

T  
S P O R T

# RUNNING COURSE REC SYSTEM

RECSYSTEM TEAM :

이채원, 광도윤, 송채은, 구기현, 조재우





# TABLE OF CONTENTS

- 1 사용자 선호 기반 러닝코스 생성 알고리즘
  - Dijkstra
- 2 사용자-아이템-상호작용 데이터셋 구축
- 3 딥러닝 모델 기반 러닝코스 추천 시스템
  - Two Tower, DCN+DeepFM, KGAT
- 4 평가지표 기반 모델 성능 비교
  - HR@1, HR@10
- 5 의의 및 한계

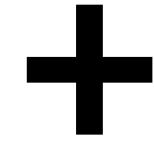
# CHAPTER 1.

사용자 선호도 기반  
러닝 코스 생성 알고리즘:  
**DIJKSTRA ALGORITHM**



# 1-1. 경로 데이터셋 구축

OSMnx 서대문구 보행자 네트워크 + OSMnx 태깅 데이터 + 외부데이터셋 결합(cctv,streetlight,grade\_net)하여 **edge dataset** 구축



# 1-2. 러닝코스 생성 알고리즘

## Dijkstra Algorithm :

출발지에서 특정 목표까지의 "하나의" 최소 **비용(cost)** 경로 탐색.

모든 후보 노드를 대상으로 누적 비용이 가장 적은 경로부터 확장하는 방식.

→ **cost**를 새로 정의하여 사용자 선호 기반 최적의 러닝 코스를 생성하자 !

## our cost logic

1. **경사 난이도** : 내리막 가중하여 경사도 기반 cost 재정의 → 오르막 경사도 기반 타겟 난이도 설정(최하/하/중/상) → 동적가중치 적용
2. **야간일** 경우, CCTV 밀도와 가로등 밀도 낮은 곳에 패널티
3. **횡단보도** 있는 경우, 추가 거리 패널티(ex. 횡단보도 1개 → 추가 +30m)
4. **편의점/화장실/조경/지하철** 포함 여부 → 하드 제약
5. 조경 포함할 경우, 코스 내 조경 비율 설정 → 타겟 비율 자동탐색



ex) 연세대 정문 출발, 5km, 야간모드, 난이도“최하”, 화장실 포함

# CHAPTER 2.

## 사용자-아이템-상호작용 데이터셋



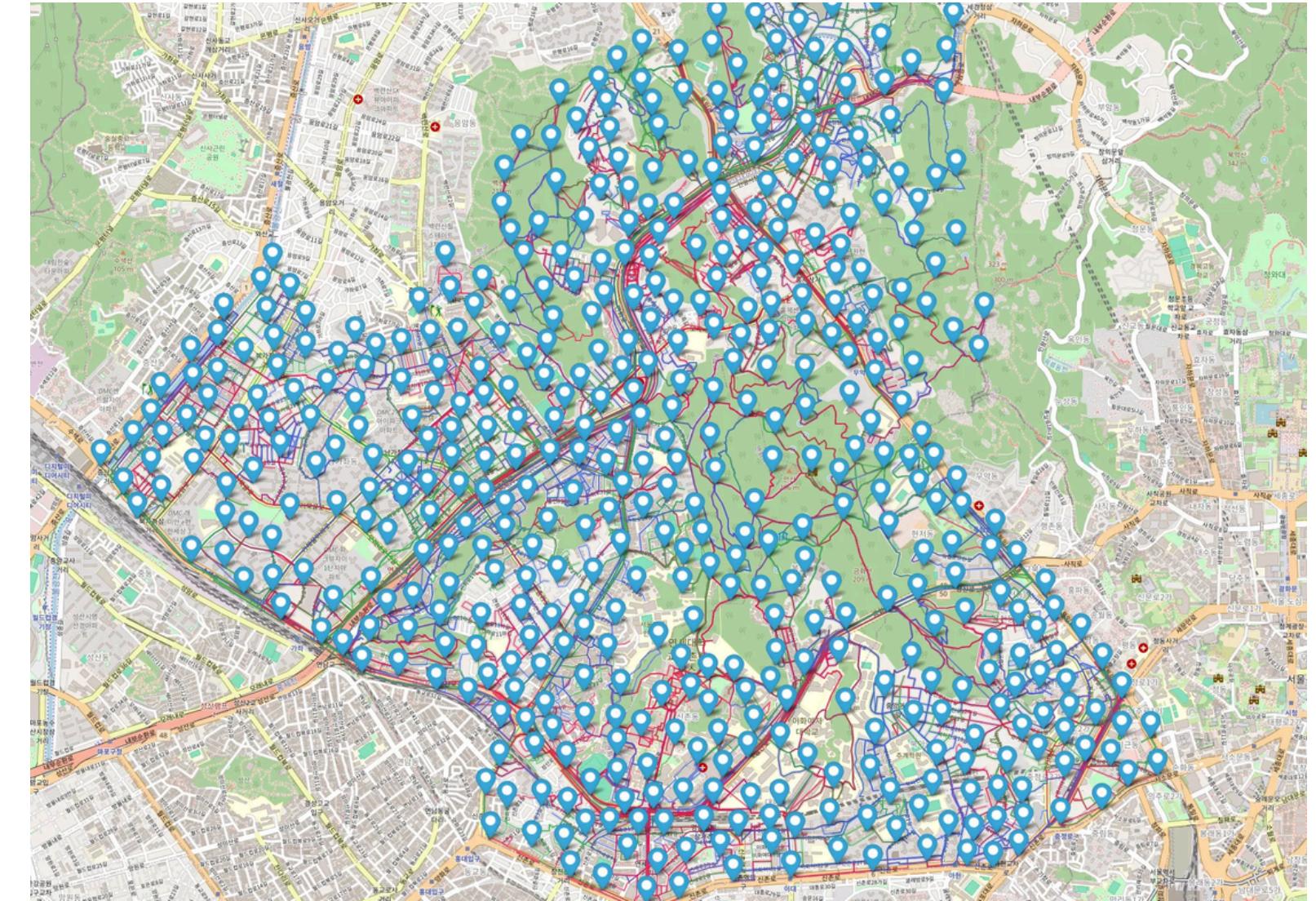
# 2-1. 아이템 데이터셋(ROUTES)

For What?

일반적으로 선호되는 러닝 환경을 고려하여  
서대문구의 잠재적 러닝 코스 생성 및 DB화

## our logic

1. **출발 노드 선정** : 서대문구 전역을 최대한 촘촘히 커버 하는 500개 노드 샘플링
2. **일반적으로 선호되는 러닝 환경 고려** : 보행로 선호, 골목길 회피, 직진 선호 규칙 적용
3. **루프 형태** : 출발 노드로 돌아오는 루프 형태의 경로 생성되도록 품질관리



