

이게 Merge..? 추천 잘 될 Sudo?

Recsys-02 R_AI_SE조



우리는 32개의 서버를 죽였고 ...
15개의 모델을 쓰레기통에 넣었다.

...

목차

- 서론
 - 팀 소개
 - DKT 대회 개요
 - 프로젝트 계획 및 협업 방식
- 본론
 - Feature Engineering & Feature Selection
 - 가설 설정과 모델링
 - Static Data
 - Sequential Data
- 결론
 - 단일 모델 학습 결과
 - 양상별 학습 결과
 - 잘한 점 & 아쉬웠던 점 & 배운 점
 - 궁금한 점



팀 소개

R_AI_SE (Recsys AI SeekEr)

추천시스템 인공지능 탐구자

- 우리 조의 장점? 특징?

- 24시간 코딩 이후 4시간 취침의 열정맨덜
- 티키타카 꿀잼
- 슬랙 허들은 개미지옥이야..
- 10시 데일리 스크럼, 16시 피어세션, ~~23시45분 양상블세션~~

```
# 모듈에 따라 값은 주어야 할까
def modify_value(value):
    if value < 0.55:
        return value*0.5/0.55
    else:
        return 0.5/0.45*(value-0.55)+0.5

# 'prediction' 변수 값에 함수 적용
submit['prediction'] = sasrec['prediction'].apply(modify_val)

[29] submit['prediction'] = 0.5 * s129['prediction'] + 0.5 * s130
[21] submit['prediction'] = 0.3 * s130['prediction'] + 0.3 * s81
[ ] submit['prediction'] = 0.4 * s130['prediction'] + 0.15 * s81
[ ] submit['prediction'] = 0.3 * sas['prediction'] + 0.3 * rec['prediction']

[42] submit.to_csv('kimera.csv', index=False)
catboost03['prediction'] = catboost03['prediction'].clip(lower=0, upper=1, axis=None, inplace=True)

0 0.667576
1 0.857322
2 0.334051
3 0.769468
4 0.291483
...
739 0.017262
740 0.702938
741 0.747758
742 0.618134
```

양상블을 위한 굿판 - 온라인 ver.1.0.0 minor

DKT 대회 개요

프로젝트 요약

- 주제: 유저의 학습 상태에 대해 새로운 문제를 풀 가능성 예측
- 배경
 - 시험 성적은 우리가 얼마만큼 아는지 평가하는 한 방법이지만, 학생 개개인의 이해도를 기리키는 ‘지식 상태’는 알 수 없다.
 - 이를 해결하고자 딥러닝 방법론인 ‘Deep Knowledge Tracing’이 등장하였고 맞춤화 교육을 위해 아주 중요한 역할을 한다.
 - 대회에서는 지식 상태를 예측하기보다는, 주어진 문제를 맞출지 틀릴지 예측한다.
- Task: test_data 마지막 문제의 정답여부가 주어지지 않으며, 이를 예측해야 한다.

