KIPS Tr. Software and Data Eng.
Vol.4, No.1 pp.45~56 pISSN: 2287-5905

http://dx.doi.org/10.3745/KTSDE.2015.4.1.45

Context-sensitive Word Error Detection and Correction for Automatic Scoring System of English Writing

Yong Seok Choi† · Kong Joo Lee††

ABSTRACT

In this paper, we present a method that can detect context-sensitive word errors and generate correction candidates. Spelling error detection is one of the most widespread research topics, however, the approach proposed in this paper is adjusted for an automated English scoring system. A common strategy in context-sensitive word error detection is using a pre-defined confusion set to generate correction candidates. We automatically generate a confusion set in order to consider the characteristics of sentences written by second-language learners. We define a word error that cannot be detected by a conventional grammar checker because of part-of-speech ambiguity, and propose how to detect the error and generate correction candidates for this kind of error. An experiment is performed on the English writings composed by junior-high school students whose mother tongue is Korean. The f1 value of the proposed method is 70.48%, which shows that our method is promising comparing to the current-state-of-the art.

Keywords : Context-sensitive Word Error, Error Detection/Correction, Confusion Set, English Automatic Scoring, Grammar-level Word Error

영작문 자동 채점 시스템을 위한 문맥 고려 단어 오류 검사기

최 용 석† · 이 공 주††

요 약

본 연구에서는 문맥 정보를 함께 고려해야만 인식할 수 있는 단어 오류에 대하여 오류 인식 방법과 수정 후보 생성 방법을 제안한다. 이 문제는 기존의 영어권에서 이미 많이 다룬 연구 주제이다. 본 연구에서는 영어 자동채점 시스템에서 사용하도록 특화된 방법을 제안한다. 문맥 정보를 고려한 단어 오류 검사에서는 자주 혼동되어 사용되는 단어집합(confusion set)을 활용한다. 비영어권 사용자의 작문 특성을 반영하기 위해 기존의 영어권에서 구축한 혼동집합 이외에 자동으로 혼동집합을 구축하여 실험해 보았다. 또한 품사 중의성으로 인해 기존의 구문오류 검사기가 다루지 못하는 오류를 정의하고 오류 인식과 오류수정 후보를 생성하는 방법을 제안한다. 실제 한국어가 모국어이면서 초/중급 작문 수준의 수험생들이 작성한 영어 문장에 대해 평가해 본 결과, 약 70.48%의 f1 값을 얻어 기존의 영어권 결과에 비해 뒤지지 않는 성능을 보였다.

키워드 : 문맥 고려 단어 오류, 오류 인식/수정, 혼동집합, 영어 자동 채점, 구문 수준 단어 오류

1. 서 론

단어 오류 검사기는 자연언어처리 분야에서는 오랫동안 연구되어 온 응용 프로그램 중 하나이다. 단어 오류 검사는 크게 두 가지 접근 방법으로 나눌 수 있다[1]. 첫 번째 종류의 연구는 사전 미등록 단어에 대한 오류 수정을 하는 것이다. 입력 대상으로부터 사전(lexicon)에 정의되어 있지 않은 오류 단어가 있는지를 검사한 후, 오류 단어가 존재하면, 오류 단어와 가장 유사한 형태의 수정 단어 후보를 생성하고 다양한 유사도 척도를 이용하여 가장 적합한 수정 단어를 결정한다. 이와 같은 접근 방법은 오류 여부를 명확히 판단할 수 있기 때문에 정확한 후보를 생성하는 것이 연구의 초점이 된다. 두 번째는 문맥을 고려한 오류 검사 및 수정을 수행하는 연구로써 일반적으로 사전에 등록되어 있는 단어이지만 문맥을 고려해 보았을 때 부적합한 단어가 사용된 경우이다. 이 경우는 오류를 판별하는 것부터가 연구의 핵심

※ 이 연구는 충남대학교 학술연구비에 의해 지원되었음
† 비 회 원 : 충남대학교 정보통신공학과 학사과정
†† 정 회 원 : 충남대학교 정보통신공학과 교수
Manuscript Received : October 31, 2014
First Revision : December 19, 2014
Accepted : December 19, 2014
* Corresponding Author : Kong Joo Lee(kjoolee@cnu.ac.kr)

주제가 된다. 영어 작문 자동평가 시스템에서 단어 오류를 정확히 검출해 내는 것은 무척이나 중요하다. 특히 다소 길이가 짧은 단문 형식의 영어 작문[1]인 경우 철자 오류나 구문 오류 인식기의 정확도에 따라 언어사용 능력 영역 점수의 정확도가 크게 좌우된다.

철자 오류 인식은 일반적으로 쉽다고 알려져 있으나 자동채점의 경우에는 문맥을 고려하여 사전에 등록된 단어라도 오류로 추정해야 하는 경우가 존재한다. 다음의 예를 살펴보자. (예 1)과 (예 2)의 밑줄 친 단어는 모두 사전에 등록되어 있는 단어이기 때문에 철자 오류로 인식되지 않는다. 그러나 이 경우에는 각각 'slimmer'와 'about'을 잘못 쓴 철자 오류에 해당한다.

(예 1) Exercises will make you simmer.
(예 2) If you want to look better, how abut having a haircut?

또한 다음의 예제문장들을 살펴보자.

(예 3) The students had been scold by their teacher.
(예 4) He took the taxi, he has show paper to driver.

(예 3)과 (예 4)는 구문 오류에 해당한다. 그러나, 대부분의 구문 오류 인식기가 (예 3)과 (예 4)의 문장을 오류가 없는 정문으로 판단한다. 구문 오류 인식기가 명백한 오류를 제대로 인식하지 못하는 원인 중 하나는 품사 중의성 때문이다. (예 3) 문장의 경우 의미를 고려할 때 'scold'는 'scolded'의 형태로 사용되어야 한다. 즉, 동사구 결합 형태의 오류가 존재한다. 그러나 'scold'가 명사와 동사로 모두 사용될 수 있으며 대부분의 구문 오류 인식기가 품사 정보만을 이용하여 오류를 인식하다 보니 구문 오류가 제대로 인식되지 못한다. 예문의 경우 품사 나열로만 본다면, [had, been, NOUN, by, DET, NOUN]으로 오류 없는 품사 나열이 되기 때문에 구문 오류로 인식되지 못한다. (예 4) 문장의 경우에도 'show paper' 부분이 [NOUN, NOUN]의 품사 열로 인식되면서 동사구 결합 형태 오류를 인식하지 못하게 된다. 이와 같은 문제들로 인해 명백한 구문 오류가 발생하지 않았다 하더라도 구문 오류의 추정이 필요하다.

본 연구에서는 이와 같이 기존의 철자 오류 검사나 구문 오류 인식기가 수행하지 못한 오류들을 찾아내고 적절한 수정 후보를 찾아내는 방법에 대한 제안을 하고자 한다. 이와 같은 시스템은 영어 작문 평가 시스템에 적용되어 그 성능을 평가해 본다.

1) 본 연구에서 대상으로 한 영어 작문 평가는 주어진 조건에 대하여 10단어 이내로 단문을 완성하거나 또는 25단어 이내의 2~3문장으로 구성된 짧은 글쓰기이다[2].

2. 관련 연구

단어 오류에 대한 오류 검출은 크게 두 가지 방향으로 나눌 수 있다[1]. 첫째는 미리 구축된 사전에 존재하지 않는 단어를 찾아냄으로써 오류 단어를 인식하고 오류 단어와 가장 유사한 단어 리스트를 오류 단어의 수정 후보 단어로 생성해 내는 방법이다. 두 번째 접근 방향은 사전에는 존재하는 유효단어이지만 해당 문장에서는 부적합하게 사용된 단어를 오류로 인식해 내는 문맥 의존(context sensitive) 오류 단어 추정이다. 본 연구에서는 두 번째 접근 방법인 문맥 의존 오류 인식을 다룬다.