500개 노드들과 각 노드별 러닝코스 시각화

# 2-2. 사용자 데이터셋(USER)

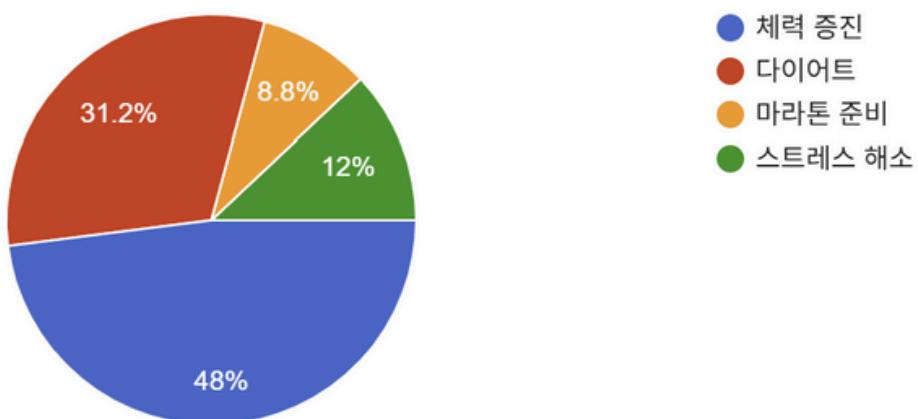
잠재적 사용자들의 러닝 관련 선호도 수집

리서치 매칭 플랫폼을 통해  
온라인 설문조사 진행  
(Surveeasy; <https://gosurveeasy.com/#/>)

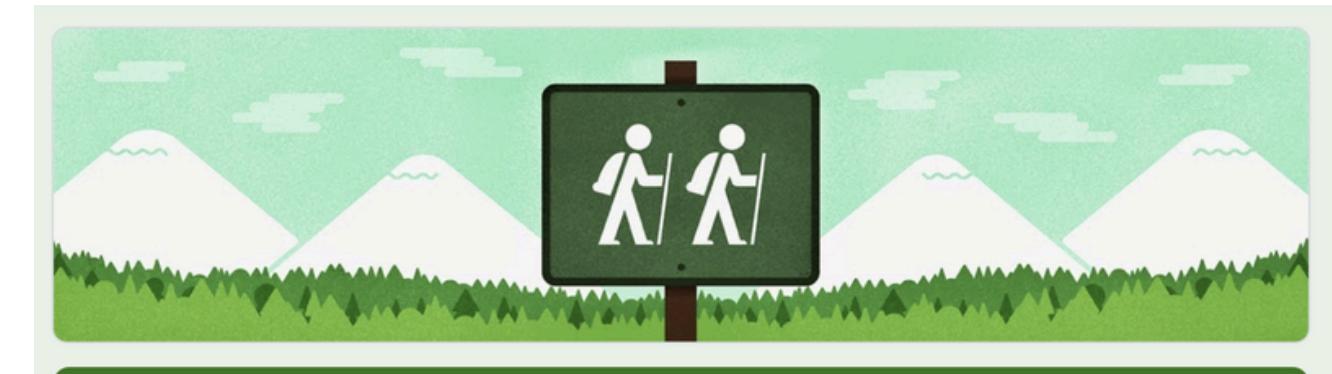
- 설문 기간 : 2025.09.22 ~ 24 (2일간)
- 설문 응답자 : 총 125명

1. 러닝 또는 산책을 하는 주목적은 무엇인가요?

응답 125개



Sur✓easy



## 러닝&산책코스 추천시스템

안녕하세요.

연세대학교 소속 Data Science 학회 DSL입니다.

저희는 현재 러닝&산책 코스 생성을 위한 추천시스템 모델링 프로젝트를 수행하고 있습니다.

프로젝트 연구를 위해 러닝 또는 산책 시 사용자의 선호도 정보가 필요하여 여러분의 소중한 의견을 듣고자 설문조사를 진행합니다.

설문 소요 시간: 약 1분

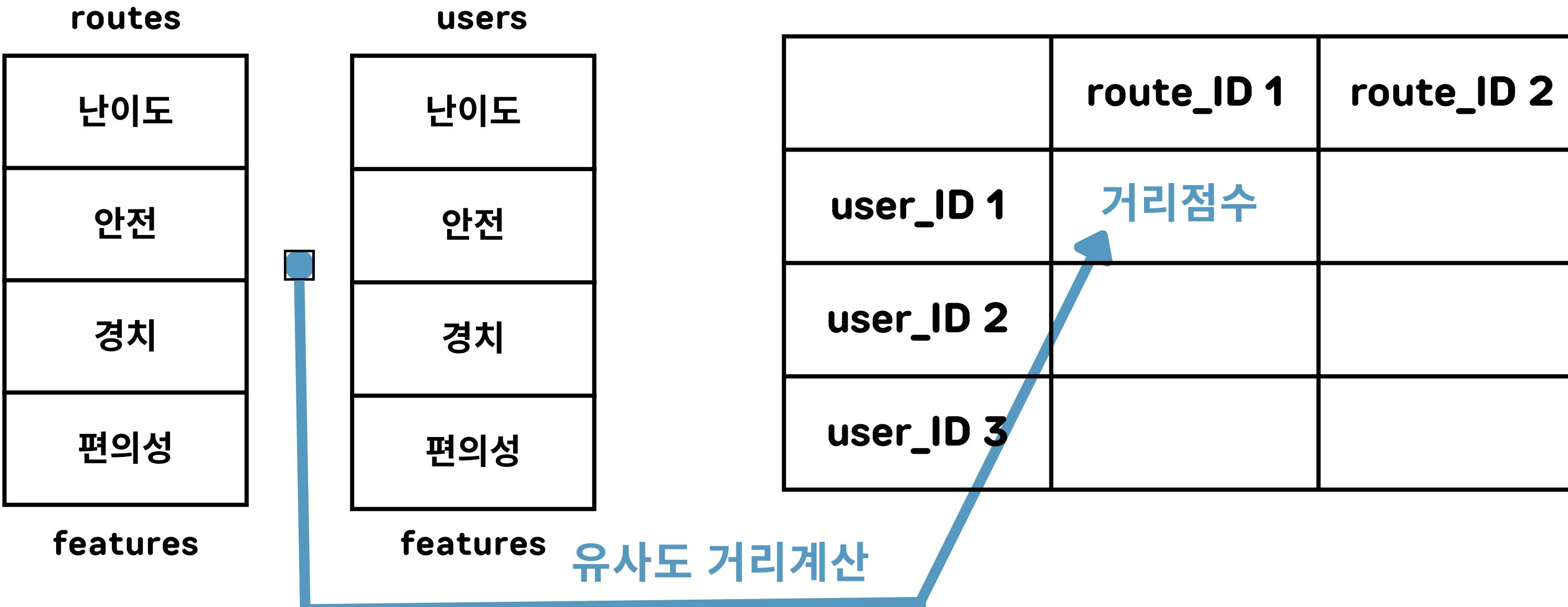
※ 본 설문을 통해 수집된 모든 정보는 익명으로 처리되며, DSL의 자료로만 활용됩니다. 프로젝트 종료와 동시에 즉시 파기될 예정임을 알려드립니다.

깊이 감사드립니다.

\* 표시는 필수 질문임

## 2-3. 상호작용 데이터셋

Main logic : 사용자와 러닝코스 간 유사도 거리가 작을수록 더 선호할 것이다!



