지식 그래프를 활용한 추천 서비스

방 학 3 주 차

가천대학교 이다

방학 2주차

RippleNet

이번 주차에서 리뷰할 논문

RippleNet: Propagating User Preferences on the Knowledge Graph for Recommender Systems
(2018, Wang et al.)

RippleNet

Collaborative Filtering이 가진 문제점

(1) Sparsity

데이터가 sparse 하면 추천에 참고할 근거가 부족해서 정확도가 떨어짐

(2) Cold Start Problem

한번도 평점을 남겨 본 적이 없으면 추천이 불가능함

RippleNet

이를 해결하기 위해 내놓은 대책

"부수적 정보를 추가적으로 제공하자! "

ex1) Mohsen Jamali , Martin Ester (2010) :

- **"사용자 간의 소셜 네트워크를 가정하고, 친한 사람들의 평점을 기반으로 추천하자!"**

ex2) Hongwei Wang, Fuzheng Zhang 외 4인 (2018) :

- **"엔티티 레벨에서 추출해서 사용자 정보와 사용자 간 관계 정보, 프로필 지식을 뽑아내고 참고하자"**

ex2) Yu Sun, Nicholas Jing Yuan 외 3인 (2017) :

- **"맥락(context) 정보를 뽑아내서 활용하자!"**

Knowledge Graph

이러한 부수적 정보 중 하나가 바로 Knowledge Graph임

- 다른 것들보다 더 유익하고 아이템에 관련되어 있음
- Knowledge Graph를 부수적 정보로 제시한 연구 :

1. NELL : Never-Ending Language Learning / 데이터셋

2. DBpedia : 위키백과 정보로부터 구조화된 내용을 추출하는 프로젝트

3. Google Knowledge Graph : 구글에서 사용하는 상업용 지식 그래프 이자 지식 베이스

4. Microsoft Satori : 마이크로소프트에서 사용하는 상업용 지식 그래프 이자 지식 베이스

RippleNet

Knowledge Graph

지식 그래프는 3가지 측면에서 추천에 도움이 됨

1. 항목 간의 의미적인 연관성을 도입하여 잠재적 연관성을 찾아냄

2. 다양한 유형의 관계로 구성되어 사용자의 관심사를 합리적으로 확장하고 다양성을 높임

3. 사용자의 과거 기록과 추천을 연결하여 설명력을 부여함

RippleNet

Knowledge Graph

지식 그래프는 다양한 분야에서 활용되고 있음.

- 활용되는 분야 :

1. Knowledge Completion : 완성되지 않은 그래프를 완성하는 분야
2. Question Answering : 사용자로부터 받은 질문에 대해 대답을 해주는 분야
3. Word Embedding : 단어를 벡터로 표현하는 방법을 연구하는 분야
4. Text Classification : 주어진 문장에 대해 label이나 class를 분류하는 분야

RippleNet

Exsting KG-Aware Recommendation

현재 존재하는 방식 1 : Embedding-based Method

- 지식 그래프를 'KG Embedding'이라는 방식으로 전처리하고, 여기서 얻은 entity embedding을 추천 프레임워크에 넣음.

1. Deep Knowledge-aware Network (Hongwei Wang et al., 2018)
   - entity embediing과 word embedding을 서로 다른 채널로 보고, 이들을 결합하는 CNN 프레임워크를 설계함.

2. Collaborative Knowledge base Embedding (Fuzheng Zhang et al., 2016)
   - 하나의 Bayesian 프레임워크에 CF 모델, knowledge embedding, text embedding, image embedding을 결합함.

RippleNet

Exsting KG-Aware Recommendation

현재 존재하는 방식 1 : Embedding-based Method

- 지식 그래프를 'KG Embedding'이라는 방식으로 전처리하고, 여기서 얻은 entity embedding을 추천 프레임워크에 넣음.

3. Signed Heterogeneous Information Network Embedding (Hongwei Wang, 2018)

- **sentiment/social/profile network를 임베딩하기 위해 Deep autoencoder을 설계함**

이러한 임베딩 기반 밥업들은 높은 유연성을 가지고 있지만, 추천보다는 link 예측 같은 곳에 더 적합함.
=> 이를 통해 얻은 entity embedding들은 항목 간의 관계를 확인하는데 덜 직관적이고 덜 효과적임.

