단어 의미의 상호작용을 모델링하는 방법

2022 세계한국어 한마당

한국어와 기계학습(국어학회)

2022. 10. 7.

박진호(서울대 국문과)

목차

- 1. 문제제기
  - 1.1. isotopy
  - 1.2. 언어요소의 벡터화
- 2. isotopy 실험
  - 2.1. 한국어 KorBERT를 이용한 실험
  - 2.2. 중국어 MacBERT를 이용한 실험
- 3. 동형어 중의성 해소 실험
  - 3.1. Word2Vec + FSE를 이용한 실험 1: 통상적 방법
  - 3.2. Word2Vec + FSE를 이용한 실험 2: 구문분석 활용
  - 3.3. KorBERT를 이용한 실험
  - 3.4. AutoML을 이용한 벡터의 분류
- 4. 맺는 말

1. 문제제기
1.1. isotopy

- 한 문장 안에서 공기하는 단어들은 의미상 서로 잘 어울리는 경향이 있음.
  - 그는 밥을 먹었다: 명사 ‘밥’(음식물)과 동사 ‘먹다’는 잘 어울림.
  - 그는 물을 마셨다: 명사 ‘물’(액체)과 동사 ‘마시다’는 잘 어울림.
- 원래는 특별히 잘 어울린다고 보기 어려운 단어들도 한 문장 안에서 공기하면 서로 잘 어울리게끔 해석되는 경향이 있음.
  - 이 약 빨리 마셔라: ‘약’은 꼭 액체인 것은 아니라, 이 문장에서는 동사 ‘마시다’와 공기함으로써 액체로 해석됨.
- 복수의 facet을 지닌 단어(다면어)는 문장 안에서 공기하는 단어들의 영향을 받아 그 중 어느 한 facet이 부각됨.
  - 책을 읽다, 책이 어렵다, 책이 재미있다: ‘책’의 텍스트로서의 facet이 부각됨.
  - 책을 찢다, 책이 두껍다, 책이 무겁다: ‘책’의 구체물로서의 facet이 부각됨.

동사의 isotopy

- ‘읽다’는 목적어가 무엇이냐에 따라 의미에 차이가 있음.
  - ‘책을 읽다’: 단순히 눈으로 글자를 보고 텍스트를 이해.
  - ‘마음/생각/작전을 읽다’: 눈에 보이지 않는 추상적인 것을 추론을 통해 간파.
- ‘차다’는 목적어가 무엇이냐에 따라 점진적, 급진적 의미의 차이를 보임.
  - ‘공을 차다’: 발로 충격을 가하여 공을 멀리 이동시킴.
  - ‘문을 차다’: 발로 충격을 받은 문은 움직이기는 하나 제자리에서 회전 이동.
  - ‘의자를 차다’: 극장의 붙박이 의자는 차도 움직이지 않음. 보통의 의자는 약간 움직일 수 있음.
  - ‘벽을 차다’: 발로 충격을 가해도 웬만해서는 벽은 움직이지 않음.

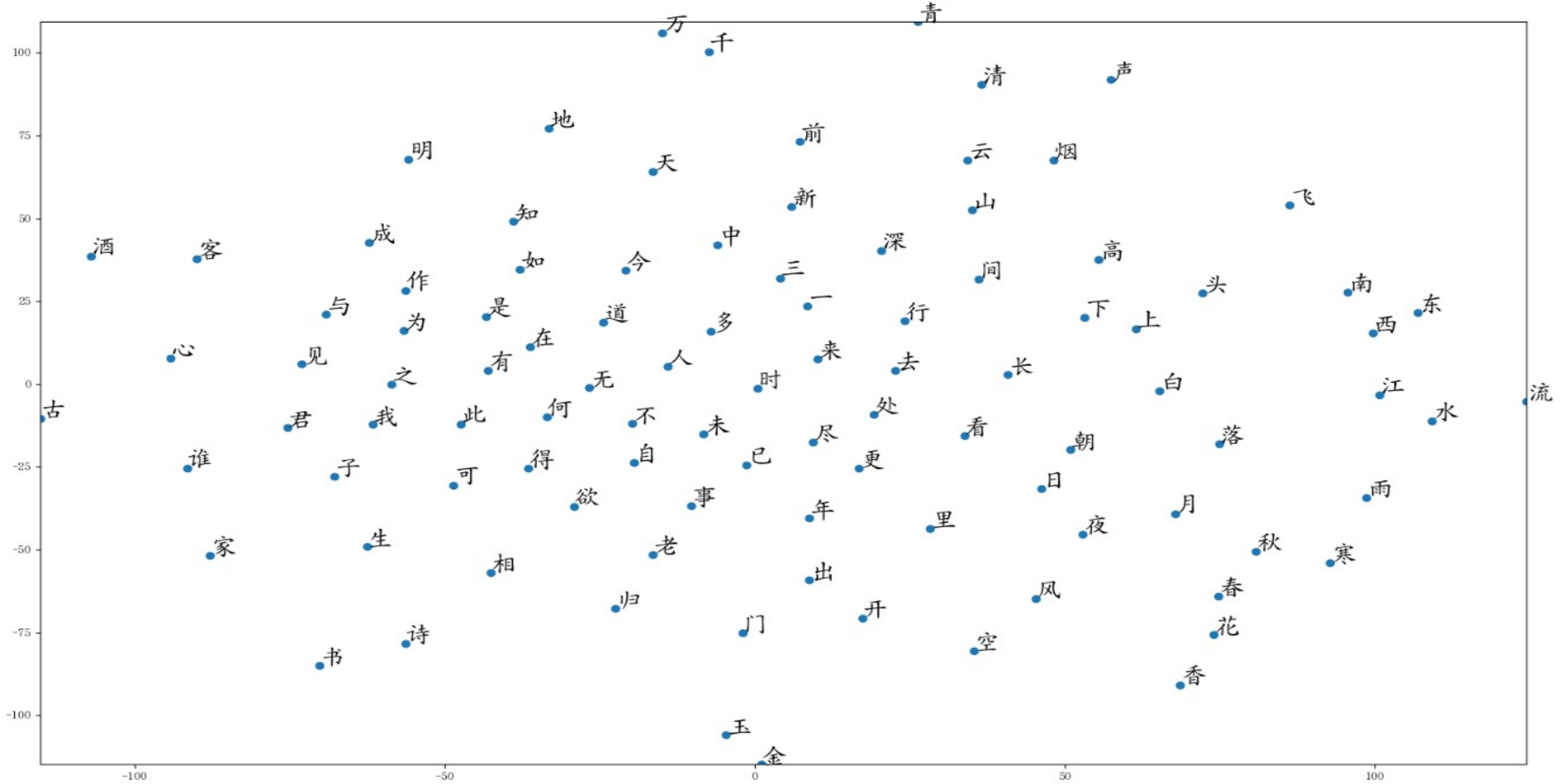
1.2. 언어 요소의 벡터화

- 자연언어의 언어 단위들(단어, 문장 등)은 본질적으로 분절적(discrete)이기 때문에 연속적인 수치로 나타내기에 부적합
- 그래도 기계학습을 언어에 적용하려면 어떻게 해서든 수치화해야 함.
- 언어 요소가 지닌 풍부한 의미를 하나의 수치에 담기에는 부족하기 때문에 대개 수백 개의 수치들로 이루어진 벡터로 나타냄.
- 단어를 벡터화할 때는 타깃 단어 가까이에 나타나는(共起하는) 단어들을 바탕으로 함.
- 어떤 두 단어가 주위에 데리고 다니는 이웃 단어들이 비슷하다면 이 두 단어는 의미도 비슷하다고 간주하고 비슷한 벡터로 나타냄.
- 2012년경 이 아이디어를 구현한 Word2Vec 알고리즘이 나와서, 단어의 벡터화에서 혁명적 발전을 이룸.

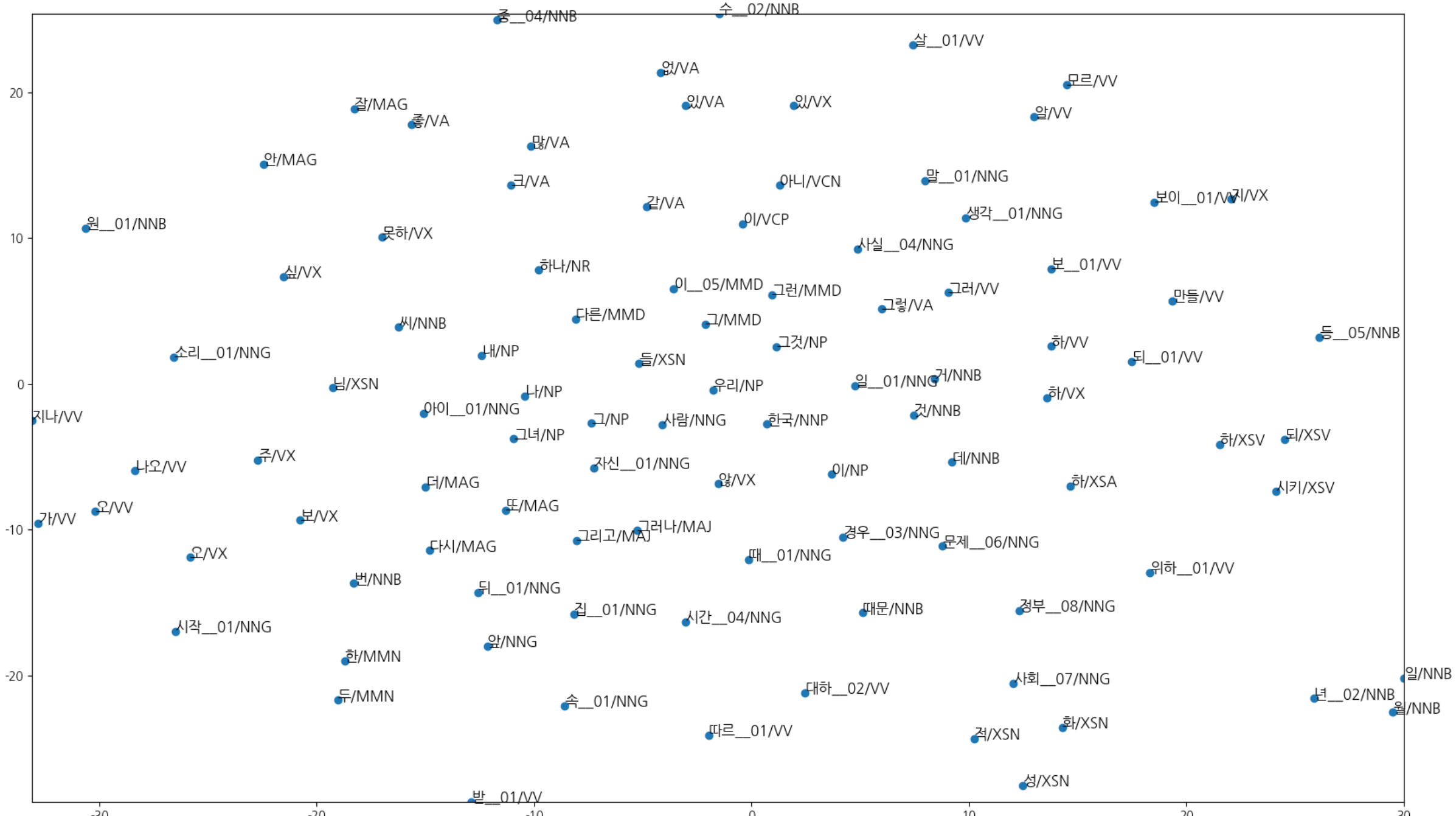
언어 요소 벡터의 품질 평가 기준

- 언어 요소를 적절히 벡터화하면 언어 요소들 사이의 의미 관계가 벡터들 사이의 거리로 반영됨.
- 1. 어떤 두 언어 요소가 의미가 비슷하다면 이에 대응하는 두 벡터는 거리가 가까움: 유사성 기준
- 2. 두 언어 요소가 의미상 매우 다르다면 이에 대응하는 두 벡터는 거리가 멂: 차이 기준
- 3. 네 언어 요소가 A:B=C:D 식으로 유추 관계에 있다면(예: 남자:여자=아버지:어머니) 이에 대응하는 벡터들을 두 개씩 연결한 선도 평행하거나 그에 가깝게 됨: 유추 기준

