[UCI]I804:24011-200000267235

2019년 2월

박사학위 논문

워드 임베딩을 이용한 미등록어의 의미적 대체

조선대학교 대학원

컴퓨터공학과

김 정 인

워드 임베딩을 이용한 미등록어의 의미적 대체

Word Embedding based Semantic Alternation for Out-of-Vocabulary Word

2019년 2월 25일

조선대학교 대학원

컴퓨터공학과

김 정 인

워드 임베딩을 이용한
미등록어의 의미적 대체

지도교수 김 판 구

이 논문을 공학박사학위신청 논문으로 제출함

2018년 10월

조선대학교 대학원

컴퓨터공학과

김 정 인

김정인의 박사학위논문을 인준함

위원장 조선대학교 교수 정 일 용 (인)

위 원 조선대학교 교수 반 성 범 (인)

위 원 조선대학교 교수 양 희 덕 (인)

위 원 중앙대학교 교수 정 재 은 (인)

위 원 조선대학교 교수 김 판 구 (인)

2018년 12월

조선대학교 대학원

목 차

그 림 목 차

표 목 차

ABSTRACT

Word Embedding based Semantic Alternation for Out-of-Vocabulary Word

Jeongin Kim
Advisor: Prof. Pankoo Kim, Ph.D.
Department of Computer Engineering,
Graduate School of Chosun University

Natural language refers to a language developed naturally to express intention or exchange opinions as a group of human passes through historically. Unlike artificial languages, natural languages are often ambiguous and have many omitted words or paraphrases. Furthermore, social knowledge is also required to properly comprehend a natural language; it is very difficult for computers to understand it. Processing a natural language using a computer is referred to as natural language processing; one of the primary research goals in the field of natural language processing is to understand and imitate natural languages in a computer environment.

Morphological analysis in natural language processing refers to the analysis of a word in a sentence in terms of morphemes, the smallest unit

of meaning. A morpheme is the smallest unit of a word which has a certain meaning in linguistics; moreover, it is a unit of a word whereby the meaning disappears if analyzed further. Although it can be the word itself, in general, it is a unit smaller than the word. One of the most common problems in analyzing a natural language is finding a word that is similar to a out-of-vocabulary word.

When a person understands a sentence containing a out-of-vocabulary word, he/she determines its most appropriate meaning with a substituted word by using the context to determine the meanings of words based on the conventional concept system that has been learned. The core of such a concept originates from the distribution hypothesis; this hypothesis is explained by John Rupert Firth's famous saying, "You shall know a word by the company it keeps." In other words, words that tend to appear together in similar contexts tend to have similar meanings.

This study proposes the use of the Word2VnCR algorithm that substitutes a out-of-vocabulary word with a similar word. To extract similar candidates for out-of-vocabulary words, word-embedding is learned using a training dataset; afterwards, similar word candidates are extracted. For the similar word candidates that have been extracted, the sematic similarities of adjacent words around the out-of-vocabulary word are measured, and a similar word that has the highest similarity value is selected. This word replaces the out-of-vocabulary word.

To prove the excellence of the proposed Word2VnCR algorithm, a comparative experiment was performed using the Word2VnCR algorithm and the Word2Vec algorithm for similar word substitutions of out-of-vocabulary words from the NUS sms Corpus. The results showed that the Word2VnCR algorithm showed higher performance than the Word2Vec algorithm in terms of accuracy for the substitution of a out-of-vocabulary word with a similar word.

As the final outcome, the Word2VnCR algorithm proposed in this study showed high accuracy when substituting a out-of-vocabulary word with a similar word. However, the result of this experiment is affected depending on how the training dataset is built. Similar word candidates of out-of-vocabulary words cannot be accurately extracted because the word-embedding learning of training dataset is not properly done. Therefore, the Word2VnCR algorithm needs the task of adding texts having the following characteristics to the training data: few out-of-vocabulary words appear, and the words adjacent to these out-of-vocabulary words are composed based on sematic meanings.

Ⅰ. 서 론

1. 연구 배경

인간 집단이 역사적으로 지나오는 동안에 의사 전달이나 의견 교환을 하기 위하여 자연적으로 발생한 언어를 자연 언어(自然言語, Natural Language) 혹은 자연어(自然語)라 한다[1]. 자연 언어는 인공언어와 달리 애매함이나 그때마다 여러 가지 생략이나 환언함[1]이 있다. 더욱이 사회적인 지식 등도 필요하기 때문에 컴퓨터가 이해하는 것은 매우 곤란하다[2]. 인공 언어(Artificial Language)는 자연 언어와 대치되는 개념이다[3]. 컴퓨터에서 사용하는 언어는 보통 인공 언어라고 하며 사용 방법이나 쓰는 방법 등이 매우 자세히 정해져 있다[2]. 인공 언어는 컴퓨터가 참(True)과 거짓(False) 두 개의 상태밖에 표현하지 않는 논리 회로(Logic Circuit)[2]로 구성되어 있는 데 기인한다. 컴퓨터를 사용하여 자연 언어를 처리하는 것을 자연어 언어 처리(自然言語處理, Natural Language Processing)라고 하며 컴퓨터 환경에서 자연 언어를 이해하고 모방하는 것이 자연 언어 처리 분야의 연구 목표 중 하나이다. 자연 언어 처리 또는 자연어 처리(自然語處理)는 사람의 언어 현상을 컴퓨터와 같은 기계를 이용해서 모사 할 수 있도록 연구하고 이를 구현하는 분야이다[4]. 자연 언어 처리는 연구 대상이 언어이기 때문에 언어 자체를 연구하는 언어학과 언어 현상의 내적 기재를 탐구하는 언어 인지 과학과 연관이 깊다[4]. 자연 언어의 구현을 위하여 수학적 통계적 도구를 많이 활용하며 특히 기계학습(機械學習, Machine Learning) 도구를 많이 사용하는 대표적인 분야이다.

자연 언어 처리에서 말하는 형태소(形態素, Morpheme) 분석이란 문장의 어절을 최소의 의미 단위인 형태소로 분석하는 것을 의미한다. 형태소는 언어학에서 일정한 의미가 있는 가장 작은 말의 단위로 더 분석하면 뜻이 없어지는 말의 단위이다[5]. 단어 그 자체가 될 수도 있고 일반적으로는 단어보다 작은 단위이다[4]. 형태소 분석

1) 어떤 내용의 말을 더 적절한 다른 형식의 말로 바꾸어 말함.

2) 디지털 신호에서 논리 연산을 수행하는 회로를 논리 회로라고 하며, 논리 회로를 구성하는 기본 단위를 논리 게이트라고 함.

단계에서 문제가 되는 부분은 미등록어, 오탈자, 띄어쓰기 등에 의한 형태소 분석의 오류, 중의성 해소, 신조어 처리 등이 있다[4]. 이들은 형태소 분석에 치명적인 약점이라 할 수 있다. 자연 언어를 분석하는데 가장 대표적인 문제점은 앞서 언급한 형태소 분석의 오류 중 단어를 변칙적으로 표기한 미등록어(Out-of-Vocabulary Word)[3]를 의미적으로 유사한 단어로 대체 하는 부분이다.

단어(Word, 單語)[4]는 문법 단위 중 기본이 되는 언어 단위의 하나이다[6]. 문장 구성상 단어와 형태소는 매우 헷갈리나 그림 1에서와 같이 단어는 형태소보다는 상위개념이다.

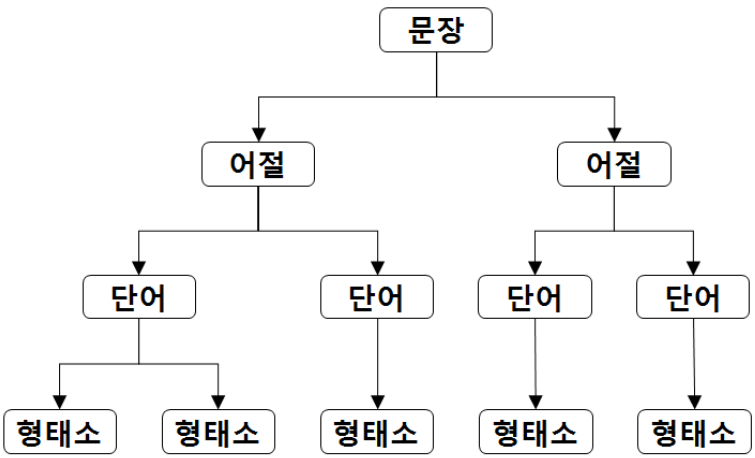


그림 1. 문장의 구조화

구체적으로 말하면 형태소는 단어의 부분집합에 속하기 때문이다. 형태소면서 동시에 단어인 말들이 존재하기 때문에 매우 헷갈리기 쉽다. 특히 단어에 대한 정의가 매우 모호한데 단어의 정의는 쉽게 말해서 형태소보다 큰 범위이며 어절보다 작은 단위정도로 이해하면 된다.

3) 본 연구에서는 형태소 분석의 오류 중 명사의 품사를 지닌 미등록어를 의미함.

4) 분리하여 자립적으로 쓸 수 있는 말이나 이에 준하는 말. 또는 그 말의 뒤에 붙어서 문법적 기능을 나타내는 말.

표 1. 미등록어를 포함하고 있는 영어 단문의 예

영어 단문	Microsoft UK so a stupid Windows 10 update just killed all my **sw** thanks a lot I really **appericate** it

사람은 표 1과 같이 미등록어가 포함된 문장을 이해할 때, 기존에 학습된 개념체계를 바탕으로 문맥 내 동시출현단어들의 의미를 이용해 미등록어를 의미가 가장 유사한 단어로 대체해 문장을 이해한다. 이러한 개념의 핵심은 분포 가설(Distributional Hypothesis)에 기인하며, 이 가설은 언어학자 존 루퍼트 퍼스(John Rupert Firth)의 유명한 말 “단어는 포함된 문맥 속에서 이해할 수 있다.”로 설명되곤 한다. 즉, 비슷한 맥락에서 함께 나타나는 경향이 있는 단어들은 비슷한 의미를 가지는 경향이 있다는 것을 의미한다[7].

표 2. 미등록어와 의미적 대체 후보 단어의 예

미등록어	미등록어의 의미적 대체 후보 단어
sw	software, Samurai Warriors, Snow White, Star Wars, Strike Witches, Smith & Wesson, Southwest Airlines, Southwest, Schweinfurt, Swietochlowice, Social worker, Adobe Shockwave, Shortwave, Sport wagon, Station wagon, switch, Silver Week
appericate	appreciate, appreciant, appreciable, appreciated, appreciates, appredicate

표 2는 표 1의 영어 단문에 포함된 미등록어의 의미적 대체 후보 단어를 나타내고 있다. 사람은 미등록어인 ‘sw’의 대체 단어를 주변 문맥의 단어들을 통해 ‘software’로 판단할 수 있으며, 오탈자인 ‘appericate’의 경우에는 ‘appreciate’로 판단하여 문장의 의미를 유추(類推, Analogy)[5]할 수 있다. 하지만 컴퓨터와 같은 기계는 미등록어를 의미적으로 유사 단어로 대체하는 방법에 있어 아직 해결하지 못한 문제들이 다수 존재한다.

5) 연합되어 있는 언어가 서로 일정한 어형상의 의미와 기능, 혹은 음성형식 등의 유사성을 가졌으나 그 중 일부의 언어형식이 이 특색에서 벗어나는 일이 있을 경우, 이 일부의 언어형식을 다수의 언어형식의 공통된 특색에 맞도록 그 언어형식을 새로운 언어형식으로 만들어내는 심리적 과정.

본 연구에서는 형태소 분석 오류인 미등록어를 의미적으로 유사한 단어로 대체하는 연구를 수행하고자 한다. 미등록어의 대체 후보 단어 추출은 대체 단어가 미등록어의 주변 인접 단어와 함께 출현할 확률이 높음을 이용한다. 대체 단어의 선정은 대체 후보 단어와 미등록어의 주변 인접 단어와 의미적 유사도를 측정해 높은 유사도 값을 가지는 대체 후보 단어를 미등록어의 대체 단어로 선정한다. 이를 이용해 미등록어를 의미적으로 유사한 단어로 대체함으로써 형태소 분석 오류를 해소하는 연구의 성능을 향상시키고 보다 지능적으로 문서를 분석하고자 한다.

2. 연구 내용 및 범위

본 연구는 지능적으로 문서를 분석하기 위하여 영어 단문에 출현하는 미등록어를 의미적으로 유사한 단어로 대체함에 있어서 미등록어와 의미가 유사한 대체 후보 단어를 추출하는 방법과 대체 단어를 선정하기 위해 대체 후보 단어와 미등록어의 인접 단어와 의미적 유사도를 측정하는 방법을 제안한다. 미등록어가 대체 될 후보 단어는 워드 임베딩 방법을 이용해 미등록어의 대체 후보 단어를 추출한다. 이는 두 단어가 동일한 문헌이나 문단 또는 문장과 같이 특정 범위 내에서 같이 발견된다는 이론을 바탕으로 한다[8].

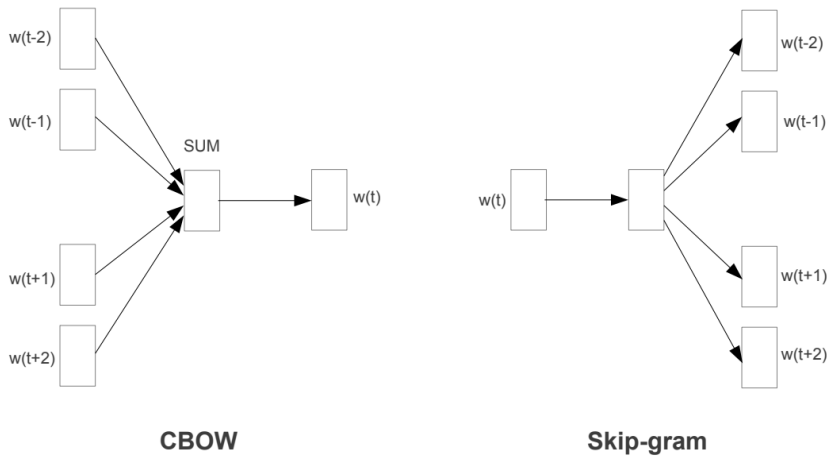


그림 2. 워드 임베딩 모델의 예

워드 임베딩 방법은 문장의 문맥을 기반으로 단어를 예측하는 방법과 단어가 주어진 주변 단어를 예측하는 방법이 있다. 본 연구에서는 워드 임베딩 방법의 문맥을 기반으로 단어를 예측하는 방법을 통해 미등록어의 대체 후보 단어를 추출한다. 그림 2는 워드 임베딩 방법을 대표하는 모델의 예 나타낸다.

미등록어가 대체 될 단어는 앞서 언급한 워드 임베딩 방법을 이용해 추출한 대체 후보 단어와 미등록어의 주변 인접 단어와 의미적 유사도 측정 방법을 이용해 유사도가 높은 값을 가지는 대체 단어로 선정한다. 이는 두 단어 개념간의 의미 유사성을 워드넷 계층 구조를 기반으로 측정할 수 있다는 이론을 바탕으로 한다. 이때, 단어 간의 의미적 유사도는 워드넷에 정의된 개념간의 관계성을 기반으로 한다. 본 연구에서는 영어 단문 내 형태소 분석 오류인 미등록어를 의미적으로 유사 단어로 대체하기 위해 단어의 벡터(Vector)값과 워드넷의 개념 관계(Conceptual Relation)를 이용한 Word2VnCR 알고리즘을 제안한다. 또한 다양한 실험을 통하여 본 연구의 유효성과 효율성을 입증하고 Word2VnCR 알고리즘의 우수성을 보인다.

본 논문의 구성은 다음과 같다.

본 장인 서론에 이어 2장에서는 본 연구의 이론적 배경이 되는 형태소 분석 오류인 미등록어를 해소하는 기존 연구들에 대해 살펴본다. 그리고 본 연구 수행에 있어 이론적 배경이 되는 관련 연구들을 기술해 연구 내용의 이해를 돕는다.

3장에서는 Word2VnCR 알고리즘의 기반이 되는 워드 임베딩 방법과 워드넷 유사도 측정 방법에 대해 예제와 함께 상세히 기술한다. 워드 임베딩 방법은 미등록어가 대체 될 대체 후보 단어를 추출하는 방법이며, 워드넷 유사도 측정 방법은 대체 후보 단어 중 미등록어가 대체될 대체 단어를 선정하는 방법이다.

4장에서는 3장에서 입증하고 제안한 미등록어를 의미적으로 유사한 단어로 대체하는 Word2VnCR 알고리즘을 제시한다. 미등록어의 의미적 단어 대체를 위한 Word2VnCR 알고리즘의 전처리 과정, Word2VnCR 알고리즘의 수행 과정을 예제와 함께 상세히 설명한다.

5장에서는 Word2VnCR 알고리즘의 실험결과와 Word2Vec 알고리즘의 실험결과를 비교하여 Word2VnCR 알고리즘의 의미적 대체 성능을 평가한다. 또한 문장 내 미등록어의 개수별로 Word2VnCR 알고리즘의 성능의 차이를 비교 평가한다.

마지막으로 6장에서는 본 연구의 결론을 맺고 향후 연구방향을 제시함으로써 본 연구의 필요성과 활성 방안에 대해 기술한다.

Ⅱ. 관련 연구

1. 미등록어의 의미적 대체 연구개요

미등록어를 의미적으로 유사한 단어로 대체하는 연구는 자연어 처리 과정에서 초기단계의 작업으로써 컴퓨터가 자연어를 처리하고 이해하는 분야와 기계번역의 분야에서 완벽히 해결하지 못한 대상이 되어 왔다[9]. 형태소 분석 오류인 미등록어를 의미적 유사 단어로 대체하는 작업은 기계번역, 정보검색, 시맨틱 웹, 음성 및 텍스트 처리 등의 분야에서 자연어로 표현된 문서 정보를 컴퓨터가 이해하고 그 의미를 분석하게 하는 기술이다[10]. 사람이 작성한 글을 이해하고 글쓴이의 의도를 분석하기 위하여 자연어로 표현되는 정보를 이해시키는 기술은 그 자체가 최종 목표가 되는 것이 아니고 보다 정확하고 지능적인 서비스를 실현하기 위한 필수 과정이다.

본 장에서는 지능적으로 문서를 분석하기 위하여 기존 국내외 연구에서 다양하게 수행된 미등록어의 의미적 대체 방법들을 살펴보고 기존 연구의 한계점 및 개선사항을 도출해 본다. 특히, 기존의 선행 연구들 중 미등록어의 의미적 단어 대체 연구를 수행한 연구를 중점적으로 분석해 이를 극복할 수 있는 방법을 모색한다. 이런 내용들을 통해 3장부터 전개되는 연구 내용의 이해도를 높이고 미등록어의 의미적 대체 연구의 필요성과 중요성을 살펴본다.