달리고 싶은 거리 차이가 클 시 (1km) 무조건 후순위로, 횡단보도 패널티 적용

# CHAPTER 3.

## 딥러닝 모델 기반 러닝코스 추천 시스템:

- TWO-TOWER
- DCN + DEEPM
- KGAT

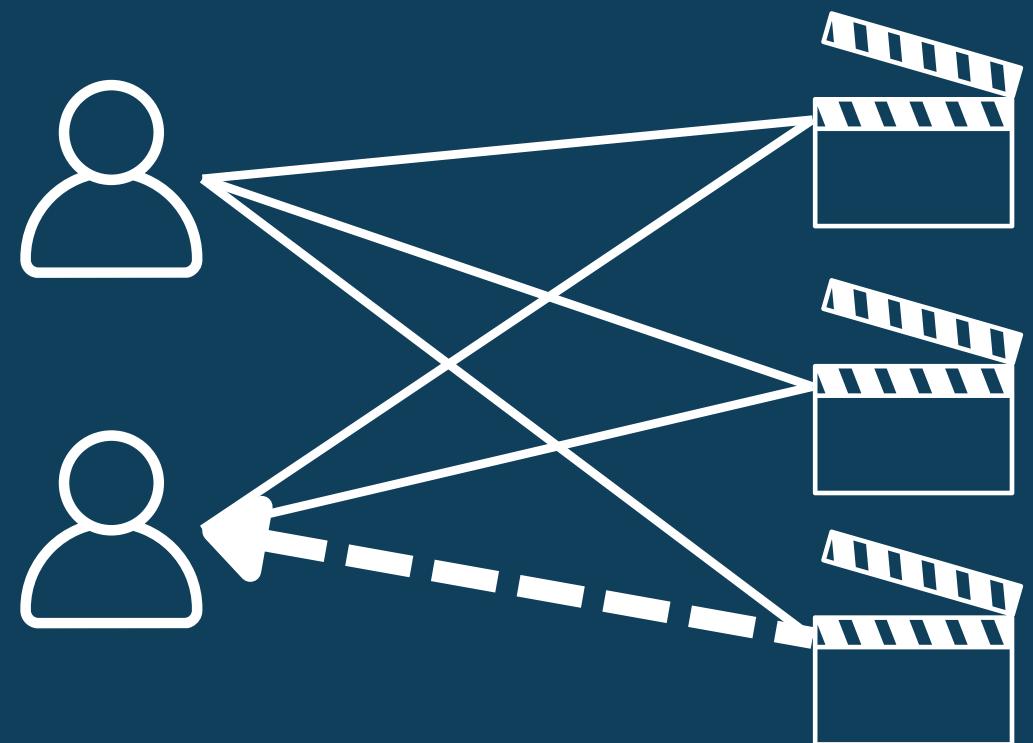


# 3-0. WARMSTART VS COLDSTART

## Warm Start

충분히 기록된 **기존 사용자**에게 새 아이템을 추천해주는 것

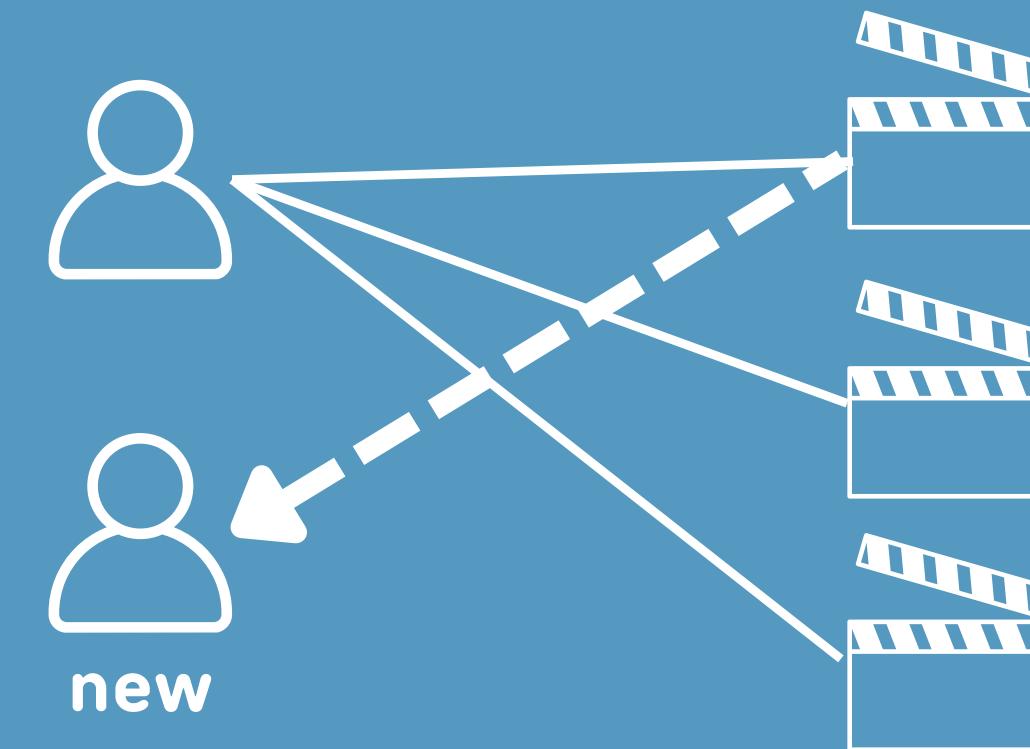
EX : A가 영화 20편에 대한 **평점을 남김** → A에게 새로운 영화 추천



## Cold Start

**새로운 사용자** 또는 새로운 아이템에 대해 추천해주는 것

EX : 평점 기록 없는 신규가입자 B → B에게 영화 추천



# 3-0. WARMSTART VS COLDSTART INPUT

## Warm Start

각 사용자의 **선호 코스**들을 8:2로 분할

positive label (1) : Top1000개 코스

negative label (0): train 1:4, test 1:50 비율로 샘플링

user1	1st itemID	2nd itemID	...	last item ID
user2				
...				
user124				
user125				

