

학사학위 논문

순환신경망과 합성곱 신경망을 이용한
한국어 감성분석

Sentiment Analysis based on Korean using Recurrent Neural Network
and Convolutional Neural Network

2019年 11月 26日

한국외국어대학교
공과대학 정보통신공학과

이 소 향

순환신경망과 합성곱 신경망을 이용한 한국어 영화 리뷰 감성분석

지도 교수 한 희 일

한국외국어대학교
공과대학 정보통신공학과

이 소 향

2019年 11月 26日

순환신경망과 합성곱 신경망을 이용한 한국어 영화 리뷰 감성분석

Sentiment Analysis based on Korean using Recurrent Neural Network
and Convolutional Neural Network

소 속:	공 과 대 학 전 공:	정보통신공학
학 번:	201502373 성 명:	이 소 향

이 논문을 정보통신공학과 졸업논문으로 제출하오니 승인하여 주십시오.

2019年 11月 26日

성 명: 이 소 향(인)

위 학생의 논문 제출을 승인함.

2019年11月 26日

지 도 교 수: 한 희 일(인)

국 문 초 록

소셜 네트워크가 활성화되어 사용자들이 올린 SNS 텍스트 데이터를 사용하여 다양한 연구들이 진행되고 있다. 그 중 후기는 제품과 서비스의 실질적 구매에 영향을 미치기 때문에 기업의 중요한 마케팅 전략으로 활용될 수 있다. 이에 언어에 포함된 의견과 태도와 같은 주관적인 정보를 탐지하는 감성분석 연구가 이루어졌다. 초기에는 감성 강도를 나타낸 감성사전을 통해 이루어졌으나 최근에는 딥러닝 기반 감성분석 연구가 증가하고 있다. 감성분석과 함께 말뭉치를 학습해 다음에 나올 단어 혹은 문장을 예측하는 언어모델에서도 딥러닝을 이용하고 있다. 본 논문에서는 네이버 영화 리뷰 말뭉치를 이용해 문장의 감성을 긍정 또는 부정, 더 나아가 중립으로 판별하는 감성분석을 시도한다. 기존 감성분석에 사용한 감성사전을 구축하지 않고 문장을 구성하는 말뭉치로만 감성을 예측한다. CNN과 LSTM layer를 포함하는 형태소 단위 모델을 구축하였으며 성능지표로는 정확도(Accuracy)를 측정하였다. 실험 결과로는 CNN과 LSTM layer를 둘 다 가지는 모델이 정확도가 더 뛰어났다. 한국어 감성분석 실험을 통해 영어에 비해 연구가 부족한 한글 기반 언어처리 및 음성인식 분야에 활용될 것으로 기대한다.

목 차

내용

1. 서론	9
1.1. 연구 배경	9
1.2. 연구 목적	11
2. 관련연구	12
2.1.1 감성사전을 활용한 감성 분석	12
2.1.2 신경망을 활용한 감성 분석	13
2.2. 감성분석에 사용되는 모델	17
2.2.1 벡터 할당	17
2.2.1. RNN의 기본 구조	18
2.2.2 LSTM의 기본구조	20
2.2.3 CNN의 기본 구조	24
3. 연구 방안	26
3.1. 실험 환경	26
3.1 실험 데이터	26
3.2. 형태소 기반의 감성분석 모델	28
3.2.1. 한국어 형태소 분석	28
3.2.2. CNN 모델	30
3.2.3. CNN-LSTM 합성 모델	35
4. 연구 결과 및 분석	37
4.2. 실험 결과	37

4.3. 결과 분석.....	42
5.결론 및 향후연구.....	47
6. 참고 문헌	48

그림 목 차

<그림 1> RECURSIVE NEURAL TENSOR NETWORK 모형	13
<그림 2> BIDIRECTIONAL RNN 모형	14
<그림 3> CONVOLUTIONAL NEURAL NETWORK 모형	14
<그림 4> MULTI-CHANNEL CNN 모형	15
<그림 5> SKIP-CONNECTED LSTM 모형	16
<그림 6> WORD2VEC 구조	17
<그림 7> RNN의 기본 구조	18
<그림 8> LSTM 셀 구조	20
<그림 9> LSTM 동작 예시	22
<그림 10> TANH 함수(A), SIGMOID 함수(B)	22
<그림 11> CNN이 이미지에서 합성곱 연산을 수행하는 모습	24
<그림 12> MAX POOLING	25
<그림 13> CNN 동작 예시	25
<그림 14> CNN 모델 흐름도(A), CNN-LSTM 모델 흐름도(B)	29
<그림 15> RELU 함수	31
<그림 16> 시그모이드 함수	32
<그림 17> LSTM 계층 모습	35
<그림 18> CNN 계층의 출력 후 LSTM 계층의 입력이 되는 모습	36
<그림 19> 파라미터에 따른 CNN 모델의 정확도	40
<그림 20> 파라미터에 따른 CNN-LSTM 모델의 정확도	40
<그림 21> 최종 정확도 비교	41
<그림 22> 분류하지 못한 700문장 분석 결과	42
<그림 23> TRUE POSITIVE 문장의 분류 근거 시각화	44
<그림 24> TRUE NEGATIVE 문장의 분류 근거 시각화	45
<그림 25> FALSE NEGATIVE 문장의 분류 근거 시각화	45
<그림 26> FALSE POSITIVE 문장의 분류 근거 시각화	46

표 목 차

<표 1> 데이터 셋 예시(2 LABELS)	27
<표 2> 데이터 셋 예시(3 LABELS)	27
<표 3> 형태소 분석 전 문장	28
<표 4> 형태소 분석 전 문장	28
<표 5> 문장행렬 예시.....	30
<표 6> 시그모이드 함수와 소프트맥스 함수의 비교	33
<표 7> 혼동행렬(CONFUSION MATRIX)	37
<표 8> 파라미터에 따른 CNN모델의 정확도 비교.....	38
<표 9> 파라미터에 따른 CNN-LSTM모델의 정확도 비교	39
<표 10> 최종 정확도 비교	41
<표 11> 모델에 부적합한 문장 예시	43

1. 서론

1.1. 연구 배경

최근 인터넷과 이동통신이 발달함에 따라 수많은 사람들이 시간과 장소에 구애 받지 않고 자신의 의견을 남길 수 있고, 다른 이용자가 쉽게 접근 할 수 있게 되었다. 그 중에서도 가장 쉽게 접할 수 있는 것이 상품에 대한 평가이다. 사용자가 작성한 평가는 하나의 상품에 대해 실제 사용자의 좋고 나쁨에 대한 감정을 표현한 결과가 되고, 긍정 또는 부정적인 의견으로 나뉘게 된다. 다양한 소셜 플랫폼에 올라오는 사용자의 상품 리뷰는 잠재적 구매자들에게 유용한 정보로 활용될 수 있으며, 각종 피드백 데이터를 수집하여 체계적으로 수치화하는 것은 마케팅적 관점에서도 중요한 성공 요인이다. 사용자들이 자유롭게 남긴 정형화 되어있지 않은 평가 데이터 중에서 원하는 정보를 찾아내는 기법을 텍스트 마이닝 이라고 하며, 이 데이터를 이용해 사용자 감성의 극성을 판단하는 기법을 오피니언 마이닝 혹은 감성 분석이라고 한다. 감성 분석 연구는 2000년대 이후 급증하였는데 이는 웹의 발전으로 인한 방대한 데이터의 이용 가능성, 자연어처리 및 정보 검색에 사용한 기계 학습의 증가에 있다. 이뿐만 아니라 기업의 규모가 커지고 시장 규모가 방대해질수록 소비자의 의견 각각을 추적하는 것 자체가 매우 어렵다는 문제가 있다. 이러한 방대한 작업을 처리하기 위해 수많은 기업이 소셜 데이터를 자동으로 처리하는 감성 분석 기술에 주목하며 효과적인 분석 솔루션을 차례로 제시하고 있다.

식품회사 마즈는 소셜미디어에서 사람들이 생성하는 문자 코멘트를 실시간으로 분석해 사람들의 감성을 파악하고, 그 결과를 스니커즈의 가격에 실시간으로 반영하는 캠페인을 실시하였다. 이 결과 캠페인 기간 중 매출이 67%이상 성장하였고, 브랜드 관련 언급량이나 웹사이트 방문 고객 수 등은 1000%이상 증가하는 성과를 거두었다. 어도비 시스템즈는 예측 퍼블리싱 기능을 발표한 바 있다. 어도비의 예측 퍼블리싱 솔루션은 페이스북과 같은 소셜 네트워크에서 사용자의 행동과 반응을 분석하여, 기업 콘텐츠에 대한 참여를 예측해 참여도를 높일 수 있는 최적의 타이밍을 제시하는 것이다.

Sk플래닛의 경우 2007년 부터 oms(opinion mining system)개발을 시작으로 2011년부터는 한국어와 영어 자연어 처리 기술을 포함해 oms를 구성하는 핵심기반 기술을 전면적으로 개선시키는 등 다양한 기술을 개발하고 있다.

초기 감성 분석은 긍정 및 부정을 내포하는 단어를 구축한 사전을 이용해 감성을 판별하였다. 대표적으로 영어 기반 감성 사전인 SentiWordNet 이 존재하는데, 전체 어휘에 대한 긍정 및 부정의 감성 강도를 이용해 다양한 분야에서 활용하였다. 또한 다양한 기계 학습의 발전으로 이를 이용해 감성 분류기를 구축한 연구가 이루어졌으며 최근에는 딥러닝(deeplearning) 기법이 발달함에 따라서 신경망 알고리즘을 이용한 연구도 활발히 진행 되어지고 있다. 대표적으로 순서가 있는 정보를 처리하는 순환신경망모델과, 특징 추출에 효과적이고 이미지 및 영상 인식에서 많이 쓰이는 합성곱 신경망을 활용한 텍스트의 감성 분류를 하는 연구가 활발하게 진행되고 있다.

1.2. 연구 목적

감성분석이 소비자 모니터링 과정에서 빠른 대응을 가능하게 하는 전략적인 툴이 되고 있고, 이에 따른 기업들의 연구, 개발이 활발하게 이루어지고 있다. 본 논문에서는 이러한 연구 배경을 바탕으로, 비정형 데이터에 대한 감성분석 연구에 보탬이 되고자 딥러닝 기법 중 CNN과 LSTM을 이용하여 한국어 감성분석 모델을 연구하였다. 우선 문장의 지역적 특성을 추출하는 CNN을 이용하여 모델을 구현하고, 순차적인 입력을 받는 LSTM layer를 추가하여 문장 표상을 얻고자 하였다. CNN과 LSTM을 결합한 C-LSTM 모델은 문장 성분의 지역적 특징을 잡아낼 뿐 아니라 전역적이고 연속적인 문장의 감성을 탐지 할 수 있다.

본 논문에서 제안하는 감성분석 모델은 모두 형태소 분석된 문장을 입력으로 받는다. 모델의 학습과 평가에 사용된 데이터는 네이버 영화 리뷰 데이터셋을 사용하였는데, 기존의 연구에서는 긍정과 부정으로 분류하는 이진 분류를 수행하였으나 본 연구에서는 중립 문장을 추가적으로 수집하여 데이터 셋을 구축하고 긍정, 부정, 중립으로 분류 할 수 있도록 학습하였다. 또한 결과 분석을 통하여 모델의 분류 근거를 예측하고, 사용한 데이터의 문제점과 정확도를 높일 수 있는 방안을 모색하였다.

본 논문의 2장에서는 감성 분석의 개념을 정의하고 단어 임베딩 기술, 순환신경망과 합성곱 신경망에 대한 설명과 함께 관련 연구에 대해 기술한다.

