 걸러내는 작업 등을 포함한다[4]. 특히, 소셜 리뷰의 경우 신조어나 고유 명사를 포함하는 경우가 많으므로, 꼬꼬마[5]나 한나눔[6] 등 사전에 기반한 기존의 형태소 분석기를 이용하기 어려운 문제점이 있다. 이는 신조어나 고유 명사가 사전에 포함되지 않음에 기인한다.

위 문제의 해결을 위해, 단어별 출현 빈도수를 기반으로 한 통계적인 형태소 분석 방법들이 제안되었다. 이는 브랜칭 엔트로피[7]나 응집도 점수[8, 9] 등을 계산하여 분석에 이용하며, [10, 11]에서는 해당 방법이 신조어나 고유명사의 처리에 있어 사전에 기반한 기존 방법에 비해 높은 정확도를 나타내는 동시에, 실시간 처리 역시 가능함을 보여주었다.

이에 반해, 해당 방법에서는 복합 명사를 제대로 인지하지 못하는 문제점이 발견되었다. 이는 해당 방법이 형태소 분석을 위한 계산에 있어서 단어 간의 전후 관계를 계산에 반영하지 않기 때문이다. 예를 들어 리뷰에서 “붓가케 우동”의 출현 빈도가 10회, “사누키 우동”의 출현 빈도가 5회라고 가정해 보자. 단어들의 출현 빈도만을 고려하여 키워드를 추출할 경우, “우동” 15회, “붓가케” 10회, “사누키” 5회 등으로 빈도가 집계된다. 따라서 형태소 분석을 통한 키워드 추출 결과는 {“우동”, “붓가케”, “사누키”}가 되며, 해당 결과에는 복합 명사가 가졌던 원래 의미들이 모두 사라져 부정확한 결과로 이어지게 된다.

위 문제는 가격이나 인지도 등 키워드와 관련한 추가적인 정보를 찾고자 할 때 더욱 큰 문제로 발전할 수 있다. 리뷰에서 가격 정보를 추출하고자 할 경우, 일반적으로 추출된 키워드 다음이나 일정 거리 내에 가격을 나타내는 패턴(xx,xxx원)이 존재하는지 확인한다. 해당 패턴이 존재할 경우 이를 가격 정보로 추가하여 사용자에게 제시한다. 예를 들어, 리뷰에 “붓가케 우동 12,000원”이라는 표현이 포함되어 있다고 가정해 보자. 이 경우 키워드 분석을 통해 얻어진 키워드가 “붓가케 우동”이 아닌 “우동”이라면, 우동 가격으로 12,000원이 설정되며, 이는 왜곡된 분석 결과로 이어진다.

위 문제를 해결하기 위해, 본 논문에서는 소셜 빅데이터의 키워드 분석에 있어 복합 명사를 추출하기 위한 방법을 제안한다. 제안 방법은 이전 연구[10, 11]에서 제안한 형태소 분석 방법에 대한 후처리 과정으로 이루어진다. 예를 들어 형태소 분석을 통해 키워드가 얻어질 경우, 이들을 조합하여 복합 명사 후보군을 만든다. 그리고 후보군에 포함된 각각의 표현에 대해 소셜 리뷰에서 얼마나 자주 출현되는지 빈도를 조사한다. 출현 빈도가 미리 설정된 임계값 이상일 경우 복합 명사로 추출하여 다음 형태소 분석에 반영함으로써 분석 정확도를 높일 수 있다.

아래에서는 복합 명사 추출을 위한 알고리즘과 시스템 구성에 대해 소개한다. 특히 복합 명사 후보군의 구성 방법에 따라 알고리즘의 성능이 크게 달라질 수 있다. 이와 관련하여 아래에서는 후보군 구성을 위한 두 가지 알고리즘을 소개하고, 각각의 성능을 수식으로 표현하고 비교한다. 그리고 온라인에서 수집된 실제 데이터를 대상으로 실험을 통해 비교 결과를 검증하고, 제안 알고리즘이 실시간 처리에도 적합함을 보여준다.

II. Related Work

소셜 빅데이터는 고객들의 의견이나 불만 사항을 텍스트 형식으로 바로 파악할 수 있다는 장점으로부터 다양한 응용 및 연구 주제에서 활용되고 있다. 예를 들어, 소셜 리뷰로부터 잠재적인 광고 키워드를 추출하거나[12], 블로그 리뷰로부터 영화 흥행 요인 분석[13, 14], 온라인 쇼핑몰의 상품평 분류를 통한 감성 분석 [15] 등의 주제를 위해 소셜 빅데이터가 분석/이용되고 있다. 특히 최근에는 기계학습 모델을 이용한 감성 분석 연구가 활발하게 진행되고 있으며, [16-18]에서는 순환 신경망(Recurrent Neural Network)이나 LSTM(Long Short-Term Memory) 알고리즘을 활용하여 영화 리뷰를 대상으로 감성 분석을 수행한 바 있다.

