Autocorrection with Levenshtein Distance Algorithm to Spell Error in Legal Terms

아주대학교 AJOU UNIVERSITY

Sungho Gong[1] & *Jinwoo Shim*[1] & *Minho Hyeon*[1]
University of Ajou[1]

인문대학 College of Humanities

Introduction

본 연구는 법률 문서에서 발생하는 **오탈자**에 주목하며, 법률용어의 정확한 검색을 위해 Levenshtein-Distance 알고리즘을 제안한다.

법률문서는 법제처의 노력에도 불구하고 아직까지도 비전문가에게 어려운 언어, 표현을 사용해 이를 이해, 활용하는 데 어려움이 있으며, 전문가들도 판결문 작성 시에 실수로 인한 피해가 발생할 수 있다.

M.E.W Putra et al.(2015)를 비롯한 연구자들은 Levenshtein-Distance 알고리즘을 사용해 **한글 법률용어의 오탈자를 자동으로 교정**하고, 이를 통해 검색 서비스의 효율성을 향상시킬 수 있을 것으로 기대하고 있다. 사전 연구와는 달리 한글을 위 알고리즘에 적용, 법률용어 데이터를 사용해 오탈자를 자동으로 교정하고자 한다.

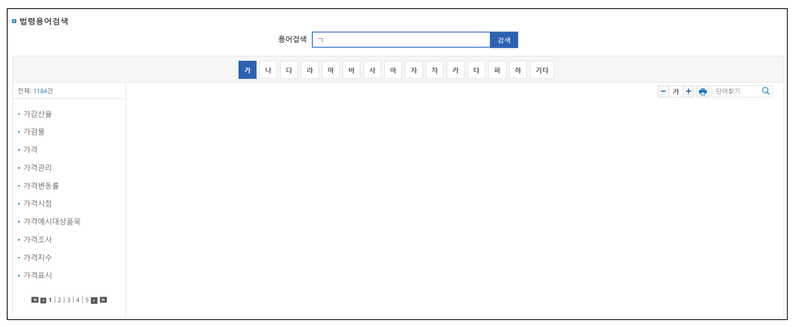
Corpus

① 말뭉치(Corpus) 수집

연구에서는 기본적인 단어의 오탈자 교정을 위해 국립국어원의 '**우리말샘**' 단어 사전을 활용하였다. 오른쪽의 사진과 같이 약 100만개의 단어가 포함된 24개의 JSON 파일을 Python을 이용하여 딕셔너리로 정제하고, "wordinfo" 안의 "word"만 추출하여 텍스트 파일로 저장했다.


```
{
 "channel": {
  "total": 50000,
  "title": "사전 검색",
  "description": "사전 검색 결과",
  "item": [
   {
    "wordinfo": {
     "word_unit": "어휘",
     "origin": "< German",
     "word": "게르만",
     "original_language_info": [
      {
       "original_language": "←Geruman",
       "language_type": "안 밝힘"
      }
     ],
     "word_type": "외래어"
    },
    "group_order": 1,
    "group_code": 46558,
    "link": "http://opendict.korean.go.kr/dictionary/view?sense_no=413165",
    "target_code": 413165,
    "senseinfo": {
     "definition": "게르만 어파에 속한 언어를 쓰는 민족. 백색 인종으로 키가 크고 금발이며 눈이 푸
     "cat_info": [
      {
       "cat": "고유명 일반"
      }
     ],
     "relation_info": [
      {
       "link": "http://opendict.korean.go.kr/dictionary/view?sense_no=523779",
```


법률 용어의 오탈자 수정을 위해서는 아래 사진인 한국법제연구원의 **법령용어검색서비스**를 활용하였으며, Python의 BeautifulSoup과 Selenium 패키지를 사용하여 단어를 추출하고 저장하였다. 총 6178개의 단어를 추출하여 텍스트 파일에 저장했다.



실제 판결문에서 사용되는 법률 용어를 추출하기 위해 '**대한민국 법원 종합 법률 정보**' 사이트의 880개의 화제 판례를 활용하였다. 이를 위해 KIWI 형태소 분석 패키지와 Python을 사용하여 판례의 핵심 내용을 형태소 분석하여 **Content Word**(내용어)를 추출하였다. 추출한 단어들은 중복을 제거하여 **텍스트 파일**에 저장되었으며, 총 101,3644개의 단어가 포함되어 용량은 14.8MB이다.

References

1. **참고문헌**
   Putra, M. E. W., & Suwardi, I. S. (2015). Structural off-line handwriting character recognition using approximate subgraphmatching and levenshtein distance. Procedia Computer Science, 59, 340-349.
   김지현, 이종서, 이명진, 김우주, 홍준석.(2012). 법령정보 검색을 위한 생활용어와 법률용어 간의 대응관계 탐색 방법론.지능정보연구,18(3),137-152.
2. **참고 사이트**
   『법령용어검색서비스』, https://www.klri.re.kr/kor/business/bizLawDicKeyword.do.
   『대한민국 법원 종합 법률 정보』, https://glaw.scourt.go.kr/wsjo/intesrch/sjo022.do.
   손성배, "[불친절한 법원은 무죄일까] 어려운 용어와 긴 문장… 성역이 된 법원의 언어", ,『경인일보』, 2021.11.29 http://www.kyeongin.com/main/view.php?key=20211123010004462.

