석사학위 논문

딥러닝을 이용한 SNS 분석기반 개인 관심사 추출

Extraction of individual interests based on SNS analysis using Deep Learning

2018년 6월

숭실대학교 정보과학대학원

소프트웨어공학과

김 형 태

석사학위 논문

딥러닝을 이용한 SNS 분석기반 개인 관심사 추출

Extraction of individual interests based on SNS analysis using Deep Learning

2018년 6월

숭실대학교 정보과학대학원

소프트웨어공학과

김 형 태

석사학위 논문

딥러닝을 이용한 SNS 분석기반 개인 관심사 추출

지도교수 홍 지 만

이 논문을 석사학위 논문으로 제출함

2018년 6월

숭실대학교 정보과학대학원

소프트웨어공학과

김 형 태

김 형 태 의 석 사 학 위 논 문 을 인 준 함

심 사 위 원 장 정 기 철 인

심 사 위 원 허 준 영 인

심 사 위 원 홍 지 만 인

2018년 6월

숭실대학교 정보과학대학원

목 차

표 목 차

그 림 목 차

국문초록

딥러닝을 이용한 SNS 분석기반 개인 관심사 추출

김 형 태

소프트웨어공학과

숭실대학교 정보과학대학원

최근 스마트폰 보급률이 높아짐에 따라 SNS를 이용하는 사람들이 급증하고 있다. 이러한 SNS의 사용은 초창기에는 인간관계 형성 및 유지를 목적으로 이용되었으나 최근에는 개인의 생각이나 좋아하는 사진을 게재하는 등 직, 간접적으로 개인의 관심사가 노출되고 있다. 따라서 SNS를 분석하는 것은 개인의 성향 및 관심사를 추출하기에 좋은 데이터가 되며 이러한 분석결과는 마케팅, 추천시스템 등 많은 분야에 활용될 좋은 자료가 된다.

본 논문에서는 구글의 PageRank 알고리즘을 응용한 TextRank 알고리즘을 이용하여 SNS에서 키워드를 추출 한 후 그 내용을 CNN (Convolutional Neural Network)을 이용하여 분류하여 SNS내용을 바탕으로 개인관심사를 추출하는 방법을 제안한다.

기존의 분류기법을 이용한 결과 평균 70.4%의 정확도를 보이나 제안한 방법은 정확도가 약 82.8%로 크게 향상되었고 개인 관심사를 추출하는데 보다 효율적임을 보인다.

ABSTRACT

Extraction of individual interests based on SNS analysis using Deep Learning

Kim, Hyeong-Tae

Department of Software Engineering

Graduate School of Information Sciences

Soongsil University

Recently, the number of people using social network sites(SNS) is increasing rapidly as the penetration rate of smartphones has increased.

Although SNS was used to build and maintain relationships in the beginning, but it has recently been used to express personal interests directly and indirectly, such as posting personal thoughts or favorite photos.

Therefore, analyzing SNS is good data to extract personal tendencies and interests, and these analysis results can be used in many ways, including marketing and recommendation system.

In this thesis, contents of SNS are summarized using algorithm of Text Rank, and classify the contents using CNN(Convolutional Neural Network) to extract individual interests.

While Conventional CNN(Convolutional Neural Network) - based classification techniques show an average of 70.4% accuracy, the proposed method has significantly improved accuracy to about 82.8% and appears to be more efficient in extracting individual interests.

제 1 장 서 론

1.1 배경 및 목적

최근에 모바일 디바이스(스마트폰, 태블릿 등)의 보급률이 크게 늘어남에 따라 인터넷 이용이 간편해졌다. 이에 따라 다양한 형태의 소셜 네트워크서비스(SNS, Social Network Service) 가 나타났고 SNS이용자가 크게 늘어났다. SNS의 종류에는 텍스트 위주의 페이스북, 트위터가 있고 이미지 위주의 인스타그램이 있다.

이러한 SNS의 사용은 초창기에는 개인의 인맥이나 사회적 관계강화에 주로 사용되었으나 최근에는 정보 및 개인의 관심사를 공유할 수 있는 플랫폼으로 진화하였다. SNS 사용자들은 친구관계 등의 인맥뿐만 아니라 텍스트, 사진, 동영상등을 만들고 공유한다.

따라서 SNS에는 개인의 의견과 같은 단문 Text에서부터 이미지, 동영상 등 자연스럽게 개인의 관심사가 표출되고 있는 실정이다. SNS에서 습득할 수 있는 정보를 이용하여 사용자의 특성이나 관심사 등을 판별하고 사용자에게 맞춤형 서비스를 제공하기 위한 연구가 활발하게 이루어지고 있으나 단편적인 방법으로 정확도가 떨어지는 부분이 존재한다.

본 논문에서는 구글의 PageRank 알고리즘[8]을 응용한 TextRank 알고리즘을 이용하여 SNS에서 키워드를 추출 한 후 CNN(Convolutional Neural Network)으로 분류하여 정확도를 높이는 방법을 제안한다.

1.2 연구내용 및 구성

본 논문에서는 SNS게시물을 이용하여 사용자의 관심사를 추출하는 방법을 제안하고 기존 연구와의 정확도 차이를 분석한다.

2장에서는 기존 연구들에 대해 살펴보고 본연구의 이론적 배경이 되는 구글의 PageRank 알고리즘 및 이를 바탕으로 텍스트 키워드 추출에 사용할 TextRank 알고리즘, CNN(Convolutional Neural Network) 에 대해 살펴본다. 또한 기계학습을 위해 구글(google)에서 공개한 OpenSource Library 인 Tensorflow의 동작원리 및 사용법을 간단히 살펴본다.

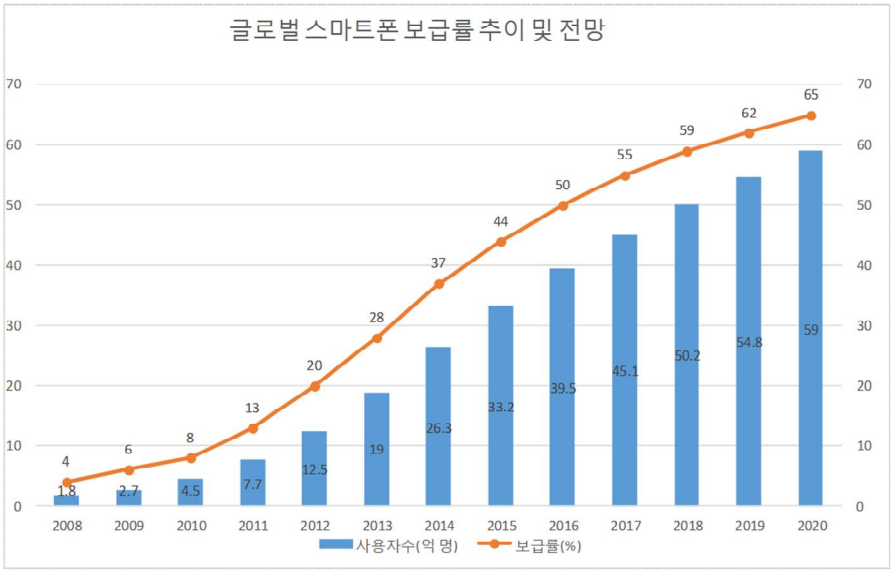
3장에서는 키워드 추출 방법에 대해 기술하고 요약된 내용을 바탕으로 CNN(Convolutional Neural Network)을 통해 분류하는 방법을 기술한다. 키워드 추출을 위해 TextRank 알고리즘을 구현하여 SNS내용을 요약하고 CNN(Convolutional Neural Network)으로 분류하여 실제 사용자의 관심사를 구분해내는 방법을 제안한다.