2. 사전 기반 미등록어 대체 기법

사전(Dictionary)을 기반으로 하는 미등록어의 단어 대체 방법은 미등록어의 문자열과 사전에 등록되어 있는 단어의 문자열을 비교하여 두 단어가 가지고 있는 부분문자열(Substring) 또는 부분열(Subsequence)의 길이를 측정해 미등록어를 사전에 등록되어 있는 단어로 대체한다. 최장 길이 공통 부분문자열(Longest Common Substring) 방법은 두 단어에서 부분문자열을 최장 공통 부분열(Longest Common Subsequence) 방법은 두 단어에서 부분열을 추출한다. 부분문자열은 두

단어에서 문자(Character)가 작성된 순서와 길이가 일치하는 문자열을 의미하며, 부분열은 몇몇 문자가 지워지나 작성된 순서는 바뀌지 않는 문자열을 의미한다. 즉 문자열의 부분문자열은 항상 부분열이 되지만 부분열은 항상 부분문자열이 되는 것은 아니다[11].

$$arr[i][0] = 0(0 \le i < m), arr[0][j] = 0(0 \le j < n)$$
$$arr[i][j] = \begin{cases} if(A[i-1] == B[j-1])\ arr[i][j] = arr[i-1][j-1] + 1 \\ else\ arr[i][j] = 0 \end{cases} \quad (1)$$

최장 길이 공통 부분문자열은 수식 1과 같이 정의한다. 수식 1에서 은 단어 의 문자열 길이를 n은 단어 B의 문자열 길이를 의미한다. 예를 들어 단어 A는 'abcdabcd'이고 단어 B는 'efcdagh'일 때, 수식 1을 이용한 두 단어의 최장 길이 공통 부분문자열은 'cda'가 되며 길이 값을 3이 된다.

$$LCS(X_i, Y_j) = \begin{cases} \emptyset & if\ i = 0\ or\ j = 0 \\ LCS(X_{i-1}, Y_{j-1}) + 1 & if\ x_i = y_i \\ longest(LCS(X_i, Y_{j-1}), LCS(X_{i-1}, Y_j)) & if\ x_i \neq y_i \end{cases} \quad (2)$$

최장 공통 부분열은 수식 2과 같이 정의한다[12]. 단어 X의 문자열의 길이는 m이고, 단어 Y의 문자열의 길이는 n일 때, 두 단어의 문자 집합은 $X = (x\ , x_2, \cdots x_m)$와 $Y = (y_1, y_2, \cdots y_n)$로 나타낸다. 단어 X와 Y의 최장 공통 부분열을 구하기 위해서 두 단어의 집합 x_i와 y_j를 비교한다. 두 단어 중 하나의 문자열의 길이가 0이라면 최장 공통 부분열의 길이는 0이 된다. 서로 일치하는 문자가 전혀 존재할 수 없기 때문이다. 두 단어의 문자열 길이가 같다면 $LCS(X_{i-1}, Y_{j-1})$는 최장 공통부분 수열에 포함되는 부분열이다. 두 단어의 문자열 길이가 같지 않다면 $LCS(X_i, Y_{j-1})$와 $LCS(X_{i-1}, Y_j)$ 중 더 긴 부분열이 얻어진다. 예를 들어 단어 X는 'acbdegcedbg'이고 단어 Y는 'begcfeubk' 일 때, 'bee', 'beeb'은 부분열이 될 수 있지만 두 단어에서 최장 공통 부분열은 'begceb'가 되며 부분열의 길이 값은 6이 된다.

최장 길이 공통 부분문자열 방법과 최장 공통 부분열 방법은 위와 같이 문자열의 길이를 측정해 미등록어가 대체될 대체 후보 단어의 우선순위를 구분할 수 있다. 하지만 최장 길이 공통 부분문자열 방법과 최장 공통 부분열 방법은 부분문자열 또는 부분열이 여러 가지로 나올 수 있다는 점과 부분문자열 또는 부분열의 길이가 같을 경우 미등록어의 대체 후보 단어 중 대체 단어를 선정하는데 있어 문제를 갖는다. 또한 사전에 등록된 단어만을 비교하기 때문에 사전에 등록되어 있지 않은 단어를 미등록어의 대체 단어로 사용할 수 없다는 문제점이 있다.

3. 엔그램 기반 미등록어 대체 기법

엔그램(N-Gram)을 기반으로 하는 미등록어의 단어 대체 방법은 미등록어가 등장한 단어열 주변의 단어를 기반으로 미등록어가 작성된 부분의 대체 단어를 확률적 수치를 측정해 미등록어를 대체한다. 엔그램 기법은 형태소 분석 오류인 미등록어의 문제를 해소하는 대표적인 확률적 방법이다. 엔그램 방법은 기본적으로 학습 데이터를 이용해 확률을 구하는 것으로 문장에 주어진 단어나 음절의 값을 구하거나 분류하는 데에 있어서 주위의 개의 단어나 음절을 정보로서 이용하는 방법이다[13].

자연 언어의 문법은 매우 복잡하기 때문에 인간의 발화(發話)[6]는 정규 문법에 따르지 않는 경우가 많다[14]. 이전의 문장 인식에서는 단어 인식을 한 후에 규칙에 따라 기술된 문법 처리를 적용하고 마지막에 오류를 수정하는 방법을 사용하였으나 근래에는 언어 모델 방법을 적용해 문장을 인식한다[14]. 엔그램 모델은 잡음 채널(Noisy-Channel) 모델에서 높은 성능을 얻기 위해 연구되어 1980년대부터 음성인식분야에 사용되었으며, 그 이후에 기계번역, 철자교정, 단어 의미 클래스 분류 등에 높은 성능을 보이며 지금까지 널리 사용되고 있다[15, 16]. 엔그램은 확률적 언어 모델의 대표적인 방법으로서, n개 단어의 연쇄를 확률적으로 표현해 두면 실제로 발성된 문장의 기록을 계산할 수 있다[14]. 그림 3은 일반적인 잡음 채널 모델을 나타낸다.

6) 입을 열어 사물을 말하는 행위.

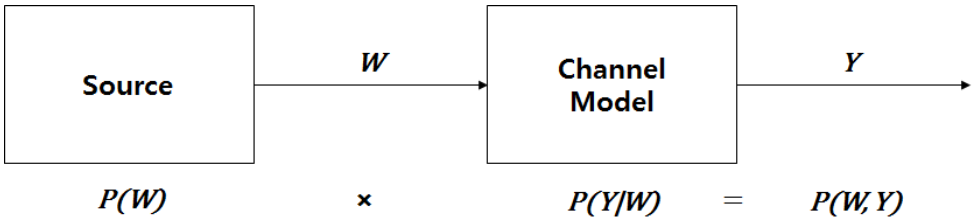


그림 3 잡음 채널 모델

잡음 채널(Noisy-Channel) 모델은 원래 통신 분야에서 처음 제안된 모델로 원래 데이터 가 노이즈를 가진 채널을 통해 전송되면서 실제는 원래 데이터와는 다른 Y를 전송받게 된다[13]. 이러한 잡음 채널 모델을 자연어 처리 분야에 적용하게 되면 Y는 실제 관찰하게 되는 문자열 또는 문장이 되며 W는 찾아야 하는 원래의 문자열 또는 문장이 된다[17]. 즉, W는 사전확률 $P(W|Y)$의 값이 최대가 되도록 하는 W를 찾으면 된다. 원래 데이터가 단어의 시퀀스(Sequence)[7]라고 가정하면 모든 나타날 수 있는 단어는 시퀀스 집합의 원소들 중에서 관찰될 데이터 Y가 주어졌을 때 나타날 확률이 가장 높은 시퀀스를 찾는 것이다. 이를 수식으로 표현하면 다음과 같다. 수식에서 W는 나타날 수 있는 모든 데이터의 집합이며, W는 각각의 데이터이다.

$$\hat{w} = \underset{w_i \in W}{\arg max}\, P(w_i|y) \tag{3.1}$$

$$= \underset{w_i \in W}{\arg max}\, \frac{P(w_i|y}{P(y)} \tag{3.2}$$

$$= \underset{w_i \in W}{\arg max}\, \frac{P(y|w_i)P(w_i)}{P(y)} \tag{3.3}$$

$$= \underset{w_i \in W}{\arg max}\, P(y|w_i)P(w_i) \tag{3.4}$$

7) 연관된 연속의 문자열 또는 단어열을 의미함.

수식 3.1의 $w|y)$는 베이즈 규칙(Baye's rule)에 의해 수식 3.2로 바뀔 수 있으며, 이는 다시 수식 3.3의 형태로 바뀔 수 있다[13]. 수식 3.3에서 분모 $P(y)$는 모든 w_i에 독립이므로 생략할 수 있다[13]. 따라서 결과적으로 찾아야 하는 원래 데이터의 시퀀스 W는 수식 3.4로부터 계산이 가능하다. $P(y|w_i)$는 공산(likelihood)[8] 함수이며, $P(w_i)$는 단어 시퀀스에 대한 사전 확률분포가 된다. 잡음 채널 모델에서 $\hat{W}$ 중 최적의 해 W^*를 구하기 위해서는 현재 주어진 정보만 이용하는 것이 아니라 이미 앞에서 입력되었던 데이터들의 정보까지 이용하여 확률 $P(w_i|y)$를 구함으로써 더욱 좋은 잡음 채널 모델의 성능을 기대할 수 있다[13]. 이것이 엔그램 모델의 가장 가본적인 아이디어이며, 이는 언어학적인 관점에서 본다면 당연한 것으로 원래 데이터가 단어의 시퀀스라고 할 때 현재 단어 w_n이 나타날 확률은 앞에서 이미 나왔던 단어 시퀀스에 따라 달라진다. 즉 문맥을 정보로써 활용해야만 한다는 것이다[13]. 단어의 시퀀스가 나올 확률을 식으로 표현하면 수식 4와 같다.

$$P(w_{1,N}) = P(w_1)P(w_2|w_1)P(w_3|w_{1,2})\ldots P(w_N|w_{1,N-1}) \quad (4)$$

수식 4에서 시퀀스의 길이를 N이라고 했을 때 전체 단어의 시퀀스는 $w_{i,}$ 이 되며 각 단어의 확률을 곱하게 되면 전체 시퀀스가 나타날 확률이 된다. 각 단어의 확률을 계산할 때 문맥을 정보로 활용하므로 문장의 시작부터 각 단어 앞에 나왔던 모든 단어가 문맥의 정보로 사용되고 있다. 어떠한 단어의 시퀀스가 나왔을 때 현재 단어가 나올 조건부 확률이 각 단어의 확률이 된다[13]. 따라서 모든 문맥을 정보로 사용하게 되면 계산량이 매우 많고 학습 데이터에 없던 새로운 단어 시퀀스가 나올 확률이 매우 높다[13]. 시퀀스의 길이인 N의 크기에 따라 계산량이 커질 뿐만 아니라 안정적인 확률계산을 위해 필요한 학습 데이터의 크기 역시 너무나 커지게 되므로 엔그램 모델에서는 가정에 의해 정보로 활용하는 문맥의 크기를 마르코프(Markov) 가정에 의해 정보로 활용하는 문맥의 크기를 $(n-1)$로 제한하는 것이다. 즉, 현재의 단어는 단지 앞의 $(n-1)$개의 단어로부터만 영향을 받는 다고 가정하는 것이다. 이는 매우 극단적인 가정이라고 할 수 있지만 실제 모델에서의 계산량과 데이터의 크기를

8) 어떤 일이 그렇게 될 확실성. 확률에 비례하는 함수.

고려하였을 때는 효율적인 모델이라고 할 수 있다[13]. 만일 $=2$라면 바이그램(Bi-gram) 단어는 앞의 한 단어에만 영향을 받는다고 가정했기 때문에 이 때 단어의 시퀀스 w , 이 나타날 확률은 수식 5와 같이 구할 수 있다[13].

$$\begin{aligned} P(w_{1,N}) &= P(w_1)P(w_2|w_1)P(w_3|w_{1,2})\ldots P(w_N|w_{1,N-1}) \\ &= P(w_1)P(w_2|w_1)P(w_3|w_2)\ldots P(w_N|w_{N-1}) \\ &= P(w_1)\prod_{i=1}^{n} P(w_i|w_{i-1}) \end{aligned} \quad (5)$$

현재의 단어는 바로 앞의 단어에만 영향을 받기 때문에 각각의 단어에 대해 $(w_i|w_{i-1})$만 계산한 다음 모두 곱하기만 하면 된다. 이렇게 앞의 단어 하나만을 문맥 정보로 이용한다면 현재 음절의 앞에 있는 모든 단어를 정보로 활용할 때보다 입력 데이터의 차원이 낮아지므로 데이터 부족의 문제가 현저하게 줄어들게 되며 이에 따라 필요한 학습 데이터의 크기도 줄일 수 있다[13]. 또한 계산량 역시 현저하게 줄어들게 된다. 하지만 정보량 자체가 많이 줄어들게 되므로 모델의 성능 역시 낮아지게 된다[18]. 표 3은 시퀀스의 길이 N에 따른 엔그램을 지칭하는 용어와 엔그램의 예를 나타내고 있다.

표 3. 엔그램 시퀀스 길이에 따른 엔그램 용어와 예

N	용어	엔그램 예
1	Uni-gram	1, 2, 3, 4, 5, 6, 7, 8, 9, 10
2	Bi-gram	(1, 2), (2, 3), (3, 4), … (8, 9), (9, 10)
3	Tri-gram	(1, 2, 3), (2, 3, 4), … (7, 8 ,9), (8, 9, 10)
4	Quad-gram, Tetra-gram	(1, 2, 3, 4), (2, 3, 4, 5), … (7, 8, 9, 10)
5	Penta-gram	(1, 2, 3, 4, 5), (2, 3, 4, 5, 6), … (6, 7, 8, 9, 10)
6	Hex-gram	(1, 2, 3, 4, 5, 6), … (5, 6, 7, 8, 9, 10)
7	Sept-gram	(1, 2, 3, 4, 5, 6, 7), … (4, 5, 6, 7, 8, 9, 10)
8	Oct-gram, Octa-gram	(1, 2, 3, 4, 5, 6, 7, 8), … (3, 4, 5, 6, 7, 8, 9, 10)
9	Non-gram, Nona-gram	(1, 2, 3, 4, 5, 6, 7, 8, 9), … (2, 3, 4, 5, 6, 7, 8, 9, 10)
10	Dec-gram, Deca-gram	(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)

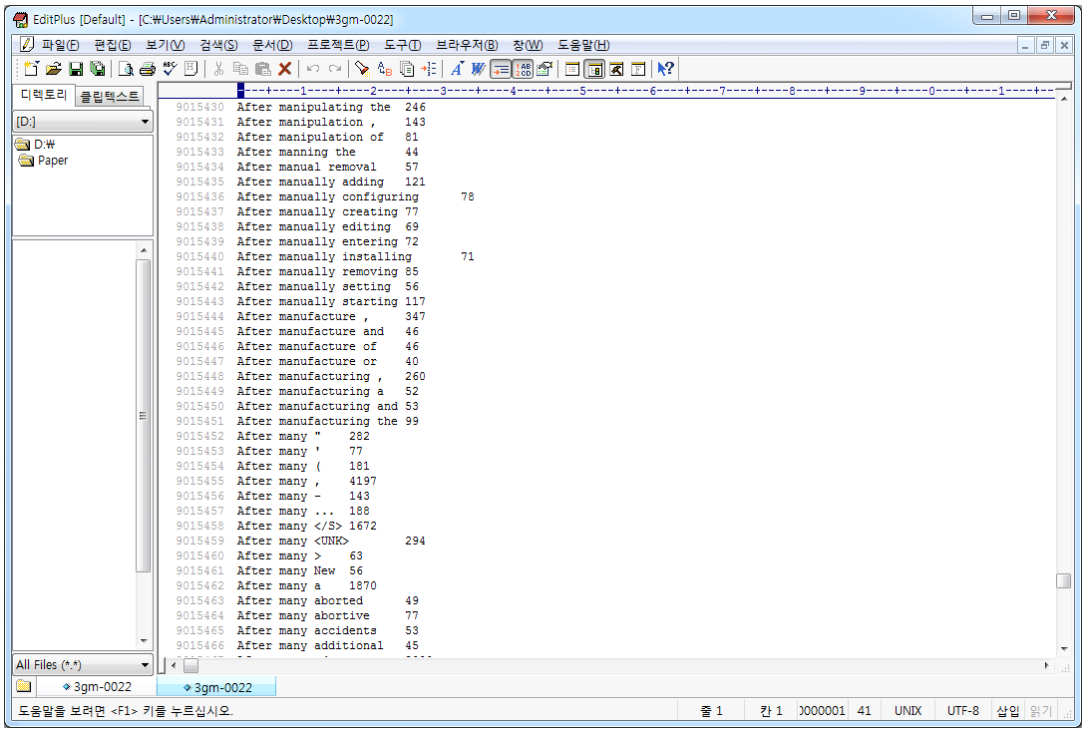


그림 4. 구글 엔그램의 트라이그램 학습 데이터

그림 4는 구글 엔그램의 트라이그램(Tri-gram) 학습 데이터 예를 나타내고 있다. 구글 엔그램은 단어 시퀀스 길이에 따라 학습 데이터를 구축하였으며, 학습 데이터는 시퀀스 길에 따른 단어열과 단어열이 구글 웹 문서상에서 사용된 빈도수를 제공하고 있다. 구글 엔그램을 이용한 미등록어의 단어 대체 방법은 미등록어의 주변 단어열 기반으로 구글 엔그램 학습 데이터에서 미등록어의 대체 후보 단어를 추출한 뒤, 추출된 대체 후보 단어 중 가장 높은 사용 빈도수를 가진 대체 후보 단어를 미등록어의 대체 단어로 선정해 미등록어를 대체한다. 이 방법은 형태소 분석 오류인 미등록어의 문제를 가장 정확하게 해소한다. 하지만 엔그램 방법은 방대한 양의 엔그램 학습 데이터가 존재해야 하며, 학습 데이터에 미등록어 주변의 단어열이 포함되어 있어야 미등록어의 대체 단어를 추출 할 수 있다는 문제점이 있다.

4. 동시출현단어 기반 미등록어 대체 기법

동시출현(Co-Occurrence)단어를 기반으로 하는 미등록어의 단어 대체 방법은 문서에서 동시에 출현하는 단어들을 단어 쌍으로 생성한 학습 데이터를 기반으로 미등록어의 인접 단어와 쌍이 되는 단어를 미등록어의 대체 단어로 선정해 미등록어를 대체한다. 동시출현단어 분석은 두 단어가 동일한 문헌이나 문단 또는 문장과 같은 특정 범위 내에서 많이 발견될수록 두 단어의 연관성이 높다는 이론을 기반하고 있다. 또한 이 분석 방법은 특정 주제 영역의 지식 구조를 관찰할 수 있는 또 다른 계량적 방법인 동시인용 분석이나 주제 분류보다 더 객관적으로 문서의 주제 유사도를 측정해내는 것으로 평가받았다[19].