3장에서 실험환경과 실험에 사용한 데이터 셋을 서술하고, 한국어 문장에 대한 감성 분석을 위한 CNN모델의 구조와 작동 원리를 살펴본다. 또한 합성곱 신경망과 순환신경망 모델 중 하나인 LSTM을 조합하여 감성분석을 수행하는 방법을 제안한다. 문장에서 지역적인 특징만을 추출하는 CNN에, 데이터를 순차적으로 처리하여 문장에서 고유한 순차적인 성격을 포착할 수 있는 LSTM 레이어를 연결하였다.

4장에서는 모델의 성능을 최적화 하는 하이퍼 파라미터 값을 찾고, 모델이 분류하지 못한 문장들에 대해서 분석한다.

5장에서는 본 연구에 대한 결론과 향후 연구계획을 제시하고, 마지막으로 6장은 참고문헌을 기술하였다.

2. 관련연구

감성 분석 방법은 크게 두 가지 방법이 주를 이룬다. 하나는 감성사전을 이용하는 방법이고 다른 하나는 사전에 라벨링이 된 긍정, 부정의 텍스트 데이터를 학습하여 분류 모델을 만들고 긍정, 부정을 구분하는 방법이다.

2.1.1 감성사전을 활용한 감성 분석

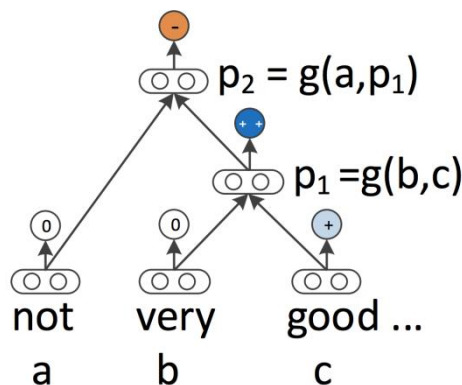
사전 기반 감성분석은 긍정 또는 부정을 의미하는 단어를 미리 구축한 사전을 통해 감성을 판별하는 분석으로, 영어의 경우 대표적인 감성사전 SentWordNet(SWN)이 존재한다. 이는 전체 어휘에 대한 긍정 또는 부정 강도를 저장한 것으로 감성분석의 핵심 중 하나이다. 대부분 형용사 위주로 감성 강도를 측정하거나 문장 구조 및 어휘 품사 등을 고려한 자연어처리 기법, TF-IDF, 통계기법을 이용해 구축하였다. 대표적인 비지도 학습 기반 감성분석으로 Turney[15]는 다양한 분야의 후기를 이용해 감성분석을 시도하였다. POS tagging을 이용해 문장안의 연속한 두 단어 쌍을 추출하는데, 하나는 후기의 성향을 나타내는 형용사나 부사이며 다른 하나는 문맥을 포함한 단어이다. 추출한 단어 쌍을 두 단어의 의미론적 연관성의 강도를 측정할 때 사용하는 PMI-IR 알고리즘을 이용하여 긍정 및 부정에 대한 Semantic Orientation(이하 SO)값을 구한다. SO값의 평균을 이용하여 후기의 감성을 판별하였다.

한국어 감성분석 연구로는 주로 형태소 단위로 분할한 말뭉치를 사용하였다.[16][17] 긍정과 부정을 나타내는 동사, 형용사, 부사를 분류하여 감성사전을 구축하였고 이를 이용해 감성분류기를 학습하였다[17]. 실험 결과로는 분류기 성능이 뛰어났지만, 사전 구축 시 사용하는 오픈소스 형태소분석기 성능이 실험 결과에 영향을 미쳤다.

2.1.2 신경망을 활용한 감성 분석

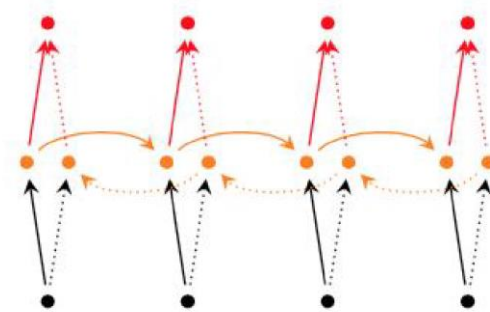
학습을 통해서 감성 분류를 하는 방법은 긍정과 부정으로 분류된 데이터 셋을 사용 하여 분류 모델을 구축하고 분류 모델을 사용하여 문서의 긍정, 부정을 분류하는 방법이다. 이러한 모델 학습은 각 문서에서 특정한 변수들을 추출하거나 Term Frequency-Inverse Document Frequency (TF-IDF), word2vec 등의 방식으로 문서를 표현하고 표현된 문서를 머신러닝, 딥러닝으로 분류하는 방법이다. 기존의 학습을 통한 감성분석에는 네이브베이와 로지스틱회귀 등의 전통적인 기계학습 방법이 많이 사용되었으나, 최근에는 머신러닝 기법 중 신경망이 발전한 형태인 딥러닝을 이용한 감성분석 연구가 이루어졌다. 딥러닝은 인공신경망이 깊은 구조를 가질 때 발생하는 기울기 사라짐 현상, 학습속도의 저하, 과적합 문제 등의 다양한 문제들을 보다 잘 극복 한 일련의 인공신경망 기법들을 총칭하는 개념이다. 딥러닝 기법을 이용한 감성분석 연구에는 다음과 같은 것들이 있다.

Socher[3]은 Recurrent neural network와 함께 Recursive neural network을 기반으로 한 Recursive Neural Tensor Network 모델을 구축하였다. Recursive neural network는 <그림 1>과 같이 입력 값으로 주어지는 몇 개의 단어를 묶어서 분석하는 트리구조를 지닌다. 아래의 그림에서 b와 c는 자식노드(child node)에 해당하는 단어벡터로 1차적으로 부모노드(parent node) p1으로 합쳐지고, 이후 a와 함께 p2를 만들어낸다. G는 두 단어벡터간 연산을 의미하는 기호로 스코어와 벡터를 반환하는데, 감성분석의 경우 스코어는 문장의 긍정 또는 부정을 예측하는데 이용한다. 하지만 학습해야할 가중치와 편향의 수가 많으며 단어사전의 크기에 의존하는 문제점이 있다. 이에 모든 노드에 대해 동일한 텐서 기반 합성 함수를 사용한 Recurrent neural tensor network을 구축하여 감성분석을 시도하였다.



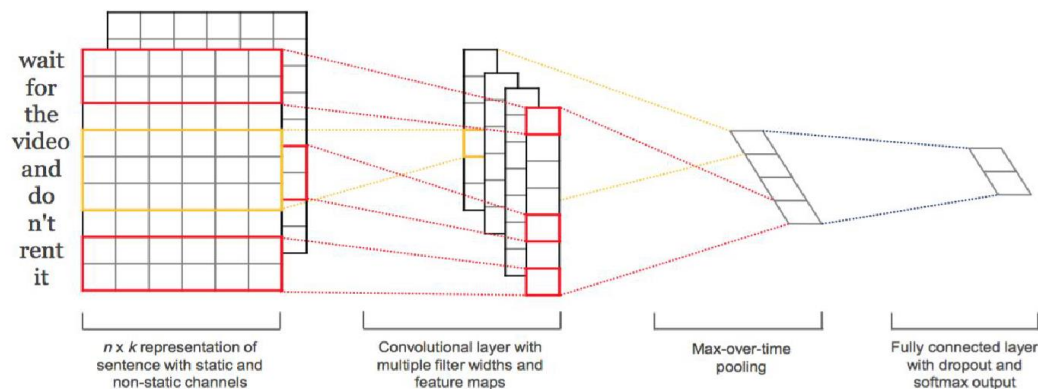
<그림 1> Recursive Neural Tensor Network 모형

Irsoy와 Cardie [4]는 감성분석의 한 종류로 문장에 포함된 의견을 탐지하는 실험에 순환신경망을 이용하였다. <그림 2>와 같이 양 방향으로 이전 시점 정보가 현 시점 정보에 영향을 주는 구조를 갖는 bidirectional RNN[5]을 여러 층으로 쌓은 모형을 구축하였다. Recurrent neural network와 Bidirectional RNN을 다층으로 구축한 모형 성능을 비교한 결과, Bidirectional RNN으로 구축한 모형 성능이 더 뛰어났다.



<그림 2> Bidirectional RNN 모형

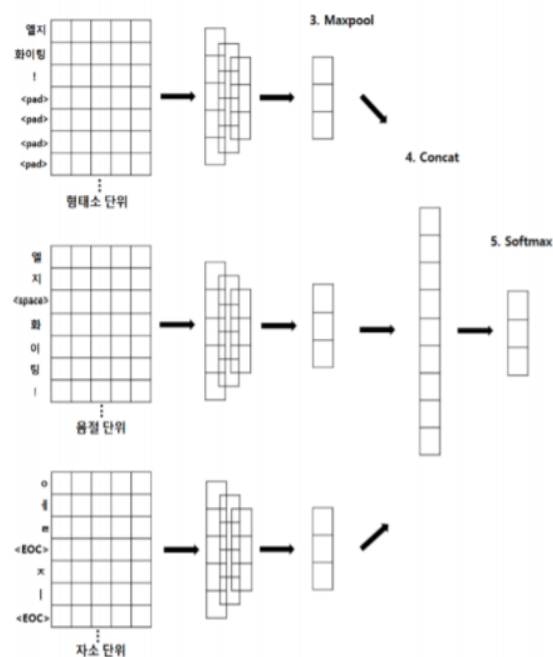
이미지 인식 및 분류에 뛰어난 성능을 보이는 Convolutional Neural Network(CNN)도 감성분석에 이용하였다[1]. CNN은 여러 개 층에서 일반적 행렬 곱셈 대신 convolution을 사용하는 신경망이다[6]. 위치에 상관없이 시각적 특징을 추출 하기 위해 제안되었으며 지역적 특징을 적용한 필터를 사용한다.



<그림 3> Convolutional Neural Network 모형

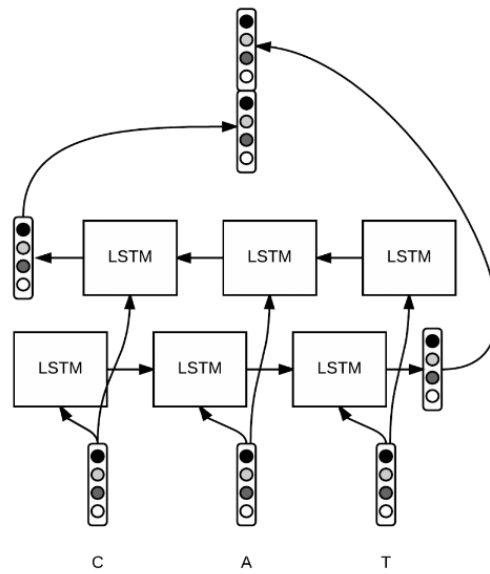
kim yoon의 textCNN 연구를 기점으로 텍스트에 CNN을 활용한 연구가 활발히 진행되었다[1]. kim yoon의 textCNN 모델은 문장을 단어기반으로 분류하여 하나의 합성곱층과 하나의 완전연결층으로 구성되었는데, 여기서 레이어들을 추가하여 더 깊은 깊이를 가진 CNN 모델들이 등장하였다. 이때 여러 개의 채널이 CNN으로 입력되도록 하는 Multi-channel CNN에 대한 연구가 이루어졌다. Multi-channel CNN을 이용한 감성 분석 연구로는 [9,10,11]이 있다. [9]의 연구의 경우에는 영어 구어체 문장을 분류 할 시에 단어와 글자(character)를 동시에 사용하는 Multi-channel CNN 이 단어 기반 CNN 혹은 글자 기반 CNN 보다 뛰어난 성능을 확인하였다. [10]에서는 감성분류에서 Word2vec, Glove, Syntactic 등의 word embedding 을 Multi-channel CNN 을 통하여 동시에 사용하였고 이는 하나의 word embedding 을 사용했을 때보다 향상된 성능을 보였다.

