딥러닝 기반 게임 리뷰 만족도 및 카테고리 분류 시스템 설계 및 개발

양유정, 이보현, 김진실, 이기용
숙명여자대학교 소프트웨어학부 컴퓨터과학전공
e-mail : {diddbwjd96, kiyonglee}@sookmyung.ac.kr
{ lbhsos29, kjinsil0819}@gmail.com

Design and implementation of a satisfaction and category classifier for game reviews based on deep learning

Yu-Jeong Yang, Bo-Hyun Lee, Jin-Sil Kim, Ki Yong Lee
Division of Computer Science, Sookmyung Women's University

요 약

모바일 게임 산업의 발달로 많은 사용자들이 게임을 이용하면서, 그들의 만족감을 사용리뷰를 통해 드러낸다. 실제로, 각 리뷰의 범주가 모두 다르지만 현재 구글 플레이 앱 스토어(Google Play App Store)의 게임 리뷰 범주는 3 가지로 매우 제한적이다. 따라서 본 연구에서는 빠르고 정확한 고객의 요구를 필요로 하는 게임 소프트웨어의 특성을 고려하여 게임 리뷰를 입력했을 때, 게임의 운영 및 시스템에 맞도록 리뷰의 카테고리를 세분화하고 만족도를 분석하는 시스템을 개발한다. 제안 시스템은 인공신경망 모델인 CNN 을 평점을 기반으로 훈련시켜 리뷰에 대한 만족도를 도출한다. 또한 Word2Vec 을 이용해 단어들 간의 유사도를 구하고, 이를 활용한 단어 배열을 이용하여 가장 스코어가 높은 카테고리로 배정한다. 본 논문은 제안한 리뷰 만족도 및 카테고리 분류 시스템이 실제 효과적으로 리뷰를 보다 의미 있는 정보로써 제공할 수 있음을 보인다.

1. 서론

모바일 게임 산업의 발달로 많은 사용자들이 게임을 이용하면서, 그들의 만족감을 사용 리뷰들을 통해 드러내고 있다. 하지만 각 리뷰들의 주제가 다양하고 제한된 범주로 정렬되어 있는 탓에 개발자들이 시스템 개선 단계에서 사용자들의 의견을 파악하고 리뷰를 반영하는 데에 어려움을 겪고 있다. 따라서 리뷰를 각 범주별로 분류하고 사용자들의 의견을 분석해야 하는 필요성이 증가하고 있다. 본 논문에서는 사용자의 리뷰를 분석하여 게임에 대한 사용자의 반응을 파악하고 각 리뷰들을 게임의 구성, 운영 및 시스템에 대한 주제별로 분류하는 시스템을 설계 및 개발한다.

제안 시스템은 게임 리뷰의 평점을 만족도에 관한 세 가지 범주로 나누어 진행한다. 인공신경망 모델을 평점을 기반으로 훈련시켜 리뷰에 대한 만족도를 도출한다. 또한 단어들 간의 유사도를 활용하여 리뷰에 대한 카테고리를 분류한다. 각각의 카테고리와 유사도가 높은 단어들을 추출하여, 가장 스코어가 높은 카테고리로 배정한다. 본 논문은 제안한 리뷰 만족도 및 카테고리 분류 시스템이 실제 효과적으로 리뷰를 보다 의미 있는 정보로써 제공할 수 있음을 보인다.

본 논문의 구성은 다음과 같다. 2 장에서는 관련 연구를 간략히 살펴보고, 3 장에서는 딥러닝에 기반한 만족도 및 카테고리 분류 시스템 설계를 자세히 설명한다. 4 장에서는 실제 구현된 분석 및 분류 시스템의 수행 결과와 정확도를 보여준다. 마지막으로 5 장에서는 결론을 맺는다.

