DRN

강화 학습 기반 뉴스 추천 + DQN

> https://s3-us-west-2.amazonaws.com/secure.notion-static.com/6a9d1d89-7a27-4cd0-a460-5cba51f7cd21/dqn.pdf

[추천시스템]DRN: A Deep Reinforcement Learning Framework for News Recommendation

논문 링크
http://www.personal.psu.edu/~gjz5038/paper/www2018_reinforceRec/www2018_reinforceRec.pdf
https://www.dropbox.com/s/8uyo6vt7isvq7dk/DRN. A Deep Reinforcement Learning Framework for News Recommendation.pdf?dl=0

RL 발전과정 간략히 훑어보기

기 모델의 문제점 + 본 모델의 기여
기존 추천 모델들은 크게 다음의 세 가지 문제점을 가지고 있었다. (그리고 우리 모델이 이 문제를 해결했다)
1) 지속적으로 변화하는 개인화 뉴스 추천 상황적 특성을 제대로 고려하지 못함
개인화 뉴스 추천 서비스는 다음의 특성을 지닌다.

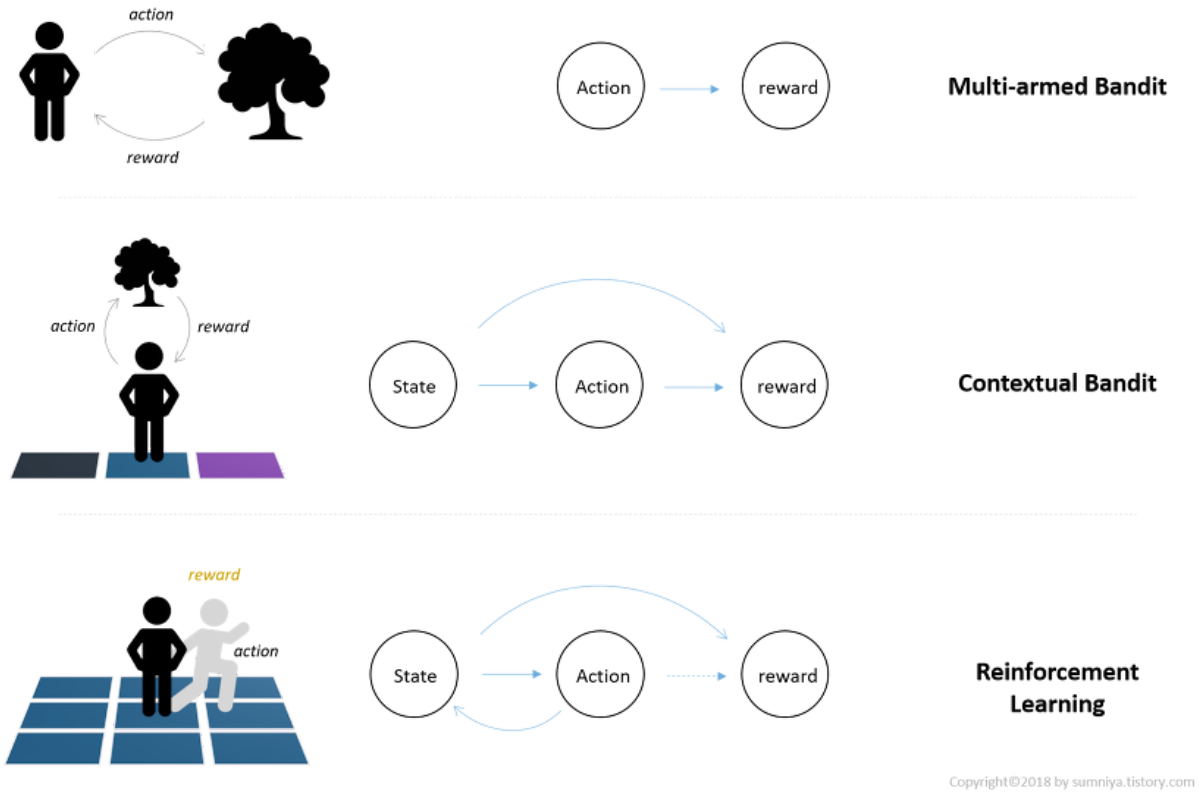
뉴스는 아주 빨리 outdated된다. 즉 뉴스 feature와 candidate set이 빠르게 변화한다.
기사 발행 후 마지막 사람이 보기까지 걸린 시간은 평균 시간은 4.1 시간에 불과

동일 사용자 interest도 시간이 지나면서 점진적으로 변한다(evolve).

예) 위 그림은 한 유저의 시간이 지나면서 관심 뉴스 분야 변화를 살펴본 것이다. 첫 몇 주는 정치에 관심있다가 점차적으로 연예와 tech로 관심사가 옮겨간다.

이에 따라 계속해서 변하는 뉴스 feature과 candidate set, user interest를 반영하여 주기적으로 모델을 업데이트 하려는 online recommendation methods 시도들이 있었다.
하지만 MAB(Multi Armed Bandit. 보다 정확히는 Contextual Bandits를 의미) 기반 모델들은 current reward(즉 CTR)에 대해서만 optimize하면서, future reward, 즉 이번 추천이 가져올 미래에 대해서는 explicit하게 고려하지 못한다는 단점을 지닌다.
예) 유저A가 현재 재난 뉴스를 클릭할 확률과 농구 뉴스를 클릭할 확률 동일하다고 하더라도, 재난 뉴스를 본 후 추가로 뉴스를 볼 확률보다 농구 뉴스를 본 후 추가로 뉴스를 볼 확률이 높다고 판단된다면, 후자를 추천해야 한다.
Contextual Bandit에서 state(bandit)는 고정된 것으로 가정. 하지만 MDP에서는 agent가 취하는 action에 따라 state가 변한다고 가정함. 이를 통해 future reward를 명시적으로 계산 → 정확도 향상
반면 MDP(Markov Decision Process) 기반 모델들은 MAB와 달리 future reward는 고려했지만 discrete user log를 그대로 이용하여 scale-up이 어려움
item candidate set이 커지면 state space size가 exponentially 증가하여 이를 모두 저장하여 사용하기 어려움
→ 'DQN(Deep Q-Learning)' 기반 추천 프레임워크를 통해 1. 현재와 미래 reward를 동시에 고려하고, 2. user를 나타내기 위해 continuous state representation와 News list(items)를 나타내기 위해 continuous action representation을 multi-layer DQN의 입력값으로 사용하기 때문에 scale-up 가능

[Ch.2] Markov Decision Process



2) user feedback으로 click 여부나 rating만 고려
특히 CTR은 사용자가 실수로 잘못 누른 기사와 1시간 가량을 쓴 기사에 대한 reward가 동일해 추천에 대한 만족도를 제대로 반영하지 못하고 있다.