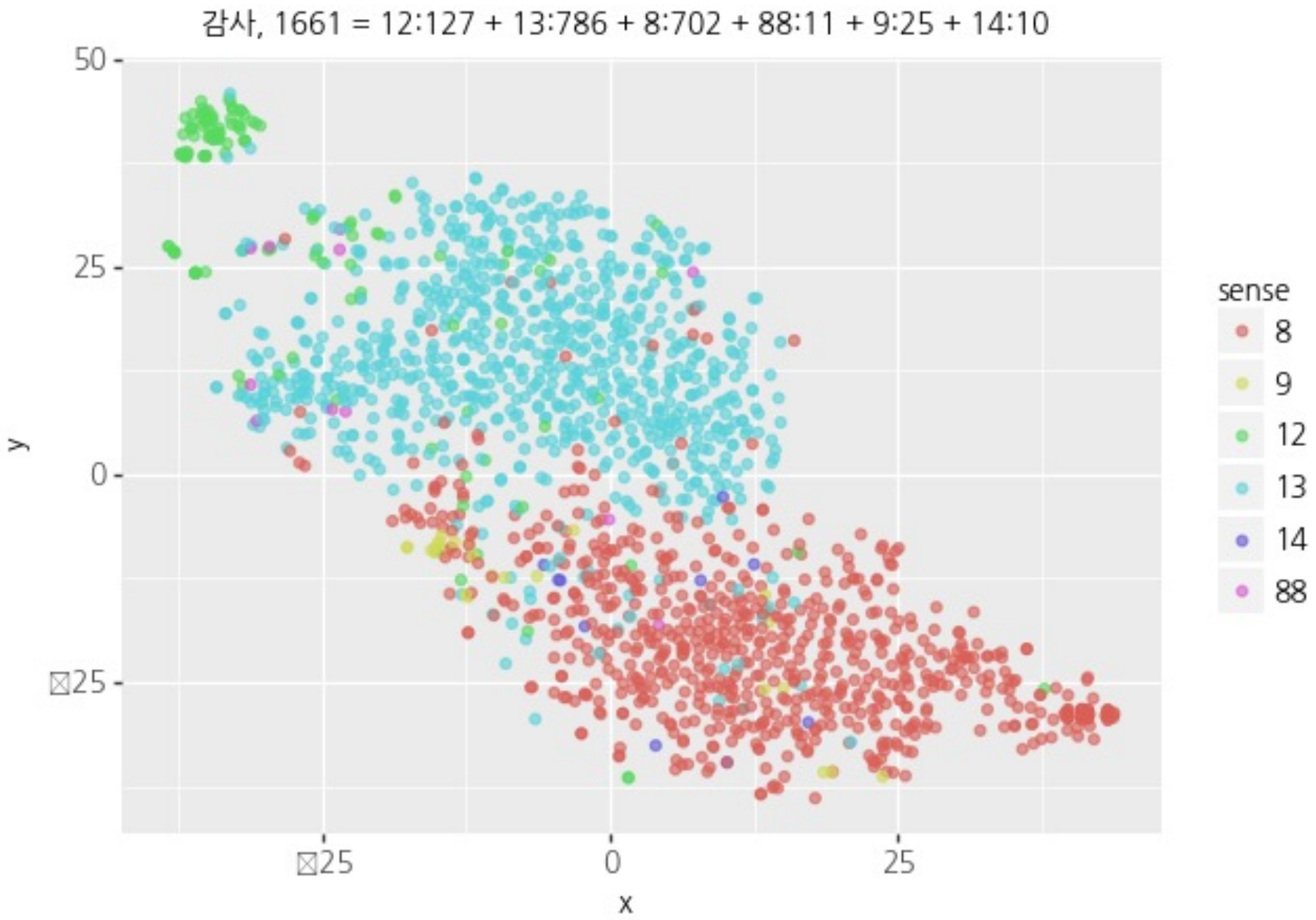
<그림 1> 한문 텍스트의 고빈도 한자들을 벡터화한 결과



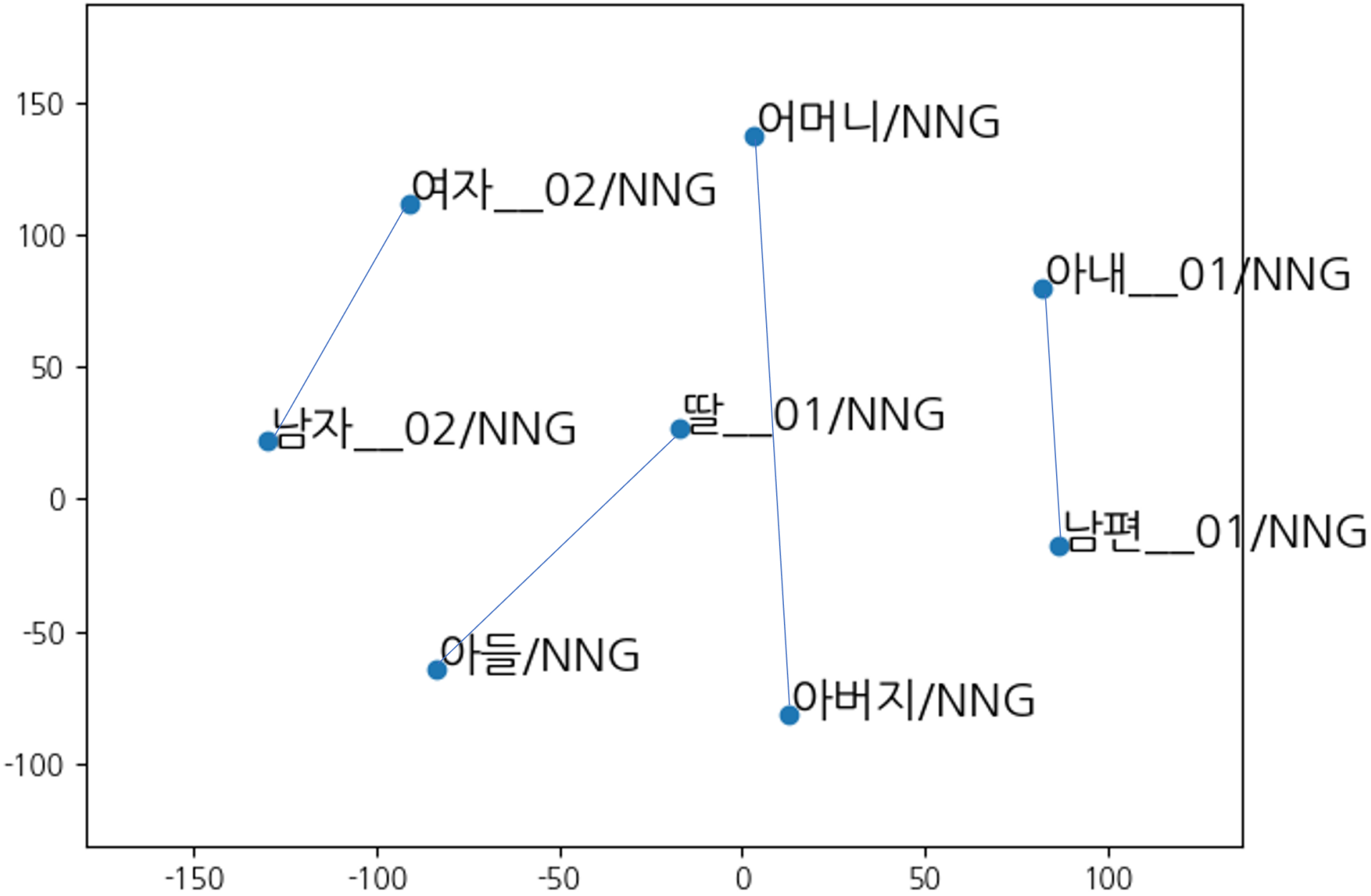
<그림 2> 세종 형태의미분석 말뭉치의 고빈도 형태소를 벡터화한 결과



<그림 3> 동형어 '감사'의 예문들을 Word2Vec과 FSE로 벡터화한 결과



<그림 4> 세종 형태의미분석 말뭉치의 4개 단어쌍을 벡터화한 결과



isotopy를 모델링하기

- Word2Vec이 꽤 품질 좋은 벡터를 만들어내기는 하나, 각 단어를 고정된 벡터로 표상하기 때문에, isotopy를 제대로 포착할 수 없음.
- 최근 각광을 받고 있는 BERT 언어모델은 attention mechanism을 통해 isotopy를 포착할 수 있음.
  - 각 단어 벡터(query)는 문장 내 모든 단어 벡터(key)들의 가중평균으로 조정됨.
  - 가중평균 계산시에 사용되는 가중치(value)는 신경망 학습으로 얻음.
  - 신경망에, 문장 중간중간에 masking된 단어들을 알아맞히는 과제를 부여하여, 이 과제를 잘 수행하게끔 신경망을 훈련시킴.
  - 가중평균 계산시 자기자신의 가중치가 당연히 가장 크지만
  - 문장 내의 다른 단어들에도 약간의 가중치가 부여되어 이 단어들의 영향을 받게 됨.
  - query 단어와 관계가 밀접한 단어(key)일수록 가중치(value)가 큼.

2. isotopy 실험
2.1. 한국어 KorBERT를 이용한 실험

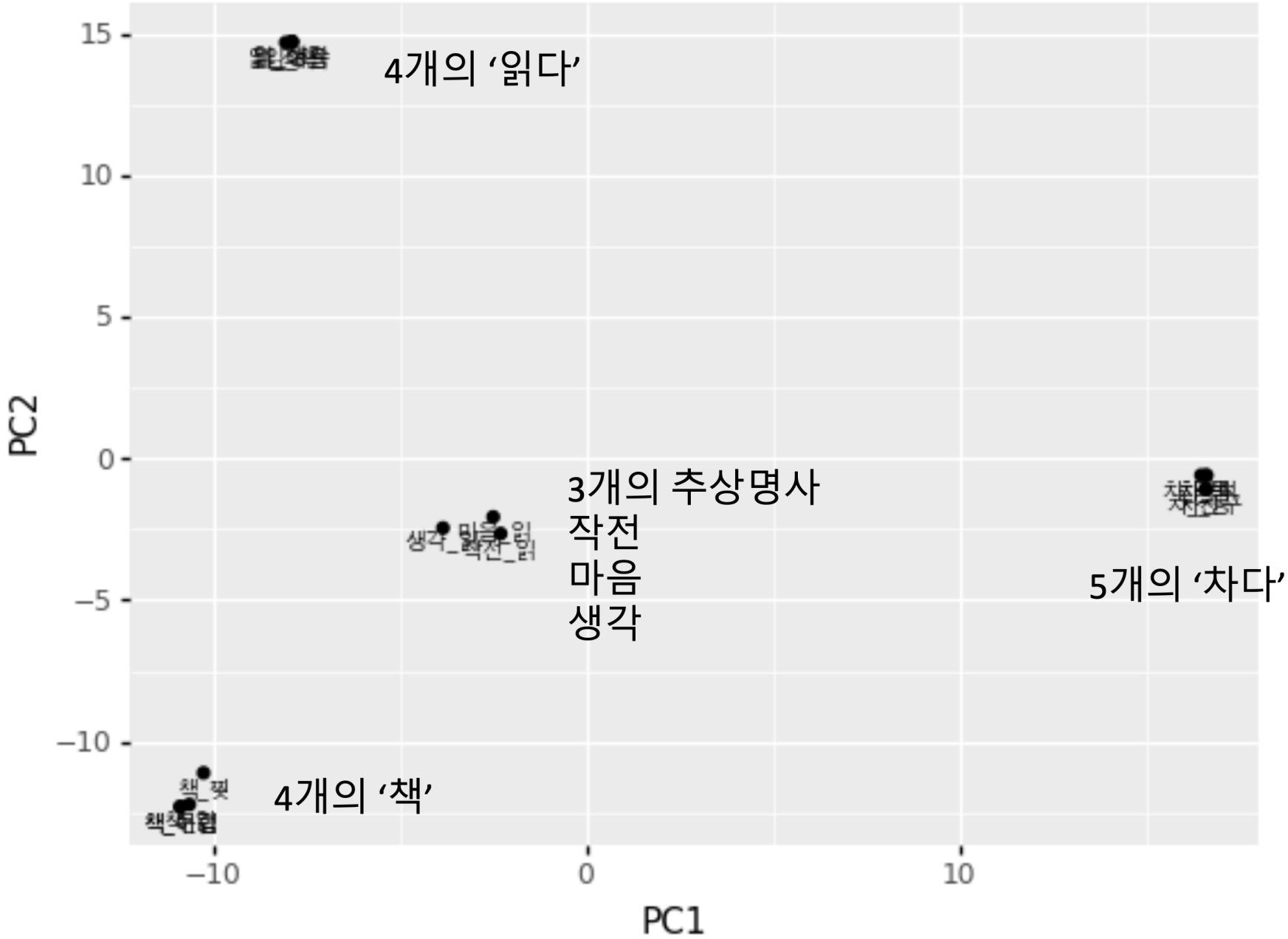
- 실험에 사용한 문장(12개)
- (1) 재미있는 책을 읽었다.
- (2) 두꺼운 책을 찢었다.
- (3) 이 책은 너무 어렵다.
- (4) 이 책은 너무 두껍다.
- (5) 상대의 마음을 읽었다.
- (6) 상대의 생각을 읽었다.
- (7) 적군의 작전을 읽었다.
- (8) 축구공을 차서 골문에 넣었다.
- (9) 발로 문을 차서 열었다.
- (10) 발로 벽을 차서 발이 아프다.
- (11) 발로 앞좌석을 차서 앞사람이 화났다.
- (12) 남자친구를 차서 마음이 안 좋다.

- 이 12개 문장에 나타나는 16개 토큰의 벡터를 조사. (타깃_문맥)
- ①읽_책
- ②책_읽
- ③책_찢
- ④책_어렵
- ⑤책_두껍
- ⑥읽_마음
- ⑦마음_읽
- ⑧읽_생각
- ⑨생각_읽
- ⑩읽_작전
- ⑪작전_읽
- ⑫차_공
- ⑬차_문
- ⑭차_벽
- ⑮차_좌석
- ⑯차_친구

2.1. 한국어 KorBERT를 이용한 실험

- 실험에 사용할 언어모델로 ETRI의 KorBERT를 선택.
  - 기타 한국어 BERT 모델들은 한국어/한글에 대한 지식이 없는 tokenizer를 사용했기 때문에 토큰화 결과가 형태소 단위와 불일치하는 경우가 많음. 본 발표의 목적에 맞지 않음.
  - KorBERT는 글자 단위 토큰화 모델도 제공하나, 여기서는 형태소 단위 토큰화 모델 사용.
- BERT에 입력 문장을 넣으면, 토큰화하여 토큰(형태소) 단위로 분리하고, 각 토큰을 일단 벡터화함.
  - 벡터의 차원의 default는 768차원.
  - 입력 층(input layer)에서 각 토큰의 벡터는 아직 문장 내 다른 토큰의 영향을 받기 전 상태.
  - 그 뒤에 은닉 층(hidden layer)들을 거치면서, attention mechanism에 의해 다른 토큰의 영향을 받아 각 토큰의 벡터가 조정됨.
- 4개의 '읽-', 5개의 '차-', 4개의 '책', 3개의 추상명사('생각', '마음', '작전')이 각각 매우 가까이에 몰려 있음.

