공학석사 학위논문

한국어 트위터 데이터의 감성 분석 알고리즘 구현

Implementation of the Sentiment Analysis Algorithm with Korean Twitter Data

2016년 8월

서울과학기술대학교 일반대학원

컴퓨터공학과

윤 한 중

한국어 트위터 데이터의 감성 분석 알고리즘 구현

Implementation of the Sentiment Analysis Algorithm with Korean Twitter Data

지도교수 석상기

이 논문을 공학석사 학위논문으로 제출함

2016년 7월

서울과학기술대학교 일반대학원

컴퓨터공학과

윤 한 중

윤한중의 공학석사 학위논문을 인준함

2016년 7월

심사위원장 趙炳鎬 (인)

심사위원 李昌勳 (인)

심사위원 石尙基 (인)

목 차

요 약

제 목 : 한글 트위터 데이터의 감성 분석 알고리즘 구현

모바일 기기가 발전함에 따라 소셜 네트워크 서비스는 생활에 깊게 파고들어 사회 이슈 및 여론의 조성에 까지 영향을 끼치며, 개인의 의견 표출의 장이 되기도 한다. 특히 오프라인의 인간관계와 영향이 적은 트위터는 각종 사회 정치 이슈 및 특정 제품 또는 인물에 대한 언급이 잦다. 트위터를 통한 트렌드 분석, 제품 수요의 분석 등은 지속적으로 연구되어 사용자에게 서비스로 제공되고 있다.

이처럼 트위터의 데이터를 활용하고자 하는 움직임은 계속 있어왔으나, 트위터를 통한 감성분석에 대한 연구는 활발하지 않은 실정이다. 감성 분석은 데이터의 통계화에 그치지 않고 패턴을 파악하거나 극성을 계산하여 사용자의 인식을 파악할 수 있는 기술이다. 국내에는 이런 감성 분석 알고리즘에 대한 연구가 부족할 뿐 아니라 감성 사전의 구축이 미비하고 대부분 기계학습을 통한 감성 분석에 치중하고 있기 때문에 한국어 문법에 기초한 감성 분석 알고리즘의 연구는 미흡하고 부족한 실정이다.

본 논문에서는 이러한 부분을 고려하여 트위터 데이터의 활용과 한국어 감성 분석을 위한 알고리즘 및 감성분석 시스템을 제안한다. 주제어와 관련된 트윗의 문장의 극성을 문장과 형태소, 단어 단위로 분석하고 각 극성을 계산함으로서 문장 구성이 단순하고 짧은 트위터 데이터 분석에 최적화한 감성 분석 알고리즘을 제시하였다. 극성 분석을 위하여 한국어 감성 사전의 구축이 요구되었기에, SentiWordNet이라는 영어 감성사전의 데이터를 추출하여 제시한 알고리즘에 맞게 재구성하고 경량화 한 뒤 한국어로 번역하여 특정 분야에 특화되지 않은 범용적인 한국어 감성 사전을 구축하였다.

표 목 차

그림목차

Ⅰ. 서 론

1. 연구 배경 및 목적

가. 연구 배경

최근 급속한 소셜 네트워크 서비스(Social Network Service, SNS)의 발달과 이로 인한 정보의 범람으로 온라인에서 정보를 수집하고 이를 분석하는 데이터마이닝, 데이터 사이언스에 대한 관심이 지속적으로 높아지고 있다. 또한 모바일 기술의 발달과 함께 현대인들의 생활패턴에 SNS는 이미 깊게 자리를 잡았다. 특히 트위터의 경우 여타 SNS와 비교하여 정치, 문화에 관한 게시글이 풍부하며 사회적 이슈 및 트렌드가 즉각 반영된다. 또한 감성 분석은 외국에서는 활발하게 연구 중이지만 국내에서 트위터를 활용한 감성 분석 시스템의 구축 및 데이터 마이닝에 대한 연구는 부족한 실정이다 [1, 2, 3]. 이러한 데이터의 범람 속에서 필요한 정보를 찾아내고 분석하는 데에는 기계학습에 대한 지속적인 연구 뿐 아니라 감성 분석 및 감성 사전의 구축에 대한 연구가 필요하며, 향후 효율적인 감성 분석 환경을 위한 시스템 구축 연구를 통해 트위터 데이터를 활용할 필요가 있으며, 기업 및 정부 차원에서 트렌드를 분석하고 사회 이슈를 파악하는데 도움을 줄 수 있어야 한다. 그림 1.1과 같이 기업 및 정부는 SNS에서 수집한 정보를 바탕으로 제품 또는 서비스의 이용자에게 더 나은 서비스를 제공할 수 있고 다양한 연구 발전에 정보를 활용할 수 있도록 해야 한다 [4, 5, 6].

그림 1.1 감성 분석 서비스의 개념도

나. 연구 목적과 의의

본 연구는 그림 1.2와 같이 자연어 처리를 통한 한글 형태소 분석 및 감성 사전의 구축을 통한 감성 분석의 알고리즘을 제안함으로써 한국어 트위터 데이터를 효율적으로 활용하는데 있다. 이를 통해 감성 분석이 필요한 기업체나 연구과정에서 실용적인 서비스를 제공할 수 있을 것으로 판단된다. 또한 한국어 트위터 분석 시스템을 제공하여 정보 수집 과정에서 불필요한 정보는 필터링하고, 원하는 정보에 대한 감성 분석만을 정확하게 제공하여 오류를 최소화 하는 시스템을 제안하였다.

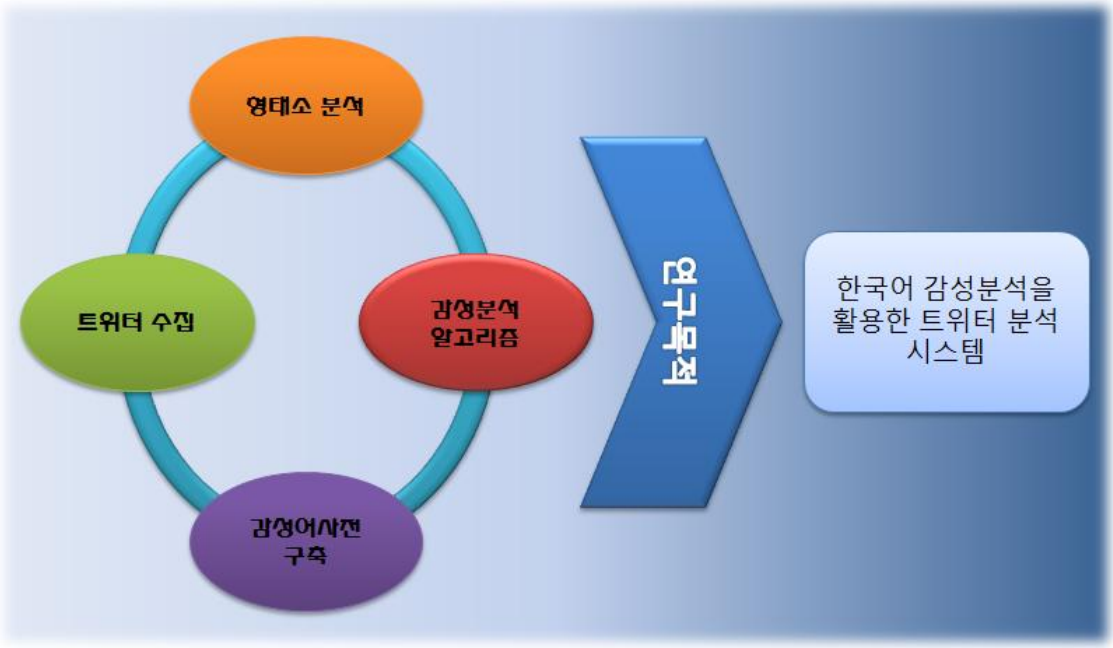


그림 1.2 연구 목적

본 연구의 목적은 감성 분석에 있어 기계학습이 아닌 알고리즘으로 짧은 문장을 효율적이고 정확하게 분석하는 것을 중점으로 한다. 이를 위하여 범용적인 감성 사전의 구축을 통해 트위터 게시글을 분석하여 검색어에 대한 감성을 분석하고 의견을 수집할 수 있는 모델을 제안한다.

2. 연구 범위 및 구성

가. 연구 범위

본 논문은 오피니언 마이닝의 한 종류인 감성 분석에 대하여 한국어 단문으로 구성된 트윗이 가지는 주제어에 대한 감성을 파악하는 효율적인 알고리즘의 구현에 초점을 맞추고 있다. 알고리즘은 한국어 품사 및 어순의 분석을 통한 극성의 판단과 강도를 계산한다.

본 논문에서는 효율적인 감성 분석을 위한 한글 형태소 분석과 감성 사전에 관한 연구를 포함한다. 한글 문장은 교착어의 특징으로 단어와 단어 사이에 조사와 접미사가 교착된다 [7]. 이 같은 특징의 한글 문장을 정확히 분석하기 위해 단어의 어간을 추출하고(stemming) 조사와 접미사가 가지는 의미를 분석할 수 있는 형태소의 분석이 필수적이며, 단어의 의미를 파악하기 위한 감성 사전의 구축이 연구 환경으로 요구된다.

나. 연구 구성

본 논문은 4장으로 구성되어 있으며 각 장의 내용은 다음과 같다. 2장에서는 감성 분석 연구의 동향, 제안하는 감성 분석 알고리즘의 관련 연구 및 기존 연구에 대한 논의를 한다. 3장에서는 한글 트위터 데이터를 활용한 감성 분석 알고리즘 구현에 대한 연구를 논의한다. 제안하는 알고리즘의 구성과 구현 방식을 중심으로 논의한다. 4장에서는 알고리즘을 적용한 실험을 제안한다. 제안하는 실험의 구성 방식과 실험에 필요한 사전 연구, 실험 결과에 대하여 중점적으로 논의한다. 마지막으로 5장의 결론과 한계점, 향후 연구 방향에 대한 내용으로 논문을 마무리한다.

Ⅱ. 관련 연구

본 장에서는 감성 분석과 관련된 국내 동향 및 사례 연구, 효율적인 감성 분석 알고리즘 구현을 위한 형태소 분석 및 어간 추출, 그리고 트위터의 특성에 대한 이해와 트위터와 감성 분석에 대한 기존 연구에 대해 논의한다.

1. 감성 분석 관련 연구 및 동향

감성 분석은 어떤 사건, 대상에 대한 의견을 극성으로 정의하여 긍정, 부정으로 분류한다. 이런 감성 분석은 문장의 어순 및 단어의 의미에 따라 정확도가 좌우되며 이런 감성을 정확히 분석하기 위해서는 많은 정보의 수집 및 분석이 필요하다.

감성 분석과 관련된 연구는 외국에서 많이 진행되고 있다. 상품의 리뷰와 영화의 평점 분석을 이용해 주로 연구가 진행되며, 트위터는 소셜 네트워크 중에서 가장 활발하게 감성 분석에 이용되고 있다. 그러나 한국어의 감성 분석 연구는 상대적으로 적은 수를 보인다. 국내 학술 데이터베이스인 DBPIA에서 '트위터 분석'으로 검색한 결과 총 374편의 논문이 나왔고, 이 중 감성 분석과 관련된 연구는 25편이었다 (2016.5.30.).