4장에서는 실험 결과에 대해 정확도를 분석하고 기존연구와의 차이점을 비교하여 문제점에 대해 파악한다.

5장에서는 결론 및 향후 연구에 관해 서술한다.

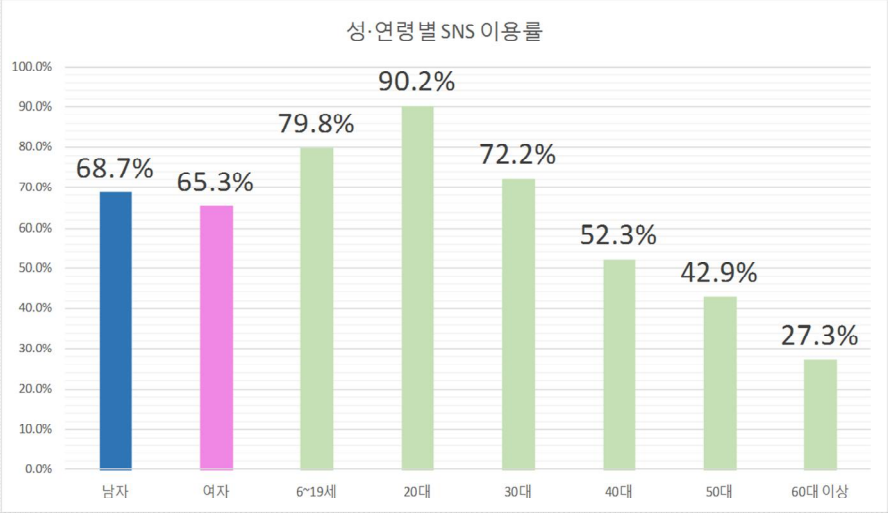
제 2 장 관련 연구

아래 [그림 2-1]에서 알수 있듯이 최근 모바일 기기 보급량이 크게 증가하고 있고 이에 따라 모바일 인터넷 사용도 간편해졌다.



[그림 2-1] 글로벌 스마트폰 보급률 추이 및 전망

스마트폰 보급이 늘면서 사용자의 SNS(Socail Network Service) 사용률도 크게 늘고 있으며 이는 10대~30대 뿐만 아니라 모든 연령대에 고루 나타나고 있다[그림 2-2].



[그림 2-2] 성.연령별 SNS 이용률

따라서 SNS에 공개되는 정보는 매우 다양하며 그 안에 사용자 개인의 관심사가 담겨 있어 이를 추출하여 분석하면 다양한 분야에 사용할 수 있기 때문에 최근 다양한 방법들이 연구되고 있다.

SNS에서 특정 키워드로 게시글을 수집하여 이를 학습한 후 분류하여 동일한 분류 카테고리 내의 사용자를 추천하는 방식을 제안하고 있다[1].

영화 리뷰 사이트에 게시된 댓글과 평점 정보를 이용해 각 리뷰가 긍정인지 부정인지 CNN(Convolutional Neural Network)을 이용하여 분장을 분류하는 기법을 소개하고 있다[2].

사용자가 작성한 글과 친구관계인 사용자가 주고받은 내용을 기반으로 카테고리화 함으로써 사용자의 관심사를 자동으로 분류 및 추천하는 방법을 제안한다[3].

SNS 단문 텍스트의 특성과 주요 단어의 노출 빈도수의 상관성 분석을 통해 SNS를 카테고리 별로 분류하는 방법을 제안하였다[4].

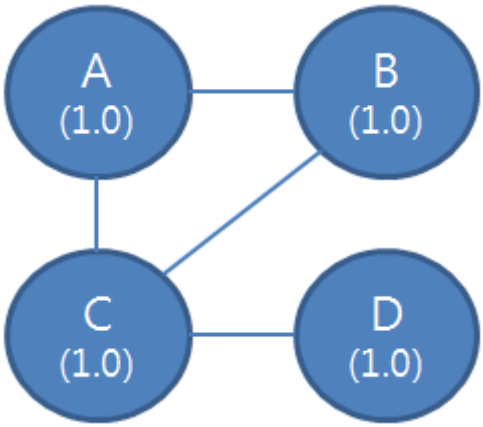
인터넷을 통해 생성되는 많은 대용량 텍스트 데이터들을 CNN이용하

여 분류하는 방법을 제안하였다[7].

2.1 PageRank 및 TextRank 알고리즘

PageRank는 각각의 페이지를 Node로 보고, 페이지와 페이지를 연결하는 링크를 Edge로 하여 만들어진 그래프를 대상으로 하는 알고리즘이다.

페이지 A, B, C, D 4개가 있고, A와B, A와C, B와C 그리고 C와D가 연결되어 있다고 하고 먼저 모든 정점에 1의 가중치를 준다.(방향성이 없다고 가정한다.)



[그림 2-3] 페이지별 가중치 표시

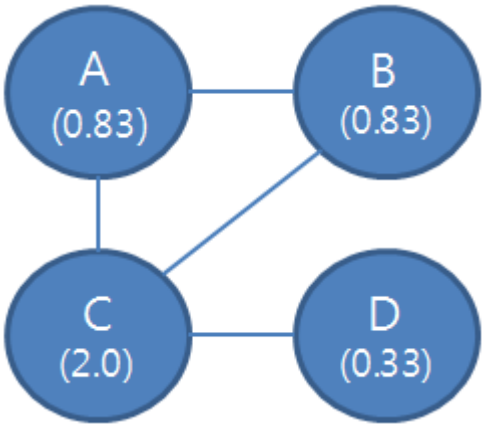
다음 단계로 각 정점(페이지)이 가지고 있는 가중치를 연결된 정점(페이지)에게 나눠준다.

즉 페이지 A의 가중치는 1이고 페이지 B와 페이지 C가 연결되어 있으니 각각 0.5, 0.5씩 넘겨준다. 마찬가지로 페이지 B의 가중치도 1이었는데 페이지 A, 페이지 C에게 0.5씩 넘겨준다. D는 C에게만 연결되어있으므로 C에게 1 전체를 넘겨준다. C는 A, B, D에게 연결되어있으므로 0.333 씩 각각 넘겨준다.

각 페이지의 새로운 가중치 값은 연결된 페이지로부터 넘겨받은 가중

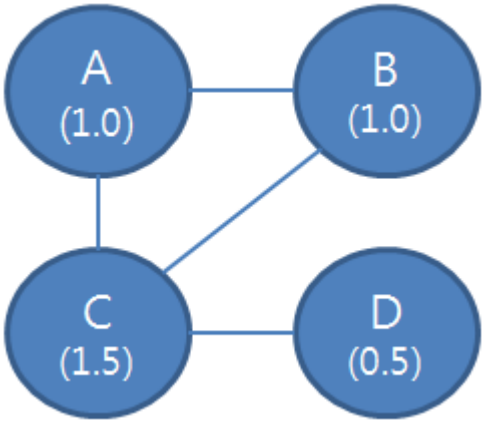
치들의 합이 된다. 페이지 A는 페이지 B로부터 0.5를 받았고, 페이지 C에게 0.333을 받았으므로 0.833이 된다.