소셜 빅데이터 분석 기술과 관련해서도 다양한 주제가 논의되었다. [4]는 소상공인들의 마케팅을 돕기 위한 서비스 지표들을 추출하고 이를 구현한 시스템을 소개하였다. 그리고 소셜 빅데이터 분석에 있어 형태소 분석 및 노이즈 리뷰 필터링의 중요성을 언급하였다. 형태소 분석과 관련

하여 [10, 11]은 소셜 빅데이터 분석의 특성상 신조어나 고유 명사가 많아, 사전에 기반한 기존의 형태소 분석기법 등을 활용할 경우 정확도가 저하될 수 있음을 지적하였다. 그리고 문제점 해결을 위해 브랜칭 엔트로피(Branching Entropy)[7]나 응집도 점수 개념[8, 9] 등이 효과적으로 이용될 수 있음을 보였다. 이들 방법은 공통적으로 단어 내부에서는 불확실성이 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. 이외에 [19] 역시 같은 맥락에서 비지도 기계학습법을 활용한 형태소 분석 방법이 효과적으로 이용될 수 있음을 제안하였다.

노이즈 리뷰 필터링과 관련하여, [4]에서는 패턴 매칭을 이용하여 주어진 키워드와 관계없는 노이즈 리뷰를 추출하고 제거하는 기법을 소개하였다. 이에 반해, [20]에서는 여행지 평점 예측을 위해 합성곱 신경망(Convolutional Neural Network) 기반의 필터링 방법을 제안하였으며, [21]에서는 협업 필터링(Collaborative Filtering)을 기반으로 리뷰의 연관도를 수치화하고 필터링하는 방법을 논의하였다. 이외에도 노이즈 리뷰 필터링을 위해 [22]에서 소개한 기계학습 기반의 다양한 알고리즘들이 적용될 수 있다.

Table 1. Classification of References by social big data analysis steps

Analysis Step	Techniques	Reference #
Review Collection	Open search API, Web crawling	[4, 10]
Morphological Analysis	Statistical approach	[7-11]
	Dictionary-based approach	[5, 6]
Noise Review Filtering	Pattern matching	[4]
	Machine learning	[18-20]
Review Analysis and Applications	Sentiment analysis	[15-17]
	Keyword analysis	[10-12]
	Miscellaneous	[13, 14]

III. Proposed Method

제안 방법은 이전 연구[10, 11]에서 소개한 응집도 점수 기반의 형태소 분석 알고리즘을 이용한다. 이 장에서는 먼저 이전 연구에 대해 간략히 소개하고, 이를 기반으로 복합 명사를 추출하기 위한 처리 구조와 알고리즘들을 설명하고 비교한다.

1. Previous Work

응집도 점수 기반의 형태소 분석 알고리즘은 주어진 텍스트로부터 단어를 구성하는 글자들의 정보만을 이용하여 단어의 경계를 판단하는 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 L 파트를 분리해내는 것이며, 응집도 점수는 이를 위한 산정식을 제공한다.

예를 들어, 입력으로 주어진 소셜 리뷰 집합에서 네 개의 어절과 각 어절에 대한 빈도가 아래와 같이 주어졌다고 가정해 보자.

자 (192)
자장 (102)
자장면 (94)
자장면을 (8)

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

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

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

P(자장|자) = 102/196 ≒ 0.52
P(자장면|자장) = 94/102 ≒ 0.92
P(자장면을|자장면) = 8/94 ≒ 0.08

위 확률로부터 식 (1)을 이용하여 "자장면"에 대한 응집도 점수를 다음과 같이 계산할 수 있다. 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(자장면) = (P(자장|자) × P(자장면|자장))$^{0.5}$
= $(0.52 \times 0.92)^{0.5}$ ≒ 0.69

식 (1)을 통해 역시 f (자장)과 f (자장면을) 값을 계산할 수 있으며, 계산을 통해 각각 0.52와 0.34를 f 값으로 얻을 수 있다. 결과적으로 "자장면"의 f 점수가 가장 높게 나타나며, 이는 어절이 "자장면" + "을"의 형태로 구성되었음을 의미한다. 해당 결과로부터 L 값으로 "자장면"을 반환한다.

위 방법은 주어진 텍스트만을 이용하여 단어의 L값을 추출할 수 있다는 특징이 있으며, 1장에서 언급한 바와 같이, 신조어나 고유 명사를 포함한 소셜 리뷰를 처리할 경우 사전에 기반한 기존 형태소 분석 알고리즘에 비해 효과적으로 활용될 수 있다.

이에 반해, 해당 방법은 단어의 전후 관계를 고려하지 않아, 추출된 L 파트들 간의 관계를 유추하기가 어려운 문제점이 있다. 특히 복합 명사의 경우, 해당 명사에 포함된 L 파트들이 서로 분리된 형태로 추출되어 원래의 의미를 알기 어려워질 수 있다. 또한 1장에서 언급한 바와 같이, 가격이나 인지도 등 키워드와 관련한 추가적인 정보를 찾고자 할 때, 복합 명사 추출의 오류로 인해 분석 결과가 왜곡되는 문제점도 발생할 수 있다. 따라서 소셜 빅데이터 분석의 정확도를 높이기 위해서는 형태소 분석 결과로부터 복합 명사를 추출하고 처리하기 위한 별도의 메카니즘이 함께 필요함을 알 수 있다.