→ CTR에 더해 'activeness'라는 사용자가 앱 사용 후 다시 메인에 돌아올 확률 feedback을 추가한다.
activeness: 사용자가 다시 서비스로 돌아올 확률

가장 최근 return interval만 고려하는게 아닌 multiple historical return interval 정보를 활용
본 모델은 user activeness를 실제로 유저가 돌아온 순간 뿐만 아니라 항상 이 값을 추정할 수 있게함

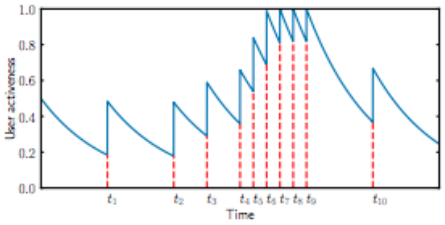


Figure 5: User activeness estimation

$$S(t) = e^{-\int_0^t \lambda(x)dx} \qquad (4)$$

처음은 모든 유저의 activeness는 0.5

시간이 지나면서 식(4)에 따라 decay되다가 유저가 t1에 접속하면 Sa만큼 activeness 값을 더함

위를 반복

Sa 등은 기 데이터를 통해 정함: 0.32

3) 추천 아이템의 낮은 다양성
이 문제 해결을 위해 기존 강화학습 모델에서도 randomness(즉 exploration)를 더하는 simple ϵ-greedy strategy나 UCB(Upper Confidence Bound)를 적용했다. 하지만 입실론-그리디는 전혀 관련 없는 아이템을 추천할 가능성이 있고, UCB는 해당 아이템을 몇 번 추천해 데이터를 확보하기 전에는 정확한 reward를 추정하기 어렵다(cold start)
→ 'Dueling Bandit Gradient Descent'라는 효과적인 exploration 전략을 적용한다.
뉴스 추천 시스템을 강화 학습 모델로 이해하기
이해

Environment: user pool and news pool
agent: our recommendation algorithms
state: feature representation for users
action: feature representation for news

Table 1: Notations

Notation	Meaning
G	Agent
u, U	User, User set
a	Action
s	State
r	Reward
i, I	News, Candidate news pool
L	List of news to recommend
B	List of feedback from users
Q	Deep Q-Network
W	Parameters of Deep Q-Network

Deep Q-network

user가 추천을 요청하면 user 특성(state)과 뉴스 리스트(a set of action)를 agent에 전달
agent는 추천 뉴스목록(best action)을 추출 후 추천
fetch user feedback as reward(click labels and estimation of user activeness)
All these recommendation and feedback log will be stored in the memory of the agent. 매 시간 the agent will use the log in the memory to update its recommendation algorithm

Model framework
모델은 크게 offline 부분과 online 부분으로 구성되어 있다. offline part는 사전에 학습 시켜서 가중치를 적용해놓은 모델이고, online 방식은 reward가 들어오면 순차적으로 학습시켜서 가중치를 업데이트하는 과정을 지칭한다.(본 논문에서는 기존 데이터로 사전 학습 시키고-offline, 진짜 유저의 실시간 feedback을 토대로 학습해나가는-online 모델을 사용한다.)
News와 User로 부터 추출한 4가지(News, User, User news, Context) feature 이용

state

User features

해당 유저의 특정 시간 단위별 뉴스 요소별 클릭 수

last 1 hour, 6 hours, 24 hours, 1 week, and 1 year

413 × 5 = 2065 dimensions.

Context features

time, weekday, and the freshness of the news (the gap between request time and news publish time)
32-dimensional features

action

News features

뉴스가 다음의 요소를 가지고 있는지 여부를 417 dimension one hot features로 나타냄

headline, provider, ranking, entity name, category, topic category property

User news features: 유저와 특정 뉴스간의 interaction

the frequency for the entity (also category, topic category and provider) to appear in the history of the user’s readings.
25-dimensional features

Offline Part

4가지 feature를 통해 reward(a combination of user-news click label and user activeness)를 predict 하면서 학습
"RL 분석에 집중하기 위해 textual 같은 추가 features를 넣지 않았지만 이를 결합하면 더 좋은성능을 보여줄 것이다"

이용 data: offline user-news click logs

offline part는 static한 dataset만 다룬다. 그래서 exploration하는 데 제약이 있을 뿐 아니라 시간에 따른 user activeness도 파악하기 힘들다.

Online Part(실시간)

PUSH: user가 뉴스를 요청하는 매 timestamp마다, agent G는 입력 값으로 current user와 news candidates(현재 추천 받은 리스트와 유사도가 높은 기사를 무작위 추출)의 features를 받고 top-k 추천 뉴스 목록 L을 생성한다(exploitation, exploration 둘 다 고려되어 만들어진 기사 리스트 L)

exploitation
total reward는 다음과 같다.

γ is a discount factor to balance the relative importance of immediate rewards and future rewards

Wt and Wt′ are two different sets of parameters of the DQN. 우선 Wt를 가지고 maximum future reward를 주는 action을 선택한 후 Wt′를 기준으로 다시 계산. 매 iteration마다 Wt and Wt′ 교환
이를 통해 immediate and future situations을 모두 고려한 결정을 내릴 수 있음
하나의 Network를 사용하지 않고 분리된 target network를 둔다는 것은 target을update를 하는 동안 사용되는 parameter를 (이전과 동일하게) 고정한다는 것입니다. 그 이유는, target의 값을 구할 때, 이 parameter의 영향을 크게 받기때문에 보다 안정적으로 계산하기 위해서.
https://sumniya.tistory.com/19?category=781574

Action value function Q-function을 value function V (s)와 advantage function A(s, a) 2가지로 나누어서 각각 계산하고, 마지막에 이를 다시 결합합니다.
https://t1.daumcdn.net/cfile/tistory/994F6B395B9A045532
V(s)는 (state) value function
A(s, a)는 the state와 action 모두를 받아 다른 action을 취했을 때보다 얼마나 더 좋은지를 나타내는 함수 (advantage function)

이렇게 분리함으로써, 특정 state에서 action을 취했을 때 reward는 모든 feature와 밀접하게 관련되어 있음 (A(s, a))과 동시에, user 자신의 특징(예: 이 사용자가 활동 중인지, 이 사용자가 오늘 충분히 뉴스를 읽었는지)에 따라 결정되는 보상은 사용자의 상태와 상황에 따라 더 큰 영향을 받는다(V(s)). 일종의 앙상블?
https://sumniya.tistory.com/19?category=781574