동시출현단어 분석은 일반적으로 다음과 같은 5단계를 통해 수행된다[20].

① 특정 영역이나 주제에 대한 문헌을 수집한다.
② 여러 통제어휘와 비 통제 어휘 중 적절한 단어를 선정하여 데이터를 추출한다.
③ 단어 빈도수를 입력한 단어-문헌 행렬( )과 행과 열의 위치를 바꾸어 만든 전치 행렬(M)을 곱하여 동시출현단어 행렬을 작성한다.
④ 자카드계수, 포함지수, 근접성지수, 코사인계수, 피어슨 상관계수 등 다양한 유사도 계수를 사용하여 유사도 행렬을 작성한다.
⑤ 포함지도, 근접지도, 군집지도, 전략적 다이어그램, 자기상관지도(Auto-Correlation maps), 자기조작화지도(Self-Organizing map), 다차원척도법(Multidimensional Scaling), 사회연결망 네트워크(Social network analysis) 등으로 분석한 결과를 시각화한다.

동시출현단어의 연관도는 동시출현 빈도를 직접 사용할 수도 있고, 자카드계수나 포함지수 등의 지수를 이용하여 정규화 할 수도 있다[21]. 동시출현 단어 분석에서는 단어 간 연관도를 구하기 위해 주로 자카드계수(Jaccard Coefficient), 포함지수(Inclusion Index), 근접성지수(Proximity Index), 대등지수(Equivalence Index) 등을 사용한다[22]. 자카드계수 J_j, 포함지수 I_{ij}, 근접성지수 P_{ij}, 대등지수 E_{ij}의 수식은 다음과 같으며 수식에서 C_i는 문헌 내 단어 의 출현빈도, C_j는 문헌 내 단어 j의

출현빈도, $_j$는 문헌 내 단어 C_i와 단어 C_j의 동시출현빈도, N은 총 문헌수를 나타낸다.

$$J_{ij} = \frac{C_{ij}}{(C_i + C_j - C_{ij})} \quad (6)$$

자카드계수는 단어 사이의 상대적인 중복도(Degree of Overlap)를 측정하는 것으로 빈도가 높은 단어가 빈도가 낮은 단어와 함께 나타날 경우에 상대적으로 값이 낮아지기 때문에 중간빈도의 단어들의 중복도를 나타낼 때 더 적합하다[23].

$$I_{ij} = \frac{C_{ij}}{min(C_i, C_j)} \quad (7)$$

포함지수는 본질적으로는 단어 가 들어있는 문헌이 주어졌을 때 문헌 안에 단어 j가 나타날 조건 확률로서 출현빈도의 범위가 넓은 경우 단어 연관도를 구하는데 사용될 수 있다[24]. 포함지수는 단어 빈도가 높은 영역을 강조하는 효과를 가지게 된다. 포함지수를 응용한 지수인 'Half Side Inclusion Index'는 단어 i가 들어 있는 문헌이 주어졌을 때 문헌 안에 단어 j가 나타날 조건 확률과 문헌에서 단어 j가 발견되었을 경우에 단어 i가 나타날 조건 확률을 따로 구하여 단어의 동시출현빈도 행렬을 비대칭적으로 정규화한 것이다[25].

$$P_{ij} = \frac{C_{ij}/min(C_i, C_j)}{min(C_i, C_j)/N} = \frac{C_{ij}}{C_i C_j} \times N \quad (8)$$

근접성지수는 대칭적인 성격을 가지고 있으며 낮은 빈도를 가진 단어들 간의 연관도를 향상시켜 주기 때문에 새로운 분야나 소수 분야를 표현하는데 적합한 지수이다[26].

$$E_{ij} = \frac{C_{ij}}{C_i} \times \frac{C_{ij}}{C_j} = \frac{(C_{ij})^2}{(C_i \times C_j)} \quad (9)$$

대등지수는 일종의 코사인계수로 설튼 지수(Salton Index)라고도 한다. 대등지수는 다른 지수들에 비해 결과를 해석하기 쉽고 이용하기가 편리하다는 장점이 있다[27].

단어 간 연관도를 측정하기 위해 피어슨 상관계수(Pearson's Correlation Coefficient)를 사용하였다[28]. 피어슨 상관계수 은 두 단어의 유사성을 측정하기 위해 두 단어와 동시출현단어 행렬의 다른 단어들과의 동시출현빈도를 이용하여 구해지고 -1에서 +1까지의 값을 갖는다[24]. 피어슨 상관계수 r의 수식은 10과 같으며, 수식에서 x는 단어 x와 행렬의 i번째 단어와의 동시출현빈도를 y_i는 단어 y와 행렬의 i번째 단어와의 동시출현빈도를 나타낸다.

$$r = \frac{\sum_{i=1}^{n}(x_i-\bar{X})(y_i-\bar{Y})}{\sqrt{\sum_{i=1}^{n}(x_i-\bar{X})^2}\sqrt{\sum_{i=1}^{n}(y_i-\bar{Y})^2}} = \frac{n(\sum_{i=1}^{n} x_i y_i)-(\sum_{i=1}^{n} x_i)(\sum_{i=1}^{n} y_i)}{\sqrt{n(\sum_{i=1}^{n} x_i{}^2)-(\sum_{i=1}^{n} x_i)^2}\sqrt{n(\sum_{i=1}^{n} y_i{}^2)-(\sum_{i=1}^{n} y_i)^2}} \quad (10)$$

동시출현단어의 패턴 정보의 추출은 의존 문법을 이용한 부분 구문 분석 기법을 통해 문장의 의존 구조를 파악한 후 의존 관계에 놓인 단어들에 대해서만 조사한다[29, 30].

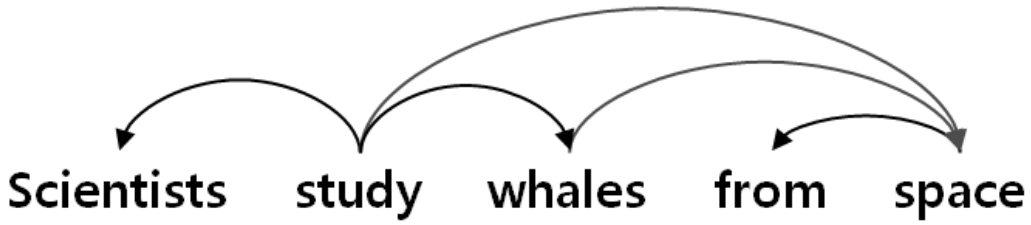


그림 5. 문장의 의존 구조

동시출현단어의 단어 쌍을 생성하기 위한 문장의 의존 구조는 그림5와 같다[31]. ‘Scientists’이라는 단어에 대한 동시발생 빈도는 ‘study’라는 단어에 대해서만 계산된다. 따라서 ‘study’하는 ‘Scientists’이기 때문에 ‘Scientists, study’의 단어 쌍을 생성할 수 있다. 미등록어의 대체 단어를 추출할 때에도 모든 단어 쌍에 대한 의미 거리를 측정하는 것이 아니라 의존 관계에 놓인 단어에 대해서만 계산하게 된다[31]. 하지만 하나의 문장에서 여러 의존 구조를 생성할 수 있기 때문에 모호함(Ambiguity)이 발생한다. 그림 5에서는 전치사 뒤에 오는 명사인 ‘space’ 때문에 모호성의 문제가 발생한다. ‘space’가 ‘study’를 수식하는 것인지 ‘whales’를 수식하는 것인지 분명하지 않기 때문에 단어 쌍을 만드는데 있어 모호성을 가지게 된다. 즉, 의존의 모호함은 단어 간의 수식관계에서 발생한다.

동시출현단어를 이용한 미등록어의 단어 대체 방법은 미등록어와 미등록어 주변 단어의 단어 쌍을 기반으로 미등록어 주변 단어가 쌍을 이루는 단어를 미등록어의 대체 후보 단어로 추출한다. 추출된 대체 후보 단어 중 동시발생 빈도가 높은 값을 가지는 대체 후보 단어를 미등록어의 대체 단어로 선정하여 미등록어를 대체 한다. 이 방법은 엔그램의 학습 데이터를 생성하는 것보다 효율적으로 동시출현 단어의 학습 데이터를 생성하지만 엔그램 방법의 문제점과 같이 미등록어 주변 단어의 단어쌍이 학습 데이터에 있어야 미등록어를 대체 할 수 있다는 문제점이 있다.

5. 워드 임베딩 기반 미등록어 대체 기법

워드 임베딩을 기반으로 하는 미등록어의 단어 대체 방법은 문서에 작성된 단어를 벡터로 변환하여 미등록어와 유사한 벡터 값을 갖는 대체 단어로 미등록어를 대체한다. 워드 임베딩이란 최근 많은 자연어처리 분야에서의 관심을 집중적으로 받고 있는 분야이다[32]. 워드 임베딩은 주어진 말뭉치에 있는 모든 단어에 대한 벡터 표현을 학습하는 기술이다[32]. 워드 임베딩 이전의 연구들은 단어를 원 핫 인코딩(One-Hot Encoding) 형태로 표현했다. 원 핫 인코딩은 개의 단어를 각각 N차원의 벡터로 표현하는 방식으로 단어를 벡터로 표현하기 위해 단어가 포함되는 자리엔 1을 부여하고 나머지 단어에는 0을 부여하는 단어의 벡터 표현 방식이다. 그림 6은 단어의 벡터 표현 방식을 나타낸다[33].

word	one-hot
King	[1, 0, 0]
Love	[0, 1, 0]
Queen	[0, 0, 1)]

그림 6. 단어의 벡터 표현 방식

원 핫 인코딩 이전의 연구들에서 좋은 성능을 내었고 지금까지도 사용되어지고 있다. 하지만 원 핫 인코딩 방법은 두 가지 문제점을 가지고 있다. 첫 번째는 각 단어가 사전의 사이즈와 같은 벡터를 가지고 있기 때문에 벡터의 사이즈가 크다는 것이다[32]. 두 번째는 단어 간의 유사성의 형태가 존재하지 않기 때문에 단어들이 서로 어떠한 차이를 가지고 있는지 단어들이 서로 유사한 뜻을 가지고 있는지에 대해서 이해 할 수 없다는 것이다[32]. 예를 들어 '제주도 숙소'라는 단어를 검색한다고 할 때 제대로 된 검색 시스템이라면 '제주도 숙소'라는 검색어에 대해서 '제주도 게스트 하우스', '제주도 관광지', '제주도 호텔'과 같은 유사 단어에 대한 결과도 함께 보여줄 수 있어야 한다. 하지만 단어 간 유사성을 계산할 수 없다면 '게스트 하우스', '관광지', '호텔'이라는 연관 검색어를 보여줄 수 없다.

워드 임베딩은 이러한 원 핫 인코딩의 단점을 보완하기 위해 단어 자체가 가지는 의미를 차원의 공간에서 벡터화하는 방법이다. 이러한 워드 임베딩은 여러 개의 단어들 사이의 유사도를 측정할 수 있고 벡터화 된 의미들을 가지고 벡터 연산을 하여 추론할 수 있다[34]. 그리고 벡터 공간에 실수로 맵핑을 하고 이를 저차원으로도 표현 할 수 있다[32]. 이러한 워드 임베딩 방법을 사용하는 기계학습 방법에는 CBOW(Continuous Bag-of-Words) 모델과 Skip-gram 모델이 있다.

Word2Vec은 2013년 구글에서 발표된 연구로 Tomas Mikolov라는 사람을 필두로 여러 연구자들이 모여서 만든 연속 워드 임베딩(Continuous Word Embedding) 학습 방법이다[35]. Word2Vec 방법은 기존 신경망 언어 모델(Neural Network Language Model) 방법에 비해 크게 달라진 것은 아니지만, 계산량을 엄청나게 줄여서 기존의 방법에 비해 몇 배 이상 빠른 학습을 가능케 하여 현재 가장 많은 이들이 사용하는 워드 임베딩 방법이 되었다[36]. Word2Vec에서는 학습을 시키기 위한 네트워크 모델을 두 가지 제시하였다. 한 가지는 CBOW 모델이고, 다른 하나는 Skip-gram 모델이다. 그림 7은 CBOW 모델의 학습 방식을 나타낸다.

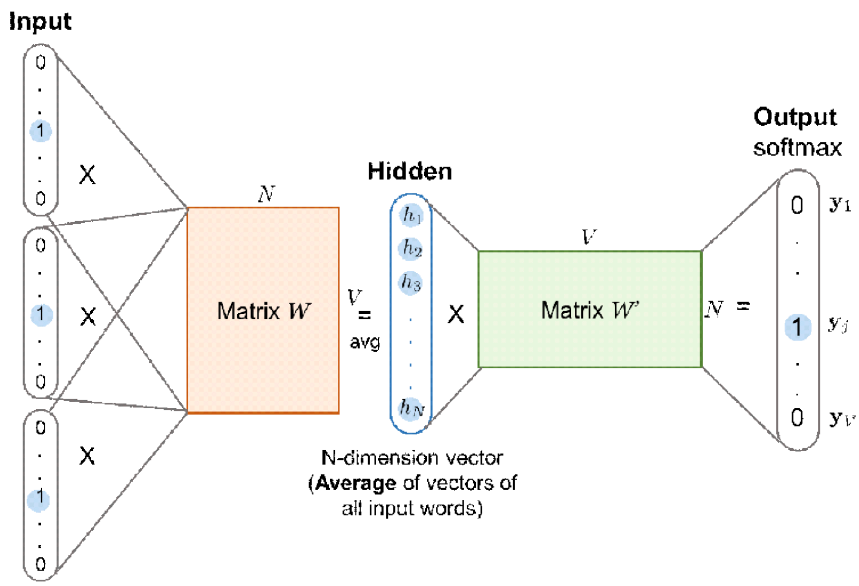


그림 7. CBOW 모델의 학습 방식

CBOW 모델의 경우 다음과 같은 방식을 채용하고 있다. '집 앞 편의점에서 아이스크림을 사 먹었는데, ___ 시려서 너무 먹기가 힘들었다.'라는 문장에서 사람들은 '___' 부분에 들어갈 단어가 정확하게 주어지지 않아도 앞뒤의 단어들을 통해 '이가' 라는 말이 들어갈 것을 추측할 수 있다[37]. CBOW 모델도 마찬가지의 방법을 사용한다. 주어진 단어에 대해 앞뒤로 2개 씩 총 C개의 단어를 입력으로 사용하여 주어진 단어를 맞추기 위한 네트워크를 만든다. 그림 7에서 V는 문장의 개수, N은 문장의 길이(단어의 개수), W는 $V\times N$ 크기의 가중치 행렬(Weight Matrix), $W^{'}$는 $N\times V$ 크기의 가중치 행렬을 나타낸다. CBOW 모델은 입력층(Input Layer), 은닉층(Hidden Layer), 출력층(Output Layer)으로 이루어져 있다. CBOW 모델의 은닉층은 입력층에서 중간층으로 가는 과정에서 가중치(Weight)를 곱해 주는 것이라기보다는 단순히 예상(Projection)하는 과정에 가까우므로 투영층(Projection Layer)이라고도 표현된다[38].

CBOW 모델의 입력에서는 신경망 언어 모델과 똑같이 단어를 원 핫 인코딩으로 넣어주고, 여러 개의 단어를 각각 예상 시킨 후 그 벡터들의 평균을 구해서 은닉층에 보낸다. 그 뒤 은닉층에 가중치 행렬 $W^{'}$를 곱해서 출력층으로 보내고 소프트맥스(Softmax) 계산을 한 후, 이 결과를 입력 단어의 원 핫 인코딩과 비교한다. CBOW 모델에서 하나의 단어를 처리하는 데에 드는 계산량은 $C\times N+ N\times V$과 같다. $C\times N$는 C개의 단어를 예상하는 계산량이며, $N\times V$은 은닉층에서 출력층으로 가는데 계산되는 계산량이다. 문장의 개수 V를 $\ln V$ 줄이는 방법을 사용하면 전체 계산량이 $C\times N+N\times \ln V$가 된다. 이 식을 통해 CBOW 모델이 신경망 언어 모델보다 계산이 빠른지를 알 수 있다. CBOW 모델에서 C는 10 내외의 크기로 설정한다. 전체 계산량은 문장의 길이 N과 $\log \ln V$의 크기의 곱에 비례하게 된다. 즉, C=10, N=500, V=1,000,000일 때, 약 10,000 ($500\times(10+\ln(1{,}000{,}000))$)의 계산량 밖에 들지 않는 것이다. 이는 신경망 언어 모델에 비해 엄청나게 줄어든 계산량이라는 것을 확인할 수 있다[39].

다른 하나의 모델인 Skip-gram 모델은 CBOW와는 반대 방향의 모델이라고 할 수 있다[40]. 현재 주어진 단어 하나를 가지고 주위에 등장하는 나머지 몇 가지의 단어들의 등장 여부를 유추하는 것이다. 이 때 예측하는 단어들의 경우 현재 단어

주위에서 샘플링 하는데, '가까이 위치해있는 단어일수록 현재 단어와 관련이 더 많은 단어일 것이다'라는 개념을 적용하기 위해 멀리 떨어져있는 단어일수록 낮은 확률로 택하는 방법을 사용한다[36, 37]. 나머지 구조는 CBOW와 방향만 반대일 뿐 굉장히 유사하다. 그림 8은 Skip-gram 모델의 학습 방식을 나타낸다.

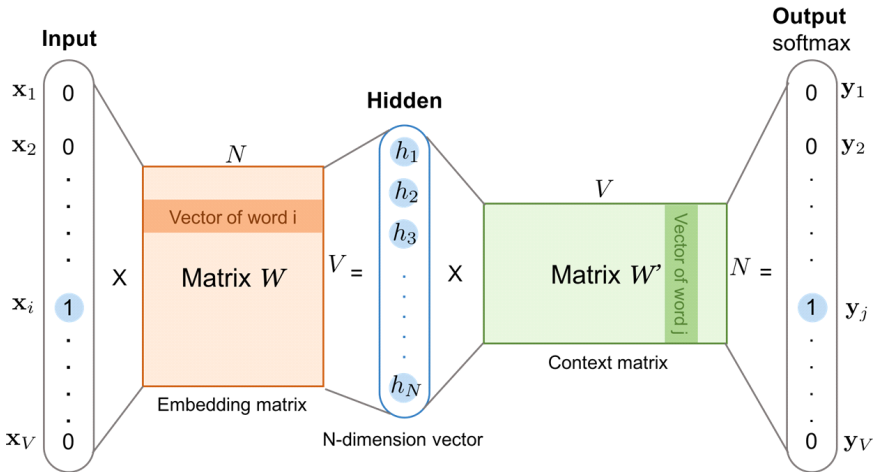


그림 8. Skip-gram 모델의 학습 방식

워드 임베딩 방법을 이용한 미등록어의 단어 대체 방법은 미등록어를 벡터 값으로 변환하여 벡터 공간상에서 미등록어와 유사한 값을 가지는 단어들을 대체 후보 단어로 선정 한 뒤, 미등록어와 대체 후보 단어의 코사인 유사도 값을 측정해 코사인 값이 1과 가깝게 측정된 대체 후보 단어를 선정해 미등록어를 대체한다. 워드 임베딩 방법은 대량의 학습 데이터 셋의 단어를 빠르게 벡터화 하여 미등록어를 대체한다. 하지만 미등록어와 대체 단어 사이의 의미적 관계성을 고려하지 않기 때문에 의미가 반대되는 대체 단어로 미등록어를 대체할 수 있다는 문제점이 있다.