RippleNet

Exsting KG-Aware Recommendation

현재 존재하는 방식 2 : Path-based Method

- 추가적인 가이드를 제공하기 위해 지식 그래프 내 다양한 연결 패턴을 탐색함

1. Personalized Entity Recommendation (Xiao Yu et al., 2014)
   - 사용자 별로 다른 관계 정보를을 결합하여 고품질의 개인화 추천 결과를 제공함

2. Meta-Graph Based Recommendation (Huan Zhao et al., 2017)
   - 메타 그래프를 도입하고, matrix factorization과 factorization machine을 사용함.

RippleNet

Exsting KG-Aware Recommendation

현재 존재하는 방식 2 : Path-based Method

- 추가적인 가이드를 제공하기 위해 지식 그래프 내 다양한 연결 패턴을 탐색함

Path-based Method는 자연스럽고 직관적으로 지식 그래프를 이용할 수 있음.
하지만,

1. 수동적으로 설계된 경로에 크게 의존하기에 최적화하기가 어려움
2. 엔티티와의 관계가 여러개인 경우에는 수작업으로 메타 경로를 설계할 수는 없음

RippleNet

이 논문에서 제안하는 것

RippleNet : end-to-end framework for KG-aware recommendation

- (사용자, 아이템) 쌍을 입력으로 받아 클릭률을 예측하기 위함
  - 클릭률 : 사용자가 해당 항목과 상호작용할 확률을 출력하기
- Preference Propagation
  - 사용자의 과거 관심사를 시드로 하여 지식 그래프를 통해 사용자의 관심사를 반복적으로 확장함.
  - 이러한 모습은 "빗방울로 인해 물 위로 퍼지는 잔물결"로 비유되기도 함
  - 이러한 잔물결들이 중첩되면서 지식 그래프를 통한 사용자 선호도 분포를 형성함.

RippleNet

이 논문에서 제안하는 것

RippleNet : end-to-end framework for KG-aware recommendation

- **RippleNet의 장점**
  - **임베딩 기반 방법에서의 'KGE'를 선호도 전파를 통해 추천에 자연스럽게 통합함.**
  - **경로 기반 방법에서의 한계점인 '수작업'을 사용자에서 후보까지의 경로를 자동으로 발견할 수 있도록 개선함.**

RippleNet

이 논문에서 제안하는 것

RippleNet : end-to-end framework for KG-aware recommendation

- 요약
  - 임베딩 기반 방법과 경로 기반 방법을 결합함
  - 자동적/반복적으로 사용자의 계층적 잠재 관심사를 발견하고, 선호도를 확장함
  - 영화, 책, 뉴스 추천에서 좋은 성능을 보임 (AUC로 평가)
    - 영화 : 2% -> 40.6%
    - 책 : 2.5 -> 17.4%
    - 뉴스 : 2.6 -> 22.4%

RippleNet

PROBLEM FORMULATION

user-item interaction matrix Y에 대하여

$$y_{uv} = \begin{cases} 1, & \text{if interaction } (u, v) \text{ is observed;} \\ 0, & \text{otherwise.} \end{cases}$$

u : 유저 집합의 원소
v : 아이템 집합의 원소

RippleNet

PROBLEM FORMULATION

y_uv의 값이 1이면 u와 v 사이에 implicit interaction이 있음
(클릭하기, 시청하기 등)

또한, head-relation-tail 구조로 되어 있는 지식 그래프 G = (h, r, t)또한 존재함
ex) (쥬라기 공원, film.film.director, Steven Spielberg)

RippleNet

PROBLEM FORMULATION

목표

- Interaction matrix Y와 Knowledge Graph G를 이용해서
- 사용자 u가 이전에 상호작용 한 적 없는 항목 v에 대해
- 잠재적인 관심이 있는지 예측하기

=> Function : (y^)_uv = F(u, v, theta)