## Cold Start

**사용자**를 8:2로 분할

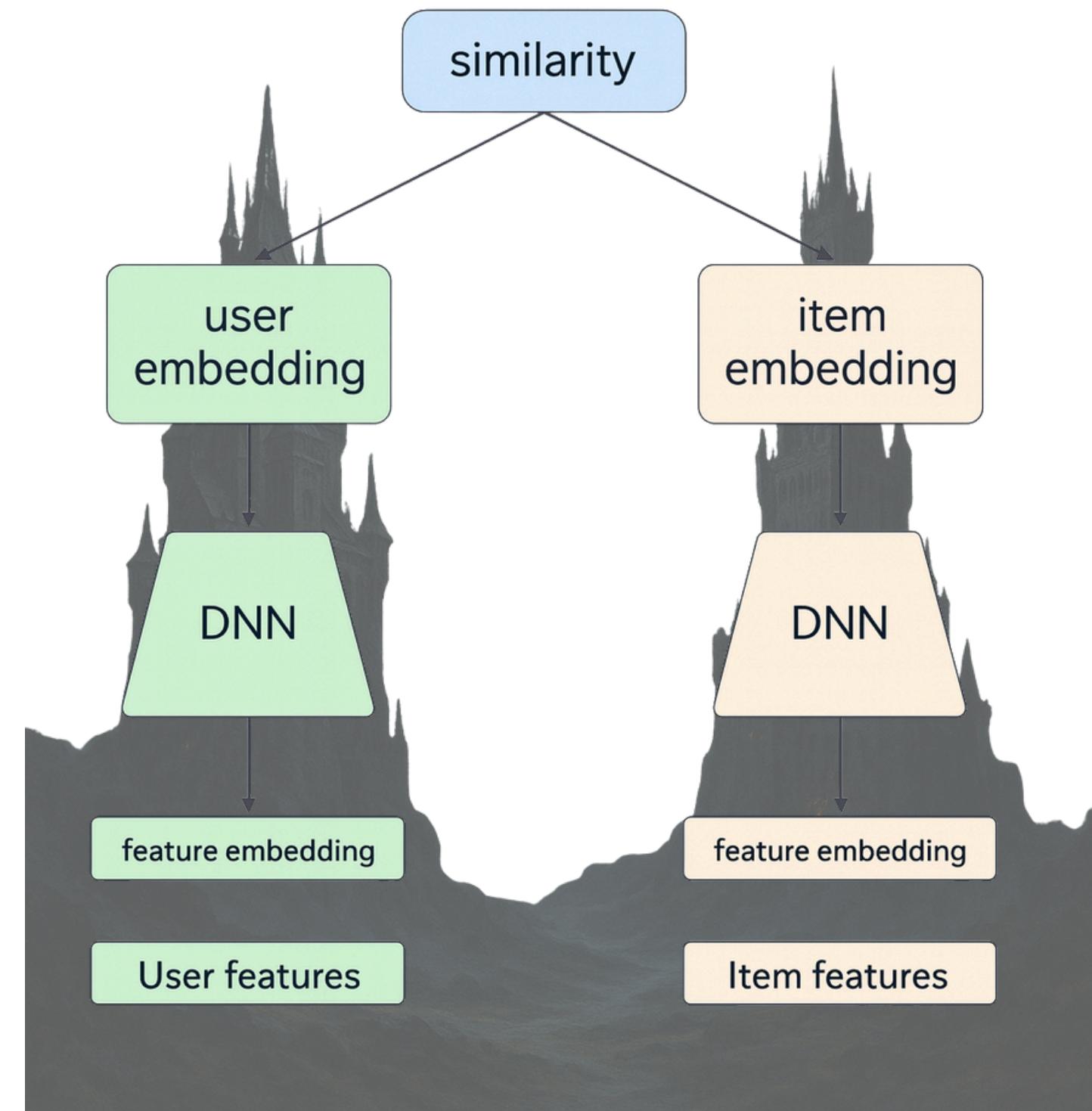
positive label (1): 각 사용자별 Top1000개 코스

negative label (0): 나머지 코스들 중 2000개 샘플링

user1	1st itemID	2nd itemID	...	last item ID
user2				
...				
user124				
user125				