2. 관련 연구

Word2Vec[1]은 한 문장 내에서 단어의 등장 빈도 데이터를 이용해 이를 벡터 공간에 임베딩(embedding)하는 방식으로, 신경망을 통해 각 단어가 같이 등장한 빈도에 따라 두 단어 벡터의 거리를 정한다. 이를 통해 문장들 간의 유사도를 정의해 유사한 문장들 별로 군집화[2]할 수 있다. 하지만 특정 쿼리 단어 하나만을 이용해 해당 문장의 유사도를 결정하는 경우, 정확도가 낮아진다. 또한 한 문장 당 하나의 카테고리로 배정되기 때문에 여러 카테고리가 내포된 문장의 경우 정확히 분류되지 못한다.

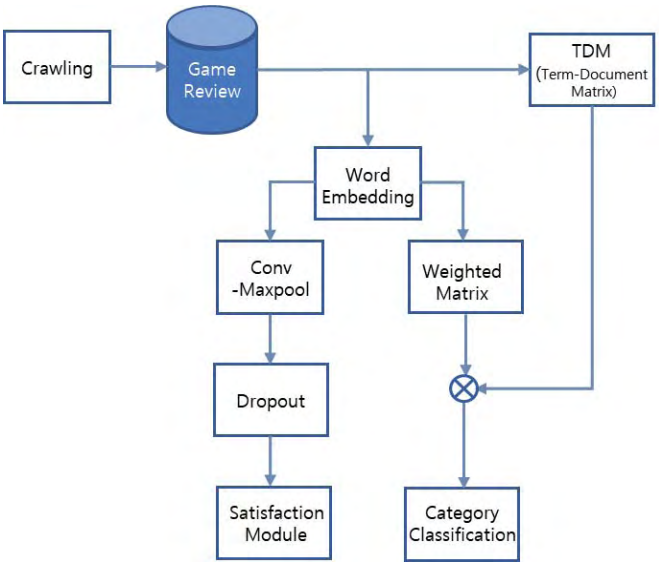
인공신경망 모델인 CNN(Convolutional Neural Networks)은 본래 이미지 처리를 하기 위해 만들어진

모델로 필터가 이미지의 지역인 정보를 추출, 보존하는 형태로 학습이 이루어진다. CNN 의 이러한 특징은 텍스트에 적용해 텍스트의 지역적인 정보, 즉 단어의 등장 순서 및 문맥의 정보를 보존해 문장 분류에도 높은 성능을 보였다 [3].

따라서 본 시스템은 CNN 을 활용하여 리뷰에 대한 사용자의 만족도를 분류한다. 또한 Word2Vec 을 이용하여 단어 간의 유사도를 반영한 단어 배열을 통해, 보다 정확도를 높여 해당 리뷰가 하나 이상의 카테고리를 배정받을 수 있는 시스템을 제안한다.

3. 게임 리뷰 데이터 카테고리 및 만족도 분류

본 논문에서 제안하는 시스템의 구조는 (그림 1)과 같다. 크롤링(crawling)을 통해 수집한 데이터를 바탕으로 단어들 간의 유사도를 이용하여 카테고리를 분류하고, 리뷰의 평점으로부터 만족도를 도출한다. 다음은 각 단계를 상세히 설명한다.



(그림 1) 시스템 구조도

3.1 데이터 수집 및 전처리

제안 시스템은 웹을 크롤링하여 게임 리뷰들을 수집한다. 본 논문에서는 모바일 애플리케이션 구매 사이트인 구글 플레이 앱 스토어(Google Play App Store)로부터 리뷰를 수집하였으며, Python 기반의 Selenium 프레임워크[4]를 사용하였다. 본 연구에서 사용한 게임 종류는 113 개이며 수집한 리뷰는 76,124 개이다.