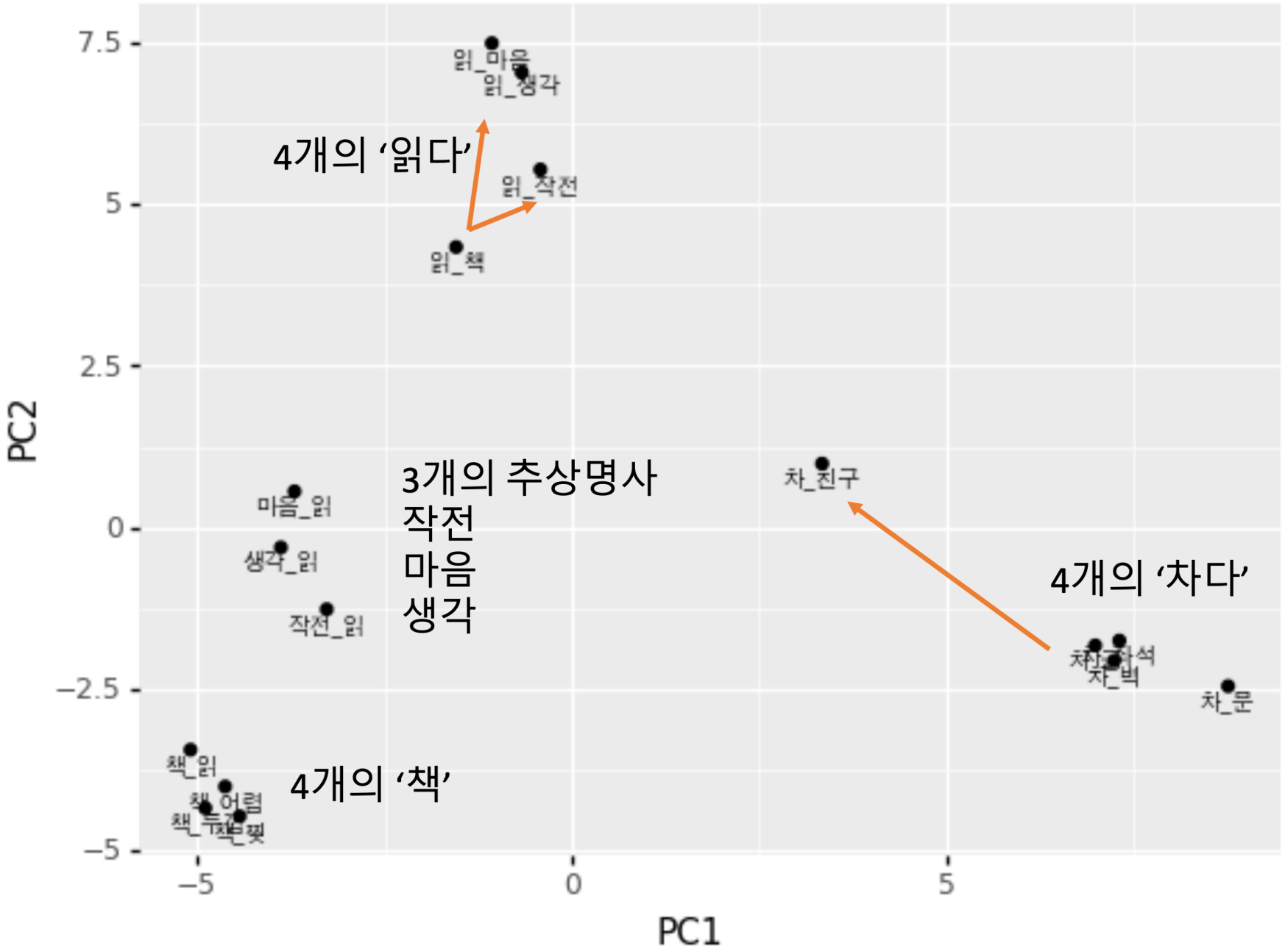
<그림 5> 16개 토큰의 입력 층에서의 벡터를 PCA로 2차원화한 결과



2.1. 한국어 KorBERT를 이용한 실험

- 문장 내 다른 토큰들의 영향을 받은 후인 마지막 은닉 층에서의 16개 토큰의 벡터를 추출하여, 마찬가지로 PCA를 통해 2차원으로 축소하여 plotting하였음.
- '책': 공기한 다른 토큰의 영향이 상대적으로 작게 나타났으나
- 3개의 추상명사: 공기한 동사의 영향으로 서로 거리가 좀 멀어졌음.
- 동사 '읽-'
  - 목적어가 '책'일 때에 비해, 목적어가 '작전'일 때 어느 정도 거리가 멀어졌고
  - 목적어가 '생각'이나 '마음'일 때는 한결 더 멀어졌음.
- 동사 '차-'
  - 목적어가 '공', '벽', '좌석'인 경우는 여전히 가까이에 몰려 있으나
  - 목적어가 '문'인 경우 우측으로 약간 멀어졌고
  - 목적어가 '남자친구'인 경우 좌상 방향으로 매우 멀리 멀어졌음.
- 특히 동사의 경우 공기하는 목적어의 영향을 받아 의미 해석이 달라지는 isotopy 현상을 BERT 모델이 어느 정도 포착하고 있음.

<그림 6> 16개 토큰의 마지막 은닉 층에서의 벡터를 PCA로 2차원화한 결과

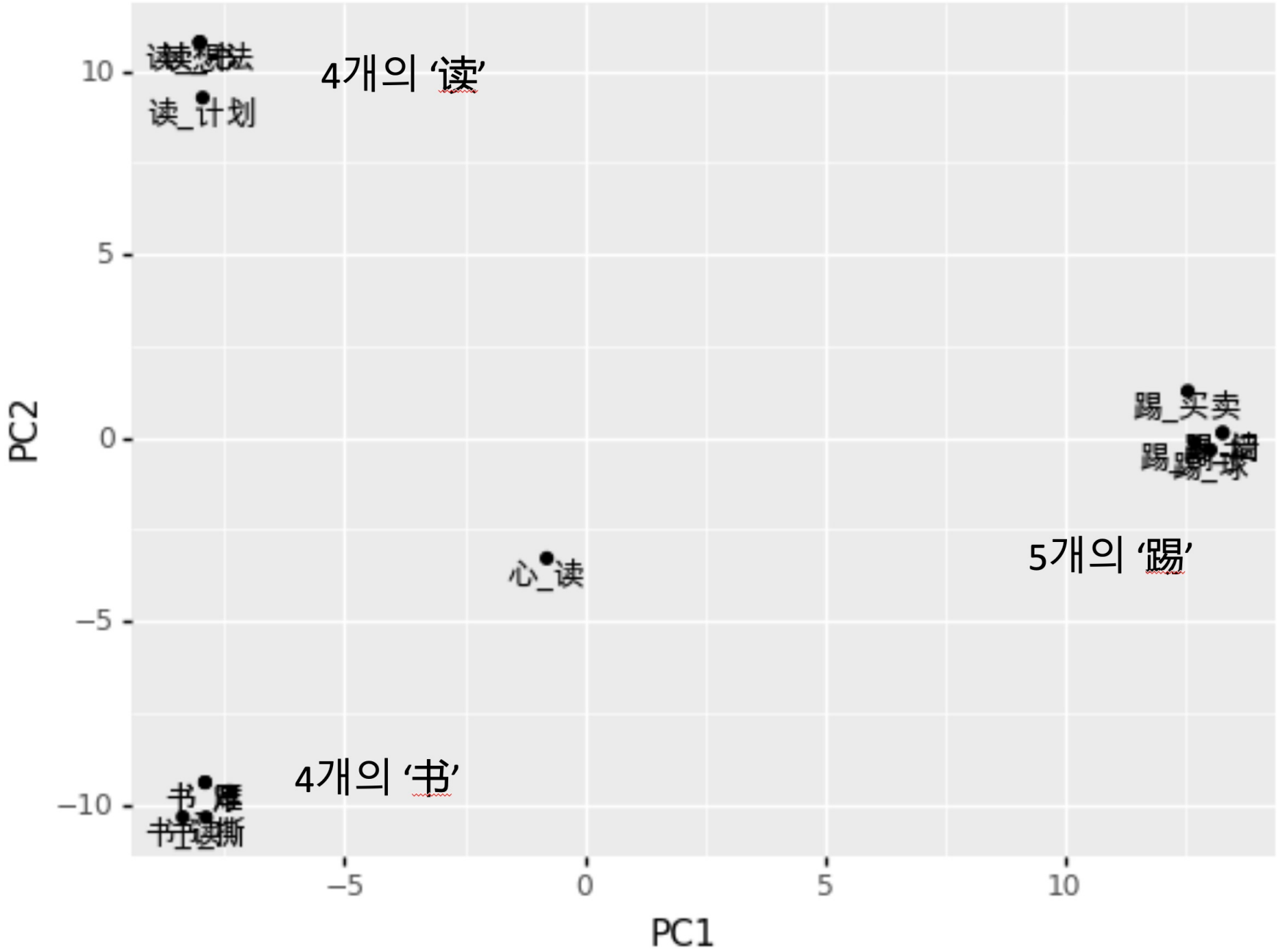


2.2. 중국어 MacBERT를 이용한 실험

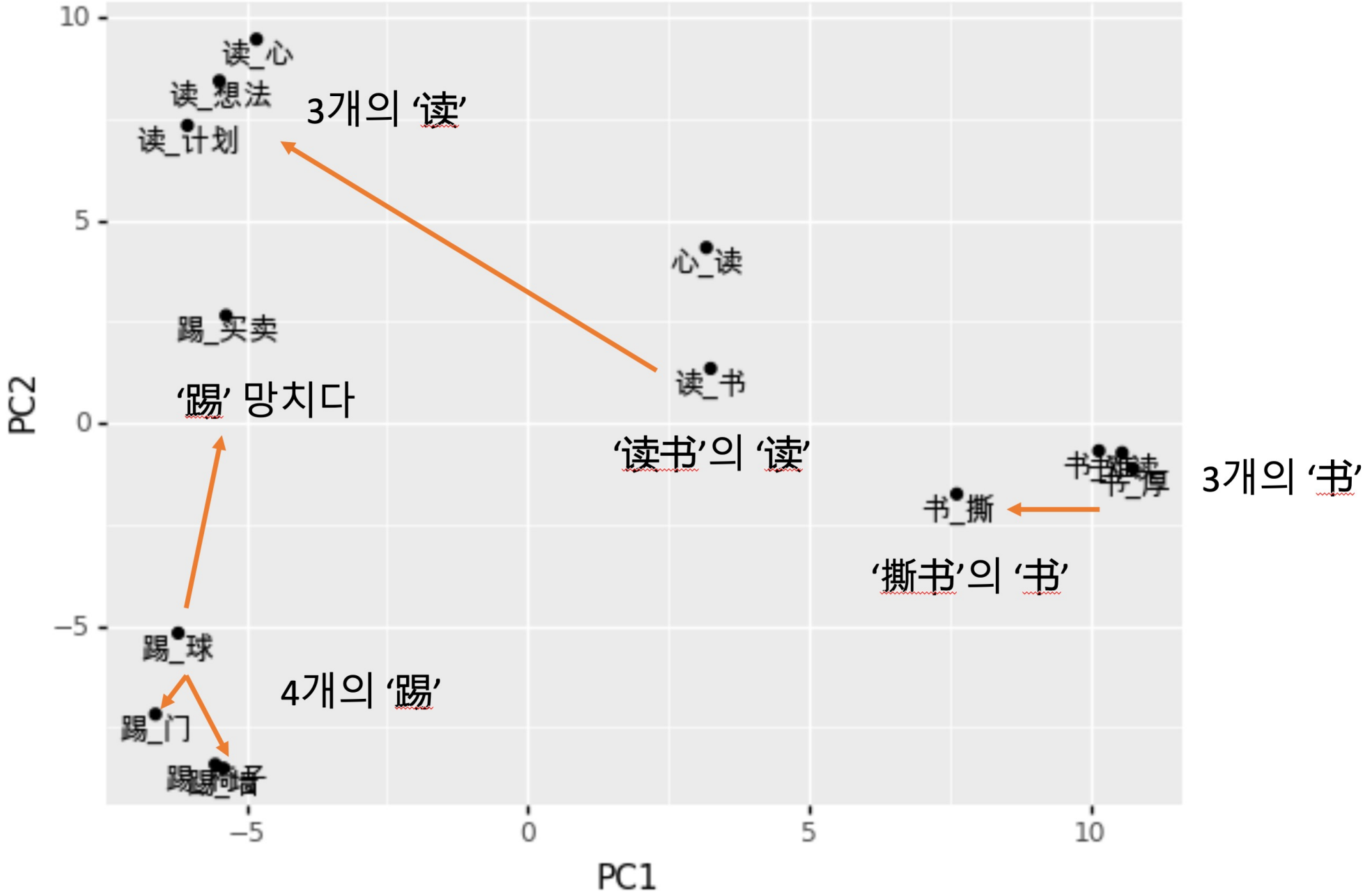
- 실험에 사용한 문장(12개)
- (1) 我读了一本有趣的书。
- (2) 我撕了厚厚的书。
- (3) 这本书太难了。
- (4) 这本书太厚了。
- (5) 我读懂了对方的心。
- (6) 我读出了对方的想法。
- (7) 我们读出了敌军的作战计划。
- (8) 我把足球踢进了球门。
- (9) 我用脚踢开了门。
- (10) 我用脚踢了墙，脚疼。
- (11) 用脚踢了椅子，前面的人生气了。
- (12) 我踢了他们俩的买卖。

- 이 12개 문장에 나타나는 14개 토큰의 벡터를 조사. (타깃_문맥)
- ①读_书
- ②书_读
- ③书_撕
- ④书_难
- ⑤书_厚
- ⑥读_心
- ⑦心_读
- ⑧读_想法
- ⑨读_计划
- ⑩踢_球
- ⑪踢_门
- ⑫踢_墙
- ⑬踢_椅子
- ⑭踢_买卖

<그림 7> 14개 토큰의 입력 층에서의 벡터를 PCA로 2차원화한 결과



<그림 8> 14개 토큰의 마지막 은닉 층에서의 벡터를 PCA로 2차원화한 결과



2.2.의 실험 결과에 대한 해석/논의

- 동사 '读'
  - '书'를 목적어로 하는 '读'가 중앙에 있는 데 비해
  - 추상명사 '心', '想法', '计划'를 목적어로 하는 '读'는 좌상 구석에 매우 멀리 떨어져 있음.
  - 보통의 독서 행위와 마음/생각/작전을 읽는 행위가 사뭇 다르다는 것을 잘 포착하고 있음.
- 동사 '踢'
  - '球'를 목적어로 하는 '踢'에 비해, '门'을 목적어로 하는 '踢'는 약간 아래로 내려갔고
  - '墙', '椅子'를 목적어로 하는 '踢'는 더 아래로 내려갔으며
  - 추상명사 '买卖'를 목적어로 하여 "망치다"를 의미하는 '踢'는 위로 멀리 이동.
  - 공을 차면 공이 꽤 멀리 이동하나, 문을 차면 문이 축을 중심으로 하는 회전 이동을 할 뿐 위치가 변하지는 않음.
  - 의자의 경우 가정의 보통 의자라면 차는 동작에 의해 약간 움직일 수 있겠으나 그리 많이 움직이지는 않고, 극장의 붙박이 의자라면 발로 차도 움직이지 않음. 그래서 목적어가 '벽'이나 '의자'일 경우는 대상물이 움직이는 경우와 사뭇 달라서 아래쪽으로 많이 이동한 듯.
  - 목적어가 '买卖'인 경우는 발로 차는 것과는 완전히 다른 의미이므로 가장 멀리 이동.
- 명사 '书'
  - 4개의 '书' 가운데 동사 '读'와 공기할 때는 텍스트로서의 부면이 부각되고, 나머지 세 경우는 구체물로서의 부면이 부각되므로 전1자와 후3자가 대비되기를 기대했으나
  - 실제로는 동사 '撕'(찢다)와 공기한 '书'와 나머지 셋(读, 难, 厚)으로 나뉨.
  - '읽다', '어렵다', '두껍다'는 '책'을 목적어로 한 꽤 일반적이고 정상적인 행위인 데 비해
  - '찢다'는 그에 비해 드물고 비정상적인 행위라는 점이 이런 결과를 낳은 듯.