exploration

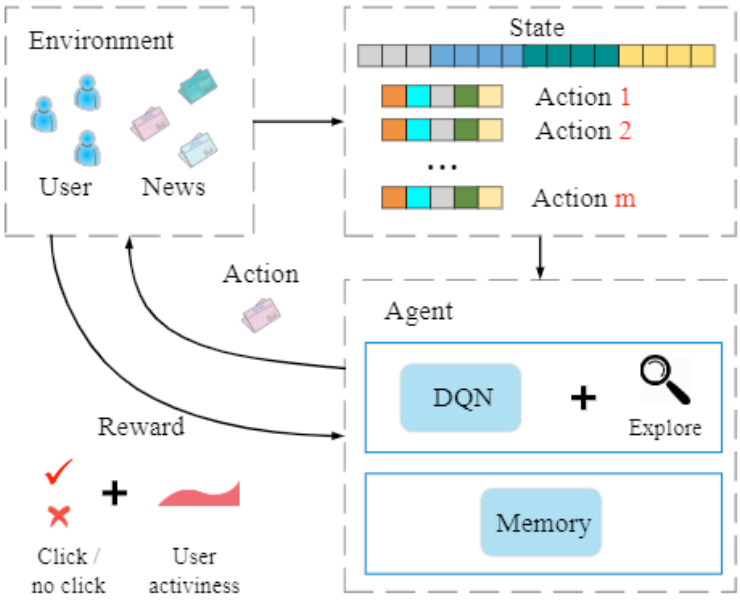


Figure 2: Deep Reinforcement Recommendation System

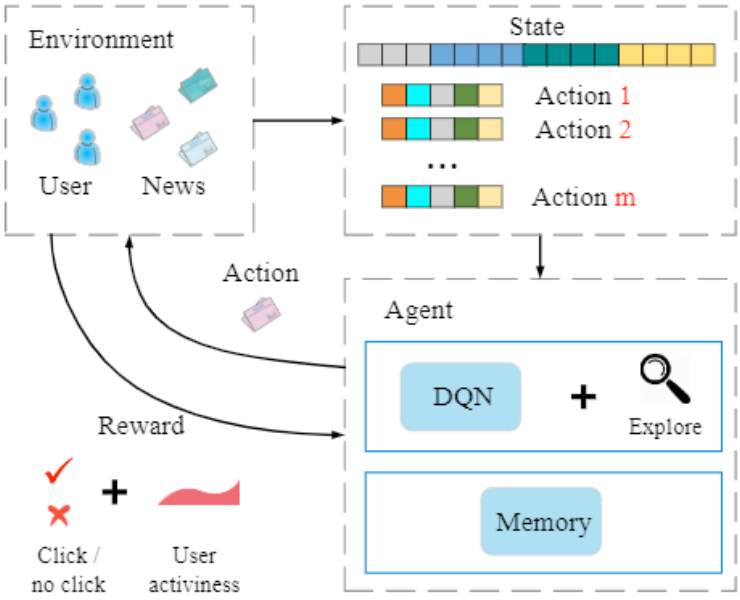


Figure 2: Deep Reinforcement Recommendation System

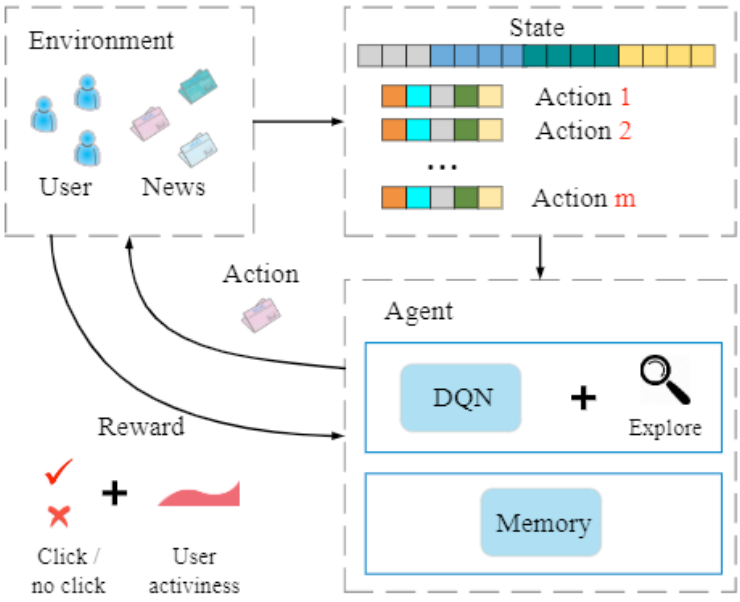


Figure 2: Deep Reinforcement Recommendation System

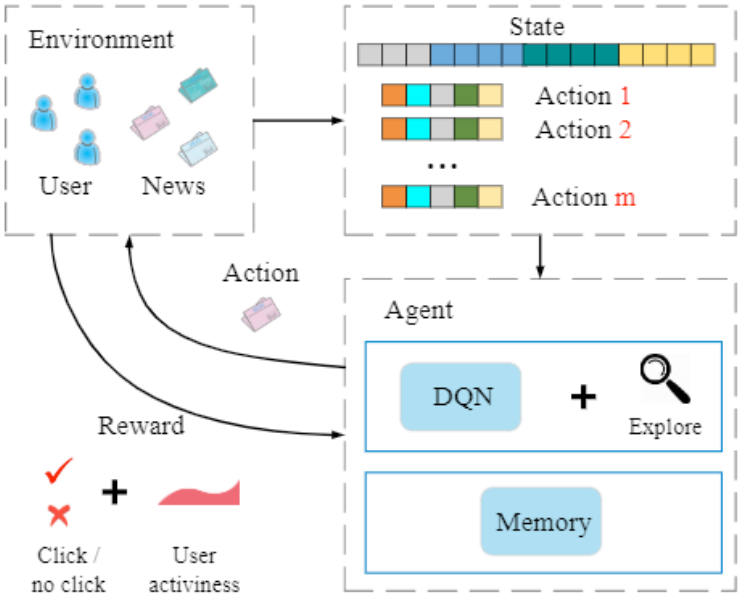


Figure 2: Deep Reinforcement Recommendation System

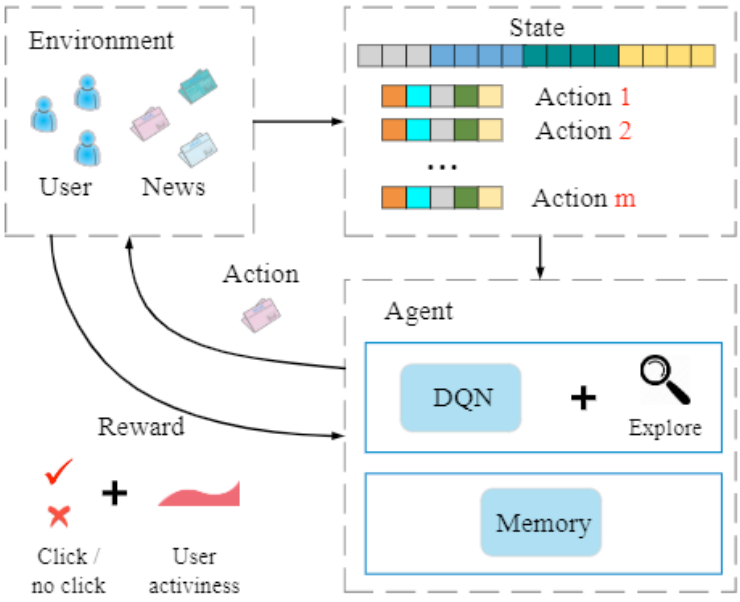


Figure 2: Deep Reinforcement Recommendation System

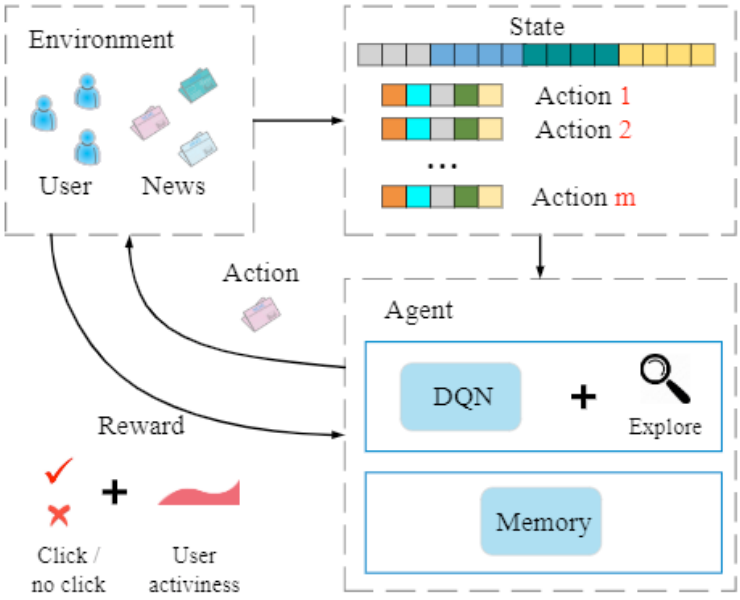


Figure 2: Deep Reinforcement Recommendation System

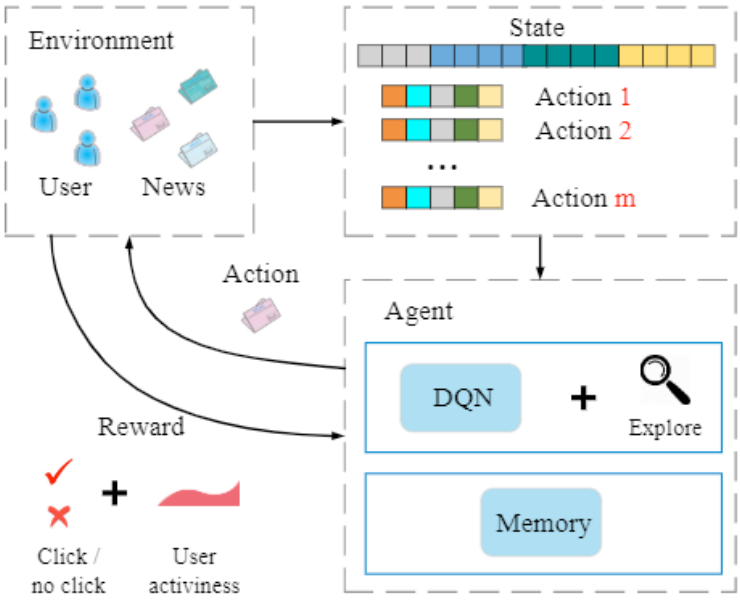


Figure 2: Deep Reinforcement Recommendation System

Table 1: Notations

Notation	Meaning
G	Agent
u, U	User, User set
a	Action
s	State
r	Reward
i, I	News, Candidate news pool
L	List of news to recommend
B	List of feedback from users
Q	Deep Q-Network
W	Parameters of Deep Q-Network

