Word2vec 임베딩을 이용한 책 추천 시스템

Book Recommendation System using Word2vec

요 약

현대 사회는 다양한 선택지와 방대한 정보로 인해 결정을 내리기 어렵다. 추천 시스템은 사용자의 성향과 상품을 분석하여 현재 상황에 맞는 선택지를 제시해준다. 본 논문에서는 신경망 기반의 Word2vec 임베딩을 이용해 책 추천 모델을 구현하였다. 그리고 다양한 책 추천 모델을 만들고 비교하며 최적의 책 추천 모델을 찾았다. 이를 통해 Word2vec을 사용한 추천 시스템의 구현 가능성에 대해 살펴보았다.

1. 서론

삶은 무한한 선택의 연속이다. 특히 현대 사회는 다양한 선택지와 방대한 정보로 인해 결정을 내리기 힘들다. 추천 시스템은 현재 상황에 맞는 선택지를 제시하여 선택에 소모되는 비용을 감소시켜준다.

추천이란 특정 시점에 특정 고객이 필요로 하는 상품을 제공하는 것을 의미한다. 추천 시스템은 영화, 음악, 책, 물건 등 다양한 상품을 추천한다. 사람들은 영화가 보고 싶을 때 현재까지 개봉된 모든 영화 정보를 검색할 필요가 없다. 추천 시스템에서 제공해주는 추천 목록 중 보고 싶은 영화를 선택하면 된다.

추천 시스템의 핵심은 사용자가 필요로 하는 상품을 정확히 추천해주는 것이다. 추천 시스템의 성능을 높이기 위해 다양한 알고리즘을 사용한 연구들이 진행되었다. Zhang and Min은 Random Forest를 사용해 추천 시스템을 만들고 MovieLens 데이터셋을 사용해 실험을 진행하였다[1]. Ghazanfar 와 Bennett은 Naive Bayes Classifier를 사용해 추천 시스템을 만들고 MovieLens와 FilmTrust 데이터셋을 사용해 성능을 측정하였다[2]. Sarwar과 Karypis는 Clustering을 사용해 실시간으로 상품을 추천하는 연구를 진행하였다[3].

정확한 추천 시스템은 회사 매출에 영향을 미친다. 아마존, 구글, 넷플릭스와 같은 세계적 기업들은 사용자가 필요로 하는 상품을 적절한 시점에 추천해주며 매출을 올리고 있다[4].

본 논문에서는 신경망 기반의 Word2vec[5, 6] 임베딩을 이용해 책 추천 시스템을 구현하였다. Word2vec은 고차원 데이터를 저차원 데이터로 변환시키고 단어를 벡터로 변환시켜주는 단어 임베딩 기법이다. 본 논문에서는 Word2vec에 문장이 아닌 사용자가 읽은 책 목록을 입력하여 책 추천 모델을 만들었다. 그리고 Word2vec의 하이퍼파라미터를 변경하며 18개의 책 추천 모델을 만들고 성능을 비교해보았다. 이를 통해 최적의 책 추천 모델을 찾고 Word2vec 임베딩을 이용한 추천 시스템의 구현 가능성에 대해 살펴보았다.

2. 관련 연구

2.1 협업 필터링 (Collaborate Filterling)

협업 필터링은 사용자의 구매 패턴을 분석해 유사한 성향의 사용자에게 상품을 추천해주는 방법이다[7]. 협업 필터링은 사용자의 선호 상품과 구매 패턴 같은 데이터를 많이 가지고 있을수록 성능이 높아진다. 협업 필터링은 사용자 기반(User based) 협업 필터링과 아이템 기반(Item based) 협업 필터링으로 나누어진다.

사용자 기반 협업 필터링은 사용자 간의 유사도를 계산하여 가장 유사한 성향의 사용자가 구매했던 상품을 추천해주는 방법이다. 하지만 신규 사용자 같은 경우에는 분석할 정보가 존재하지 않기 때문에 추천에 많은 어려움을 겪는다.

아이템 기반 협업 필터링은 아이템 간의 유사도를 측정하여 사용자가 아이템을 조회했을 때 유사한 아이템을 추천해주는 방법이다[8]. 아이템에 대한 평가가 없는 신규 사용자에게도 추천할 수 있다.

본 논문에서 구현한 추천 시스템은 사용자가 읽은 책 목록을 분석해 아이템 간의 유사도를 측정하는 아이템 기반 협업 필터링에 속한다.

2.2 Word2vec

Word2vec은 2013년 구글에서 발표한 단어 임베딩 방법이다. 학습 방법에는 Continuous Bag Of Word(CBOW)와 Skip-gram 두 가지 방법이 존재한다.