또한 [11]에서는 합성곱층 6개와 완전연결층 3개로 구성된 CNN모델에 한국어 문장의 자소와 형태소를 동시에 사용하여 영화 리뷰의 감성을 분류하였다. 그러나 [10]에서는 문장의 음절을 고려하지 않았고 온라인 댓글 중에서 비교적 정형화가 되어 있고 문장의 길이가 긴 편인 네이버 영화 리뷰 데이터에서만 실험을 하였다. 이에 이어 Min Kim[7]은 한국어 문장을 형태소, 음절, 자소로 나누어 여러 개의 채널을 가지는 입력으로 받아 문장의 감성을 분류하는 Multi-channel CNN모델을 구현하였다. 기존의 CNN 모델보다 약 1.5퍼센트 정도 향상된 성능을 가졌다.



<그림 4> Multi-channel CNN 모형

또한 Skip-Connected LSTM을 이용한 연구도 진행되었다[8]. 이것은 CNN을 이용한 이미지 인식에서 네트워크의 깊이가 깊어질수록 학습 및 평가 에러가 줄지 않는 현상을 해결한 기술로 몇 단계 이전 은닉 계층(hidden layer)의 입력을 가중치 없이 현재의 입력에 더해주는 기술인 Residual network를 RNN에 적용한 모델이다. 정확도는 LSTM모델에 비해 2.37퍼센트 정도 향상되었으나 CNN 모델의 성능을 뛰어넘지 못하였다.



<그림 5> Skip-Connected LSTM 모형

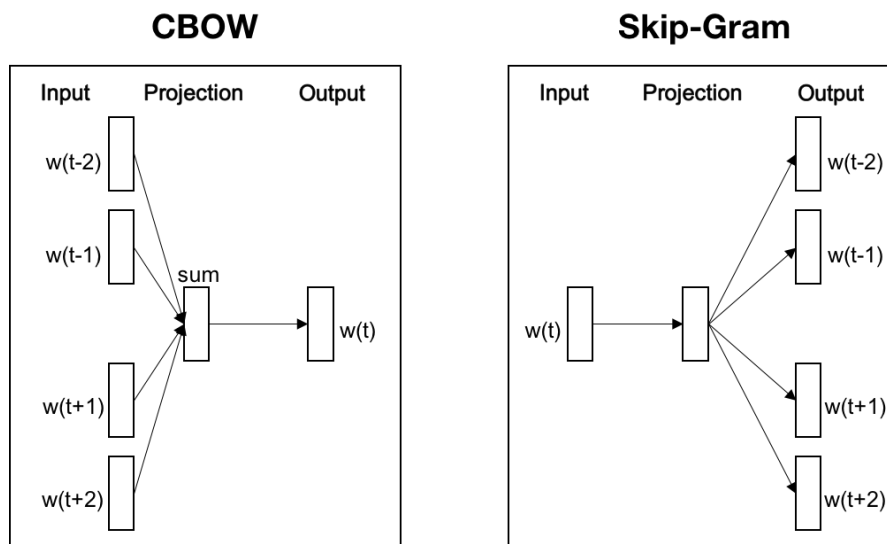
2.2. 감성분석에 사용되는 모델

이 절에서는 딥러닝에 기반한 감성분석 연구에 주로 쓰이는 딥러닝 기법에 대해서 자세히 알아본다.

2.2.1 벡터 할당

단어 벡터화라고도 하는 단어 임베딩은 자연어처리(NLP)에서 중요한 요소이다. 인공 신경망은 숫자로 이루어진 입력 값을 사용하므로 딥러닝 기반의 감성분석을 수행하기 위해서는 텍스트 데이터의 구성요소들을 숫자로 표현하는 임베딩 과정이 선행되어야 한다.

Word2vec은 단어 임베딩 방법론의 하나로, 텍스트를 처리하는 두 개의 층으로 된 신경망이다. Word2vec은 단어를 벡터화 할 때 단어의 문맥적 의미를 보존하기 때문에 위의 그림처럼 ‘엄마’와 ‘아빠’ 사이의 거리는 ‘여자’와 ‘남자’ 거리와 유사하게 된다. 벡터로 바뀐 단어들은 코사인 유사도 같은 방식으로 그 거리를 잴 수 있고 그 거리가 가까울 경우 의미가 비슷한 단어라고 해석할 수 있게 된다. Word2vec은 <그림 6> 과 같이 두가지 구조로 제안되는데 첫 번째 구조는 특정 단어를 그 단어 전후에 나타나는 단어들을 바탕으로 예측하는 CBOW이며, 두 번째 구조는 CBOW와는 반대로 특정 단어를 입력으로 하여 그 단어 전후의 단어들을 예측하는 skip-gram 이다.



<그림 6> Word2vec 구조

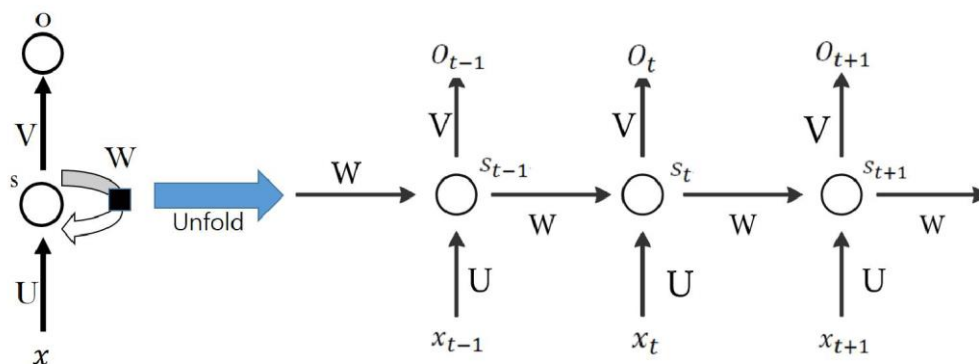
2.2.1. RNN의 기본 구조

RNN(Recurrent Neural Network)은 문자와 음성과 같은 연속된 정보를 처리하는데 적합한 신경망 모형이다[6]. 그림 7은 RNN의 기본 구조를 나타낸다. 기존 인공신경망(Artificial Neural Network)의 각 입력 값들은 독립적으로 사용한다. 그러나 문장 또는 음성의 경우 t 시점의 입력 값은 $t+1$ 시점 입력 값에 영향을 준다. 이러한 특징을 고려하여 순환신경망의 은닉계층(hidden layer)은 각 시점의 입력 값이 서로 영향을 주는 재귀적인 구조로 이루어졌다. 은닉계층 및 출력 계층 노드 값은 아래와 같은 방법으로 계산한다.

$$s_t = f(Ux_t + Wh_{t-1} + b) \quad (1-1)$$

$$o_t = g(Vs_t + c) \quad (1-2)$$

은닉계층의 노드 값 h_t 은 t 시점의 입력 값 x_t 과 이전 시점의 은닉계층 노드 값 h_{t-1} 의 선형 결합을 활성화 함수 f 를 이용하여 표현한다. 활성화 함수로는 sigmoid, tanh, ReLU(Rectified Linear Unit)와 같이 비선형 함수를 사용한다. 이후 출력계층의 노드 값 O_t 은 softmax 또는 sigmoid와 같은 활성화 함수(g)를 이용하여 0과 1사이의 확률 값으로 변환시켜 예측한다. 이는 문장 생성 실험의 경우 다음 출현할 단어를 예측하고, 감성분석[3]의 경우 긍정 또는 부정을 예측한다. 식에서 사용한 U, W, V 는 가중치를 나타내는 행렬이며 b, c 는 선형 결합에 사용한 편향을 의미한다.



<그림 7> RNN의 기본 구조

모형의 출력 값은 무한한 과거 시점의 출력에 의존(long-term dependency)하는 것을 확인할 수 있다. 이는 역전파 알고리즘을 이용하여 파라미터를 학습할 때, 목적함수의 편미분 계수가 0에 가까워지거나(vanishing gradient) 매우 커지는 문제가 발생한다[12]. 편미분 계수가 0에 가까워지면 파라미터 학습이 제대로 이루어지지 않으며, 매우 커지게 되면 학습이 불안정해진다.

2.2.2 LSTM의 기본구조

순환 신경망(RNN)은 임의의 길이의 시퀀스 정보를 처리할 수 있지만 실제로는 비교적 짧은 시퀀스만 효과적으로 처리할 수 있다. 이 문제를 장기 의존성(Vanishing gradient) 문제라고 하는데 이와 같은 문제를 해결해 주기 위해 나온 알고리즘이 순환 신경망의 변형인 LSTM(Long Short Term Memory) 이다.

LSTM은 장기 의존성 문제를 해결하기 위하여 3가지 게이트와 노드에 분리된 메모리 공간을 설치하였다. 입력 게이트(input gate) i_t , forget gate f_t , memory cell C_t (output of the hidden layer)는 hidden layer의 출력 값인 h_t 를 다음의 식으로 계산한다. (σ 는 시그모이드 함수)

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \quad (2-1)$$

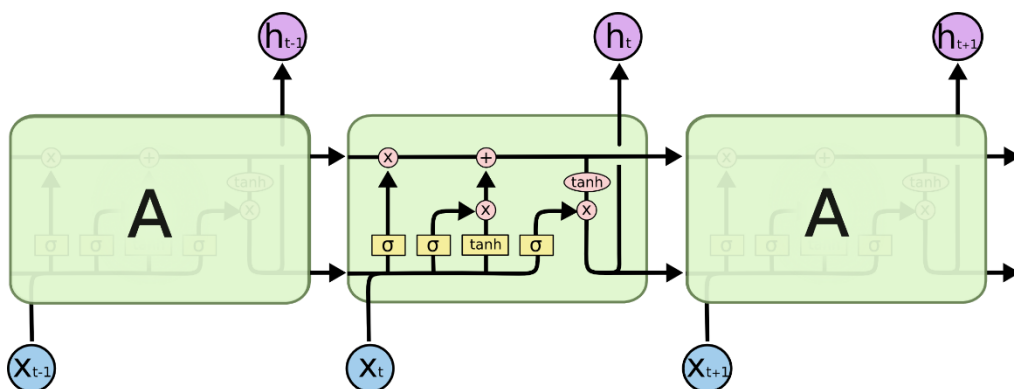
$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \quad (2-2)$$

$$\tilde{C}_t = \tanh \tanh (W_q \cdot [h_{t-1}, x_t] + b_q) \quad (2-3)$$

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \quad (2-4)$$

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \quad (2-5)$$

$$h_t = o_t * \tanh(C_t) \quad (2-6)$$



<그림 8> LSTM 셀 구조

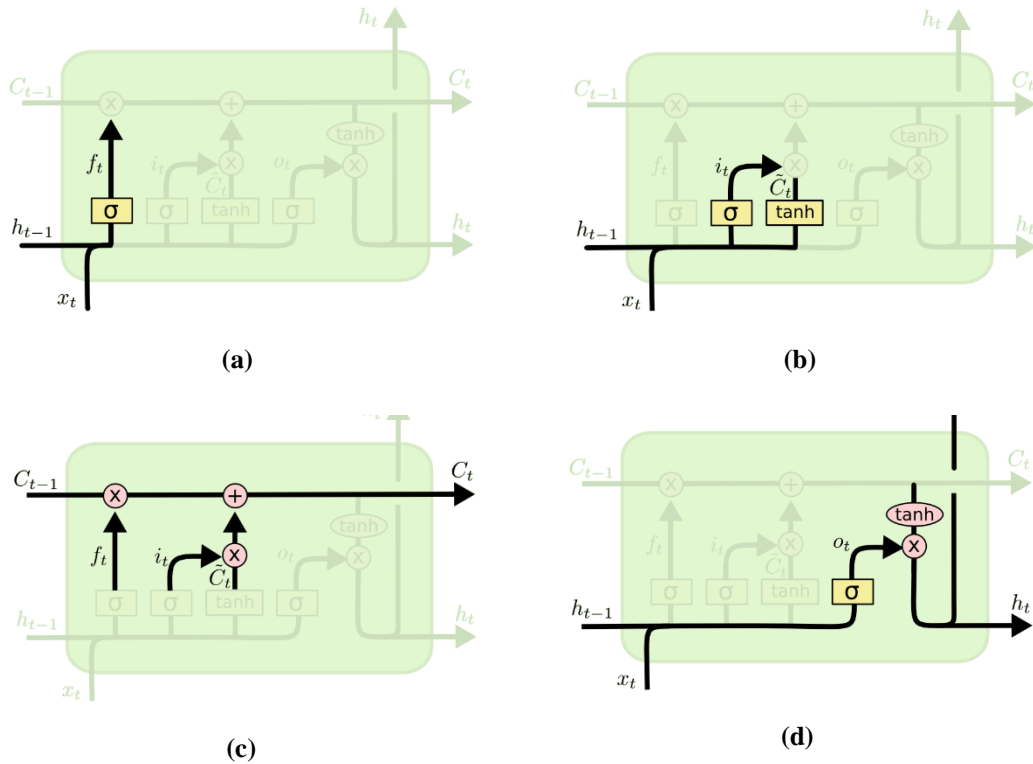
<그림 8>은 LSTM 네트워크의 전체적인 구조를 도식화 한 그림이다. 이를 구체적으로 살펴보면 다음의 네 단계로 구성된다.