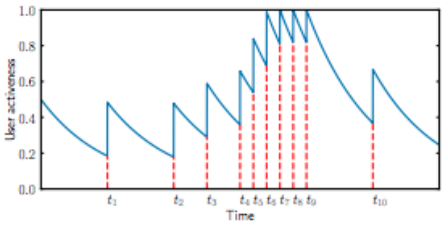


Figure 5: User activeness estimation

$$S(t) = e^{-\int_0^t \lambda(x)dx} \quad (4)$$

처음은 모든 유저의 activeness는 0.5

시간이 지나면서 식(4)에 따라 decay되다가 유저가 t1에 접속하면 Sa만큼 activeness 값을 더함

위를 반복

Sa 등은 기 데이터를 통해 정함: 0.32

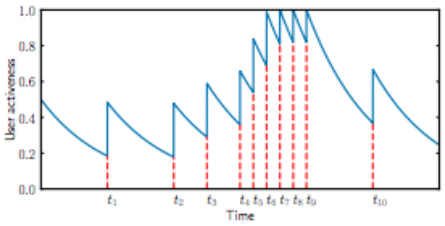


Figure 5: User activeness estimation

$$S(t) = e^{-\int_0^t \lambda(x)dx} \qquad (4)$$

처음은 모든 유저의 activeness는 0.5

시간이 지나면서 식(4)에 따라 decay되다가 유저가 t1에 접속하면 Sa만큼 activeness 값을 더함

위를 반복

Sa 등은 기 데이터를 통해 정함: 0.32

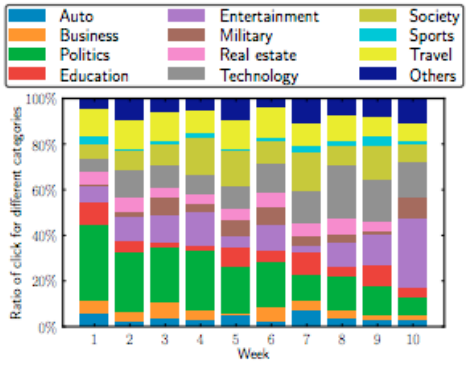


Figure 1: Distribution of clicked categories of an active user in ten weeks. User interest is evolving over time.

current network Q에서 추출한 list L과 explore network Q˜에서 추출한 list L˜을 probabilistic interleave한다.

explore network Q˜는

probabilistic interleave란 우선 랜덤하게 L과 L˜ 중 하나를 선택하고, 거기서 각 item의 확률에 따라(당연히 ranking이 높은 item이 높은 확률을 지님) 선택되어 리스트를 만듦