CBOW는 주변 단어들을 사용해 목표 단어를 예측하는 학습 방법이다. 그림 1의 CBOW와 같이 'He is _ boy' 라는 문장을 모델에 입력하면 주변 단어 'He', 'is', 'boy'를 사용해 목표 단어 'a'를 예측하도록 학습한다.

Skip-gram은 목표 단어를 사용해 주변 단어를 예측하는 학습 방법이다. 그림 1의 Skip-gram과 같이 'He is a boy' 라는 문장을 Skip-gram 모델에 입력하면 'a' 라는 목표 단어를 사용해 'He ', 'is', 'boy' 라는 주변 단어를 예측한다.

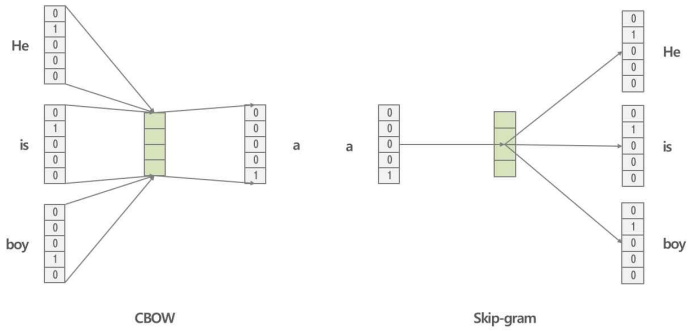

그림1. CBOW와 Skip-gram 학습 방법

3. 제안 방법

본 논문에서는 Gensim[9] 라이브러리를 사용하였다. Word2vec에 사용자가 읽은 책 목록을 학습 데이터로 입력하여 모델을 만들었다. 해당 모델을 사용해 책마다 고유한 벡터를 만들고 벡터 간의 거리를 비교해 거리가 유사한 책을 추천하였다.

비슷한 맥락에 등장하는 단어들은 서로 유사한 의미가 있는 경향이 있다[10]. 사용자가 읽은 책 목록 또한 이와 같은 속성을 가지기 때문에 Word2vec에 적용하면 유사한 책을 추천할 수 있다고 생각했다.

Word2vec의 여러 인자 중 training algorithm, size, iter 3개의 인자를 변경하며 총 18개의 책 추천 모델을 만들었다. training algorithm은 Word2vec의 학습 방식을 의미하며 본 논문에서는 CBOW와 Skip-gram을 사용하였다. size는 단어 벡터의 차원을 의미하며 300, 700, 1000으로 설정하였다. iter는 학습 반복회수를 의미하며 1, 10, 50으로 설정하였다. window는 문장 내 현재 단어와 예측 단어 사이의 최대거리를 의미하며 5로 설정하여 학습을 진행하였다.

algorithm	size	iter
CBOW	300	1
		10
		50
	700	1
		10
		50
	1000	1
		10
		50
Skip-gram	100	1
		10
		50
	700	1
		10
		50
	1000	1
		10
		50

표1. 18개 책 임베딩 모델 설명

4. 실험 및 결과

4.1. 데이터셋

본 논문에서는 Kaggle의 goodbooks-10k에서 제공하는 to_read.csv 데이터셋을 학습 데이터와 테스트 데이터로 사용했다[11]. to_read.csv 데이터셋은 user_id와 book_id로 구성되었고 데이터셋의 형태는 표2와 같다. to_read.csv에 저장된 사용자 수는 43,578명이고 책 권수는 9,986권이다.

Word2vec으로 만든 모델을 검증하기 위해 데이터셋을 학습 데이터와 테스트 데이터로 분류하였다. 학습 데이터는 40,578명이 읽은 책 목록으로 분류했고 테스트 데이터는 3,000명이 읽은 책 목록으로 분류했다.

user_id	book_id
1	112
1	235
1	533
2	4
2	11
2	13

표2. to_read.csv 데이터셋 형태 [11]

4.2 평가 방법

평가는 테스트 데이터를 사용했으며 본 논문에서 사용한 모델 평가 방법은 다음과 같다.

$$T_i = \frac{S_i \cap B_i}{B_i} \; (i = test\ books\ id)$$

B_i는 i번 책을 읽은 사용자 수를 의미하며 S_i는 i번 책과 관련된 책을 읽은 사용자 수를 의미한다. 임베딩 모델에서는 한 책당 100개의 관련 있는 책을 추천한다.

만약 3,000명의 사용자 중 27번 책을 읽은 사용자는 총 32명이고 27번 책을 읽은 사람 중 해당 모델에서 추천한 책(22, 28, 49 ... 257, 1024)을 읽은 사용자가 5명이면 T_{27}은 0.15가 된다.

4.3 결과

결과는 CBOW 방법의 size가 300이고 iter는 50인 모델이 0.622로 가장 높은 성능을 보였다. 다음으로 CBOW 방법의 size가 1000이고 iter가 50인 모델이 0.6182의 성능을 보였다.