나머지 페이지에 대해서도 같은 방식으로 계산을 해서 모든 페이지의 가중치를 업데이트 하면 아래 [그림 2-4] 와 같다.



[그림 2-4] 페이지별 가중치 업데이트

이 단계를 반복하다보면 가중치들이 점점 일정한 값으로 수렴하게 되는데 결과적으로 페이지 A 와 페이지 B는 1, 페이지 C는 1.5, 페이지 D는 0.5의 가중치를 갖게 되고 이 값이 각각의 페이지의 PageRank 값이 된다[그림 2-5].



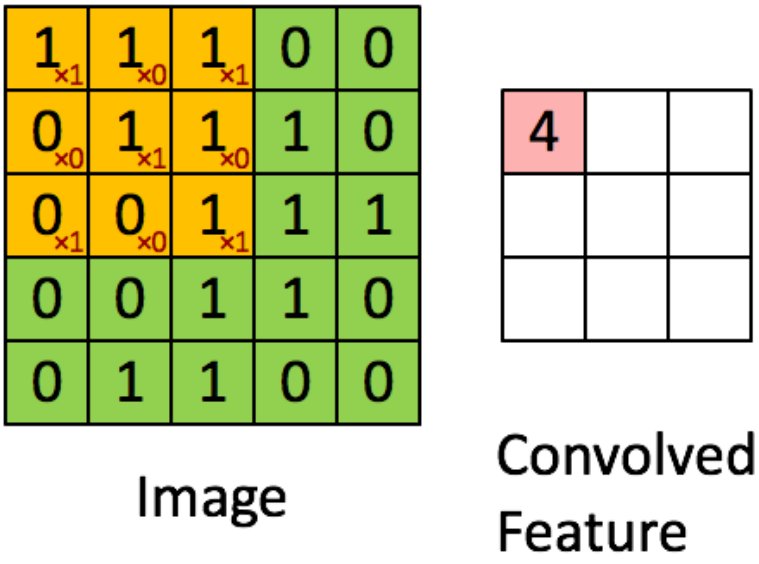
[그림 2-5] 페이지별 가중치 업데이트 최종결과

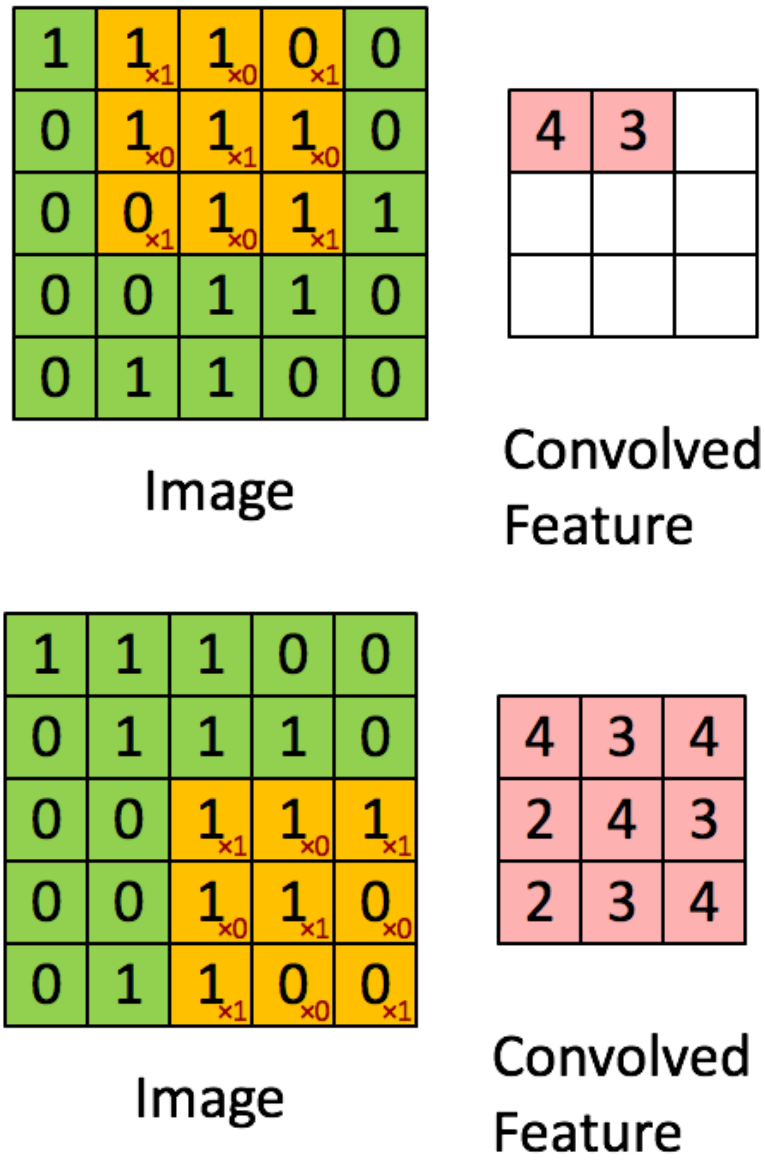
PageRank의 장점은 페이지 간 연결 상태를 가지고 쉽게 중요한 페이지를 계산해 낼 수 있다는 것이다. PageRank의 알고리즘을 텍스트 처리에 사용한 알고리즘이 바로 TextRank 이다[5].

텍스트에서 정점(Node)이 될 만한 단위(text unit) 즉 단어(Word)를 추출하고 각각의 단어를 정점(Node)으로 잡고, 같은 문장 내에서 동시 출현하는 빈도를 가지고 간선(Edge)을 구축할 수 있다. 이 경우 TextRank 값은 문장 내에서 각 단어별 중요도를 표현하게 되므로, 단어들 중 중요도가 높은 단어들을 뽑아내게 되면 이 결과가 바로 키워드 추출(Keyword Extraction)이 된다.

2.2 CNN(Convolutional Neural Network)

CNN 알고리즘은 주로 이미지의 특징을 추출하여 유사점을 찾는 이미지 판단에 사용된다. 아래 [그림 2-7]은 단계적 이미지 필터 적용방법이다[14].

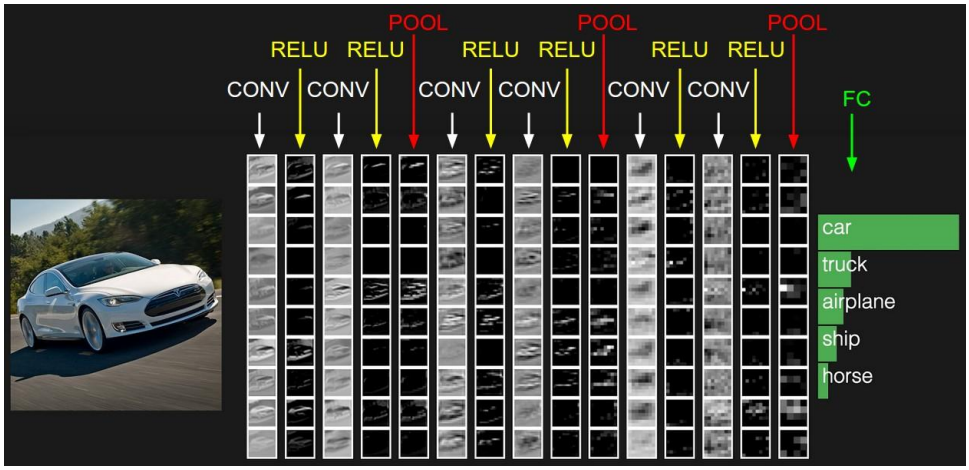




[그림 2-6] 단계별 이미지 필터 적용

위의 그림에서 왼쪽 Matrix를 흑백 이미지라 한다면 각 원소 값에서 0 은 검정, 1은 흰색으로 볼 수 있고 슬라이딩 윈도우(Sliding Window)는 3x3 필터를 사용하여 Matrix의 각 값을 곱셈하여 합산한다. 그 결과