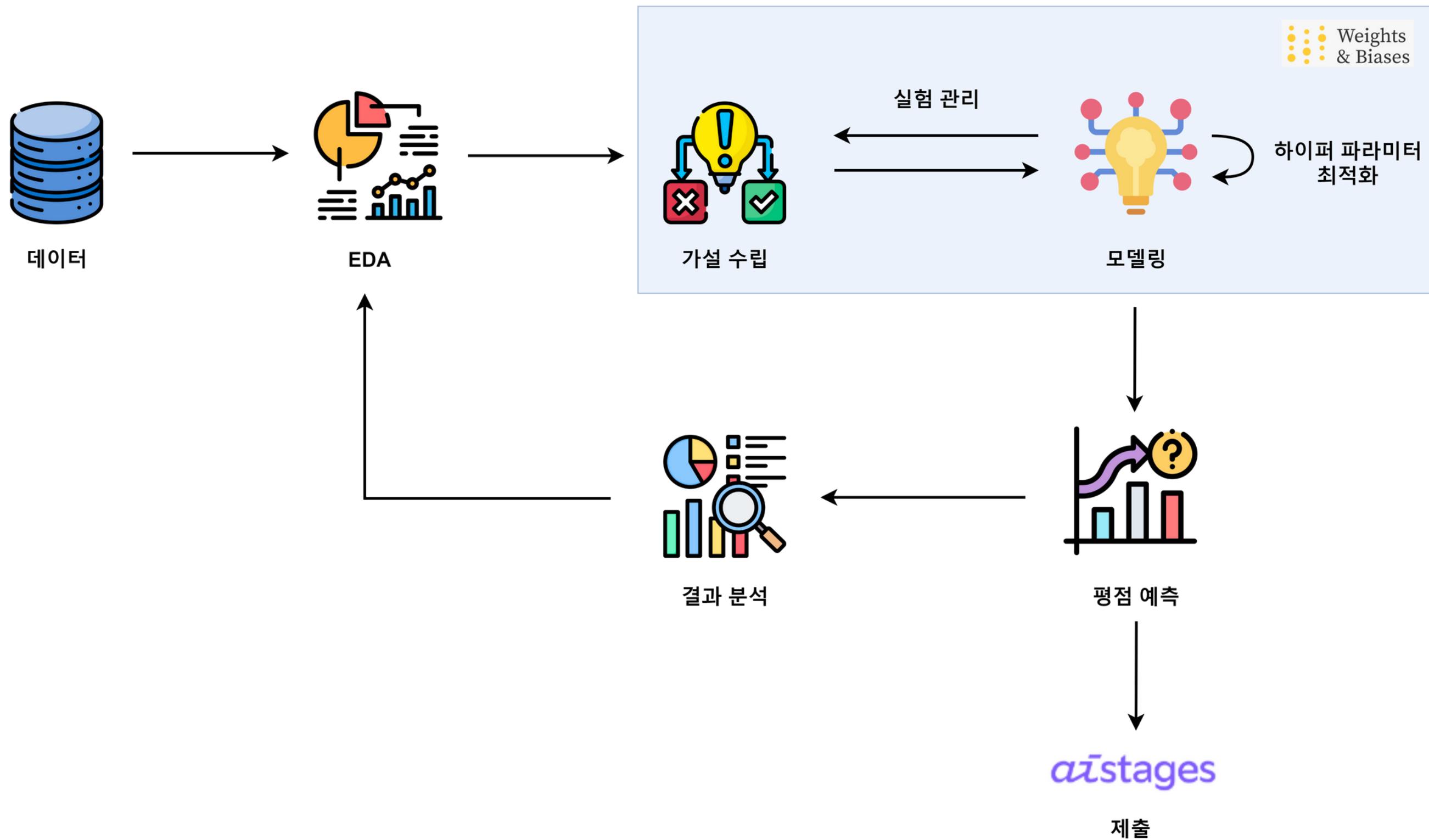
학습 목표

- 트리 모델을 최대한 사용하지 않도록 한다.
- 학습 목적이므로 내부 구조를 바꿀 수 있는 Deep Learning Model을 최대한 사용한다.
- 각자 모델별 end-to-end를 경험 후, 경험을 합친다.
- 공유하고, 질문하고, 토론한다.
- 모델 하나를 깊게 파고 이해한다.
- 가망이 없다면 미련 없이 돌아선다.
- 궁금하면 일단 시도 해본다.
- 모든 판단은 근거에 기반한다.

프로젝트 계획

31일	1월 1일	2일	3일	4일	5일	6일
	새해			EDA, 서버 세팅		
7일	8일	9일	10일	11일	12일	13일
	강의&미션 공부 및 발표				Feature Engineering	
14일	15일	16일	17일	18일	19일	20일
	Feature Engineering					
			가설 바탕으로 Modeling 진행			
21일	22일	23일	24일	25일	26일	27일
	가설 바탕으로 Modeling 진행 및 하이퍼 파라미터 튜닝					
			양상을 학습			