문맥 의존 오류 인식에 가장 널리 사용되는 방법은 발음이나 철자의 유사성으로 인해 자주 혼동되어 사용되는 단어 집합을 미리 정의해 놓고 이를 이용하여 오류를 인식하는 것이다. 혼동 단어에 대한 해결 방법은 [3]의 베이지안 분류기(Bayesian Classifier) 방법이 있다. 이 연구에서는 {desert, dessert}와 같은 혼동집합(confusion set)을 우선 수집한 후, 각 혼동단어를 해소하기 위하여 자질(feature)들을 학습데이터로부터 추출한다. 사용한 자질 정보는 대상 단어를 중심으로 k-크기 윈도우 내에 존재하는 문맥 정보와 대상 단어를 중심으로 앞/뒤쪽으로 확장한 n-gram 정보이다. 각 자질에 대한 확률 값을 학습데이터로부터 추출하고 베이지안 분류방법을 사용하여 가장 높은 확률 값을 갖는 단어를 오류 수정 후보단어로 결정하였다.

연구 [4]에서는 철자, 발음, 문법적으로 자주 혼동되어 사용되는 단어에 대한 중의성 해소를 위한 자원으로 웹 전체 규모와 맞먹는 데이터를 직접 사용하는 방법을 제안하였다. 이 연구에서 사용한 데이터는 웹 크기의 Google Web 5-gram 코퍼스를 직접 사용하였다. 동일한 크기의 n-gram을 사용한 기존의 연구와는 다르게 이 연구에서는 혼동되어 사용될 수 있는 단어를 포함한 5개의 5-gram, 4개의 4-gram, 3개의 trigram, 2개의 bigram을 사용하여 모두 14개의 주변 패턴 정보(context pattern)를 중의성 해소에 사용하였다.

연구 [5]에서는 불어를 제2외국어로 사용하는 학습자가 작성한 불어 문장에서 잘못 사용한 전치사를 찾아내서 수정하는 시스템을 제안하였다. 전치사의 사용은 전치사와 결합하는 각 어휘와의 의미관계뿐만 아니라 관용적으로 사용되는 경우도 많다. 그렇다보니 정확한 전치사 사용의 전체 규칙을 만들어 내는 것도 어렵고 이를 학습자에게 학습시키는

것도 어려우며 나아가서 이를 시스템으로 구현하는 것은 더욱 어렵다. 이와 같은 문제의 가장 일반적인 접근 방법은 대량의 코퍼스 정보를 이용하여 부정확하게 사용한 문법 요소를 찾아내는 것이다. 이 연구에서는 오류를 판별하고자 하는 대상 전치사에 대해 전치사를 포함하고 있는 구절을 우선 간결하게 정리한 후 일반화과정을 거쳤다. 이렇게 하는 이유는 대량의 코퍼스로부터 좀 더 정확한 정보를 추출하기 위함이다. 대상 전치사와 자주 혼동되어 사용되는 전치사 리스트를 만든 후 원래 구절에서 혼돈되는 전치사를 대치한 후 웹 검색을 통해서 가장 높은 빈도를 갖는 전치사를 최종 수정 전치사로 결정하였다. 중등 수준의 작문 능력을 가진 학생이 작성한 전치사를 포함한 133문장에 대해 평가한 결과 약 69.9%의 정확도를 얻을 수 있었다.

연구 [6]에서는 각 오류 유형에 따라 오류 처리 모듈을 따로 구축하였다. 대상으로 한 오류는 관사, 전치사 사용 오류와 동사관련 오류, 형용사관련 오류, 명사관련 오류를 다루었다. 약 2.5M 문장으로 구성된 학습 데이터를 사용하였으며, 오류가 발생할 수 있는 대상 단어를 중심으로 양쪽 옆 3개 단어와 그 품사 정보를 자질로 사용하여 분류기를 구축하였다. 주로 동아시아권(중국/일본/한국) 사람들이 작성한 영어 문장에서 오류를 인식하고 수정후보를 생성하는 작업에 대한 평가를 수행한 결과 관사오류에 대해서 약 83.9%, 전치사오류에 대해서는 64.3%의 정확도를 얻을 수 있었다.

본 연구에서 다루는 문제는 모두 영어 쓰기 자동채점 시스템을 고려하여 개발되었다[2]. 영어 쓰기 자동채점 시스템은 사람에 의해 수동으로 채점된 일부의 학생 답안으로부터 자동 채점에 필요한 자질 집합을 추출하고 이를 이용하여 자질집합과 점수와의 상관관계를 학습한다. 그렇기 때문에 자동채점을 위한 작문 문제와 정답집합이 존재하며 수동 채점된 답안 집합이 일부 존재한다. 본 연구는 이와 같이 기채점된 답안으로부터 철자오류나 구문오류 인식 및 후보 생성에서 활용할 수 있는 정보를 추출할 수 있는 환경이다.

3. 문맥을 고려한 단어 오류 추정

본 연구에서 다루는 단어 오류는 사전에는 존재하는 단어지만 해당 문장에서는 부적합하게 사용된 단어로서 다음과 같은 세 가지 방법으로 처리한다.

본 연구는 영어 작문 자동 채점에서 사용되기 때문에 사람에 의해 기채점된 정답 집합이 존재한다는 특징이 있다. 그렇기 때문에 첫째, 정답에 가까운 고득점 답안에 자주 등장한 단어를 참조하여 단어 오류의 수정 후보를 추정할 수 있다. 둘째 방법은 미리 정의된 혼동집합(confusing set)에 대하여 코퍼스에서 추출한 정보를 이용하여 오류 후보를 결정하는 것이다. 이것은 [3]의 베이지안 분류기를 이용하여 처리한다. 세 번째는 품사 중의성으로 인해 인식하지 못하는 구문 수준의 단어 오류이다. 이 경우는 n-gram 언어 모델을 이용하여 해결한다.

본 연구에서 다루는 영작문은 주로 초/중급의 작문 수준을 가진 학생이 작성한 문장이다. 그러다보니 처리 대상의 문장들 상당수가 비문인 경우가 많다. 그렇기 때문에 정문에 대해서만 처리하는 기존의 방식으로는 처리가 불가능한 경우가 많다. 본 연구에서 제안하는 처리 방법은 영작문에 단어 또는 문법 오류가 하나 이상 있더라도 수행가능하다.

3.1 정답 단어를 이용한 철자 오류 추정

영어 쓰기 자동 채점 시스템에서 기계 학습을 수행하기 위해서는 수험생들이 작성한 영어 작문에 대해 사람에 의해 수동으로 채점된 답안 집합이 필요하다. 이와 같이 점수가 부여된 답안 집합으로부터 50% 이상의 점수를 받은 학생의 답안을 수집하고, 이들로부터 답안에 사용된 단어들과 그 빈도수를 수집하여 정답 단어 리스트 *answer_word_list*를 구축해 놓는다. 이때 *answer_word_list*에 등록되는 단어들은 모두 사전에 존재하는 유효한 단어로만 한정시킨다.

본 연구에서 사용하는 영어 사전의 각 단어는 [엔트리, 품사정보, 활용정보, 기타정보]로 구성되어 있으며 기타정보로는 코퍼스에서 추출한 발생빈도와 단어의 수준(초등수준, 중등수준, GRE수준) 정보가 포함되어 있다. 본 연구에서 다루는 영어 작문 시험의 대상 수험생은 주로 중등 수준의 영어 능력을 갖춘 사람들이다. 그렇기 때문에 입력된 수험생 답안에 사용된 단어가 저빈도 단어이거나 초등/중등 수준을 벗어나는 단어일 경우 오류 추정을 시작한다.