LSTM의 첫 단계<그림 9-(a)>로는 cell state로부터 어떤 정보를 버릴 것인지를 정하는 것으로, sigmoid layer에 의해 결정된다. 그래서 이 단계의 gate를 "forget gate layer"라고 부른다. 이 단계에서는 h_{t-1} 와 x_t 를 받아서 0과 1 사이의 값을 C_{t-1} 에 보내준다. 그 값이 1이면 모든 정보를 저장하고, 0이면 아무것도 저장하지 않음을 의미한다.

다음 단계는<그림 9-(b)> 앞으로 들어오는 새로운 정보 중 어떤 것을 cell state에 저장할 것인지를 정한다. 이번 스텝에도 입력 값은 h_{t-1} 와 x_t 이다. 먼저 Input gate layer인 i_t 에 sigmoid layer를 적용하여 cell state의 어떤 부분을 업데이트 할지 정한다. 그 다음에 tanh layer를 통해서 새로운 후보 값들인 \tilde{C}_t 라는 vector가 생성된다. 다음 단계에서 이 두 개가 현재의 cell state인 C_t 를 업데이트 하기 위해 결합된다.

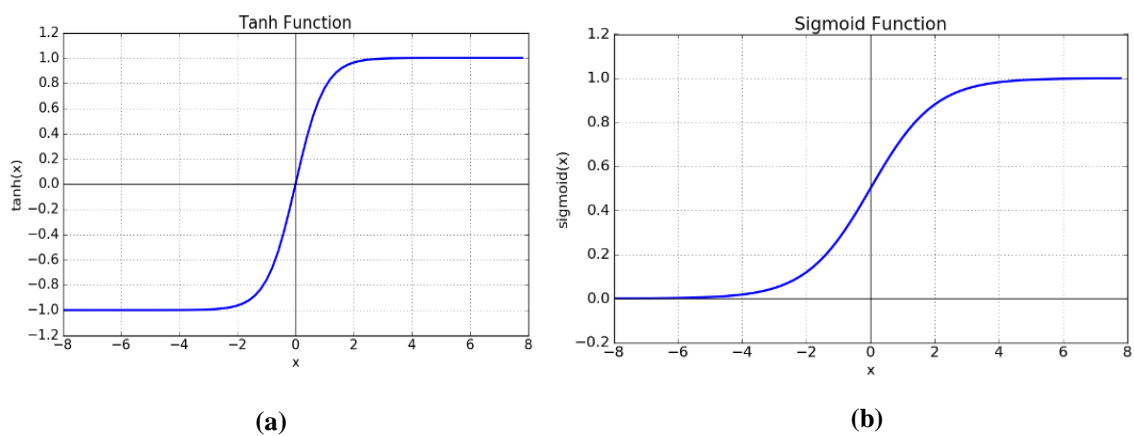
이 단계에서는<그림 9-(c)> 과거 state인 C_{t-1} 를 업데이트 해서 새로운 cell state인 C_t 를 만든다. 이전 state에 계산해 놔던 f_t 를 곱해서 가장 첫 단계에서 잊어버리기로 정했던 것을 진짜로 잊어버린다. 후에 $i_t * \tilde{C}_t$ 를 더한다. 수식 (2-5)가 더하는 값은 두 번째 단계에서 업데이트 하기로 한 값(\tilde{C}_t)을 얼마나 업데이트 할 지 정한만큼 scale한 값이 된다.

마지막으로<그림 9-(d)> 무엇을 output으로 내보낼 지 정하는 일이 남았다. 이 output은 cell state를 바탕으로 필터 된 값이 될 것이다. 가장 먼저, sigmoid layer에 input 데이터를 태워서 cell state의 어느 부분을 output으로 내보낼 지를 정한다. 그리고 나서 cell state를 tanh layer에 태워서 -1과 1 사이의 값을 받은 뒤에 방금 전에 계산한 sigmoid gate의 output과 곱해준다. 그렇게 하면 output으로 보내고자 하는 부분만 내보낼 수 있게 된다.



<그림 9> LSTM 동작 예시

Tanh는 bipolar 형태의 sigmoid로, 이처럼 RNN과 LSTM에서 중간 활성화 함수로 ReLU나 logistic Sigmoid를 쓰지 않고 tanh 함수를 사용하는 것은 tanh는 출력값이 -1~1로 평균이 대략 0이 되는 데에 반해, sigmoid는 0~1로 평균값이 0.5에 근접하게 된다. RNN과 LSTM같이 반복적인, recurrent 한 형태를 가지는 구조에서는 평균적으로 0이 아닌 0.5의 아웃풋 값을 내는 것이 반복적으로 지속되면 편향 문제가 발생하기 때문이다.



<그림 10> Tanh 함수(a), Sigmoid 함수(b)

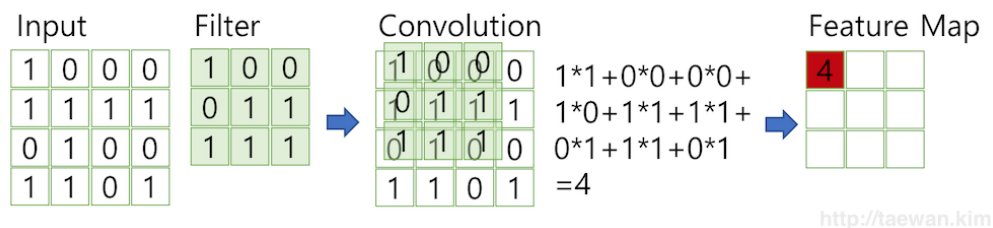
LSTM은 장기 의존성 문제를 해결하기 위하여 3가지 게이트와 노드 에 분리된 메모리 공간을 설치하였다. 데이터를 계산하는 각 길목에 입력 게이트, 망각 게이트, 출력 게이트를 설치하여 각 상태 값을 메모리 공간 셀에 저장하고 데이터를 접하는 부분을 조정하여 불필요한 연산, 오차 등을 줄여 장기 의존성 문제를 해결하였다

기본적인 아이디어는 **RNN**과 같지만 3가지 게이트가 추가된 **LSTM**은 수식 (2-3) ~ (2-6) 과 같은 게이트들의 연산으로 인해 상당히 복잡한 수식을 나 타내고 있다. **i**, **f**, **o**는 각 게이트를 조정하는 값들로 문장에서 새로 들어 온 단어와 입력 가중치를 곱하여 이전 단어에 계산된 값을 더하여 결정 한다. **g**는 활성화함수를 사용하여 새로 들어온 단어와 이전에 계산된 단어 의 기울기를 계산하고 그 차이를 다음 단어가 계산할 때 영향을 미치게 되어있다

2.2.3. CNN의 기본 구조

합성곱신경망(Convolutional Neural Networks, CNN)은 이미지 처리를 위해 고안된 특수한 연결 구조를 가진 다층신경망이다. 합성곱 층과 풀링층, 완전연결층으로 구성되는데, 대표적인 CNN으로는 LesNet[13]과 AlexNet[14]이 있다. CNN의 핵심은 이미지를 학습한다는 것이다. 입력으로 라벨이 붙은 이미지 파일을 주고 수많은 이미지를 학습시켜 추후에 새로운 이미지가 입력되었을 때 정확히 라벨을 붙이는 것을 목적으로 한다.

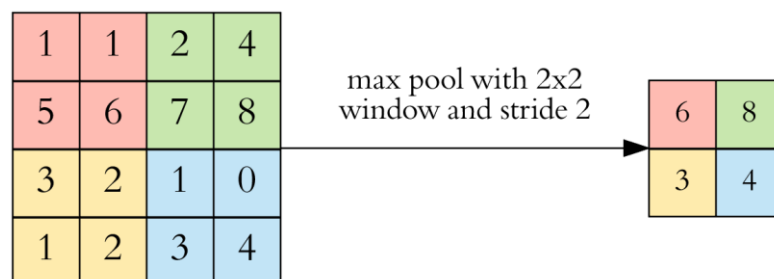
CNN은 인간이 시각 정보를 처리하는 방식을 그대로 이용하고 있다. 우리가 무언가 사물을 볼 때 우리는 뇌는 사물을 부분적으로 인식하여 처리한 후 통합하여 하나의 이미지를 만들어낸다. CNN에서 역시 동일하다. 이미지 행렬을 한쪽 구석에서부터 읽어나가 반대쪽 구석까지 차례대로 작은 이미지들로 읽어낸 후 각각을 처리하여 마지막에 통합하는 과정을 거친다. CNN은 학습 가능한 가중치(weight)와 바이어스(bias)로 구성 되어있다. 전체 네트워크는 일반 신경망과 마찬가지로 미분 가능한 하나의 스코어 함수(score function)을 갖게 된다. 스코어 함수는 맨 앞쪽에서 원본 이미지(raw image)를 읽고 맨 뒤쪽에서 각 클래스에 대한 점수를 구하게 해준다. 그리고 마지막 레이어에 손실함수(loss function)을 가지며, 일반 신경망을 학습 시킬 때 사용하던 각종 기법들을 동일하게 적용 할 수 있다. CNN의 장점은 필터(filter)에 있다. 합성곱 계층에는 특징을 추출하는 필터가 존재하게 되는데, 제일 먼저 필터를 사용하여 간단한 특징들을 뽑아내 하나의 합성곱 레이어를 만들게 되고, 이 특징들에서 조금 더 복잡한 특징을 추출하는 레이어를 추가하게 된다. 이렇게 여러 개의 합성곱 레이어를 연결하여 고차원적인 특징을 뽑아낸 후 학습을 진행하게 된다. 이때 CNN은 필터를 학습을 통해 스스로 생성한다. 이러한 특징은 포워드 함수(forward function)을 더욱 효과적으로 구현 할 수 있고 네트워크를 학습시키는데 필요한 파라미터의 수를 크게 줄일 수 있게 해준다.