FEEDBACK: 사용자가 기사 리스트(L)에 있는 각각의 아이템을 click or no click

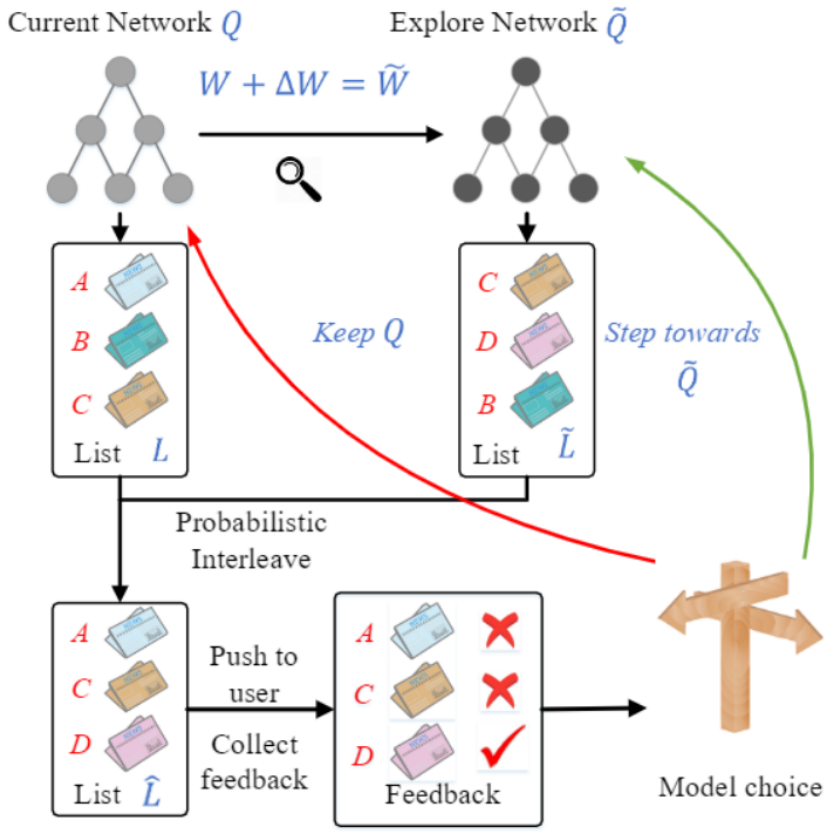
MINOR UPDATE

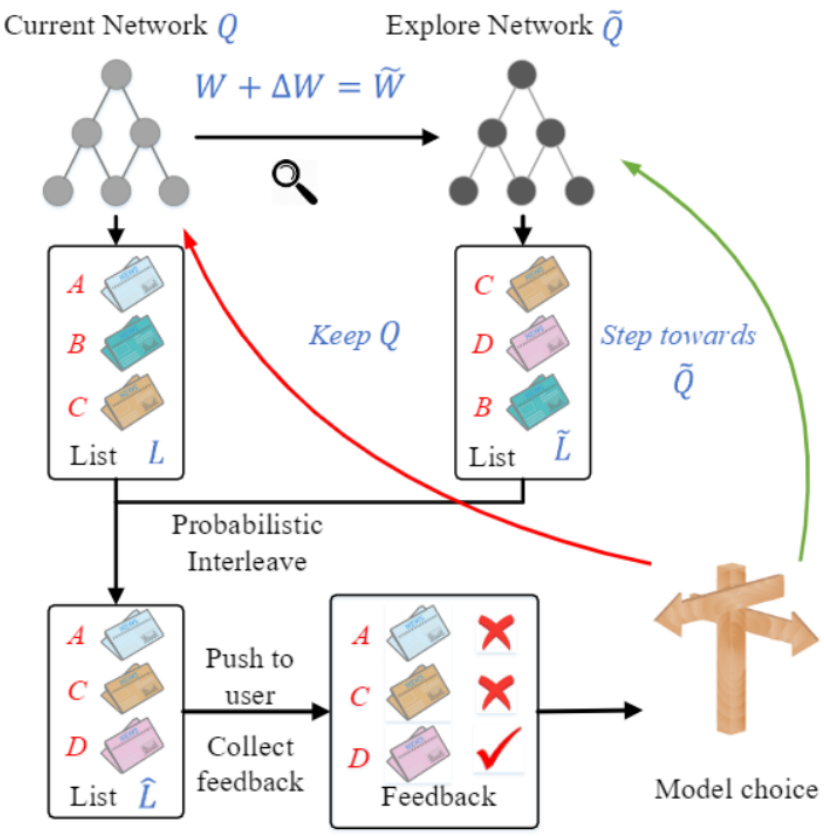
즉 previous user u and news list L, and the feedback B를 가지고, exploration network가 더 좋은 성과를 냈으면 current network를 exploration 쪽으로 업데이트하고, exploitation network가 더 성능이 좋았다면 현 상태를 유지한다.
매 timestamp마다 발생.

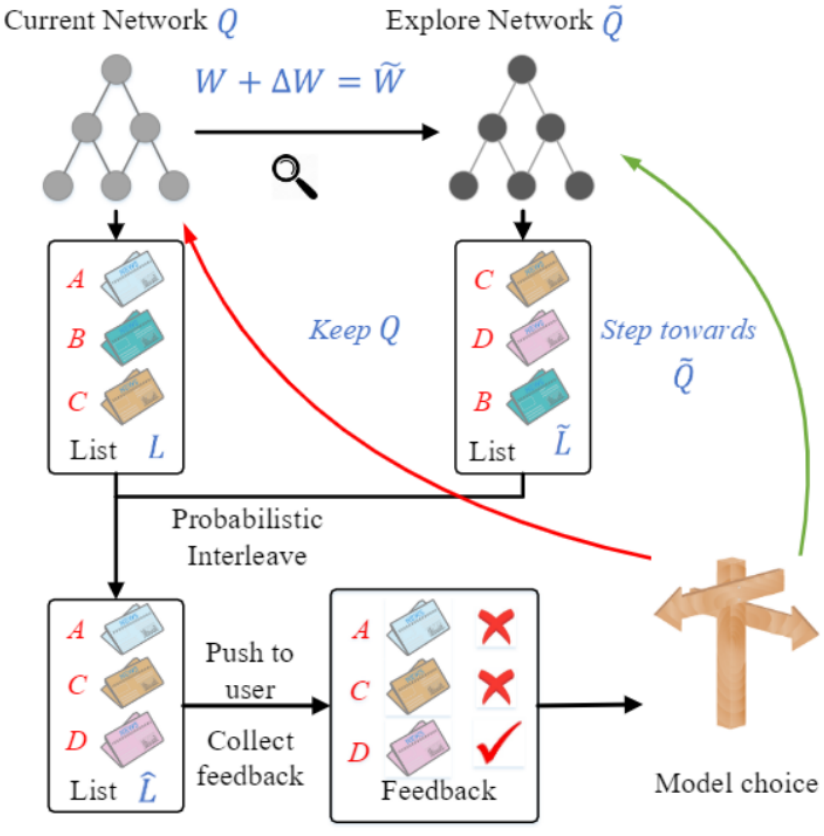
MAJOR UPDATE: experience replay technique