프로젝트 파이프라인



협업 방식

Github Issue 생성

[FEAT] Tree기반 모델 베이스라인 코드 작성 #18

(Closed) (4 tasks done) ChangZero opened this issue 2 weeks ago · 0 comments

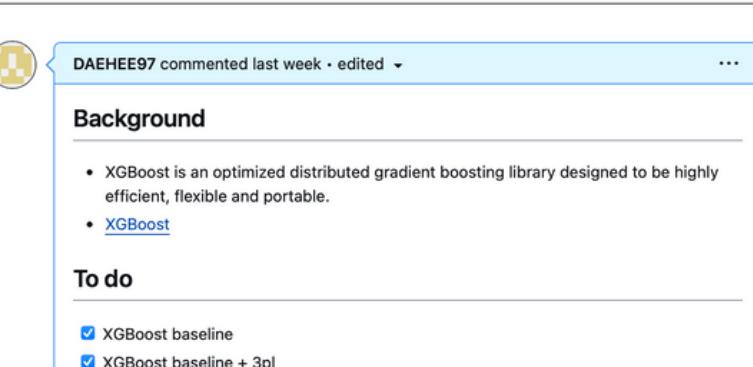


Github Kanban Board를 활용한 일정 관리

하위 이슈 생성

[FEAT] XGBoost baseline 개발 #51

(Closed) (2 tasks done) (1 comment) DAEHEE97 opened this issue last week · 0 comments · Fixed by #72

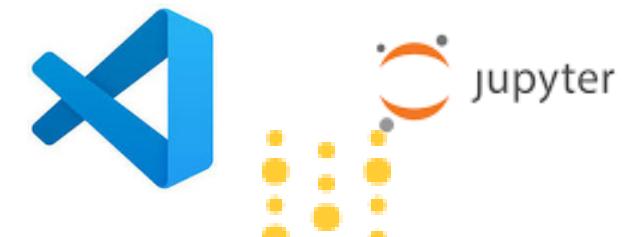


코드 리뷰 및 병합



작업 브랜치 생성 및 작업

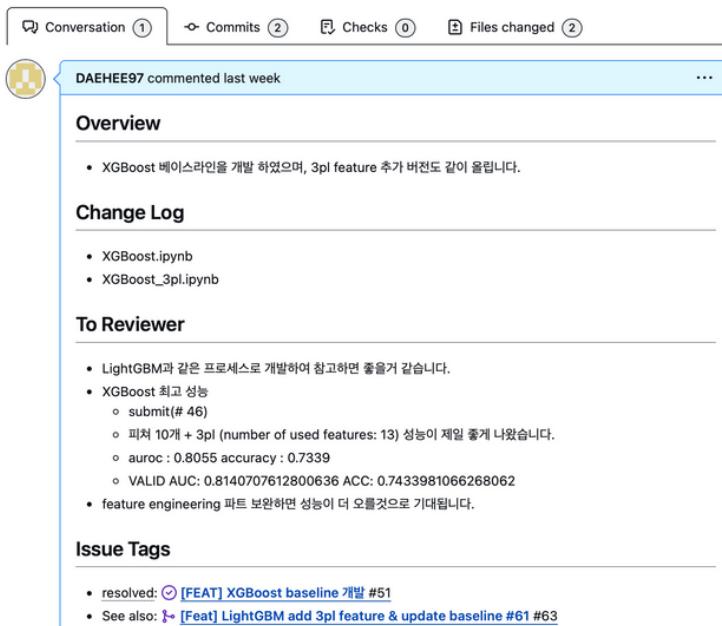
feat/51-XGBoost



PR 요청

[Feat] XGBoost baseline & add 3pl feature #51 #72

(Merged) DAEHEE97 merged 2 commits into main from feat/51-XGBoost 3 days ago



Feature Engineering

학습 데이터

num	Feature	dtype	Description
X1	UserID	catagorical	사용자의 고유번호
X2	assessmentItemID	catagorical	문항의 고유번호
X3	testID	catagorical	시험지의 고유번호
X4	Timestamp	datetime	사용자가 해당문항을 풀기 시작한 시점의 데이터
X5	KnowledgeTag	catagorical	문항 당 하나씩 배정되는 태그
Y	answerCode	int64	사용자가 해당 문항을 맞췄는지 여부에 대한 이진 데이터

Feature Engineering

IRT(Item-Response Theory, 문항 반응이론) 기반 feature 생성

문항마다 보편적인 고유한 특성을 가지는 문항함수를 이용해 모델링하여 잠재적 구인에 대한 보편적 측정을 시도하는 이론
특정 집단에 한정되지 않는 보편적인 문항 고유의 특성을 도출하여 피험자에 대한 표준화된 분석을 하는 접근