2. Compound Noun Extraction

응집도 점수 기반의 형태소 분석 알고리즘을 이용할 경우, 복합 명사 추출 과정은 형태소 분석 단계가 종료된 후 후처리 과정으로 이루어질 수 있다. 이는 후보 단어 구성, 후보 단어별 출현 빈도수 계산, 빈도수에 기반한 복합 명사 추출 등의 과정을 포함한다.

먼저 후보 단어 구성 과정은 아래와 같이 기술될 수 있다. 리뷰 집합 R로부터 형태소 분석을 통해 얻은 단어 집합을 $S = \{ w_i \}\ (1 \le i \le N)$라고 하자. 이 경우 후보 단어 집합 T에는 S에서 임의로 선택한 두 개 이상의 w_i와 w_j $(1 \le i, j \le N, i \ne j)$의 조합을 통해 얻을 수 있는 모든 단어가 포함되며, 그 수는 아래 식과 같이 표현될 수 있다.

$$|T| = \sum_{i=2}^{N} \left\{ \prod_{j=N-i+1}^{N} j \right\} \quad (2)$$

예를 들어, S = {"붓가케", "맛집", "우동"}일 경우, T = {"붓가케 맛집", "붓가케 우동", "맛집 붓가케", "맛집 우동", "우동 붓가케", "우동 맛집", "붓가케 맛집 우동", "붓가케 우동 맛집", "맛집 붓가케 우동", "맛집 우동 붓가케", "우동 붓가케 맛집", "우동 맛집 붓가케"}가 된다. 그리고 T의 크기 |T|는 (3 × 2) + (3 × 2 × 1) = 12가 된다.

T가 얻어지면, T에 속한 각 복합 명사 별로 출현 빈도수를 계산한다. 이를 위해, 리뷰 집합 R에 속한 각각의 리뷰에 대해 특수 문자 및 공백 등을 없애는 전처리 과정을 수행하여, 리뷰에 글자만 포함되도록 구성한다. 복합 명사에서도 공백을 제거한다. 이후 T에 속한 각각의 명사에 대해 R에서 몇 번 나타나는지 빈도수를 계산한다.

마지막으로 후보군의 출현 빈도수를 기반으로 복합 명사를 추출한다. 이전 연구[10, 11]에서 제안한 형태소 분석 방법은 단어의 빈도수를 기반으로 하며, 만족할 만한 분석 정확도를 얻기 위해 단어별로 필요한 최소 빈도수는 5로 설정되었다. (5회 이하일 경우, 이전 절에서 설명한 f 값을 신뢰하기 어렵다.) 이는 S에 포함된 모든 단어가 R에서 최소 5회 이상 나타난다는 것을 의미한다. 따라서 S의 단어로 구성된 복합 명사에 대해서도, 해당 명사가 적법하다면 R에서 최소 5회 이상 나타나야 함을 알 수 있다. 위 내용으로부터, T의 단어 중 R에서 출현 빈도수가 5 이상이면 해당 단어를 복합 명사로 간주하고 결과로 출력한다.

```
function getCompoundNouns(keys, revs) {
    // [STEP.1] build a candidate list
    var cands = [];
    for (var i=0; i<keys.length; i++) {
        for (var j=0; j<keys.length; j++) {
            if (i!=j) cands.push(keys[i] + " " + keys[j]);
        }
    }

    // [STEP.2] calculate frequency of each word in
       candidate list
    var res = [];
    for (var i=0; i<cands.length; i++) {
        var cand = stripSpecialChars(cands[i]);
        var freq = 0;
        for (var j=0; j<revs.length; j++) {
            var rev = stripSpecialChars(revs[j]);
            freq += getFreqs(rev, cand);
        }
        // [STEP.3] add to a result if freq >= 5
        if (freq >= 5) res.push(cands[j]);
    }

    // [STEP.3] return the extracted compound nouns
    return res;
}
```

Fig. 1. Algorithm to get compound nouns from the keywords obtained by morphological analysis based on cohesion scoring

Fig. 1은 위 내용을 구현한 알고리즘이다. 해당 알고리즘은 JavaScript 함수 형태로 구현되었으며, 입력 파라미터로 형태소 분석 알고리즘을 통해 얻은 단어 리스트, 즉 S 집합에 해당하는 keys와, 리뷰 집합인 R에 해당하는 revs를 전달받아, 복합 명사 집합을 추출하여 반환한다. 함수는 앞서 설명한 바와 같이 크게 3부분으로 구성되어 있으며, 아래에서는 [Step.1] ~ [Step.3]로 표현하였다.