- (y^)_uv : 사용자 u가 항목 v를 클릭할 확률
- theta : 함수 F의 모델 파라미터

RippleNet

RippleNet의 Framework

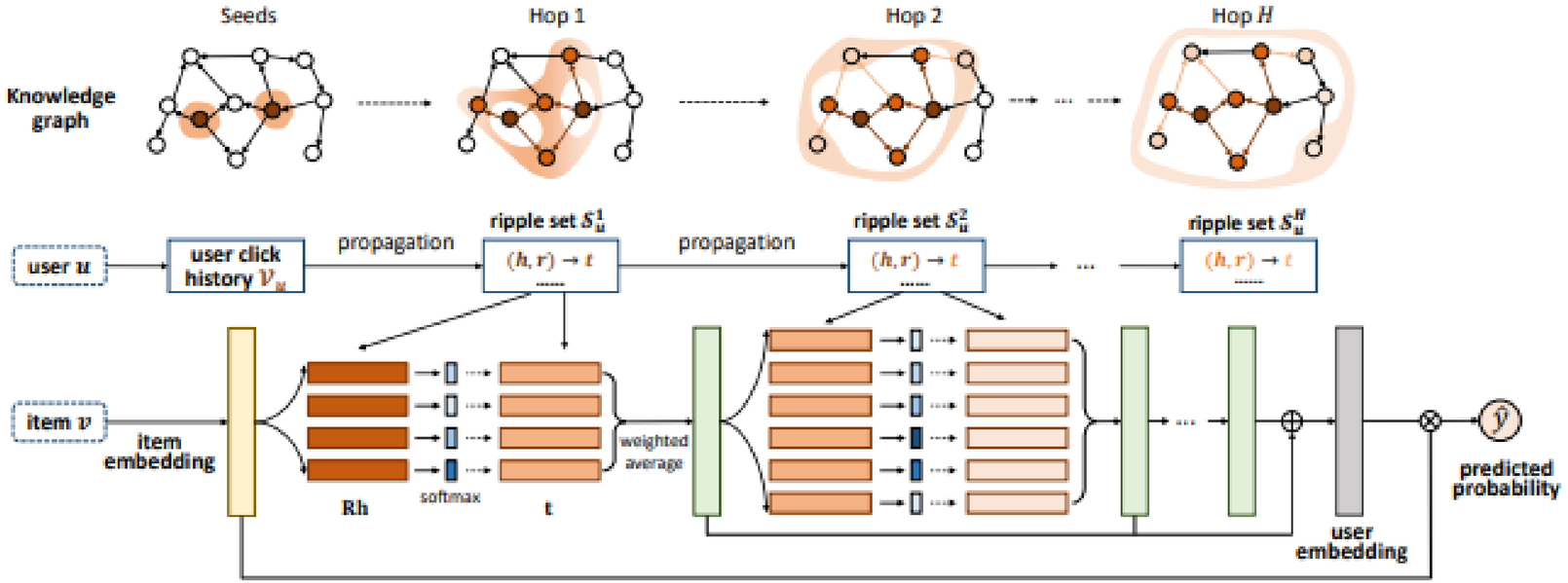


Figure 2: The overall framework of the RippleNet. It takes one user and one item as input, and outputs the predicted probability that the user will click the item. The KGs in the upper part illustrate the corresponding ripple sets activated by the user's click history.

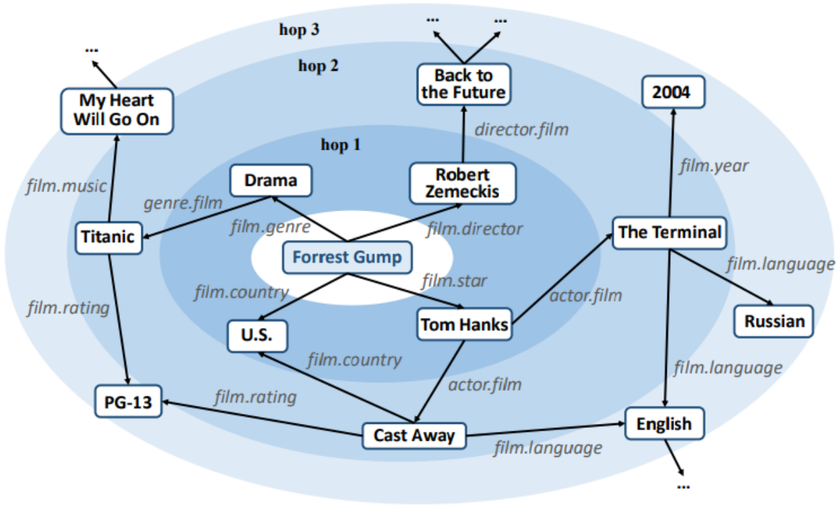
RippleNet의 Framework

1. 사용자 u와 항목 v를 입력으로 받아서 클릭률을 출력함.
2. 사용자 u의 경우 과거 관심사 V_u를 seed로 사용함.
3. seed가 된 V_u는 확장되어 S_uk를 형성함.
   - S_uk는 seed인 V_u로부터 k번 hop한 Ripple Set임.
   - 이 Ripple set은 item embedding과 상호작용 해서 item에 대한 사용자의 응답을 얻어내고, 이를 결합하여 최종 user embedding을 형성하도록 도움.
4. user embedding과 item embedding을 사용하여 y_uv를 계산함.