수험생 입력 답안 중, *answer_word_list*에 존재하지 않으면서 저빈도 단어이거나 초등/중등 수준을 벗어나는 입력 단어를 W_0라고 하자. *answer_word_list*의 단어 중, W_0와 철자가 유사하거나 발음이 유사한 단어가 존재한다면 그 중 최고 빈도를 갖는 단어를 W_0의 오류 수정 후보로 결정한다. 철자 유사성은 편집거리(edit distance)를 계산해서 사용하며 발음 유사성은 Double Metaphone 알고리즘[7]을 사용하였다. 이 알고리즘은 입력된 영어 단어를 핵심되는 발음 정보로 변환하여 출력해준다. 추정된 오류 후보를 W_1라고 하고 W_0와 W_1을 포함하는 어휘 트라이그램의 빈도 값을 확인하여 빈도가 높은 것을 최종 오류 수정 단어로 결정한다. 만약 W_0을 포함한 어휘 트라이그램의 빈도 값 합이 W_1을 포함한 어휘 트라이그램의 빈도 값보다 높으면 W_0가 제대로 사용

된 것으로 결정한다.

예를 들어 수험생 입력 문장이 다음과 같을 때, 단어 'abut'는 사전 등록단어이지만 저빈도 단어이면서 중등 수준을 벗어나는 단어이기 때문에 오류 추정을 수행해 본다.

(예 2) If you want to look better, how abut having a haircut?

수험자 답안에 쓰인 단어와 그 출현 빈도로 구성된 리스트 *answer_word_list* = {(you, 23), (make, 16), (slimmer, 4), (look, 9), (about, 5), (want, 10)...}가 존재할 때, (예 2)의 단어 'abut'는 *answer_word_list*에 존재하지 않지만 유사한 철자를 갖는 'about'는 *answer_word_list*에 존재함을 알 수 있다. 이런 경우, 철자 오류의 가능성이 있음을 추정해 볼 수 있다.

대상단어가 *answer_word_list*에 존재하지 않고 대상단어가 중등 수준을 벗어난 단어일 때, Double Metaphone 알고리즘과 최소편집거리(Minimum Edit Distance)를 계산하여 편집거리가 1 이하이고 발음 값이 동일할 경우 정답으로 간주하여 오류를 제안한다.

이 방법은 문제와 답안에 매우 의존적인 방법이다. 그러나 문제가 다소 쉬운 영작문이거나 답안의 내용이 매우 제한적일 수 있는 문제인 경우, 그리고 수험생의 영작문 수준이 높지 않은 경우에는 효과적으로 사용될 수 있다.

```
01  function GuessSpellingError (W0, answer_word_list)
02  {
03    w_candidates = []
04    if (W0 != UnknownWord)
05      AND (W0 not in answer_word_list)
06      AND ((level(W0) not in [Elem, Midd])
07      OR (freq(W0) < Threshold))
08    {
09      w_meta = get_metaphone(W0)
10      for (e, e_freq) in answer_word_list:
11        edit_distance = calc_edit_distance(W0, e)
12        e_meta = get_metaphone(e)
13        if ((w_meta ∩ e_meta) != ∅)
14          OR (edit_distance <= 1):
15          w_candidates += [e, e_freq, edit_distance]
16        if w_candidates != []
17          W1 = best(sort(w_candidates))
18          //edit_distance와 e_freq를 고려하여 정렬/선택
19                return (W0, 'PSeudoSpellError', W1)
20    }
21  }
```

Fig. 1. Error correction algorithm using correct answer words

Fig 1의 알고리즘 09, 12줄의 get_metaphone() 함수는 주어진 영어 단어에 대한 발음 정보를 리턴해 주는 함수로써 Double Metaphone 알고리즘을 사용하여 구현하였다. 단어 w와 발음이 유사하거나(줄 번호 12) 철자가 유사한(줄 번호 11) 단어 e가 *answer_word_list*에 존재할 때, w를 e의 철자 오류로 간주한다.

Table 1. Two collections of confusion sets

Confusion sets	Examples
(A) 423 sets	{accept, except} {advice, advise} {allow, aloud} {assure, ensure, insure} {along, a long} {morning, mourning} {praise, prays, preys} {plain, plane} {poll, pole} {root, rout, route} {wrote, rote} {see, sea} {seam, seem} {it's, its} {steel, steal} {some, sum} {vain, vane} {to, too, two} {very, vary} {wait, weight} {wail, whale} {you're, your, yore} {who's, whose} {which, witch} {weather, whether}...
(B) 3,873 sets	{excise, excuse} {extend, extent} {extraction, extrication} {expand, expend} {expert, export} {action, auction} {whale, while, whole, wale} {aluminate, illuminate} {alien, align} {allusion, illusion} {adit, audit, edit} {fool, foul, fowl} {failing, falling, filling} {hear, here, hero} {hair, hear, heir} {crate, create} {cast, caste} {leach, leech} {manage, mange} {miner, minor} {moral, morale, morel} {morning, mourning} {massage, message} {mask, musk} {ballet, ballot} {soar, sour} {cite, site} {tanker, thinker, tinker} {through, thorough}...

3.2 미리 정의된 혼동집합을 이용한 단어 오류 추정

혼동집합(confusing set)은 의미나 형태, 발음으로 인하여 자주 혼동되어 사용되는 단어 집합이다[3]. 혼동집합에 있는 단어가 문장에서 발견된 경우 해당집합에 있는 다른 혼동단어에 대한 오류 가능성을 검사한다. 주변 문맥 정보를 활용하여 혼동집합에 있는 단어 중, 해당 문맥에 가장 적합한 단어를 결정한다. 이때는 대량의 코퍼스에서 학습한 n-gram 정보나 연어정보(collocation)와 같은 문맥 정보를 활용한다.

본 연구에서는 2개의 서로 다른 혼동집합 모음을 사용하였다. 첫 번째 혼동집합 모음(A)는 기존의 여러 문헌에서 언급한 혼동 단어들을 수동으로 모은 것으로 모두 423개의 집합으로 구성되어 있다. 그 중 일부를 Table 1에 제시하였다.

두 번째 혼동집합 모음(B)는 영어 사전으로부터 자동으로 구축하였다. 발음과 철자가 유사한 혼동을 만들기 위해 사전의 모든 단어 쌍 (a, b)에 대해 두 단어 사이의 편집거리가 1 이하이며 Double Metaphone 알고리즘을 수행하여 두 단어의 발음이 유사할 때, 이를 혼동집합으로 만들었다. 이렇게 만들어진 혼동집합의 크기는 3,873개이다. 혼동집합 모음(A)와 (B)는 서로 253개를 공유했다. 모음(A) 중 자동으로 생성하지 못한 혼동집합은 주로 사전에 존재하지 않는 합성형태이거나 (예 : you're vs. your, all ready vs.