[Step.1]은 후보 단어를 구성하는 부분으로, cands 변수는 해당 단어들을 포함한 배열을 나타내며, keys에 포함된 단어의 조합으로 구성된다. 위 구현에서는 논의를 단순화하기 위해 2개의 단어로 구성되는 복합 명사만을 고려하였으며, 3개 이상의 단어로 구성되는 복합 명사를 처리하고자 할 경우, 내부에 for문을 추가하여 구현될 수 있다.

[Step.2]에서는 cands에 포함된 각각의 후보에 대해 revs에서 나타나는 빈도수를 계산한다. stripSpecialChars() 함수는 입력된 스트링으로부터 특수 문자 및 공백을 제거한다. getFreqs()는 주어진 rev에 cand 패턴이 나타나는 빈도수를 계산하기 위한 함수로, 아래와 같이 구현될 수 있다. 아래는 2개의 단어로 구성되는 경우만을 고려하였으며, 3개 이상의 단어로 구성되는 명사일 경우 if문 조건을 추가하여 처리할 수 있다.

```
function getFreqs(rev, cand) {
    var toks = rev.split(" ");
    var nouns = cand.split(" ");
    var freq = 0;
    for (var i=0; i<toks.length-1; i++) {
        if (toks[i].endsWith(nouns[0] &&
            toks[i+1].startsWith(nouns[1]) freq++;
    }
    return freq;
}
```

Fig. 2. Algorithm to get frequency of a compound noun in the given review text

[Step.3]에서는 주어진 복합 명사 cand에 대해 R에 포함된 모든 리뷰에서 출현되는 빈도수를 구한 후, 해당 빈도수가 5 이상이면 결과에 포함하여 반환한다.

3. Gradual Algorithm

위 알고리즘은 직관적인 반면, [Step.1]에서 모든 가능한 복합 명사를 미리 구성해 두고 각각에 대한 출현 빈도수를 계산한다는 측면에서 다소 비효율적일 수 있다. 예를 들어 S = {"붓가케", "맛집", "우동"}에서 조합된 "맛집 붓가케"의 경우, 실제 복합 명사가 아니므로 출현 빈도가 매우 낮을 것으로 예상되는데 반해, 위 알고리즘에서는 빈도수 계산을 위해 해당 단어에 대해서도 모든 리뷰를 검색하므로 비효율적일 수 있다.

따라서 알고리즘의 효율을 높이기 위해, [Step.1]에서 T를 미리 구성하지 않고 [Step.2]의 출현 빈도수를 계산하는 과정에서 T를 점진적인 방법으로 구성할 수 있다. 예를 들어 리뷰에서 S에 포함된 "붓가케"가 나타난다면, 리뷰의 그 다음 단어가 S에 포함된 다른 단어, 즉 "맛집"이나 "우동"이 아닌지 순차적으로 확인하는 방법이다.

Fig. 3은 점진적인 방식으로 복합 명사를 구성하기 위한 알고리즘이다. 해당 알고리즘 역시 크게 3부분으로 구성되어 있으며, 아래에서 [Step.1] ~ [Step.3]로 표현하였다. [Step.1]은 입력 파라미터로 주어진 리뷰 집합으로부터 토큰을 분리해내는 과정이며, 분리된 토큰은 toks 배열에 저장된다. [Step.2]는 복합 명사 후보에 대한 출현 빈도수를 계산하기 위한 과정으로, keys에 주어진 각 단어에 대해 toks 배열에 나타나는지 확인한다. toks 배열에 나타날 경우, checkCand() 함수를 이용해 복합 명사 후보 여부를 체크한다. 만약 복합 명사 후보로 판명된다면, addFreq() 함수를 이용하여 해당 단어에 대한 빈도수를 추가한다.

```
function getCompoundNouns2(keys, revs) {
    // [STEP.1] build a token list from revs
    var toks = [];
    for (var i=0; i<revs.length; i++) {
        var rev = stripSpecialChars(revs[i]);
        toks = toks.concat(rev.split(" "));
    }

    // [STEP.2] calculate frequency of each candidate
    var res = [];
    for (var i=0; i<keys.length; i++) {
        for (var j=0; j<toks.length-1; j++) {
            if (toks[j].endsWith(keys[i])) {
                var cand = checkCand(keys, i, toks, j);
                if (cand != null) res = addFreq(res, cand);
            }
        }
    }

    // [STEP.3] sort a result list by frequency and
    //  eliminate an element if its frequency < 5
    res.sort(function(a, b) {
        return a.freq > b.freq ? -1 : 1;
    });
    for (var i=res.length-1; i>=0; i--) {
        if (res.freq < 5) res.splice(i, 1);
    }
    return res;
}
```

Fig. 3. Algorithm to get compound nouns in a gradual fashion

[Step.3]는 복합 명사 후보로 추출된 단어들에 대해 빈도수를 기반으로 내림차순 정렬을 수행하며, 빈도수가 5이하일 경우 후보로부터 해당 단어를 삭제한 후 결과로 반환한다.