리뷰 데이터
시뮬레이션모드예전껄로해주세요 보는맛도안나고 더재미없어졌어요
형태소 태깅
시뮬레이션/Noun, 모드/Noun, 예전/Noun, 껄/Noun, 로/Josa, 해주다/Verb, 보다/Verb, 맛/Noun, 도/Josa, 안나/Noun, 고/Josa, 더/Noun, 재미없다/Adjective
품사 제거
시뮬레이션/Noun, 모드/Noun, 예전/Noun, 껄/Noun, 해주다/Verb, 보다/Verb, 맛/Noun, 안나/Noun, 더/Noun, 재미없다/Adjective

(그림 2) 데이터 전처리 예시

각 리뷰별로 작성자 이름, 내용, 평점, 유용한정도, 날짜, 카테고리를 추출해 낸다. KoNLPy Twitter 라이브러리[5]를 통해 형태소를 나눈 뒤, 분석에 필요하다고 판단되는 명사, 형용사, 동사만 선별한다. 수집한 리뷰는 데이터를 학습에 적합한 형태로 (그림 2)와 같이 변환한다.

3.2 Word2Vec 기반 카테고리 분류

우선, 형태소가 태깅(tagging)된 데이터를 입력 데이터로 하여, Word2Vec의 Skip-gram방식[6]을 사용하여 100차원의 벡터로 임베딩 시킨다. 임베딩 과정에서 정확도를 향상시키기 위해 말뭉치(corpus) 출현 빈도가 20번 미만인 단어는 분석에서 제외하였다. 이후 각 카테고리에 배정하기 위해 카테고리별 세부 단어를 지정한다. 본 연구에서는 Word2Vec을 이용해 해당 카테고리의 중심 단어와 유사도가 높은 단어들을 추출한 후, 단어 배열을 구성함으로써 정확성을 높였다. 카테고리는 게임의 운영 및 시스템에 관한 8개의 범주로 지정하였다. 카테고리별 세부 단어는 5개씩 구성하였으며, 그 내용은 <표 1>과 같다. 임베딩 된 단어 벡터 행렬은 100차원의 벡터 공간 좌표를 가진다. 따라서 계산의 편의성을 위하여 유클리디안 거리(Euclidean distance) 공식을 사용하여 2차원의 거리 행렬로 변환한다. 거리행렬을 0에서 1사이의 범위로 정규화 시킨 후, 카테고리의 세부 단어들만 남기고 나머지 단어들을 제거한다. 이렇게 세부 단어에 따른 행렬을 다시 카테고리 별로 합산하여 최종 가중치 배열(weighted matrix)을 구한다. (그림 3)은 가중치 행렬로 카테고리 내 특정 쿼리 단어인 '결제'와 유사한 '환불'은 높은 가중치(0.9)를, 상대적으로 유사도가 떨어지는 '캐릭터'는 낮은 가중치(0.1) 갖는 것을 보여준다.

리뷰를 구성하는 단어의 개수만큼 가중치 행렬을 참조하게 되면 계산량이 증가하기 때문에, 수집한 리뷰들을 바탕으로 TDM(Term-Document Matrix)을 구축한다. TDM은 단어문서행렬로 행은 '리뷰', 열은 '단어'이며 해당 행에서 열에 해당하는 단어의 등장 여부에 따라 0 또는 1로 표시한다.

마지막으로 카테고리별 가중치 행렬과 TDM을 내적하여 각 리뷰 별 해당 카테고리에 대한 스코어를 구한다. 스코어 배열을 내림차순으로 정렬하여 1순위를 가지는 카테고리를 우선적으로 배정하였으며, 다음 순위의 카테고리와 스코어 차이가 기준 값(0.35) 이하인 경우에는 두 개의 카테고리에 배정받을 수 있도록 하였다.