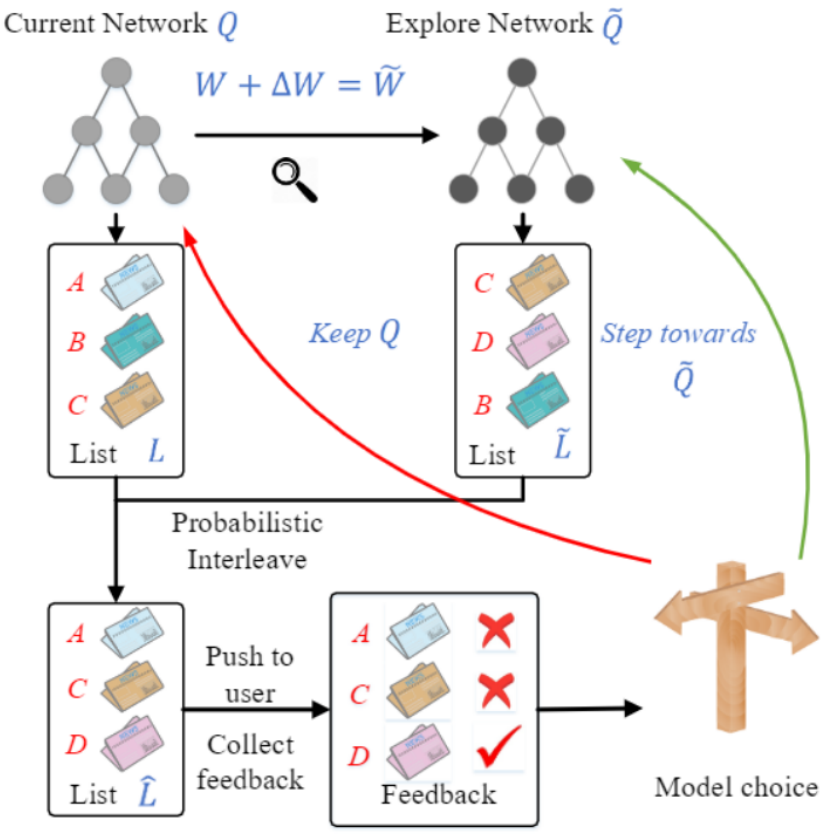
메모리에 저장된 사용자의 최근 click, activeness를 sample a batch of records하여 모델을 업데이트한다. 특정 시간 이후(논문에서는 1시간) 수천번의 recommendation impressions이 일어나 유저 피드백이 저장되면 시행

위 단계들을 반복




Current Network Q
Explore Network $\tilde{Q}$
$W + \Delta W = \tilde{W}$
A
B
C
List L
Keep Q
C
D
B
List $\tilde{L}$
Step towards $\tilde{Q}$
Probabilistic Interleave
A
C
D
List $\hat{L}$
Push to user
Collect feedback
A
C
D
Feedback
Model choice


Current Network Q
Explore Network $\tilde{Q}$
$W + \Delta W = \tilde{W}$
A
B
C
List L
Keep Q
C
D
B
List $\tilde{L}$
Step towards $\tilde{Q}$
Probabilistic Interleave
A
C
D
List $\hat{L}$
Push to user
Collect feedback
A
C
D
Feedback
Model choice


Current Network Q
Explore Network $\tilde{Q}$
$W + \Delta W = \tilde{W}$
A
B
C
List L
Keep Q
C
D
B
List $\tilde{L}$
Step towards $\tilde{Q}$
Probabilistic Interleave
A
C
D
List $\hat{L}$
Push to user
Collect feedback
A
C
D
Feedback
Model choice