제2절의 실험 결과에 대한 해석/논의

- BERT 모델의 13개 은닉층 가운데 attention mechanism에 의해 다른 단어들의 영향을 받기 전 상태에서는, 한 단어(query)의 벡터들은 (출현 문장, 문장 내 다른 단어(key)들과 상관없이) 매우 가까이에 위치함.
- attention mechanism에 의해 문장 내의 다른 단어(key)들의 영향을 받고 난 뒤에는, 한 단어(query)의 벡터들이 (공기 단어들에 따라) 서로 상이해져서 거리가 멀어지게 됨.
- 공기 단어(key)가 (같은 자리에 나올 수 있는 다른 것들과 이질적인) 특이한 것일수록 query 단어가 달라지는 폭이 커짐.
- 이는 attention mechanism이 isotopy를 어느 정도 포착하고 있음을 시사함.
- 한국어 KorBERT보다 중국어 MacBERT가 isotopy를 더 잘 포착함.
- BERT의 attention mechanism이 isotopy를 포착하기 위한 길을 열어 놓은 것은 맞지만, 그것만으로 isotopy를 완벽하게 포착하는 것은 아니며, 더 좋은 결과를 얻기 위해서는 다방면의 궁리가 필요함

3. 동형어 중의성 해소 실험
3.1. Word2Vec + FSE를 이용한 실험 1: 통상적 방법

- 중의성: 하나의 언어 형식이 둘 이상의 의미로 해석되는 현상
  - 동형어(homonym)도 중의성을 낳은 대표적 원인
- 감사
  - 선생님께 감사 인사를 드렸다: 感謝
  - 이 회사는 회계 감사를 받고 있다: 監査
- 가장
  - 그는 이 집의 가장으로서 많은 책임을 지고 있다: 家長
  - 아무런 가장 없이 솔직하게 자신을 표현해라: 假裝

중의성 해소 방법 1: Symbolic AI, GOFAI(good old-fashioned AI)

- 인간: 문맥과 세상에 대한 지식을 바탕으로 어떤 해석이 더 적절한지 판단
- 인간이 중의성을 해소하는 기제를 본떠서 기계로 구현하려고 시도
- 인간이 지닌 방대한 세상 지식을 적절히 표상
- 문맥으로부터 필요한 정보를 추출하는 규칙 작성
- 코드가 지나치게 복잡해져서, 유지-관리가 어려움.
- 작은 toy 정도를 만드는 데 그치고, 상용화에 실패.
- 산업계와 학계에서 묻혀 버림.

중의성 해소 방법 2: 기계학습

- 기계학습: 어떤 과제를 수행하기 위해 필요한 정보(분류 과제의 경우 각 범주에 속하는 개체들을 구분하기 위해 알아야 할 패턴)를 인간이 기계에게 넣어 주는 것이 아니라 기계가 데이터로부터 스스로 찾음.
- 기계가 그 일을 할 수 있으려면 데이터가 수치로 표상되어야 함.

중의성 해소에 벡터의 활용

- 단어를 벡터화하면 중의성 해소에도 써먹을 수 있음.
- 동형어가 2개의 의미(①과 ②)를 가지고 있다고 할 때, ①의 의미일 때 공기하는 단어와 ②의 의미일 때 공기하는 단어가 사뭇 다를 것이므로, 이 둘이 상당히 다르게 벡터화될 것으로 기대할 수 있음.
  - '감사'가 "感謝"의 뜻일 때는 '드리-', '깊-', '-에게' 등과 흔히 공기
  - '감사'가 "監査"의 뜻일 때는 '회계', '회사', '기관', '국정' 등과 흔히 공기
- 동형어의 구별에 벡터를 써먹으려고 한다면, 단어 벡터 그 자체뿐 아니라 문장 벡터를 이용하면 더 좋음.
  - 단어 벡터를 만들 때, 말뭉치에서 그 단어 가까이에 자주 나타난 단어들의 영향이 이미 반영되어 있겠으나
  - 특정 문장에서 타깃 단어와 공기한 단어들의 벡터까지 종합적으로 고려하면 더 많은 정보를 얻을 수 있음.

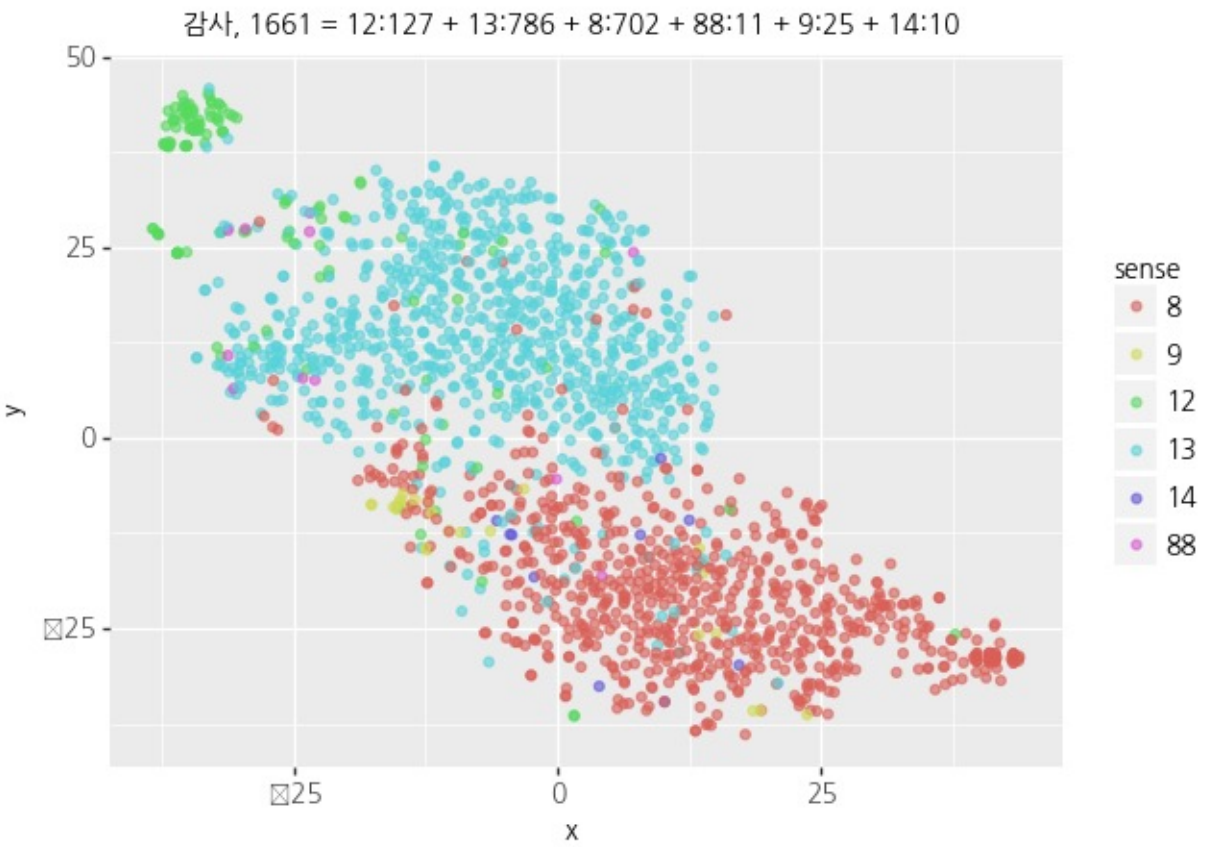
문장 벡터를 얻는 방법

- 단어 벡터들을 종합하여 문장 벡터를 얻는 상식적인 방법은 단어 벡터들의 평균을 내는 것.
  - 한 학급의 수학 성적을 하나의 수치로 나타내려면, 그 학급 학생들의 수학 성적의 평균을 내는 것과 같은 이치.
- 다만 문장을 이루는 모든 단어의 비중/중요도가 같은 것은 아니므로 적절히 가중치를 부여하여 가중평균(weighted average)을 내는 것이 좋음.
  - 매우 많은 문장에 두루 나타나는 단어는 타깃 단어의 중의성 해소에 별 도움이 안 되므로 가중치를 낮춤.
  - 소수의 문장에만 특징적으로 나타나는 단어는 가중치를 높임.
  - FSE(Fast Sentence Embedding)는 바로 그런 아이디어를 구현한 문장 벡터화 알고리즘

‘감사’의 벡터화

- 세종 형태의미분석 말뭉치에 나타나는 명사 ‘감사’의 1660여개의 예문들에 대해, Word2Vec으로 단어를 벡터화하고 FSE로 문장을 벡터화
- 원래 하나의 단어, 하나의 문장을 100차원의 벡터로 나타냈으나, TSNE 알고리즘으로 2차원으로 축소하여 시각화
- ‘감사08(感謝, 적색)’, ‘감사13(監査, 청색)’, ‘감사12(監事, 연두색)’가 비교적 잘 구분되어 있음.

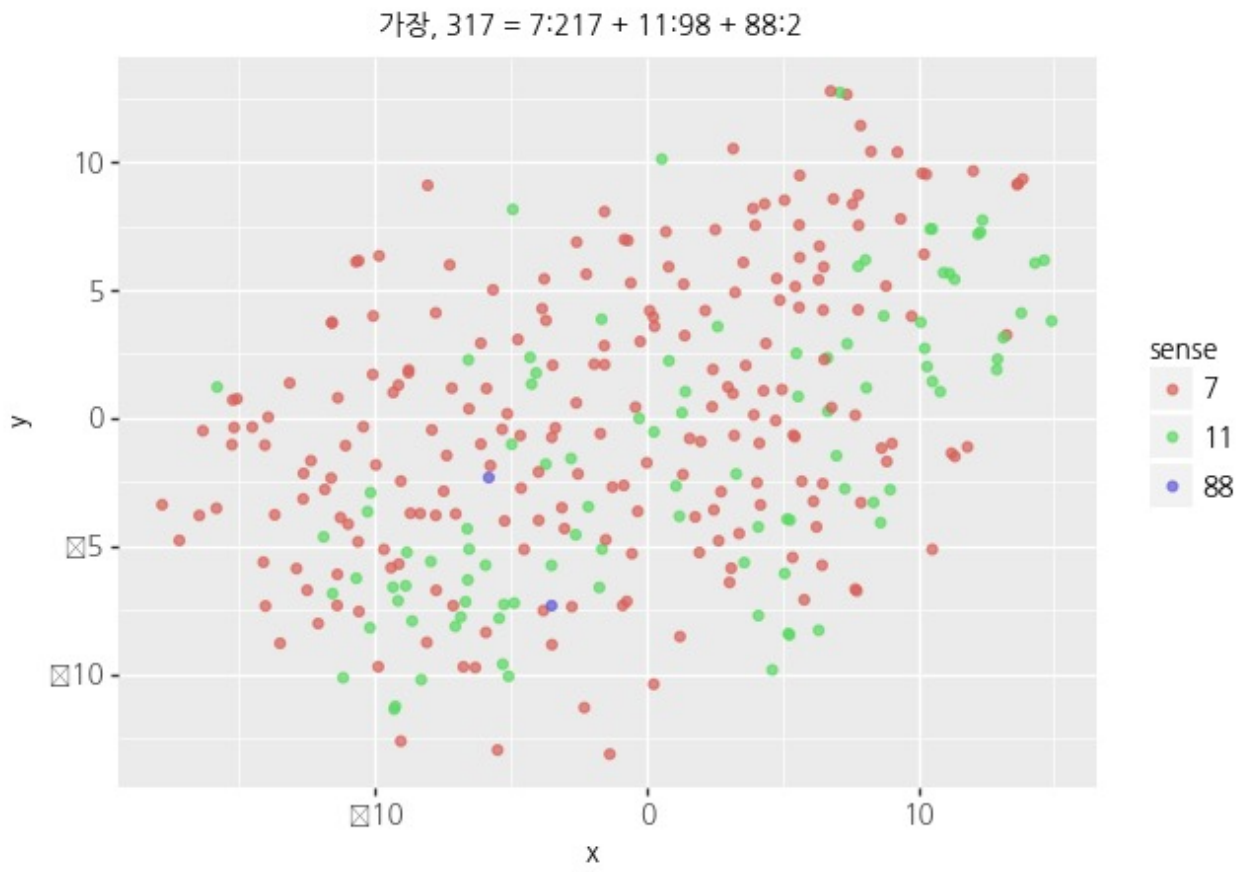
<그림 3> 동형어 ‘감사’의 예문들을 Word2Vec과 FSE로 벡터화한 결과



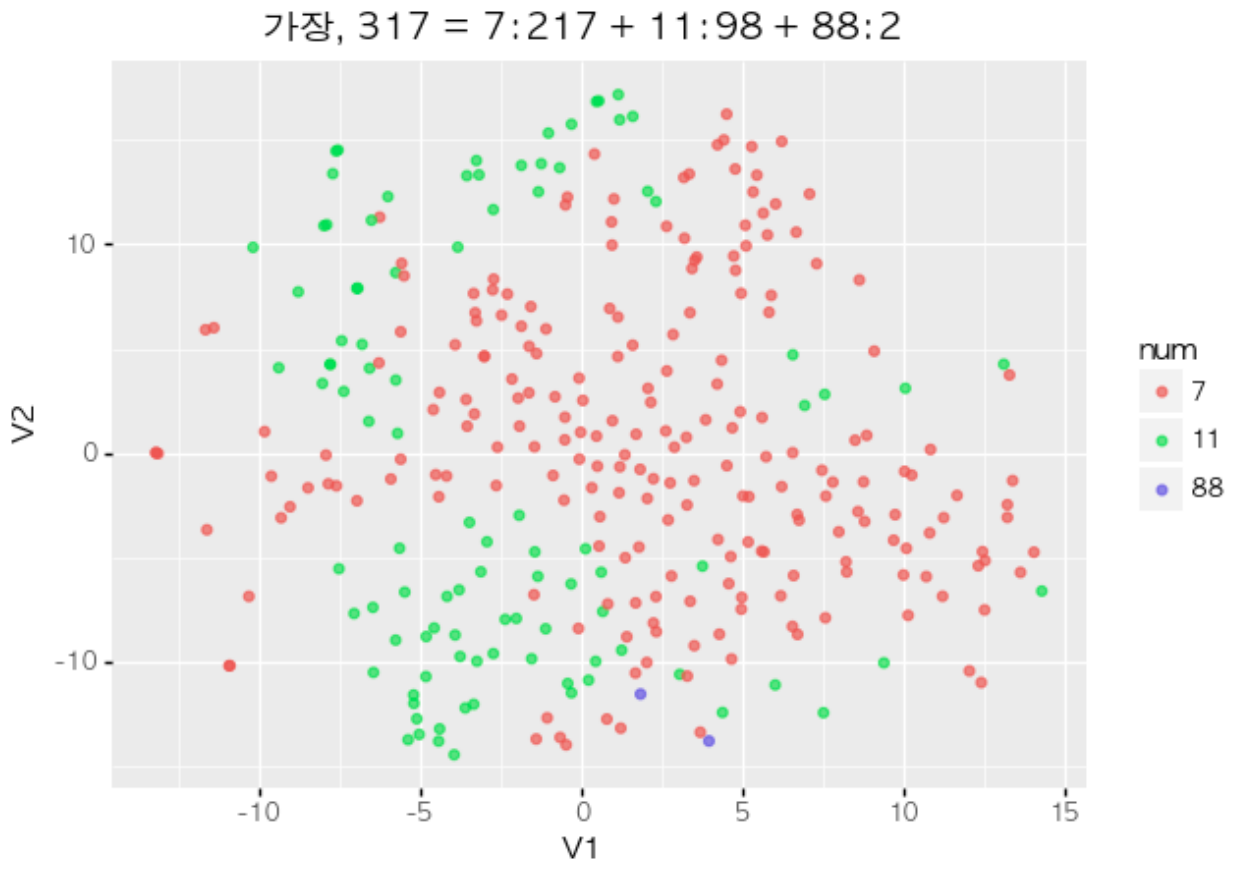
'가장'의 벡터화

- Word2Vec+FSE 같은 벡터화 알고리즘이 동형어 구별을 위해 항상 좋은 결과만을 낳는 것은 아님.
- '가장'의 경우에는 "家長"과 "假裝"의 경우들이 뒤섞여 있어서, 이것으로는 중의성 해소를 제대로 할 수 없음(좌측).
- 문장을 너무 길게 끊으면, 문장 안에 중의성 해소를 위한 단서뿐 아니라 노이즈도 많이 포함되게 되어 이것이 중의성 해소를 방해할 가능성도 있음.
- 그래서 문장을 약간 짧게 끊어서 벡터화를 해 보았는데(우측), '가장'의 두 의미가 뒤섞이는 현상이 약간 나아지기는 했으나, 여전히 매우 불만족스러움.