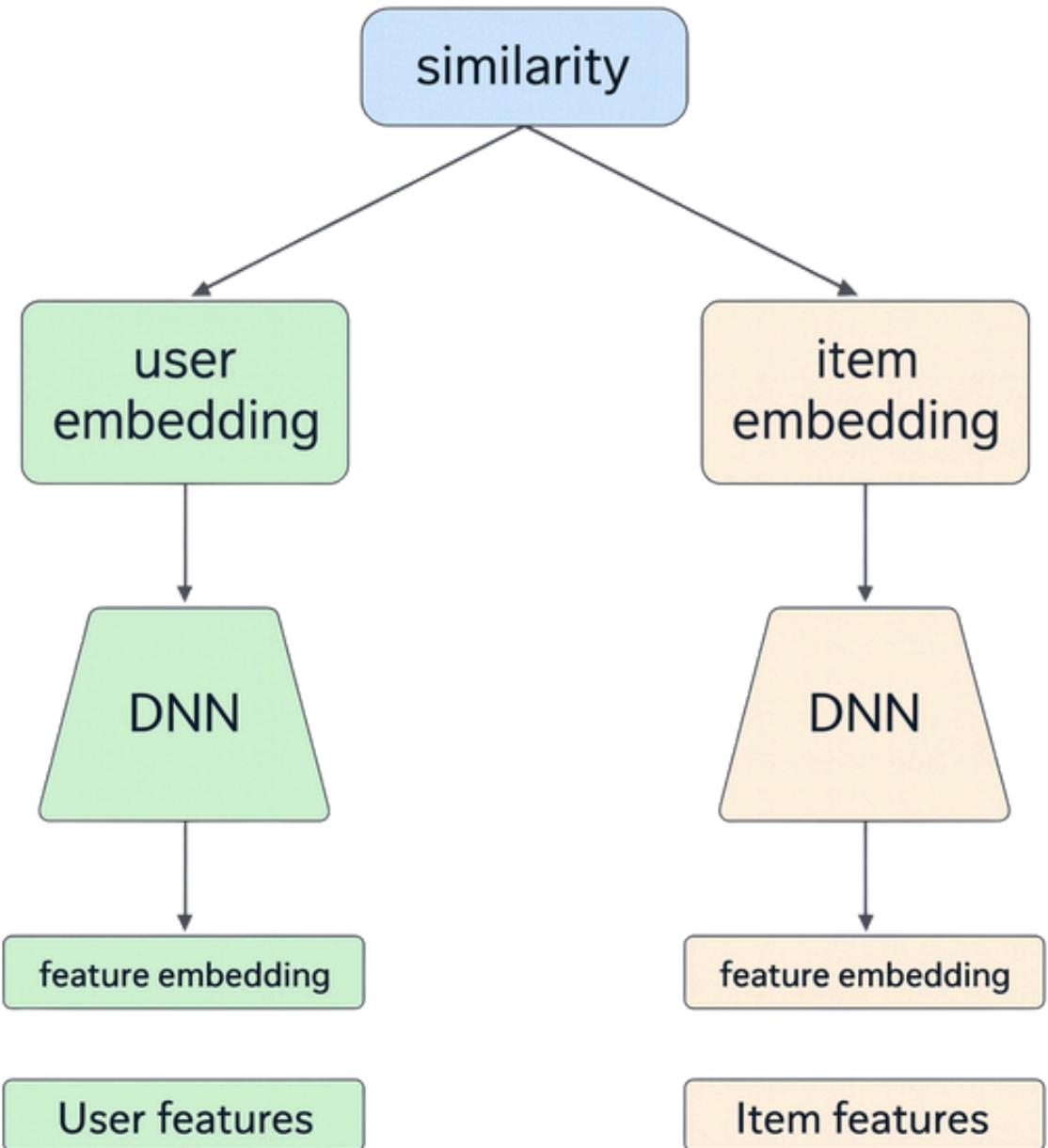


# 3-1. TWO TOWER



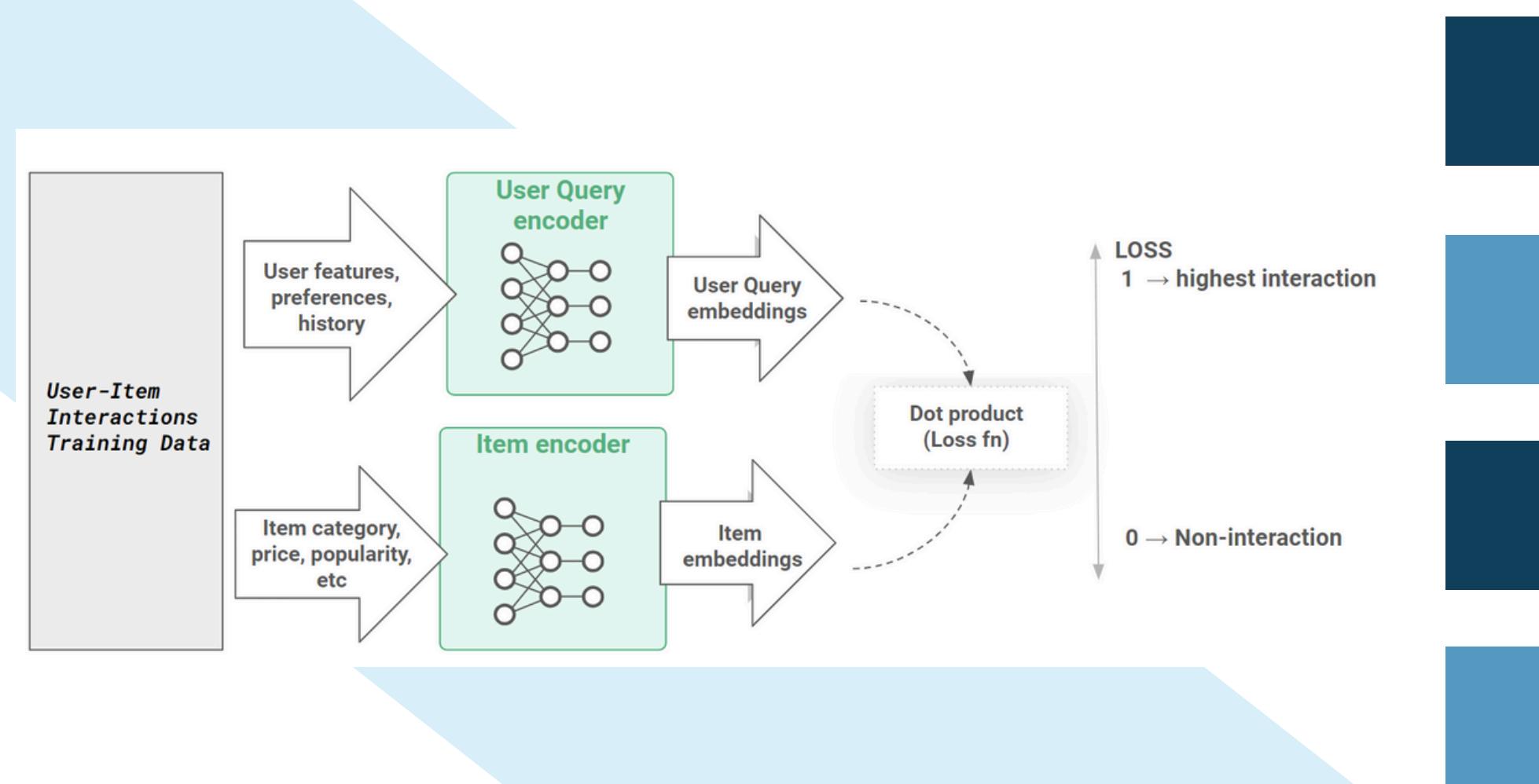
# 3-1. TWO TOWER

사용자 취향 - 러닝코스 특징 - 정답 Mapping 과정



User Tower Input	Item Tower Input	Model's Training Logic
park, river, pc_scenery	has_park (공원 보유 여부)	상호작용 패턴에서 두 개념의 관계를 추론하여 벡터를 가깝게 조정
subway, cvs, toilet, no_conv	has_subway, has_store, has_toilet	선호도(User)와 실제 피처(Item)간의 개념을 매핑하여 벡터 거리를 좁히도록 가중치 업데이트
level	difficulty_score, total_length, uphill_share	유저의 'Level'과 아이템 수치 특성 간 비선형 관계를 학습하여 좌표에 매핑
day_or_night	night_safe_mean, bright_share	긍정 쌍(Positive Pair)의 내적(Dot Product) 값은 최대화, 부정 쌍(관련 없는 코스)은 최소화

## 3-1. TWO TOWER TRAINING 과정



### Prediction:

사용자 정보와 아이템 정보를 각각의 타워에 넣어 **Embedding Vector**  
→ 두 벡터가 얼마나 유사한지 **dot product**로 계산

### Loss Calculation:

모델의 예측이 실제 정답과 얼마나 다른지 Loss값으로 계산  
→ user가 **선호하는 item의 예측 점수 UP, 비선호하는 item은 DOWN**