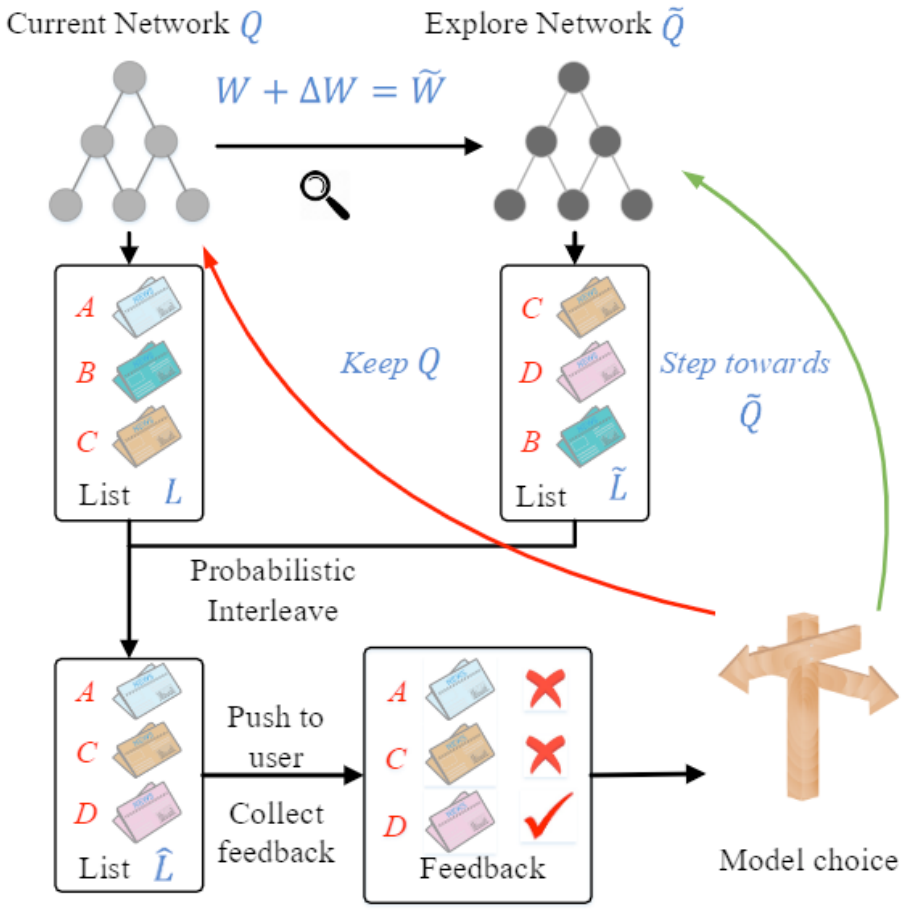


Figure 7: Exploration by *Dueling Bandit Gradient Descent*

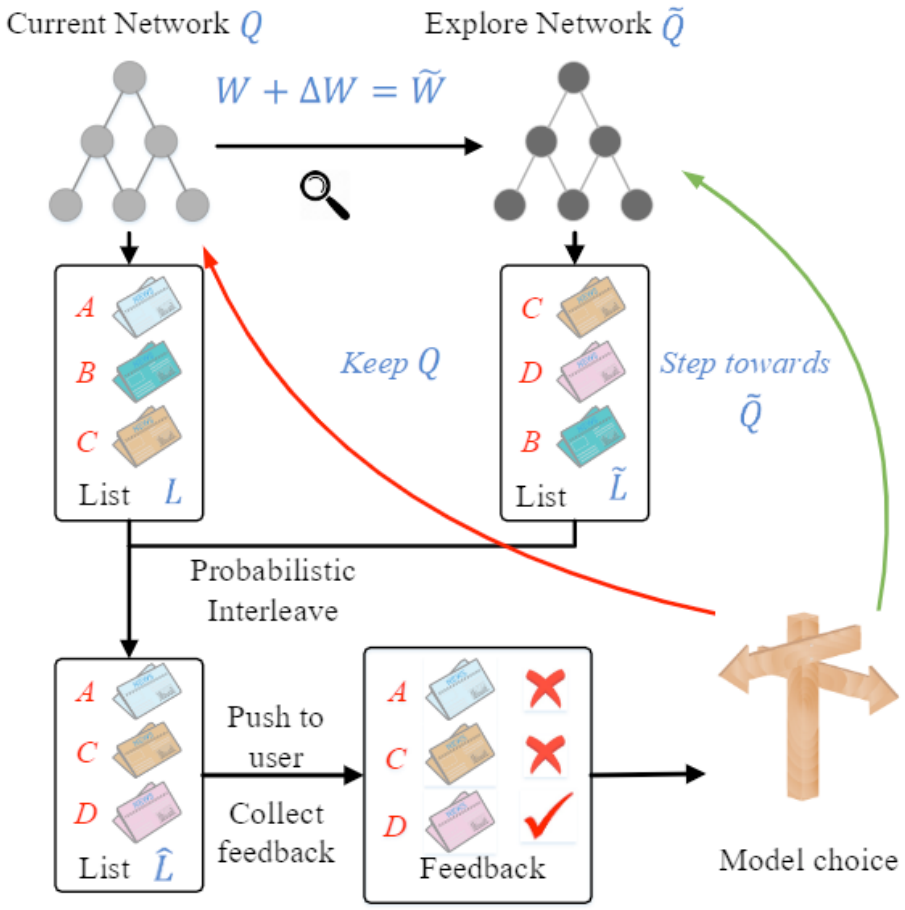


Figure 7: Exploration by *Dueling Bandit Gradient Descent*

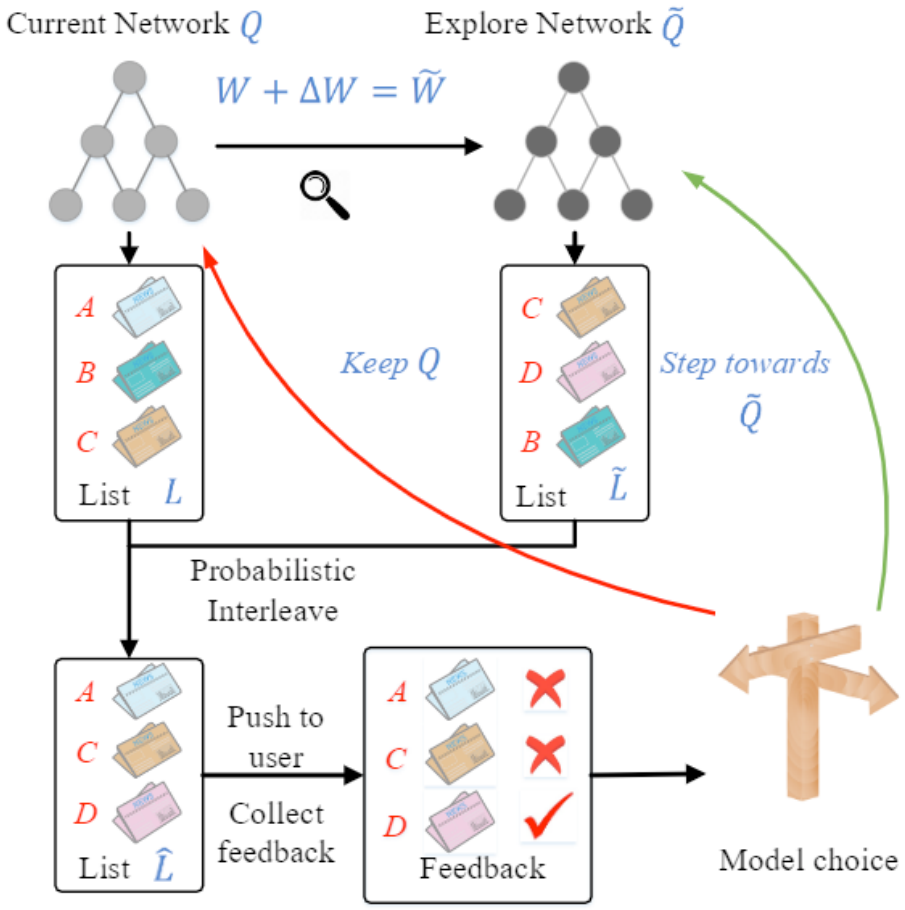


Figure 7: Exploration by *Dueling Bandit Gradient Descent*

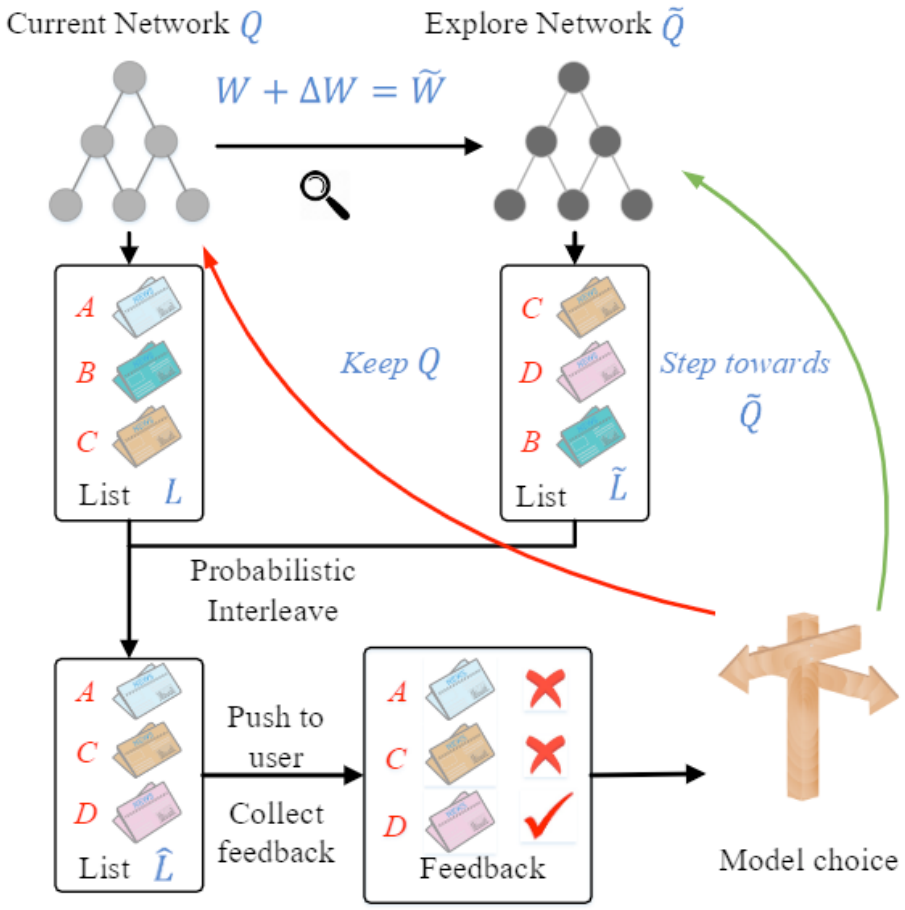


Figure 7: Exploration by *Dueling Bandit Gradient Descent*

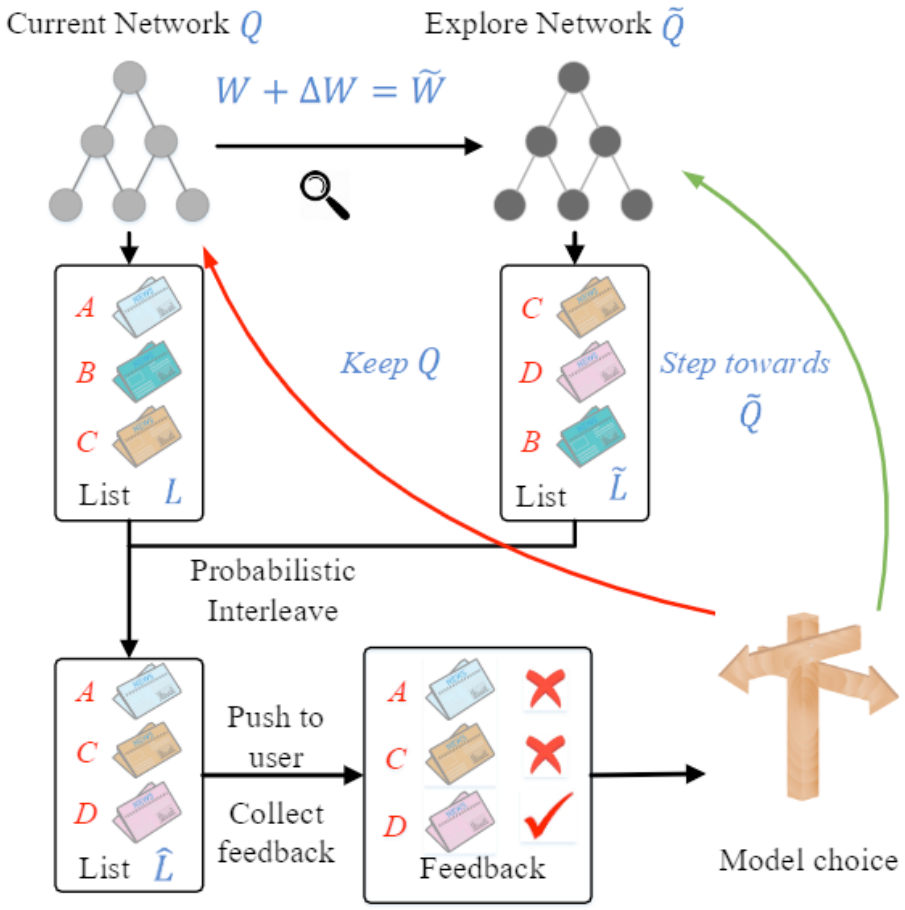


Figure 7: Exploration by *Dueling Bandit Gradient Descent*

모델 이해한 흐름대로 설명하기
(offline, online dataset은 다름!)

user가 뉴스를 요청하는 매 timestamp마다, exploitation, exploration 둘 다 고려하여 top-k 추천 뉴스 목록 L을 생성

offline part에서 높은 score를 갖는 News List를 반환

한 유저에게 candidates라는 기사 셋을 네트워크에 보낸다.

두 개의 모델에 같은 인풋을 적용한다.(push)

Current network라는 모델은 유저가 좋아할 것 같은 k개의 뉴스 리스트를 리턴하고,
Explore network 또한 똑같이 실행한다.

(여기서 Explore network는 current network 모델에서의 가중치 파라미터를 Gaussian Random Noise기법을 통해 업데이트한다. - 아래 Figure 7)