<그림 9> '가장'의 벡터화: Word2Vec + FSE (문장 길게)



<그림 10> '가장'의 벡터화: Word2Vec + FSE (문장 짧게)



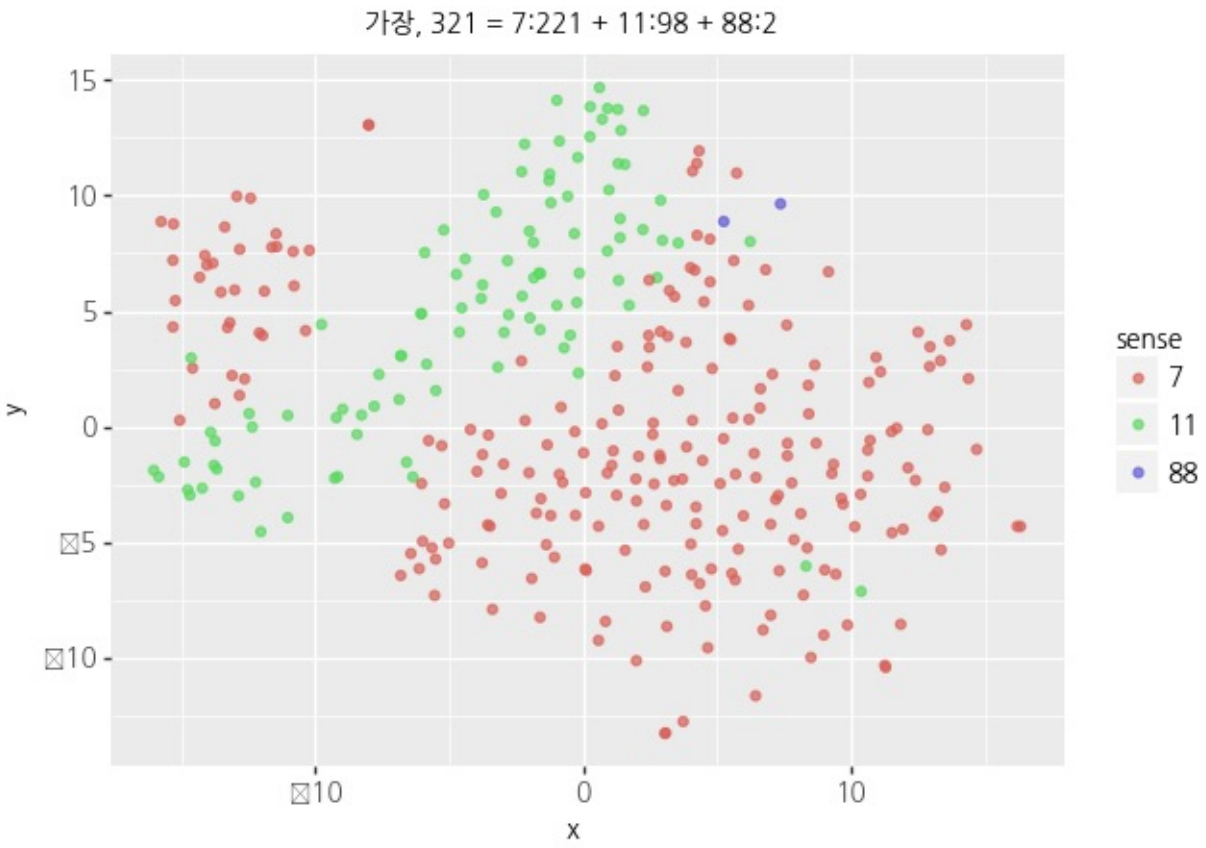
3.2. Word2Vec + FSE를 이용한 실험 2: 구문분석 활용

- 여기서 언어학의 도메인 지식이 도움이 될 수 있음.
- 문장을 구성하는 형태소나 단어들이 서로 맺는 관계의 긴밀도가 모두 동일한 것은 아님.
  - '맛있는 사과를 빨리 먹었다'와 '잘못했으면 상대방에게 사과를 빨리 하는 게 좋다'에서 동형어 '사과'의 중의성 해소에 '맛있-', '먹-', '상대방', '-에게' 등은 도움이 되지만 '빨리' 같은 것은 별로 도움이 되지 않음.
- 대개 타깃 단어와 통사적으로 밀접한 관계에 있는 요소는 중의성 해소에 도움이 되지만, 그 외의 요소는 별 도움이 안 될 때가 많고 오히려 방해가 될 때도 있음.
- 따라서 문장의 모든 토큰을 포함해서 벡터화할 게 아니라, 중의성을 해소하고자 하는 타깃 단어와 통사적으로 긴밀한 관계에 있는 요소들만 뽑아서 벡터화하는 게 더 좋음.

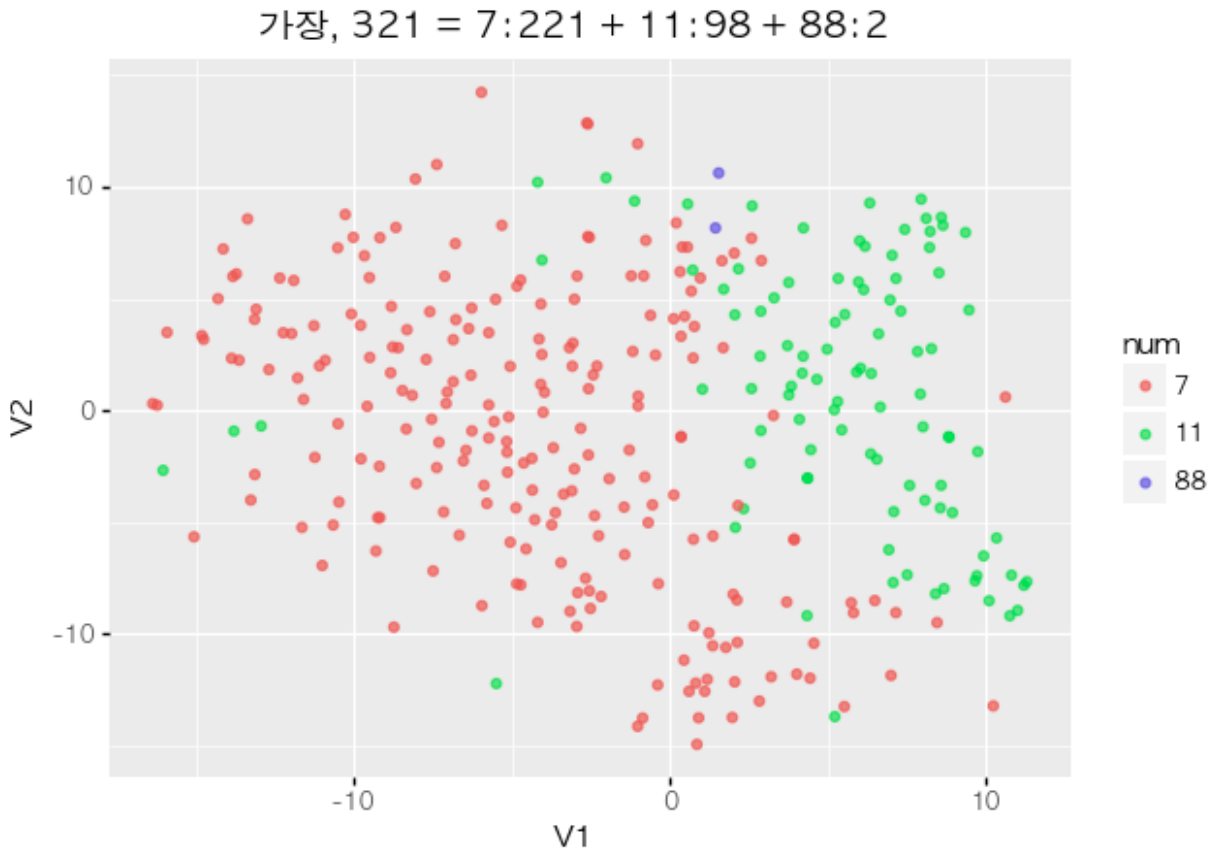
‘가장’ 예문들을 통사 구조를 고려하여 벡터화

- ‘가장’의 예문들을 구문분석한 뒤
- 타깃 단어 ‘가장’과 통사적으로 긴밀한 관계에 있는 요소들만 뽑아서 Word2Vec+FSE로 벡터화.
- 문장을 길게 끊든(좌측), 짧게 끊든(우측) ‘가장’의 두 의미가 뚜렷하게 구분되어 있음.
- 문장의 통사 구조에 대한 언어학적 고려가 동형어의 중의성 해소에 도움이 될 수 있음.

<그림 11> '가장'의 벡터화: Word2Vec + FSE + 구문분석 (문장 길게)



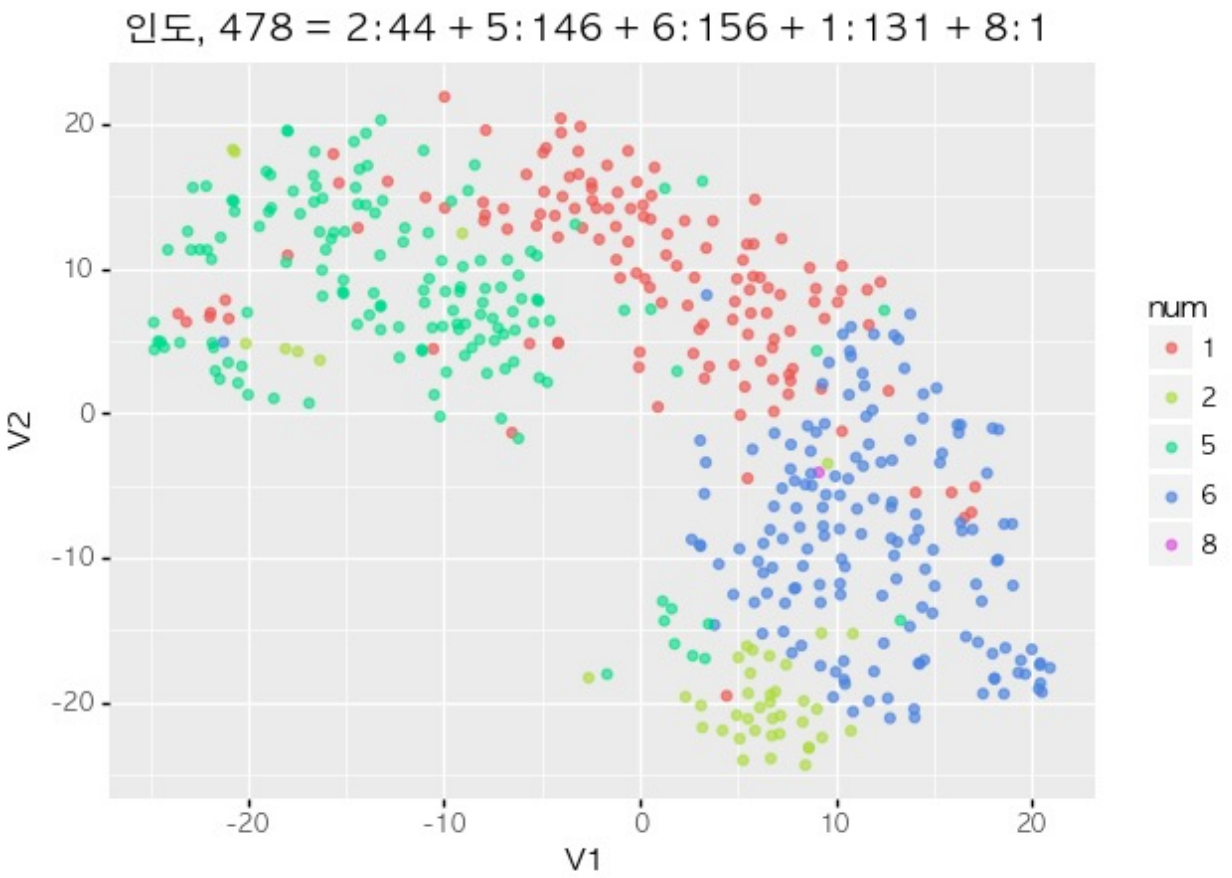
<그림 12> '가장'의 벡터화: Word2Vec + FSE + 구문분석 (문장 짧게)