Table 2. Error detection-correction rules according to POS ambiguities

	POS ambiguity of Wi	Condition	Rules of generating candidates	Example sentences → set of error-correction candidates
1	{NN, VB}	be + Wi/NN	Wi/VB → VBG | VBN	The students had been scold by their teacher. → {scold/NN, scolded/VBN, scolding/VBG}
2	{NN, VB}	have + Wi/NN	Wi/VB → VBN	He has show maps to driver. → {show/NN, shown/VBN}
3	{NN, VB}	singular noun + Wi/NN	Wi/VB → VBZ | VBD	We know that the factory produce filters. → {produce/NN, produces/VBZ, produced/VBD }
4	{JJ, VB}	be + Wi/JJ	Wi/VB → VBN | VBG	The road is separate by the river. → {separate/JJ, separating/VBG, separated/VBN}
5	{JJ, NN}	NN|NNS|PN + Wi/JJ	Wi/NN → NNS	They have many relative to visit. → {relative/JJ, relatives/NNS} *They have several minute to think over. → {minute/JJ, minutes/NNS}
6	{IN, VB}	NN + Wi/IN	Wi/VB → VBZ | VBD	The baby like nanny dogs. → {like/IN, likes/VBZ, liked/VBD}

already) 의미를 함께 고려해야 하는 경우였다 (예 : healthy vs. healthful, childish vs. immature, innocent vs. childlike).

혼동집합 모음(A)는 기존의 영어권 연구에서 수집한 결과이다. 본 연구의 대상은 영어를 제2외국어로 사용하는 초/중급 작문수준의 학생들이 작성한 문서이다 보니 혼동집합 모음(A)로는 처리가 안 되는 경우가 발생할 수 있다. 그렇기 때문에 자동으로 혼동집합 모음(B)을 만들어 적용해 보고자 한다. 또한 수동으로 구축된 모음(A)에는 주로 혼동단어의 기본형에 대한 혼동집합만이 (예 : affect vs. effect) 존재하는 반면, 자동으로 구축한 모음(B)에는 기본형 이외의 활용형도 모두 포함되어 있어 (예 : affective vs. effective, affectation vs. effectuation) 실제 적용에 도움이 될 것이다.

혼동집합 중에서 문맥에 가장 적절한 단어를 결정하는 방법은 베이지안 분류기를 이용한다. 혼동단어가 사용된 문장에서 혼동단어와 혼동단어 양쪽 옆 2단어씩과의 연어정보를 분류기의 속성정보로 사용하였다.

혼동집합에 $\{W_1, W_2\}$가 있고, 입력 문장 S에서 혼동집합의 단어 W_1이 발견되었을 때, W_1을 중심으로 양쪽 옆 문맥정보 "C_{-2}, C_{-1}, W_1, C_{+1}, C_{+2}"와의 collocation 정보를 이용하여 혼동집합 중 문맥에 가장 적합한 단어를 결정한다. 즉, $P(W_1|C_{-2}, C_{-1}, C_{+1}, C_{+2})$와 $P(W_2|C_{-2}, C_{-1}, C_{+1}, C_{+2})$ 중 그 확률 값이 높은 것으로 결정한다.

$$P(W_1|C_{-2}, C_{-1}, C_{+1}, C_{+2})$$
$$= P(W_1)\ P(C_{-2}, C_{-1}, C_{+1}, C_{+2}|W_1) \quad (1)$$
$$\simeq P(W_1)\ P(C_{-2}|W_1)\ P(C_{-1}|W_1)\ P(C_{+1}|W_1)\ P(C_{+2}|W_1) \quad (2)$$

값의 비교만 수행하면 되기 때문에 (Eq. 1)과 같이 바꾸어 쓸 수 있고, 문맥정보를 독립적으로 고려하여 (Eq. 2)와 같이 바꾸어 사용한다. 각 확률 값은 대량의 코퍼스로부터 학습한다.

3.3 문맥을 고려한 문법 수준의 단어 오류

일반적으로 구문 오류 검사기는 입력 문장의 품사 정보(POS)만을 이용하여 구문 분석을 수행한다. 구문 분석기가 사용하는 규칙의 형태가 비어휘화된(non-lexicalized) 문맥자유문법(context free grammar)을 사용하기 때문에 각 단어가 갖는 특성을 충분히 반영하지 못하는 경우가 허다하다. 그러다 보니 입력문장의 어휘와 의미를 고려했을 때는 문법오류가 명백한 경우라도 구문 분석기는 정문으로 분석해 내는 경우가 많다. 특히 입력 문장 단어에 품사 중의성이 있는 경우에는 그런 경향이 가중된다. 예를 들어 다음과 같은 예제 문장을 보자. (예 3)의 'scold'는 명사(NN)와 동사원형(VB) 두 개의 품사를 가지며 (예 3)에서는 'scold'를 'scolded'로 작성했어야 하는 구문 오류를 갖고 있는 문장이다. 그러나 'scold'가 동사원형 이외에 명사 품사를 갖고 있으며 (예 3)에서는 명사로 분석되면서 구문 오류가 검출되지 못한다.

(예 3) The students had been scold by their teacher.

본 연구에서는 일반적인 구문 오류 중 입력단어의 품사 중의성으로 인해 일반적인 구문 분석기로는 인식할 수 없는 단어 수준의 구문 오류만을 다룬다. 그렇기 때문에 오류 수정 후보는 주로 입력단어에 대한 어형변화(inflection)를 통해 얻을 수 있다.

Table 2는 입력 단어 W_i가 가질 수 있는 품사 중의성에 따라 오류를 추정해 봐야 하는 주변 조건과 그 조건이 만족했을 때 어떻게 어형 변화를 시켜 오류 수정 후보 단어를

Table 3. Two smoothing methods

Smoothing with lexical bigram	Smoothing with POS
L3: $P(W_i\|W_{i-2}W_{i-1}) \times P(W_{i+1}\|W_{i-1}W_i) \times P(W_{i+2}\|W_iW_{i+1})$	L3: $P(W_i\|W_{i-2}W_{i-1}) \times P(W_{i+1}\|W_{i-1}W_i) \times P(W_{i+2}\|W_iW_{i+1})$
L2: $P(W_i\|W_{i-1}) \times P(W_{i+1}\|W_i)$	L2P1: $[\ P(T_i\|W_{i-2}W_{i-1})+P(W_i\|T_{i-2}W_{i-1})+P(W_i\|W_{i-2}T_{i-1})\ ] \times$ $[\ P(T_{i+1}\|W_{i-1}W_i)+P(W_{i+1}\|T_{i-1}W_i)+P(W_{i+1}\|W_{i-1}T_i)\ ] \times$ $[\ P(T_{i+2}\|W_iW_{i+1})+P(W_{i+2}\|T_iW_{i+1})+P(W_{i+2}\|W_iT_{i+1})\ ]$
	L1P2: $[\ P(T_i\|T_{i-2}W_{i-1})+P(W_i\|T_{i-2}T_{i-1})+P(T_i\|W_{i-2}T_{i-1})\ ] \times$ $[\ P(T_{i+1}\|T_{i-1}W_i)+P(W_{i+1}\|T_{i-1}T_i)+P(T_{i+1}\|W_{i-1}T_i)\ ] \times$ $[\ P(T_{i+2}\|T_iW_{i+1})+P(W_{i+2}\|T_iT_{i+1})+P(T_{i+2}\|W_iT_{i+1})\ ]$
	P3: $[\ P(T_i\|T_{i-2}T_{i-1})+P(T_i\|T_{i-2}T_{i-1})+P(T_i\|T_{i-2}T_{i-1})\ ] \times$ $[\ P(T_{i+1}\|T_{i-1}T_i)+P(T_{i+1}\|T_{i-1}T_i)+P(T_{i+1}\|T_{i-1}T_i)\ ] \times$ $[\ P(T_{i+2}\|T_iT_{i+1})+P(T_{i+2}\|T_iT_{i+1})+P(T_{i+2}\|T_iT_{i+1})\ ]$

생성할 것인가를 알려주는 규칙과 예제 문장이다. 기본적으로 Table 2의 규칙이 적용되기 위한 입력 문장은 일반 구문 분석기로 구문 오류가 없다고 판정된 것들이다.