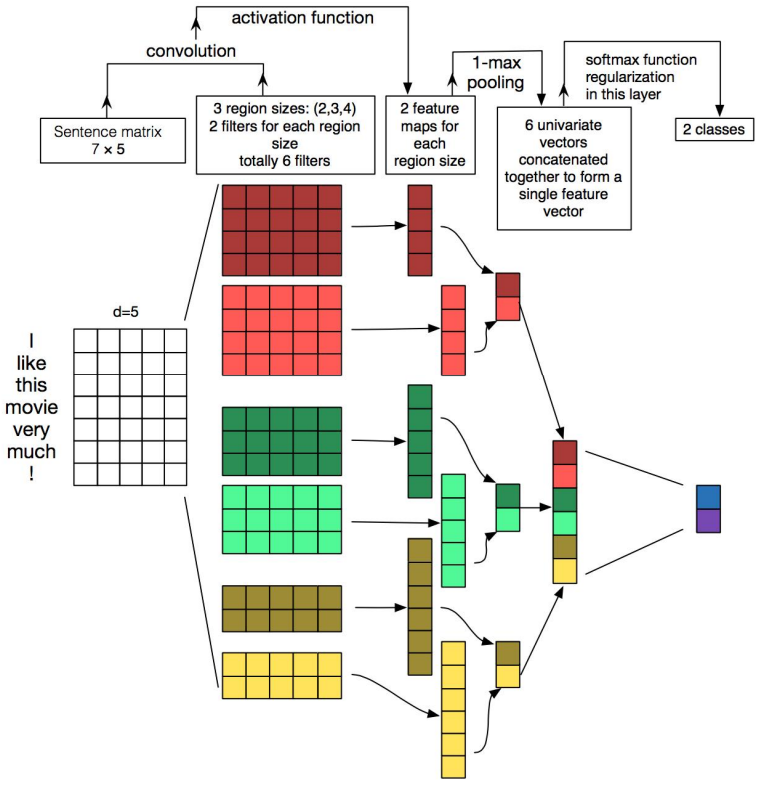
전체 Matrix에 필터를 슬라이딩한 각 엘리먼트 전체 합성곱을 얻는다.



[그림 2-7] 이미지 판별 과정

이미지 판별 과정은 위의 [그림 2-7]과 같이 각 이미지에서 필터링 된 합성곱(CONV) 과 활성화 함수(ReLU) 그리고 맥스 풀링(POOL) 과정을 반복하여 피쳐 벡터(Feature vector)를 생성한다. 그리고 피쳐 벡터와 이미 학습된 이미지를 비교하여 판별한다[15].

이미지 뿐만 아니라 자연어처리에도 CNN(Convolutional Neural Network) 알고리즘을 적용하여 높은 정확도를 낸 논문이 발표됐다[2].



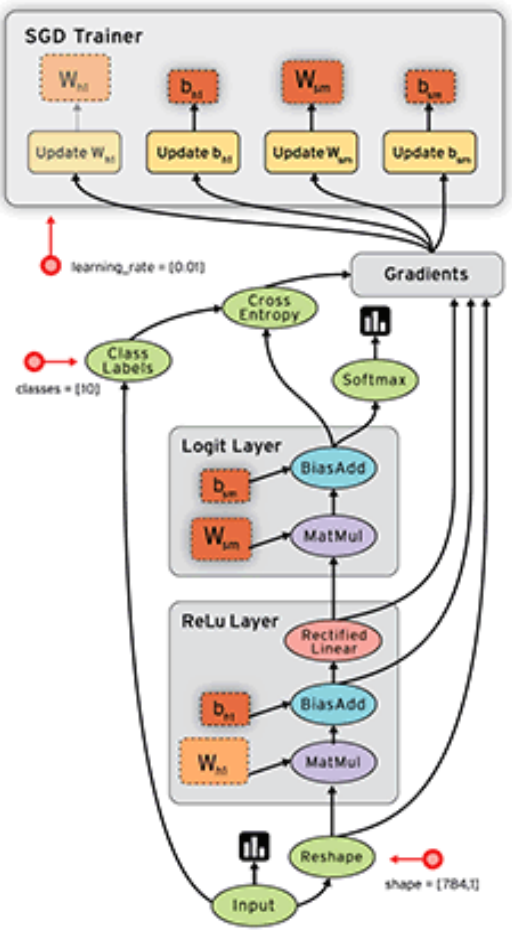
[그림 2-8] 문장 분류를 위한 CNN 아키텍처

위 이미지는 문장 분류를 위한 CNN(Convolutional Neural Network) 아키텍처다[6]. 이는 이미지 분류와 크게 다르지 않은데 처음에 단어를 벡터로 임베딩하는 과정이 필요하다.

사이즈가 다른(사이즈 2,3,4) 필터 3개를 두 개씩, 즉 필터 6개를 이용하여 문장 매트릭스에 합성곱(Convolution)을 수행하고 피쳐 맵을 생성한다. 이후 각 맵에 대해 맥스 풀링을 진행하여 각 피쳐 맵으로부터 가장 큰 수를 추출한다. 이들 6개 맵에서 단변량(univariate) 벡터가 생성되고, 이들 6개 피쳐는 두 번째 레이어를 위한 피쳐 벡터로 연결한다. 최종적으로 소프트맥스 레이어는 피쳐 값을 받아 문장을 분류한다.

2.3 Tensorflow

텐서플로우(TensorFlow)는 머신러닝과 딥러닝을 위해 구글(Google)에서 만든 오픈소스 라이브러리다. [그림 2-9] 과 같이 수학 계산과 데이터의 흐름을 정점(Node)와 간선(Edge)을 이용하여 방향 그래프(Directed Graph)로 표현하는 데이터 플로우 그래프(Data Flow Graph)를 사용하였다[12].



[그림 2-9] 데이터 플로우 그래프

정점은 데이터 입/출력 및 읽기/저장, 수학적 계산 등의 작업을 수행한다. 간선은 정점들 간 데이터의 입출력 관계를 내고 텐서라 불리 우는 동적 사이즈의 다차원 배열을 적재하여 나르는데, 여기에서 텐서플로우라는 이름이 지어졌다.