감성 분석은 크게 기계학습과 감성 사전을 이용하는 두 가지 방식으로 분류할 수 있다. 일반적으로는 기계학습을 활용하여 패턴을 학습하고 그에 따라 주어진 데이터를 분석하는 방식이 주로 활용되나, WordNet, SentiWordnet과 같은 감성 사전이 외국에 이미 구축되어 감성 분석에 활용되고 있다. 또한 감성 사전의 극성 계산을 원활히 하기 위해 형태소의 분석 및 어근 추출에 관한 연구가 필요한 실정이다 [7, 8, 9].

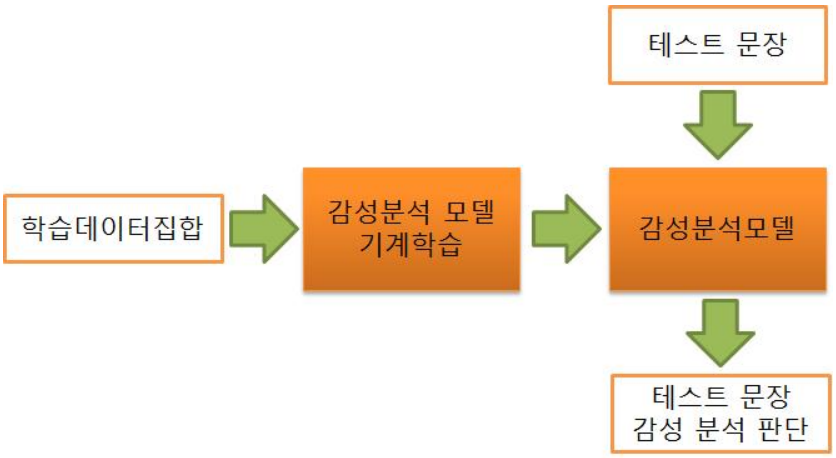


그림 2.1 기계학습에 의한 감성 분석 프로세스

가. 기계학습

기계학습 기반 감성 분석은 그림 2.1에서 보이는 것과 같이 감성분석 모델의 학습단계와 테스트 데이터를 활용한 감성분석 단계로 나눌 수 있다. 학습단계에서는 감성 분석 모델을 학습시켜야 하며, 이 때 '감성 레이블이 있는 문장'이라는 학습 데이터를 이용하여 학습한다. 학습데이터는 긍정 및 부정으로 분류된 문장들로 이루어져 있어 감성 분석 모델이 문장에서 특성을 추출하고 벡터를 생성한다. 테스트 단계에서는 테스트 집합의 문장들이 가진 특성을 다시 추출하여 벡터화 한 뒤 앞서 생성한 감성 분석 모델에 이를 입력해 문장의 감성을 분류한다.

감성 분석에 사용되는 기계학습 알고리즘은 크게 서포트 벡터 머신(Support Vector Machine, SVM)과 나이브 베이즈(Naive Bayes)를 기반으로 연구되고 있다.

1) 서포트 벡터 머신(SVM)

SVM은 1998년 통계학자 Vapnik에 의해 제안된 학습기법이다. 비선형모델의 데이터라도 선형모델의 문제로 변환하여 서로 다른 분류가 가능한 학습 데이터 간에 간격이 최대가 되는 직선인 초평면을 학습 알고리즘을 이용하여 찾은 뒤 이를 기준으로 이후 데이터를 분류하는 모델이다. 그림 2.2는 간단한 선형 SVM을 나타낸 것으로 O와 X의 데이터를 분류하는 하나의 직선을 계산한 것으로, 해당 직선은 분류한 데이터로부터 가장 멀리 떨어진 간격(margin)을 계산하여 도출된다. SVM은 조정할 변수의 수가 적어 비교적 간단하게 기계학습에 필요한 요소를 마련할 수 있다. 또한 수학적으로 데이터를 분류하기 수월하여 지속적으로 텍스트 마이닝, 영상 인식 등에서 연구되어 왔다 [10].

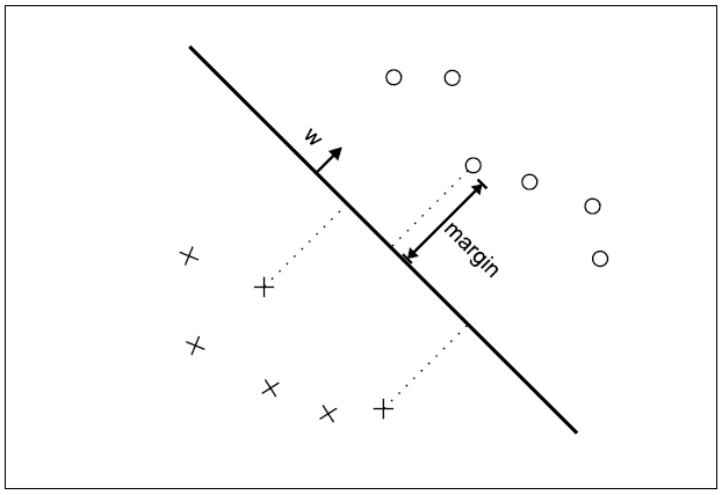


그림 2.2 간단한 선형 Support Vector Machine

2) 나이브 베이즈 분류

나이브 베이즈 분류는 Bayes‘ Theorem에 기반한 분류법으로, 학습 데이터가 가지고 있는 요소들이 서로 독립적이라는 가정에 의해 통계적 확률을 측정하여 학습한 뒤 테스트 데이터를 확률이 높은 쪽으로 분류한다. 나이브 베이즈 알고리즘은 비교적 높은 학습 시간을 가지지만 일반적으로 정확도가 높은 편이며, 알고리즘의 구조 와 가정이 비교적 단순하지만 실제 상황에서 매우 잘 작동한다. 또한 나이브 베이즈 분류기는 학습데이터의 양이 많을 때 더욱 정확도가 높다 [11].

나. 감성 사전 기반 연구

감성 사전을 이용한 감성 분석은 감성 분석 대상은 문장에서 단어를 추출한다. 다음으로 그 단어가 감성 사전에 포함되어 있는지 확인하고, 사전에 정의된 극성과 값을 추출하여 문장의 감성극성을 계산한다. 감성 사전을 이용한 극성 계산과 관련된 연구로는 낱말망, 말뭉치, 감성사전의 구축 등이 있으며 감성 사전에 대하여 국내 연구는 대부분 주제 지향의 감성 사전을 구축하고자 하는 연구를 하였으며 실제로 구축되어 범용적인 사용을 목표로 한 한국어 감성사전은 ‘집단 지성을 이용한 한글 감성 사전 구축 연구’가 유일하며, 실제 사용을 위한 서비스는 제공되지 않고 있다 [12].

외국의 낱말망 및 감성 사전은 워드넷(WordNet)과 SentiWordNet(SentiWordNet)이 있으며 각각 낱말망과 감성 사전을 구축하고 있다.

1) WordNet

WordNet 영어의 단어들로 구성된 의미 어휘 목록이다. WordNet은 2005년 프린스턴 대학의 인지과학 연구소에서 개발되었고, 크리스티안 펠바움(Christiane Fellbaum)에 의해 현재 관리되고 있다. WordNet의 제작 목적은 단어의 모음과 유의어, 반의어의 배합을 조합하여 텍스트 분석과 인공 지능 분야에서 사용하기 위해 개발되었다. 그림 2.2는 WordNet의 온라인 검색 기능을 이용해 very 라는 단어를 검색한 것으로, 단어의 품사에 따라 유의어 및 뜻에 대하여 설명하고 있다. WordNet에는 15만개 이상의 영어 단어가 저장되어있으며 11만5천개의 동의어 집합을 보유하고 있고 20만개의 단어와 의미의 조합이 구성되어있다 [8].

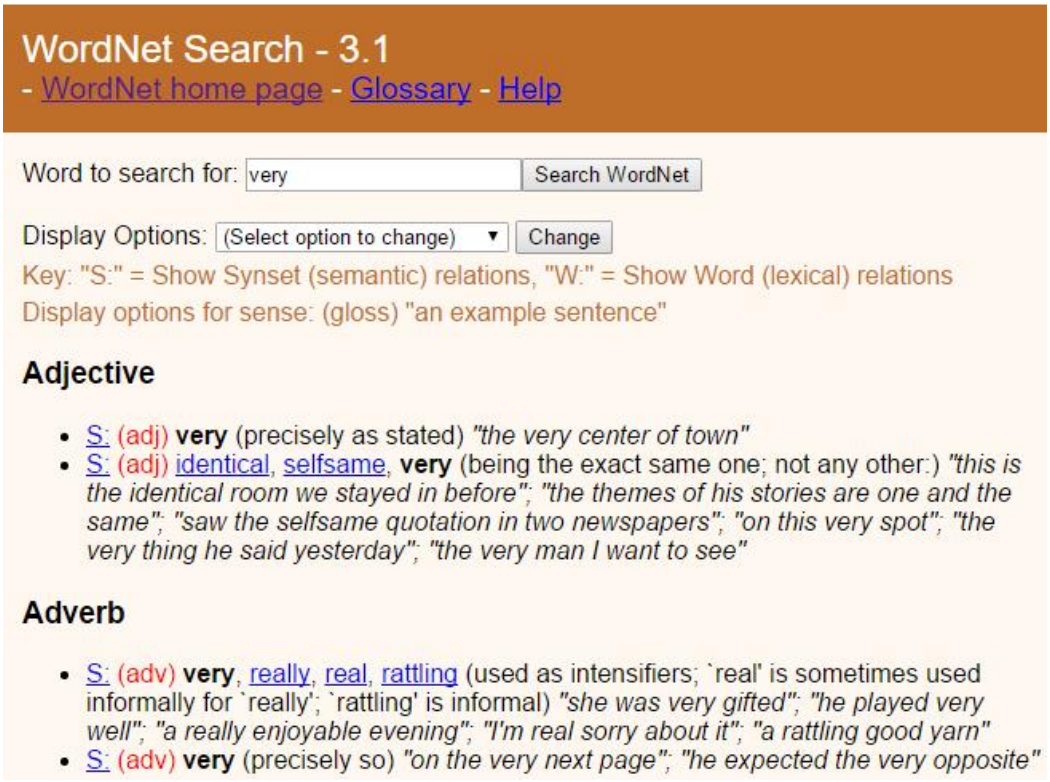


그림 2.3 WordNet을 이용한 단어 검색 예시

2) SentiWordNet

SentiWordNet은 영어 단어가 가진 극성을 정의하고 구분하기 위해 개발된 감성 사전으로, 각 단어에는 객관-주관 극성과 긍정-부정 극성이 정의되어 있으며, 극성 값의 크기가 지정되어 있다. 각 단어의 그룹에는 동의어 및 유의어가 하나의 집합으로 지정되어 있다. SentiWordNet은 기계학습을 통해 구축되었으며 WordNet의 데이터를 이용해 작성되었기 때문에 현재 가장 거대한 감성 사전이라고 할 수 있다 [9].

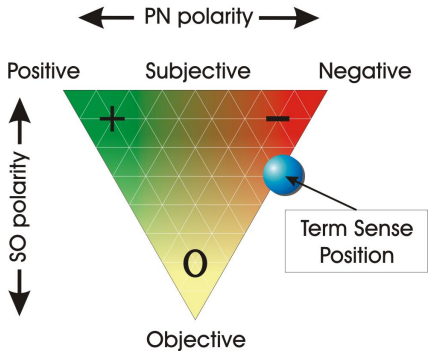


그림 2.4 SentiWordNet의 극성 분류 방식

본 연구에서는 SentiWordNet의 데이터를 한글화하여 이용할 것이다.