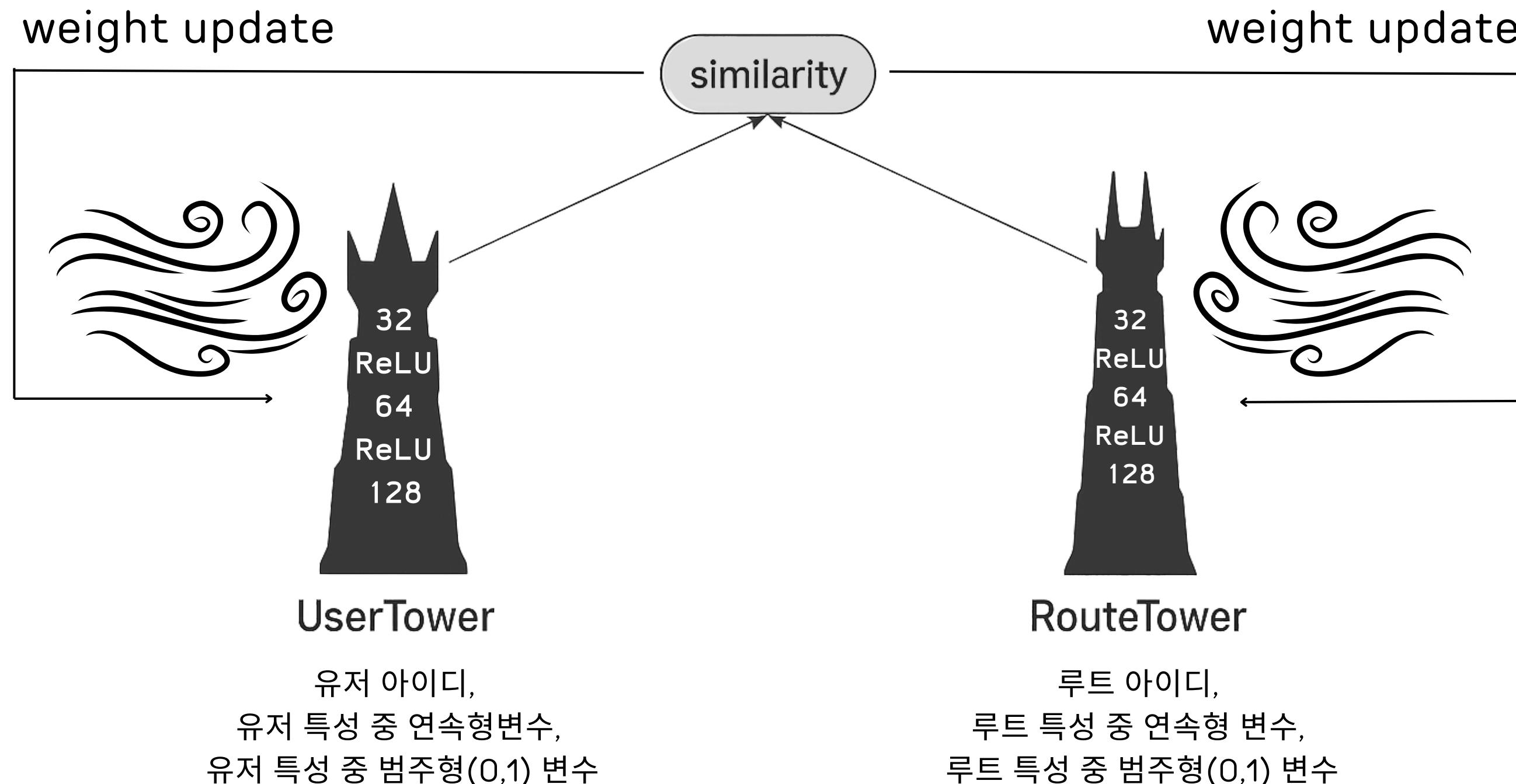
### Backpropagation

계산된 손실을 기반으로, 원인을 거꾸로 추적

### Weight Update

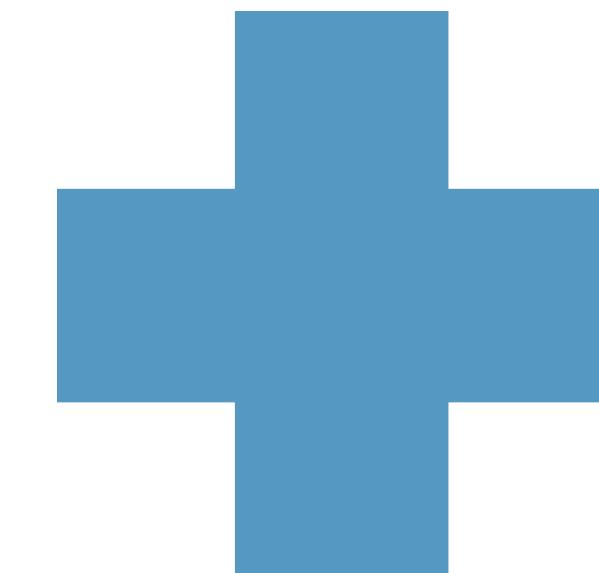
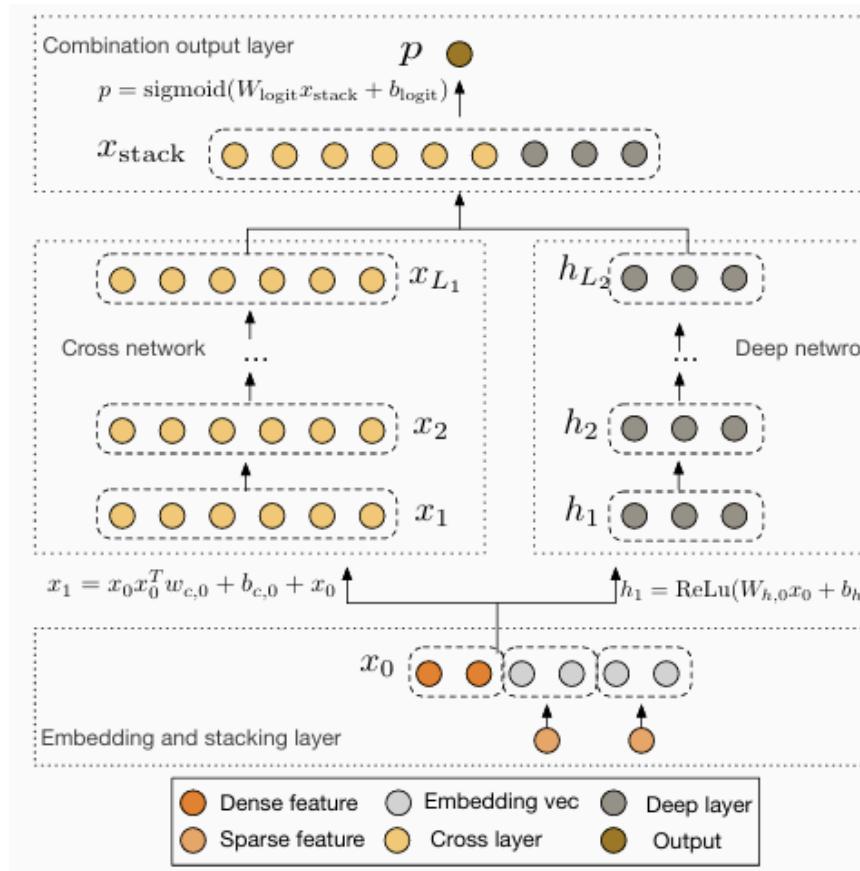
역전파를 통해 알아낸 걸 바탕으로 각 타워의 내부 **파라미터(가중치)**를 수정