<표 1> 각 카테고리별 세부단어

카테고리	카테고리 세부 단어
결제	결제, 구입, 구매, 현질, 환불
계정	계정, 아이디, 연동, 구글, 로그인
서버	연결, 서버, 접속, 로딩, 렉
구성	맵, 이벤트, 퀘스트, 스테이지, 미션
연출	배경, 그래픽, 퀄리티, 사운드, 디자인
캐릭터	스킬, 너프, 영웅, 캐릭터, 신캐
시스템	용량, 다운, 앱, 실행, 설치
기타	광고, 신고, 채팅, 욕, 처벌

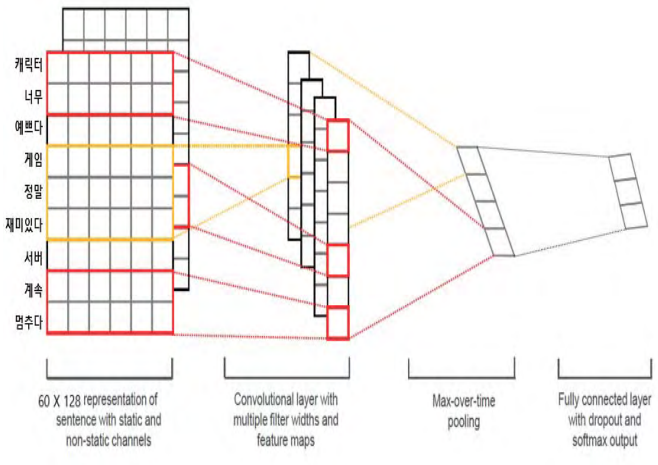
	결제	이벤트	...	캐릭터	환불
결제	1.0	0.2	...	0.1	0.9
계정	0.2	0.1	...	0.3	0.2
구성	0.1	0.8	...	0.7	0.1
...	...	...	...	...	...

(그림 3) 카테고리 별 가중치 행렬

3.3 CNN 기반 만족도 분류

각 리뷰에 대한 만족도 분류 모델의 학습 알고리즘으로는 CNN을 사용하였다. 본 기능은 TensorFlow[7]로 구현하였으며 만족도 분류 과정은 (그림 4)와 같이 진행된다.

데이터 전처리 과정에서 얻어진 형태소가 태깅된 단어별로 ID를 부여해 단어 사전을 만든다. 구축된 사전을 기반으로 각 리뷰를 ID의 나열로 변환한다. 모델에 사용할 데이터는 리뷰 내용과 1점에서 5점사이의 평점으로 구성된다. 본 기능에서는 평점에 따른 만족도를 3가지 범주로 분류하였으며, 리뷰 데이터의 평점이 1점인 경우 '불만족', 2점과 3점의 경우 '보통', 4점과 5점의 경우 '만족'으로 분류하였다. 분류 모델의 입력 값과 출력 값은 (그림 5)와 같다.



(그림 4) CNN 기반 만족도 분류 프로세스

입력 값	출력 값
[게임 재미있고 (하략), 4.0]	[[22, 453, …], [0, 0, 1]]
[서버 느려요 (하략), 1.0]	[[63, 984, …], [1, 0, 0]]
[캐릭터 예쁘고 (하략), 3.0]	[[115, 314, …], [0, 1, 0]]

(그림 5) 입력 값과 출력 값 예시

분류 모델의 첫 번째 층(Layer)은 단어 색인을 저차원 벡터 표현으로 맵핑(mapping)하는 Lookup 테이블이다. 입력 데이터에 해당하는 문장의 길이는 60개의 단어 ID로 고정하였으며, 이를 128차원으로 임베딩 한다. 개별 단어 ID에 대응하는 벡터는 가중치로 사용하며 초기값은 임의로 설정한 후, 학습 과정에서 조금씩 업데이트하는 방법을 사용하였다.