Ⅲ. Word2VnCR 알고리즘을 위한 배경 이론의 제안

본 장에서는 형태소 오류인 미등록어의 대체 문제를 해소하기 위하여 앞선 2장에서 살펴본 기존 연구들이 지닌 한계점을 극복하고 보다 성능이 우수한 미등록어의 의미적 단어 대체 시스템을 구축하기 위한 배경 이론에 대해 기술한다. 본 이론은 주어진 문장을 대상으로 사람이 문맥 내 단어를 이해할 때 거치는 절차를 바탕으로 한다[10]. 기존 Word2Vec 알고리즘이 지능적인 문서 분석의 초석을 마련하였다면, 본 연구에서는 Word2Vec 알고리즘 보다 견고하고 지능이 높은 문서 분석 시스템을 위하여 Word2VnCR 알고리즘의 기반이 되는 두 이론을 제안한다.

1. 문맥을 통한 단어 간 연관성

1) 단어의 주변을 보면 그 단어를 안다

문서를 분석한다는 것은 문서에 작성된 단어를 수치화하는 것이다. 단어를 숫자로 바꾸어야만 알고리즘에 넣고 계산을 한 후 결과 값을 낼 수 있기 때문이다[41]. 단어를 숫자로 바꾸는 방법 중 한 가지는 단어를 벡터로 바꾸는 방법이 있다. 단어를 벡터로 바꾸는 가장 단순한 방법은 단어에 번호를 매기고 그 번호에 해당하는 요소만 1이고 나머지는 0을 갖는 벡터로 바꾸는 것이다. 그림 9는 원 핫 인코딩의 예를 나타낸다.

그림 9. 원 핫 인코딩의 예

예를 들어 총 5개의 단어가 있고 '강아지'라는 단어가 2번째 있다고 한다면 '강아지'는 2번째 요소만 1이고 나머지는 모두 0인 5차원의 벡터로 표현이 된다[41]. 이렇게 단어를 벡터로 바꾸는 방식을 원 핫 인코딩이라고 부른다. 개의 단어가 있을 때 각 단어는 한 개의 요소만 1인 N차원의 벡터로 표현된다. 원 핫 인코딩의 단점은 벡터 표현에 단어와 단어 간의 관계가 전혀 드러나지 않는다. '강아지'와 '멍멍이'라는 두 단어가 있을 때 이 두 단어는 의미가 비슷한데도 전혀 다른 벡터로 표현이 된다[41]. '강아지'와 '멍멍이'의 관계가 '강아지'와 '고양이' 간의 관계와 차이가 없는 것이다. 원 핫 인코딩은 어떤 단어가 유사한 의미를 갖고 어떤 단어가 반대의 의미를 갖는지 등 단어 간의 관계는 전혀 반영하지 못한다[41]. 이렇게 단어를 벡터로 바꾸는 모델을 워드 임베딩 모델(Word Embedding Model)이라고 부른다.

데이터는 대상의 속성을 표현해놓은 자료이다. 어떤 대상이든 대상의 속성들을 표현하고 그것을 바탕으로 모델을 만든다. 예를 들어 버섯을 조사해 놓은 데이터가 있다면 이것은 버섯이라는 대상을 색깔, 크기 같은 속성들로 표현한 것이다[41]. 대상을 어떤 속성으로 표현하는지는 모델의 성능에 매우 중요하다. 이렇게 대상의 속성을 표현하는 방식을 대표 특징(Feature Representation)이라고 한다. 자연어 처리의 경우 대상은 텍스트이고 이 텍스트의 속성을 표현해놓은 것이 데이터가 된다[41]. 예를 들어 해당 단어가 '강아지'라면 그 단어가 '강아지'라는 것 자체가 이 대상의 속성이 된다[41]. 또한 단어의 품사가 중요한 속성일 수도 있다. 앞 단어가 무엇인지 또는 문장에서 몇 번째 단어인지가 중요할 수도 있다[41]. 풀려는 문제에 따라서는 단어 자체가 긴지 짧은지가 중요할 수도 있다. 이런 언어적 정보(Linguistic Information)를 추출해서 표현하는 것이 언어의 대표 특징이다[41].

언어의 속성을 표현하는 방법으로 크게 희소 표현(Sparse Representation)과 밀집 표현(Dense Representation)이라는 두 가지 방식이 있다. 희소 표현은 앞서 언급했던 원 핫 인코딩을 뜻하고, 밀집 표현은 워드 임베딩 방법을 뜻한다. 원 핫 인코딩은 해당 속성이 가질 수 있는 모든 경우의 수를 각각의 독립적인 차원으로 표현한다[41]. 그림 10은 희소 표현의 예를 나타낸다.

그림 10. 희소 표현의 예

그림 10에서 '강아지'라는 속성을 표현해보자. 단어의 개수가 총 개라면 이 속성이 가질 수 있는 경우의 수는 총 N개이다. 원 핫 인코딩에서는 이 속성을 표현하기 위해 N차원의 벡터를 만든다. 그리고 '강아지'에 해당하는 요소만 1이고 나머지는 모두 0으로 둔다. 이런 식으로 단어가 가질 수 있는 N개의 모든 경우의 수를 표현할 수 있다. 마찬가지 방식으로 품사가 '명사'라는 속성을 표현하고 싶다면 품사의 개수만큼의 차원을 갖는 벡터를 만들고 '명사'에 해당하는 요소만 1로 두고 나머지는 모두 0으로 둔다. 다른 속성들도 모두 이런 방식으로 표현할 수 있다. 이렇게 원 핫 인코딩으로 만들어진 표현을 희소 표현이라고도 부른다. 벡터나 행렬이 희소(Sparse)하다는 것은 벡터나 행렬의 값 중 대부분이 0이고 몇몇 개만 값을 갖고 있다는 것을 뜻한다[41]. 원 핫 인코딩으로 만들어진 벡터는 0이 대부분이기 때문에 희소한 벡터가 되는 것이다. 희소 표현은 가장 단순하고 전통적으로 자주 쓰이던 표현 방식이다.

밀집 표현은 각각의 속성을 독립적인 차원으로 나타내지 않는다[41]. 대신, 사용자가 정한 개수의 차원으로 대상을 대응시켜서 표현한다[41]. 예컨대 해당 속성을 5차원으로 표현할 것이라고 정하면 그 속성을 5차원 벡터에 대응시키는 것이다[41]. 이 대응을 임베딩이라고 하며 임베딩 하는 방식은 머신 러닝을 통해 학습하게 된다[41]. 임베딩 된 벡터는 더 이상 희소하지 않다. 원 핫 인코딩처럼 대부분이 0인 벡터가 아니라 모든 차원이 값을 갖고 있는 벡터로 표현이 된다[41]. 그래서 희소의 반대말인 밀집을 써서 밀집 표현이라고 부른다. 밀집 표현은 또 다른 말로 분산 표현(Distributed Representation)이라고도 불린다. 분산(Distributed)라는 말이 붙는

이유는 하나의 정보가 여러 차원에 분산되어 표현되기 때문이다. 희소 표현에서는 각각의 차원이 각각의 독립적인 정보를 갖고 있지만 밀집 표현에서는 하나의 차원이 여러 속성의 정보를 들고 있다. 즉, 하나의 차원이 하나의 속성을 명시적으로 표현하는 것이 아니라 여러 차원들이 조합되어 나타내고자 하는 속성들을 표현하는 것이다[41]. 그림 11은 밀집 표현의 예를 나타낸다.

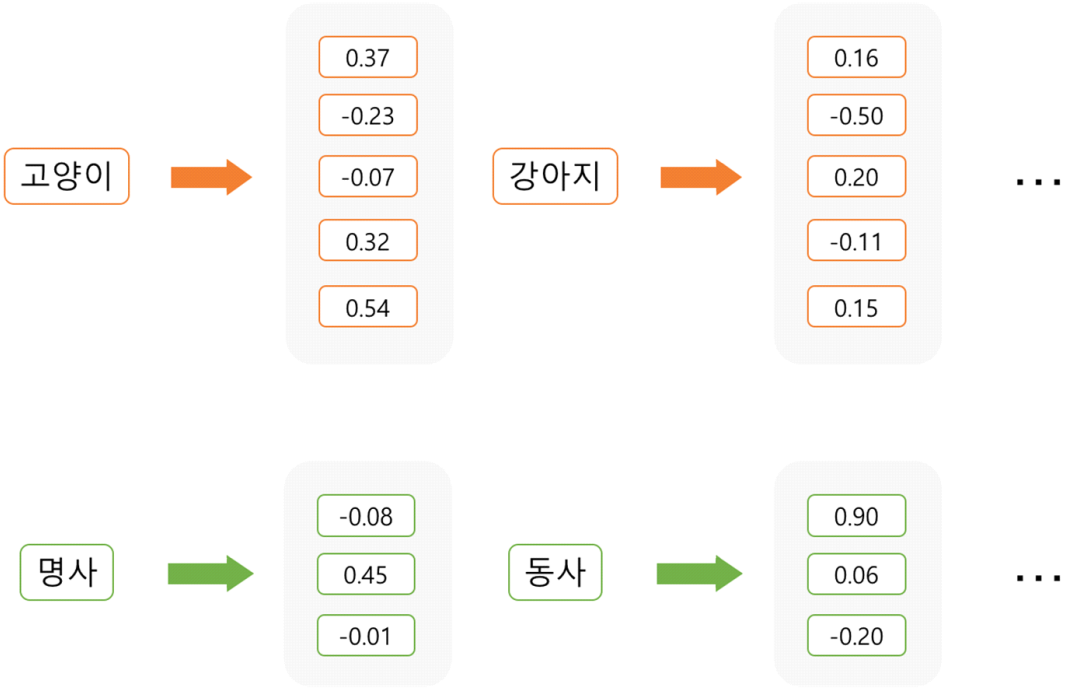


그림 11. 밀집 표현의 예

그림 11에서 '강아지'란 단어는 [0.16, -0.50, 0.20. -0.11, 0.15]라는 5차원 벡터로 표현된다[41]. 이 때 각각의 차원이 어떤 의미를 갖는지는 알 수 없다. 여러 속성의 정보가 표현되었기 때문이다. 다만 '강아지'를 표현하는 벡터가 '멍멍이'를 표현하는 벡터와 얼마나 비슷한지, 또는 '고양이'를 표현하는 벡터와는 얼마나 다른지는 벡터 간의 거리를 통해 알 수 있다[41]. 이러한 관계에서 단어 벡터의 의미가 드러난다. 단어의 벡터 값들은 머신 러닝을 통해 학습이 된다.

밀집 표현은 적은 차원으로 대상을 표현할 수 있다. 희소 표현으로 대상을 표현하면 보통 차원 수가 엄청나게 높아진다. 일상적인 텍스트에서 쓰이는 단어의 개수는 몇 천개에 이른다. 이 단어들을 희소 표현으로 표현하려면 몇 천 차원이 필요하다. 게다가 이렇게 만들어진 벡터들은 대부분의 값이 0을 갖는다. 입력 데이터의 차원이 높으면 차원의 저주(Curse of Dimensionality)라는 문제가 생긴다[41]. 입력 데이터에 0이 너무 많으면 데이터에서 정보를 뽑아내기 어려워진다. 따라서 희소 표현을 쓰면 모델의 학습이 어렵고 성능이 떨어지기 쉽다. 밀집 표현으로 단어를 표현할 때는 보통 20~200차원 정도를 사용한다[41]. 희소 표현에서 몇 천 차원이 필요했던 것에 비해 훨씬 적은 차원이다[41]. 게다가 0이 거의 없고 각각의 차원들이 모두 정보를 들고 있으므로 모델이 더 작동하기 쉬워지는 것이다[41]. 밀집 표현은 더 큰 일반화 능력(Generalization Power)을 갖고 있다. 예를 들어 '강아지' 라는 단어가 학습 데이터 셋에 자주 나왔고 '멍멍이'라는 단어는 별로 나오지 않았다고 가정해보자[41]. 희소 표현에는 '강아지'와 '멍멍이' 간의 관계가 전혀 표현되지 않는다[41]. 그 때문에 '강아지'에 대해 잘 알게 되더라도 '멍멍이'에 대해 더 잘 알게 되는 것은 아니다[41]. 또한 '강아지'가 '개'의 아기 상태라는 것을 알게 되었더라도 '멍멍이'가 '개'와 어떤 관계인지는 여전히 모르는 것이다[41]. 그러나 밀집 표현에서 '강아지'와 '멍멍이'가 서로 비슷한 벡터로 표현이 된다면 '강아지'에 대한 정보가 '멍멍이'에도 일반화될 수 있다[41]. 예컨대 '강아지'라는 단어를 입력으로 받고 '애완동물'이라는 출력을 하도록 모델이 학습이 된다면 '멍멍이'도 비슷한 입력이기 때문에 비슷한 출력이 나올 가능성이 높다[41]. 즉, '강아지'라는 단어에 대해 배운 지식을 '멍멍이'라는 단어에도 적용할 수 있는 것이다[41].

본 연구에서 제안하는 워드 임베딩 방법은 미등록어의 주변 단어로 미등록어의 대체 후보 단어를 추출한다. 주변 단어란 미등록어의 직전 몇 단어와 직후 몇 단어를 뜻한다. 미등록어의 앞뒤에 있는 단어들을 대체 후보 단어와 연관이 있다고 보는 것이다. 이 주변 단어의 범위를 윈도우(Window)라고 부른다. 윈도우를 이용해 학습 데이터를 생성하는 것을 윈도우 접근법이라 한다. 그림 12는 워드 임베딩 모델의 윈도우 접근법을 나타낸다.

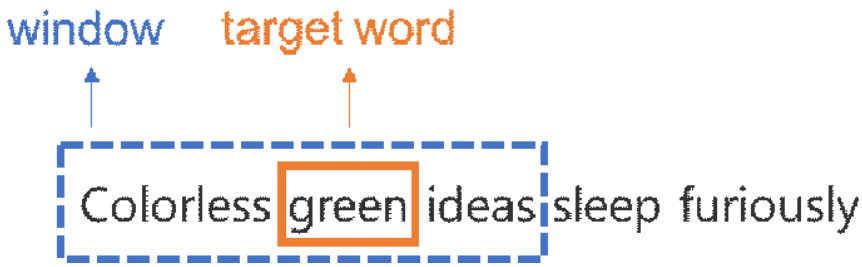


그림 12. 워드 임베딩 모델의 윈도우 접근법

워드 임베딩 모델의 윈도우는 단어의 범위를 지정해줄 수 있다. 이를 윈도우 사이즈(Window Size)라고 한다. 학습 데이터를 만들 때 워드 임베딩 모델은 슬라이딩 윈도우(Sliding Window)라는 방법을 쓴다. 'green'을 타겟 단어(Target Word)로 놓고 'Colorless'부터 'ideas'까지 한번 본 다음에 윈도우를 밀어서 다음에는 'ideas'를 타겟 단어 단어에 놓는다. 그 다음은 'sleep'을 타겟 단어에 놓고 본다. 이렇게 윈도우를 점차 옆으로 밀면서 타겟 단어를 계속 바꾸는 방식을 슬라이딩 윈도우라고 부른다. 윈도우 슬라이딩을 통해 만들어진 윈도우 하나하나가 학습 데이터가 된다. 그림 13은 워드 임베딩 모델의 슬라이딩 윈도우를 나타낸다.

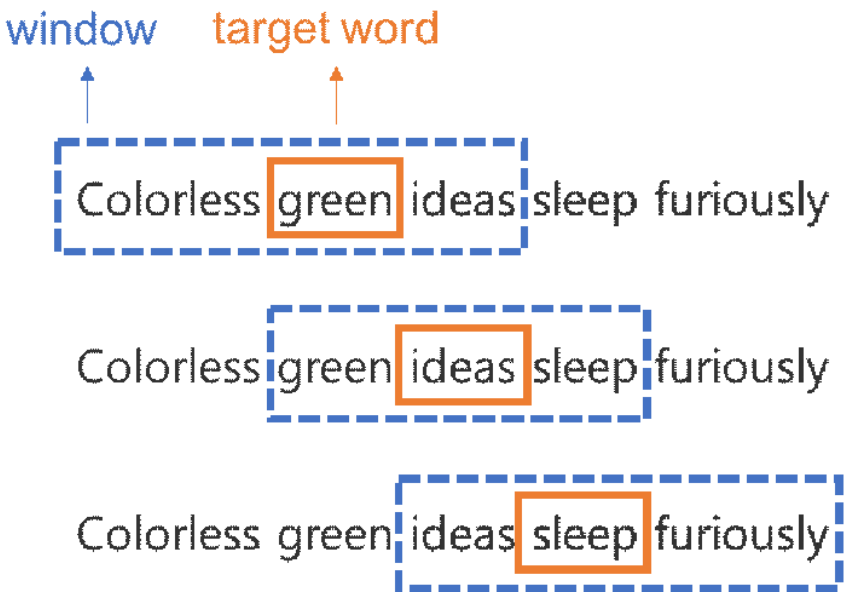


그림 13. 워드 임베딩 모델의 슬라이딩 윈도우

본 연구의 워드 임베딩 방법은 미등록어의 주변 단어로 미등록어의 대체 후보 단어를 추출한다. 즉, 미등록어를 입력으로 하면 미등록어의 주변 단어로 대체 후보 단어가 출력이 되는 것이다. 대체 후보 단어가 학습되는 방식은 일반적인 머신 러닝과 딥 러닝 모델이 학습되는 방식과 같다. 처음 대체 후보 단어는 랜덤으로 초기화된 상태(Random Initialization)로 시작한다. 랜덤으로 초기화 된 상태의 대체 후보 단어를 예측을 하고, 실제 값과 차이가 생기면 다른 만큼 대체 후보 단어를 변경한다. 이 과정을 학습 데이터 셋을 돌아가며 반복한다. 뉴럴 네트워크(Neural Network) 용어로는 이를 역전파(Backpropagation)라고 부르며 그 원리는 경사 하강법(Gradient Descent)와 같다. 즉, 비용 함수(Cost Function)가 최소화되는 쪽으로 대체 후보 단어를 업데이트해 가는 것이다. 따라서 본 연구에서는 미등록어를 대체할 대체 후보 단어를 추출하기 위해 문맥의 단어 간 연관성을 고려하는 워드 임베딩 방법을 이용한다.