# 3-1. TWO TOWER

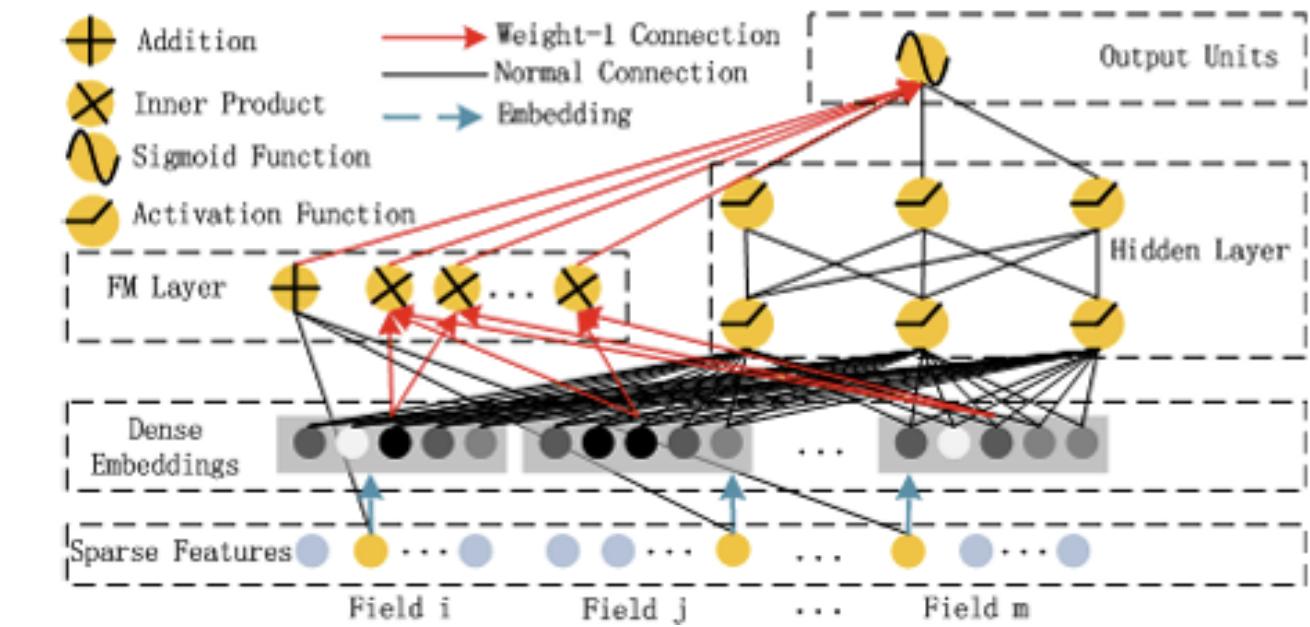


# 3-2. DCN + DEEP FM

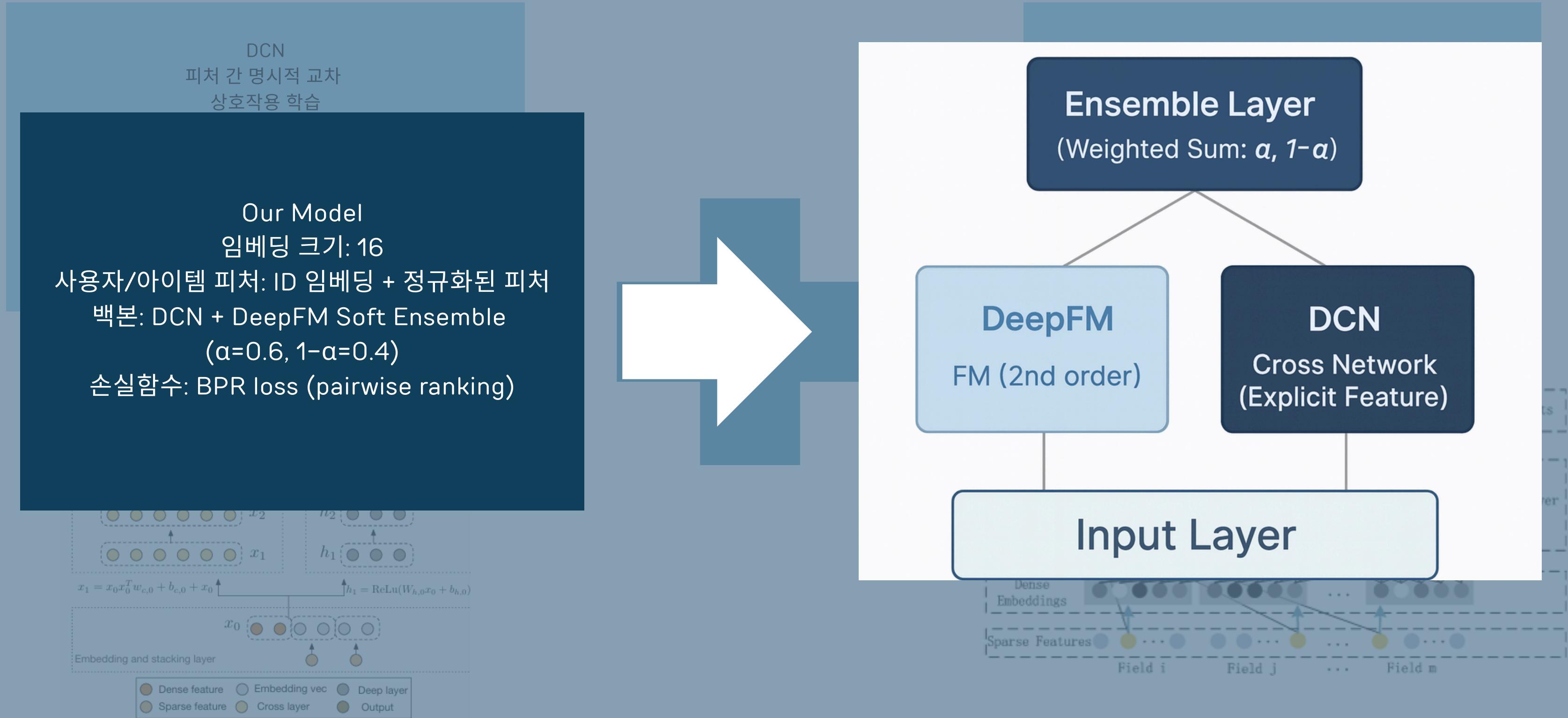
**DCN**  
피처 간 명시적 교차 상호작용 학습  
피처 쌍 간 관계를 선형적으로 포착함  
DNN을 통해 비선형적 고차원 패턴도 함께 학습함



