# Recurrent Neural Networks with Top-k Gains for Session-based Recommendations

| | |
|---|---|
| Property | |
| Property 1 | |
| abstract | RNN은 특히 유저의 세션이 있는 순차적인 데이터에서 좋은 모델을 보여줘 왔다. RNN의 사용은 고정적인 session-based 추천모델보다 인상적인 성능을 제공했다. 우리는 새로운 Ranking loss functions을 소개한다. |
| 기타 | |
| 논문 링크 | https://paperswithcode.com/paper/recurrent-neural-networks-with-top-k-gains |
| 발제자 | 김성주 |
| 발표자 | 석우 강 |
| 키워드 | Rnn, loss function, ranking, session-based recommendation |
| 한글 링크 | |

## Abstract

RNN은 세션 기반으로 한 데이터와 연속적인 데이터에서 훌륭한 성능을 보였다. 세션 기반 추천에서 RNN을 사용해서 기존 방법에 비해 굉장한 성능 향상을 보였다. 이 논문에서는 추천 환경에서 RNN을 사용할 때를 맞춤으로 만든 새로운 Ranking Loss 함수를 소개한다.  그리고 다른 모든 대안으로 사용할 수 있는 방법 들과 성능 비교를 했다. ( 거의 사기 수준인데 ? 35% , 20%, 53%). Data Augmentation 방법으로 성능 향상하는 것과 다르게 시간이 많이 늘어난 것도 아니다. A/B 테스트를 사용해서 성능이 개선 됨을 추가 증명했다.

# Keywords

RNN, Loss, Ranking, Session-Based Recommendation, Recommender System

# 1. Introduction

세션 기반 환경에서는, 과거 유저 히스토리 로그 들을 사용할 수 없는 경우가 있다. ( 신규, 비 로그인, 그 밖에 추적 못하는 경우 ) 그래서 추천 시스템이 현재 세션 내 유저 행동에 의존해서 정확한 추천을 제공해야 한다. RNN이 16년도에 세션 기반의 모델에 적용 되어서 엄청난 결과를 보였다.

RNN 방식의 기존 방법 대비 강점은 (클릭, 조회 등등)의 전체 세션에 대해 효율적으로 모델링이 가능한 점이다. RNN은 세션의 테마에 대해 학습이 가능해서 기존 방법 대비 정확도가 20-30%가 상승된다.

주요 목적은 유저 선호도로 아이템의 Rank를 매기는 것. 유저가 좋아할 것들을 상위로 Rank 하는 것은 중요하다. 그래서 Learning To Rank과 Ranking Objectives 와 Loss Function을 이용한다. Ranking Loss를 잘 선정하는 것은 성능에 많은 영향을 끼친다. 추천 시스템의 특성 상 output space가 크기 때문에 ( 추천할 아이템이 많아서 ) 적절한 ranking loss 함수를 만드는 건 어려운 문제다.

여기서 성능 측정으로 MRR(Mean Reciprocal Rank)와 Recall at 20을 사용한다.

▼ MRR

주어지는 Query에 대해 그 Query와 정확히 일치 하는게 몇 번째에 있는가?

정확도가 높으면서, 가능한 적은, 상위 랭크 된 검색 결과를 원하는 상태에서 적합성 평가 척도로 활용됨

$$MRR = \frac{1}{|Q|}\sum_{i=1}^{|Q|}\frac{1}{rank_i}$$

## 1.1 Related Work

세션 기반 추천에서 user-profile이 없는 경우를 대비해서 item-to-item 추천을 사용 했었다. item-to-item 유사도 매트릭스는 가능한 세션 데이터에서 미리 생성되고 유저가 현재 클릭한 상품의 가장 유사한 상품을 추천 했다.

RNN은 User-Item CF 에서 user와 item 의 Factor의 시간에 따른 변화를 모델링 하기 위해서 사용 되었는데, single domain에서는 큰 의미는 없다는 걸로 결론이 났다.