2. 단어 간 의미적 유사도

1) 단어 간 의미적 유사도는 측정이 가능하다

사람은 미등록어가 포함된 문장을 이해할 때 문장 내 미등록어와 동시 출현한 단어의 의미적 관계를 통해 미등록어의 의미를 이해한다. 사람이 단어를 이해하는 방식을 컴퓨터와 같은 기계에 적용하기 위해 개발된 것이 워드넷(WordNet)이다. 워드넷은 1985년 프린스턴(Princeton) 대학 인지과학연구실을 주축으로 구축되어진 영어의 어휘 사전(Lexical Dictionary)이다. 그림 14는 프린스턴 대학에서 개발한 워드넷 2.1과 워드넷 3.1을 나타낸다.

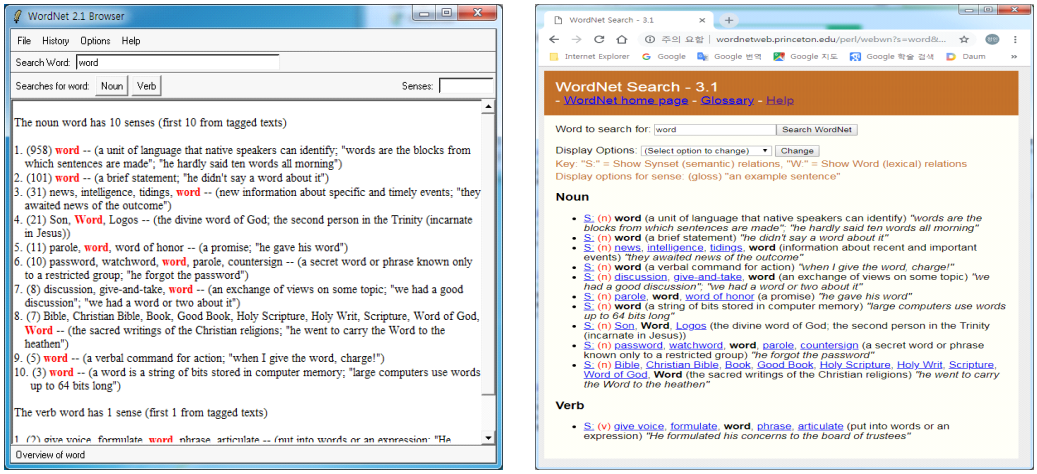


그림 14. 워드넷 2.1과 워드넷 3.1의 예

워드넷을 개발하게 된 동기는 어휘의 형태만을 고려하여 어휘 사전을 구성하였을 경우에 관련 어휘를 검색하고 어휘들의 연관성을 나타내기엔 부적합한 면이 많기 때문이다[42, 43]. 어휘의 형태만을 고려한 어휘 사전의 문제점을 해결하기 위해 동의(Synonymy), 반의(Antonymy), 상위(Hyperonymy), 하위(Hyponymy) 등과 같은 어휘의 연관 관계를 어휘 사전에 도입하게 되었다. 동의 관계는 서로 같은 단어의 의미를 가지고 있는 어휘 사이의 관계를 나타낸다. ‘tabby’, ‘cat’, ‘pussy’와 같은 단어는 동의 관계이다. 반의 관계는 서로 상반되는 단어의 의미를 가지고 있는 어휘

사이의 관계를 나타낸다. ‘rise’, ‘ascend’와 ‘fall’, ‘descend’는 반의 관계이다[42]. 상위 관계와 하위 관계는 단어들 사이의 의미 관계(Semantic Relation)로 단어의 의미 계층 (Semantice Category)을 중추적으로 나타낸다[42]. ‘tree’는 ‘maple’의 상위 관계가 이며, ‘maple’은 ‘tree’의 하위 관계이다. 부분(Meronym) 관계와 전체(Holonymy) 관계는 단어들 사이에서의 포함 관계를 나타낸다. ‘leaf’는 ‘tree’의 부분 관계이며, ‘tree’는 ‘leaf’의 전체 관계이다. 이러한 어휘들의 연관 관계가 워드넷에서는 잘 정의되어 있다[44].

표 4. WordNet 어휘의 개념 행렬

Word / Meanings		F_2	F_3	F_4	…	F_n
M_1	E_1 1	$E_1, 2$				
M_2		$E_2, 2$				
M_3			$E_3, 3$			
M_4						
…					…	
M_n						E_1, n

워드넷의 원리는 개념 행렬을 기초로 정의 되었다[45]. 워드넷은 어휘의 의미에 대한 카테고리 분류가 잘 정의되어 있으며, 어휘들의 계층 구조와 연관 관계가 잘 표현되어져 있다[46]. 표 4는 워드넷의 어휘 의미 관계를 나타내는 개념 행렬 모델이다. 표 4에서 F는 어휘의 형태, M은 의미, E는 F의 어휘 형태를 갖고 있는 M의 의미를 나타낸다. F_1과 F_2는 M_1의 의미에 있어서 동일한 의미를 가지고 있으므로 동의 관계를 나타낸다. F_2는 하나의 어휘 형태를 가지면서 어휘의 의미는 M_1과 M_2를 가지므로 다의(Polysemy)어 가 된다[47]. 어휘 집합 {F_1, F_2}는 M_1의 의미에 있어서 동의어 집합(Set of Synonyms Synset)이 된다[43, 48].

워드넷 단어의 개념들의 관계를 사용자가 열람하고 이를 활용할 수 있게 정보를 제공하고 있다. 워드넷에서 제공하는 정보 중 개념간의 계층관계는 두 단어의 개념간의 의미적 유사도가 얼마나 가까운가를 측정하는데 매우 중요한 척도로 사용될 수 있다. 그림 15는 ‘bike’와 ‘truck’의 워드넷 계층 구조를 나타낸다.

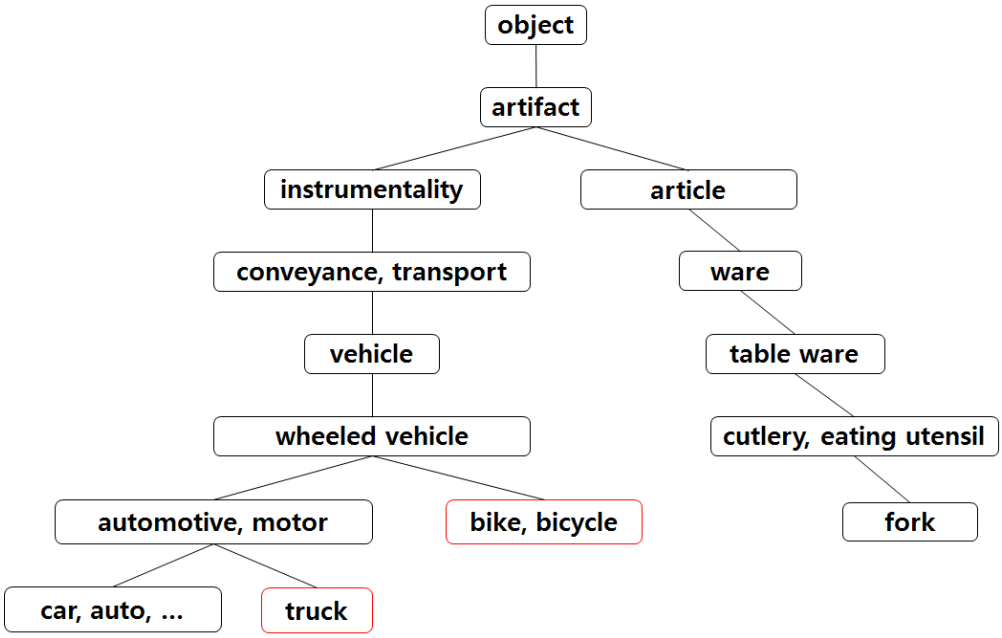


그림 15. 워드넷 계층 구조

Wu와 Plamer가 제안한 WUP(Wu & Palmer) 유사도는 두 단어 개념간의 의미 유사성을 워드넷 계층 구조의 에지를 기반으로 측정하며 수식은 11과 같다[49].

$$UP \quad \frac{2 \times depth(LCS(concept_1, concept_2))}{depth(concept\ ) + depth(concept_2)} \tag{11}$$

수식 11에서 깊이($depth$)는 단어가 워드넷 상에서 확장된 정도를 말하며 부모 노드인 LCS(Least Common Subsumer)는 두 단어가 최소공통분모로써 측정되는 깊이를 의미한다. 그림 9에서 'bike'의 깊이는 7을 'truck'의 깊이는 8을 부모노드의 'wheeled vehicle'의 깊이는 6을 갖게 된다. 이를 통해 'bike'와 'truck'의 WUP 유사도를 측정하면 0.8의 유사도 값을 갖게 된다. 이는 WUP 유사도의 최댓값이 1임을 감안할 때 높은 수치의 유사도를 갖는다고 판단할 수 있다. 이처럼 두 단어의 계층거리가 짧을수록 두 단어는 의미적으로 높은 유사성을 갖는다.

워드넷 유사도 측정 방법 중 하나인 WUP 유사도를 이용하여 철자교정을 한 연구가 수행되었다[50]. 이 연구는 트위터에 게시되는 트윗의 명사 단어 중 철자 오류가 있는 명사 단어를 확인 하고 철자 오류가 발생한 명사 단어의 철자교정을 WUP 유사도 측정 방법을 통해 교정하는 방법에 관해 제안한다. 그림 16은 의미적 유사도 측정 방법을 활용한 철자교정 시스템의 구성도를 나타낸다.

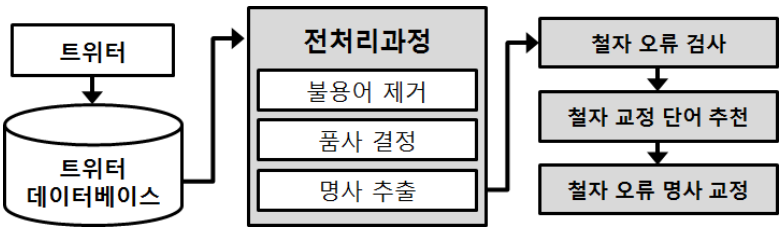


그림 16. 의미적 유사도 측정 방법을 활용한 철자교정 시스템 구성도

트위터의 트윗을 수집하기 위해 트위터 openAPI 사용하여 트윗을 트위터 데이터베이스에 저장한다. 저장된 트윗은 전처리 과정을 수행하는데, 전처리 과정에선 트윗의 불용어 삭제, 품사 결정, 명사 추출을 순서대로 수행한다. 전처리 과정을 통해 추출된 트윗의 명사는 파이썬 인챈트(Python Enchant)를 이용해 철자 오류가 있는지 확인하고 철자 오류가 발생한 경우 철자 맞는 명사 후보 단어들을 추천 받는다. 그림 17은 파이썬 인챈트의 예를 나타낸다.

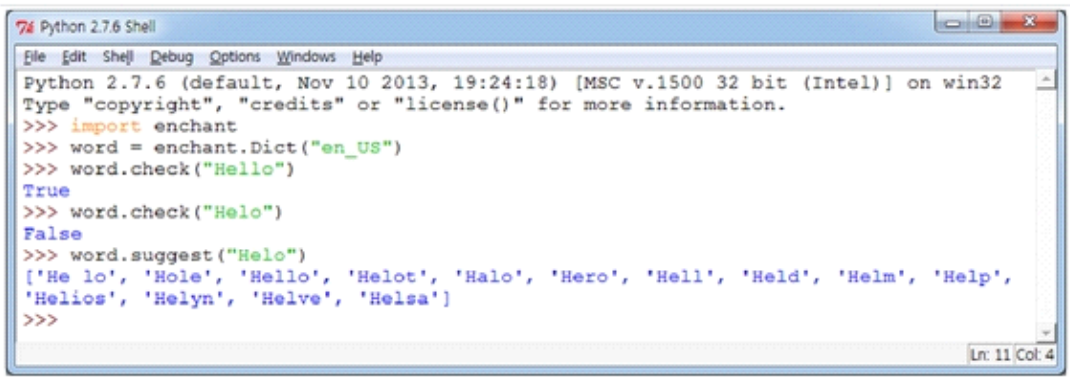


그림 17. 파이썬 인챈트의 예

철자 오류 단어의 명사 후보 단어들은 트윗에서 철자에 맞게 쓰인 주변 명사 단어들과 WUP 유사도 측정 방법을 통해 WUP 유사도 값을 측정하게 된다. 측정된 WUP 유사도 값 중 가장 높은 유사도 값을 가지는 명사 후보 단어를 철자 오류가 발생한 명사 단어와 대체하여 트윗의 철자교정을 한다. 이러한 워드넷 유사도 측정 방법을 통해 철자교정을 한 경우 58.2%의 재현율로 74.2%에 달하는 정확도 성능을 보였다. 따라서 본 연구에서는 미등록어를 대체할 대체 단어를 선정하기 위해 워드 임베딩 방법을 통해 추출된 미등록어의 대체 후보 단어와 미등록어가 작성된 문장의 명사 단어 간 유사도를 WUP 유사도를 이용해 측정하고 측정된 유사도 값 중 가장 높은 값을 가지는 대체 후보 단어를 대체 단어로 선정하여 미등록어를 대체한다. 이 두 가지 이론은 본 연구에서 제안하고자하는 Word2VnCR 알고리즘의 핵심이 되는 내용으로 제 4장에서 이를 적용한 알고리즘을 상세하게 기술한다.

Ⅳ. Word2VnCR 알고리즘 기반 미등록어의 대체 방법

본 장에서는 앞선 3장에서 제안하고 입증한 두 가지 배경 이론을 적용하여 형태소 분석 오류인 미등록어를 의미적으로 유사한 단어로 대체 하는 Word2VnCR 알고리즘에 대하여 상세하게 기술한다.

1. 전처리(Preprocessing)

실험에 사용된 데이터는 싱가포르 국립 대학교에서 수집한 NUS sms 말뭉치(Corpus)를 사용하였다. 그림 18은 Word2VnCR 알고리즘의 전처리 시스템 구성도를 나타낸다.

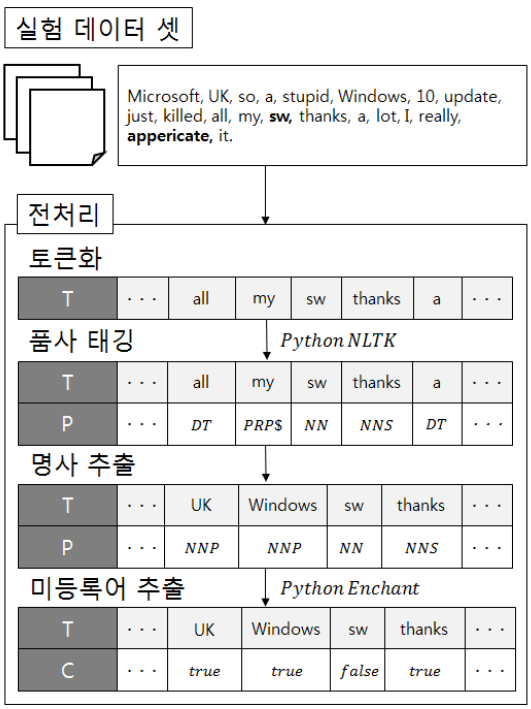


그림 18. Word2VnCR 알고리즘의 전처리 시스템 구성도

NUS sms 말뭉치의 텍스트는 자연어로 작성된 문장이기 때문에 컴퓨터가 문장을 이해하고 분석할 수 있도록 전처리 과정을 수행해야 한다. 전처리 과정은 토큰화, 품사 태깅(Part of Speech Tagging), 명사 추출, 미등록어 추출 순으로 진행된다.

토큰화는 하나의 단어를 기준으로 문장을 세분화시키는 작업으로 영어의 경우 공백을 기준으로 토근화를 한다. 표 5는 NUS sms 텍스트를 공백을 기준으로 토큰화한 예를 나타낸다.

표 5. NUS sms 텍스트의 토큰화

구분	예시
NUS sms 텍스트	Microsoft UK so a stupid Windows 10 update just killed all my sw thanks a lot I really appericate it
토큰화	Microsoft, UK, so, a, stupid, Windows, 10, update, just, killed, all, my, sw, thanks, a, lot, I, really, appericate, it.

토큰화된 텍스트는 파이썬(Python)으로 제작된 파이썬 NLTK(Natural Language Toolkit)를 이용해 품사 태깅한다. 파이썬 NLTK는 자연어 처리를 위하여 파이썬 프로그램 언어로 제작된 자연어처리 라이브러리로써 품사 태깅 및 엔그램, 워드넷 계층구조, 워드넷 기반 유사도 등의 기능을 제공한다. 본 연구의 목적은 단어의 정확한 품사 태깅이 아니라 보다 정확하게 미등록어를 의미적으로 유사한 단어와 대체를 시키는 문제를 해소하는 것이기 때문에 본 연구에서는 단어의 품사 태깅을 위하여 파이썬 NLTK를 이용한다. 표 6은 파이썬 NLTK를 이용해 토근화된 NUS sms 텍스트를 품사 태깅한 결과이다.

표 6. 토근화된 NUS sms 텍스트의 품사 태깅

구분	예시
토큰화	Microsoft, UK, so, a, stupid, Windows, 10, update, just, killed, all, my, sw, thanks, a, lot, I, really, appericate, it.
품사 태깅	Microsoft/NNP UK/NNP so/RB a/DT stupid/JJ Windows/NNP 10/CD update/NN just/RB killed/VBD all/DT my/PRP$ sw/NN thanks/NNS a/DT lot/NN I/PRP really/RB appericate/NN it/PRP

파이썬 NLTK를 이용하여 단어의 형태를 분석할 경우 품사 태깅되는 단어 형태별 기호와 그 설명은 표 7과 같다.

표 7. 파이썬 NLTK의 단어 형태별 표시 기호와 설명[10]

기호	설명	기호	설명	기호	설명
CC	등위접속사	NN	단수 명사	RBR	비교 부사
CD	기수	NNS	복수 명사	RBS	최상급 부사
DT	한정사	NNP	고유명사	SYM	기호
EX	유도부사	NNPS	복수 고유명사	TO	to부정사
FW	Foreign word	PDT	전치 한정사	VB	현재형 동사
IN	전치사	POS	소유격	VBD	과거형 동사
JJ	형용사	PRP	인칭대명사	VBG	동명사
JJR	비교 형용사	PRP$	소유대명사	VBN	현재완료 동사
JJS	최상급 형용사	RB	부사	VBP	비3인칭 단수현재 동사

명사 추출 과정에서는 품사 태깅된 NUS sms 텍스트에서 명사(NN), 복수 명사(NNS), 고유명사(NNP), 복수 고유명사(NNPS)로 태깅된 단어만을 추출한다. 명사 유형의 단어만을 추출하는 것은 파이썬 NLTK의 경우 단어의 형태 중 명사 형태의 단어를 가장 잘 태깅하기 때문이다. 기존 파이썬 NLTK의 성능을 실험한 결과로써 SemCor 말뭉치의 명사와 동사의 품사 태깅 결과로 명사의 경우 약 85.1%, 동사는 약 61.3%의 성능을 보였다[10]. 이에 따라 본 연구에서도 미등록어 추출의 성능을 높이기 위해 명사 유형의 단어만을 추출한다. 표 8은 품사 태깅된 NUS sms 텍스트에서 명사 유형의 단어를 추출한 결과 값을 나타낸다.