두 번째 층은 합성곱 레이어로, 배치(batch) 단위로 진행하며 필터를 이용하여 합성곱 연산을 수행한다. 필터의 차원 수는 (임베딩 차원 수)*(채널 수)*(필터 사이즈) 크기를 가지며 본 기능에서는 채널 수는 1, 배치 사이즈는 64로 고정하였다. 필터 사이즈는 [3, 4, 5]로 주어 각각 크기가 다른 필터를 사용하였으며, 최대 풀링(max-pooling) 과정을 통해 이를 하나의 큰 피쳐(feature) 벡터로 병합한다. CNN의 과적합(overfitting)을 방지하기 위해 드롭아웃(dropout) 층을 추가하여 뉴런의 일부를 확률적으로 비활성화한다. 학습 시에는 드롭아웃 비율을 0.5로 적용하였고, 실제 평가시에는 1을 사용하였다.

최대 풀링 결과로 나온 피쳐 벡터를 전결합(fully-connected) 층에 통과시켜 각 클래스(불만족, 보통, 만족)에 해당하는 스코어를 구한 후, 교차 엔트로피 오차(cross-entropy loss)를 구한다. 이를 이용해 역전파(backpropagation)를 수행하여 필터와 Lookup 테이블의 가중치 등 파라미터들을 업데이트한다. 마지막으로 스코어 중 가장 높은 점수를 분류로 선택하는 예측 과정을 수행한다.

4. 시스템 수행 결과 및 정확도 분석

4.1 카테고리 분류 결과 및 정확도 분석

(그림 6)은 실제 리뷰를 바탕으로 카테고리 분류를 수행한 결과이다. 리뷰가 너무 짧은 경우 명확한 분류가 어려워, 형태소 분석 결과로 나온 토큰(token)이 6 개 이상인 리뷰만을 대상으로 정확도를 측정하였다. (그림 6)에서 스코어 차는 우선 순위가 가장 높은 카테고리와 그 다음 순위 간의 배열 값의 차이다. 스코어 값의 차가 기준치(0.35) 보다 큰 경우, 그 내용이 명확하다고 판단하여 하나의 카테고리로만 배정하였다. 그렇지 않은 경우, 리뷰의 내용이 두 가지 이상의 카테고리를 포함한다고 판단하여 가장 높은 스코어를 지니는 두 개의 카테고리로 배정하였다.

분류 정확도는 수집한 리뷰 데이터 중 100 개를 임의로 추출하여 5 번 측정하였으며, 약 85%의 정확도를 보였다.

리뷰	카테고리	스코어 차
잘못 결제했어요. 환불해 주세요	결제	0.76
폰 새로 개통 했는데 애니팡3 몇년 했는데 처음부터 다시해야되네요 카톡 옛계정으로 할려고하면 있는 계정이라 카톡 삭제하라고 뜨네요.	계정	0.5
이번 퀘스트 너무 어렵고 로딩이 너무 길어서 자꾸 멈춰요.	구성, 서버	0.15
적대모드랑 유저 신고하기 같은 기능도 있었으면 좋겠습니다!!월챗모드에서 왜 이렇게들 싸우시는지 모르겠네용	기타	0.7
업뎃하라고 해서 업뎃 눌렀는데 게임재설치가 안되고 아이템 환불해주세요	시스템, 결제	0.27

(그림 6) 카테고리 분류 결과 예시

리뷰	만족도
잼있네요 할만하네요	만족
캐릭이랑 게임은 이쁜데 게임은 그냥 그럭저럭이에요	보통
왜 업뎃이 안됩니까 재시도하란 말만 나오고 안돼요 왜 짜증나네 진짜	불만족
진짜 재재밌어요 추천!!! 제작자님 감사해요 이런 게임 만들어 주셔서	만족
검은사막 컴퓨터로 나왔을 때부터 하고 싶었는데 모바일로 나와서 좋음	만족