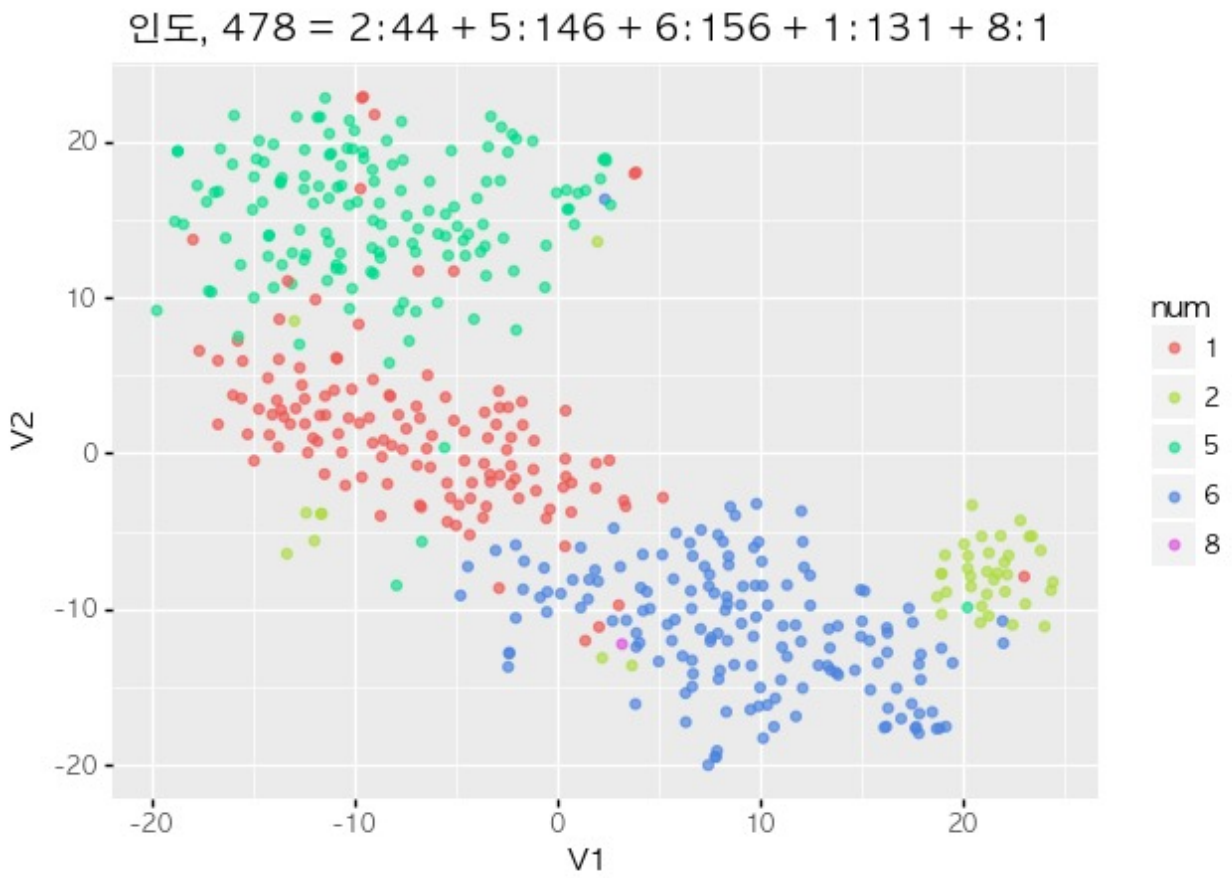
'인도'의 벡터화

- 구문분석을 고려하지 않고 통상의 방법으로 벡터화했을 때도 용법들이 꽤 잘 구분되나(좌측)
- 구문분석을 고려하여 벡터화하면 더 잘 구분됨(우측).

<그림 13> '인도'의 벡터화: Word2Vec + FSE



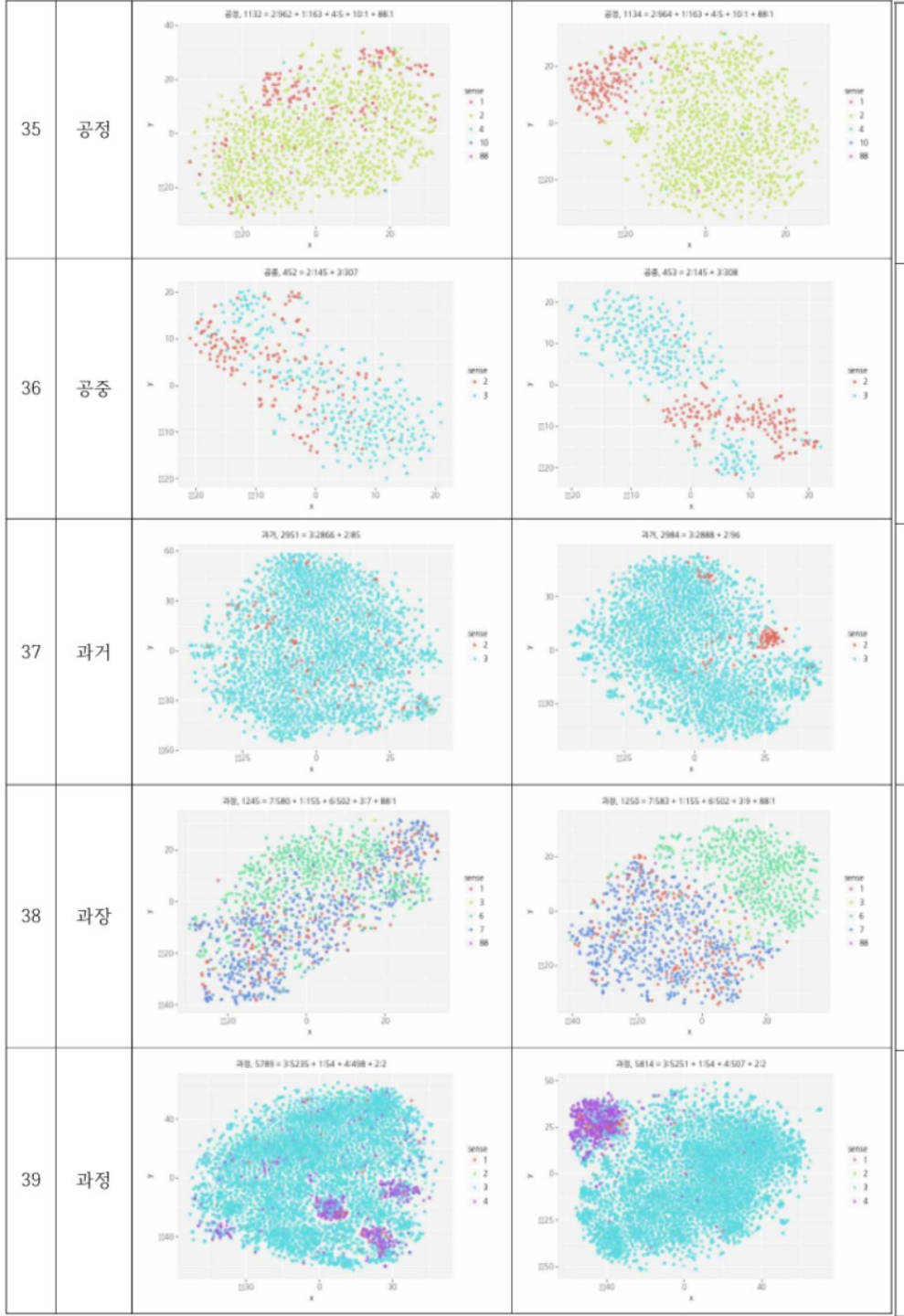
<그림 14> '인도'의 벡터화: Word2Vec + FSE + 구문분석

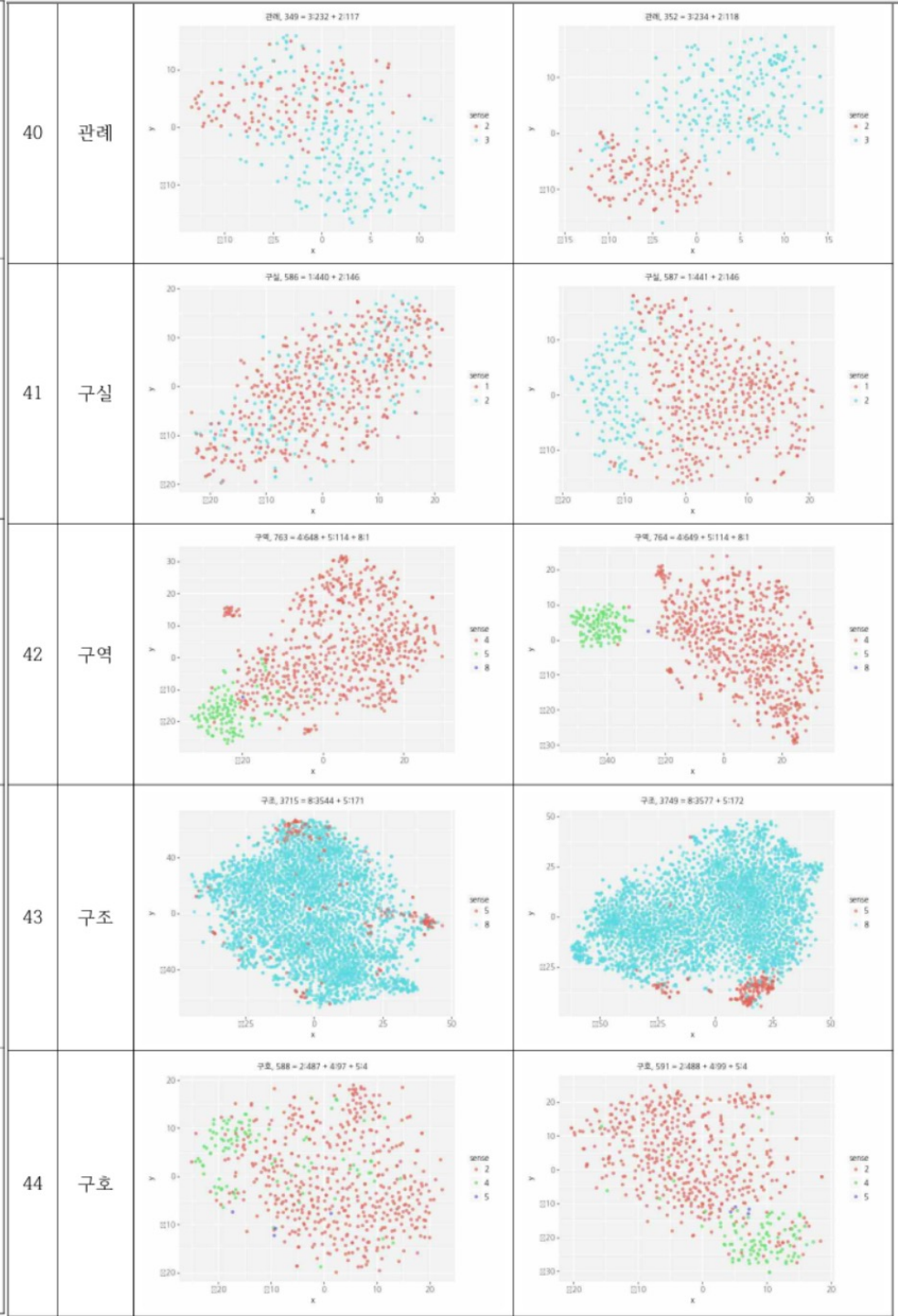


285개 동형어 명사 실험 결과

- 285개 명사 전체의 실험 결과를 종합하면, 통사 관계를 반영하여 벡터화했을 때 (그렇지 않았을 때에 비해) 동형어들을 훨씬 더 뚜렷하게 나누는 경향이 매우 강함.
  - 뒤 슬라이드의 좌측 plot에 비해 우측 plot이 향상된(동형어가 더 뚜렷하게 나뉜) 경우가 대부분임.
  - 두 plot이 별 차이가 없는 경우도 소수 있음. 예: 강화, 개정, 규정, 사채, 성적, 세계
  - 좌측 plot에 비해 우측 plot이 더 악화된(동형어가 더 뒤섞인) 경우는 없음.
- 야구 팀에서 별 특징 없는 그저그런 선수들을 제외하고 팀 컬러를 대표하는 선수들만 추리면, 해당 팀의 팀 컬러가 강화되듯이
- 문장에서 별 특징 없는 단어들을 제외히고, 타깃 단어의 특징을 잘 보여주는 단어들만 추리면, 해당 문장의 특징이 더 뚜렷해짐.
- 타깃 단어의 특징을 잘 보여주는 단어는, 타깃 단어와 통사적으로 밀접한 관계를 맺고 있는 단어임.

<그림 15> 통상적인 벡터화와 통사구조를 고려한 벡터화의 비교: 10개의 동형어 명사

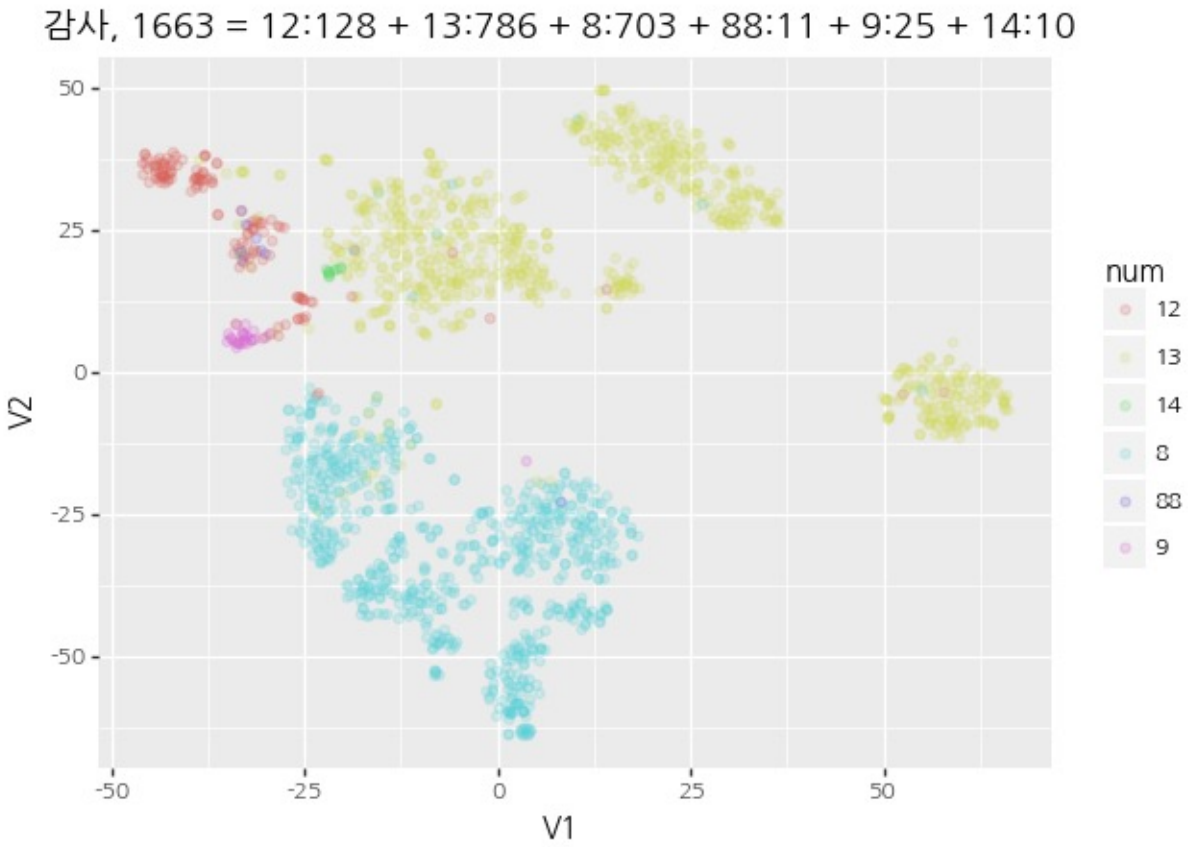




3.3. KorBERT를 이용한 실험

- 세종 형태의미분석 말뭉치에 나타나는 명사 '감사'의 1660여개의 용례에 대해, KorBERT를 이용하여 벡터화
- ETRI 형태소분석기의 오류로 인해 '감사'를 보통명사로 제대로 태깅하지 못한 경우도 있음.
  - 이런 경우 문장 전체의 벡터를 구했음.
- 같은 색깔로 표시된 하나의 의미의 예문들이 둘 이상의 덩어리로 쪼개져 있음.
  - 단어 벡터를 구한 경우와 문장 벡터를 구한 경우가 있어서 그런 듯.
  - 여러 색깔의 관측점들이 뒤섞이지는 않았기 때문에, 동형어의 중의성 해소에는 지장이 없음.