표 8. NUS sms 텍스트의 명사 추출

구분	예시
품사 태깅	Microsoft/NNP UK/NNP so/RB a/DT stupid/JJ Windows/NNP 10/CD update/NN just/RB killed/VBD all/DT my/PRP$ sw/NN thanks/NNS a/DT lot/NN I/PRP really/RB appericate/NN it/PRP
명사 추출	Microsoft/NNP UK/NNP Windows/NNP update/NN sw/NN thanks/NNS lot/NN appericate/NN

미등록어 추출 과정에서는 NUS sms 텍스트에서 추출된 명사 유형의 단어를 파이썬 인챈트(Enchant)를 이용하여 미등록어의 여부를 판단한다. 표 9는 NUS sms 텍스트의 미등록어 추출 예를 나타낸다.

표 9. NUS sms 텍스트의 미등록어 추출

구분	예시
명사 추출	Microsoft/NNP UK/NNP Windows/NNP update/NN sw/NN thanks/NNS lot/NN appericate/NN
미등록어 추출	Microsoft/true UK/true Windows/true update/true **sw/false** thanks/true lot/true **appericate/false**

전처리 과정을 통해 NUS sms 텍스트에서 추출된 미등록어 'sw'와 'appericate'는 텍스트에 작성된 순서대로 Word2VnCR 알고리즘을 이용하여 의미적으로 유사한 단어로 대체된다.

2. 미등록어의 대체를 위한 Word2VnCR 알고리즘의 적용

본 절은 본 연구에서 제안하는 Word2VnCR 알고리즘이 실제적으로 수행되는 절차를 다루는 핵심 부분으로써 앞서 기술한 전처리 과정에 이어 미등록어의 의미적 대체 방법의 수행 절차를 설명한다. 본 연구의 Word2VnCR 알고리즘은 워드 임베딩 방법과 워드넷 유사도 측정 방법을 기반으로 3장에서 제시한 두 가지 이론을 적용한 알고리즘으로 그림 19와 같다.

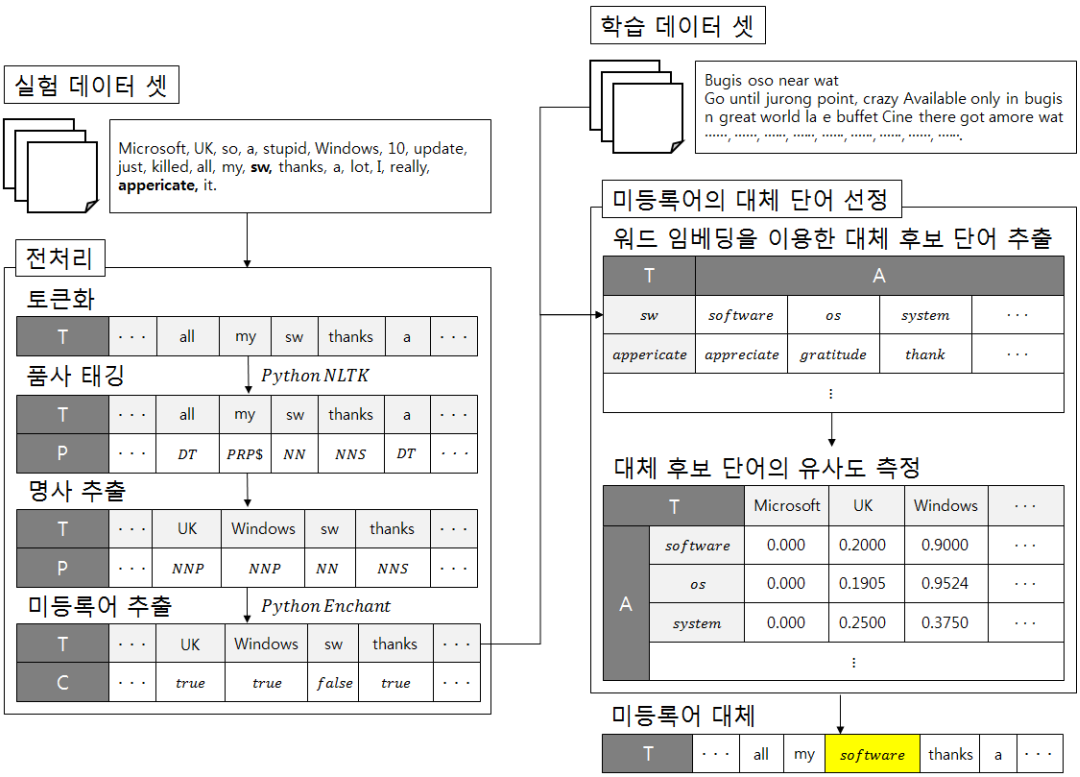


그림 19. Word2VnCR 알고리즘의 전체 시스템 구성도

다음 그림 19는 Word2VnCR 알고리즘을 이용하여 미등록어가 의미적으로 유사한 단어로 대체되어 형태소 분석 오류를 해소하는 절차를 각 단계별로 상세하게 보이고 있다. 표 5의 예시 문장 'Microsoft UK so a stupid Windows 10 update just killed all my sw thanks a lot I really appericate it'을 Word2VnCR 알고리즘의 입력

받았을 때 그림 19의 좌측에 보이는 것처럼 입력된 문장은 토큰화, 품사 태깅, 명사 추출, 미등록어 추출의 과정을 거쳐 명사 유형만을 가지는 미등록어를 추출하게 된다. 예시 문장 'Microsoft UK so a stupid Windows 10 update just killed all my sw thanks a lot I really appericate it'에서 추출된 명사 유형의 미등록어는 'sw'와 'appericate' 이다. 추출된 미등록어는 대체 단어를 찾기 위해 그림 19의 우측의 미등록어 대체 단어 선정 과정을 수행한다. Word2VnCR 알고리즘에서는 미등록어의 대체 후보 단어를 찾기 위해 워드 임베딩 방법을 사용하는 과정과 추출된 대체 후보 단어와 미등록어 인접 단어의 의미적 유사도를 측정하는 과정을 수행한다.

워드 임베딩을 이용한 미등록어의 대체 후보 단어 추출 과정에서는 미등록어 'sw'와 'appericate'의 대체 후보 단어를 생성하기 위해서는 그림 7의 워드 임베딩 방법을 수행한다. 워드 임베딩 방법의 입력 값 는 미등록어 'sw'을 원 핫 인코딩한 행렬 [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0]과 미등록어 'appericate'을 원 핫 인코딩한 행렬 [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0]이 된다. 입력 값 X는 가중치 행렬 W를 곱해 은닉층의 벡터를 생성한다. W는 학습 데이터 셋을 행렬로 변환한 것으로 V는 학습 데이터 셋의 개수, N는 학습 데이터 셋의 텍스트의 길이를 의미한다. 즉, 행렬 W의 13번째 행과 19번째 행이 은닉층의 h벡터가 되는 것이다. 입력 값 X와 가중치 행렬 W의 곱을 통해 생성된 h벡터는 가중치 행렬 W의 행과 열의 크기가 바뀐 가중치 행렬 W'를 곱해준 뒤 소프트맥스 계산을 거쳐 출력층 y을 생성한다. 이를 통해 미등록어 'sw'와 'appericate'가 대체될 대체 후보 단어들을 학습 데이터 셋에서 추출한다. 워드 임베딩 방법에 단어의 확률은 소프트맥스 수식 12로 정의 될 수 있다.

$$y = \frac{\exp(a_k)}{\sum_{i=1}^{n} \exp(a_i)} \tag{12}$$

수식 12의 $\exp(x)$는 지수함수(Exponential Function) 이며, n은 출력층의 단어의 수, y_k는 그 중 k번째 출력을 뜻한다. 위 수식 12와 같이 소프트 맥스 함수의 분자는 입력 단어 ak의 지수함수, 분모는 모든 입력 단어의 지수함수의 합으로 구성된다.

그림 20은 NUS sms 학습 데이터 셋을 Word2VnCR 알고리즘의 워드 임베딩 방법을 적용했을 때 학습 데이터를 생성하는 과정을 나타낸다. 미등록어 'sw'가 입력되면 'sw'의 인접 단어를 기반으로 워드 임베딩 방법을 하면 학습 데이터는 {my, thanks → software}, {my, thanks → OS}, {my, thanks → system} 등이 생성이 된다. 이를 통해 미등록어 'sw'를 입력 값으로 넣게 되면 미등록어 'sw'의 대체 후보 단어는 'software', 'os', 'system', 'computer', 'pc'가 출력된다.

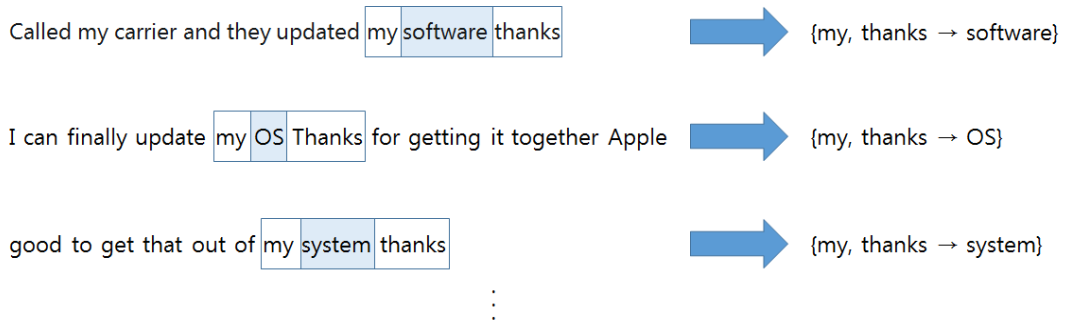


그림 20. Word2VnCR 알고리즘의 워드 임베딩 학습 데이터의 예

본 연구에서는 NUS sms 말뭉치를 학습 데이터로 사용하여 Word2VnCR 알고리즘의 워드 임베딩 방법을 적용한 경우 미등록어 'sw'의 대체 후보 단어는 'software', 'os', 'system', 'computer', 'pc'가 'appericate'의 대체 후보 단어는 'appreciate', 'gratitude', 'thank'가 추출되었다. 워드 임베딩 방법을 이용해 추출된 미등록어 'sw'와 'appericate'의 대체 후보 단어는 미등록어가 출현한 문장의 명사 단어들과 의미적 유사도를 측정한다. 의미적 유사도 측정하기 위해 사용되는 방법은 개념간의 관계를 보다 의미적으로 분석할 수 있는 워드넷의 의미적 유사도 측정 방법인 WUP 유사도 측정 방법을 이용한다. 표 10은 미등록어의 대체 후보 단어와 문장 내 명사 단어의 WUP 유사도 값을 계산한 결과를 나타나고 있다.

표 10. 미등록어의 대체 후보 단어와 미등록어 인접 단어 간 의미적 유사도 측정 결과

미등록어	대체 후보 단어의 워드넷 개념	문장 내 명사 단어	문장 내 명사 단어의 워드넷 개념	의미적 유사도
sw	software#n#1	Microsoft		
		UK	UK#n#1	0.2000
		Windows	Windows#n#1	0.9000
		update	update#n#1	0.4706
		thanks	thanks#n#2	0.3333
		lot		
		appericate		
		의미적 유사도 합		**1.9039**
	os#n#3	Microsoft		
		UK	UK#n#1	0.1905
		Windows	Windows#n#1	0.9524
		update	update#n#1	0.4444
		thanks	thanks#n#2	0.3158
		lot		
		appericate		
		의미적 유사도 합		1.9031
	system#n#2	Microsoft		
		UK	UK#n#1	0.2500
		Windows	Windows#n#1	0.3750
		update	update#n#1	0.4615
		thanks	thanks#n#2	0.4286
		lot		
		appericate		
		의미적 유사도 합		1.5151
	computer#n#1	Microsoft		
		UK	UK#n#1	0.3810
		Windows	Windows#n#1	0.1905
		update	update#n#1	0.2222
		thanks	thanks#n#2	0.2105
		lot		
		appericate		
		의미적 유사도 합		1.0042
	pc#n#1	Microsoft		
		UK	UK#n#1	0.3478
		Windows	Windows#n#1	0.1739
		update	update#n#1	0.2000
		thanks	thanks#n#2	0.1905
		lot		
		appericate		
		의미적 유사도 합		0.9122

미등록어	대체 후보 단어의 워드넷 개념	문장 내 명사 단어	문장 내 명사 단어의 워드넷 개념	의미적 유사도
appericate	appreciate#v#1	Microsoft		
		UK	UK#n#1	
		Windows	Windows#n#1	
		update	update#n#1	
		sw		
		thanks	thanks#n#2	
		lot		
		의미적 유사도 합		
	gratitude#n#1	Microsoft		
		UK	UK#n#1	0.2222
		Windows	Windows#n#1	0.3333
		update	update#n#1	0.4000
		sw		
		thanks	thanks#n#2	0.3750
		lot		
		의미적 유사도 합		**1.3305**
	thank#v#1	Microsoft		
		UK	UK#n#1	
		Windows	Windows#n#1	
		update	update#n#1	
		sw		
		thanks	thanks#n#2	
		lot		
		의미적 유사도 합		

Word2VnCR 알고리즘의 의미적 유사도 측정 결과에 따라 미등록어 'sw'는 대체 후보 단어 'software', 'os', 'system', 'computer', 'pc' 중 'software'로 미등록어 'appericate'는 대체 후보 단어 'appreciate', 'gratitude', 'thank' 중 'gratitude'로 선정되었다. 이에 따라 미등록어가 작성된 예문 'Microsoft UK so a stupid Windows 10 update just killed all my sw thanks a lot I really appericate it'는 미등록어 'sw'를 'software'로 'appericate'는 'gratitude'로 대체하여 'Microsoft UK so a stupid Windows 10 update just killed all my software thanks a lot I really gratitude it'로 수정한다.

V. 실험 및 결과

본 장에서는 앞선 장들에서 제안한 Word2VnCR 알고리즘을 이용하여 형태소 분석 오류인 미등록어를 의미적으로 유사한 단어로 대체 하는 방법에 대한 성능 평가를 수행한다. 본 연구에서는 NUS sms 말뭉치를 이용해 실험을 수행함으로써 Word2VnCR 알고리즘의 우수성을 입증한다.

1. 실험 데이터

NUS sms 말뭉치는 싱가포르 국립 대학교에서 수집한 영어 단문 메시지로 메시지 작성자 아이디, 메시지 텍스트, 메시지를 작성한 폰 모델, 메시지 작성자의 나이, 성별, 국가, 지역 등의 정보를 xml형태로 표현하고 있다. 그림 21은 NUS sms 말뭉치의 예를 나타낸다.

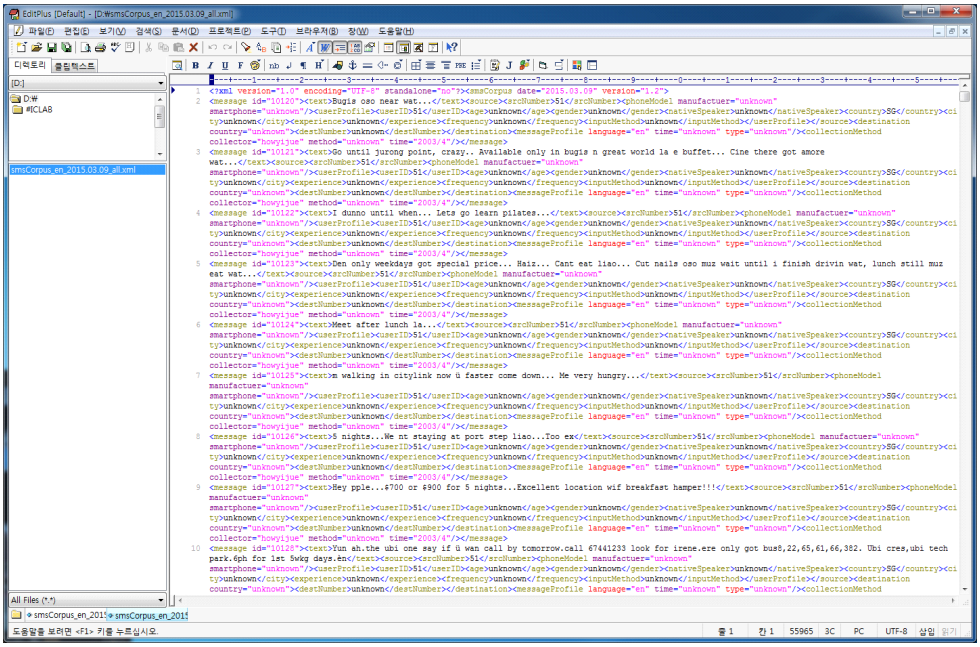


그림 21. NUS sms 말뭉치

xml형태로 표현된 NUS sms 말뭉치를 바로 실험에 사용하지 않고 보다 신뢰성 있는 실험을 수행하기 위하여 학습 데이터 셋과 실험 데이터 셋으로 분리하였다. 학습 데이터 셋은 NUS sms 말뭉치에서 작성된 메시지 텍스트를 열개씩 나누어 텍스트 파일에 저장하여 생성하였다. 그림 22는 NUS sms 말뭉치에서 추출한 학습 데이터 셋의 예를 나타낸다.

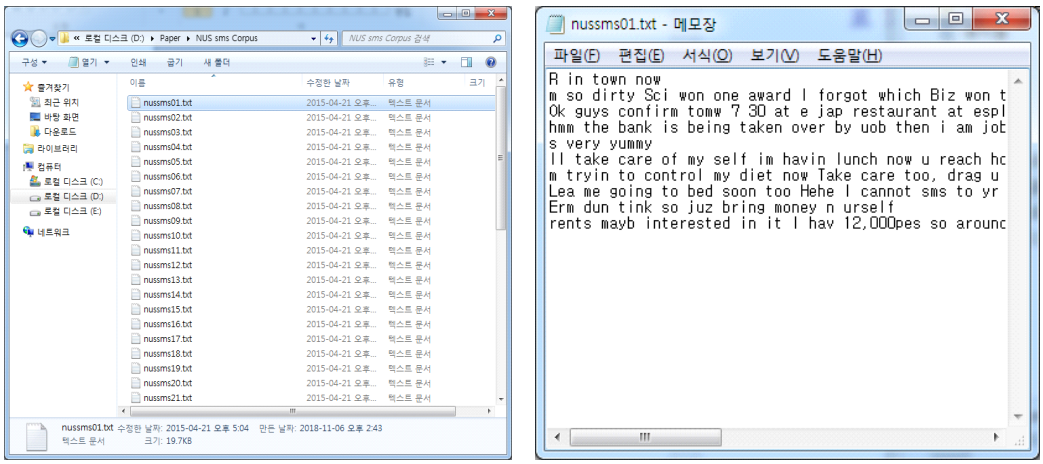


그림 22. NUS sms 말뭉치에서 추출한 학습 데이터 셋

실험 데이터 셋은 NUS sms 말뭉치에서 추출한 메시지 텍스트를 저장하였으며, 실험에 입력되는 텍스트는 실험 데이터 셋에 있는 문장 하나로 제약하였다. 그림 23은 NUS sms 말뭉치에서 추출한 실험 데이터 셋의 예로 NUS sms 말뭉치의 텍스트로 구성되어 있다.

