위키피디아 문서를 이용한 전문용어 N-Gram 구축

최준호*, 고병규*, 이준*, 김판구*
*조선대학교 컴퓨터공학부
e-mail : xdman@paran.com

Constructive Method for Terminology N-Gram using Wikipedia Document

Jun-Ho Choi*, Byung-Gyu Go*, Jun Lee*, Pan-Koo Kim*
*Dept. of Computer Engineering, Chosun University

요 약

자연어 처리 분야 중 현재 가장 활용도가 높은 분야는 질의어 추천기능, 단어 자동 완성 기능 등으로 정보검색에서 사용자가 입력한 문자들을 바탕으로 질의어를 완성해주는 것이다. 이러한 기능을 위해서는 문서 내용을 고려한 N-Gram 데이터 구축이 필수적이다. 본 논문에서는 문서 편집기나 검색엔진의 질의어 추천 등에 많이 활용되는 N-Gram 데이터의 전문용어별 구축을 위해 위키피디아 문서를 이용하는 방안을 제시하였다.

1. 서론

자연어 처리는 인간이 보통 사용하는 언어를 컴퓨터에 인식시켜 처리하는 것으로 정보검색, 질의응답, 자동번역 및 통역 등이 이에 포함된다. 자연어가 가지는 단어와 단어의 문법적, 의미적 상호관계 등으로 인해 컴퓨터가 처리하기에는 다소 곤란하고 연구가 깊이 진행되고 있음에도 아직 크게 실용화에 이르지 못하고 있는 것이 현실이다. 이 중 현재 가장 활용도가 높은 분야는 질의어 추천기능, 단어 자동 완성 기능 등으로 정보검색에서 사용자가 입력한 문자들을 바탕으로 질의어를 완성해주는 것이다. 이러한 기능을 위해서는 문서 내용을 고려한 N-Gram 데이터 구축이 필수적이다[03][04].

N-Gram은 대표적으로 문서 편집기에서 많이 사용되며, 최근에는 검색엔진에서 질의어 추천(Query Recommendation), 음성 인식 후처리 분야, 및 장애인의 동공기반 타이핑에 까지 활용되고 있다. 이러한 N-Gram 데이터 중 가장 대표적인 것은 구글 N-Gram으로 그 규모가 너무 방대하기 때문에 실제 활용 단계에서는 어려움이 발생한다. 위키피디아의 문서는 해당 분야의 전문가들에 의해 정형적인 어휘와 형식으로 작성되고 있다. 또한 위키피디아에서 작성된 문서들은 DBPedia에서 다양한 연구에 활용될 수 있도록 Infobox, Title, Short/Long Abstract, Image 등으로 나누어 제공하고 있고, 약 2,900,000개의 문서들을 분석하여 제공하고 있다[01].

본 논문에서는 이러한 위키피디아 문서 내 전문용어 N-Gram(Uni-Gram, Bi-Gram, Tri-Gram, 4-Gram, 5-Gram)을 추출할 수 있는 방안을 제시하고자 한다.

2. 위키피디아 문서 내 전문용어 추출

위키피디아 문서에서 전문 용어 추출 모델은 크게 전문 용어를 추출하기 위한 전처리 과정과 State Machine 기반 전문용어를 인식하는 과정으로 이루어진다. 위키피디아는 웹 기반의 무료 온라인 백과사전 서비스로 전 세계 모든 사용자들이 정보의 생산자 혹은 가공자로 참여하여 광범위한 지식 플랫폼을 제공해주고 있다. 이는 집단지성이 가장 잘 반영된 곳으로, 특정 도메인의 전문용어를 추출하는데 있어서 훌륭한 대상이다. 일반적으로 전문용어 추출을 위해 통계적인 방법을 이용하는 경우, 신뢰성이 높은 결과를 산출하기 위해 대형 코퍼스(Corpus)를 이용하는데, 이는 대형 코퍼스를 이용하는 경우, 옳지 않은 데이터, 즉, 잡음(Noise)이 통계적인 방법으로 걸러져 양질의 결과를 산출해내는 효과를 기대하기 위함이다. 위키피디아의 경우, 위키피디아에서 제공하는 정보는 잡음이 없으며 동시에 어떠한 용어에 대해 가장 정확하고 충분하게 기술하고 있다고 가정한다면, 대용량 코퍼스를 이용하는 것보다 더욱 양질의 결과를 추출할 수 있다[01][02].

전문용어 추출 과정은 상태머신(state machine)을 기반으로 이루어진다. 전문용어(Terminology)란 주어진 도메인 안에서 의미를 가지고 있는 단어들의 집합으로 도메인 내에서 사용되는 개념을 표현시켜 줌으로서 주제를 특성

화해주는 어휘적 단위를 말한다. 이러한 전문용어는 하나의 도메인을 이해하는데 필요한 요소이기 때문에 특정 도메인에 대한 기계번역이나 정보검색을 보다 효율적이고, 정확히 수행하기위해서 전문용어에 대한 언어자원은 중요하다. 낮은 모호성(Ambiguity)과 높은 특정성(Specificity)을 지닌 전문용어는 특정 도메인을 개념화하거나 도메인 온톨로지를 구축할 때 매우 유용하다. 본 논문에서는 문서내 전문용어를 자동으로 추출하기 위하여 그들의 출현 형태를 분석하였다.