시험자가 다른 시험을 보더라도 점수를 비교가능하도록 표준화된 평가 점수를 부여하기 때문에 PISA, TIMSS와 같은 국제 성취도 평가
나 TEPS, TOEIC 등의 시험에서 활용

R의 Itm 패키지를 이용하여 문항별 3모수(난이도, 변별도, 추측도)와 각 사용자의 능력 수준을 feature로 생성

$$P_i(\theta) = P(X_i = 1 | \theta) = c_i + (1 - c_i) \frac{e^{a_i(\theta - b_i)}}{1 + e^{a_i(\theta - b_i)}}$$

어떤 능력 θ 를 가진 사람이 문항 i 를 맞출 확률을 로지스틱 모형으로 표현
이때 확률 분포에 적용되는 모수가 난이도(b), 변별도(a), 추측도(c)

Feature Engineering & Feature Selection

Feature	dtype	Description
`dffclt`	float64	각 문항별 난이도(로지스틱 모형의 모수 b) : 문항을 맞출 확률이 0.5일 때 해당되는 능력 수준
`dscrmn`	float64	각 문항별 변별도(로지스틱 모형의 모수 a) : 문항을 맞출 확률이 0.5일 때의 모형에서의 상승 기울기
`gussng`	float64	각 문항별 변별도(로지스틱 모형의 모수 c) : 능력 수준이 음의 무한대로 접근할 때 수렴하는 값
`user_level`	float64	문항반응모형으로부터 구한 학습자의 능력 수준. 각 test를 기준으로 구하였음.

*user_level은 feature로 사용했을 때, overfitting이 심하게 발생하여 활용하지 않았습니다.

Feature Engineering & Feature Selection

Feature	dtype	Description
`user_correct_answer`	float64	유저별 누적 정답 횟수
`user_total_answer`	int64	유저별 누적 제출 횟수
`user_acc`	float64	유저별 누적 정답률
`user_mean`	int64	유저별 평균 정답률
`prior_testTag_frequency`	int64	유저별 testID 가운데 수 별 이전에 몇번 풀었는지
`time_to_solve`	float64	유저별 문항을 해결하는데 걸리는 시간
`time_to_solve_mean`	float64	유저별 문항을 해결하는데 걸리는 시간의 평균
`relative_answer_mean_level`	float64	유저 학습 수준 1 (유저별 상대적 정답률의 평균)
`assessmentItemID_time_level`	float64	유저 학습 수준 2 (문항당 문제를 푸는 평균시간 - 각 유저의 문제 푸는 시간)

모델링 - DKT 문제를 관점에 따라 분석하기

- **Static Data (정적 데이터)** - 데이터의 시간 관계를 고려하지 않음
 - **Decision Tree 모델**
 - XGBoost / LGBM / CatBoost
 - **AutoML**
 - AutoGluon
 - **DNN 모델**
 - TabNet
 - **Matrix Factorization**
 - **GNN 모델**
 - LightGCN
 - SGCN
- **Sequential Data (시계열 데이터)** - 유저로 그룹화 된 시퀀스 데이터
 - **RNN**
 - **Transformer**

Static Data

**Task를 마지막 문제의 정답여부만을 예측하는
‘정형 데이터 분류 문제 접근방식’으로 처리해보자!**
-> Decision Tree modeling

Decision Tree modeling

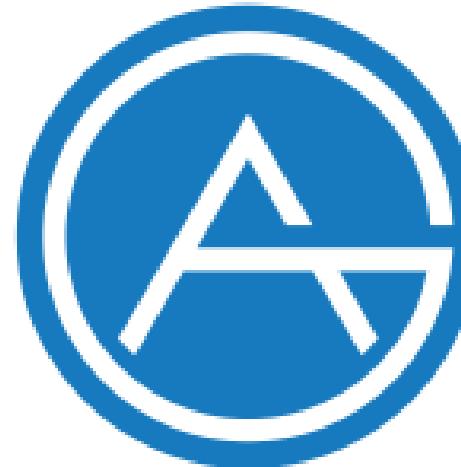


GridSearchCV와 Wandb sweep을 이용하여 하이퍼파라미터 튜닝 자동화 처리

AutoML (Automated Machine Learning)