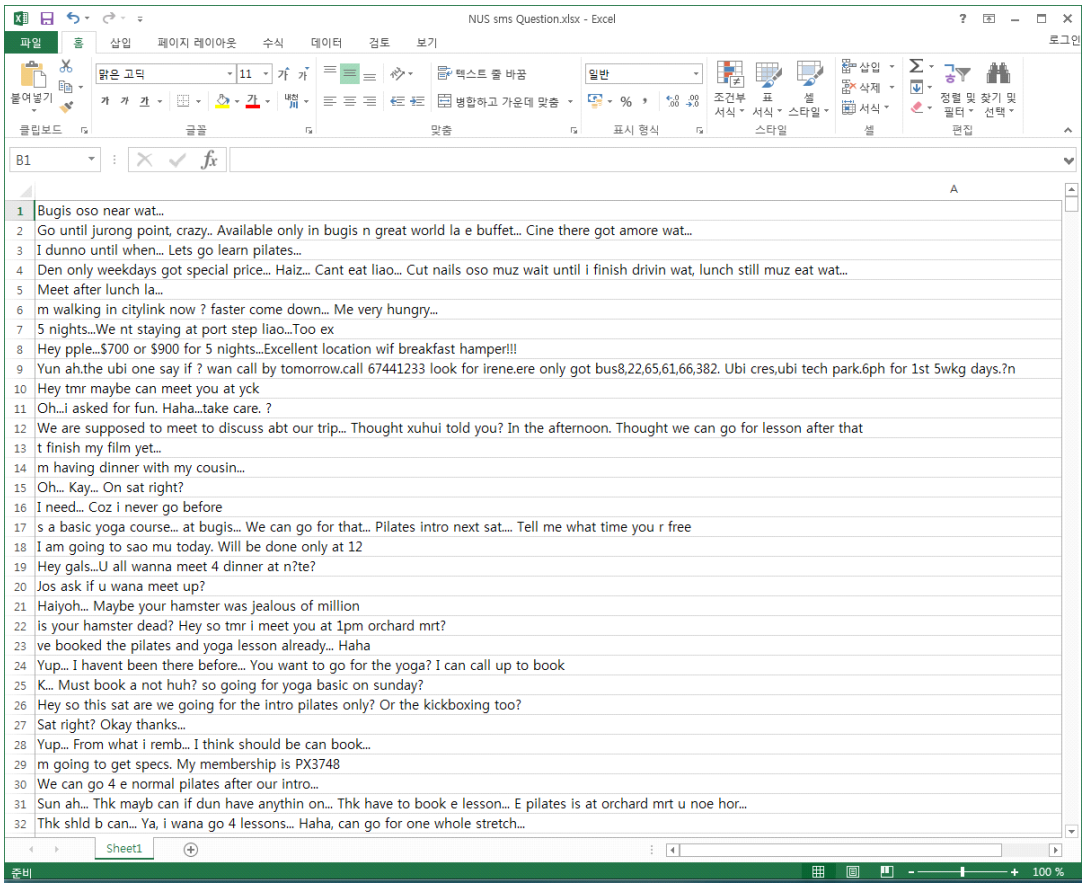
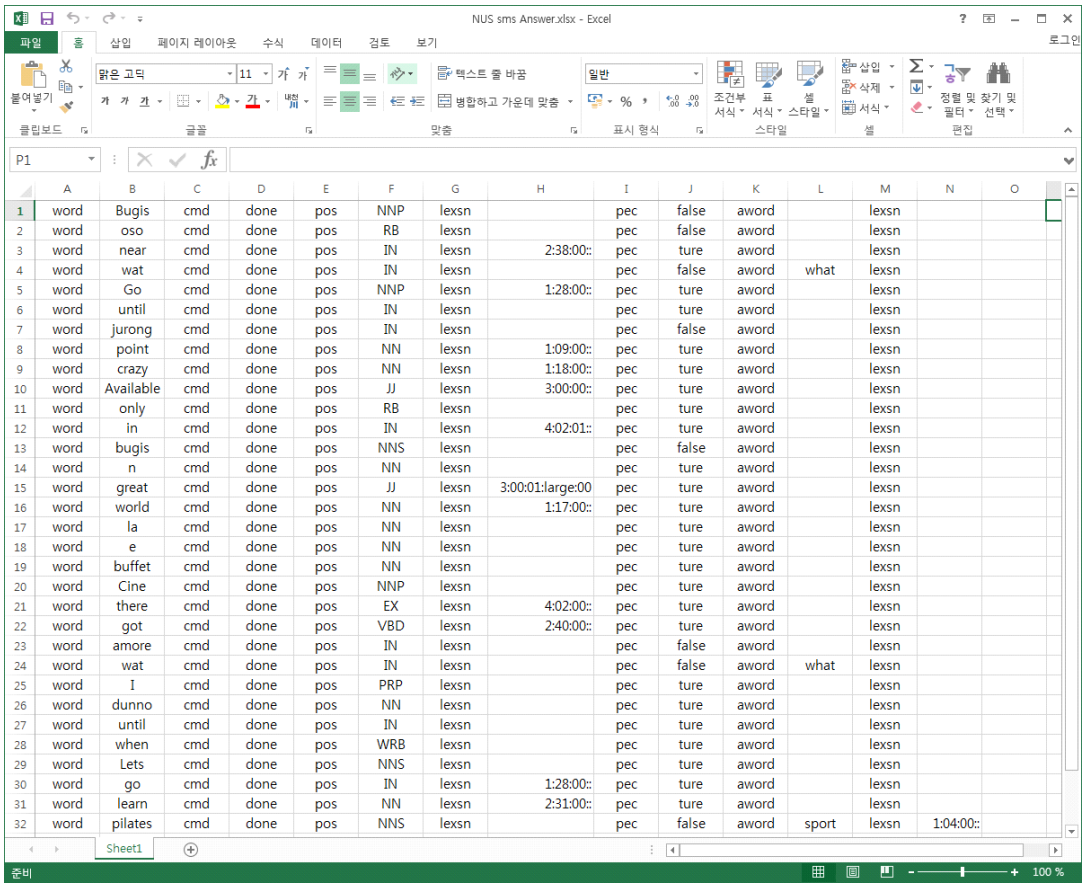


그림 23. NUS sms 말뭉치에서 추출한 실험 데이터 셋

정답 데이터 셋은 실험 데이터 셋의 미등록어를 GUN Aspell[9]과 인터넷 및 텍스트 속어 사전 및 번역기(Internet & Text Slang Dictionary & Translator)[10]를 이용해 대체 단어를 선정하고, 선정된 대체 단어를 미등록어와 대체하여 정답 데이터 셋을 생성하였다. 그림 24는 NUS sms 실험 데이터를 기반으로 한 정답 데이터 셋를 나타낸다.

9) http://aspell.net/

10) https://www.noslang.com/dictionary/l/



NUS sms Answer.xlsx - Excel

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	word	Bugis	cmd	done	pos	NNP	lexsn		pec	false	aword		lexsn		
2	word	oso	cmd	done	pos	RB	lexsn		pec	false	aword		lexsn		
3	word	near	cmd	done	pos	IN	lexsn	2:38:00::	pec	ture	aword		lexsn		
4	word	wat	cmd	done	pos	IN	lexsn		pec	false	aword	what	lexsn		
5	word	Go	cmd	done	pos	NNP	lexsn	1:28:00::	pec	ture	aword		lexsn		
6	word	until	cmd	done	pos	IN	lexsn		pec	ture	aword		lexsn		
7	word	jurong	cmd	done	pos	IN	lexsn		pec	false	aword		lexsn		
8	word	point	cmd	done	pos	NN	lexsn	1:09:00::	pec	ture	aword		lexsn		
9	word	crazy	cmd	done	pos	NN	lexsn	1:18:00::	pec	ture	aword		lexsn		
10	word	Available	cmd	done	pos	JJ	lexsn	3:00:00::	pec	ture	aword		lexsn		
11	word	only	cmd	done	pos	RB	lexsn		pec	ture	aword		lexsn		
12	word	in	cmd	done	pos	IN	lexsn	4:02:01::	pec	ture	aword		lexsn		
13	word	bugis	cmd	done	pos	NNS	lexsn		pec	false	aword		lexsn		
14	word	n	cmd	done	pos	NN	lexsn		pec	ture	aword		lexsn		
15	word	great	cmd	done	pos	JJ	lexsn	3:00:01:large:00	pec	ture	aword		lexsn		
16	word	world	cmd	done	pos	NN	lexsn	1:17:00::	pec	ture	aword		lexsn		
17	word	la	cmd	done	pos	NN	lexsn		pec	ture	aword		lexsn		
18	word	e	cmd	done	pos	NN	lexsn		pec	ture	aword		lexsn		
19	word	buffet	cmd	done	pos	NN	lexsn		pec	ture	aword		lexsn		
20	word	Cine	cmd	done	pos	NNP	lexsn		pec	ture	aword		lexsn		
21	word	there	cmd	done	pos	EX	lexsn	4:02:00::	pec	ture	aword		lexsn		
22	word	got	cmd	done	pos	VBD	lexsn	2:40:00::	pec	ture	aword		lexsn		
23	word	amore	cmd	done	pos	IN	lexsn		pec	false	aword		lexsn		
24	word	wat	cmd	done	pos	IN	lexsn		pec	false	aword	what	lexsn		
25	word	I	cmd	done	pos	PRP	lexsn		pec	ture	aword		lexsn		
26	word	dunno	cmd	done	pos	NN	lexsn		pec	ture	aword		lexsn		
27	word	until	cmd	done	pos	IN	lexsn		pec	ture	aword		lexsn		
28	word	when	cmd	done	pos	WRB	lexsn		pec	ture	aword		lexsn		
29	word	Lets	cmd	done	pos	NNS	lexsn		pec	ture	aword		lexsn		
30	word	go	cmd	done	pos	IN	lexsn	1:28:00::	pec	ture	aword		lexsn		
31	word	learn	cmd	done	pos	NN	lexsn	2:31:00::	pec	ture	aword		lexsn		
32	word	pilates	cmd	done	pos	NNS	lexsn		pec	false	aword	sport	lexsn	1:04:00::	

그림 24. NUS sms 실험 데이터 기반 정답 데이터 셋

정답 데이터 셋의 word 는 실험 데이터 셋의 단어, cmd는 실험 데이터 셋 단어의 태깅 여부, pos는 실험 데이터 셋 단어의 품사 정보, lexsn은 태깅된 개념의 워드넷 신셋 번호, pec는 미등록어의 여부, aword는 미등록어의 대체 단어를 의미한다. 본 연구에서는 실험 데이터 셋을 입력으로 실험을 수행하고 실험 데이터 셋과 동일한 형태로 결과를 저장한 뒤 대체된 단어가 정답 데이터 셋의 대체 단어와 일치하는지 비교하여 Word2VnCR 알고리즘의 성능을 측정한다.

2. 베이스라인(Baseline) 실험

본 절에서는 Word2VnCR 알고리즘의 베이스라인 실험을 수행하고 이에 대한 결과에 대하여 분석한다. 베이스라인 실험은 미등록어의 대체 후보 단어와 미등록어의 인접 단어의 의미적 유사도를 측정하는 실험이다. 이 방법은 미등록어의 대체 후보 단어 중 대체 단어를 선정하는데 있어 가장 좋은 워드넷 유사도 측정 방법을 선택하기 위함이다. 본 연구에서는 NUS sms 실험 데이터 셋을 입력으로 받아서 해당 문장에 존재하는 미등록어의 대체 단어를 선정하는데 있어 가장 적합한 워드넷 유사도 측정 방법을 결정하는 실험을 수행한다. 워드넷 유사도 측정 방법에는 에지기반으로 두 개념간의 유사도를 측정하는 PATH, LCH(Leacock & Chodorow), WUP 방법과 정보량을 기반으로 두 개념간의 유사도를 측정하는 RES(Resnik) 방법을 이용하였다.

표 11. 워드넷 유사도 측정 방법에 따른 대체 정확도 결과

구분	미등록어의 개수				
	1	2	3	4	5
path	50.82%	47.72%	45.81%	46.30%	45.20%
lch	54.22%	48.82%	46.84%	46.90%	45.42%
res	56.28%	49.75%	48.12%	47.88%	46.55%
wup	59.09%	52.03%	48.73%	49.50%	47.64%

베이스라인의 실험 결과는 다음 표 11에서 나타난 것과 같다. 실험의 결과 미등록어의 대체 후보 단어와 미등록어의 주변 인접 단어와 의미적 유사도를 측정했을 때 WUP 유사도 측정 방법이 다른 워드넷 유사도 측정 방법 보다 미등록어의 대체 후보 단어 중 의미적으로 가장 유사한 대체 단어를 선정하는 것을 확인할 수 있었다. 이에 따라 미등록어의 대체 후보 단어 중 대체 단어를 선정하는데 있어 WUP 유사도 측정 방법이 가장 적합한 것을 확인 할 수 있었다.

그림 25는 워드넷 유사도 측정 방법에 따른 대체 정확도의 성능을 나타낸다. 워드넷을 이용한 단어 간 의미적 유사도를 측정한 결과 미등록어의 개수가 증가할수록 미등록어의 대체 정확도가 낮아지는 것을 확인할 수 있었다. 이는 미등록어의 대체 후보 단어와 미등록어의 인접 단어를 통해 의미적 유사도를 측정하는 과정에 있어 미등록어 주변의 인접 단어가 워드넷 상에 존재하지 않는 단어로 이루어진 문장의 개수가 증가했기 때문이다. 즉, 미등록어의 대체 후보 단어와 미등록어의 인접 단어 간 의미적 유사도를 측정할 수 없기 때문이다. 이에 따라 미등록어의 개수가 증가 할수록 대체 정확도의 성능이 낮아졌다.

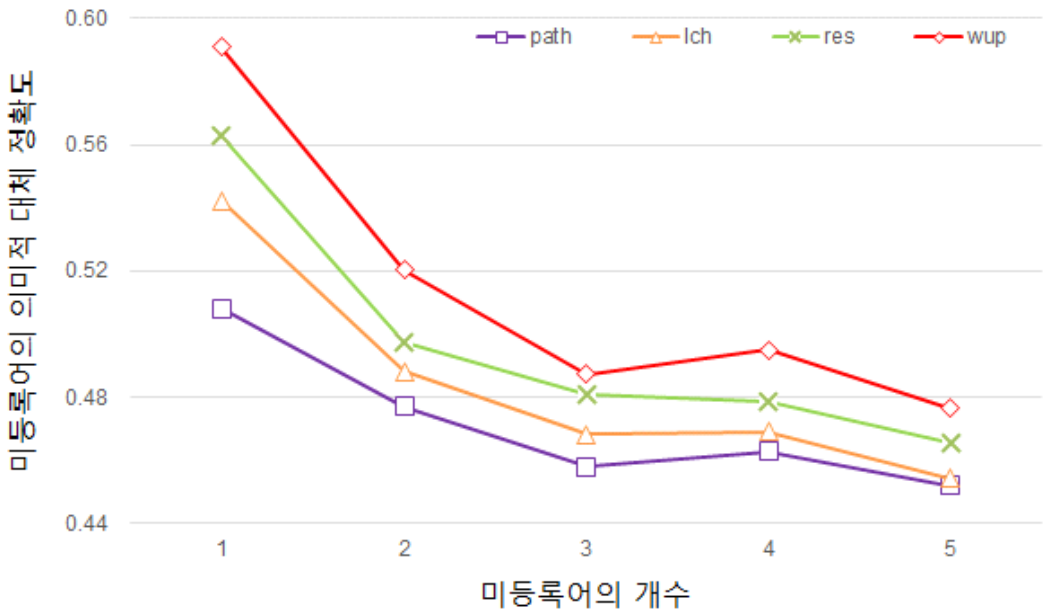


그림 25. 워드넷 유사도 측정 방법에 따른 대체 정확도의 성능

베이스라인 실험의 결과 미등록어의 대체 후보 단어 중 대체 단어를 선정하는데 워드넷 유사도 측정 방법 중 WUP 유사도 측정 방법이 가장 적합하다는 것을 확인할 수 있었다. 또한 문장 내 미등록어의 개수가 증가 할수록 미등록어의 의미적 유사 단어를 대체 하는 성능이 낮아지는 것을 확인 할 수 있었는데 이는 문장 내 미등록어의 개수가 증가할수록 미등록어의 대체 후보 단어와 의미적 유사도를 측정할 미등록어의 주변 단어가 줄어들기 때문임을 알 수 있었다.

3. Word2VnCR 알고리즘 기반 실험

본 절에서는 4장에서 기술한 미등록어를 의미적으로 유사한 단어로 대체하는 Word2VnCR 알고리즘의 실험을 수행하고 이에 대한 결과를 분석한다. 본 연구의 Word2VnCR 알고리즘의 비교 평가는 워드 임베딩 알고리즘을 대표하는 Word2Vec 알고리즘의 실험 결과와 비교를 통해 수행한다.

NUS sms 말뭉치는 총 55,963개의 텍스트로 구성되어 있으며, 이 중 약 80%에 해당하는 44,770개의 텍스트는 NUS sms 학습 데이터 셋으로, 약 20%에 해당 하는 11,193개의 텍스트는 NUS sms 실험 데이터 셋으로 나누어 실험을 진행하였다. 표 12는 NUS sms 실험 데이터 셋의 구성을 나타내는 표로 NUS sms 실험 데이터는 총 116,195개의 단어로 구성되어 있다. 이 중 명사 단어는 30,211개, 미등록어는 10,526개로 구성되어 있었다. NUS sms 실험 데이터 셋의 명사 단어 중 미등록어로 분류된 단어는 11,631개로 판별 되었지만 본 연구에서는 미등록어를 의미적으로 유사한 단어로 정확하게 해소 하고자 띄어쓰기의 오류 형태를 가진 미등록어 1,105개의 단어는 제외 하였다.

표 12. NUS sms 실험 데이터 셋의 구성

구분	단어의 수	명사의 수	미등록어로 분류된 단어의 수	미등록어의 수
NUS sms 실험 데이터	116,195	30,211	11,631	10,526

Word2VnCR 알고리즘과 Word2Vec 알고리즘의 실험은 실험 데이터 셋에서 추출한 미등록어의 대체 단어를 찾아 미등록어를 대체 단어로 대체한 뒤 대체된 단어가 정답 데이터 셋의 대체 단어와 일치하는지를 판단한다. 본 연구에서는 미등록어의 대체 단어가 잘 대체 되었는지를 정확하게 판단하는 것임으로 정확도를 중심으로 성능을 평가한다. Word2VnCR 알고리즘과 Word2Vec 알고리즘의 실험 결과는 다음 표 13에서 나타난 것과 같다.