Algorithm

최소편집거리(Minimum-Edit-Distance) 라고도 불리는 이 알고리즘은 두 개의 단어를 대상으로 단어1이 단어2로 바뀌기 위해서 몇 번의 작업이 필요한지를 계산하는 알고리즘이다. 편집의 종류로는 '**삽입**(Insertion)', '**삭제**(Deletion)', '**대체**(substitution)'가 있는데, 각각 과정마다 1점을 부여한다. 오른쪽 사진이 이 알고리즘에 대한 과정이다.

```
① matrix = []
  for i in range(len(str1)+1):
    row = []
    for j in range(len(str2)+1):
      row.append(0)
    matrix.append(row)
② for i in range(1, len(str1)+1):
    matrix[i][0] = i
  for j in range(1, len(str2)+1):
    matrix[0][j] = j
③ for i in range(1, len(str1)+1):
    for j in range(1, len(str2)+1):
      if str1[i-1] == str2[j-1]:
        matrix[i][j] = matrix[i-1][j-1]
      else:
        matrix[i][j] = min(matrix[i-1][j], matrix[i][j-1], matrix[i-1][j-1]) + 1
```

한국어는 영어와는 달리 자음과 모음으로 이루어진다. 그래서 Levenshtein 알고리즘에 직접 한글 단어를 입력하면 한 글자씩 편집되기 때문에 정확한 결과를 얻기 어렵다. 이를 보완하기 위해 python-jamo 패키지를 사용하여 자음과 모음을 분리한 후 Levenshtein-Distance를 계산하는 방법을 적용하였다. 이를 설명하기 위해 아래 사진에서 법률 용어인 "**흡수합병**"과 오탈자 "**흐ㅆㅜㅎ밥병**"의 Levenshtein-Distance를 예시로 사용하였다.

[진행과정]

① python-jamo 패키지를 이용해 두 개의 단어를 자음, 모음 단위로 분리한다.

② 분리된 두 단어의 길이는 11이므로 12X12 크기의 영행렬을 생성한다.

③ 첫 행과 첫 열이 각 0부터 11까지 채워진다.

④ 아래와 같은 과정이 이루어진다.

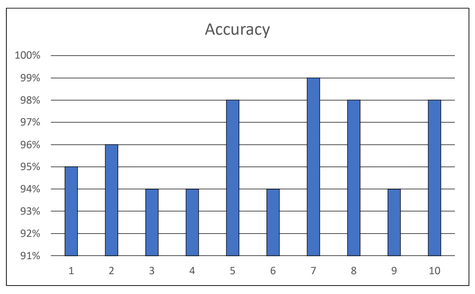
	0	ㅎ	ㅡ	ㅆ	ㅜ	ㅎ	ㅂ	ㅏ	ㅂ	ㅂ	ㅕ	ㅇ
0	0	1	2	3	4	5	6	7	8	9	10	11
ㅎ	1	0	1	2	3	4	5	6	7	8	9	10
ㅡ	2	1	0	1	2	3	4	5	6	7	8	9
ㅂ	3	2	1	1	2	3	3	4	5	6	7	8
ㅅ	4	3	2	2	2	3	4	4	5	6	7	8
ㅜ	5	4	3	3	2	3	4	5	5	6	7	8
ㅎ	6	5	4	4	3	2	3	4	5	6	7	8
ㅏ	7	6	5	5	4	3	3	3	4	5	6	7
ㅂ	8	7	6	6	5	4	3	4	3	4	5	6
ㅂ	9	8	7	7	6	5	4	4	4	3	4	5
ㅕ	10	9	8	8	7	6	5	5	5	4	3	4
ㅇ	11	10	9	9	8	7	6	6	6	5	4	3

1. ㅎㅡ**ㅆ**ㅜㅎㅂㅏㅂㅂㅕㅇ -> ㅎㅡ**ㅂ**ㅜㅎㅂㅏㅂㅂㅕㅇ(**대체**)
2. ㅎㅡㅂㅜㅎㅂㅏㅂㅂㅕㅇ -> ㅎㅡㅂ**ㅅ**ㅜㅎㅂㅏㅂㅂㅕㅇ(**삽입**)
3. ㅎㅡㅂㅅㅜㅎ**ㅂ**ㅏㅂㅂㅕㅇ -> ㅎㅡㅂㅅㅜㅎㅏㅂㅂㅕㅇ(**삭제**)

대체, 삽입, 삭제의 3단계를 거쳐 최소편집거리는 3이 나오게 된다.

Results

앞서 소개한 말뭉치와 알고리즘을 이용하여서 사용자에게 법률용어를 입력받으면 **자음과 모음으로 분리**한 후 텍스트 파일의 단어들과의 **Levenshtein-Distance를 비교**한다. 그 후 **편집거리가 제일 짧은 단어**를 사용자에게 출력해준다. 만약 텍스트 파일에 있는 옳은 단어를 입력했을 때는 입력받은 그대로 출력한다.



대한민국 법원 종합 법률 정보 사이트의 공개된 판례에서 Content Word를 추출하였고, 단어 100개씩 **총 10번의 테스트**를 진행했다. 위의 사진에 결과가 나와있고, 결과의 평균은 **96%**이다.

약 100만개의 단어가 있는 텍스트파일을 전부 확인하면서 알고리즘이 진행되기 때문에 시간이 조금 소모된다. 알고리즘 최적화를 통해 시간을 줄이는 것을 향후 과제로 삼고 있다.