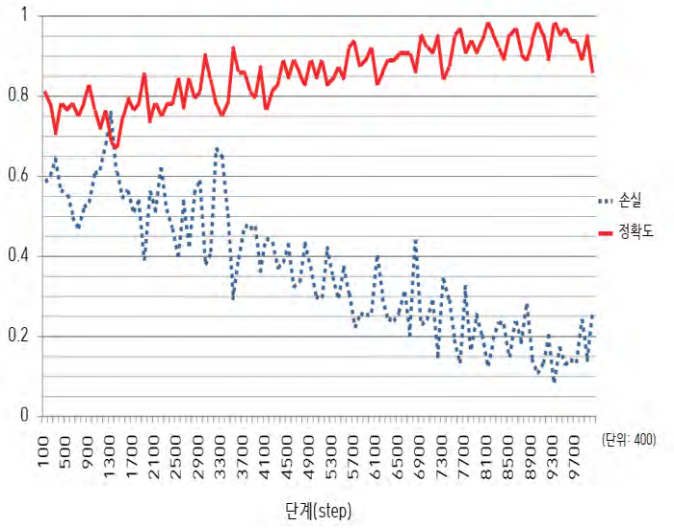
(그림 8) 만족도 분류 결과 예시

4.2 만족도 분류 결과 및 정확도 분석

본 논문에서는 수집한 데이터 중, 80%를 학습용 데이터로 20%를 검증용 데이터로 분리하였으며, 분류 모델은 10,000 회 학습시켰다.

정확도 분석을 위하여 집합 사이의 확률 분포 차이인 교차 엔트로피 오차를 사용하였다. (그림 7)은 모델에 대한 정확도를 보여주는 그래프로 정확도 그래프와 손실 그래프로 구성된다. 정확도 그래프는 검증 데이터에 대한 학습된 모델의 예측 결과의 정확성을 나타낸 그래프이며, 손실 그래프는 교차 엔트로피 손실 값의 평균 값을 표현한 그래프이다. 검증 데이터에 대한 예측 결과, 단계가 7500 을 넘어서면서 정확도는 90% 내외로 수렴하는 모습을 보였고 교차 엔트로피 평균값은 20% 내외인 것을 볼 수 있다. 학습이 진행됨에 따라 두 그래프 간의 간격이 넓어지는 것으로 보아, 검증 데이터에 대한 모델의 정확도가 높아지는 것을 확인할 수 있다.

(그림 8)은 리뷰에 따른 만족도 분석 결과이다. 검증용 데이터를 기반으로 정확도를 측정하였으며 약 90%의 정확도를 보였다.



(그림 7) 만족도 분류 모델의 정확도 및 손실 그래프

5. 결론

본 논문에서는 게임 리뷰를 크롤링하여 카테고리를 분류하고, CNN 을 통하여 사용자 만족도 도출하는 시스템을 구현하였다. 게임 소프트웨어 특성상 사용자들의 반응과 요구사항을 정확하게 알아내어 시스템에 사용자의 피드백을 빠르게 반영하는 것이 중요하다. 본 시스템을 통해 사용자들의 게임에 대한 만족도를 분석하여 사용자의 반응을 확실하게 파악할 수 있다. 또한 게임 시스템의 유지 및 보수가 필요한 부분 혹은 사용자의 요구가 많은 부분에 대한 카테고리를 명확하게 파악하여 시스템 개선에 도움이 될 것을 기대한다.

Acknowledgement

이 논문은 2018 년도 정부(교육부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임 (No.2018R1D1A1B07045643).

참고문헌

[1] Word2vec, https://en.wikipedia.org/wiki/Word2vec
[2] 손지영, “word2vec 을 이용한 거리 기반의 음악 가사 클러스터링 기법”, 숭실대학교 소프트웨어특성화대학원 학위논문(석사), 2018
[3] Kim, Yoon. "Convolutional Neural Networks for Sentence Classification", Empirical Methods on Natural Language Processing, 2014
[4] Selenium https://en.wikipedia.org/wiki/Selenium_(software)
[5] KoNLPy tag Pakcage, http://konlpy.org/ko/v0.4.3/api/konlpy.tag/
[6] Skip-gram https://en.wikipedia.org/wiki/N-gram#Skip-gram
[7] TensorFlow, https://www.tensorflow.org/