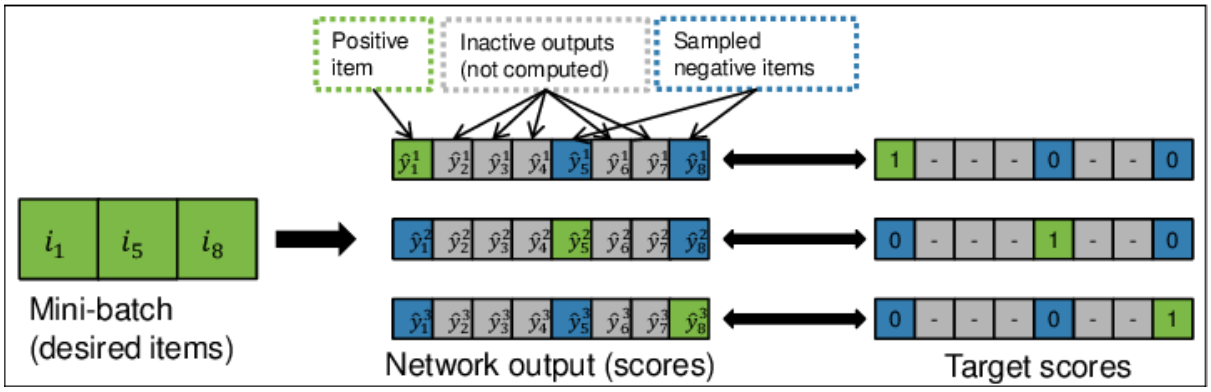
Matrix Factorization에서 사용되던 Ranking Loss Function가 RNN에서 효과를 낼 것을 보장 할 수 없다. 변경이 필요하다.

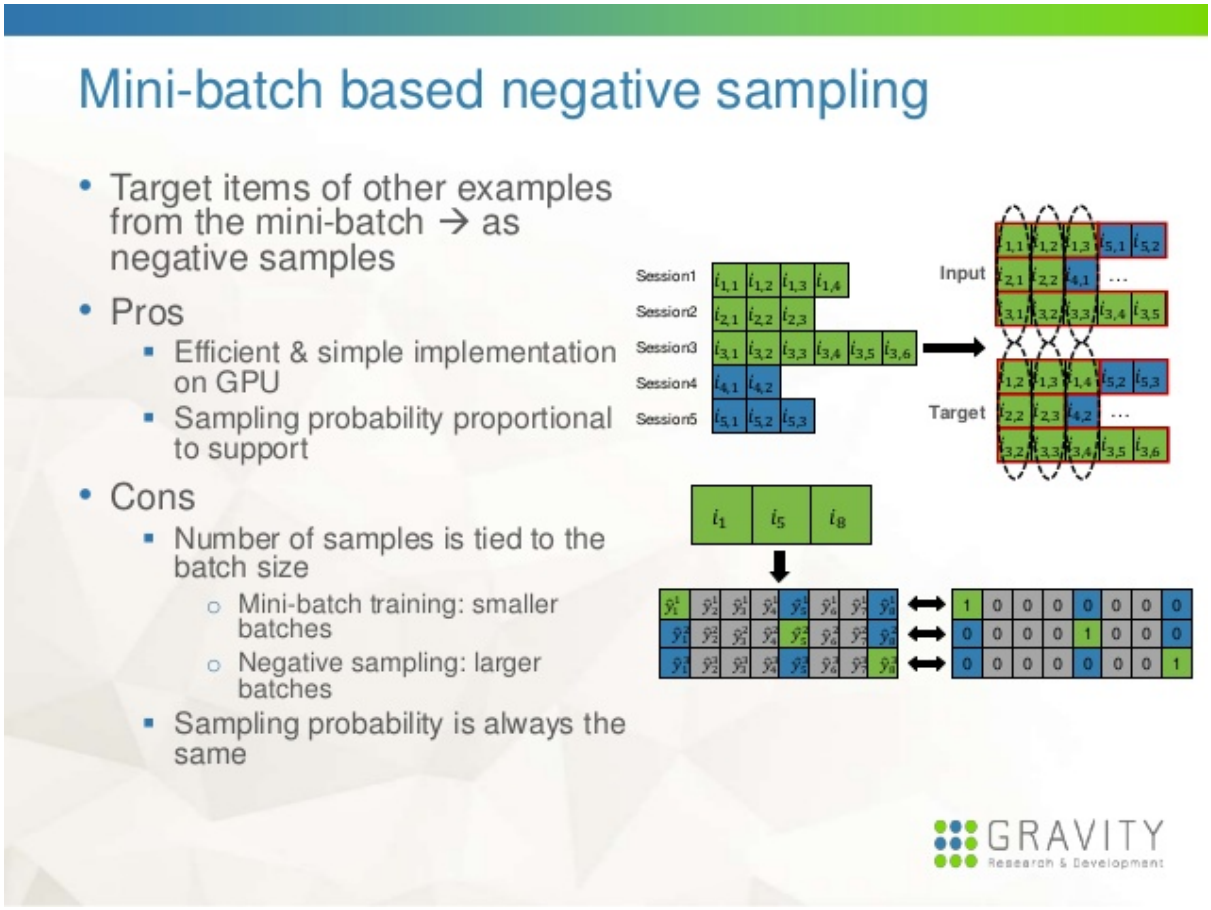
black out 메소드 라고  language model에서 사용하는 sampling 방법이 있는데 효율이 괜찮다.