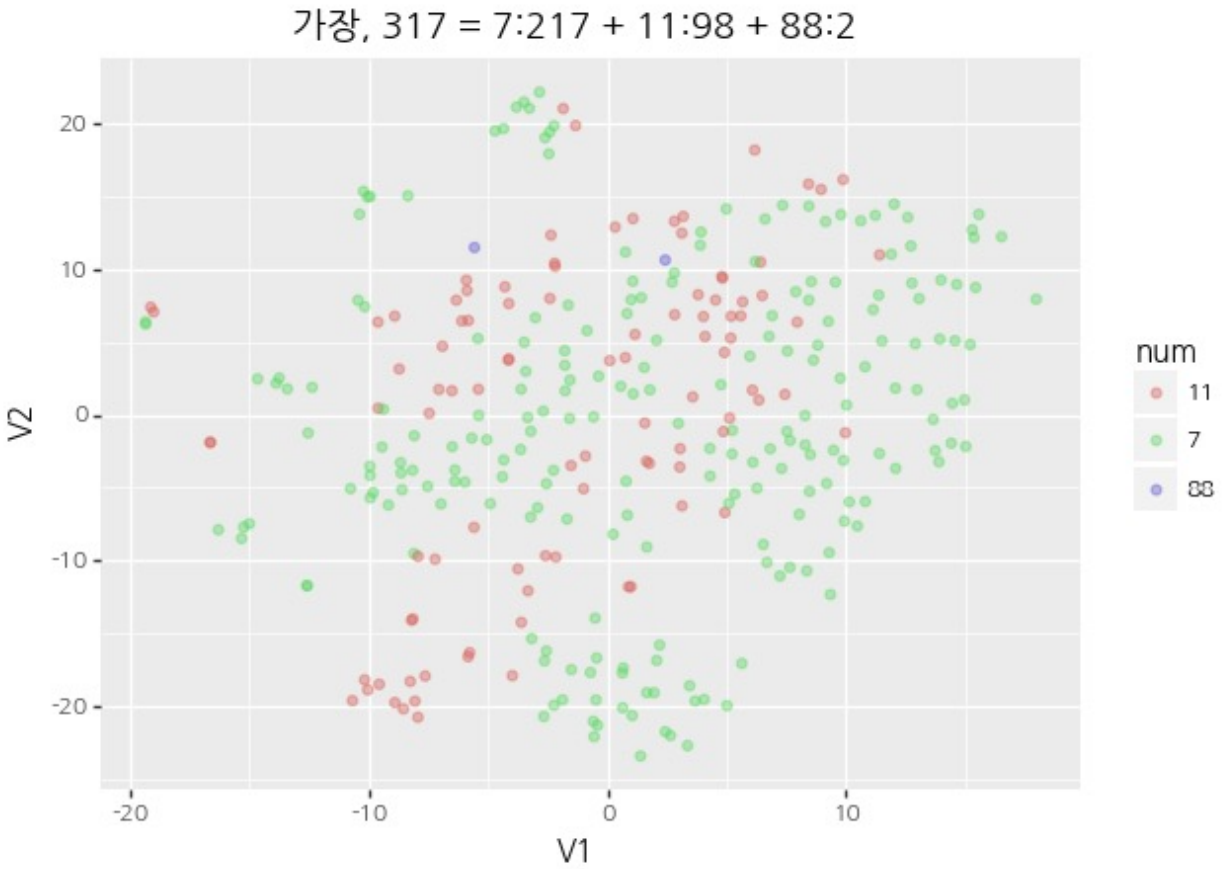
<그림 16> '감사'의 벡터화: KorBERT



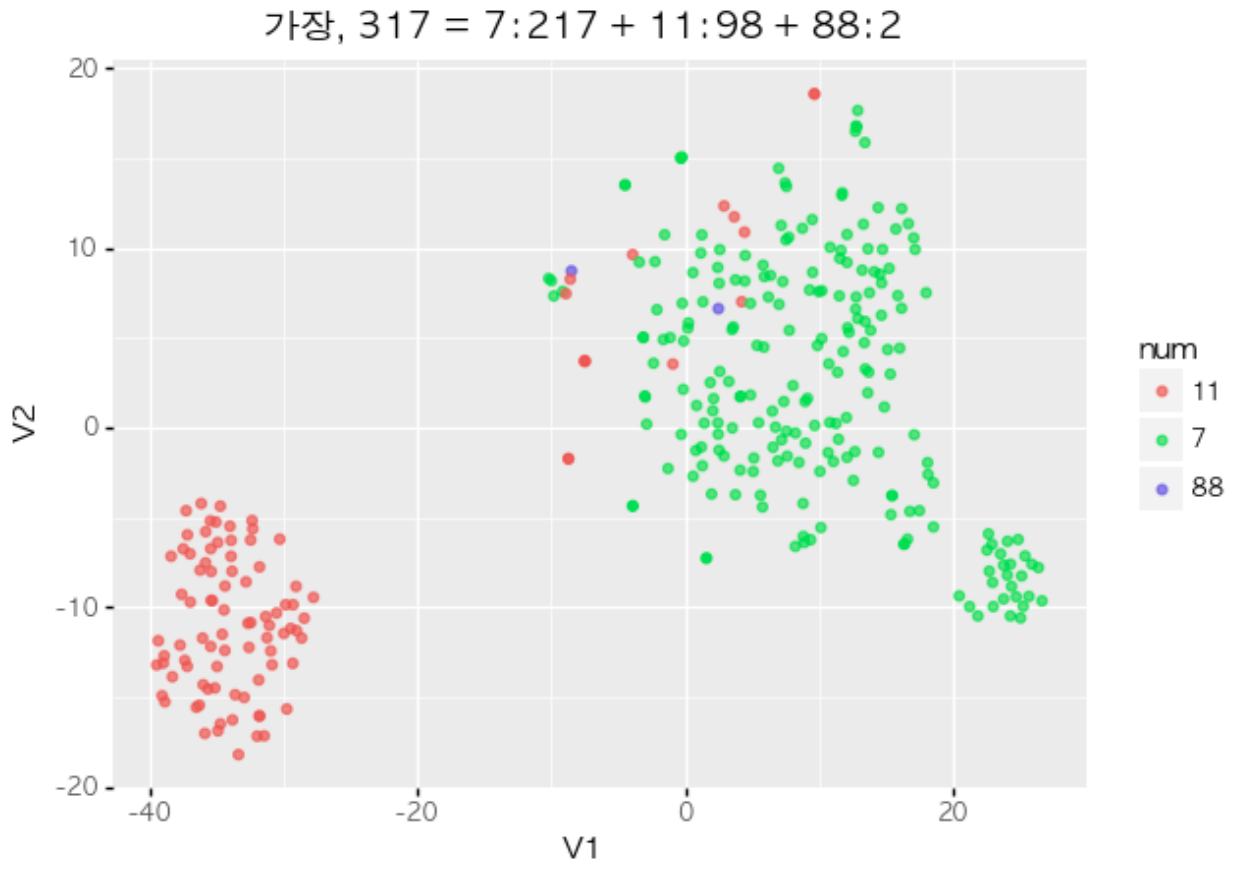
'가장'의 벡터화

- 문장을 길게 끊었을 때는 용법들이 명확히 구분되지 않으나(좌측)
- 문장을 짧게 끊으면 용법들이 잘 구분됨(우측).

<그림 17> '가장'의 벡터화: KorBERT (문장 길게)



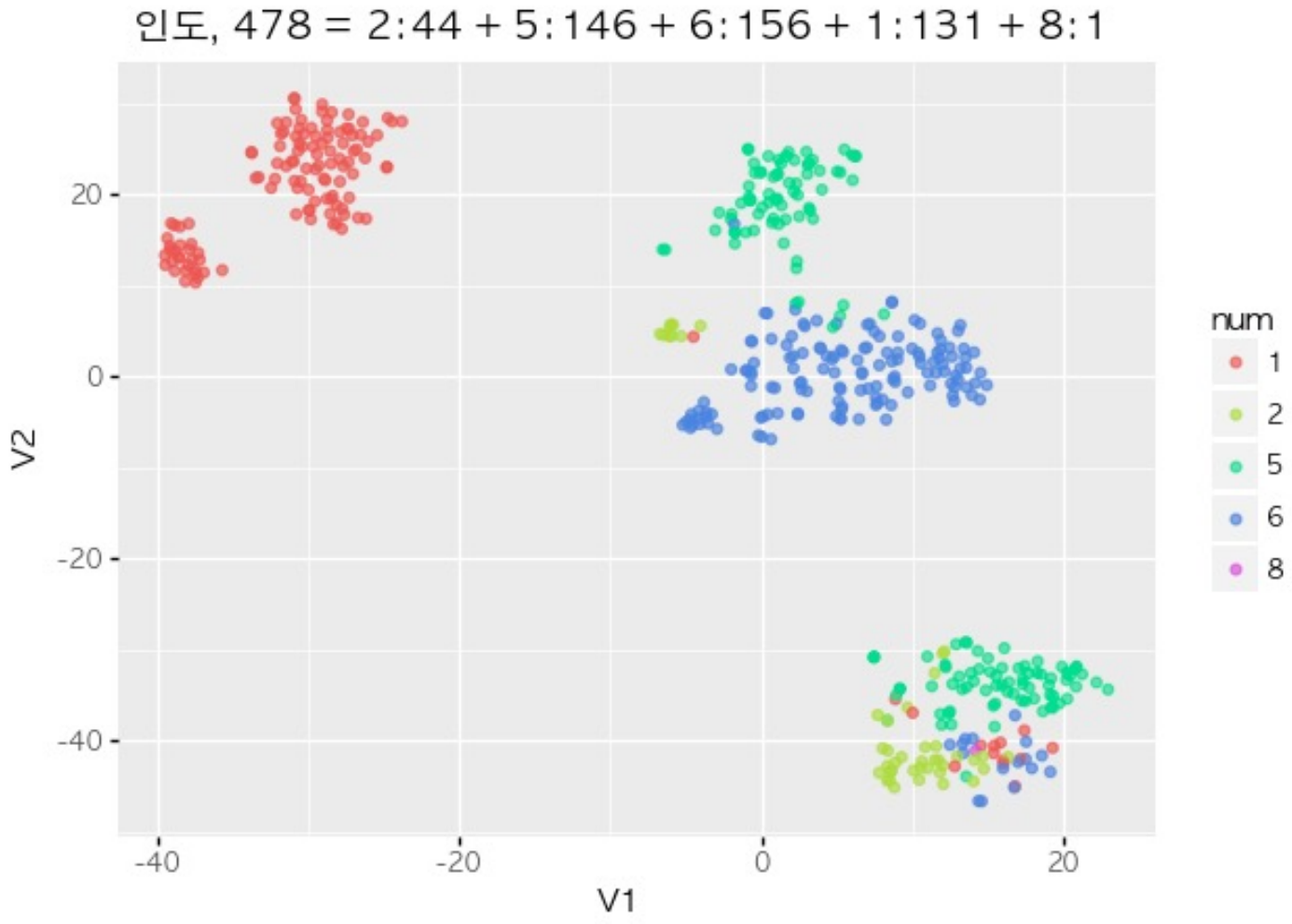
<그림 18> '가장'의 벡터화: KorBERT (문장 짧게)



‘인도’의 벡터화

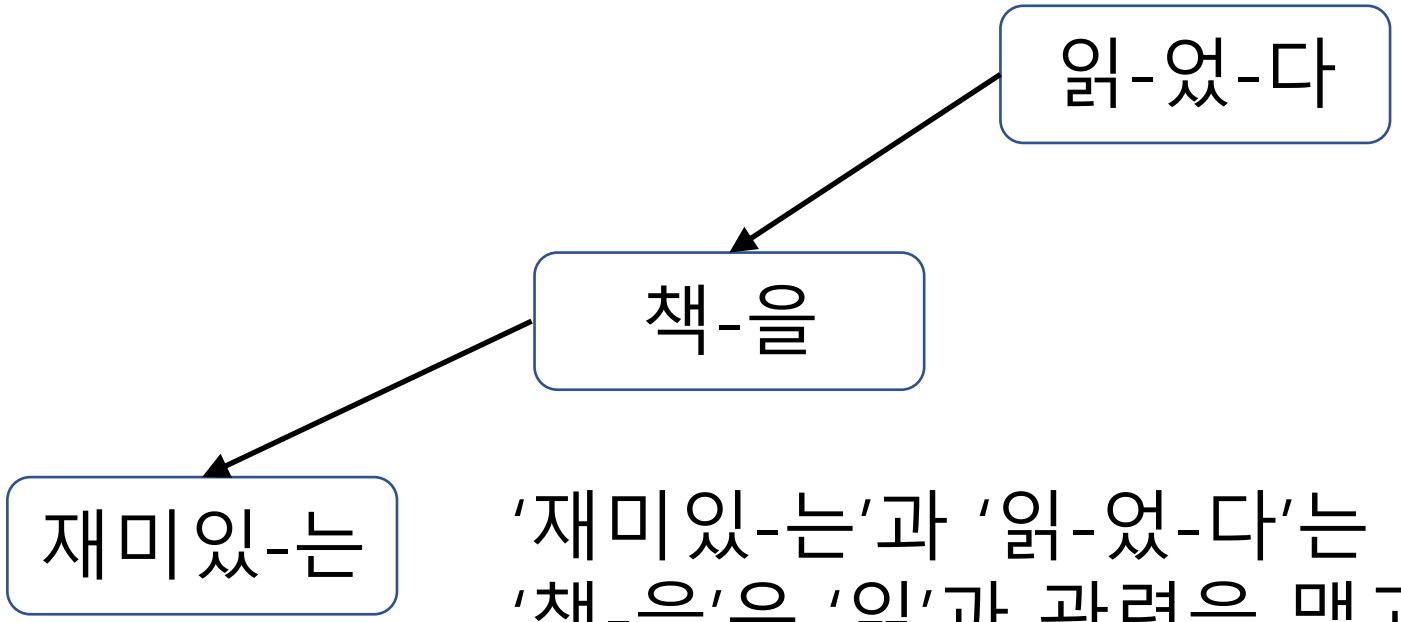
- 각 용법들이 상당히 멀리 떨어져서 잘 구분됨.
- 다만 같은 색깔의 군집이 둘 이상으로 쪼개지는 현상은 여전히 나타남.