문장 (예 3)에서 대상 단어 W_i는 'scold'가 되며 Table 2의 (1) 경우에 해당한다. 이때 후보 단어 생성 규칙은 W_i의 원형동사를 -ing가 붙은 형태(VBG[2])와 과거완료형태(VBN)로 생성하라는 것이다. 이와 같이 생성된 오류 후보 단어 집합은 원래 입력 단어를 포함하여 {'scold/NN', 'scolding/VBG', 'scolded/VBN'} 가 된다. 본 연구에서는 단어 n-gram을 이용하여 오류 후보 중 어떤 단어를 최종 수정 단어로 선택할지 결정한다.

주어진 입력 문장 S에 대하여 S에서의 대상 단어 W_i를 오류 후보 단어 집합의 단어들로 대치한 S'를 구하고 S와 S'의 확률 값을 계산하여 최댓값을 주는 오류 후보 단어를 구한다. 이 단어가 원래 입력 단어와 동일하지 않은 경우 오류 발생을 추정한다. 이를 형식적으로 기술해 보면 다음과 같다.

입력 문장 S = $W_1W_2...W_{i-2}W_{i-1}\mathbf{W_i}W_{i+1}W_{i+2}...W_n$에서 W_i가 오류 추정의 대상 단어이다. 문장 S의 확률 값을 트라이그램 모델을 적용하여 계산해 보면 (Eq. 3)을 구할 수 있다.

$$\begin{aligned} &P(S) \\ &\approx P(W_1)P(W_2|W_1)P(W_3|W_1W_2)...P(W_i|W_{i-2}W_{i-1})\times \\ &P(W_{i+1}|W_{i-1}W_i)P(W_{i+2}|W_iW_{i+1})...P(W_n|W_{n-2},W_{n-1}) \end{aligned} \quad (3)$$

오류 추정 대상 단어는 W_i이다. W_i의 오류 후보 단어 집합을 $W_i^{0..m}$로 표현해 보자. W_i^0은 원래 단어이며 $W_i^{1..m}$이 m개의 오류후보추정단어이다. 우리가 해야 할 작업은 입력 문장의 i-th 자리에 $W_i^{0..m}$ 후보 단어 중, 문장의 확률값 P(S)를 최대로 하는 단어를 찾는 것이다. 찾아진 단어가 W_i^0이 아닌 경우 해당 단어에 활용(inflection) 오류가 발생했다고 추정할 수 있다.

(m+1)개 문장 확률값을 계산할 때, i-th 위치의 단어를 제외하고 나머지 단어는 모두 동일하다. 그렇기 때문에 값의 비교를 위해서는 다음 (Eq. 4)의 세 트라이그램의 확률 곱만 비교해 보면 된다.

$$P(W_i|W_{i-2}W_{i-1}) \times P(W_{i+1}|W_{i-1}W_i) \times P(W_{i+2}|W_iW_{i+1}) \quad (4)$$

(m+1)개의 후보 단어에 대하여 (Eq. 4)를 계산하여 그 확률 값이 최대가 되는 단어를 최종 수정 단어로 선택하면 된다.

학습 데이터에서 트라이그램의 확률 값이 없을 때 (Eq. 4)의 값이 0이 될 수 있다. (m+1)개의 (Eq. 4) 값이 모두 0일 때 어떤 단어가 가장 적절한 형태인지 판단할 수가 없다. 본 연구에서는 이런 경우 데이터 부족에 대한 평탄화(smoothing) 방법으로 Table 3과 같은 두 가지 방법을 수행해 보았다.

첫 번째 방법은 어휘정보에 대한 트라이그램 정보가 없을 때 이를 바이그램 정보로 대체하는 것이다 (smoothing with lexical bigram). 두 번째 방법은 어휘정보에 대한 트라이그램 정보가 없을 때, 이를 품사정보로 대체하는 것이다 (smoothing with POS). 두 가지 방법을 수행하여 그 성능을 비교해 보고자 한다.

본 연구에서는 정확한 확률 값이 필요한 것이 아니고 후보단어 중 최대 확률 값을 갖는 단어만을 결정하면 되기 때문에 평탄화는 순차적으로 수행한다. 즉, '어휘 바이그램을 이용한 평탄화 방법'(smoothing with lexical bigram) 경우에는 L3 (Lexical 3-gram)으로 최대 확률을 갖는 단어가 결

2) 품사정보는 Penn treebank[8]의 품사체계를 따른다.

정되지 못하는 경우에만 L2 (Lexical 2-gram) 정보를 이용한다. '품사정보를 이용한 평탄화 방법'(smoothing with POS)에서는 L3 방법에서 최대 확률을 갖는 단어가 결정되지 못하는 경우에 3단어 중 하나를 품사(POS) 정보로 대치한 L2P1 (Lexical-2 POS-1) 모델을 사용하고, L2P1에서도 결정이 되지 못하는 경우에만 L1P2 (Lexical-1 POS-2)를 사용하고 마지막으로 P3(POS-3)을 사용한다.

품사정보를 이용한 평탄화 방법에서는 (Eq. 4)의 값이 모두 존재하지 않을 경우, 한 단어를 품사 정보로 대치하여 (Eq. 5)과 같이 계산한다.

$$P(W_i|W_{i-2}W_{i-1}) \approx \lambda 1P(T_i|W_{i-2}W_{i-1}) + \lambda 2P(W_i|T_{i-2}W_{i-1}) + \lambda 3P(W_i|W_{i-2}T_{i-1}) \quad (5)$$

(Eq. 5)에서 세 항의 중요도를 모두 동일하게 간주하여 λi는 각각 1/3으로 설정한다. (Eq. 5)을 각각의 항에 모두 적용하면 비교 대상으로 계산해야 하는 (Eq. 4)은 다음과 같이 수정하여 계산될 수 있다. ((m+1)개의 값만 서로 비교하면 되기 때문에 (Eq. 6)에서 λi 값은 제외하였다.)

$$[P(T_i|W_{i-2}W_{i-1})+P(W_i|T_{i-2}W_{i-1})+P(W_i|W_{i-2}T_{i-1})] \times [P(T_{i+1}|W_{i-1}W_i)+P(W_{i+1}|T_{i-1}W_i)+P(W_{i+1}|W_{i-1}T_i)] \times [P(T_{i+2}|W_iW_{i+1})+P(W_{i+2}|T_iW_{i+1})+P(W_{i+2}|W_iT_{i+1})] \quad (6)$$

(Eq. 6)의 경우에도 모든 대상 단어의 확률 값이 0일 경우, 세 단어 중, 두 단어를 품사로 대치하고(L1P2) 그렇게 한 경우에도 확률 값이 모두 0인 경우에는 세 단어를 모두 품사로 대치한다(P3).

(예 3)의 입력 문장 "The students had been scold by their teacher."에서 'scold'는 Table 2의 (1)번 조건에 해당하며 이때 오류 후보 대상 단어 집합은 {scold/NN, scolded/VBN, scolding/VBG}가 된다. 각각의 오류 후보 단어에 대해 (Eq. 4)를 계산한 것이 (Eq. 7)이다.

$$P(\text{scold}|\text{had been}) \times P(\text{by}|\text{been scold}) \times P(\text{their}|\text{scold by}) \quad (7\text{-}1)$$

$$P(\text{scolded}|\text{had been}) \times P(\text{by}|\text{been scolded}) \times P(\text{their}|\text{scolded by}) \quad (7\text{-}2)$$

$$P(\text{scolding}|\text{had been}) \times P(\text{by}|\text{been scolding}) \times P(\text{their}|\text{scolding by}) \quad (7\text{-}3)$$

(Eq. 7-1)~(Eq. 7-3) 중에서 최댓값을 갖는 경우가 (Eq. 7-2)로 판별되었을 경우, 입력 문장의 단어 'scold'에 어형 변화와 관련된 단어 오류가 발생했다고 판단하고 이때 입력 문장은 "They had been scolded by their teacher."로 수정할 수 있다.