전문용어의 형태결합 방식은 매우 다양하다. 해당 도메인에 출현하는 대부분의 전문용어들은 복합명사 형태로 출현하며, 크게 두 가지의 결합 형태로 나눌 수 있다. 하나는 단일어절(Singleton Term) 즉, 띄어쓰기가 없는 한 어절로 나타나는 단순한 결합 형태이고, 다른 하나는 다중어절(Multi-word Term)의 형태로 띄어쓰기가 나타나며 앞의 어절성분과 의미적으로 관련이 있는 두 어절이상으로 이루어진 복합명사이다. <표 1>은 이를 기반으로 전문용어의 출현 형태를 파악하고 있다.

<표 1> 전문용어의 구조

Structure	example
1. 단일명사(NN,NNS,NNP)	(chromosome, NN), (genes, NNS), (DNA, NNP)
2. 복합명사((1+ 1+ (1))	Ribonucleic acid, Nucleic Acids,
3. 복합명사(JJ+1 + (1))	Recombinant DNA, oxidative lesions

단일어절로 이루어진 전문용어들은 약어로 이루어진 경우가 많아서 주로 NNP로 파악되는 경우가 많았으며 <표 1>의 2번과 같은 경우는 명사와 명사 또는 수식어와 명사의 결합으로 전문용어의 완전한 표현이나 복합어를 통한 새로운 용어들이 이에 해당한다. <표 1>의 3번에서는 기존의 전문용어들에 수식어가 결합함으로 새로운 의미를 갖게 되는 경우이다.

3. 전문용어 N-Gram 구축

전문용어 추출을 위해 품사 태깅된 각 단어들을 정해진 패턴에 따라 핵심어 후보를 생성한다. 용어의 JJ(형용사) 태그와 NN(명사) 태그를 기준으로 하여 Single Term(단일어절)과 Multi-word Term(복합어절)을 생성한다. 핵심어 추출을 위한 알고리즘은 다음과 같다.

전문용어 추출은 품사 태깅된 각 단어마다 정해진 패턴에 의해 수행된다. <표 2>는 태깅된 어휘의 형용사(JJ)와 명사(NN)를 기준으로 단일어(Singleton Term)와 복합어(Multi-word Term)를 추출하는 알고리즘이다.

<표 2> 전문용어 추출 알고리즘

```
while taggedTerms:
    term, tag, norm = taggedTerms.pop(0)
    if state == SEARCH and tag.startswith('N'):
    // 검색 중이고, 명사로 시작되는 경우
        state = NOUN //  복합 어절 탐색 상태
        _add(term, norm, multiterm, terms)
    elif state == SEARCH and
            tag == 'JJ' and term[0].isupper():
    // 검색 중이고, 형용사로 시작되는 경우
        state = NOUN
        _add(term, norm, multiterm, terms)
    elif state == NOUN and tag.startswith('N'):
    // 복합 어절 탐색 중이고, 명사로 시작되는 경우
        _add(term, norm, multiterm, terms)
        // 검색된 용어와 복합 어절 용어를 각각 추가
    elif state == NOUN and not tag.startswith('N'):
    // 복합 어절이 아닌 경우
        state = SEARCH
        if len(multiterm) > 1:
          word = ' '.join([word for word, norm in multiterm])
          terms.setdefault(word, 0)
```

<표 2>에서 먼저 태그가 NN 또는 JJ로 시작하는지를 파악한다. NN으로 시작할 경우 핵심어 후보에 추가한 후 Multi-word 탐색을 시작한다. Multi-word 탐색 중이며, 다음 태그가 NN일 경우는 현재 단어를 핵심어 후보에 추가한 후 현재까지의 용어를 저장하고 다음 단어의 tag를 파악한다. 현재 단어의 태그가 NN이 아닐 경우 이전까지의 용어가 2어절 이상인지를 파악하고 핵심어 후보에 추가한다. JJ로 시작할 경우 현재 단어를 저장 후 multi-word 탐색한다. JJ이후 단어의 tag가 NN일 경우 현재까지 용어를 핵심어로 추가하고 저장한 후 다음 단어 파악한다.

<표 3> N-Gram 추출을 위한 Wikipedia 문서 전처리

1. Wikipedia 문서 오픈
2. Paragraph의 (p|table|form) Tag 삭제
3. (div|br|tr) Tag 삭제
4. HTML 코드 변환
5. Unicode 중 특수문자 공백 처리
6. Paragraph 마지막에 CR/LF 처리
7. 순수 문장 추출 및 저장
8. 전문 용어 인식 및 고유명사 추출
9. 고유명사의 Word N-Gram 분할

<표 3>에서와 같이 전처리된 위키피디아 문서에서 특정 도메인의 전문용어 N-Gram은 추출된 전문용어를 N-Gram화하고 이에 대한 출현빈도수를 카운트하기 위해서 <표 4>와 같은 N-Gram 알고리즘을 이용한다.