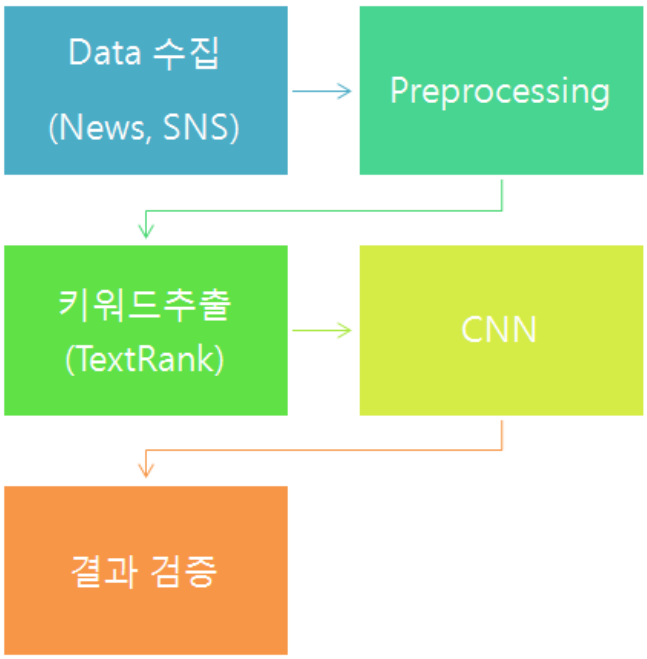
텐서플로우(Tensorflow)는 또한 데이터 플로우 그래프를 통해 풍부한 표현력을 가지고 있으며 복잡한 계산을 처리할 수 있고 코드의 수정 없이도 CPU/GPU 모드로 동작시킬 수 있다.

제 3 장 SNS분석 및 관심사 추출

본 논문에서는 SNS 게시 글을 이용하여 TextRank를 이용하여 요약하고 요약한 내용을 CNN을 통해 페이스북 그룹 카테고리를 기준으로 분류하여 최종적으로 관심사를 추출하는 방법을 제안한다.

3.1 시스템 프로세스 흐름도

본 논문에서 제안하는 SNS 분석을 통한 개인 관심사 추출방법에 대한 전체적인 프로세스 흐름도는 아래 [그림 3-1] 이다.



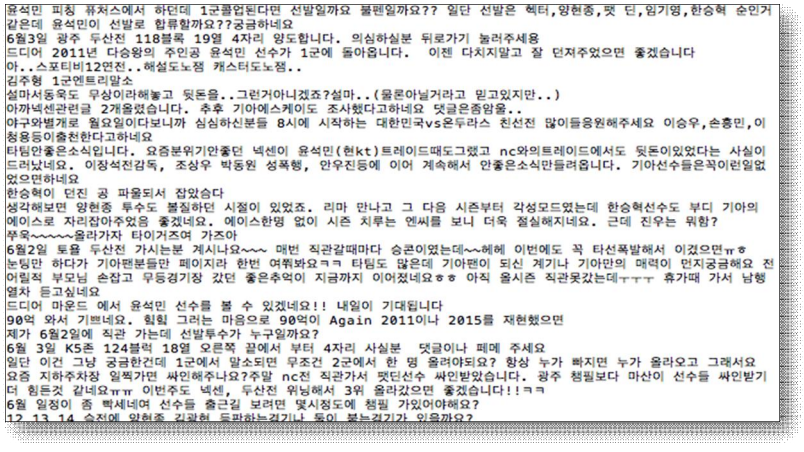
[그림3-1] 시스템 프로세스 흐름도

3.2 SNS데이터 수집

페이스북 그룹에는 아래와 같이 25개의 카테고리가 있다.

[그림 3-2] 페이스북 그룹 카테고리

이 중에서 카테고리를 선정하여 각 카테고리 내에서 특정 그룹의 게시글을 수집한다. Crawler는 C#으로 개발하였고 특정그룹을 선택 후 해당 그룹의 게시 글을 최근 글부터 수집한다. 수집된 데이터에서 특수문자 및 가비지 데이터를 삭제하는 등의 전처리 작업을 통해 학습데이터를 추출했다.


윤석민 피칭 퓨처스에서 하던데 1군콜업된다면 선발일까요 불펜일까요?? 일단 선발은 헥터,양현종,팻 딘,임기영,한승혁 순인거 같은데 윤석민이 선발로 합류할까요??궁금하네요
6월3일 광주 두산전 118블록 19열 4자리 양도합니다. 의심하실분 뒤로가기 눌러주세용
드디어 2011년 다승왕의 주인공 윤석민 선수가 1군에 돌아옵니다. 이젠 다치지말고 잘 던져주었으면 좋겠습니다
아..스포티비12연전..해설도노잼 캐스터도노잼..
김주형 1군엔트리말소
설마서동욱도 무상이라해놓고 뒷돈을..그런거아니겠죠?설마..(물론아닐거라고 믿고있지만..)
아까넥센관련글 2개올렸습니다. 추후 기아에스케이도 조사했다고하네요 댓글은좀암울..
야구와별개로 월요일이다보니까 심심하신분들 8시에 시작하는 대한민국vs온두라스 친선전 많이들응원해주세요 이승우,손흥민,이청용등이출전한다고하네요
타팀안좋은소식입니다. 요즘분위기안좋던 넥센이 윤석민(현kt)트레이드때도그랬고 nc와의트레이드에서도 뒷돈이있었다는 사실이 드러났네요. 이장석전감독, 조상우 박동원 성폭행, 안우진등에 이어 계속해서 안좋은소식만들려옵니다. 기아선수들은꼭이런일없었으면하네요
한승혁이 던진 공 파울되서 잡았습다
생각해보면 양현종 투수도 불질하던 시절이 있었죠. 리마 만나고 그 다음 시즌부터 각성모드였는데 한승혁선수도 부디 기아의 에이스로 자리잡아주었음 좋겠네요. 에이스한텐 없이 시즌 치루는 엔씨를 보니 더욱 절실해지네요. 근데 진우는 뭐함?
쑤욱~~~~~올라가자 타이거즈여 가즈아
6월2일 토욜 두산전 가시는분 계시나요~~~ 매번 직관갈때마다 승콘이었는데~~헤헤 이번에도 꼭 타선폭발해서 이겼으면ㅠㅎ
눈팅만 하다가 기아팬분들만 페이지라 한번 여쭤봐요ㅋㅋ 타팀도 많은데 기아팬이 되신 계기나 기아만의 매력이 먼지궁금해요 전 어릴적 부모님 손잡고 무등경기장 갔던 좋은추억이 지금까지 이어졌네요ㅎㅎ 아직 올시즌 직관못갔는데ㅜㅜㅜ 휴가때 가서 남행열차 듣고싶네요
드디어 마운드 에서 윤석민 선수를 볼 수 있겠네요!! 내일이 기대됩니다
90억 와서 기쁘네요. 힘힘 그러는 마음으로 90억이 Again 2011이나 2015를 재현했으면
제가 6월2일에 직관 가는데 선발투수가 누구일까요?
6월 3일 K5존 124블럭 18열 오른쪽 끝에서 부터 4자리 사실분 댓글이나 페메 주세요
일단 이건 그냥 궁금한건데 1군에서 말소되면 무조건 2군에서 한 명 올려야되요? 항상 누가 빠지면 누가 올라오고 그래서요
요즘 지하주차장 일찍가면 싸인해주나요?주말 nc전 직관가서 팻딘선수 싸인받았습니다. 광주 챔필보다 마산이 선수들 싸인받기 더 힘든것 같네요ㅠㅠ 이번주도 넥센, 두산전 위닝해서 3위 올라갔으면 좋겠습니다!!ㅋㅋ
6월 일정이 좀 빡세네여 선수들 출근길 보려면 몇시정도에 챔필 가있어야해요?
12 13 14 승전에 양현종 김광현 등판하는경기나 투이 붙는경기가 있을까요?