3) 트립어드바이저 정서분석 코퍼스

트립어드바이저 한국어 정서분석 코퍼스는 한국어 호텔 리뷰를 정서분석에 사용하기 위한 목적으로 구축된 말뭉치로, 국립창원대학교의 AirLab에서 개발되었으며, 호텔의 이름과 평점, 사용자의 리뷰에 사용된 문장을 분석하여 리뷰의 대상과 연관된 단어의 쌍을 이용하여 각 단어가 가지는 관계에 대한 집합을 생성한다. 이 과정에서 말뭉치를 활용하기 위하여 코퍼스를 생성하였으며, 문장의 극성을 분류한 리뷰 2000 개와 극성 분류가 되어있지 않는 리뷰 4853개를 활용하여 총 14495개의 문장을 분석하였으며, 코퍼스의 정보는 각 리뷰별로 텍스트 문서로 저장되어 사용자들의 리뷰 원문에 극성이 표시되어 있는 형식이다. 표 2.1은 트립어드바이저 정서분석 코퍼스 내용의 일부를 예시로 나타낸 것으로 문장과 문장에 부여된 극성을 나타내었으며, 극성에 대한 설명을 본인이 작성하였다 [13].

표 2.1 트립어드바이저 정서분석 예시

문장	극성	설명
크리스마스에 예약을 하고 갔었다. 예약을 했는데 예약한 방이 없단다. 그러더니 한단계 위인 주니어 스위트룸을 내주었다.	객관	3개의 문장마다 객관적인 극성이 부여되어 문장에 정서가 묻어나지 않는 객관적인 문장
솔직히 말해서 이곳은 최고 중에 하나이며, 제가 지금껏 묵었던 5성급 호텔중에서는 최고라고 말할 수 있겠습니다.	긍정	긍정적인 정서를 담고 있는 문장
마일리지를 써서 왔는데 숙박과 레스토랑을 따로따로 이용해야 해서 좀 불편했습니다.	부정	부정적인 정서를 담고 있는 문장

4) Korean Sentiment Analysis Corpus(KOSAC)

KOSAC은 한국어 감정 및 의견 분석에 필요한 한국어 감정 낱말망을 구축 및 분석한 결과로, 서울대학교 신문 기사의 문장을 대상으로 총 7744개의 문장에서 17582개의 감정 표현을 추출하였다. 분석한 감정 표현의 속성들을 이용해 전체 문장의 극성 분류 실험을 SVM을 사용하여 실행하였으며, 82.52%의 정확도를 나타내 한국어 감정 분석 낱말망의 활용 가능성을 보였다. KOSAC은 직접 개발한 GUI를 이용하여 신문 기사를 3명의 주석자가 수작업으로 분류 및 분석하였다. 표 2.2는 KOSAC의 데이터 구성을 나타낸 것으로, KOSAC의 구성 방식은 각 문장이 가지는 극성을 csv 문서에 관계형 데이터베이스의 형식으로 기록하여 비교적 사용은 용이하지만 극성 값을 High, Medium, Low, None

으로 분류하여 수치화하기 난해한 점이 있다 [14].

표 2.2 KOSAC 데이터의 구성 방식

원문	허씨는 “그저 남을 돕고 싶었을 뿐“이라고 말해 안씨 부부를 감동시켰다.
극성 분류	강한 긍정
극성 단어	돕고 싶다(강한 긍정), 감동 시키다(긍정).

위와 같이 다양한 국내 연구를 통한 감성사전 및 낱말망 연구가 존재하지만, 본 연구의 주제와 다른 분야의 데이터를 수집하여 분석한 측면이 있으며, 각 낱말에 대한 극성 및 강도가 수치화 되어있지 않아 데이터를 직접 가공하여 극성 계산 알고리즘으로 사용하기에 난해한 부분이 있기 때문에, 본 논문에서는 단어별로 수치화가 이루어져있는 SentiWordNet을 한글화 한 뒤에 트위터에서 제공하는 형태소 분석 API를 사용하여 트위터 데이터 분석에 최적화 하고자 하였다.

라. 트위터 특성 분석

트위터(Twitter)는 2006년 비즈 스톤, 잭 도시, 에반 윌리엄스가 공동 설립한 회사이자 짧은 문장으로 소통하는 소셜 네트워크 서비스의 지칭이다. 트위터의 특성은 편이성, 신속성 및 확장성, 비대칭 네트워크를 들 수 있다.

편이성의 관점에서 트위터는 140자 제한의 짧은 문장 중심의 콘텐츠로 작성에 부담이 없다. 또한 모바일 기기 및 API를 통한 외부 웹페이지와 연동이 쉽다는 장점을 가지고 있다.

신속성 및 확장성의 측면에서 트윗의 작성에 시간과 공간의 제약을 적게 받으며 메시지를 공유할 수 있다는 것이다. 그리고 리트윗 방식으로 쉽게 메시지가 확산되고 전달된다.

비대칭 네트워크는 트위터가 가진 큰 특징으로, 상대방의 동의 없이 팔로우가 가능하기 때문에 사용자간에 비대칭한 연결 구조를 가지게 된다. 그렇지만 트위터의 이용자는 자신이 누구에게 팔로우 되고 있는지, 누구를 팔로윙 하는지 확인이 가능하며, 이러한 비대칭성으로 인해 영향력이 있는 사회적인 유명인들이 콘텐츠 제작자로서 트윗을 남기면 해당 인물을 팔로우하는 다수에 의해서 메시지가 공유되는 형태로 활용된다.

위와 같은 특징으로 트위터는 월간 이용자수가 3억명 이상이며 이는 엄청난 양의 트위터 데이터가 저장되어있고, 앞으로도 지속적으로 증가할 것이며, 더 정확한 방식의 트위터 데이터의 분석이 필요한 것이다. 단순 통계가 아닌 이용자의 의도를 분석하고 트렌드를 파악할 수 있는 감성 분석과 같은 기법이 트위터를 분석하는데 요구된다는 것이다 [15, 16].

2. 트위터와 감성 분석에 관한 기존 연구

홍진주 외 3인은 감성 분석과 SVM을 이용한 인터넷 악성댓글 탐지 기법에 대하여 연구하였다. 악성 댓글에 주로 사용되는 변형된 단어 및 비속어가 기존 감성 분석 연구로 정확히 추적하는데 한계를 보여 제안된 연구로, 그림 2.3에서와 같이 악성댓글에 특화된 감성 사전을 구축하여 악성 지수를 계산하는 방법으로 일반적인 감성 분석 연구들보다 높은 정확도를 보였다 [17].

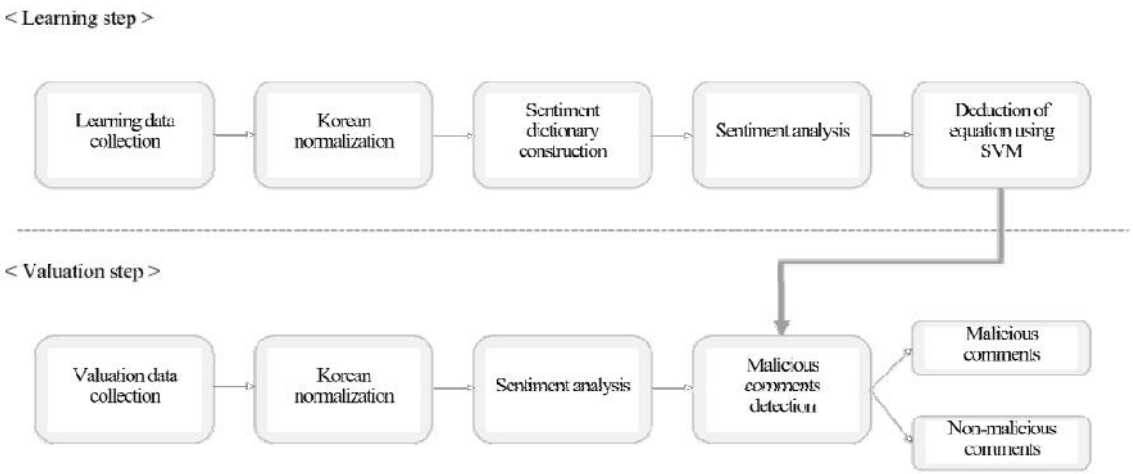


그림 2.5 감성분석과 SVN을 이용한 인터넷 악성댓글 탐지기법 FlowChart

안정국, 김희웅은 집단지성을 이용한 한글 감성 사전 구축에 대하여 연구하였다. 한국어의 감성 사전에 대한 연구가 미흡한 부분을 고려하여 집단지성의 활용을 위하여 소셜 네트워크를 통한 대학생 대상의 집단지성 감성 사전을 구축, API를 제공하였다. 참여자가 누적될수록 신뢰도가 높아지며 시간의 흐름에 따라 단어가 가지는 의미의 변화가 실시간으로 반영될 수 있다는 점에서 긍정적으로 볼 수 있다. 그러나 현재 API 제공 웹사이트 내에서 간단한 쿼리의 입력만을 제공하며 연구를 위한 API의 다운로드 및 사용은 불가능한 상태이다 그림 2.4는 본 논문을 기반으로 구성된 웹페이지 ‘오픈한글’의 캡처 화면으로, 감성어 수와 API 서버 상태를 확인할 수 있다.[12].

그림 2.6 집단 지성을 이용한 한글 감성 사전 웹페이지

임좌상, 김진만은 한국어 트위터의 감정 분류를 위한 기계학습의 실증적 비교에 대한 연구를 제안하였다. 다양한 기계학습 알고리즘을 사용하여 SVM(Support Vector Machine) 계열의 기계학습이 NB(Naive Bayes) 계열의 알고리즘에 비하여 더 높은 정확성을 보인다는 것을 확인하였다. 그러나 8가지 알고리즘의 구성과 실험방식에 대한 설명이 부족하고 한국어 데이터에 특화된 알고리즘이 포함되지 않았다 [7].

장재영은 온라인 쇼핑몰의 상품평 자동 분류를 위한 감성 분석 알고리즘을 제안하였다. 특정 제품군의 상품평이 가지는 정확도를 측정하여 기존 기계학습을 통한 감성 분석 알고리즘 연구와 유사한 정확도를 확보하였다. 그러나 부정적인 상품평에 상대적으로 취약한 모습을 보였으며, 띄어쓰기 및 문법적 오류에 대해 알고리즘에 오차가 있음을 드러내었다. 그림 2.7은 온라인 쇼핑몰의 상품평 분석 과정에 대한 상품평 분석 과정이다 [18].