checkKey() 함수는 복합 명사 후보 여부를 체크하기 위해 이용되며, keys에 포함된 특정 단어가 toks에 포함된 경우에 한해 호출된다. 따라서 해당 함수는 toks 배열의 다음 요소가 keys에 포함된 다른 단어와 일치하는지 확인한다. 예를 들어 keys에 포함된 "붓가케"가 리뷰를 토큰화한 toks 배열에 나타났다면, toks 배열의 그 다음 단어가 keys에 포함된 다른 단어, 즉 "맛집"이나 "우동"이 아닌지 확인한다. 만약 toks의 다음 단어가 "우동"을 포함한다면, 후보 단어로 "붓가케 우동"을 반환하는 원리이다.

```
function checkKeys(keys, kidx, toks, tidx) {
    for (var i=0; i<keys.length; i++) {
        if (i == kidx) continue;
        if (toks[tidx+1].startsWith(keys[i])) {
            return keys[kidx] + " " + keys[i];
        }
    }
    return null;
}

function addFreq(res, cand) {
    for (var i=0; i<res.length; i++) {
        if (res[i].key == cand) {
            res[i].freq++;
            return res;
        }
    }
    res.push({key: cand, freq: 1});
    return res;
}
```

Fig. 4. Implementation of function checkKey() and addFreq()

addFreq() 함수는 파라미터로 주어진 res 리스트에서 cand에 해당하는 단어를 찾아 빈도(freq)를 1씩 증가시킨다. 만약 res에 cand에 해당하는 단어가 없으면 해당 단어에 대한 객체를 새로이 생성한 후 res에 추가한다. 생성 시 key와 freq 속성 값을 각각 cand와 1로 설정한다.

4. Comparison of Algorithm Performance

3.2절과 3.3절에서 소개한 알고리즘의 성능은 Fig.1과 Fig.3의 [Step.2]로 언급된 복합 명사 후보에 대한 출현 빈도수 계산 부분에 의해 결정된다. 리뷰 집합 R로부터 형태소 분석을 통해 얻은 단어 집합 S의 크기를 N, 리뷰 집합 R의 리뷰를 토큰화하여 얻어진 토큰의 개수를 M이라고 하자. 이 경우, 3.2절에서 설명한 알고리즘의 성능은 아래와 같이 표현될 수 있다.

$$C_1 = N^2 M(\Pr\{h(k_0)\} + \Pr\{h(k_1|k_0)\}) \quad (3)$$

위 식에서 N^2은 복합 명사 후보의 수를 의미한다. Fig.1의 알고리즘에서는 cand 변수의 크기에 해당한다. 그리고 cand의 각 요소에 대해 getFreq() 함수를 통해 주어진 리뷰의 토큰들을 검사하므로, 요소별로 M번의 검사가 수행됨을 알 수 있다. 위 식에서 k_0와 k_1은 cand의 요소, 즉 복합 명사를 구성하는 단어를 의미하며, $h(k_0)$는 k_0가 주어진 토큰과 일치하는지 확인하는 함수를 나타낸다. $Pr\{h()\}$는 h() 함수가 수행될 확률을 나타내며, Fig.1에서는 k_0와 k_1에 대한 검사가 무조건 수행되므로 $Pr\{h(k_0)\}$와 $Pr\{h(k_1|k_0)\}$은 모두 1이 된다. 따라서 3.2절의 알고리즘 성능, 즉 C_1은 $2N^2M$으로 표현될 수 있다.

3.3절의 점진적인 알고리즘 성능은 (3)과 약간 다른 방법으로 계산된다. Fig.3의 [Step.2]에서는 keys, 즉 단어 집합 S에 대한 요소를 대상으로 리뷰의 토큰들을 검사한다. 따라서 [Step.2]에서 수행되는 검사 수는 NM이 된다. 그리고 $h(k_0)$ 부분까지 [Step.2]에서 수행된다. (이는 if(toks[j].endsWith(...)) 문에 해당한다.)

이에 반해, $h(k_1|k_0)$은 $h(k_0)$의 조건이 만족될 경우에 한해 checkCand() 함수에서 수행된다. 이러한 경우, k_1에 대한 검사는 S에 대한 모든 요소를 대상으로 이루어진다. 따라서 3.3절의 점진적인 알고리즘 성능은 아래와 같이 표현될 수 있다.

$$C_2 = NM(\Pr\{h(k_0)\} + \Pr\{h(k_1|k_0)\} \times N) \quad (4)$$

[Step.2]에서 이루어지는 NM번의 모든 검사에 대해 k_0에 대한 검사는 수행되므로, $Pr\{h(k_0)\}$는 역시 1이 된다. 이에 반해 $h(k_1|k_0)$은 조건적으로 수행되므로, 이에 대한 확률은 0에서 1사이의 값이 된다. 편의상 $Pr\{h(k_1|k_0)\}$을 σ로 표시하면 3.3절의 알고리즘 성능, 즉 C_2는 $NM(1 + \sigma N)$으로 표현될 수 있다.