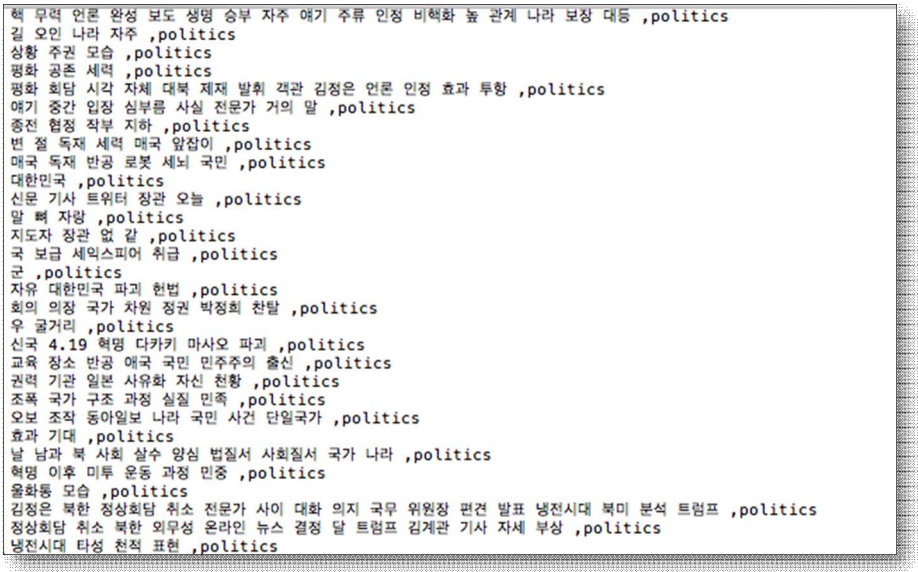
[그림 3-3] 추출된 SNS 데이터

3.3 TextRank를 이용한 키워드 추출

TextRank 알고리즘의 기본 원리는 문장을 그래프로 표현한 후, 각 간선(edge)의 값이 문장들끼리 영향을 미치는 정도라 보고 가장 중요한 정점(node)을 찾는 것이다. 문장 내에서 단어들도 서로 영향력을 행사한다고 생각하고, 여기서 가장 중요한 단어들을 찾으면 키워드 추출이 된다.

이를 위해 수집된 데이터를 문장단위로 분리 한 후 이를 단어단위로 추출해야 한다. 단어를 추출하기 위해 KoNlpy라는 형태소 분석기를 사용했다[13]. 문장내의 단어들 중 'NNG', 'NNP', 'VV', 'VA' (일반명사, 고유명사, 동사, 형용사)를 추출한다.

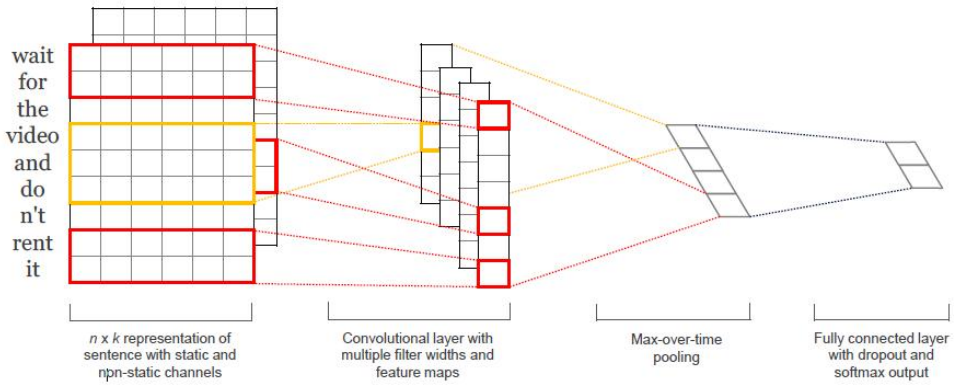
추출된 단어를 NetworkX 패키지를 이용하여 그래프 화 한 후 동시출현빈도를 가중치(weight)로 엣지(Edge)를 추가한 뒤 PageRank 함수를 이용하여 키워드를 추출한다.



[그림 3-4] 키워드 추출 결과

3.4 CNN(Convolutional Neural Network)을 이용한 분류

아래[그림3-5]은 “Convolutional Neural Networks for Sentence Classification”[2]에서 발표한 CNN을 활용한 문장분류 아키텍쳐이다.



[그림 3-5] 두 개의 채널로 구성된 모델 아키텍쳐

첫 번째 레이어는 n개의 단어를 k차원의 행벡터로 임베딩(embedding)한다. 그 다음 사이즈가 다른 여러 필터를 이용해 합성곱 변환을 하고 피쳐 맵(feature map)을 만들고 맥스 풀링(Max Pooling)하는 과정을 거친다. 이후에 드롭아웃(Drop-out) 정규화를 추가하고, 소프트맥스(Softmax) 레이어의 결과로 분류를 수행한다.

아래 [그림 3-6]는 맥스 풀링, 합성곱 레이어를 만드는 부분이다. 각각 사이즈가 다른 여러 필터를 사용하여 합성곱(Convolution) 텐서를 반복적으로 생성하고 이를 하나의 큰 피쳐 벡터로 병합한다[16].

```
pooled_outputs = []
for i, filter_size in enumerate(filter_sizes):
    with tf.name_scope("conv-maxpool-%s" % filter_size):
        # Convolution Layer
        filter_shape = [filter_size, embedding_size, 1, num_filters]
        W = tf.Variable(tf.truncated_normal(filter_shape, stddev=0.1), name="W")
        b = tf.Variable(tf.constant(0.1, shape=[num_filters]), name="b")
        conv = tf.nn.conv2d(
            self.embedded_chars_expanded,
            W,
            strides=[1, 1, 1, 1],
            padding="VALID",
            name="conv")
        # Apply nonlinearity
        h = tf.nn.relu(tf.nn.bias_add(conv, b), name="relu")
        # Max-pooling over the outputs
        pooled = tf.nn.max_pool(
            h,
            ksize=[1, sequence_length - filter_size + 1, 1, 1],
            strides=[1, 1, 1, 1],
            padding='VALID',
            name="pool")
        pooled_outputs.append(pooled)

# Combine all the pooled features
num_filters_total = num_filters * len(filter_sizes)
self.h_pool = tf.concat(3, pooled_outputs)
self.h_pool_flat = tf.reshape(self.h_pool, [-1, num_filters_total])
```

[그림 3-6] Convolution Layer, Max-pooling 코드

드롭아웃(Drop-Out)은 합성곱 신경망의 오버피팅을 방지하는 방법으로 좋은 성능을 낸다. 드롭아웃(Drop-Out) 레이어는 특정뉴런의 값을 0으로 만들어 일부를 확률적으로 제외하는 효과를 낸다.