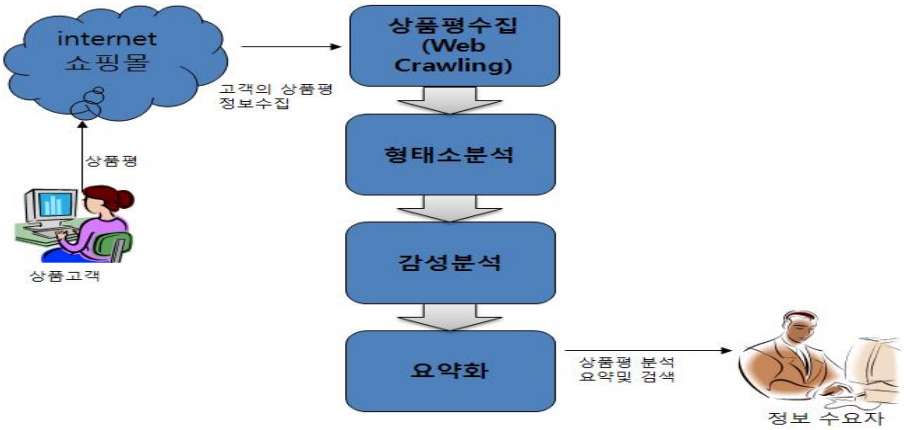


그림 2.7 온라인 쇼핑몰의 상품평 분석 과정

이경호, 이공주는 신문기사로부터 추출한 최근 동향에서 키워드를 추출하고 추출한 키워드를 이용해 수집된 트윗의 감성 분석을 통해 최근 동향에 대한 여론을 분석하였다. 신문기사를 k-means 알고리즘을 이용해 군집화 하고 단어의 출현빈도를 이용해 주제어를 선정하였다. 그림 2.6은 이 연구 과정의 전체적인 진행 과정을 그림으로 나타낸 것이다. 이 연구는 주제어에 대하여 수집된 트윗을 SVM을 이용하여 학습하여 긍정과 부정으로 분류하였다 [19].

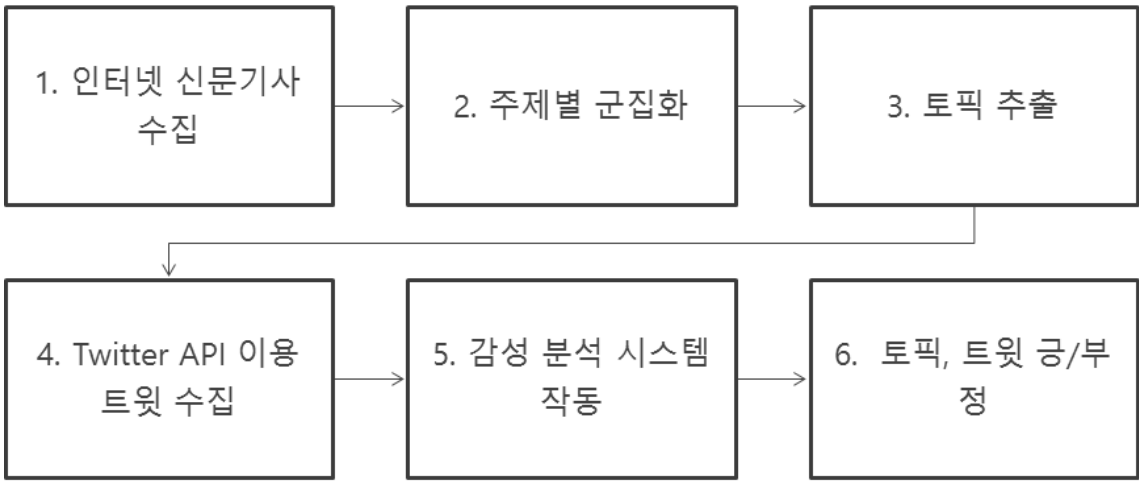


그림 2.8 신문기사를 통한 최근동향 감성분석 Overall Process

3. 자연어 처리 관련 기존연구

가. 형태소 분석

형태소 분석은 대부분 문법적 절차나 규칙에 기반을 두어 연구되어 왔다. 그 중에서 말뭉치에 기반을 둔 통계적 방식은 형태소 분석 방법으로 다양하게 연구되어 왔으며 새로운 연구로 쉽게 확장이 가능하다. 형태소 분리, 형태소 원형 복원, 형태소 태깅의 3가지 단계로 구성된다. 형태소 분리는 입력된 문장을 분리 가능한 경우를 검토하여 가장 적절한 방향으로 분석하는 것이다. 형태소 원형 복원은 한글 단어에 붙은 접미어 또는 조사, 어미 등을 없애고 단어의 사전적 정의에 맞는 원형으로 변경시키는 것이다. 형태소 태깅은 단어 요소들의 성격을 정의하는 것으로, 크게 부사, 명사, 형용사, 수사, 동사 등으로 구분할 수 있다. 각 과정들은 형태소 분석 방법에 따라 차례가 바뀌거나 동시에 처리될 수 있다. 형태소 분석은 문장의 정확한 감성 분석을 위해 필요하며 감성 사전을 이용하는 경우 정확한 형태소 분석 없이 효율적인 감성 사전의 활용이 어렵다 [20].

나. 한국어 조사의 특징

한국어 조사의 기본형은 100여개로 추정된다. 1999년에 간행된 [표준국어대사전]에 152개의 표준어 조사가 올라 있으며, 이 중에서 기본형 조사에서 파생된 조사나, 결합형으로 판단되는 조사를 제외해도 100여개에 이른다. 그러나 한국어의 조사는 두 개 이상이 결합되어 사용될 수 있기 때문에, 실제로 사용되는 조사의 수는 수백 개에 이르며 이는 문법적으로 볼 때 매우 많은 개수이다. 한국어 조사는 크게 격조사와 특수조사로 나눌 수 있으며, 격조사는 문장의 뜻에 영향을 주기보다는 문법적인 구성을 이루기 위한 조사로, 이/가, 을/를, 의 와 같은 조사를 예로 들 수 있다. 특수 조사의 경우 문장의 뜻에 영향을 주는 경우도 존재하며, 만, 까지, 조차, 보다, 뿐 등의 조사를 특수 조사의 예로 들 수 있다 [21].

본 논문은 문장의 구성에 기본 틀인 격조사 보다 문장의 뜻과 단어의 강조에 영향을 주는 특수 조사에 대하여 고려하였으며, 문장의 관계 및 단어의 극성을 계산하는데 최종적으로 딸려오는 조사에 의해 극성이 계산되는 알고리즘을 제안하였다.

본 논문에서는 형태소 분석을 위한 연구를 진행하는 과정에서 다양한 API 및 라이브러리를 사용하였다. 특히 한국어 형태소 분석을 지원하는 KoNLPy를 이용하여 KoNLPy에서 지원하는 코모란 형태소 분석기 및 Twitter 형태소 분석기를 이용하였으며, 두가지 형태소 분석기를 함께 사용하여 감성 사전의 한글화 및 트윗 원문의 분석 과정에서 단어의 원형을 추출하고, 조사, 명사, 형용사 등의 품사를 태그하는데 활용하였다 [22].

Ⅲ. 한글 트위터 데이터를 활용한 감성 분석 알고리즘

본 장에서는 한글 트위터 데이터를 활용한 감성 분석 알고리즘을 제안한다. 알고리즘은 짧은 문장의 감성 분석을 위해 감성 사전, 형태소 분석을 이용하고, 문장이 뜻한 의미를 분석하여 감성 정보를 제공한다.

1. 감성 분석 알고리즘

제안하는 알고리즘은 단어별로 분리된 한글 형태소에 대해서 주제어에 대하여 문장이 가지는 감성의 극성과 강도를 판단한다. 기존의 감성 사전 기반의 감성 분석 연구는 한국어 구성 요소에서 감탄사, 조사, 수사와 같은 단어들은 감성 분석과 대부분 관련이 없다는 이유로 고려 대상에서 제외하는 경향을 보였으나, 본 연구는 트위터 상의 한국어 문장에서 조사와 어미가 가지는 의미가 작지 않다고 고려하여 조사 및 어미를 통한 단어 간의 연결 관계를 알고리즘에 포함하였다.

제안한 감성 분석 알고리즘은 표 3.2에 수도코드로 설명되어 있으며 본 연구에서의 핵심 알고리즘이다. 이번 장에서는 알고리즘의 순서에 따른 진행 방식을 설명한다. 제안한 알고리즘은 문장을 구성하는 단어가 이미 극성 및 강도가 추출되어 감성 사전에 저장된 상태를 가정하고 서술한다.

알고리즘의 첫 단계에서는 문장을 형태소 단위로 분리한다. 형태소 단위로 분리하는 경우 단어와 조사까지 모두 분리되어 개별적으로 관리된다. 이 과정에서 각 형태소는 어근을 추출하여 단어를 최소 단위로 분리하고, 의미 있는 조사나 접미사 등의 단어가 연결된 경우 어미나 접미사를 추출한다. 예를 들어 “좋지만”, “느려도”과 같이 감성어에 연결된 부정적인 어미에 따라 다음 혹은 이전 문장과 극성이 연결되거나 반전된다. 따라서 “좋지만”은 “좋다 + -지만”, “느려도”는 “느리다 + -여도”와 같이 분류된다.

알고리즘의 두 번째 단계에서는 각 단어가 가지는 감성의 극성과 강도를 계산한다. 먼저 각 단어를 감성 사전에서 검색하여 추출한다. 이 과정에서 단어가 감성어이지만 감성 사전에 단어가 등록되어있지 않은 경우에는 신규 단어로, 감성어가 아닌 경우에는 중립어로 수작업으로 등록하게 된다. 극성이 중립인 경우에는 어절이 가지는 극성이 중립으로 계산되며, 긍정 또는 부정인 경우 강도를 추출한다. 감성어의 극성은 형태소의 품사에 따라 정의된다. 형용사나 명사, 동사의 경우 감성 어휘가 될 수 있다. “좋다”, “착하다”와 같이 대부분의 감성어는 형용사로 구성된다. 또한 “총체적 난국”, “강화하다”와 같은 경우 명사와 동사 역시 감성어로 분류할 수 있다. 부사의 경우 상기

전술한 감성어를 강조하는 “매우”, “너무”, “약간” 등과 같은 단어가 있으며, 이 경우 감성어에 가중치를 부여한다. 가중치가 없는 중립적인 단어의 경우 1의 값이 부여된다. 이는 강조의 정도에 따라 1보다 작은 값이 부여될 수도 있다. 부정형 부사의 경우 음수의 값을 부여받아 감성어의 극성을 반전시킨다. 동사의 경우 역시 부사와 동일하게, 중립적인 경우 1의 가중치를 가지지만 이후 어절 단위의 계산 과정에서 동사 전후에 위치한 부사의 가중치에 따라 동사의 극성 및 가중치가 결정된다. 또한 동사가 극성값을 가지는 경우에도 감성사전에서 나타나는 극성 및 가중치에 따라 문장이 가지는 극성값에 합연산으로 계산된다.

다음 단계에서는 조사와 어미를 분석하여 단어 간의 관계를 분석하고, 단어가 서로 연결되는 경우에 각 어절의 값을 더한다. 일반적으로 조사와 어미의 경우 극성에 영향을 주지 않지만, “-는데”, “-지만” 등과 같은 어미가 감성어휘와 함께 사용되는 경우 해당 형용사가 가진 극성의 값이 약화 또는 반전 될 수 있다. 한국어 조사의 경우 주격 조사, 보격 조사, 관형격 조사, 목적격 조사, 부사격 조사, 호격 조사, 인용격 조사, 접속조사, 보조사 등으로 나눌 수 있다. 이 중에서 조사 자체로서 뜻을 가지는 조사는 보조사로, 특수조사라고도 불리며, 대표적인 보조사로는 ‘-조차’, ‘-마저’, ‘-나마’와 같은 강조 또는 약화의 의미를 가지는 조사가 존재한다.

표 3.1 문장의 형태소 분석 예시

원문	새로 산 자전거가 색은 예쁜데 프레임이 너무 얇아서 불안하다.
명사	자전거, 색, 프레임
형용사	예쁘다, 얇다
부사	새로, 너무
동사	사다, 불안하다
조사	ㄴ, 가, 은, ㄴ데, 이, 서,다.