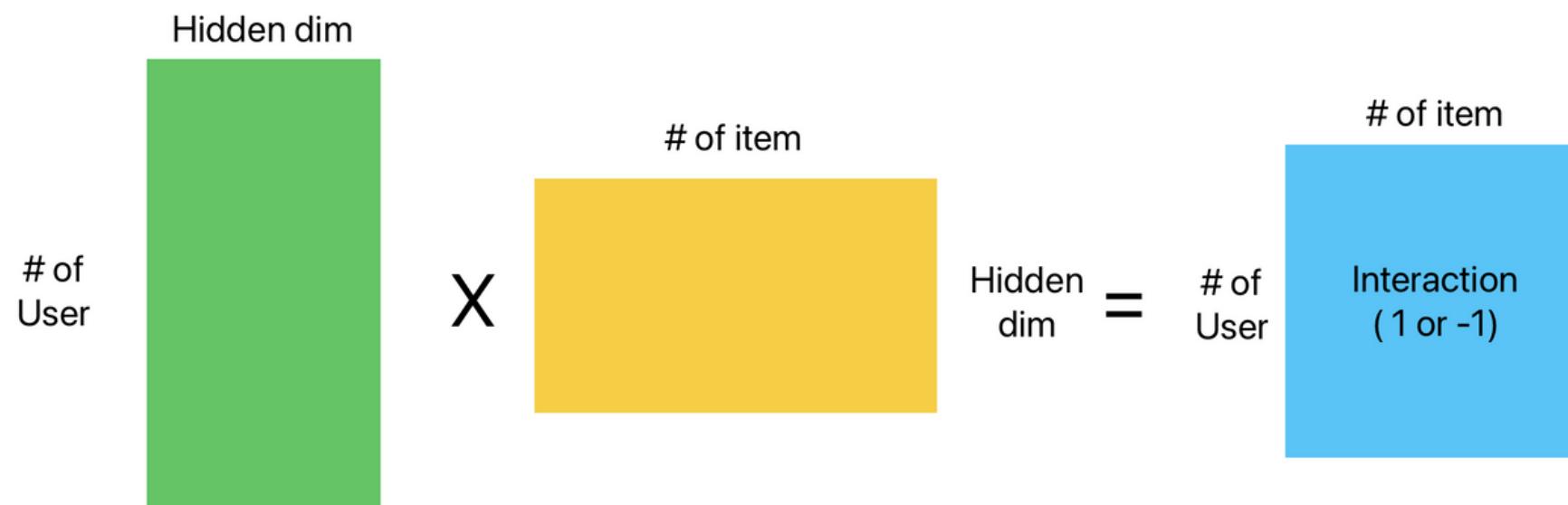
전처리 과정, 알고리즘 선택, 모델 학습 및 평가, 파라미터 튜닝 등 복잡한 단계를
단순화 하고, 자동화 할 수 있는 방법이 있을까?

-> AWS에서 개발한 AutoML 오픈 소스 프레임 워크 AutoGluon 도입!