손실 함수를 최적화하기 위해 텐서플로우에서 제공하는 여러가지 옵티마이저(Optimizer) 중 아담 옵티마이저를 사용하였다.

```
global_step = tf.Variable(0, name="global_step", trainable=False)
optimizer = tf.train.AdamOptimizer(1e-4)
grads_and_vars = optimizer.compute_gradients(cnn.loss)
train_op = optimizer.apply_gradients(grads_and_vars, global_step=global_step)
```

[그림3-7] 아담 옵티마이저(Adam Optimizer)

TextRank로 키워드를 추출한 결과를 이용하여 학습데이터를 만들었고 해당 데이터를 CNN을 이용하여 학습하였다. 아래 그림[3-8]은 해당 데이터가 학습되는 과정이다.

```
2018-06-08T13:38:56.521694: step 1411, loss 0.732486, acc 0.765625
2018-06-08T13:38:57.129602: step 1412, loss 0.793643, acc 0.6875
2018-06-08T13:38:57.703477: step 1413, loss 0.880029, acc 0.78125
2018-06-08T13:38:58.272896: step 1414, loss 1.21454, acc 0.75
2018-06-08T13:38:58.842514: step 1415, loss 0.986631, acc 0.796875
2018-06-08T13:38:59.301233: step 1416, loss 0.940323, acc 0.72549
2018-06-08T13:38:59.877960: step 1417, loss 0.882557, acc 0.734375
2018-06-08T13:39:00.449761: step 1418, loss 0.804656, acc 0.796875
2018-06-08T13:39:01.010511: step 1419, loss 0.355433, acc 0.859375
2018-06-08T13:39:01.581723: step 1420, loss 0.571376, acc 0.859375
2018-06-08T13:39:02.156241: step 1421, loss 0.333927, acc 0.875
2018-06-08T13:39:02.726397: step 1422, loss 0.698687, acc 0.765625
2018-06-08T13:39:03.287074: step 1423, loss 0.872103, acc 0.71875
2018-06-08T13:39:03.748233: step 1424, loss 0.734594, acc 0.745098
2018-06-08T13:39:04.321076: step 1425, loss 0.429172, acc 0.796875
2018-06-08T13:39:04.881135: step 1426, loss 0.478888, acc 0.8125
2018-06-08T13:39:05.456376: step 1427, loss 0.454832, acc 0.859375
2018-06-08T13:39:06.038174: step 1428, loss 0.435049, acc 0.78125
2018-06-08T13:39:06.605265: step 1429, loss 0.561875, acc 0.796875
2018-06-08T13:39:07.173833: step 1430, loss 0.47393, acc 0.8125
2018-06-08T13:39:07.754069: step 1431, loss 0.371687, acc 0.84375
2018-06-08T13:39:08.213458: step 1432, loss 0.390754, acc 0.862745
2018-06-08T13:39:08.788532: step 1433, loss 0.357882, acc 0.875
2018-06-08T13:39:09.366269: step 1434, loss 0.735299, acc 0.71875
```

[그림3-8] 데이터 학습과정

관심사 추출을 위하여 CNN을 이용하여 학습하는 과정을 나타내고 있다. 총 2000단계를 거쳐 학습 모델을 구축하였고 정확률을 나타내는 acc와 손실률을 나타내는 loss의 값이 함께 출력된다. 학습 중 검증데이터에 대한 정확도는 평균 80%내외로 나타났다.

제 4 장 실험 및 결과분석

4.1 실험환경

본 연구에 사용된 실험환경은 아래 [표 4-1] 과 같다.

[표 4-1] 실험 환경

H/W	CPU	AMD 라이젠2700
	GPU	NVidia Geforce GTX 1060 6G
	RAM	32GB
	OS	Windows 10
S/W	Framework	Tensorflow 1.1
	Language	Python 3.5
	Package	NetworkX

페이스북의 그룹에서 스포츠, 정치, 연예 분야의 데이터 약 3000천 건을 수집하여 데이터 정제 후 2853건의 데이터를 얻었다. 학습 후 정확도를 측정할 검증 데이터 셋은 이 학습데이터 셋의 약 10%를 생성하였다. 실험 데이터 셋은 학습데이터 셋의 약 15% 내외로 생성하였다.

[표 4-2] 실험 데이터 셋

카테고리	학습데이터셋	검증 데이터셋	실험 데이터 셋	
			TextRank적용	미적용
스포츠	1035	105	165	165
정치	959	96	150	150
연예	859	85	137	137
총합	2853	286	453	452

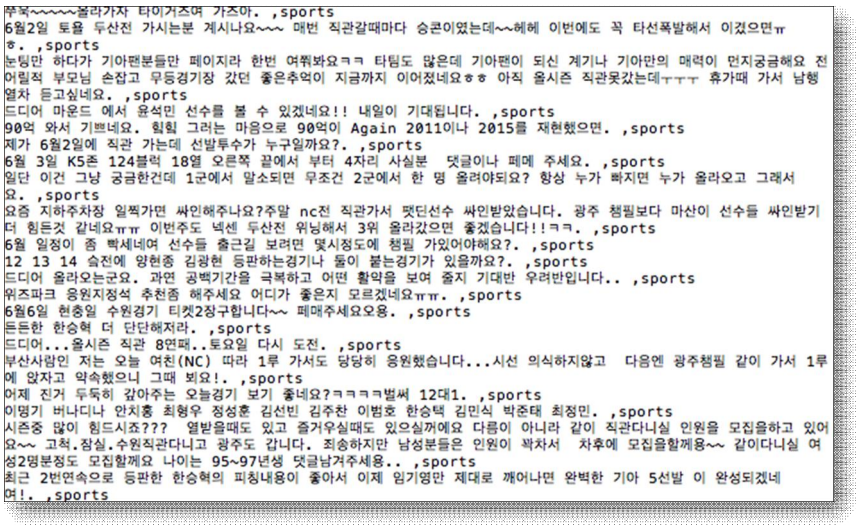
실험 데이터 셋도 마찬가지로 학습 데이터와 동일한 방법으로 실험데이터를 준비하였다. 역시 페이스북에서 해당 데이터를 크롤링 한 후 TextRank를 이용하여 해당 텍스트에서 키워드를 추출하여 실험데이터

를 만들었다.

[그림 4-1] 실험데이터

이 데이터를 TextRank로 키워드 추출 한 학습데이터 셋[그림 4-2]과 전처리작업만을 한 학습데이터 셋[그림 4-3] 두 개를 만들었다.

