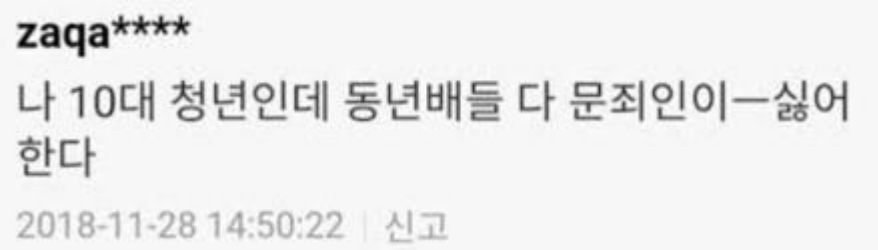
자연어처리 및 정보검색 보고서

Natural Language Processing & Information Retrieval

어연자 팀	
학 번	**이 름**
20153378	이 로 히
20150804	정 민 준
20131699	정 인 석

목 차

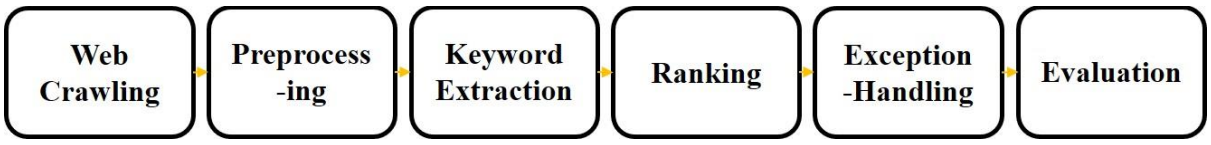
1. 서 론



위의 사진은 최근에 화재가 된 네이버 정치 뉴스의 한 댓글이다. 글쓴이는 자신이 10대임을 주장하고 있지만 누가 읽어도 40대 이상으로 보이는 어휘인 청년, 동년배 등을 사용하고 있다. 사용하는 말투와 선택하는 단어를 통해 나이대를 추청할 수 있을 것이라 생각하여 댓글 연령 판별 시스템을 구상하게 되었다.

주제: 커뮤니티의 댓글을 쓴 사용자의 연령대를 판별해주는 프로그램 구현한다.

2. 구현 계획



1. web crawling과정을 통해 인터넷 커뮤니티에서 나이가 기록된 댓글 데이터를 수집한다.

2. preprocessing과정을 통해 데이터 가운데 한국어 명사만을 추출하는 전처리한다.

3. keyword extraction과정을 통해 TF-IDF 개념을 이용해서 각 나이대별 키워드를 추출한다.

4. ranking과정을 통해 query에 대해서 나이대를 추정한다.

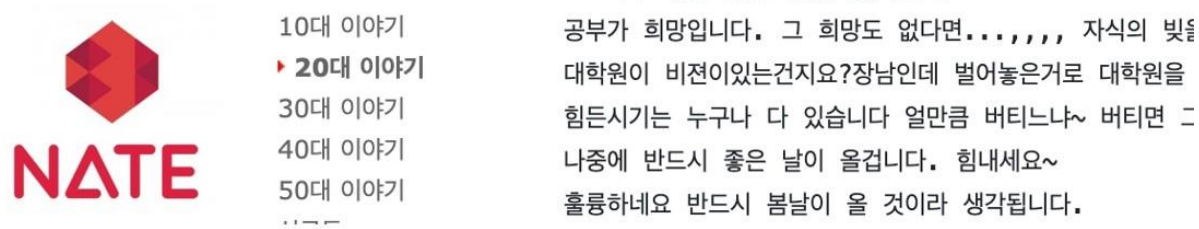
5. 학습되지 않은 새로운 단어가 들어왔을 때 처리하는 과정을 수행한다.

6. 우리가 만든 시스템을 precision을 계산하여 평가한다.

4. 구현 설명

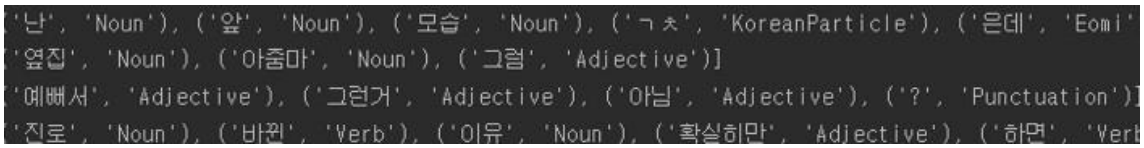
4-1. Web Crawling

데이터 수집을 위하여 데이터 웹 크롤링 작업을 진행했다. 연령대 별로 나뉘여 있는 네이트판의 10대 게시판, 20대 게시판, 30대 게시판, 40대 게시판, 50대 게시판에 있는 댓글들을 대상으로 데이터를 수집했다. 크롤링 과정에서는 파이썬의 bs4, selenium 모듈을 사용하였다. 크롤링 결과는 각 연령대별로 텍스트 파일을 만들어 저장했다.