<표 4> 전문용어 N-Gram 추출 알고리즘

```
def nnp_ngrams(original_text, highlight=3, minsize=0):
    minsize = minsize-1
    if minsize<0:
        minsize = 0
    tokens = nltk.wordpunct_tokenize(original_text)
    tagged = nltk.word_tokenize(original_text)
    tagged =  nltk.pos_tag(tokens)
    doc_length = len(tokens)
    counter = 0
    counter2 = 0
    if highlight==0:
        concated_test = doc_length
    else:
        concated_test = highlight
    list_of_NNPs = []
    while counter < (doc_length-1):
        while counter2 < concated_test:
          counter2 = counter2+1
          counter3 = 0
          temp_array = []
          all_nnp = True
          while counter3 < counter2:
            if counter < (doc_length-counter3):
              temp_array.append(tokens[counter+counter3])
            if tagged[counter+counter3][1] != 'NNP':
                        all_nnp = False
              counter3 = counter3+1
            counter3 = 0
            if all_nnp == True:
                if(len(temp_array)>minsize):
                    list_of_NNPs.append(temp_array)
        counter2 = 0
        counter = counter+1
    import itertools
    list_of_NNPs.sort()
    unique_NNPs = list(list_of_NNPs for list_of_NNPs,_ in
          itertools.groupby(list_of_NNPs))
    return unique_NNPs
```

<표 4>는 특정 문장의 전문용어 중 고유명사를 N-Gram 형식으로 추출하는 알고리즘이다. 입력 데이터는 추출하고자 하는 문장과 추출하고자 하는 N-Gram의 크기(highlight), 그리고 추출 N-Gram의 최소 사이즈(minsize) 등이다. 입력된 문장의 품사 중 NNP(고유명사)에 해당되는 단어만 취하기 위한 옵션을 부여했고, 단일명사 및 복합명사 추출은 해당 품사 태그를 수정한다. 제시한 알고리즘의 실험을 위해 <표 5>와 같은 문장을 적용하여 N-Gram 추출 결과를 <표 6>과 같이 얻을 수 있다.

<표 5> 전문용어 추출을 위한 샘플 문장

Mozilla also began offering a new feature, Mozilla Sync, which allows people with the Firefox browser to automatically sync their bookmarks, open browser tabs and settings between multiple computers, mobile phones and Android-based tablet computers. The Mozilla Sync feature will not work with Apple's iOS devices, which include the iPhone and iPad, because the browser is not available on Apple's mobile platform.

<표 6> 전문용어 중 고유명사 N-Gram 추출 결과

```
고유명사 N-Gram 추출 결과 :
3 ['Mozilla']
2 ['Sync']
2 ['Mozilla', 'Sync']
2 ['Apple']
1 ['iPhone']
1 ['iPad']
1 ['iOS']
1 ['Firefox']
1 ['Android']
```

<표 5>에서 제시된 샘플 문장을 본 논문에서 제시한 N-Gram 추출 알고리즘에 적용하면 <표 6>과 같은 결과를 얻을 수 있다. <표 6>의 결과는 전문용어로 추출된 용어 중 고유명사에 해당되는 어휘와 출현 횟수를 표시한 예로, Bigram을 비롯하여 Trigram, 4-Gram, 5-Gram에 해당되는 N-Gram 형태별로 추출한다.

4. 결론

본 논문에서는 문서 편집기나 검색엔진의 질의어 추천 등에 많이 활용되는 N-Gram 데이터의 전문용어별로 구축하기 위해 위키피디아 문서를 이용하는 방안을 제시하였다. 이를 이용하면 사용자가 관심을 갖는 특정 도메인의 어휘만을 추출할 수 있으며, 보다 전문적인 N-Gram을 구축하는데 효과적임을 보였다. 향후, 다양한 확률측정 방법을 적용하여 보다 의미적인 N-Gram을 구축하는 방안을 제시할 예정이다.

Acknowledgements

"이 논문은 2010년도 정부(교육과학기술부)의 재원으로 한국연구재단의 기초연구사업 지원을 받아 수행된 것임(No. 2010-0011656)"

참고문헌

[01] M Hepp, K Siorpaes, D Bachlechner. "Harvesting WikiConsensus:Using Wikipedia Entries as Vocabulary for Knowledge Management", IEEE InternetComputing 2007.

[02] F Wu, D Weld. "Automatically refining the Wikipedia infobox Ontology". portal.acm.org2008.

[03] W. B. Cavnar and J. M. Trenkle, "N-Gram-Based Text Categorization," In Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval, pp. 161-175, 1994.

[04] M. G. Hwang, D. J. Choi, H. G. Lee, and P. K. Kim, "Text Editor based on Google Trigram and its Usability", submitted to the 26th Symposium on Applied Computing 2011