[그림 4-2] TextRank 적용 학습 데이터 셋



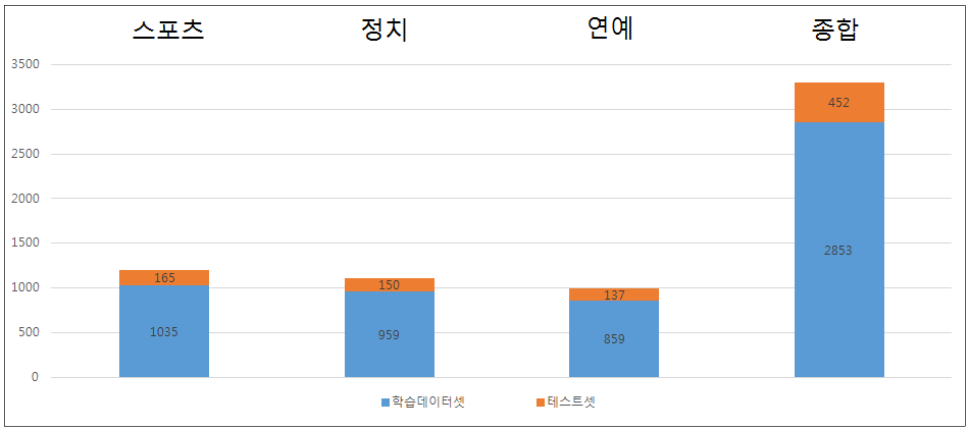
[그림 4-3] 전처리 작업만 된 학습 데이터 셋

4.2 결과분석

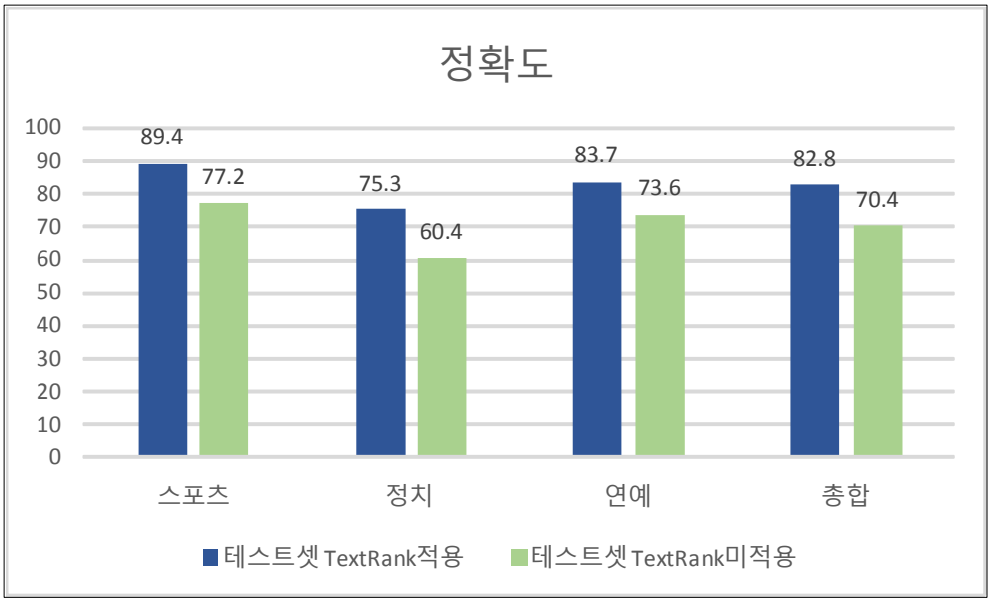
스포츠, 정치, 연예 분야의 SNS글 약 3000천 건을 수집하여 학습 후 실험 데이터 셋을 이용하여 실험한 결과는 아래의 [표 4-3] 와 [그림 4-4], [그림 4-5] 와 같다.

[표 4-3] 실험 결과

카테고리	학습데이터셋	테스트셋	정확도	
			TextRank적용	미적용
스포츠	1035	165	89.4	77.2
정치	959	150	75.3	60.4
연예	859	137	83.7	73.6
총합	2853	452	82.8	70.4



[그림 4-4] 학습데이터 셋 및 테스트 셋



[그림 4-5] 테스트 셋 TextRank 적용, 미적용 별 정확도

위 결과와 같이 TextRank를 적용했을 때 평균 82.8%의 정확도를 얻었고 TextRank를 적용하지 않고 분류 했을 때는 약 70.4%의 정확도를 얻었다.

실험 결과 [그림 4-5] 에서 볼 수 있듯 기존 방법 보다 TextRank를 이용하여 요약한 데이터를 분류하는 것이 카테고리 별로 다소 차이는 있지만 전반적으로 정확도가 높음을 알 수 있다.

요약하지 않은 데이터의 경우 많은 단어들이 포함되어 있으므로 CNN을 이용한 피쳐 맵의 종류가 좀더 다양한 결과가 나와 데이터 간 유사도를 판별할 때 정확도가 낮은 것으로 보여진다.

제 5 장 결론 및 제언

본 논문에서는 사용자의 SNS를 통해 관심사를 추출하기 위한 방법을 제안하였다. 페이스북 데이터를 수집하여 TextRank 알고리즘으로 키워드를 추출한 뒤 CNN을 이용하여 분류하였다.

기존 방법과 제안한 방법과의 성능을 검증하기 위해 동일한 데이터를 가지고 키워드 추출을 한 것과 안한 것 두 가지 데이터셋을 가지고 실험을 하였고 실험 결과 제안한 방법이 약 10% 높은 정확도를 보인다. 하지만 카테고리 별로 편차가 있었고 이는 해당 카테고리에 새로운 주제가 많이 생길 경우 기존의 학습데이터와의 차이 때문인 것으로 보여 진다.

SNS특성상 뉴스와는 달리 주어+동사+목적어 와 같은 형식의 완전한 문장보다는 개인의 감정이나 줄임말이 주를 이루기 때문에 짧은 데이터가 많았는데, 이런 데이터는 요약에 효율적이지 못한 측면이 있었다. 향후 짧은글에서 어떻게 핵심 키워드를 추출할 지에 대한 연구가 필요할 듯하다. 또한 정확도를 높이기 위해선 특정 카테고리의 경우 단순한 SNS 뿐 만 아니라 다른 데이터를 활용하여 학습하는 방법 등의 추가적인 연구가 필요할 듯하다.

참고문헌

[1] 홍택은, CNN을 이용한 SNS 사용자 관심사 카테고리 분류 및 팔로잉 추천방법, 조선대학교 산업기술융합대학원, 2016

[2] Y. Kim, Convolutional Neural Networks for Sentence Classification, EMNLP, 2014

[3] 유소엽, 정옥란, 소셜 카테고리를 이용한 추천방법, 인터넷정보학회논문지, 제 15권, 제 5호, 73-82, 2014

[4] 나성희, 김정인, 이은지, 김판구, SNS가 가지는 특징정보를 활용한 단문텍스트 카테고리 분류방법에 관한 연구, 한국정보기술학회논문지 제14권 제6호, 159-165, 2016

[5] Rada Mihalcea, Paul Tarau, TextRank: Bringing Order into Texts, Proceedings of EMNLP, 404 - 411, 2004

[6] Ye Zhang, Byron Wallace, A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification, IJCNLP, 2015

[7] 조휘열, 김진화, 윤상웅, 김경민, 장병탁, 컨볼루션 신경망 기반 대용량 텍스트 데이터 분류 기술, 동계학술발표회 논문집, 792-794, 2015

[8] Sergey Brin, Lawrence Page, The Anatomy of a Large-Scale Hypertextual Web Search Engine, Computer Science Department, Stanford University, Stanford, CA 94305, 1998

[9] 서형조, 이석원, 태깅 이미지와 시맨틱 관심사 표현을 이용한 SNS친구 추천 시스템, 한국컴퓨터종합학술대회 논문집, 945-947,2015

[10] 나성희, 김정인, 이은지, 김판구, SNS가 가지는 특징정보를 활용한 단문텍스트 카테고리 분류방법에 관한 연구, 한국정보기술학회논문

지, 159-165, 2016

[11] 유소엽, 정옥란, 소셜 카테고리를 이용한 추천 방법, 한국인터넷정보학회, 73-82, 2014

[12] Google, Tensorflow, https://www.tensorflow.org/programmers_guide/graphs

[13] KoNLPy, KoNLPy, http://konlpy-ko.readthedocs.io

[14] Stanford, Feature extraction using convolution, http://deeplearning.stanford.edu/wiki/index.php/Feature_extraction_using_convolution

[15] Stanford, CS232n Convolution Neural Network, http://cs231n.github.io/convolutional-networks/

[16] Denny Britz, Implementing a CNN for Text Classification in TensorFlow, http://www.wildml.com/2015/12/implementing-a-cnn-for-text-classification-in-tensorflow/