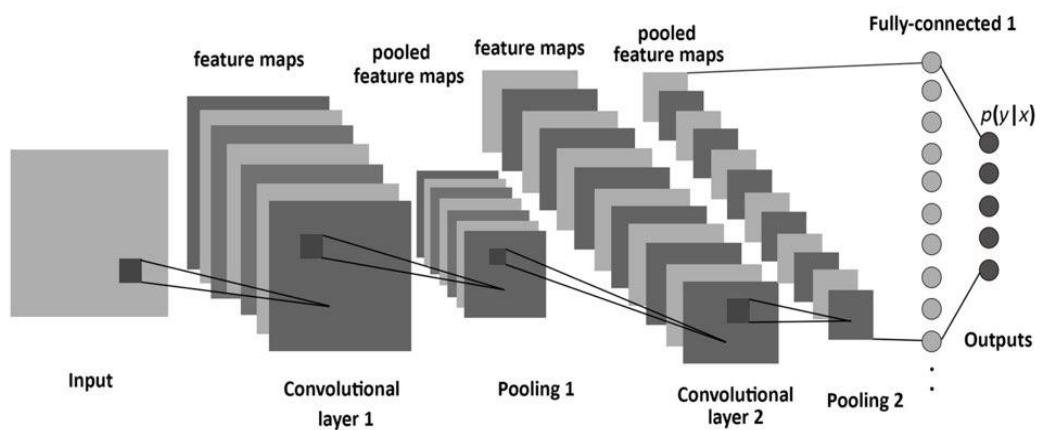
<그림 11> CNN이 이미지에서 합성곱 연산을 수행하는 모습

이 때 오버피팅(overfitting)을 방지하는 방법으로 풀링(pooling)을 수행하는데, 이러한 pooling에는 대표적으로 최댓값을 뽑아내는 max pooling, 평균값을 뽑아내는 mean pooling이 있다. pooling은 matrix 연산을 사용하지 않고 각 pixel에서 하나의 값을 뽑아내는 과정과 같다.

텍스트 처리에 쓰이는 CNN의 필터는 텍스트의 지역적인 정보, 즉 단어의 등장 순서 및 문맥 정보를 보존한다는 개념으로 적용된다. 합성곱 신경망은 문장의 지역 정보를 보존함으로써 단어와 표현의 등장순서를 학습에 반영하는 역할을 한다.



<그림 12> max pooling



<그림 13> CNN 동작 예시

3. 연구 방안

이 장에서는 실험환경과 실험에 쓰인 데이터 셋을 서술한다. 또한 감성 분석을 실시할 데이터의 전처리가 어떻게 되었는지를 설명하고, 본 논문에서 제안하는 CNN과 CNN-LSTM 감성분석 모델의 구조를 알아본다.

3.1. 실험 환경

램 16GB의 CPU의 환경에서 오픈소스 기계학습 신경망 라이브러리인 tensorflow를 사용하여 감성 분류 모델의 개발과 학습을 진행하였다. Python3 언어로 구현하였으며, 한국어 형태소 분석을 위해서는 KoNLpy 패키지의 Twitter 형태소 분석기를 사용하였다. 64개의 문장을 하나의 batch로 하고 학습 최적화 함수로 Adam optimizer method를 사용하였다. Adam method(3-1)는 누적된 그레디언트에 가중 이동 평균 기법을 적용하는 RMSProp(3-3)에 모멘텀(3-2)을 추가로 적용한 알고리즘이다.

$$g = \frac{\partial J}{\partial w} \quad (J = \text{error function}) \quad (3-1)$$

$$v = a_1 v - (1 - a_1)g \quad (3-2)$$

$$r = a_2 r + (1 - a_2)g \odot g \quad (3-3)$$

3.1 실험 데이터

본 연구에서 사용한 데이터는 첫번째로 공개된 영화 리뷰 데이터 중 하나인 NSMC(naver sentiment movie corpus-출처:네이버영화 리뷰 사이트) 데이터 셋을 사용하였다. 데이터셋은 영화당 100개 그리고 140자를 초과하지 않는 범위 내에 별점이 부정(1~4), 긍정(9~10)인 리뷰들로 구성되어있다. 그림 1은 NSMC 데이터셋의 예시이다. 추가적으로 좀더 세분화된 감정을 분류하기 위해, 같은 사이트에서 별점 5~7점의 리뷰문장들을 중립으로 labeling 하여 긍정, 부정, 중립으로 구성된 새로운 데이터 셋을 수집하였다. (그림 2) 데이터 전처리는 python Konlpy의 twitter 형태소 분석기를 사용하여 형태소 분석을 진행하였다.

<표 1> 데이터 셋 예시(2 labels)

corpus	label
아 더빙... 진짜 짜증납니다 목소리	0
흠...포스터 보고 초딩영환줄. 오버연기 조차 가볍지 않구나.	1
한숨 나온다	0
그냥 매번 긴장되고 재밌음 ㅠㅠ	1
주제는 좋은데 중반부터 지루하다.	0
평점에 속지 마시길 시간낭비 돈낭비임	0
아 진짜 너무 좋아요 ㅠㅠ 짱짱!	1

<표 2> 데이터 셋 예시(3 labels)

corpus	label
액션이 최고입니다 강추	1
그냥 그냥 볼 만 한듯	2
한숨 나온다	0
재밌는데 지루하다	2
아버지와 저녁에 한 그게임, 그게 임팩트라고 하기엔 부족하다	2
아이도 부모도 너무 지루한 영화였다	0
이런류의 영화는 기본은 하죠... 음악이 좀더 좋았으면	2

3.2. 형태소 기반의 감성분석 모델

3.2.1. 한국어 형태소 분석

형태소란 언어에 있어서 최소 의미 단위를 말한다. 이때 의미는 어휘적 의미와 문법적 의미를 모두 포함한다. 형태소 분석이란 형태소 보다 단위가 큰 언어 단위인 어절, 혹은 문장을 최소 의미 단위인 형태소로 분절하는 과정이다.

형태소 분석은 모든 자연언어 처리 분야에서 가장 중요하면서도 기본적으로 필요한 요소이다. 특히 어미와 조사의 사용에 의해 단어의 형태적 변형이 심한 한국어에서는 형태소 처리부의 역할이 중요하다. 형태소 분석은 형태소 분석, 품사 부착, 구절 단위 분석, 구문 분석 등의 분석이 이루어진 후 활용할 수 있다. 형태소 분석은 자연언어 처리에 필요한 정보를 추출하는 처리 시스템의 초석으로써 형태소 분석의 기능과 효율성은 전체 시스템에 바로 직결되는 중요한 요소이다. 본 논문에서는 Konlpy의 twitter 형태소 분석기를 사용하였다. 이때 형태소를 원형으로 변형하여 분석하게끔 하였다.

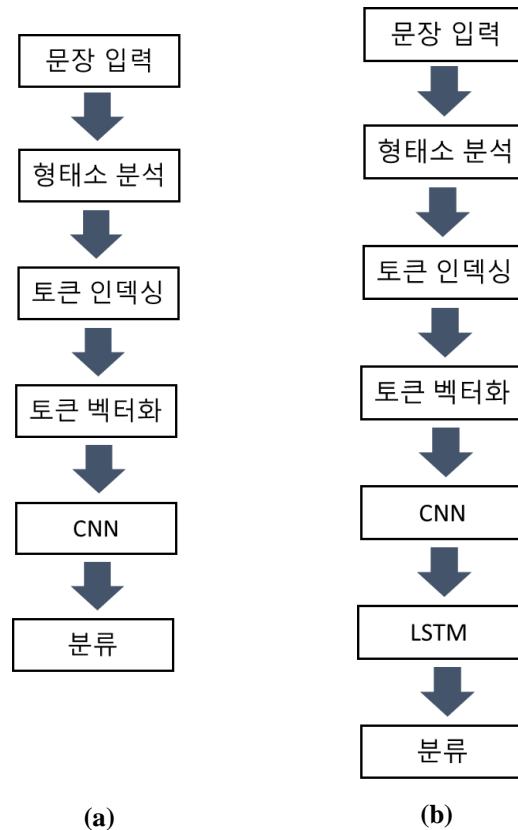
분석된 형태소들이 모델에 입력으로 들어가기 전에, 수치화가 필요하기 때문에 형태소마다 고유한 인덱스를 할당하는 인덱싱 작업을 수행하였다.

<표 3> 형태소 분석 전 문장

아 더빙... 진짜 짜증나네요 목소리	0
너무 재밌었다 그래서 보는걸 추천해요	1

<표 4> 형태소 분석 전 문장

(['아/Exclamation', '더빙/Noun', '.../Punctuation', '진짜/Noun', '짜증/Noun', '나다/Verb', '목소리/Noun'], '0')
(['너무/Noun', '재/Noun', '밋었/Noun', '다/Josa', '그래서/Adverb', '보다/Verb', '추천/Noun', '하다/Verb'], '1')



<그림 14> CNN 모델 흐름도(a), CNN-LSTM 모델 흐름도(b)

<그림 14>은 감성분석 모델이 문장의 감성을 예측, 분류하기까지 거치는 과정을 대략적으로 도식화 한 것이다.


CNN모델은<그림 14 -(a)> 입력문장을 형태소 분석 후 나뉜 토큰들을 인덱싱 시킴으로써 문장을 숫자의 조합으로 수치화 시킨다. 각 문장요소들을 숫자로 표현한 후 각 숫자들에 대해 n 차원 벡터를 할당하는데, 이를 벡터화, 또는 임베딩 라고 한다. 문장요소들의 벡터화로 인하여 문장은 최종적으로 행렬의 형태로 변환되어 CNN 모델에 입력으로 들어가게 된다.

CNN-LSTM 모델<그림 14 -(b)> 은 CNN모델을 거쳐 생성된 각 단어에 대한 특징벡터(feature)를 순차적으로 LSTM셀에 입력으로 받게 된다.

3.2.2. CNN 모델

CNN 모델을 사용하기 위해서는 입력이 이미지, 즉 행렬 형태로 변환되어야 한다. 문장을 구성하는 단어와 대응되는 단어벡터들을 세로로 붙이게 되면 행의 길이가 문장 속 단어의 개수, 열은 하이퍼 파라미터로 지정되는 임베딩 벡터 차원만큼의 길이를 가지는 행렬이 된다. 이 행렬을 문장행렬이라고 지칭한다.

<표 5> 문장행렬 예시

아		0.28002	0.06650	-0.265	-0.08	0.31791
더빙		-0.135	0.52208	0.39680	0.12382	0.55685
진짜		0.04849	0.15080	-0.181	0.38363	-0.135
짜증		0.16284	-0.262	0.25076	-0.221	0.44962
나다		-0.093	0.00611	-0.033	0.56291	0.13293
목소리		0.1144	0.27957	0.193352	0.05686	0.10291

감성 분석을 위한 합성곱신경망 모델의 구조는 다음과 같다. 하나의 합성곱 층(convolution layer)와 하나의 완전연결층 (fully connected layer)을 가진다.

1. embedding layer

문장이 일련의 숫자(인덱싱된 토큰)들로 변환되어 입력으로 들어오면, 각각의 숫자들을 벡터로 표현한다. 그림-임베딩 예시 문장 내의 모든 단어 임베딩 벡터를 합치면 2차원의 문장행렬이 되어 CNN의 입력으로 들어가게 된다.