위 내용으로부터 두 알고리즘의 성능을 비교하기 위해 C_1/C_2를 계산하면,

$$C_1/C_2 = 2/(1/N + \sigma) \quad (5)$$

위 식에서 σ의 경우, 확률 값이므로 최대값이 1이며, N은 항상 2 이상의 값을 가진다. N이 1인 경우 S에 단어가 하나만 존재함을 의미하며, 이 경우 복합 명사를 추출하기 위한 문제 자체가 성립하지 않는다. 따라서 1/N + σ의 최대값은 1.5임을 알 수 있다. 결과적으로 위 식은 항상 1보다 적은 값을 갖게 됨을 알 수 있으며, 이로부터 점진적인 알고리즘의 성능이 더욱 우수함을 입증할 수 있다.

IV. Experimental Results

제안 알고리즘의 성능을 검증하기 위해, 온라인에서 수집된 실제 리뷰 데이터를 이용하여 실험을 수행하였다. 실험을 위해 울산의 각 지역구별로 5개씩, 총 50개의 스토어를 선정하였으며, 해당 스토어들을 대상으로 온라인으로부터 수집된 1만여 건의 리뷰를 실험에 이용하였다.

Table 2. Summary of source data used for our experiments

	Nam gu	Dong gu	Buk gu	Ulju	Jung gu
# of reviews	2,204	1,869	1,767	2,128	1,904
# of tokens	69,320	59,847	57,604	71,525	62,968
# of keywords	439	380	361	423	425

위 표는 실험에 이용된 데이터에 대한 요약 정보를 보여준다. 50개 스토어를 대상으로 수집된 리뷰 수의 평균값은 197개, 토큰 수의 평균값은 6,425개였으며, 이들로부터 형태소 분석을 통해 얻어진 키워드 수(S의 크기)는 평균 41개로 조사되었다.

Table 3. Number of compound nouns identified from candidate expressions in our algorithm

	Nam gu	Dong gu	Buk gu	Ulju	Jung go	Mean
# of candidates	1,875	1,406	1,274	1,739	1,772	1,613
# of composite nouns	15	14	13	9	12	13
Ratio	0.81%	1.00%	0.99%	0.53%	0.69%	0.78%

먼저 제안한 알고리즘으로부터 키워드의 조합을 통해 얻어질 수 있는 복합 명사의 수와 비율을 조사하였다. 위 표는 각 지역구를 대상으로 키워드 조합을 통해 얻어진 복합 명사 후보 수와 제안 알고리즘을 통해 실제 추출된 복합 명사 수를 보여준다. 후보 수는 평균 1,613개였으며, 추출된 복합 명사 수는 13개로 전체의 0.78%에 해당하였다. 위 값은 3.2절과 3.3절에서 설명한 두 알고리즘에서 모두 동일하다.

Table 4. Number of comparisons for compound noun extraction in algorithms discussed in Section 3.2 and 3.3 (unit: 1,000)

	Nam gu	Dong gu	Buk gu	Ulju	Jung go	Mean
Algorithm 3.2	25,062	9,259	6,836	11,989	16,103	13,849
Algorithm 3.3	496	274	236	330	347	337
Ratio	1.98%	2.96%	3.46%	2.75%	2.51%	2.43%

다음으로 3.2절과 3.3절에서 제시한 알고리즘의 성능을 비교하였다. 위 표는 두 알고리즘에서 복합 명사 추출을 위해 수행된 비교 횟수를 보여준다. 3.2절의 기본 알고리즘은 평균 138만회의 비교를 수행하였으며, 3.3절의 점진적인 알고리즘은 평균 3만 3천회의 비교를 수행하였다. 이로부터 점진적인 알고리즘이 기본 알고리즘에 비해 실행 횟수 측면에서 약 46배(≒13,849/337) 정도 우수함을 확인하였다.

Table 5. Execution time of two algorithms discussed in Section 3.2 and 3.3 (unit: milliseconds)

	Nam gu	Dong gu	Buk gu	Ulju	Jung go	Mean
Algorithm 3.2	2737.3	1560.4	1141.8	1978.8	2759.4	2035.5
Algorithm 3.3	36.4	18.9	23.9	17.1	26.5	24.6
Ratio	1.33%	1.21%	2.09%	0.87%	0.96%	1.21%

위 표는 두 알고리즘의 실행 시간을 보여준다. 실행 시간은 실행 횟수에 비해 더욱 큰 차이를 보였으며, 점진적인 알고리즘이 약 83배(≒2035.5/24.6) 정도 빠른 것으로 확인되었다. 아래 그림은 위 내용을 그래프의 형태로 비교하였다. 점진적인 알고리즘의 성능이 압도적으로 우수함을 한 눈에 파악할 수 있다.