임의의 문장을 형태소로 분석하면 표 3.1과 같이 명사는 자전거, 색, 프레임, 형용사는 예쁘다, 얇다, 부사는 새로, 너무, 동사는 사다, 불안하다, 조사는 ㄴ, 가, 은, ㄴ데, 이, 서, 다 의 조사가 순차적으로 문장에 나타난다. 위 문장은 새로, 사다, 예쁘다, 너무, 얇다, 불안하다 등의 다양한 감성어가 존재한다. 각각의 극성은 한글화한 SentiWordNet을 이용해 추출한 결과 ‘새로’는 0.5의 긍정적인 값을 가진다. ‘예쁘다’는 1.5의 값을

가지고, ‘사다;는 0.25, 얇다는 긍정과 부정 두가지 모두 가능하지만 평균적인 값으로 부정적인 -0.5의 값을 SentiWordNet에서 보여준다. “너무”가 가지는 가중치는 1.25로, 강한 강조의 의미를 가지고 있다. 얇다와 불안하다는 각각 -0.5, -1.5의 값을 감성사전에서 찾을 수 있었다. 감성 사전에서 나타난 값으로 위 문장의 극성을 계산하게 되면, 아래와 같다.

문장의 극성 = 새로(0.5) + 사다(0.25) + 예쁘다(1.5) * -ㄴ데(0.5)
+ 너무(1.25) * 얇다(-0.5) + 불안하다(-1.5)
= 1.5 -2.25
= -0.75

“-ㄴ데”라는 연결 어미로 인해 극성의 강도가 약화되고 (0.5) “불안하다”의 앞에 부사와 형용사로 “너무 얇아서” 라는 단어의 존재로 부정적인 극성이 강화되어 결과적으로 자전거에 대한 부정적인 극성 -0.75를 가진 문장이 된다.

이와 같이 문장의 구성 요소를 전부 분석하여 의미 있는 형태소의 경우 가중치 및 극성을 추출하여 트윗의 극성을 계산하고자 하였다. 기존 연구에서는 이런 과정에서 한국어 문장의 연결과정에 핵심이 되는 조사에 대하여 배제하거나, 기계학습을 통해 명사 및 형용사, 부사의 순서로 나타나는 패턴을 분석하는 방식을 취하였으나, 이는 한국어가 가지는 특성의 일부를 배제하고 영어 문장을 분석하는 것과 유사하게 계산한 것으로, 본 연구에서는 한국어가 가진 특징적인 문법을 활용하여 기존 연구와 다른 점을 보인다.

본 연구에서 제안한 알고리즘의 프로그램 코드는 본 논문 결론 이후의 부록에 수록되어 있다.

표 3.2 감성 분석 알고리즘

감성 분석 알고리즘

입력: 트윗 내의 한 개의 문장

출력: 문장의 감성 분석 결과

```
입력된 문장(S )을 단어의 집합(L1, L2, …, Ln) 으로 분류
BEGIN
    weight = 1       //문장 내의 부사와 동사의 가중치를 임시저장
    for each Li (1 ≤ i ≤ n) {
        score[Li] = 0      //문장의 극성값
    }
    for each Li(1≤i≤n) {
        for each word w in Li {          //문장의 각 단어에 대하여
            if (POS(w) = 강조부사)
                weight = weight × point[부사]
            if (POS(w) = 감성어)
                score[Li] = score[Li] + polar[w] × weight
                weight = 1  //가중치 초기화
            if (POS(w) = 연결어미)
                score[Li] = score[Li] × point[연결어미]
            if (POS(w) = 부정부사)
                weight = weight × -1  //가중치 반전
            if (POS(w) = 부정동사)
                weight = weight × -1
            if (POS(w) = 종결어미)
                score[Li] = score[Li] × point[종결어미]
                weight = 1                //가중치 초기화
            if (POS(w) = 조사)
                score[Li] = score[Li] × point조사]
                weight = 1
            score[Li] = score[Li] × weight
        }
    score[S] = score[S] + score[Li]
    }
return score[S]
END
```

Ⅳ. 구현 및 실험

1. 실험 환경

본 실험은 제 3장에서 제시한 감성 분석 알고리즘을 기반으로 트위터 기반 감성 분석 시스템을 구축하였다. 프로그램 언어는 Python 2.7을 사용하였고 OS는 Windows 10 64bit이고, 개발도구는 Eclipse를 사용하였다. DBMS는 로컬에서 활용하기 편리한 SQLite3을 사용하였다. 한글 형태소 분석을 위해 한글 코모란 형태소 분석기를 사용하였다.

감성 사전의 구축을 위해 SentiWordNet 데이터를 한글로 번역하면서 불필요한 단어의 뜻 설명과 같은 데이터는 삭제하여 감성 사전을 경량화 하였다. SentiWordNet의 한글 번역에는 네이버 기계번역 API 및 Glosbe API를 이용하였다. Glosbe API의 속도 및 허용된 단어 조회수가 부족하여 진행 도중에 네이버 API로 교체하여 한글화를 진행하였다.

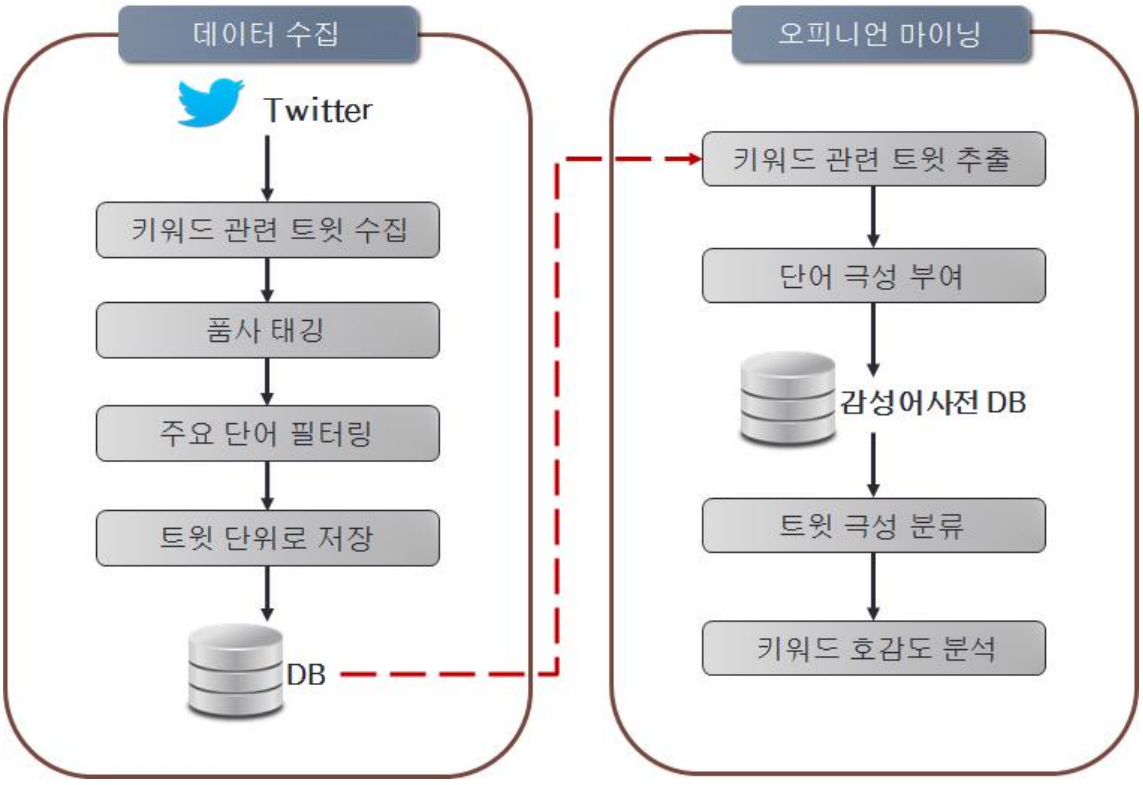


그림 4.1 트위터 데이터를 활용한 감성 분석 실험 모델

2. 실험 절차

본 실험은 트위터로부터 특정 대상에 관한 데이터를 수집하고 이를 통해 감성 사전을 구축하고 감성 분석 알고리즘을 적용하기 위해 사전 구축한 실험으로, 제안하는 실험 모델은 그림 4.1과 같다.

가. 데이터 수집

본 연구에서는 tweepy라는 트위터 API의 streaming 클래스를 사용하여 수집된 선거 데이터를 원시데이터로서 DBMS에 그대로 저장하였다.. 저장한 데이터의 요소는 그림 4.2와 같이 '트윗 원문(tweet_text), 게시된 장치(source)' 로 게시된 장치의 정보를 이용해 스팸이나 봇의 필터링이 가능하다. 수집한 데이터는 2016년 제 20대 국회의원 선거전의 데이터를 활용하였으며, 선거와 관련된 트윗은 짧은 글이지만 개인의 정치적 성향이 드러나기 때문에 데이터에 잡음이 상대적으로 적고 감성 분석을 하기 용이하다. 실험에 사용된 한글 트위터 데이터는 "새누리당", "더불어민주당", "정의당", "무소속" 을 검색어로 설정하여 약 12000개의 트윗을 제 20대 국회의원 선거전 7일간 수집하였다. (2016/3/26 ~ 2016/4/1)

tweet_text	source
RT @heureuxavecvous: 정청래 「#더컸유세단」에 이...	Twitter for Android Tablets
RT @halo1440: 은평갑엔.. 최홍재 아닌 더민주 박주...	Twitter for Android
아직 시간이 많이 남았으니...최선을 다해 승리하기 ...	Twitter Web Client
RT @isisanews: 더민주 홍영표 의원, 정의당 김성진 ...	Twitter Web Client
#국민의당 #계양을 #최원식https://t.co/T9HtJq6uRh	Cubi.so

그림 4.2 수집된 데이터 샘플

나. 형태소 분석

수집된 데이터에서 게시된 장치를 통해 스팸이나 봇, 기타 반복적인 뉴스 등의 필터링을 종료한 뒤에 각 트윗의 문장에서 불필요한 단어들을 제거하였다. RT 및 아이디, URL, 특수기호 등을 제거하고 한글로 구성된 트윗 원문을 생성하였다. 그리고 한글 코모란 형태소 분석기를 활용하여 트위터 원문을 각각 형태소 단위로 분리하였다. 이를 통해 형태소로 분리된 단어 하나 하나를 변수화 하였다. 이는 그림 4.3과 4.4에서 확인할 수 있으며, 이 과정을 거쳐 각 단어의 형태를 단순화 하고 트위터 원문을 단어 단위로 분석하여 감성사전에서 단어를 검색할 수 있도록 하였다.