2. convolution layer

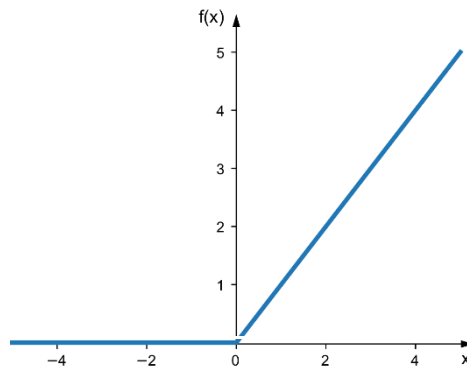
각 필터에 의해 합성곱이 수행되는 단계이다. m개의 필터들이 문장의 h-gram 지역정보를 추출하기 위해 사용된다. (h-gram = h개의 단어를 하나로 보고 특징을 추출-설명이 앞으로) 이러한 특징들을 기반으로 다시 상위 레벨의 자질들을 추출하는 것이 가능하다. 문장행렬에 대한 합성곱 필터 연산은 다음과 같다(4-1).

$$y[i,j] = (x * w)[i,j] = \sum_{n=-\infty}^{\infty} \sum_{m=-\infty}^{\infty} x[m-i, n-j]w[m,n] \quad (4)$$

$x_i \in R^k$ 가 임의의 문장의 i-번째 단어에 대한 k-차원의 단어 벡터라고 할 때, $x_{i:i+j}$ 는 단어 $x_i, x_{i+1}, \dots, x_{i+j}$ 를 연결한 형태를 의미한다. 합성곱 연산은 새로운 특징을 추출하기 위해 h개의 단어 윈도우(h-gram과 같은 의미)에 필터 $W \in R^{h \times k}$ 를 적용한다. 특징 c_i 는 단어 $x_{i:i+j}$ 로부터 다음 연산에 의해서 생성된다.

$c_i = f(w \cdot x_{i:i+h-1} + b)$ (f: 하이퍼볼릭 탄젠트(hyperbolic tangent) 또는 렐루(relu)와 같은 비선형 함수(non-linear function), $b \in R$: 바이어스(bias))

필터는 해당 문장의 모든 가능한 윈도우인 $\{x_{1:h}, x_{2:h+1}, \dots, x_{n-h+1:n}\}$ 에 대해 적용되어 특징맵(feature map)인 $c = [c_1, c_2, \dots, c_{n-h+1}]$ 이 생성된다. 이때 특징맵의 요소들인 컨볼루션 한 결과값에 ReLU 함수를 적용하여 비선형성을 가지도록 했다.



<그림 15> ReLU 함수

여기서 필터의 열 개수는 입력 문장 행렬의 열 개수, 즉 단어 임베딩 차원수와 같게 고정하였는데, 이는 행렬의 모든 요소가 독립적인 픽셀 값으로 이루어져있는 이미지와 달리 임베딩을 통해 차원확장된 문장행렬은 한 행의 모든 요소들이 특정 단어를 수치화 한 값이기 때문에 필터가 합성곱을 수행할 때 행 단위로 수행하여야 한다.

3. max-pooling layer

맥스 풀링 연산은 이러한 특징 맵(feature map)에 대해 다음 식과 같이 전역 최대값을 구한다. $\hat{c} = \max\{c\}$ 윈도우 사이즈가 다른 여러 개의 필터를 적용하면 여러 특성의 자질들을 추출 할 수 있다.

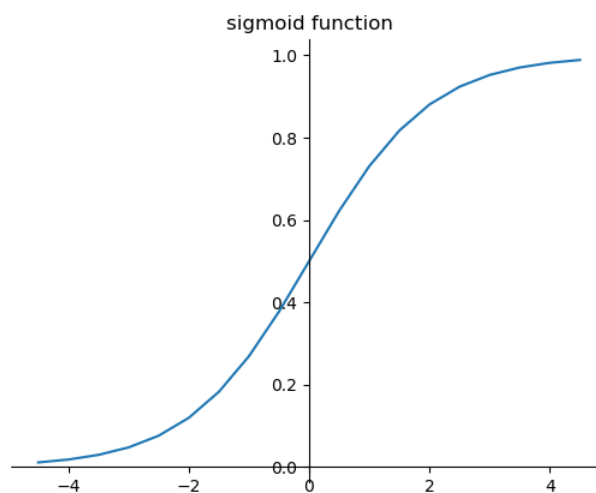
4. output layer

최댓값 풀링한 값들이 합쳐지고 완전연결층을 지나면, 마지막으로 입력 신호의 총합을 출력 신호로 변환하는 활성화 함수를 거친 뒤 최종 분류를 수행하게 된다. 대표적인 활성화 함수는 Sigmoid와 Softmax 함수가 있다.

(1) Sigmoid (연속형 0~1 사이값 출력)

시그모이드 함수는 로지스틱 회귀분석 또는 Neural network의 Binary classification 마지막 레이어의 활성화 함수로 주로 사용된다.

$$y = \frac{1}{1 + e^{-z}} \quad (5)$$



<그림 16> 시그모이드 함수

(2) Softmax

소프트맥스 함수는 출력값이 여러 개로 주어지는데, 다범주 분류를 하는 경우 각 범주에 속할 사후 확률을 제공한다. 소프트맥스는 출력 노드의 중간 계산 결과 최댓값은 더욱 활성화하고 작은 값은 억제하는데, 이로써 출력값을 normalization 한 효과를 얻게 된다. 따라서 본 논문에서는 2진 분류 시에도 소프트 맥스 함수를 사용하였다. 다음 표는 두 활성화 함수를 이용하여 각 클래스에 속할 확률 값을 출력한 예시이다.

$$y_i = \frac{e^{(z_i)}}{\sum_{i=1}^L e^{(z_i)}} , j = 1, \dots, L \tag{6}$$

<표 6> 시그모이드 함수와 소프트맥스 함수의 비교

	sigmoid	softmax
Class 1	0.8808	0.1131
Class 2	0.7685	0.0508
Class 3	0.9820	0.8360

6. back-propagation (역전파 학습)

본 논문에서는 크로스-엔트로피 에러를 최소화 시킴으로써 전체 모델을 학습시켰다. 훈련 데이터가 $x^{(i)}$ 이고, 라벨 값 $y^{(i)} \in \{1, 2, \dots, k\}$ (k 는 가능한 label의 수), 그리고 모델이 측정한 라벨의 가능성 $\tilde{y}_j^{(i)} \in [0, 1]$ (0과 1사이의 값) (각 라벨을 나타내는 $j \in \{1, 2, \dots, k\}$) 일 때 에러는 다음과 같이 정의된다(7).

$$L(x^{(i)}, y^{(i)}) = \sum_{j=1}^k 1_{\{y^{(i)} = j\}} \log(\tilde{y}_j^{(i)}) \quad (7)$$

7. drop out 기법 사용

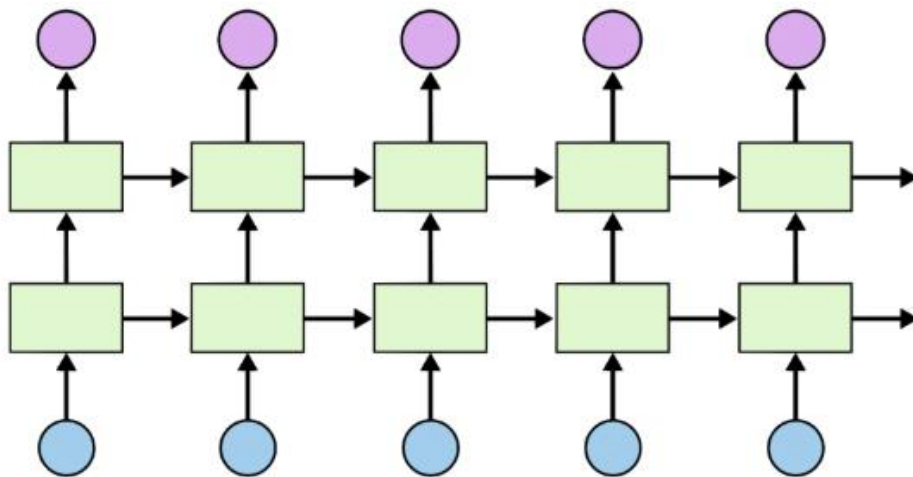
마지막으로 maxpooling layer 뒤에 Dropout layer 를 추가하였다. 드롭 아웃은 오버피팅(over-fit)을 막기 위한 방법으로 뉴럴 네트워크가 학습중일때, 랜덤하게 뉴런을 꺼서 학습을 방해함으로써, 학습이 학습용 데이터에 치우치는 현상을 막아주는 효과가 있다.[5]

3.2.3. CNN-LSTM 합성 모델

<그림 14> 에서 설명한 흐름도를 통해 CNN,LSTM을 이용한 감성 분류기의 구조를 살펴본다.

CNN-LSTM 모델은 기존의 CNN 감성 분석 모델의 output layer 바로 전에 LSTM layer를 추가한 것이다. 그림 x는 CNN-LSTM 모델의 기본적인 구조를 나타낸다. CNN-LSTM 모델이 CNN 모델과 다른 점은 우선 합성곱을 수행하는 convolution layer를 거친 후 최댓값 풀링(max pooling)을 하지 않는 데에 있다. 이는 LSTM이 순서를 가지고 있는 입력을 받기 때문에, 문장을 구성하는 형태소들의 순서를 해야 하는데 최댓값 풀링을 해버리면 순서가 없어져 버리기 때문이다.

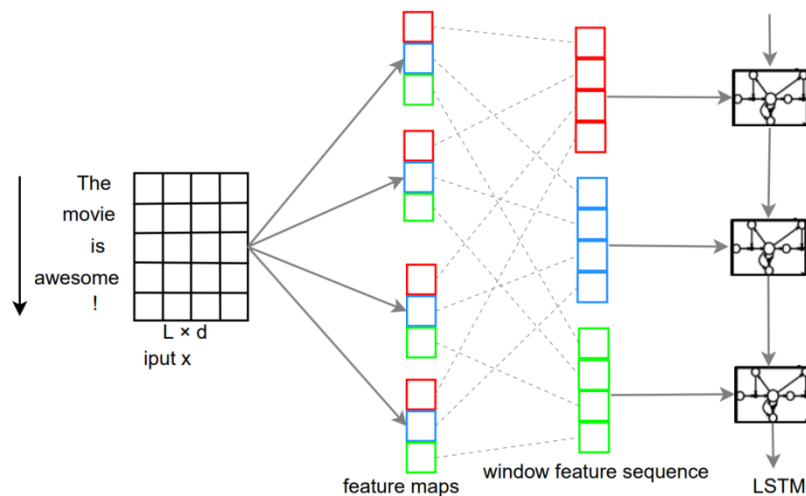
또한 두개의 LSTM layer를 쌓은 형태로 구현하여, 마지막 셀의 출력 값 과 상태 값을 반환하도록 했다. 마지막 셀의 출력 값을 soft max layer에 적용하여 최종적인 감성 분류를 하도록 했다.



<그림 17> LSTM 계층 모습

본 논문에서는 감성 분류를 위해, 문장 전체의 감성 척도를 나타내는 마지막 **hidden state**의 출력만을 취급 하였고 그 위에 **softmax layer**를 추가하였다. 또한 CNN모델과 같이 크로스-엔트로피 에러를 최소화 시킴으로써 전체 모델을 학습시켰다. **Dropout** 기법도 동일하게 적용하였다.

CNN-LSTM 모델의 구조는 그림 몇과 같다. CNN으로부터 높은 수준의 문장 특성에 대한 시퀀스를 추출하고, 윈도우 사이즈, 즉 필터 크기에 국한된 시퀀스에 대하여 장기 의존성 문제를 해결하기 위해 **LSTM** 레이어를 적용하였다. CNN모델과 동일한 방식으로 **feature maps**이 추출되면 이 벡터들의 i 번째 값들을 합쳐서 벡터를 생성하고 **LSTM layer**의 i 번째 입력으로 만든다. 특징맵은 CNN 필터가 문장의 요소들을 순차적으로 합성곱 수행하며 생긴 값 이므로, 결론적으로 하나의 문장 요소에 대한 특징들을 합친 것이 **LSTM layer**가 입력으로 인식하는 i 번째 단어를 나타내는 새로운 벡터인 셈이다.