4. 실험 및 평가

본 논문에서 제시한 방법에 대한 평가를 위해 다음의 환경에서 실험을 수행하였다.

학습 데이터로는 LDC Web 1T 5-gram[9] 중, 어휘 트라이그램 정보를 사용하여 학습하였다. 미리 정의된 혼동집합 단어오류추정에서 혼동집합(A)에 학습에 사용한 전체 트라이그램 개수는 16,317,462이며, 혼동집합(B)에서는 65,891,295이다. 문맥을 고려한 문법수준의 단어오류 추정에서 학습에 사용한 전체 트라이그램의 개수는 248,575,522이다. 품사 정보가 포함된 확률 값을 추정하기 위해서는 Penn treebank이 품사 태깅된 코퍼스[8]를 사용하였다. 품사 정보가 포함된 트라이그램의 개수는 총 4,322,555개이다.

Table 4. Collection of student answers for evaluation (test sets)

Question Categories	Number of answers	Sentences /answer	Total number of sentences	Average words (word/ sentence)
(1) single sentence writing	14,703	1.0	14,703	6.14
(2) 4-sentence completion	1,206	3.97	4,784	10.02
(3) paragraph writing (2~3 sents)	3,324	1.97	6,564	10.01
(4) paragraph writing (> 40 words)	147	4.25	625	10.95
(5) Inferencing and paragraph writing (> 40 words)	164	4.27	701	12.87
(6) paragraph writing (> 60 words)	2,139	5.63	12,037	12.80
(7) paragraph writing (> 80 words)	210	9.43	1,981	11.22
(8) Essay writing	85	15.16	1,289	14.61
TOTAL	21,978		**42,684**	11.08

평가를 위하여 수집한 학생들의 영작문 답안은 Table 4와 같다. 본 실험에서 사용한 영작문 문항은 모두 8 개이다. 8개 문제는 서로 다른 수준의 영작문 문항으로 '평균 단어 수' 값을 보면 어느 정도 수준의 영작문임을 가늠할 수 있다. 본 실험에서 평가 대상으로 사용한 총 문장 수는 42,684개이다.

3장에서 설명한 세 가지의 문맥을 고려한 단어 오류 추정 방법은 각각 독립적으로 수행되도록 구현되어 있다. 그렇기 기 때문에 다음 세 절에서 설명하는 실험 결과에는 동일한 오류가 중복적으로 검출될 수 있다.

4.1 정답 단어를 이용한 철자 오류 추정

정답 단어를 이용한 철자 오류 추정을 하기 위해서는 사람에 의해 수동 채점되어 점수가 부여되어 있는 답안이 있어야 한다. Table 4의 답안 중, 수동 채점된 점수가 있는 답안은 (2)와 (3)번뿐이다. 그렇기 때문에 (2)와 (3)의 총 10,348 문장에 대해서만 실험을 수행하였다. (2)번과 (3)번 문항 각각에 대해서는 세부적으로 4개의 서로 다른 문제가 있다. 각 문제별로 고득점을 받은 답안으로부터 사용단어들을 추출하고 빈도수 2 이상 되는 단어들을 수집하여 정답 단어 집합으로 사용하였다. (문항에 대한 자세한 설명은 [2]을 참조 바람.)

Table 5. Results of error correction using correct answer words.

	Error	Freq	Examples
(1)	Detection (correct) Correction (correct)	25	He **ware** striped pants,so he look like taller. → wear If you want to look better, how **abut** having a haircut? → about It make you more hansome and **confidential** → confidence talking about her **projet** → project ordering coffer to a staff → coffee **clan** customer's hair → clean Exercises will make you **simmer**. → slimmer
(2)	Detection (correct) Correction (-)	23	Im lee woo min i say woo you say min **woo**!! → why hey taxi gay do you go keng **bog** gong? → back
(3)	Detection (correct) Correction (wrong)	1	becouse if your pants longlest your **lag** very taller → like
(4)	Detection (wrong) Correction (wrong)	6	I think you are **overweight.** → everyday Maybe, I have fever and **dizzy**. → thus putting **cube** shaped sugar in his coffee → cup treating a boy who has **scar** on his right knee → score
	TOTAL	55	

Table 5는 정답 단어를 이용한 철자 오류 추정 실험 결과이다. 총 10,348 문장에 대해 실험을 수행하여 55번의 경우가 발생했는데 이때, 7개 경우((3)과 (4))가 오류였다. 오류 예제에서 보듯이 입력 문장에는 오류 추정 단어 이외에도 철자오류, 구문오류가 다수 보인다. (2)번의 경우, 입력 문장 자체에 오류가 너무 많아서 오류 수정이 의미가 없는 경우 (즉 정답이 없는 경우)이다. (4)번은 오류가 아닌 것을 인식하여 잘못 수정한 경우이다. 예제에서 보는 것과 같이 다수의 답안에 나타나지 않은 중급 수준의 단어를 사용한 경우가 주로 해당된다. 그렇기 때문에 상대적으로 영작문 능력이 우수한 학생의 답안에서 오류를 잘못 인식하게 되는 경우가 발생할 수 있다.

이와 같은 약점을 갖고 있는 방법임에도 불구하고 본 연구에서 이 방법을 시도한 이유는 초급 작문 수준을 가진 학생의 답안에서부터 철자 오류를 효과적으로 인식하고 오류 수정 역시 효과적으로 수행할 수 있기 때문이다. 이 방법을 효과적으로 적용하기 위해서는 우선 처리하고자 하는 입력 문장의 작문 완성도를 측정하고 초급 수준의 작문인 경우에만 적용하면 된다. 입력 문장의 작문 완성도는 문장의 복잡도(perplexity)값을 활용할 수 있다. 또한 문항의 특성상 제한적인 답안만이 가능한 경우 역시 매우 효과적으로 오류 수정이 가능하다. 그렇기 때문에 문항과 답안의 특성을 분석하여 적용하는 것이 중요하다.

4.2 미리 정의된 혼동집합을 이용한 단어 오류 추정

Table 6은 두 개의 혼동집합을 이용하여 혼동단어에 대한 오류 수정의 정확도를 보여준다. 집합 (B)는 (A)에 비해 5배 이상 적용되었지만 정확도는 7% 이상 하락하였다.