두 모델에서 반환하는 리스트를 합쳐 유저의 reward가 발생할때까지 기다리고,

유저의 reward feedback(click/no click)을 두 리스트들과 비교한다.

만약, explore network의 성능이 더 높다면 다음 step의 current network 파라미터로 업데이트한다.(minor update)

2~6 과정을 major update interval이 다가올때까지 반복한다.

(Major update 할 시기가 되었다.)

위 반복되는 과정 속에서 user activeness에 대한 정보를 memory에 담아오고 있었다. 메모리에 있는 데이터(recent historical click & user activeness records) 중 batch size만큼 샘플링해서 모델을 업데이트한다.

전체 흐름을 유저 수만큼 반복한다.(쉽게 말해서 Timeline의 t는 유저 한 명을 가리킨다.)

테스트 결과

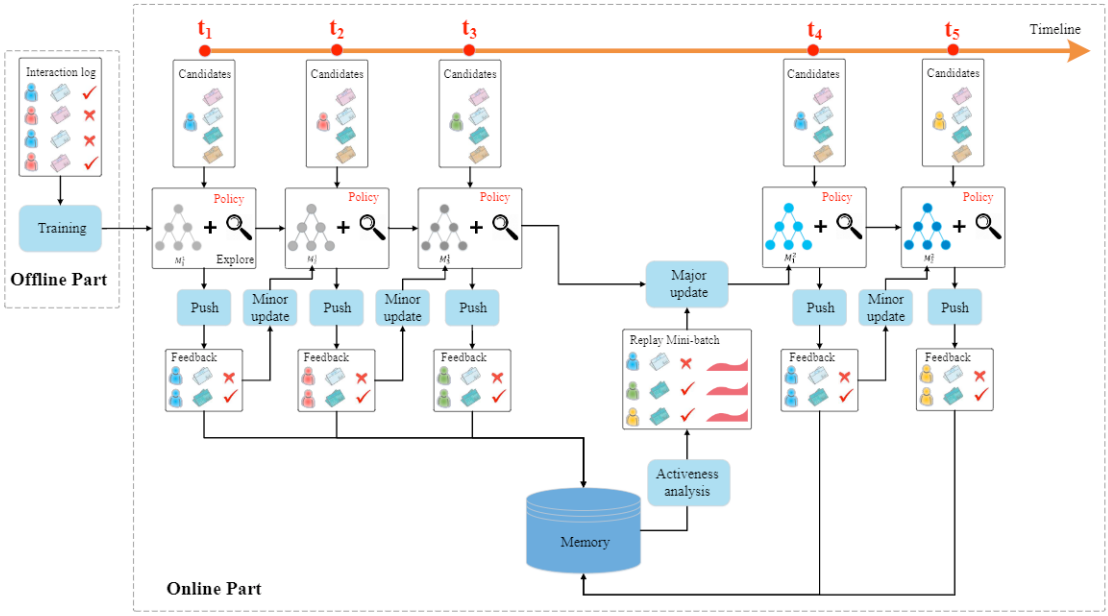


Figure 3: Model framework

Table 3: Parameter setting

Parameter	Setting
Future reward discount γ (Equation 1)	0.4
User activeness coefficient β (Equation 6)	0.05
Explore coefficient α (Equation 7)	0.1
Exploit coefficient η (Equation 8)	0.05
Major update period T_R (for DQN experience replay)	60 minutes
Minor update period T_D (for *DBGD*)	30 minutes

참고자료
숨니의 무작정 따라하기
https://jisoo-coding.tistory.com/27