# 2. Sampling the output

GRU4Rec 의 Negative Feedback을 샘플링 하는 것은 성능에 굉장히 중요해서 재조명했음. 모든 아이템에 대해서 점수를 계산하는 것은 비효율 적이고 모델을 확장 불가능하게 한다. 그래서 GRU4Rec은 학습에서 샘플링 메커니즘을 사용하고 샘플링된 결과 만 학습에 사용했다.

GRU4REC은 mini-batch 기반의 Sampling을 사용한다. GPU들을 사용할 때 굉장히 효율적이다






Ranking loss의 한 특성은 학습이 오직 타겟 상품이 Negative Samples의 점수 들에 큰 마진으로 초과하지 않는 경우에만 이루어지며, 그렇지 않으면, 아이템들이 바른 순서로 되어 있어서 학습할 것이 없다. 그래서 샘플링 과정을 사용하면, 높은 스코어의 아이템들이 Negative Sample에 대해 포함 되어야 한다. uniform sampling에 비해서는 popularity-based sampling이 더 좋은 결과를 낸다.

미니 배치 기반으로 Sampling을 하기 때문에 배치 사이즈가 줄어들면 정확도가 올라가지만 GPU를 사용하면 배치 사이즈가 크면 좋으니 잘 선택해야 한다.

어짜피 GPU 를 병렬로 사용하기 때문에 Sampling을 통한 Data Augmentation 된 데이터들의 계산은 문제가 없는데, 미니-배치 기반의 Sampling은 GPU에 인터럽트를 걸 수 있어서 캐시를 두어서 Negative Sampling을 만들어 둔다.

# Loss Function Design

Cross Entropy Loss의 수적인 불안정을 안정화 하기 위한 두가지 방법을 제안한다

## 3.1 Categorical Cross-Entroy

타겟 분포 p와 제시된 확률 분포  q와의 거리를 측정한 것이고 아래와 같음

$H(p, q) = -\sum_{j=1}^{N} p_j \log q_j$

다중 분류 문제에 사용이 주로 된다. cross entroy with softmax scores

## 3.2 Ranking Losses : Top1 & BPR

$L_{top1} = \frac{1}{N_s} \sum_{j=1}^{N_s} \sigma(r_j - r_i) + \sigma(r_j^2)$

j는 $N_s$개의 sampled negative (관련성 없는) 아이템들에 대해서 동작시키는 것이고, 관련성 있는 아이템들은 i로 인덱싱된다.

$L_{bpr} = \frac{-1}{N_s} \sum_{j=1}^{N_s} log\sigma(r_i - r_j)$

log probability of the target score

## 3.3 Ranking-max loss function familty

샘플의 수가 증가하면서, vanishing gradient 문제가 있는데, 극복하기 위해서 pairwise 가 아닌 새로운 류의 list-wise loss function 을 제시한다. individual pairwise losses 를 기반 한다.