<그림 19> ‘인도’의 벡터화: KorBERT



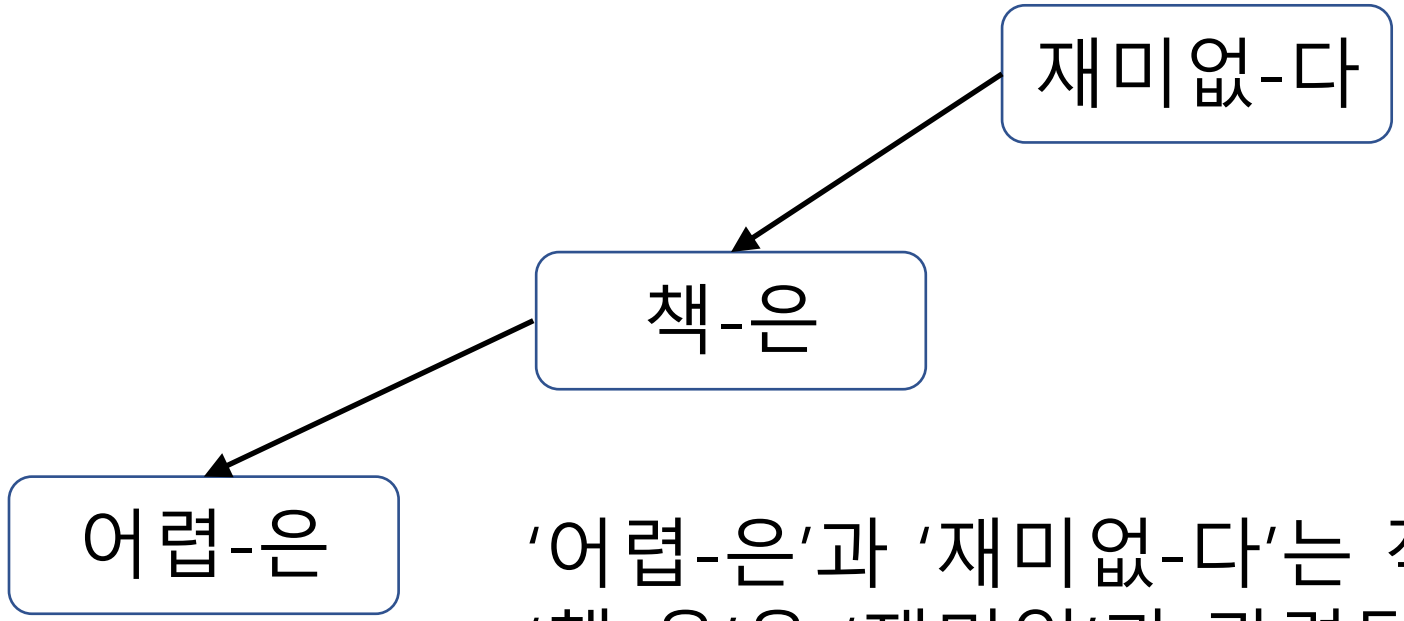
KorBERT를 이용한 벡터화에 대한 평가

- KorBERT를 이용했을 때는 굳이 구문분석 정보를 활용하지 않아도 동형어 구분이 상당히 잘 됨.
  - 아마도 BERT 모델 훈련 과정에서 문장의 통사 구조에 대한 정보를 모델이 어느 정도 섭취했을 가능성이 있음.
- 그렇기는 하지만 BERT 모델은 훈련시에 문장의 길이(토큰의 수)가 길어짐에 따라 길이의 제곱에 비례해서[quadratically] 훈련 시간이 길어짐.
- 이러한 부담을 줄이기 위해서는 문장 내의 모든 토큰들 사이의 관계를 다 고려하기보다는, 중요한 관계가 뽑아서 고려할 수 있는 방법이 필요함.
  - 구문분석 정보를 활용해서 타깃 토큰과 통사적으로 긴밀한 관계에 있는 토큰만 고려하면 됨.



‘재미있-는’과 ‘읽-었-다’는 직접적인 관련이 없음.
‘책-을’은 ‘읽’과 관련을 맺고, ‘었, 다’와는 관계 없음.
‘재미있-는’은 ‘책’과 관련을 맺고 ‘을’과는 관계 없음.

고려할 관계: ‘책, 을, 읽’ 셋 사이의 관계
‘읽, 었, 다’ 셋 사이의 관계
‘재미있, 는, 책’ 셋 사이의 관계
고려해야 할 관계의 수가 $7^2=49$개에서 $3^2+3^2-1+3^2-1=25$개로 줄어듦.



‘어렵-은’과 ‘재미없-다’는 직접적인 관련이 없음.
‘책-은’은 ‘재미없’과 관련될 뿐, ‘다’와는 관계 없음.
‘어렵-은’은 ‘책’과 관련될 뿐, ‘은’과는 관계 없음.
고려해야 할 관계: ‘책, 은, 재미없’ 셋 사이의 관계
‘재미없, 다’ 둘 사이의 관계
‘어렵, 은, 책’ 셋 사이의 관계

고려해야 할 관계의 수가 $6^2=36$개에서 $3^2+2^2-1+3^2-1=20$개로 줄어듦.

3.4. AutoML을 이용한 벡터의 분류

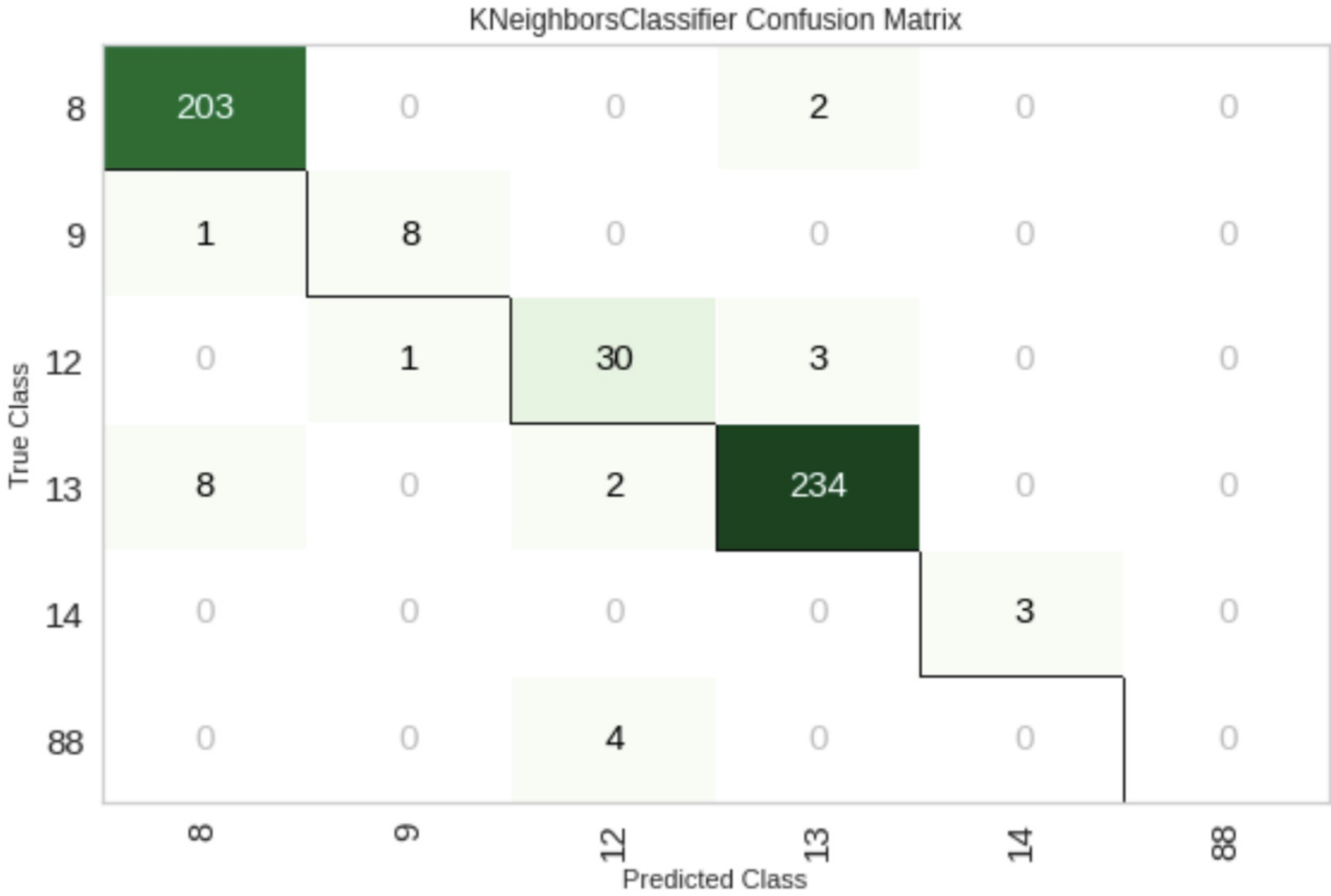
- 동형어의 벡터들이 잘 구분되어 있으면, 여기에 기계학습 알고리즘(이 경우 분류 알고리즘)을 적용하여 매우 높은 정확도로 중의성 해소를 할 수 있음.
- KorBERT는 각 토큰과 문장을 768차원의 벡터로 나타냄.
- '감사'의 1,660여개 예문에 대해 PyCaret이라는 AutoML 라이브러리를 이용하여 다양한 기계학습 기법을 적용해 본 결과 KNN(K-Nearest-Neighbor) 기법이 가장 좋은 성능을 보임.

<그림 20> '감사'가 포함된 문장들을 벡터화한 뒤 PyCaret으로 분류 기법들을 적용한 결과

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
knn	K Neighbors Classifier	0.9682	0.3955	0.8486	0.9636	0.9656	0.9464	0.9466	0.1270
lightgbm	Light Gradient Boosting Machine	0.9656	0.3982	0.8087	0.9596	0.9620	0.9417	0.9420	2.5500
lr	Logistic Regression	0.9631	0.3983	0.8463	0.9585	0.9604	0.9375	0.9378	0.2540
nb	Naive Bayes	0.9605	0.3951	0.8660	0.9604	0.9590	0.9331	0.9337	0.0950
svm	SVM - Linear Kernel	0.9588	0.0000	0.8368	0.9569	0.9570	0.9301	0.9306	0.1020
ridge	Ridge Classifier	0.9570	0.0000	0.8288	0.9532	0.9546	0.9273	0.9277	0.0190
et	Extra Trees Classifier	0.9570	0.3983	0.7124	0.9482	0.9505	0.9260	0.9272	0.1060
lda	Linear Discriminant Analysis	0.9502	0.3918	0.8590	0.9472	0.9481	0.9158	0.9162	0.0440
rf	Random Forest Classifier	0.9485	0.3975	0.6809	0.9384	0.9415	0.9114	0.9126	0.1500
gbc	Gradient Boosting Classifier	0.9433	0.3957	0.6518	0.9448	0.9427	0.9039	0.9047	9.6980
dt	Decision Tree Classifier	0.9064	0.3673	0.6738	0.9133	0.9070	0.8430	0.8445	0.1450
ada	Ada Boost Classifier	0.8368	0.3345	0.4510	0.7723	0.7968	0.7063	0.7210	0.3450
dummy	Dummy Classifier	0.4656	0.2000	0.1867	0.2168	0.2959	0.0000	0.0000	0.0150
qda	Quadratic Discriminant Analysis	0.3943	0.2252	0.3180	0.2632	0.3122	0.1021	0.1253	0.0350

```
KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
                     metric_params=None, n_jobs=-1, n_neighbors=5, p=2,
                     weights='uniform')
```

<그림 21> 동형어 '감사'의 중의성 해소에 KNN을 적용한 결과 (혼동행렬)



4. 맺는 말

- 문장 내에서 함께 공기하는 단어들은 서로 영향을 주고받아서 서로 어울리게 끔 의미 해석이 이루어짐: isotopy.
- Word2Vec처럼 단어의 의미를 고정된 벡터로 표상하는 언어 모델은 isotopy를 포착할 수 없음.
- BERT는 attention mechanism에 의해 문장 내 모든 단어의 영향을 받게끔 단어 벡터를 조정하므로, isotopy를 포착할 수 있는 길을 열었음.
- 동형어의 중의성을 해소할 때, 단어와 문장을 벡터화하는 여러 기법들이 유용하게 쓰일 수 있는데, 구문분석 정보를 활용하면 더 좋은 결과를 얻을 수 있음.
- BERT 모델은 훈련 시에 이미 문장의 통사 구조에 대한 정보를 어느 정도 섭취한 것으로 추측되나, 문장이 길어질수록 훈련 시간이 너무 오래 걸린다는 단점이 있음.
- 문장 내에서 모든 단어들이 서로 대등한 정도로 상호작용하는 것은 아님.
  - 더 긴밀히 상호작용하는 단어 쌍도 있고, 상호작용이 미미한 단어 쌍도 있음.
  - 이러한 차이를 포착하려면 역시 구문분석을 활용해야 함.
  - 구문분석을 활용하면 (중의성 해소를 더 잘 할 수 있다는 장점뿐 아니라) BERT 모델의 훈련 시간을 단축한다는 경제적 측면에서의 이점도 있음.