실험 결과를 통해 여러 인자 중 iter가 성능에 가장 큰 영향을 미치는 것을 확인할 수 있다. 표3을 보면 iter를 1에서 10으로 변경했을 때 결과가 약 2배 이상 증가한다. CBOW 모델에서는 iter가 증가할수록 성능도 향상되었다. 하지만 Skip-gram 모델에서는 iter가 10일 때 성능이 가장 좋았고 iter가 50일 때 오히려 성능이 떨어지는 것을 볼 수 있다.

CBOW와 Skip-gram 모두 size가 작을수록 성능이 높아지는 경향이 있다. 특히 Skip-gram에서 iter가 50인 모델들을 비교해보면 성능 차이가 크다는 것을 확인할 수 있다.

algorithm	size	iter	result
CBOW	300	1	0.3047
		10	0.524
		50	**0.622**
	700	1	0.2915
		10	0.5196
		50	0.6175
	1000	1	0.2826
		10	0.5182
		50	0.6182
Skip-gram	300	1	0.3573
		10	0.6038
		50	0.5643
	700	1	0.3551
		10	0.6042
		50	0.5142
	1000	1	0.3507
		10	0.6048
		50	0.5071

표3. 모델 평가 결과

5. 결론 및 향후 연구

본 논문에서는 Word2vec임베딩을 이용해 책 추천 모델을 만들고 비교하며 최적의 추천 모델을 만들었다. 결과는 CBOW 모델의 size가 300이고 iter는 50인 모델의 성능이 가장 좋았다. 실험을 통해 여러 하이퍼 파라미터 중 iter가 성능에 가장 크게 영향을 미친다는 것을 확인했다.

본 논문에서는 여러 단어 임베딩 방법 중 Word2vec 임베딩을 이용해 모델을 만들고 비교해보았다. 추후, Glove[12], ELMo[13], BERT[14]와 같은 다른 단어 임베딩 기법을 적용해 Word2vec과 비교해보고 최적의 책 추천 모델을 찾고자 한다.

참고문헌

[1] Heng-RuZhang,FanMin “Three-way recommender systems based on random forests”, Journal Knowledge-Based Systems archive Volume 91 Issue C, January 2016 Pages 275-286

[2] Mustansar Ali Ghazanfar and Adam Prugel-Bennett “An Improved Switching Hybrid Recommender System Using Naive Bayes Classifier and Collaborative Filtering”, The 2010 IAENG International Conference on Data Mining and Applications. 17 - 19 Mar 2010.

[3] Badrul M. Sarwar, George Karypis, Joseph Konstan and John Riedl “Recommender Systems for Large-scale E-Commerce: Scalable Neighborhood Formation Using Clustering” , 5th International Conference on Computer and Information Technology

[4] CARLOS A. GOMEZ-URIBE and NEIL HUNT “The Netflix Recommender System: Algorithms, Business Value, and Innovation”, Journal ACM Transactions on Management Information Systems (TMIS) TMIS Homepage archive Volume 6 Issue 4, January 2016 Article No. 13

[5] Tomas Mikolov, Kai Chen, Greg Corrado, Jeffrey Dean “Efficient Estimation of Word Representations in Vector Space”, Conference: Proceedings of the International Conference on Learning Representations (ICLR 2013)

[6] Tomas Mikolov, Kai Chen, Greg Corrado, Jeffrey Dean “Efficient Estimation of Word Representations in Vector Space”, Conference: Proceedings of the International Conference on Learning Representations (ICLR 2013)

[7] Xiaoyuan Su and Taghi M. Khoshgoftaar “A Survey of Collaborative Filtering Techniques”, Hindawi Publishing Corporation Advances in Artificial Intelligence Volume 2009, Article ID 421425, 19 pages

[8] Badrul Sarwar, George Karypis, Joseph Konstan and Jonh Ridel “Item-Based Collaborative Filtering Recommendation Algorithms”, Proceeding WWW '01 Proceedings of the 10th international conference on World Wide Web Pages 285-295

[9] Gensim, https://radimrehurek.com/gensim/. Sep. 2017

[10] Georgiana Dinu, Stefan Thater, Soren Laue “A comparison of models of word meaning in context”, 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 611-615

[11] https://www.kaggle.com/zygmunt/goodbooks-10k

[12] Jeffrey Pennington, Richard Socher, Christopher D. Manning “GloVe: Global Vectors for Word Representation” Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 1532-1543,

[13] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, Luke Zettlemoyer “Deep contextualized word representations”, Proceedings of NAACL-HLT 2018, pages 2227-2237

[14] Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova “BERT: Pre-training of Deep

Bidirectional Transformers for Language Understanding", arXiv:1810.04805v1 [cs.CL] 11 Oct 2018