<그림 18> CNN 계층의 출력 후 LSTM 계층의 입력이 되는 모습

4. 연구 결과 및 분석

4.2. 실험 결과

본 논문에서 사용 되어지는 데이터가 비율이 같다는 점에서, 모델 성능 평가 기준은 혼동행렬 (Confusion matrix)을 사용하였다. 다음 행렬에서 ‘True’ 는 실제와 예측이 일치하는 경우이고, ‘False’ 는 실제와 예측이 불일치 하는 경우를 나타낸다. 정확도는 전체 경우의 수 중에서 True 로 행동한 비율로 측정하였다. 또한 하이퍼 파라미터를 다르게 하면서 나타나는 정확도의 변화를 측정하였다.

<표 7> 혼동행렬(confusion matrix)

레이블(Ground Truth)	모델 분류 결과	
	Positive	Negative
Positive	True Positive	False Negative
Negative	False Postive	True Negative

$$Accuracy(\%) = \frac{TP + TN}{TP + TN + FP + FN} \quad (8)$$

다음은 CNN모델로 하이퍼 파라미터값을 달리하여 각각 긍정, 부정(2 class) 분류와 긍정, 부정, 중립(3 class)분류를 실행한 결과를 표로 나타낸 것이다. 하이퍼 파라미터는 CNN의 필터 크기(filter size), 필터 개수(num_filter), 임베딩 크기(embedding_size)이다. 여기서 임베딩 크기란 단어가 벡터화 될 때의 차원의 크기를 말한다.

<표 8> 파라미터에 따른 CNN모델의 정확도 비교

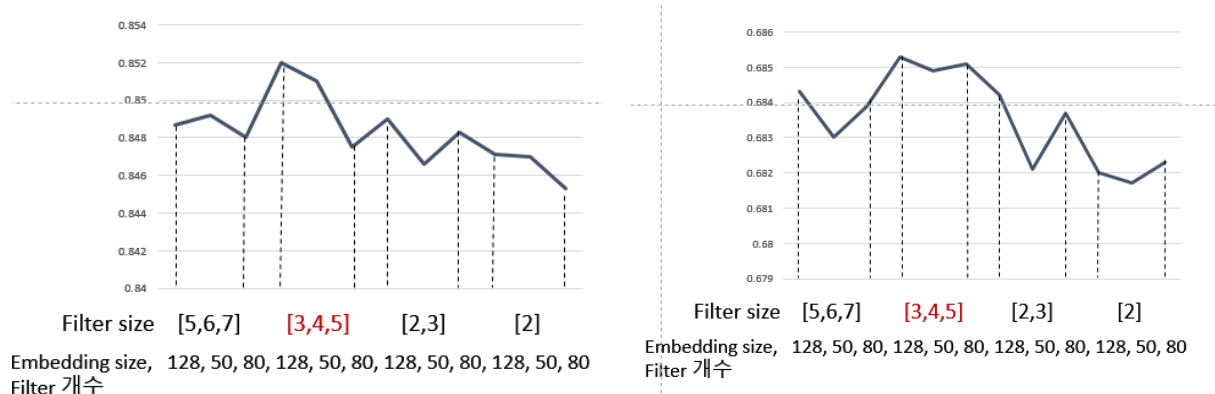
Filter size	num_filter = embedding_size	CNN(2 class)	3 class
[5,6,7]	128	0.8487	0.6843
	50	0.8492	0.6830
	80	0.8480	0.6839
[3,4,5]	128	0.8520	0.6853
	50	0.8510	0.6849
	80	0.8475	0.6851
[2,3]	128	0.8490	0.6842
	50	0.8466	0.6821
	80	0.8483	0.6837
[2]	128	0.8471	0.6820
	80	0.8470	0.6817
	50	0.8453	0.6823

다음은 CNN-LSTM 모델로 하이퍼 파라미터 값을 달리하여 각각 긍정, 부정(2 class) 분류와 긍정, 부정, 중립(3 class)분류를 실행한 결과이다.

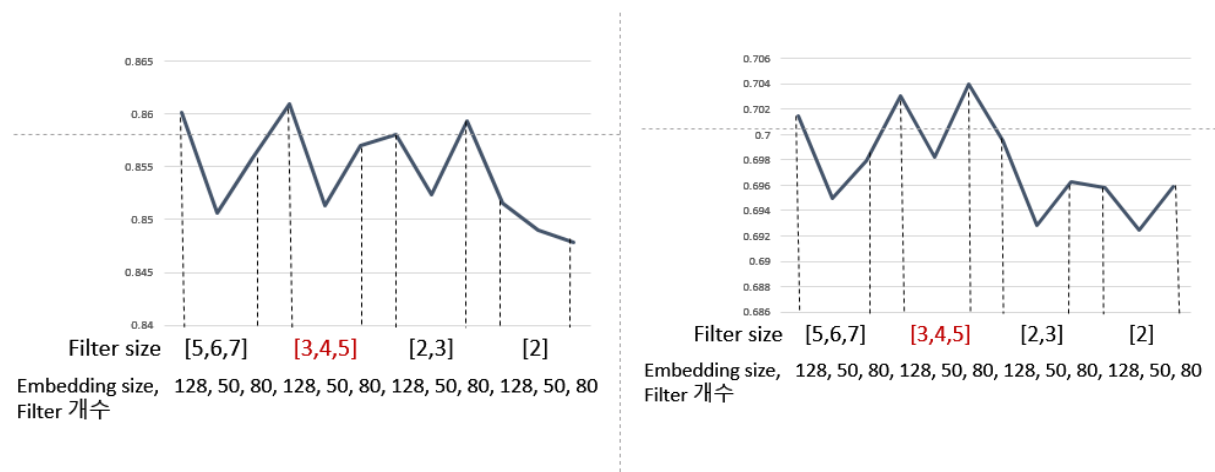
<표 9> 파라미터에 따른 CNN-LSTM모델의 정확도 비교

Filter size	필터 개수= embedding size	(2 class)	3 class
[5,6,7]	128	0.8601	0.7015
	50	0.8506	0.6950
	80	0.8558	0.6979
[3,4,5]	128	0.8609	0.7030
	50	0.8513	0.6982
	80	0.8570	0.7040
[2,3]	128	0.8580	0.6995
	50	0.8524	0.6928
	80	0.8593	0.6963
[2]	128	0.8515	0.6958
	50	0.8490	0.6925
	80	0.8479	0.6959

<그림 19>와 <그림 20>은 실험 결과를 꺾은선 그래프로 나타낸 것이다. 두 모델 모두 필터 사이즈가 [3,4,5]로 세 종류의 크기를 가지는 필터를 사용할 때, 그리고 임베딩 사이즈와 필터 개수가 각 128개 일 때 성능이 가장 좋았다.

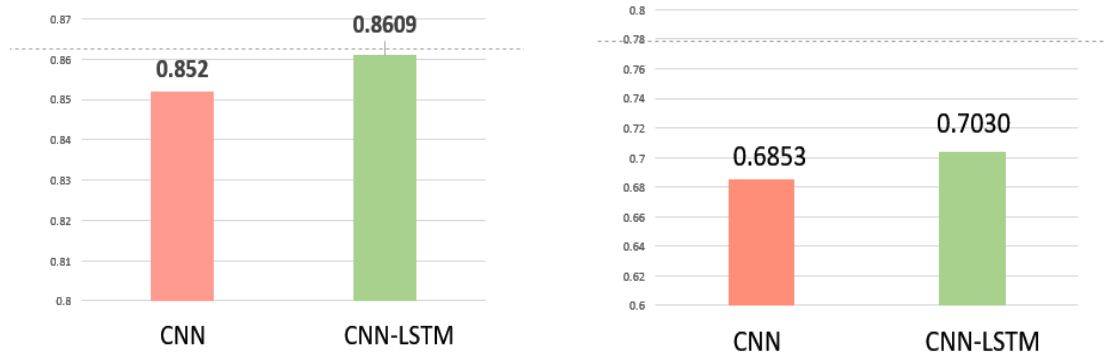


<그림 19> 파라미터에 따른 CNN 모델의 정확도



<그림 20> 파라미터에 따른 CNN-LSTM모델의 정확도

두 모델의 성능을 비교한 결과는 <그림 21>, <표 10>과 같다. 긍정과 부정으로 이진 분류하는 경우 CNN모델은 85% , CNN-LSTM 모델은 86%의 정확도를 보였다. 긍정, 부정과 중립까지 포함하여 분류한 경우는 CNN이 68% , CNN-LSTM이 70% 로 정확도가 하락하였다.



<그림 21> 최종 정확도 비교

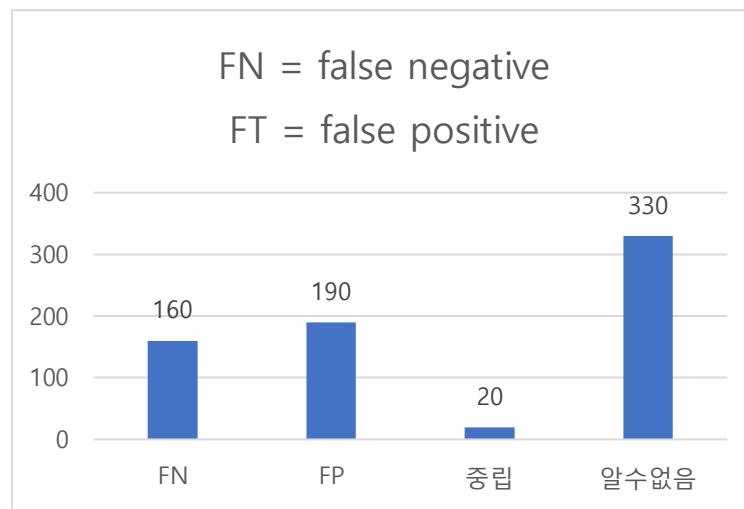
<표 10> 최종 정확도 비교

	CNN	CNN-LSTM
2 classes	0.8520	0.8609
3 calsses	0.6853	0.7030

4.3. 결과 분석

이진 분류를 수행하는 CNN 모델의 성능은 대략 85%로, 제대로 분류하지 못한 문장은 테스트 셋 5만문장 중 약 7000문장이다. 그 중 10%인 700문장을 표본으로 추려 분석해 보았다. <그림 23>

700개의 문장을 분석한 결과 긍정문장 160개 부정 문장 190개 중립적인 문장 20개 그리고 나머지 330문장은 직접 눈으로 보고 감성을 분류하기가 불가능했다. 분류가 불가능한 문장들은 함께 주어진 감성의 척도 와도 같은 별점과의 상관관계가 없는 문장들 이었는데, 본 논문에 사용된 데이터 전처리 방식은 별점으로 긍정과 부정 등의 class를 나누기 때문에, 학습과 훈련 데이터로 사용된 문장과 별점의 상관관계가 얼마나 있느냐에 따라서 모델의 성능이 달라질 수 있다. 중립적인 문장들은 레이블이 긍정 또는 부정으로 붙어있어서 분류에 실패 하게 된 것으로, 이또한 별점이 올바르게 주어지지 못한 경우이다. 감성 분석 모델 평가에 적합하지 않은 350문장을 제외하고 350 (각 160,190여개의 긍정, 부정)개의 문장들은 감성을 직접적으로 나타내지 않고 간접적인 표현, 그리고 일반적이지 않은 희귀한 표현으로 나타낸 경우가 많았다.