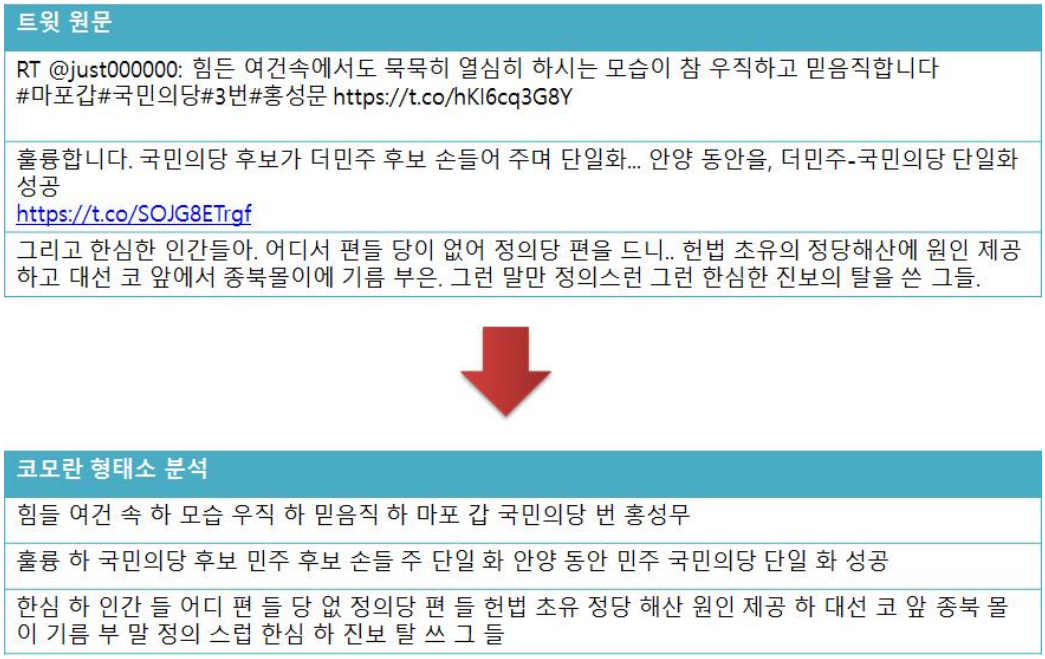


그림 4.3 코모란 형태소 분석기를 활용한 형태소 분석

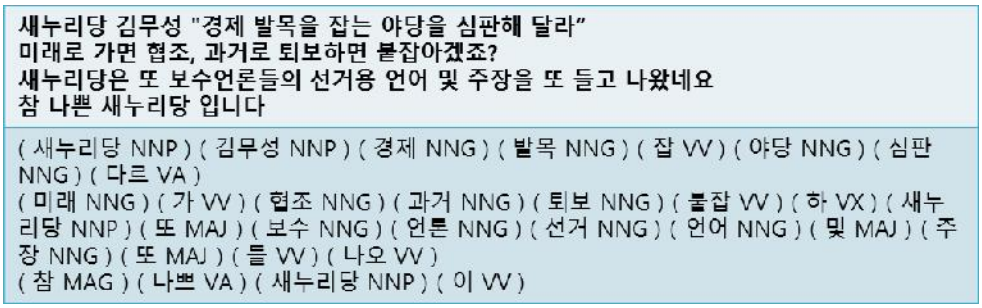
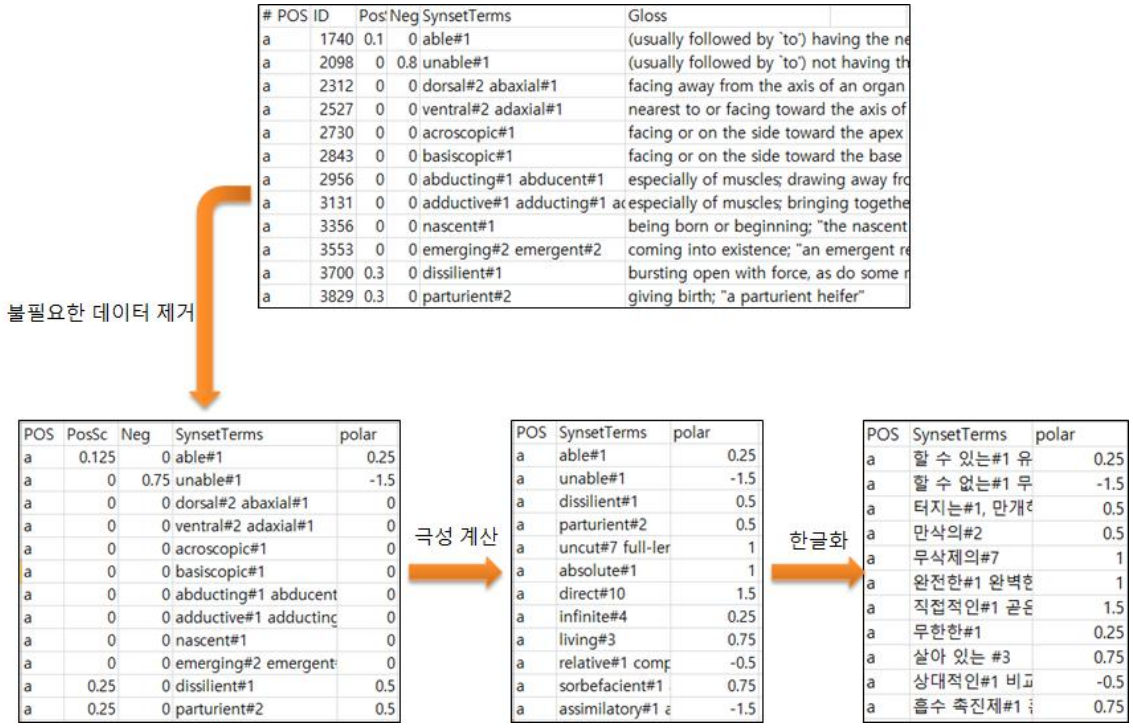


그림 4.4 형태소 태깅

다. 감성사전 활용

SentiWordNet을 활용하여 한글 감성사전을 제작하였다. 13MB의 텍스트 파일로, 내부에는 <품사,ID, 긍정강도,부정강도, 동의어묶음, 설명>으로 구성되어있다. 실제 감성사전 데이터 구축에 필요한 것은 동의어 묶음과 긍정강도와 부정강도, 품사뿐이기에 우선 기타 데이터를 전부 제거하였다. 그 결과 13MB의 텍스트 파일의 용량은 4MB로 감소하였다. 다음으로, SentiWordNet에는 극성이 없는 객관적인 단어의 비율이 높은데, 11만개의 데이터셋을 가지고 있던 SentiWordNet에서 극성이 없는 항목들을 제거하자 2만7천개의 데이터셋으로 감소하였다. 그 결과 800KB로 용량이 경량화 되었다. 이후 단어 묶음을

네이버 번역 API를 이용하여 한글화 하였다. 한글로 번역한 결과 값이 하나의 단어가 아닌 합성어인 경우는 알고리즘의 검색대상에서 제외하였다.



#	POS	ID	Pos	Neg	SynsetTerms	Gloss
	a	1740	0.1	0	able#1	(usually followed by `to') having the ne
	a	2098	0	0.8	unable#1	(usually followed by `to') not having th
	a	2312	0	0	dorsal#2 abaxial#1	facing away from the axis of an organ
	a	2527	0	0	ventral#2 adaxial#1	nearest to or facing toward the axis of
	a	2730	0	0	acroscopic#1	facing or on the side toward the apex
	a	2843	0	0	basiscopic#1	facing or on the side toward the base
	a	2956	0	0	abducting#1 abducent#1	especially of muscles; drawing away fro
	a	3131	0	0	adductive#1 adducting#1 ac	especially of muscles; bringing togethe
	a	3356	0	0	nascent#1	being born or beginning; "the nascent
	a	3553	0	0	emerging#2 emergent#2	coming into existence; "an emergent re
	a	3700	0.3	0	dissilient#1	bursting open with force, as do some r
	a	3829	0.3	0	parturient#2	giving birth; "a parturient heifer"

POS	PosSc	Neg	SynsetTerms	polar
a	0.125	0	able#1	0.25
a	0	0.75	unable#1	-1.5
a	0	0	dorsal#2 abaxial#1	0
a	0	0	ventral#2 adaxial#1	0
a	0	0	acroscopic#1	0
a	0	0	basiscopic#1	0
a	0	0	abducting#1 abducent	0
a	0	0	adductive#1 adducting	0
a	0	0	nascent#1	0
a	0	0	emerging#2 emergent	0
a	0.25	0	dissilient#1	0.5
a	0.25	0	parturient#2	0.5

POS	SynsetTerms	polar
a	able#1	0.25
a	unable#1	-1.5
a	dissilient#1	0.5
a	parturient#2	0.5
a	uncut#7 full-ler	1
a	absolute#1	1
a	direct#10	1.5
a	infinite#4	0.25
a	living#3	0.75
a	relative#1 comp	-0.5
a	sorbefacient#1	0.75
a	assimilatory#1 a	-1.5

POS	SynsetTerms	polar
a	할 수 있는#1 유	0.25
a	할 수 없는#1 무	-1.5
a	터지는#1, 만개	0.5
a	만삭의#2	0.5
a	무삭제의#7	1
a	완전한#1 완벽한	1
a	직접적인#1 곧은	1.5
a	무한한#1	0.25
a	살아 있는 #3	0.75
a	상대적인#1 비교	-0.5
a	흡수 촉진제#1	0.75

그림 4.5 SentiWordNet의 한글화 과정

극성의 계산은 3장에서 제안한 알고리즘에 최적화하기 위하여 SentiWordNet에서는 분리되어있는 긍정-부정 극성의 값을 하나로 만들어 계산하였다. 긍정강도에서 부정강도를 차감하여 각 단어가 가지는 극성 값을 알고리즘에 활용하기 용이하도록 단일화하였다.

3. 실험 결과

감성 분석 알고리즘을 적용하여 각 키워드에 대한 검색 결과를 수집하였다. 수집된 트윗의 개수는 더불어민주당이 4620개, 새누리당이 3013개, 국민의당이 3767개, 정의당이 1162개, 무소속에 관한 트윗이 630개로 총 10031개의 트윗으로 감성 분석을 진행하였다. 중복 데이터 및 트윗봇 등의 데이터를 필터링하며 수집한 자료의 개수가 감소하였다. 제안한 알고리즘은 모든 수집된 데이터에 적용되어 계산하였으나, 수작업을 통해 각 트윗이 가지고 있는 극성을 분류하여 정확도를 평가하기에는 많은 데이터를 가지고 있어, 계산된 결과에서 500개의 트윗을 무작위로 선택하여 직접 극성을 분류한 뒤 알고리즘이 정상적으로 작동하였는지 결과를 비교하였다.

아래의 표 4.1에서는 알고리즘으로 분류한 데이터가 실제로 정확하게 계산되었는지 파수치화 하여 나타낸다. 표상에서 긍정, 중립, 부정 항목은 트윗의 판별된 극성을 의미하며, 왼쪽 첫 열의 항목에서 극성은 해당 키워드에 대하여 수작업으로 분류한 트윗의 극성에 대한 개수이고, 정분류는 트윗이 가지는 극성이 알고리즘으로 정확하게 분류된 경우, 오분류는 트윗의 극성이 잘못 분류된 경우를 의미한다, 본 연구에서는 트윗 극성 분류의 정확성의 척도로 정확도를 사용하였고, 정확도는 수집된 트윗의 개수에서 정분류된 트윗의 개수를 백분율로 나타낸 수치이다.

표 4.1 제안한 알고리즘의 감성 분석 정확도 평가

항목	긍정	중립	부정	총계
극성수	165	84	251	500
정분류수	136	67	185	392
오분류수	29	17	66	108
정확도	82.4%	80.1%	74.0%	78.4%

종합 감성 분석 정확도의 평가 결과 알고리즘의 정확도는 약 78.4% 로 나타났으며, 긍정적인 트윗의 경우 82.6%의 정확도를 보였으나, 부정적인 트윗의 경우 정확도가 74%로 비교적 낮은 정확도를 보였다. 이는 실제 트윗의 내용 중 반어법이나 비꼬는 내용에 대하여 제안한 알고리즘이 약점을 보이는 것이 있으며, 문법적인 오류나, 감성사전에 등록되지 않은 단어가 많을수록 알고리즘의 정확도가 떨어지는 모습을 보였기 때문이다.