**Deep FM**  
FM 구조로 피처 임베딩 쌍 간 2차 상호작용 명시적 학습  
심층 네트워크를 통해 복잡한 비선형 관계 탐색함  
저차원 조합 + 고차원 표현을 동시에 수행함



# DCN + DeepFM



## 3-2. DCN + DEEP FM

COLD START

속성 피쳐 입력 / ID 임베딩 배제 및 속성 임베딩 구성

DCN + DeepFM 모델 학습

BPR Loss 기반 최적화

속성 기반 추천 결과 생성 : 신규 사용자·경로 예측 가능

WARM START

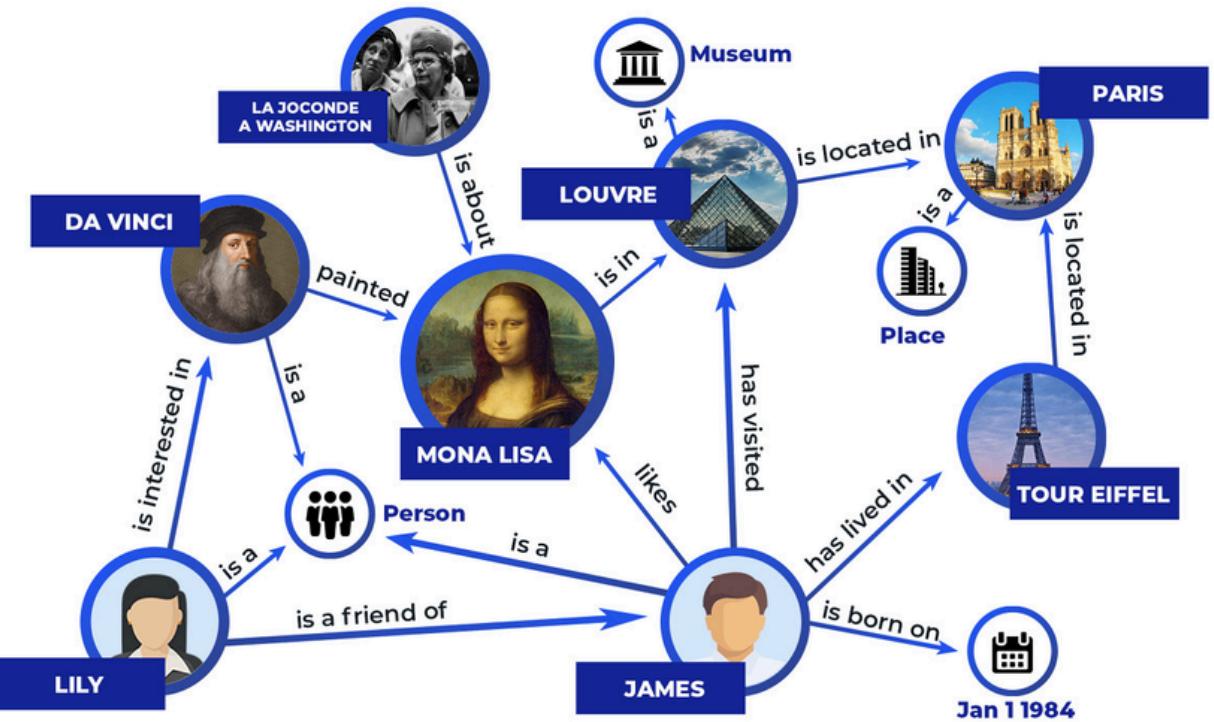
ID + 속성 피쳐 입력 / ID 임베딩 + 속성 임베딩 결합

DCN + DeepFM 모델 학습

BPR Loss 기반 최적화

개인화 추천 결과 생성

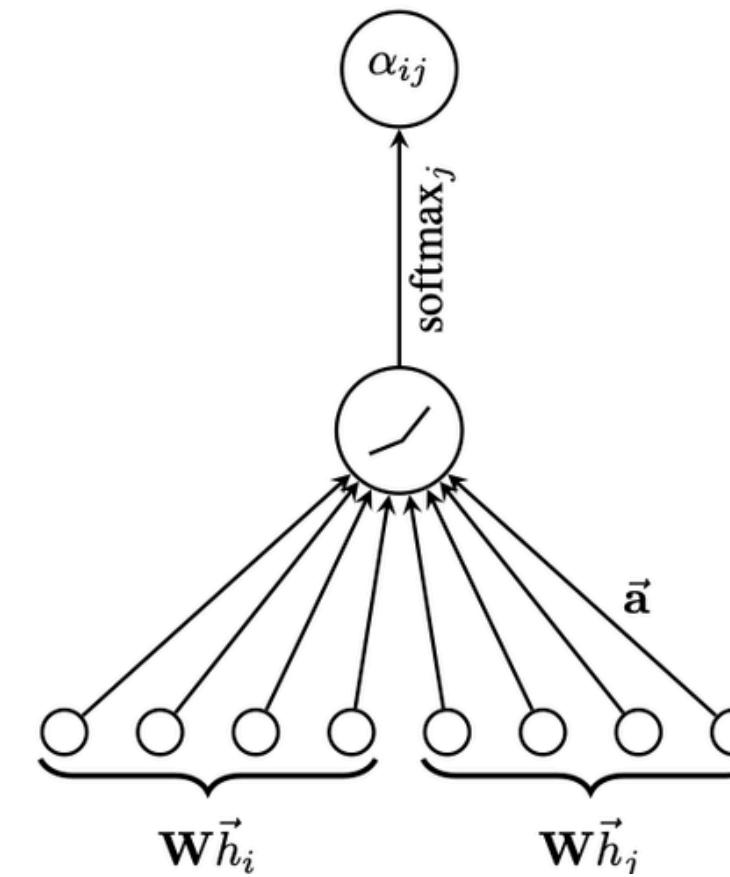
# 3-3. KGAT



## KG(Knowledge Graph)

- 개체(Entity)와 그들 사이의 관계(Relation)를 그래프 형태로 표현하여, 지식을 구조적으로 저장하는 데이터 모델
- 정보와 정보 사이의 맥락과 의미를 구조화하여, 지식을 이해하고 추론할 수 있게 만드는 기반.

+

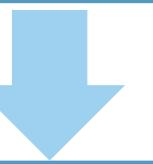


## GAT(Graph Attention Network)

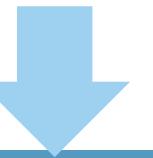
- 그래프 구조 데이터에 어텐션 메커니즘을 적용한 신경망 모델
- 이웃 노드의 정보를 취합할 때 동적으로 가중치를 부여함으로써, 문맥상 더 중요한 이웃을 식별하여 정교한 예측을 수행.

# 3-3. KGAT

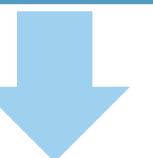
특징 부여 및 트리플 변환



임베딩 학습 및 관계 반영



어텐션 기반 임베딩 전파



선호도 예측 및 최적화

## Our Model

- 임베딩 크기: 64
- 관계 임베딩 크기: 64
- KGAT 레이어수 : 2
- 집계함수 :
  - Attention-based
  - Additive
  - Aggregation
- 손실함수 : Bpr Loss

# 3-3. KGAT

## 지식 그래프 트리플(주어-관계-목적어) 구성

주체 (Subject)	관계 (Predicate)	객체 (Object)
Route (경로)	has_feature	Yes (Park, Store, Subway, Toilet, Waterside)
	has_flaw:Many_Crossings	Yes
	has_property:Difficulty	High, Medium, Low
	has_property:Safety	High, Medium, Low
User (사용자)	likes	Route_1234 (경로 ID)
	dislikes	Route_5678 (경로 ID)
	is_a:Persona	Marathon, Dc Stress, Getting Fit 등
	prefers	Feature:Park, Feature:River, Feature:Toilet 등
	prefers_slope:Level	High, Medium-High, Medium-Low, Low
	runs_at:Time	Day, Night

ex) User ID : 4, Route ID : 555

===== 사용자 ID: 4, 경로 ID: 555에 대한 트리플 구성 예시 =====

### [1] 사용자 특징 트리플 (User Features)

'output4(in).csv'에서 User\_4의 정보를 가져와 변환합니다.

- 목적 (marathon=1) -> ('User\_4', 'is\_a:Persona', 'Marathon')
- 선호 경사도 (level=3) -> ('User\_4', 'prefers\_slope:Level', 'High')
- 선호 시설 (fountain=1) -> ('User\_4', 'prefers', 'Feature:Fountain')

### [2] 경로 특징 트리플 (Route Features)

'df\_route\_capped\_normalized.csv'에서 Route\_555의 정보를 가져와 변환합니다.

- 보유 시설 (has\_park=1) -> ('Route\_555', 'has\_feature:Park', 'Yes')
- 난이도 점수 (16.30) -> ('Route\_555', 'has\_property:Difficulty', 'High')

### [3] 상호작용 트리플 (Interactions)

'user\_preferred\_route.csv'에서 두 개체의 관계를 확인합니다.

- User\_4의 Route\_555 순위: 38211위 (Bottom 2000)
- 최종 관계 -> ('User\_4', 'dislikes', 'Route\_555')

===== 최종적으로 미 개체들에 대해 생성된 트리플 목록 =====

	subject	predicate	object
0	User_4	is_a:Persona	Marathon
1	User_4	prefers_slope:Level	High
2	User_4	prefers	Feature:Fountain
3	Route_555	has_feature:Park	Yes
4	Route_555	has_property:Difficulty	High
5	User_4	dislikes	Route_555

# CHAPTER 4.

평가지표 기반  
모델 성능 평가:  
 $HR@1$ ,  $HR@10$



# 4-1. 평가지표 :

## Hit Ratio (HR@k)

$$HR@K = \frac{\text{Number of Hits}}{\text{Number of Users (or test cases)}}$$

추천한 Top-k 리스트 안에 실제 사용자가 선호한 아이템(positive)이 존재하는 비율

【Krichene & Rendle, 2020】

### Warm Start

- **HR@1** : 기존 사용자의 다음 선호 아이템을 1순위로 맞춘 비율 (정밀 개인화)
- **HR@10** : Top-10 후보 안에 실제 선호 아이템 포함 비율 (정밀도+재현율 균형)

### Cold Start

- **HR@1** : 신규 사용자에게 첫 추천이 실제 선호와 일치 할 확률 (초기 Onboarding 정확도)
- **HR@10** : Top-10 안에 선호 아이템 포함 비율 (탐색적 추천·일반화 능력)

## 4-2. 모델 성능평가

### Warm Start

	HR@1	HR@10
<b>Two-Tower</b>	0.52	0.68
<b>DCN + DeepFM</b>	0.79	1.00
<b>KGAT</b>	0.32	0.85

### Cold Start

	HR@1	HR@10
<b>Two-Tower</b>	0.32	0.60
<b>DCN + DeepFM</b>	0.92	1.00
<b>KGAT</b>	0.56	0.84

# CHAPTER 5.

## 결론 및 한계



# 5. 결론 및 한계

## Conclusion

- 다익스트라 기반 최적화 경로 탐색을 통해 러닝코스를 구현
- 사용자-아이템 데이터셋을 구축하여 추천 시스템을 실현

→ 개별 사용자 맞춤형 러닝코스 제안 가능성 확인

## Limitation

- 데이터셋 한계 :
  - 125명의 소표본 데이터라 과적합 위험
  - 가상의 사용자-아이템 상호작용 데이터이므로 일반화의 한계 존재
- 모델의 한계 :
  - DCN + DeepFM 모델의 경우 HR@1, HR@10의 값이 recall에 비해 터무니없이 높음
  - → 추후 수정 예정

SPORT THANKS