4-2. Preprocessing

전처리는 텍스트파일을 불러와 한줄씩 읽으며 Twitter 파이썬 패키지를 이용하여 품사를 분석한다. 결과적으로 나이대별로 키워드를 담은 리스트를 Counter 자료구조를 이용해서 각 나이대별로 사용하는 어휘들의 개수를 함께 저장한다.



4-3. Keyword Extraction

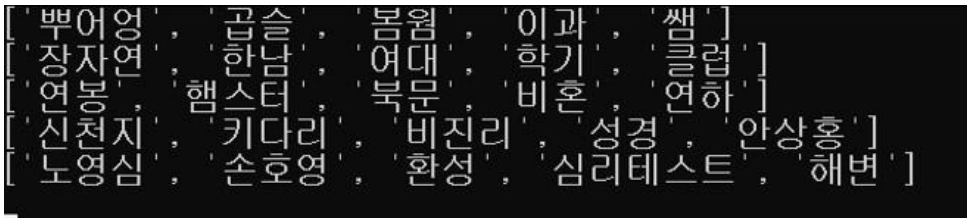
TF-IDF 개념을 응용해서 키워드를 추출했다. 인덱싱은 다음과 같이 문서에 등장하는 모든 단어에 인덱스 넘버를 부여해서 딕셔너리 형태로 저장했다.

TF와 IDF는 자연어처리 및 정보검색 시간에 학습한 식을 바탕으로 계산하여 리스트 형태로 저장했다. TF와 IDF값을 곱하여 각 문서에서 단어가 차지하는 weight값을 구하였다.

$$tf_{ij} = f_{ij} / max_i\{f_{ij}\}$$
$$idf_i = \log_2 (N/df_i)$$
(N: total number of documents)

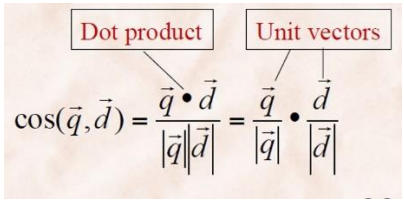
그리고 문서별로 weight값이 높은 것을 출력하여 확인했다. 아래는 문서를 대표하는 키워드 top 5를 출력한 화면이다. 위에서부터 10대 20대 30대 40대 50대 순서이며, 각 연령을 대표하는 키워드라고 인식하였다.

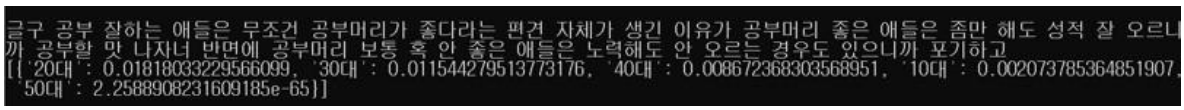
(ex. 10대는 이과, 쌤 등 학교에서 사용할 만한 어휘를 사용하였고, 20대는 대학교나 클럽 등 20대에게 친숙한 어휘가 키워드로 뽑혔다. 30대의 '연봉'도 같은 맥락으로 이해할 수 있다.)



4-4. Ranking

Vector Space Model에 query와 문서의 벡터를 위치시켜서 Cosine Similarity 를 계산하였고, query를 바탕으로 추정한 (나이대: weight값)으로 Dictionary형태로 저장하였다. 아래 캡쳐화면에 윗줄은 query text이고, 아래는 query를 바탕으로 ranking이 높은 순서대로 출력한 값이다.





4-5. Exception Handling

만약 연령대를 판별할 댓글에 저희가 가지고있지 않은 단어가 있을 경우 에러가 생기는 사실을 알게 되었고, 이 경우를 처리해주기 위해서 그 글자를 가장 유사한 단어로 대체하여 처리하는 방식으로 구현했다.

글자의 구성은 초성, 중성 그리고 종성으로 이루어져 있다. 그래서 새로운 단어를 초성, 중성, 종성으로 분해하여 리스트의 단어들과 비교하여 가장 비슷한 단어로 대체하였다.

ex) 감의 초성은 ㄱ 종성은 ㅏ 종성은 ㅁ

다음 사진은 새로운 단어가 들어왔을 때 대체한 단어들의 자료이다. 단어의 순서에 맞게 초성, 중성, 종성을 비교하여 가장 근접한 단어를 찾아 출력한 결과이다.

4-6. Evaluation

평가는 input_data와 test_data를 바탕으로 Precision을 계산하였다.