표 13. Word2VnCR 알고리즘과 Word2Vec 알고리즘의 실험 비교 결과

구분	미등록어의 수 (A)	대체 된 단어의 수(B)	대체에 성공한 단어의 수(C)	재현율(%) (B/A)	정확도(%) (C/B)
Word2VnCR	10,526	9,167	4,597	87.09	50.15
Word2Vec	10,526	9,524	4,325	90.48	45.41

NUS sms 실험 데이터 셋을 대상으로 Word2VnCR 알고리즘과 Word2Vec 알고리즘의 비교 실험을 수행한 결과 재현율은 각각 87.09%와 90.48%를 정확도는 각각 50.15%와 45.41%를 얻을 수 있었다. Word2Vec 알고리즘의 재현율이 높게 측정된 것은 Word2VnCR 알고리즘과 워드 임베딩 방식이 다르기 때문에 미등록어를 더 많이 대체 한 것으로 판단된다. Word2VnCR 알고리즘의 정확도가 높게 측정된 것은 Word2VnCR 알고리즘은 미등록어와 대체 단어의 유사도를 의미적 유사도 방법을 이용해 측정했기 때문이고, Word2Vec 알고리즘은 미등록어와 대체 단어의 유사도를 벡터 공간상에 두 단어가 위치한 각도를 이용한 코사인 유사도 방법을 이용해 측정했기 때문이다. 즉, Word2Vec 알고리즘은 미등록어를 많이 대체하였어도 의미가 다른 단어로 대체하였다. 이를 통해 본 연구에서 제안하는 Word2VnCR 알고리즘이 미등록어를 대체하는데 있어 정확도면에서 Word2Vec 알고리즘 보다 높은 성능을 보이는 것이 확인되었다.

NUS sms 실험 데이터 셋 내 텍스트의 미등록어 수 증가할 때 마다 미등록어의 대체가 어떻게 달라지는지를 살펴보기 위하여 실험을 수행하였다. NUS sms 실험 데이터 셋의 텍스트에서 미등록어가 한 개인 것부터 다섯 개인 텍스트를 각각 추출하여 본 연구에서 제안한 방법으로 실험을 한 결과 표 14, 15로 나타났다. 표 14와 표 15는 통해 텍스트 내 미등록어가 적을수록 미등록어의 대체를 정확히 할 수 있음을 판단하였으며, 미등록어가 증가할수록 의미적 유사도를 측정할 수 없어 미등록어의 대체 결과가 좋지 않은 것으로 판단할 수 있었다.

표 14. Word2VnCR 알고리즘을 이용한 미등록어의 대체 실험 결과

문장 내 미등록어의 수	미등록어의 수	대체 된 단어의 수(A)	대체에 성공한 단어의 수(B)	정확도(%) (B/A)
1	632	572	338	59.091%
2	960	836	435	52.033%
3	1,223	1,065	519	48.732%
4	1,487	1,295	641	49.498%
5	1,436	1,251	596	47.642%

표 15. Word2Vec 알고리즘을 이용한 미등록어의 대체 실험 결과

문장 내 미등록어의 수	미등록어의 수	대체 된 단어의 수(A)	대체에 성공한 단어의 수(B)	정확도(%) (B/A)
1	632	564	298	52.837%
2	960	936	413	44.124%
3	1,223	1,172	507	43.278%
4	1,487	1,375	573	41.673%
5	1,436	1,326	546	41.176%

표 16. Word2VnCR 알고리즘과 Word2Vec 알고리즘의 정확도 비교 결과

문장 내 미등록어의 수	정확도	
	Word2VnCR	Word2Vec
1	59.091%	52.837%
2	52.033%	44.124%
3	48.732%	43.278%
4	49.498%	41.673%
5	47.642%	41.176%

NUS sms 실험 데이터 셋 내 텍스트의 미등록어 수 증가할 때 마다 미등록어 대체 정확도 결과인 표 16를 그래프로 표현하면 그림 26과 같다. 아래 그림에서 확인할 수 있듯이 텍스트에 작성된 미등록어의 의미적 단어 대체는 Word2VnCR 알고리즘을 수행해 미등록어를 의미적으로 유사한 단어로 대체하는 경우 Word2Vec의 알고리즘보다 좋은 성능을 가진다는 것을 확인 할 수 있었다.

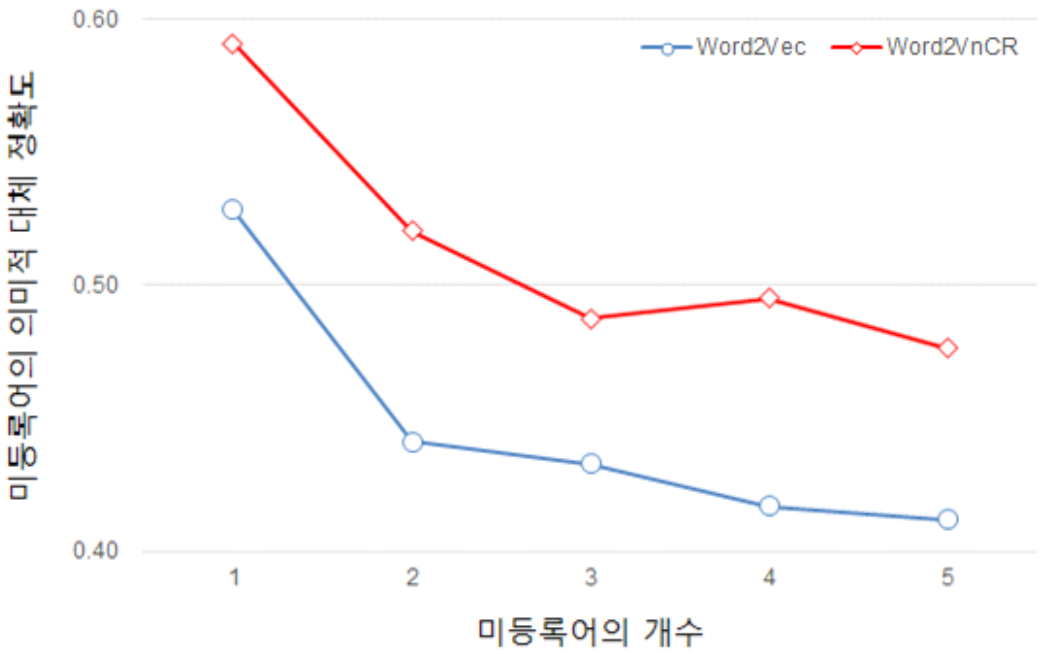


그림 26. Word2VnCR 알고리즘과 Word2Vec 알고리즘의 정확도 비교

결국, 형태소 분석의 오류인 미등록어를 의미적으로 유사한 단어로 대체하는 연구에 있어서 미등록어가 적은 텍스트를 워드 임베딩 학습하고 이를 기반으로 미등록어의 대체 후보 단어를 추출 해 미등록어 주변 인접 단어들과 의미적 유사도를 측정해 미등록어의 의미적 유사한 대체 단어를 선정해 대체 하면 가장 효과적으로 미등록어를 의미적으로 유사한 단어로 대체할 수 있다는 사실을 도출할 수 있었다.

Ⅵ. 결론 및 향후 연구

본 논문은 사람이 사용하는 자연어로 표현된 문서를 컴퓨터가 처리 및 분석해 문서에 내포된 지식정보를 학습하고 이해하기 위한 기반 연구로써 형태소 분석의 오류인 미등록어를 의미적으로 유사한 단어로 대체하는 방법에 대해 다루고 있다. 본 연구는 형태소 분석의 오류인 미등록어를 의미적으로 유사한 단어로 대체함에 있어서 미등록어와 의미가 유사한 대체 후보 단어를 추출하는 방법과 미등록어의 대체 단어를 선정하기 위해 미등록어의 대체 후보 단어와 미등록어의 인접 단어와 의미적 유사도를 측정하는 방법을 반영한 Word2VnCR 알고리즘을 새롭게 제안하였다.

미등록어가 대체 될 대체 후보 단어는 워드 임베딩 방법을 이용해 미등록어의 대체 후보 단어를 추출한다. 워드 임베딩 방법은 주어진 말뭉치에 있는 모든 단어를 벡터로 표현하는 학습방법으로 여러 개의 단어들 사이의 유사도를 측정할 수 있고 벡터화 된 의미들을 가지고 벡터 연산을 하여 추론 할 수 있다. 본 연구에서는 워드 임베딩 방법의 문맥을 기반으로 단어를 예측하는 방법을 통해 학습 데이터 셋에서 학습 데이터를 생성하고 생성된 학습 데이터를 기반으로 미등록어의 대체 후보 단어들을 추출한다.

미등록어의 대체 후보 단어 선정은 WUP 유사도 측정 방법을 이용해 미등록어의 대체 후보 단어 중 미등록어가 대체 될 대체 단어를 선정한다. 워드넷은 단어의 개념을 정의하고 있는 어휘 사전으로 단어의 개념간의 계층관계 거리를 측정해 단어간 의미가 얼마나 유사한가를 측정할 수 있다. 본 연구에서는 워드넷 유사도 측정 방법 중 하나인 WUP 유사도 측정 방법을 통해 미등록어의 대체 후보 단어와 미등록어가 출현한 문장 내 명사 단어와의 의미적 유사도를 측정해 높은 측정값을 가지는 대체 후보 단어를 대체 단어로 선정하여 미등록어와 대체를 한다.

Word2VnCR 알고리즘은 실험 데이터 셋에서 추출한 미등록의 대체 단어를 찾아 미등록어를 의미적으로 유사한 단어로 대체한 뒤 대체된 대체 단어가 정답 데이터 셋의 단어와 일치하는지를 판단하는 실험을 수행하였다. Word2VnCR 알고리즘은

대체 단어가 잘 대체 되었는지를 정확하게 판단하는 것임으로 정확도를 중심으로 성능을 평가하였다. 실험 결과 Word2VnCR 알고리즘은 Word2Vec 알고리즘 보다 높은 정확도로 미등록어를 의미적으로 유사한 단어로 대체 하였다. 또한 텍스트의 미등록어 수 증가할 때 마다 미등록의 의미적 대체가 어떻게 달라지는지를 살펴보기 위하여 실험을 수행하였다. 그 결과 텍스트 내 미등록어 수가 적을수록 미등록어의 의미적 대체를 정확히 할 수 있음을 판단하였으며, 미등록어가 증가할수록 미등록어의 의미적 대체 결과가 정확하지 않은 것으로 판단하였다. 이를 통해 Word2VnCR 알고리즘은 미등록어를 의미적으로 유사한 단어로 대체 하는 데 있어 가장 효과적이라는 사실을 도출할 수 있었다.

결과적으로 이 연구에서 제안하는 Word2VnCR 알고리즘은 미등록어를 의미적으로 유사한 단어로 대체하는데 있어 높은 정확도 나타내었다. 그러나 학습 데이터 셋을 어떻게 구축하느냐에 따라 실험의 결과는 영향을 받는다. 학습 데이터 셋의 워드 임베딩 학습이 재대로 이루어 지지 않는 다면 미등록어의 대체 후보 단어를 정확히 추출할 수 없기 때문이다. 따라서 미등록어가 적게 출현하고 미등록어의 인접 단어가 의미적인 단어로 이루어진 텍스트를 학습 데이터에 추가해 미등록어를 의미적으로 대체하는 연구가 추가적으로 진행되어야 한다.

참 고 문 헌

[1] 기계공학사전편찬위원회 저, “기계공학대사전”

[2] 전산용어사전편찬위원회 저, “컴퓨터 인터넷 IT용어 대사전”

[3] https://ko.wikipedia.org/wiki/자연어

[4] https://ko.wikipedia.org/wiki/자연어_처리

[5] https://ko.wikipedia.org/wiki/형태소

[6] https://namu.wiki/w/단어

[7] Tom, H., Yehezkel, S., and Ltay, L., “Learning TensorFlow”

[8] 이향이, “동시출현단어 분석에 기반한 한국사 분야 연구동향 분석”, 연세대학교 대학원 석사학위논문, 2015.

[9] Ide, N., and Veronis, J., “Word sense disambiguation: The state of the art”, Computational Linguistics, Vol. 24, No. 1, pp. 1-40, 1998.

[10] 최동진, “지능적 문서 분석을 위한 개선된 WSD 방법 연구”, 조선대학교 대학원 박사학위논문, 2015.

[11] 이용호, “LCS알고리즘을 이용한 자동 학습집합 구축 트래픽분류 모델”, 상명대학교 대학원 석사학위논문, 2013.

[12] https://ko.wikipedia.org/wiki/최장_공통_부분_수열

[13] 태윤식, “자기 조직화 n-gram 모델을 이용한 자동 띄어쓰기”, 경북대학교 대학원 석사학위논문, 2007.

[14] https://terms.naver.com/entry.nhn?docId=862633&cid=42346&categoryId=42346

[15] Kim, J.D., Rim, H.C., and Tsujii, J., “Self-Organizing Markov Modelsand Their Application to Part-of-Speech Tagging”, Proceedings of the 41st Annual Meeting on

Association for Computational Linguistics, Vol. 1, pp. 296-302, 2003.

[16] Lee, D.G., Lee, S.Z., Rim, and Lim, H.S., "Automatic Word Spacing Using Hidden Markov Model for Refining Korean Text Corpora", Proceedings of the 3rd Workshop on Asian Language Resources and International Standardization, Vol. 12, pp. 51-57, 2002.

[17] 송현제, "확률 모델과 웹 검색을 결합한 로마자-한글 음차표기", 경북대학교 대학원 석사학위논문, 2010.

[18] 태윤식, 박성배, 이상조, 박세영., "자기조직화 n-gram 모델을 이용한 자동 띄어쓰기", 한국정보과학회 언어공학연구회 학술발표 논문집, pp. 125-132, 2006.

[19] Morris, S.A., and Van der veer martens, B., "Mapping research specialties", Annual Review of Information Science and Technology, Vol. 42, Issue. 1, pp. 213-295, 2008.

[20] Yang, Y., Wu, M., and Cui, L., "Integration of three visualization methods based on co-word analysis", Scientometrics, Vol. 90, pp. 659-673. 2012.

[21] 이일주, "단어의 공기정보를 이용한 클러스터 기반 다중문서 요약", 아주대학교 대학원 박사학위논문, 2006.

[22] 김강현, "Co-word Analysis를 이용한 Entrepreneurship의 지적구조 분석 관련연구", 숭실대학교 대학원 석사학위논문, 2011.

[23] Callon, M., John, L., and A. Rip., "Mapping of the dynamics of science and technology", London: Macmillan, 1986.

[24] 이미경, "동시출현 단어 분석을 통한 지식 구조 파악에 관한 연구 : 인공지능 분야를 대상으로", 연세대학교 대학원 석사학위논문, 2003.

[25] 황정태, "co-word 기술지도작성을 위한 네트워크 영향 지수의 개발 : Development of network influence index for co-word map", 서울대학교 대학원 박사학위논문, 1997.

[26] Peters, H.P.F., and A.F.J. van Raan., "Co-word-based science maps of chemical engineering. Part I: Representations by direct multidimensional scaling". Research Policy, Vol. 22, Issue. 1, pp. 23-45, 1993.

[27] Coulter, N., Monarch, I., and Konda, S., “Software engineering as seen through its research literature: a study in co-word analysis”, Journal of the American Society for information Science, Vol. 49, pp. 1206-1223, 1998.

[28] Ding, Y., Chowdhury, G.G., and Foo, S., “Bibliometric cartography of information retrieval research by using co-word analysis”, Information Processing and Management, Vol. 37, pp. 817-842, 2001.

[29] Choi, D.S., et al., “A Two-Phase Dependency Parser of Korean,” Proceedings of the natural language pacific rim symposium, 1995.

[30] Lee, J.H., et al., “Structural Disambiguation Using Constraint-Satisfaction Algorithm for Dependency Parsing”, Proceesings of the International Conference on Computer Processing of Oriental Language, pp. 213-216, 1995.

[31] 김동주, “철자오류에 기인한 가의미 오류의 검출 및 교정 방법”, 한국컴퓨터정보학회논문지, Vol. 18, pp. 173-182, 2013.

[32] 윤영신, “워드 임베딩을 이용한 질병과 바이오마커/미생물 관계 분석”, 한림대학교 대학원 석사학위논문, 2017.

[33] Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., and Kuksa, P., “Natural language processing (almost) from scratch”. Journal of Machine Learning Research, Vol. 12, pp. 2493-2537, 2011.

[34] 윤병훈, “워드 임베딩을 이용한 만성 폐쇄성 폐 질환과 바이오마커의 상관관계 분석”, 한림대학교 대학원 석사학위논문, 2018.

[35] Mikolov, T., Chen, K., Corrado, G., and Dean, J., “Efficient Estimation of Word Representations in Vector Space”, 2013.

[36] http://blog.naver.com/eun9659/221233326276

[37] http://prefity.blogspot.com/2017/05/word2vec.html

[38] Rong, X., “Tomas Mikolov, Kai Chen, Greg Corrado, Jeffrey Dean”, 2016.

[39] Goldberg, V., “A Primer on Neural Network Models for Natural Language

Processing", 2015.

[40] http://dsp.yonsei.ac.kr/139926

[41] https://dreamgonfly.github.io/machine/learning,/natural/language/processing/2017/08/16/word2vec_explained.html

[42] 조우진, "의미커널과 한글 워드넷에 기반한 지능형 채점 시스템", 한림대학교 대학원 석사학위논문, 2006.

[43] 추승우, 오정석, 김유섭, 이재영., "워드넷 기반의 임의 추출 분할 방식을 이용한 동적 문제 출제 시스템 설계", 한국정보과학회, Vol. 31, pp. 283-285, 2004.

[44] Miller, G.A., "WordNet : A Lexical Database for English", Communications of th ACM, Vol. 38, Issue. 11, 1995.

[45] Miller, G.A., "WordNet : An On-line Lexical Database", International Journal of Lexicography, 1990.

[46] 김형일, 김준태., "워드넷 기반 협동적 평가와 하이퍼링크를 이용한 검색엔진의 성능 향상", 정보처리학회논문지, pp. 369-380, 2004.

[47] 조우진, 오정석, 이재영, 김유섭., "의미커널과 한글 워드넷에 기반한 지능형 채점 시스템", 한국정보처리학회, Vol. 12, No. 6, pp. 539-546, 2005.

[48] Budanitsky, A., and Hirst, G., "Evaluating WordNet-based Measures of Lexical Semantic Relatedness", Computational Linguistics, Vol. 32, No. 1, pp. 13-47, 2006.

[49] Wu, Z., and Palmer, M., "Verb Semantics and Lexical Selection", Annual meeting of the Associations for Computational Linguistics, pp. 133-138, 1994.

[50] 김정인, 김판구., "의미적 유사도 측정 방법을 활용한 철자교정", Convergent & Smart Media Systems, pp. 1-2, 2016.