아래 표는 정분류된 트윗과 오분류된 트윗의 일부를 예시로 분석한 것으로, 문법적으로 큰 오류가 없고 긍정적인 단어와 부정적인 단어가 뒤섞여 있지 않은 경우 일반적으로 오차없이 정분류 되었으나, 표 4.2에서 보이는 것과 같이 오분류 된 사례를 많이 찾을 수 있었다. 표 4.2에서 보이는 4개의 트윗 중에서 3번째 트윗이 긍정적인 내용이지

만 부정적인 극성으로 오분류된 이유는 ‘안되는데’, ‘아닐까’ 와 같은 단어의 극성이 부정적으로 판별되었기 때문이고, ‘전무후무하다’라는 단어가 감성어 사전에 존재하지 않았고, 문장에서 긍정적인 단어가 ‘대단하다’ 하나뿐이었기 때문에 결과적으로 해당 트윗이 부정적으로 계산되었다. 이는 감탄하는 어조로 사용된 부정형 부사 및 부정형 동사의 경우 부정적으로 계산되는 문제점을 보인다고 할 수 있으며, 영어를 기반으로 제작된 감성사전에 등록되지 않은 한국어, 사자성어 등에 대하여 취약점을 보인다고 할 수 있다.

표 4.2의 4번째 트윗은 부정적인 극성을 긍정 극성으로 오분류 하였는데, 이는 ‘개같은 이라는 형용사가 기본적인 단어가 아닌 조합어로 ’ 개 ‘와 ’ 같다 ‘를 따로 분석하여 긍정적인 뜻으로 분석하였고, 이후 뒤 문장에서 ’ 야권섬향 ‘의 오타로 인한 부정확성과, ’ 진정성 ‘과 ’ 후보님’에서 긍정적인 극성을 추출하여 극성 분석에 오류가 발생하였다. 이에 따라서 조합어에 대한 감성사전에의 추가와 알고리즘의 수정이 알고리즘의 정확도를 높이기 위해 필수적으로 요구된다.

표 4.2 정분류 및 오분류된 트윗의 대표적 예시

트윗 원문	극성 분류
무슨 진리의 말씀인양 퍼나르던 친노문빠들이 드글드글 득실득실 우글우글 거리는 #더민주 #국보위_김종인 집단이랑 연대를 해달라고??????	부정 극성으로 정분류
기분좋은 출발!!! 정동영후보 안철수후보 승리가 보입니다ㅎㅎ 사랑합니다 ♡ 국민의당 필승!!!!!! 정동영 36.6 김성주 33.9	긍정 극성으로 정분류
국민의당 대단합니다 창당2개월도 안되는데 지지율을 보라 정당사상 전무후무한 일 아닐까	긍정적인 트윗을 부정 극성으로 오분류
안철수의 개같은욕심에 속지마세요 진정성있는 야권섬향의 국민의당 후보님들	부정 극성을 긍정 극성으로 오분류

본 연구의 실험 결과는 표 4.3에서 보이는 것과 같이 기존 연구 중 온라인 쇼핑몰의 상품평 데이터를 활용한 감성 분석 알고리즘 연구(이하 기존연구1)의 결과 및 트위터 데이터를 활용한 기계학습 연구(이하 그러나 기존연구2에서 다양한 기계학습 알고리즘의 분석 결과로 나타난 정확도와 비교한 경우 대부분의 나이브 베이시안 분류기보다 높은 수치의 정확도를 보였으며, SVM을 변형한 기계학습 분류기와 비교할 때 평균적으로 유사한 수치의 정확도를 보였다.

표 4.3 제안하는 시스템과 감성분석 기존연구와의 비교

	기존연구1 [18]	기존연구2 [7]	제안한 알고리즘
정확도	80%	70% ~ 84.0%	78.4%
사용한 데이터	온라인 쇼핑몰 리뷰	한국어 트위터	한국어 트위터
감성분석 방법	감성사전과 극성계산 알고리즘	Support Vector Machine 및 Naive Baysian Classifier	감성사전과 극성계산 알고리즘
감성사전 구축	수작업 및 기계학습으로 구축	X	범용적 영어 감성사전 한글화
조사문법 고려	X	X	O

본 논문에서 제안한 알고리즘은 한국어 데이터 분석 과정에서 단어와 문장의 관계에 조사가 가지는 중요성을 고려하여 구성되었으나, 조사를 포함하지 않은 기존연구1과 비교하여 평균적으로 유사하거나 약간 부족한 정확도를 보였다. 또한 기존연구2와 평균적으로 유사한 정확도를 보였다. 이 결과는 극성 분석에 사용된 기존연구1에서 사용된 것과 같이 주제에 특화되어 직접 제작한 감성 사전이 아니라 범용 적으로 사용할 수 있는 해외의 감성 사전을 이용한 것이 주된 이유로 분석된다. 또한 조사의 사용이 문법적으로 맞지 않거나 생략된 문장의 경우 기존 연구1에서 사용한 감성 분석 알고리즘 연구보다 오히려 취약한 것으로 파악된다. 이런 한글 파괴에 대한 특성이 일부 트윗에서 매우 심화되어 나타나기 때문에, 본 논문에서 제안한 알고리즘을 트위터에 최적화할 필요성이 있으며, 또는 문법적으로 문제가 거의 없는 토론이나 신문의 사설과 같은 데이터를 기준으로 알고리즘을 테스트 하여 제안한 알고리즘이 SNS보다 더 최적화된 분야가 있는지 확인할 근거가 된다.

Ⅴ. 결 론

트위터에서 수집된 글은 비정형화 되어 문법과 정확히 일치하지 않아 기계학습에 비교하여 부족할 것으로 예상되었으나, 본 연구의 분석을 이용해 도출된 결과가 실제 통계로 나타난 조사 대상의 결과와 유사한 경우 선호도 및 지지도 조사에 활용성을 가진다. 기존 연구 중 한글 감성어 분석을 이용하는 경우 키워드 전후의 단어가 가진 극성만을 검색하는 경향을 보였으나, 본 연구에서는 단어와 단어의 연결어미를 분석하여 문장이 가지는 문맥상의 의미를 파악하고자 하였다.

본 연구는 몇 가지의 한계점을 가진다. 첫 번째로, 감성 사전의 구축을 영어권 사전에 의존하여 실제 한국어와 단어의 의미가 다르게 적용될 수 있다는 것이다. 또한 문장의 구성에 따라 같은 단어라도 문맥상 다른 해석이 가능하기도 할 뿐 아니라 반어법을 사용한 문장은 해석에 난해함이 발생할 수 있는데 제안한 알고리즘으로 위와 같은 문제점을 완전히 해소할 수 없었다. 이는 트위터 상에서 사용자들의 문법 무시와 신조어의 과도한 사용으로 인해 알고리즘 및 감성 사전이 오작동 하는 것이 가장 큰 이유인 것으로 보인다.

트위터 데이터의 감성 분석을 정확하게 실행할 수 있는 경우 마케팅이나 정책의 제안과 같은 중요한 사회 이슈에 대하여 SNS를 활용할 수 있다는 점에서 큰 영향을 줄 수 있을 것으로 예상된다. 이후 극성 분석 알고리즘 및 감성 사전의 정확도를 개선하고 GUI를 통해 사용자가 복잡한 작업과정 없이 검색어에 대한 감성 분석 결과 및 통계 데이터를 얻을 수 있는 감성 분석 시스템을 개발할 계획이다.

참고문헌

[1] Gerlitz, C., & Rieder, B. (2013). Mining one percent of Twitter: Collections, baselines, sampling. M/C Journal, 16(2).j

[2] Pang, B., & Lee, L. (2008). Opinion mining and sentiment analysis. Foundations and trends in information retrieval, 2(1-2), 1-135.

[3] Liu, B. (2012). Sentiment analysis and opinion mining. Synthesis lectures on human language technologies, 5(1), 1-167.

[4] Liu, B., & Zhang, L. (2012). A survey of opinion mining and sentiment analysis. In Mining text data (pp. 415-463). Springer US.

[5] 배정환, 손지은, & 송민. (2013). 텍스트 마이닝을 이용한 2012 년 한국대선 관련 트위터 분석. 지능정보연구, 19(3), 141-156.

[6] 손성일, & 박찬곤. (2014). 빅데이터 선호도 분석 시스템 설계. 멀티미디어학회논문지, 17(11), 1286-1295.

[7] 임좌상, & 김진만. (2014). 한국어 트위터의 감정 분류를 위한 기계학습의 실증적 비교. 멀티미디어학회논문지, 17(2), 232-239.

[8] Miller, G. A. (1995). WordNet: a lexical database for English. Communications of the ACM, 38(11), 39-41.

[9] Baccianella, S., Esuli, A., & Sebastiani, F. (2010, May). SentiWordNet 3.0: An Enhanced Lexical Resource for Sentiment Analysis and Opinion Mining. In LREC (Vol. 10, pp. 2200-2204).

[10] Tong, S., & Koller, D. (2002). Support vector machine active learning with applications to text classification. The Journal of Machine Learning Research, 2, 45-66.

[11] Manning, C. D., Raghavan, P., & Schütze, H. (2008). Introduction to information retrieval (Vol. 1, No. 1, p. 496). Cambridge: Cambridge university press.

[12] 안정국, & 김희웅. (2015). 집단지성을 이용한 한글 감성어 사전 구축. 지능정보연구, 21(2), 49-67.

[13] 배원식, 차정원. (2013). “정서분석을 위한 의견관계 자동추출,” 한국정보과학회논문지: 소프트웨어 및 응용, 제40권, 제5호, pp. 473-481.

[14] Shin, Hyopil, Munhyong Kim, Yu-Mi Jo, Hayeon Jang, and Andrew Cattle. (2013). KOSAC(Korean Sentiment Analysis Corpus): 한국어 감정 및 의견 분석

코퍼스, Information and Compuation, pp 181-190.

[15] Kwak, H., Lee, C., Park, H., & Moon, S. (2010, April). What is Twitter, a social network or a news media?. In Proceedings of the 19th international conference on World wide web (pp. 591-600). ACM.

[16] Huberman, B. A., Romero, D. M., & Wu, F. (2008). Social networks that matter: Twitter under the microscope. Available at SSRN 1313405.

[17] 홍진주, 김세한, 박제원, 최재현. (2016). 감성분석과 SVM을 이용한 인터넷 악성 댓글 탐지 기법. 한국정보통신학회논문지, 20(2), 260-267.

[18] 장재영. (2009). 온라인 쇼핑몰의 상품평 자동분류를 위한 감성분석 알고리즘. 한국전자거래학회지, 14(4), 19-33.

[19] 이경호, & 이공주. (2013). 신문기사로부터 추출한 최근동향에 대한 트위터 감성분석. 정보처리학회논문지. 소프트웨어 및 데이터 공학, 2(10), 731-738.ISO 690

[20] 이재성. (2011). 한국어 형태소 분석을 위한 3 단계 확률 모델. 정보과학회논문지: 소프트웨어 및 응용, 38(5), 257-268.

[21] 임동훈. (2004.6). 한국어 조사의 하위 부류와 결합 유형. 국어학, 43, 119-154.