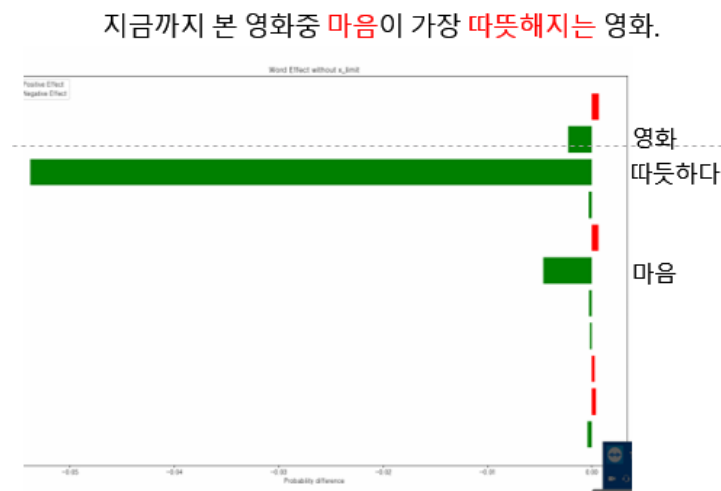
<그림 22> 분류하지 못한 700문장 분석 결과

<표 11> 모델에 부적합한 문장 예시

문장	Ground Truth	모델 판정
돈주고 봤으면 항의했을뻔한!!.	부정	긍정
역사얘기보다 영화배우, 감독들들에게 더 관심이 간다	부정	긍정
평점 10점 준 알바생에게 경배를	부정	긍정
성경에 보면 지나치게 의인도 되지 말고 지혜롭게 되지도 말라고 했다. 주인공의 지나친 착한남자 콤플렉스는 결국 모든 걸 엉망으로 만들었다.직장에서 해고되고 부인에게 거부당하고,그 모든 터전을 잃을만큼 불륜으로 생긴 아이를 보러가는게 그리 중요했을까???	긍정	부정
후속작 계획은 없나요..? ㄸ	긍정	부정
팬찮아요~	긍정	부정
오시이 마모루는 영화를 통해 책을 넘어서려 한다.	긍정	부정
작품성을 떠나서 보고 나면 너무너무 찝찝해 지는 영화..	긍정	부정
완전한 삶은 없다 그저 하나씩 배워나갈뿐.	긍정	부정

이어서 모델이 분류한 결과에 대한 근거, 즉 각 문장의 토큰들이 분류 확률에 얼마나 영향을 끼치는지를 실험을 통해 알아보았다. 우선 문장을 구성하는 토큰들을 순차적으로 하나씩 패딩과 같은 값으로 임베딩 한 후, 그에 따른 분류 확률을 구한다. 원래 문장의 분류확률에서 이것을 빼 값이 밑의 그림에서 각각의 막대그래프가 나타내는 값이다. 초록색으로 표현한 막대는 전체 문장에 긍정적인 영향을 주고, 빨간색 막대는 부정적인 영향을 끼치게 된다.

가장 먼저 ground truth 값이 긍정이고, 모델 분류 결과 또한 긍정인 문장을 살펴보겠다<그림 23>. 보면 ‘따뜻하다’ 라는 단어가 모델이 긍정으로 분류하는데 가장 큰 영향을 미쳤음을 알 수 있다. ‘마음’ 또한 긍정적으로 인식되었다.



<그림 23> True Positive 문장의 분류 근거 시각화

다음으로 ground truth 값이 부정, 모델 분류 결과도 부정인 경우이다<그림 24>. 발연기와 도저히
를 부정으로 인식하였다.



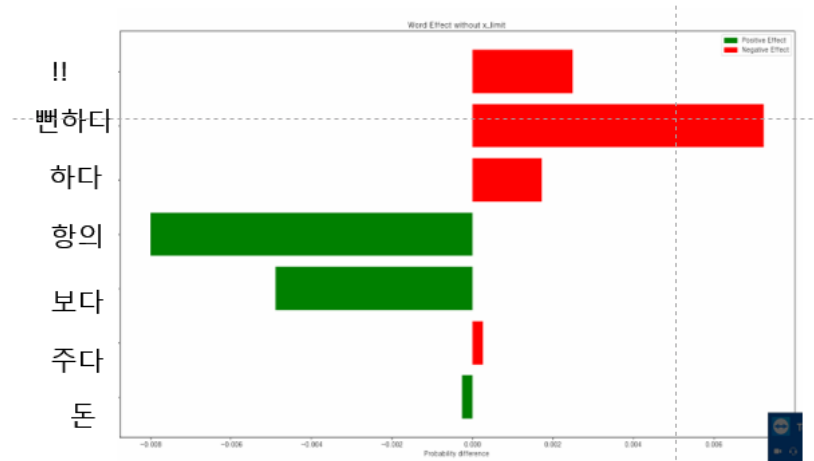
<그림 24> True Negative 문장의 분류 근거 시각화

이제 ground truth 과 모델의 분류결과가 다른 경우로, ground truth 가 긍정, 모델 분류결과가 부정인 문장을 분석하였다<그림 25>. 후속작, 계획까지는 긍정적으로 인식 되었지만, 없나요 라는 의문문이 없다라고 분석되어 부정으로 인식하고 전체 문장에 대해서도 부정으로 분류하였다. 원래 부정과 물음표 조합은 의미가 반전 되어야 하는데 그렇지 못하였다. 훈련 데이터에 이와 같은 의문문 패턴을 가진 문장이 충분하지 않아 물음표에 대한 의미 반전이 이루어지지 않은 것으로 보인다.



<그림 25> False Negative 문장의 분류 근거 시각화

마지막으로 부정 문장을 긍정으로 분류한 false positive 문장을 분석한 결과<그림 26>, ~할 뻔하다를 ‘영화가 뻔하다’의 뻔하다로 인식한 것을 볼 수 있는데, 이것은 형태소 분석이 제대로 되지 않은 경우이다. 또한 항의라는 단어가 긍정적인 어휘가 아닌데도 불구하고 모델은 긍정적으로 인식하였다. 단어에 대한 가중치 즉 단어벡터가 제대로 학습되지 않은 것으로 보이는데, 이 또한 이런 유형의 문장이 학습데이터에 별로 없었을 것으로 예상된다.



<그림 26> False Positive 문장의 분류 근거 시각화

5.결론 및 향후연구

본 논문에서는 주로 이미지 처리에 사용되는 CNN을 이용해 문장의 특징을 추출하여 감성분석을 진행하였고, 이에 문맥적 정보까지 보존하여 성능을 높이기 위해 LSTM layer를 추가하여 CNN-LSTM 합성 모델을 구현하였다. 필터 사이즈는 [3,4,5], 단어 임베딩 차원 수와 필터 개수는 각 128개일 때 성능이 가장 좋았으며, 결과적으로 CNN-LSTM 모델이 2 class 분류(공정과 부정)의 경우 약 1퍼센트, 긍정, 중립, 부정으로 이루어진 3class 분류 시에는 약 1.5퍼센트 정도 CNN 모델 보다 높은 성능을 보였다.

모델 구조 변화에 따라 성능은 크게 변하지 않았는데, 이유를 알아보기 위해 오류문장의 10퍼센트를 추출하여 분석한 결과 50퍼센트 정도가 감성분류 모델에 적합하지 않은 문장들로 구성되어 있었다. 따라서 데이터 전처리 과정에서 이러한 문장들을 제거한다면, 오류율이 크게 감소할 것으로 보인다. 이런 적합하지 않은 문장들을 제외하고 모델이 분류하지 못한 문장에 대해서 분류 근거를 예측하여 보았는데, 첫 번째로 형태소 분석이 올바르게 되지 않은 경우가 많았다. 이는 본 논문에서 사용한 tiwitter 분석기보다 좋은 성능을 가진 형태소 분석기(Mecab)를 사용해야 할 것이다. 두 번째로 자연어의 특성상 독창적인 표현 패턴이 다양하게 존재하다 보니, 각 상황에 맞게 단어의 의미대로 올바르게 가중치가 학습되어야 하는데 이것이 부족하였다. 이를 보완하기 위한 방법으로는 사전에 광대한 양의 데이터로부터 학습된 단어벡터(word2vec)를 삽입하거나, 모델이 희소성 있는 표현이더라도 충분히 학습할 수 있도록 훈련 데이터의 양을 늘리는 방법이 있다.

향후에 모델이 분류에 실패하는 오류 패턴에 대해서 추가적으로 분석하고, 그에 맞게 데이터를 늘리거나, 사전 학습된 단어벡터를 삽입하여 모델을 학습시킨다면 보다 높은 성능으로 감성 분류를 수행 할 수 있을 것으로 예상된다.

6. 참고 문헌

- [1]. Yoon Kim , “Convolutional Neural Networks for Sentence Classification”, 2014
- [2]. Chunting Zhou, Chonglin Sun, Zhiyuan Liu, Fancis C.M.Lau, “A C-LSTM Neural Network for Text Classification.” , 2015
- [3]. Socher, R., A. Perelygin, J.Y. Wu, J. Chuang, C.D. Manning, A. Ng, and C. Potts, “Recursive Deep Models for Semantic Compositionality over A Sentiment Treebank.”, Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, US, 2013, pp. 1631-1642.
- [4]. Irsoy, O. and C. Cardie, “Opinion mining with deep recurrent neural networks.”, In Proceedings of the 2014 conference on empirical methods in natural language processing, Doha, Qatar, 2014, pp. 720-728.
- [5]. Schuster, M. and K.K. Paliwal, “Bidirectional recurrent neural networks.”, IEEE Transactions on Signal Processing, Vol.45, No.11, 1997, pp. 2673-2681.
- [6]. Goodfellow, I., Y. Bengio and A. Courville, “Deep Learning”, The MIT Press, 2016.
- [7]. 김 민, “ Multi-channel CNN을 이용한 한국어 감성분석”, 2018
- [8]. Jangseong Bae, Changki Lee, “Skip-Connected LSTM을 이용한 감성 분석”, 2017
- [9] J. Park and P. Fung, “One-step and Two-step Classification for Abusive Language Detection on Twitter”, Proceedings of the First Workshop on Abusive Language Online, pp. 41-45, 2017.
- [10] Y. Zhang, S. Roller and B. Wallace, “MGNC-CNN: A Simple Approach to Exploiting Multiple Word Embeddings for Sentence Classification”, Proceedings of NAACL-HLT 2016, pp. 1522-1527, 2016.
- [11] K. Mo, “Text Classification based on Convolutional Neural Network with Word and Character Level”, Journal of Korean Institute of Industrial Engineers Online, 2018.
- [12]. Bengio, “Problem of learning long-term dependencies in recurrent networks.”, 1993
- [13]. Kaiming He , “Deep Residual Learning for Image Recognition.”, 2015
- [14]. Alex Krizhevsky , “ImageNet Classification with Deep Convolutional Neural Networks.”, 2012

- [15]. Turney, P.D , "Thumbs up or Thumbs Down? : Semantic Orientation Applied to Unsupervised Classification of Reviews. ", Proceedings of the 40th Annual Meeting on Association for Computational Linguistics, Philadelphia, Pennsylvania, US, pp. 417-424, 2002
- [16]. 김유영, 송민, “영화 리뷰 감성분석을 위한 텍스트 마이닝 기반 감성 분류기 구축”, 지능 정보 연구 , 제22권, 제3호, pp. 71-89, 2016
- [17]. 권수정, “word2vec과 rnn을 이용한 영화 리뷰의 감성분석”, 동국대학교, 2017.
- [18]. Diederik P. Kingma , “ADAM: A METHOD FOR STOCHASTIC OPTIMIZATION”, 2014
- [19]. <http://www.wildml.com/2015/12/implementing-a-cnn-for-text-classification-in-tensorflow/>
- [20]. <http://www.dbpia.co.kr/journal/articleDetail?nodeId=NODE07207329>
- [21]. Yeongtaek OhO Mintae Kim Wooju Kim , “Korean Movie review Sentiment analysis Using Parallel Stacked Bidirectional LSTM”, 2019
- [22]. 임좌상, 김진만 , “한국어 트위터의 감정 분류를 위한 기계학습의 실증적 비교” , 2014
- [23]. 이정훈, “감성분석 연구동향“, 2018년 춘계학술발표대회 논문집 제25권 제1호(2018. 5), pp. 358
- [24]. 이재준, “RNN을 이용한 한국어 감성분석. Sentiment Analysis based on Korean using Recurrent Neural Network: focused on Online Movie Review”, 2018