Table 6. Results of error correction using confusion sets

Confusing set collections	Applied frequency	Correct	Wrong	Accuracy
(A) 423 sets	1,942	1,725	217	88.83
(B) 3,873 sets	10,422	8,520	1,902	81.75

Table 7은 혼동집합 단어에 대한 오류 수정 예제이다. 예제에서 보는 것과 같이 주어진 입력 문장은 혼동단어 이외에도 다수의 오류가 있을 수 있다. 혼동집합 모음(A)에서 오류 수정이 잘못된 경우를 살펴보면 크기 2인 양쪽 문맥정보에 정답을 추론할 수 있는 충분한 정보가 부족한 경우가 많았다. 또한 문장의 작성 의도가 불분명하여 정확히 문장

Table 7. Examples of correct and wrong error-corrections

	Error correction	Examples
(A) 423 sets	Correct	but **it's** value is continued forever. → its he **meat** a person. → meet go **strait** and turn left. → straight Because **it's** weather is very snowy. → its It's **to** heavy snow. → too oh **whether** so terrible we cannat climb m. → weather
	Wrong	Then **write** the place and show the paper to taxi driver. → right I **sit** down and reading book. → set Many **rain** in the June. → reign I **read** Korea history book. → red People **buy** more and more that price is cheaper. → by
(B) 3873 sets	Correct	Would **batter** go home now. → better He **head** no lift above could longer it. → had young **ago** learn to dance very very fun. → age He **rid** on the taxi. → rode I **hoop** see you soon. → hope I **thank** fast food is good food. → think a **live** with injury on my face is horro... → life The **men** is sitting in the texi. → man Then **shell** we go to the library? → shall they **went** to go to see a movie. → want What **kink** of food do you want to eat. → kind
	Wrong	Using **nick** name kills some people → neck We **met** at gym after we had a dinner. → mat I **bake** because it makes me feel good when I create the recipe... → bike The **last** reason is that you can get fast food anytime. → least Many **pool** people can eat this food for cheap price. → pull The **sun** was just far enough in the west... → soon So **past** food is bad. → best My **best** pride habit is the helping an stranger. → past

의 의미 파악이 어렵거나 부적합한 단어의 사용이나 부적절한 단어 활용형이 사용되어 혼동집합 중에서 정확한 수정 단어를 결정하는 데 영향을 미치는 경우가 있었다.

자동으로 생성한 혼동집합 모음(B)에는 (hoop→hope), (thank→think), (shell→shall)와 같은 영어권에서는 일반적으로 혼동집합으로 간주되지 않는 것들이 존재하여 이와 같은 오류도 수정할 수 있었다. 반면에 오류가 아닌 것을 잘못 수정한 경우가 18.25%였다. 예제에서 (pool→pull)은 자동으로 생성한 혼동집합 {pool, pull} 때문에 발생하였다. 한국인들에게는 혼동집합 {pool, fool}이 훨씬 유용할 수 있다. 이와 같은 혼동집합을 생성해 내기 위해서는 발음 유사성을 결정하는 Double Metaphone 알고리즘에서 한국인들이 잘 구분하지 못하는 발음 정보를 고려해 주도록 수정하는 것이 필요하겠다.

Table 8. Examples of error correction according to error detection-correction rules in Table 2

Condition	Examples
1	you are **change** your hair stylie. → changing Sylvia's mother is **cook** the food. → cooking I am **wait** for tonight concert. → waiting Who is **call** me? → calling If I were **study** hard, I could pass the exam. → studying
2	I sometimes heard Mary had **play** piano. → played I have **study** for seven years. → studied Some people have **think** "I'm so ugly/JJ",... → thought
3	Recently the world **experience** a lot of change → experienced The price is lower, the person **buy** more → buys He **talk** that he maybe know everything. → talked
4	I am **clean** a room in free time. → cleaning
5	Firsr of all, all people have human **right** that no one can violate. → rights
6	my father **like** to eat. → likes

4.3 실험결과: 문맥을 고려한 문법 수준의 단어 오류

Table 8 예제에서 보는 바와 같이 입력 문장은 다수의 오류를 포함하고 있는 경우가 많다. 첫 번째 예제의 경우 철자 오류('stylie')를 포함하고 있으며 그 외에도 의미적이나 관용적으로 부자연스러운 문장들이 다수 발견된다. Table 8에서 보는 오류의 대부분은 입력 문장이 다른 기타 오류가 없는 정문에 가까운 경우라면 구문 분석 과정을 통해서 충분히 걸러낼 수 있는 것들이다. 그러나 본 연구에서 다루는 대상은 초/중급 작문 수준을 가진 학생들이 작성한 문장이 대부분이다. 그러다보니 오류를 고려하여 설계된 구문 분석기라 하더라도 파싱 성공률이 매우 낮을 수밖에 없다. 즉, 대부분의 문장이 구문 분석 단계에서 아예 실패를 하게 되므로 구문 분석을 통해 오류를 인식/수정하는 것은 불가능하다. 본 연구는 [2]의 시스템을 기본으로 하고 있다. [2]에서는 이와 같은 문제를 해결하기 위하여 전체 구문 분석을 수행하지 않고 부분만을 따로 분석할 수 있는 유한 상태 오토마타를 이용하여 구문 오류를 검사한다. 여기에서 다루고 있는 문제는 부분 구문 분석을 하더라도 쉽게 검출되지 않는 오류들이다.

전체 평가데이터 42,684 문장 중, Table 2의 조건에 맞는

Table 9. Accuracy of error correction according to smoothing methods

	Smoothing with lexical bigram	Smoothing with POS
	L3: 1098 / 1373 = 0.7997	L3: 1098 / 1373 = 0.7997
	L2: 1454 / 2051 = 0.7089	L2P1: 203 / 289 = 0.7024
		L1P2: 1016 / 1594 = 0.6373
		P3: 255 / 322 = 0.7919
Precision	2552 / 3424 = **0.7453**	2572 / 3578 = 0.7188
Recall	2552 / 3818 = 0.6684	2572 / 3818 = 0.6736
F1	**0.7048**	**0.6955**

오류를 갖고 있는 문장이 3,818 개였다. Table 9에서 보는 바와 같이 오류 수정은 약 74.5%의 정확도까지 얻을 수 있었다.

Table 9을 보면 어휘 바이그램을 이용한 평탄화 방법이 조금 더 좋은 성능을 보였다. 본 연구에서 다루는 오류가 대부분 품사 정보만으로도 해결되는 오류이기 때문에 품사 정보를 이용한 평탄화 방법이 더 좋은 성능을 발휘할 것으로 예상되었으나 결과는 반대였다. 그 이유는 두 가지로 추정된다. 첫째, 입력 문장에 다른 오류가 많다 보니 입력 문장의 품사 태깅 결과가 부정확한 경우가 많았다. 둘째, 학습데이터로 사용된 어휘 n-gram 코퍼스는 웹에서 추출된 것이기 때문에 상당수의 오류도 함께 학습이 이루어진 반면, 품사 정보 추출에 사용된 코퍼스는 정문만으로 이루어져 있어 오류가 있는 입력 문장에 대해서 좋은 성능을 발휘하지 못했다. 품사 태깅된 학습 코퍼스의 양이 어휘 n-gram 코퍼스에 비해 훨씬 적었기 때문에 품사정보를 이용한 평탄화 방법의 재현율(recall) 값을 낮추는 요인으로도 작용하였다.

기존의 영어권에서 수행된 연구와의 비교를 위해 동일한 문제를 다룬 연구를 찾아보았으나 동일한 문제를 다룬 연구는 찾을 수 없었다. 가장 유사한 문제로 단어 수준이면서 동시에 문법적 성격을 동시에 갖고 있는 전치사를 정확히 선택하는 문제와 비교를 수행하였다. Table 10의 (1)~(3)은 영어권에서 수행한 전치사 결정 문제이다. 모든 문제는 정확도로만 평가되었다.

(1)은 불어 전치사에서 자주 혼동되어 쓰이는 혼동집합을 만들고 비불어권 사용자가 작성한 문장에 대해 이 혼동집합 중 가장 적합한 것을 선택하였다. 본래 문장에서 혼동집합의 전치사들을 하나씩 대치한 문장을 만들어 내고 이를 웹 검색을 통해 그 검색결과가 가장 높은 단어를 수정후보로 선택하였다. 약 69.9%의 성능을 얻을 수 있었다.