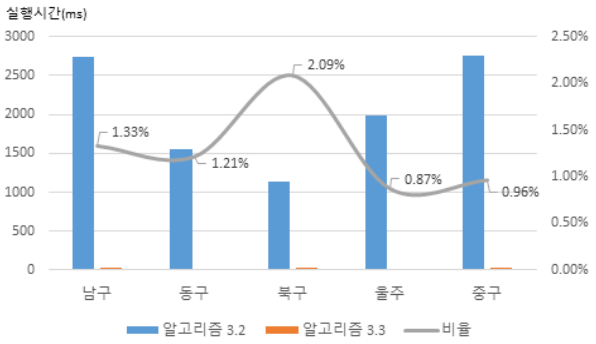

Fig. 5. Execution time (ms, milliseconds) of two algorithms discussed in Section 3.2 and 3.3

Table.4와 Table.5에서 보여준 비교 횟수와 실행 시간 간에는 강한 양의 상관관계가 있으며, 이는 아래 그래프에를 통해 확인할 수 있다. Fig.6은 기본 알고리즘, Fig.7은 점진적인 알고리즘에서의 상관관계를 보여주며, 비교 횟수와 실행 시간이 더 큰 기본 알고리즘에서 상관관계가 더욱 뚜렷이 나타남을 알 수 있다.

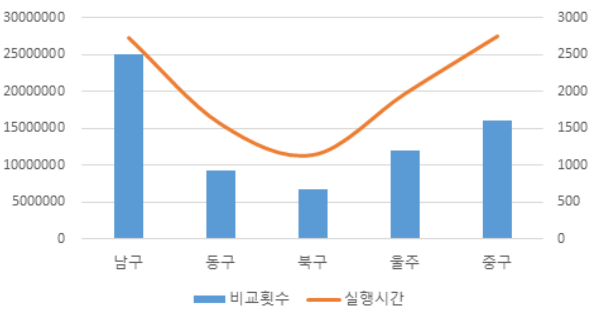

Fig. 6. Correlation between number of comparisons and execution time (ms) of algorithms discussed in Section 3.2

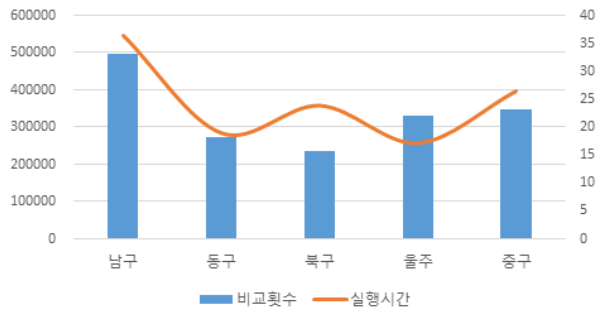

Fig. 7. Correlation between number of comparisons and execution time (ms) of algorithms discussed in Section 3.3

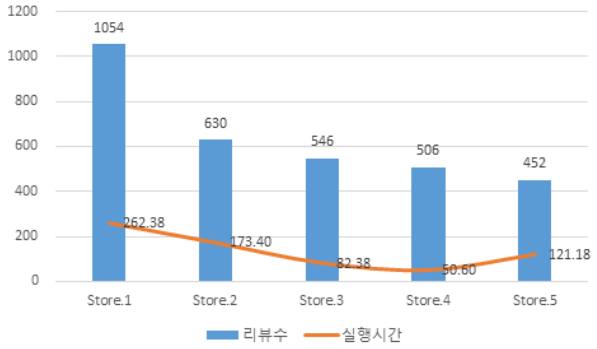

Fig. 8. Execution time (ms) of the gradual algorithm for top five stores whose numbers of reviews are larger than others in the data set of our experiments

다음으로 점진적인 알고리즘에서 checkCand() 함수가 수행되는 비율, 즉 σ값을 조사하였다. σ값은 알고리즘의 성능을 결정하는 주요 지표로 볼 수 있으며, 값이 적을수록 알고리즘의 성능이 우수해진다. 아래 표는 Fig.3에서 제시한 알고리즘의 [Step.2]에서 실험을 통해 얻어진 $h(k_0)$의 수행 횟수와 $h(k_1|k_0)$의 수행 횟수를 비교하고 있다. ($h(k_0)$는 if(toks[j].endsWith(...)) 문이며, $h(k_1|k_0)$는 check Cand() 함수에 해당한다.) 아래 표에서 수행 횟수는 평균값으로 표현되었다.

Table 6. Ratio of execution numbers of $h(k_0)$ and $h(k_1|k_0)$ in algorithm discussed in Section 3.3

	Nam gu	Dong gu	Buk gu	Ulju	Jung go	Mean
# of $h(k_0)$ executions	406,511	244,120	210,049	306,106	309,491	295,255
# of $h(k_1\|k_0)$ executions	1,693	1,111	1,083	771	1,179	1,167
Ratio	0.42%	0.46%	0.52%	0.25%	0.38%	0.4%

실험 결과, $h(k_1|k_0)$의 수행 비율, 즉 σ값은 평균 0.4% 수준으로 1%를 넘지 않았다. 이는 점진적인 알고리즘이 기본 알고리즘에 비해 실행 횟수나 실행 시간 측면에서 큰 성능 차이를 보인 이유를 뒷받침한다.