[22] Park, E. L., & Cho, S. (2014, October). KoNLPy: Korean natural language processing in Python}. In Proceedings of the 26th Annual Conference on Human & Cognitive Language Technology.

Abstract

Implementation of the Sentiment Analysis Algorithm with Korean Twitter Data

Youn, Han Jung

(Supervisor Suk, Sang Kee)

Dept. of Computer Science and Engineering

Graduate School of Seoul National University of Science and Technology

As the development of mobile devices, social networks are affecting the composition of example to dig deeply into social life issues and public opinion must also be the head of an expression of personal opinion. Twitter in particular the relationships and the influence of offline is much less mention of the various socio-political issues and the specific product or person. The sentiment analysis of Twitter data is huge active in foreign country, but this Korean sentiment analysis based on Twitter data is a study on the inactive situation.

The sentiment analysis algorithms to control the lack of pre-deployment and are focused on the sentiment analysis with machine learning research, Sentiment analysis is one of the most important technique on web marketing and political society, but sentiment analysis algorithm based on Korean grammar is inadequate and insufficient state.

In this paper, we proposed an algorithm for Korean sensitivity analysis using Twitter data . By analyzing and calculating the polarity of the main sentence of tweets related to the proposed unit ‘morpheme’, sentiment analysis algorithms are optimized for simple and short sentence configurations such as Twitter data analysis.

부록

◆ 감성 분석 알고리즘

다음은 한글 트위터 데이터를 활용한 감성 분석에 알고리즘을 정의한다.

```
# -*- coding: utf-8 -*-

from konlpy.tag import Komoran
from nltk.corpus import sentiwordnet as swn
from HTMLParser import HTMLParser
import sys
import os
import sqlite3
import csv, collections
import numpy as np

reload(sys)
sys.setdefaultencoding('utf-8') #for korean insert

def load_sent_word_net():  #SentiWordNet으로부터 단어의 극성값을 추출
    sent_scores = collections.defaultdict(list)
    with open(os.path.join("KOR_SentiWord.txt"),"r") as csvfile :
        reader = csv.reader(csvfile, delimiter='\t', quotechar='"')
        for line in reader:
            if line[0].startswith("#"):
                continue
            if len(line)==1:
                continue
            POS,SynsetTerms,Polar = line
            if len(POS)==0:
                continue
            for term in SynsetTerms.split(" "):
                term = term.split("#")[0].decode('cp949')
                term = term.replace("-", " ").replace("_"," ")
```

```
                key = "%s"%(term.split("#")[0])
                sent_scores[key].append((float(Polar)))
    for key, value in sent_scores.iteritems():
        sent_scores[key] = np.mean(value, axis=0)
    return sent_scores

def unescape_entities(value, parser=HTMLParser()):
    return parser.unescape(value)

def process(ob):
    if isinstance(ob, list):
        return [process(v) for v in ob]
    elif isinstance(ob, dict):
        return {k: process(v) for k, v in ob.items()}
    elif isinstance(ob, str):
        return unescape_entities(ob)
    return ob

KOR_SENT_DIC = load_sent_word_net()
km = Komoran()
con = sqlite3.connect("kotwit_politic.db")
cursor = con.cursor()
cursor.execute("select distinct tweet_text from kotwit")

count=0
score_text=[]
for row in cursor:
    count+=1
    km_pos = km.pos(row[0].encode('utf-8'))
    weight = 1
    score=0
    print "row[0]: ",row[0].encode('utf-8')
    for item in km_pos:
```

```
            type = item[1]
            if type.startswith('S'): continue
            if type.startswith('E'): continue
            if type.startswith('X'): continue
            if type.startswith('J'):
                if KOR_SENT_DIC[item[0]]:
                    score += KOR_SENT_DIC[item[0]] * weight
                    weight=1
            if type.startswith('VA'):
                if KOR_SENT_DIC[item[0]]:
                    score += KOR_SENT_DIC[item[0]] * weight
                    weight=1
            if type.startswith('M'):
                if KOR_SENT_DIC[item[0]]:
                    weight = weight * KOR_SENT_DIC[item[0]]
            if type.startswith('N'):
                if KOR_SENT_DIC[item[0]]:
                    score += KOR_SENT_DIC[item[0]] * weight
                    weight=1
            if type.startswith('VV'):
                if KOR_SENT_DIC[item[0]]:
                    score += KOR_SENT_DIC[item[0]] * weight
                    weight=1
        score_text.append((score,row[0].encode('utf-8')))

for score, text in score_text:
    if score>0:
        polar = "1"
    elif score==0:
        polar = "0"
    elif score<0:
        polar="-1"
    print "극성:",score,
```

```
        print "원문:",text
        cursor.execute("update kotwit set polar="+polar+",power="+str(score)+" where
tweet_text is '"+text+"';")
        con.commit()
print"count: ", count
cursor.close()
```

◆ 트위터 데이터 수집 시스템

다음은 트위터 데이터를 검색어에 기반하여 수집하는 시스템이다.

```
# -*- coding: utf-8 -*-
from tweepy.streaming import StreamListener
from tweepy import OAuthHandler
from tweepy import Stream
import json
from konlpy.tag import Komoran
from sklearn.feature_extraction.text import TfidfVectorizer
import sys
import sqlite3

access_token = "4093927464-crUhMFsJDm0B0HmnhTn8g0vynopIk4riOX66TRr"
access_token_secret = "jD3xZEoSkoosq4eKz3nxlPOORnsm8tikLCDNq3j4d3meB"
consumer_key = "Eg5gUwo64P3Evi7hnE4PpyHwu"
consumer_secret = "aMHoj7tOIVPm5dlKChGMsc5HIuyv8Q8ymBwFJqBlhFWfSa8lFW"
nav_key="7ffa7f99c6b6f89aa14e5f6e41b464fa"
reload(sys)
sys.setdefaultencoding('utf8') #for korean insert
km=Komoran()
vectorizer = TfidfVectorizer()
class StdOutListener(StreamListener):

    nTweet  = 0
    nPassed = 0

    def on_data(self, data):
        self.nTweet = self.nTweet + 1
        tweet = json.loads(data)
        con = sqlite3.connect("kotwit_politic_2ND.db")
        cursor = con.cursor()
        cursor.execute("CREATE TABLE IF NOT EXISTS kotwit(tweet_text text
```

```
primary key, source text, time text,km_keywords text)“)
        print 'text:', tweet['text']
        print 'source:', tweet['source'] #Android, iPhone, iOS,
        print 'created_at:', tweet['created_at']
        “““
        if False:
            nouns = kkma.nouns(tweet['text'])
            print “[NOUN]“,
            myprintlist(nouns)“““
        km_pos = km.pos(tweet['text'])
        km_keywords = ““
        for item in km_pos:
            type = item[1]
            if type.startswith('N'):    # nouns
                print “(“, item[0], item[1], “)“
                km_keywords += item[0]+“ “
            elif type.startswith('X'):  #  verbs
                print “(“, item[0], item[1], “)“
                km_keywords += item[0]+“ “
            elif type.startswith('V'):  #  adjectives
                print “(“, item[0], item[1], “)“
                km_keywords += item[0]+“ “

        cursor.execute(“INSERT OR REPLACE INTO kotwit
VALUES('“+tweet['text']+“','“+tweet['source']+“','“+tweet['created_at']+“','“+km_keyw
ords+“')“)
        self.nPassed = self.nPassed +1

        “““
        [u'contributors', u'truncated', u'text', u'is_quote_status', u'in_reply_to_statu
        s_id', u'id', u'favorite_count', u'source', u'retweeted', u'coordinates', u'time
        stamp_ms', u'entities', u'in_reply_to_screen_name', u'id_str',
u'retweet_count',
```

```
            u'in_reply_to_user_id', u'favorited', u'user', u'geo', u'in_reply_to_user_id_st
        r', u'lang', u'created_at', u'filter_level', u'in_reply_to_status_id_str', u'pla
        ce']
        """
        if (self.nTweet % 1 == 0): # statistics
            print "nTweet=", self.nTweet, "nPassed=", self.nPassed
        con.commit()
        con.close()
        return True

    def on_error(self, status):
        print status

#This is a basic listener that just prints received tweets to stdout.

#This handles Twitter authetification and the connection to Twitter Streaming API
l = StdOutListener()
auth = OAuthHandler(consumer_key, consumer_secret)
auth.set_access_token(access_token, access_token_secret)
stream = Stream(auth, l)
print "Finished set up !"

#to insert keyword
while True:
    try:
        track =
[u'국민의당',u'새누리당',u'더민주',u'더불어민주당',u'정의당',u'무소속']
        stream.filter(track=track, languages=["ko"])

    except UnicodeEncodeError:
        print UnicodeEncodeError
    except sqlite3.OperationalError:
        print sqlite3.OperationalError
```

◆ 트위터 데이터 수동 극성 분류

다음은 트위터 데이터를 랜덤으로 수집하여 극성을 수동으로 분류하기 위한 프로그램이다.

```
# -*- coding: utf-8 -*-
from konlpy.tag import Twitter
import sqlite3

twitter = Twitter()
con = sqlite3.connect("kotwit_politic.db")
cursor = con.cursor()
cursor.execute("SELECT * FROM kotwit where hand is null ORDER BY RANDOM() LIMIT 8;")
count=0
hand_text=[]
print "시작"
for row in cursor:
    text = row[0].encode('utf-8')
    print "row: ",text
    text_pos=twitter.pos(unicode(row[0]), norm=True, stem=True)
    #for a,b in text_pos:
        #print a.encode('utf-8'),b
    print "================================================"
    while True:
        hand_polar = input("위 트윗이 긍정이면 1, 부정이면 -1, 중립이면 0 을 입력하세요")
        hand_polar= str(hand_polar)
        if(hand_polar == '-1')|(hand_polar == '1')|(hand_polar == '0'):
            break
    count+=1
    print row[2], row[3], count
    hand_text.append((hand_polar,text))
```

```
for hand, text in hand_text:
    cursor.execute("update kotwit set hand="+hand+" where tweet_text is '"+text+"';")
    con.commit()
cursor.close()
```

감사의 글

본 연구는 서울과학기술대학교 일반대학원 컴퓨터공학과에서 이루어졌으며, 이 연구를 마무리할 때까지 정신적, 학문적으로 지도해주시고 연구를 진행하는 동안 많은 분야에 도움 주신 지도교수님이신 석상기 교수님께 먼저 감사드립니다.

석사 과정 중에 진로상담 및 전공지식, 그리고 시야를 넓힐 수 있는 기회를 제공해 주신 안희준 교수님, 심원 교수님, 박종혁 교수님께 감사드립니다. 연구 및 학교생활을 하는데에 아낌없는 지원과 도움 주신 컴퓨터 공학과 모든 교수님들, 학사일정 및 인간관계에 도움을 주신 컴퓨터공학 사무실 정두환 선생님, 김종복 선생님, 조한나 선생님께 감사드리고, 연구실을 혼자 사용하는 동안 힘들지 않을 수 있게 해준 원동력인 이승강 선배와, 데이터베이스 동아리 후배들과 UCS 연구실 선배(주재웅, 이정규, 임형진, 오대명), 동기 Pavan에게 감사드립니다.

마지막으로 석사 과정을 무사히 마칠 수 있도록 보살펴 주신 부모님과 호주에서 저를 위해 기도해준 가족들에게 감사의 말씀을 전합니다.