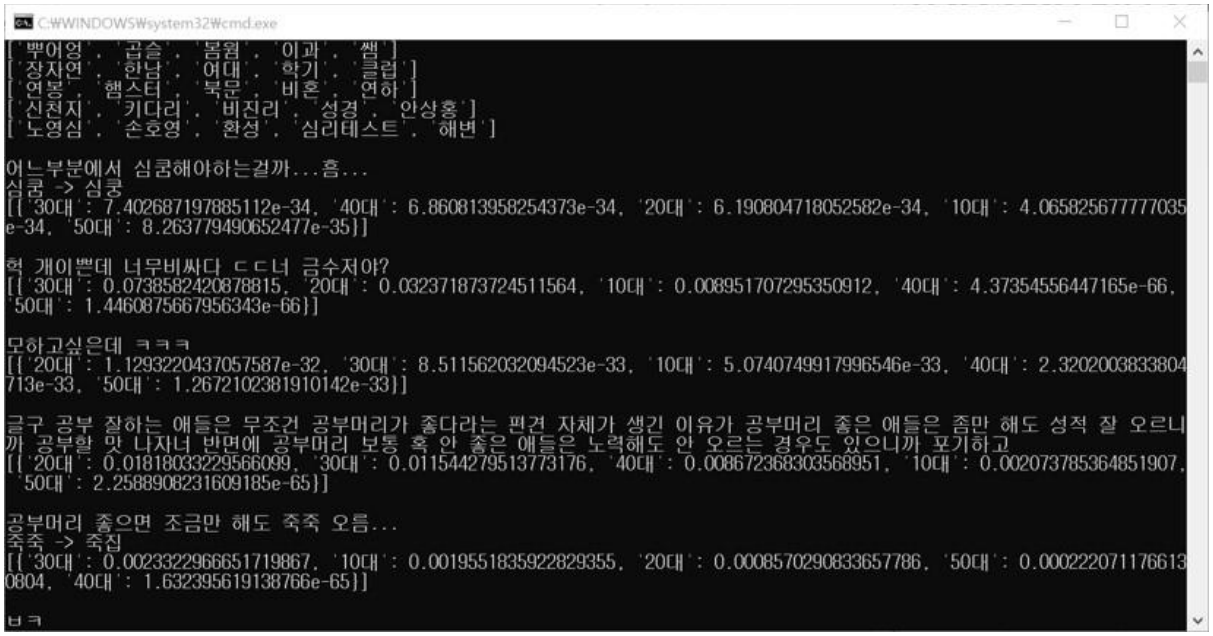
학습된 데이터를 넣고 돌렸을 때 정확도는 다음과 같다.

10대의 정확도는 약 55%, 20대의 정확도는 66%, 30대는 67%, 40대는 60%, 50대는 30%로서 37855개의 댓글 가운데 약 62%에 해당하는 Precision을 계산하였다.

테스트 데이터를 넣고 돌렸을 때 정확도는 다음과 같다.

10대의 정확도는 약 32%, 20대의 정확도는 43%, 30대는 45%, 40대는 34%, 50대는 32%로서 926개의 댓글 가운데 약 37%에 해당하는 Precision을 계산하였다.

5. 실행 결과



6. 결론

6-1. 아쉬운 점

우리가 구현한 나이 추정 시스템에서 아쉬운 점은 크게 두가지가 있다.

첫번째로 데이터셋 부분이다. 데이터셋이 특정 성향을 가진 집단의 데이터로 구성되어 있기 때문에 일반적인 속성을 반영하지 못했다는 점과 10, 40, 50대의 데이터가 20~30대에 비해 너무 적다는 점, 자신의 나이대에 맞지 않는 게시판을 사용하는 경우가 있었다.

다양한 사람들이 존재하는 플랫폼에서 나이대를 알 수 있는 댓글로 구성된 데이터셋을 얻게 된다면 좀 더 일반적이고 포괄적인 나이 추정 시스템을 구축할 수 있을 것이다.

두번째는 기술적인 한계점이다.

저희 시스템에서 단어를 형태소로 나누어 모음 자음의 조합이 비슷한 단어를 찾는 알고리즘을 구현하였는데 단어의 모음, 자음 구성만 비슷한 단어를 찾을 뿐 의미상 유사한 단어를 찾는 경우는 적었다. 또한, 전처리 과정에서 단어의 뜻을 보존하지 못하는 형태로 단어를 분리하여 전처리 되는 경우가 빈번하게 발생하였다.

Ex) 예를 들어 '앞트임'라는 단어를 '앞'과 '트임'으로 쪼갠다거나 하는 경우

6-2. 의의

나이 추정 시스템은 한국어를 기반으로 프로젝트를 진행하여 기존의 유사한 프로젝트와 차별점을 두었으며, 인덱싱에 없는 단어가 쿼리로 들어왔을 경우 형태소로 쪼개어 단어를 분석하는 알고리즘을 완성하였다.

한국어 특성상 오타가 많아도 의미 전달이 가능한 경우가 있는데 이 알고리즘에서는 오타가 포함된 단어가 쿼리로 들어왔을 때도 원래 단어를 찾을 수 있었다.

커뮤니티의 특성상 댓글에는 많은 양의 오타를 포함하고 있는데 머신러닝을 사용하지 않고도 오타를 포함한 단어를 원래 단어로 바꿀 수 있다는 것은 매우 유용하다고 생각한다. 또한 저희 주제에 관한 자료가 굉장히 부족하였는데, 수업시간에 배운 VSM을 기반으로 약 40%의 정확도를 지닌 시스템을 구현하였다.