마지막으로, 리뷰의 수가 많을 경우 점진적인 알고리즘의 실행 시간을 조사하였다. 아래 그래프는 실험에 이용된 50개 스토어를 대상으로 리뷰 수 기준 상위 5개의 스토어에 대한 리뷰 수와 복합명사 추출 시간을 보여준다. 실험 결과 1,000개의 리뷰를 처리하는데 약 250ms가 소요되며, 500개의 리뷰 처리에는 평균 108ms 정도가 소요됨을 알 수 있었다. 이는 제안 알고리즘이 실시간 처리에도 적합함을 보여준다.

V. Conclusion and Future Work

본 논문에서는 소셜 빅데이터의 키워드 분석에 있어 복합 명사를 추출하기 위한 두 가지 알고리즘을 제안하였다. 첫 번째 알고리즘은 복합 명사 후보 군을 미리 구성한 후 출현 빈도를 조사하여 복합 명사를 추출하는 방법이다. 두 번째 알고리즘은 출현 빈도를 조사하는 과정에서 후보 군을 점진적으로 찾아내고 복합 명사를 추출하는 방법이다. 각 알고리즘에 대한 성능을 식으로 표현하고 계산한 결과 두 번째 알고리즘이 더 나은 성능을 보장함을 알 수 있었다. 그리고 온라인에서 수집된 실제 리뷰를 대상으로 수행된 실험 결과를 통해 해당 비교 결과가 성립함을 확인할 수 있었다. 실험 결과, 점진적인 알고리즘이 기본 알고리즘에 비해 실행 횟수 측면에서 46배, 실행 시간 측면에서 83배 정도 우수하였으며, 점진적인 알고리즘의 경우 500개의 리뷰로부터 복합 명사를 추출하는 시간이 약 108ms 정도로 조사되어 실시간 처리에도 적합한 것으로 확인되었다.

향후 연구로는 추출된 복합 명사를 기반으로 가격이나 인지도 등의 추가적인 정보를 보다 정확히 추출하는 방법에 대해 연구할 예정이다.

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 45938720

[4] 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, pp. 65-72, Aug. 2020.
[5] Kokoma, http://kkma.snu.ac.kr/documents/index.jsp
[6] Hannanum, http://semanticweb.kaist.ac.kr/hannanum/index.html
[7] Z. Jin and K Tanaka-Ishii, "Unsupervised Segmentation of Chinese Text by Use of Branching Entropy," The Journal of Korea Navigation Institute, pp. 428-435, Jul. 2006.
[8] H. J. Kim and S. J. Cho, "Cleansing Noisy Text Using Corpus Extraction and String Match," MS. Thesis, Seoul National University, 2013.
[9] Cohesion Score, https://lovit.github.io/nlp/2018/04/09/cohesion_ltokenizer/
[10] H. G. Kim, "Efficient Keyword Extraction from Social Big Data Based on Cohesion Scoring," Journal of the Korea Society of Computer and Information, Vol. 25, No. 10, pp. 87-94, Oct. 2020.
[11] Y. W. Yu and H. G. Kim, "Interactive Morphological Analysis to Improve Accuracy of Keyword Extraction Based on Cohesion Scoring," Journal of the Korea Society of Computer and Information, Vol. 25, No. 12, pp. 145-153, Dec. 2020.
[12] 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.
[13] 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.
[14] 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.
[15] 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.
[16] C. Park and C. Lee, “Korean Movie Review Sentimental Analysis using RNN-based Variational Inference,” Proceedings of the 2018 Korea Software Congress, pp. 587-589, December 2018.
[17] Y. Oh, M. Kim, and W. Kim, “Korean Movie Review Sentiment analysis Using Parallel Stacked Bidirectional LSTM Model,” Proceedings of the 2018 Korea Computer Congress, pp. 823-825, June 2018.
[18] A. Mousa and B. Schuller, “Contextual Bidirectional Long Short-Term Memory Recurrent Neural Network Language Models: A Generative Approach to Sentiment Analysis,” Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (EACL), pp. 1023-1032, Valencia, Spain, April 2017.
[19] 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.
[20] M. Kim, S. Hong, and I. H. Suh, "Convolutional Neural Network Based Filtering-Scoring System for Rating Prediction of Travel Attractions using Social Media," Journal of the Institute of Electronics and Information Engineers, Vol. 56, No. 9, pp. 891-897, Sep. 2019.
[21] J. Yeon, J. Myung, J. Shim, and S. Lee, "Characteristic Set and Collaborative Filtering for Review Selection," Proceedings of the 2012 Korea Computer Congress, pp. 43-45, 2012.
[22] E. F. Cardoso, R. M. Silva, and T. A. Almeida, "Towards Automatic Filtering of Fake Review," Journal of Neurocomputing, Vol. 309, pp. 106-116, May 2018.

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 and analysis, artificial intelligence, and mobile computing.