RippleNet

Ripple Set

지식 그래프는 일반적으로 유의미한 사실과 개체간의 연결을 포함함.



RippleNet

Ripple Set

지식 그래프는 일반적으로 유의미한 사실과 개체간의 연결을 포함함.

- 이러한 연결은 사용자 선호도를 탐색할 있는 깊고 잠재적인 관점을 제공함.
- 이것을 KG의 관점에서 보면,

$$\mathcal{E}_u^k = \{t \mid (h,r,t) \in \mathcal{G} \text{ and } h \in \mathcal{E}_u^{k-1}\}, \quad k = 1, 2, ..., H,$$

$$\mathcal{S}_u^k = \{(h,r,t) \mid (h,r,t) \in \mathcal{G} \text{ and } h \in \mathcal{E}_u^{k-1}\}, \quad k = 1, 2, ..., H.$$

(u : 사용자, ε : V_u = 사용자가 이전에 클릭한 아이템들)

Ripple Set에 대한 우려

"홉 수가 증가하면서 리플 세트의 크기가 너무 커지는 거 아니야?"

=> 해결
(1) 대부분의 entity는 sink entity로, 들어오기만 하고 나가지는 않음.
(2) 특정 시나리오에서는 관계를 제한하여 크기는 줄이고 관계성을 개선하기도 함.
(3) 너무 멀어지게 되면 신호보다 노이즈가 더 커서 커지는 데 한계가 생김.

RippleNet

Preference Propagation

- 기존 모델 : 사용자와 아이템 간의 잠재적 표현을 학습한 후 직접 적용하여 알려지지 않은 평점을 예측함.
- RippleNet : 보다 세밀하게 모델링하기 위해 Preference Propagation을 도입함.
  - Item V은 Item Embedding v ∈ R_d와 연관됨. (d는 임베딩의 차원)
  - Item Embedding은 시나리오에 따라 ID, 속성, 단어 모음, 맥락 등을 포함함.
  - (h_i, r_i, t_i)는 아래의 식과 비교하여 relavance probability를 가짐.

$$p_i = \text{softmax}\left(\mathbf{v}^{\mathrm{T}}\mathbf{R}_i\mathbf{h}_i\right) = \frac{\exp\left(\mathbf{v}^{\mathrm{T}}\mathbf{R}_i\mathbf{h}_i\right)}{\sum_{(h,r,t)\in\mathcal{S}_u^1}\exp\left(\mathbf{v}^{\mathrm{T}}\mathbf{R}\mathbf{h}\right)},$$

RippleNet

Preference Propagation

- 이를 바탕으로 relavance probability에 해당되는 가중치가 있는 S_1u의 tail값들을 더해서 o_u1를 가짐.

$$o_u^1 = \sum_{(h_i, r_i, t_i) \in S_u^1} p_i t_i,$$

- Relavace Probability p와 벡터 o를 통해 user의 관심은 V에서 ε로 전달됨.

=> 이것을 preference propagation이라고 함.

- 사용자 u의 아이템 v에 대한 임베딩은 이 o 값들의 합으로 계산이 됨.

RippleNet

Preference Propagation

- 마지막으로 user embedding과 item embedding을 합산하여 예측 클릭 확률을 출력함.

$$\hat{y}_{uv} = \sigma(\mathbf{u}^{\mathrm{T}}\mathbf{v}),$$

where $\sigma(x) = \frac{1}{1+\exp(-x)}$ is the sigmoid function.

RippleNet

Learning Algorithm

- 이제 u와 v값을 아니까 theta 값을 최적화해서 (y^)_uv를 최대화 해야 함.

$$p(\Theta|\mathcal{G},\mathbf{Y}) = \frac{p(\Theta,\mathcal{G},\mathbf{Y})}{p(\mathcal{G},\mathbf{Y})} \propto p(\Theta)\cdot p(\mathcal{G}|\Theta)\cdot p(\mathbf{Y}|\Theta,\mathcal{G})$$

Learning Algorithm

- 이 뒤 내용은 tensor을 이용한 복잡한 수학 계산이므로 패스

- 딥러닝하는 것처럼 매니배치, 샘플링, 역전파의 과정을 거침

RippleNet

과제

Discussion ~ Result 요약해보기

https://arxiv.org/pdf/1803.03467.pdf

THANK YOU

RIPPLENET

가천대학교 이다