Table 10. Comparison with the related studies

	Related studies	Accuracy
(1)	Objective: choosing a correct preposition out of 6 most frequently used French prepositions by using confusion set [5]. Training data: frequency extracted from web search Testing data: 133 sentences with errors written by second language learners	69.9%
(2)	Objective: choosing a correct preposition out of 34 most frequently used English prepositions [4]. Training data: 1M prepositions with context from New York Time (NYT); using lexical/POS n-gram language model Testing data: 10K unseen test set from NYT (error-free sentences)	75.4%
(3)	Objective: Detecting and choosing a correct preposition out of 12 most frequently used English prepositions [6]. Training data: 2.5M sentences from various sources; lexical/POS n-gram language model (native text) Testing data1: 30% of Training data (trained by 70% of training data) Testing data2: 12,755 English sentences written by East Asian students	77.6% (Testing data1) 41.3% (77.7%)[3] (Testing data2)
(4)	Objective: correcting word errors such as [6] verb-related (ex: I have studying/studied?) adjective-related (ex: I am interested/interesting?) noun-related (ex: door of bus vs. bus door) Training data: the same as (3) Testing data: 12,755 sentences written by East Asian students	55.1% (88.0%)

(2)에서는 34개의 전치사 중 하나를 선택하는 다소 어려운 문제임에도 불구하고 75.4%의 정확도를 얻을 수 있었는데, 높은 성능을 얻을 수 있었던 요인 중 하나는 오류가 없는 정문으로 학습을 시키고 학습데이터와 유사한 정문으로만 구성된 문장에 대해 평가를 수행하였기 때문이다. (3)의 연구도 (2)와 유사한데 여기서는 12개의 전치사만을 대상으로 삼았다. (3)의 결과에서 주목해야 할 부분은 영어권 사용자가 작성한 평가 셋에서는 77.6%의 정확도를 얻은 반면, 이를 비영어권 사용자가 작성한 평가 셋에 적용했을 때에는 41.3%로 정확도가 많이 낮아졌다. (3)과 (4)는 동일한 연구인데, 이 중 (4)는 전치사 문제가 아닌 우리가 다루었던 문제와 가장 유사한 문제를 다룬 연구 결과이다. 비영어권 사용자가 작성한 평가 셋에 대해 55.1%의 정확도

를 얻었고 괄호 안의 값은 맞았는지 틀렸는지 평가 내리기 어려운 경우들을 모두 맞은 것으로 간주하여 얻은 결과 값이다.

본 연구의 결과인 Table 9와 Table 10의 직접적인 비교는 어렵다. 그러나 대상언어를 모국어로 사용하지 않는 사람들이 작성한 문제를 다룬 (1), (3), (4)의 결과와 비교해 보았을 때, 본 연구의 결과가 뒤처지지 않음을 판단할 수 있었다. 특히 가장 유사한 문제를 다룬 (4)의 결과와 비교해 본다면 본 연구 결과가 좋은 성능을 보임을 알 수 있다.

5. 결 론

본 연구에서는 기존의 철자 오류 검사기나 구문 오류 인식기가 수행하지 못하는 종류의 오류를 정의하고, 이를 찾아내어 적절한 수정 후보를 제시하는 방법을 제안하였다. 즉, 단어 자체로만 평가했을 때는 오류가 없으나 문장의 문맥을 함께 고려했을 때는 적절한 단어 사용 또는 활용 형태가 아닌 오류들을 인식하는 것이다.

오류를 인지하고 수정하는 방법은 크게 세 가지로 나누었는데, 첫째는 영어 자동채점 시스템이라는 환경에서만 사용할 수 있는 제한적 방법이다. 둘째는 기존의 영어권에서 많이 사용하던 혼동집합을 활용하여 오류를 인식/수정하는 것이다. 세 번째는 품사 중의성으로 인해 기존의 구문 분석기는 오류로 인지하지 못하는 부류에 대한 오류로 이에 대한 인식 조건 및 오류 수정 후보를 생성하는 방법을 제안하였다. 세 가지 방법 모두 비영어권 사용자가 작성한 문장을 고려하여 제안되었다. 즉, 입력 문장에는 대상 단어 이외에도 다양한 오류가 함께 공존하는 경우가 대부분이며, 이러한 환경에서도 수행이 가능한 방법을 제안하였다.

한국어가 모국어이면서 초/중급 수준의 영작문 능력을 갖춘 수험생들이 작성한 영어 문장들을 평가 데이터로 구축하여 평가해 보았다. 기존의 영어권 연구 결과와 직접적인 비교는 어려우나, 가장 유사한 형태의 연구와 간접 비교를 통해 본 연구의 결과가 매우 고무적임을 알 수 있었다.

3) 괄호 안의 값은 오류를 인식은 했으나 제대로 수정후보를 생성하지 못한 경우나 문장의 다른 오류 때문에 수정후보를 맞게 선택하지 못한 경우, 오류 후보 중 어떤 것을 선택해도 상관없는 경우들을 모두 correct로 간주하여 평가한 것임.

References

[1] Xu Sun, Jianfeng Gao, Daniel Micol, and Chris Quirk, “Learning Phrase-Based Spelling Error Models from Clickthrough Data,” pp.266-274. The Association for Computer Linguistics, 2010.

[2] GyoungHo Lee and Kong Joo Lee, “Developing an Automated English Sentence Scoring System for Middle-school Level Writing Test by Using Machine Learning Techniques,” Journal of KISE, Vol.41, No.11, 2014.

[3] Andrew R. Golding, “A Bayesian hybrid method for context-sensitive spelling correction,” In Proceedings of the Third Workshop on Very Large Corpora, pp.39-53, 1995.

[4] Shane Bergsma, Dekang Lin, Randy Goebel, “Web-Scale N-gram Models for Lexical Disambiguation,” In Proceedings of the Twenty-First International Joint Conference on Artificial Intelligence (IJCAI-09), pp.1507-1512, 2009.

[5] Hermet, M., Désilets, A., Szpakowicz, S., “Using the Web as a Linguistic Resource to Automatically Correct Lexico-Syntactic Errors,” In The 6th Edition of the Language Resources and Evaluation Conference (LREC 08). May 28-30, 2008. Marrakech, Morocco.

[6] Michael Gamon, Claudia Leacock, Chris Brockett, Jianfeng Gao, and Alexander Klementiev, “Using Statistical Techniques and Web Search to Correct ESL Errors,” 2009, CALICO Journal, Vol.26, No.3, pp.491-511.

[7] Lawrence Phillips, “The Double Metaphone Search Algorithm”, C/C++ Users Journal, Vol.18, No.6, June, 2000.

[8] Mitchell P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini, Building a large annotated corpus of English: the penn treebank, Journal Computational Linguistics - Special issue on using large corpora: II archive Vol.19, Issue.2, June, 1993, Pp.313-330.

[9] https://catalog.ldc.upenn.edu/LDC2006T13

[10] A Winnow-Based Approach to Context-Sensitive Spelling Correction, Andrew R. Golding and Dan Roth, 1999.

[11] Kong Joo Lee and Sonwook Lee, “A comparison of grammatical error detection techniques for an automated English scoring system,” Journal of the Korean Society of Marine Engineering 01/2013; 37(7).

최 용 석

e-mail : yongseok.choi.92@gmail.com

2011년~현 재 충남대학교 정보통신공학과 학사과정

관심분야 : 자연언어처리

이 공 주

e-mail : kjoolee@cnu.ac.kr

1992년 서강대학교 전자계산학과(학사)

1994년 한국과학기술원 전산학과 (공학석사)

1998년 한국과학기술원 전산학과 (공학박사)

1998년~2003년 한국마이크로소프트(유) 연구원

2003년 이화여자대학교 컴퓨터학과 대우전임강사

2004년 경인여자대학 전산정보과 전임강사

2005년~현 재 충남대학교 정보통신공학과 교수

관심분야 : 자연언어처리, 기계번역, 정보검색, 정보추출