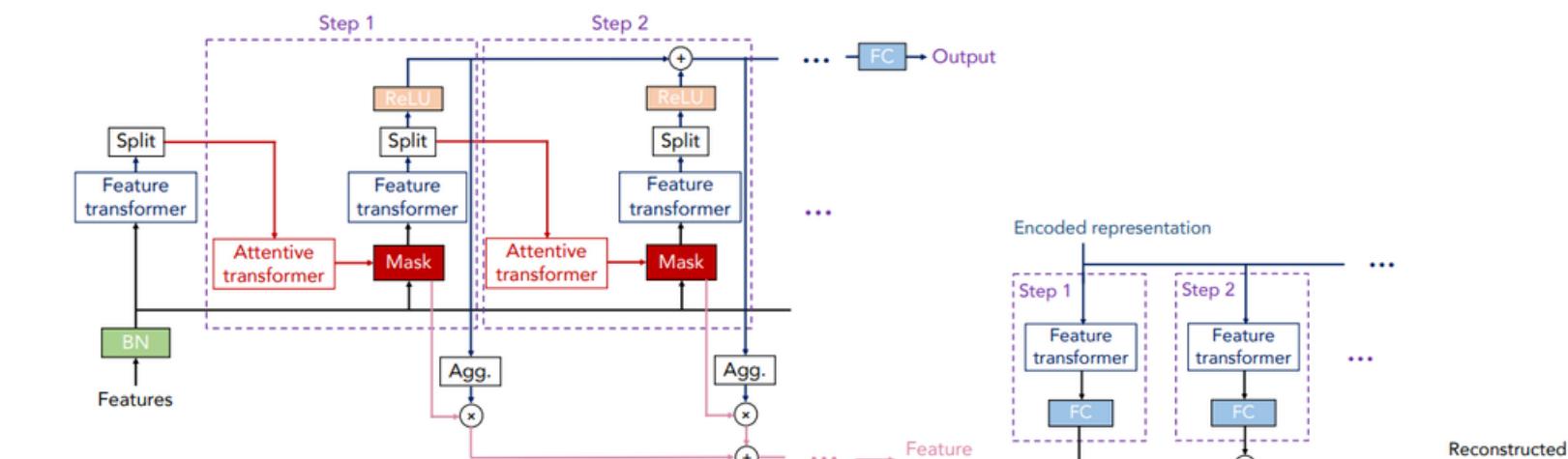
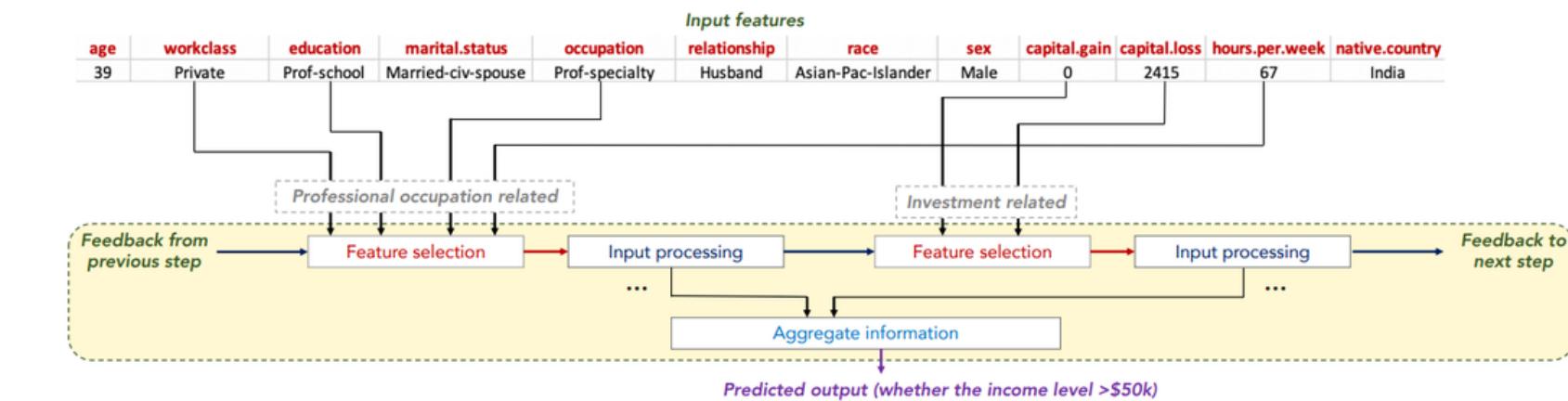
AutoGluon

Matrix Factorization

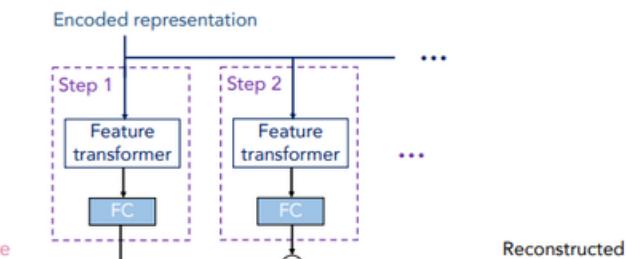


“Simple and easy but powerful”
단순하지만 성능이 잘 나오는 모델

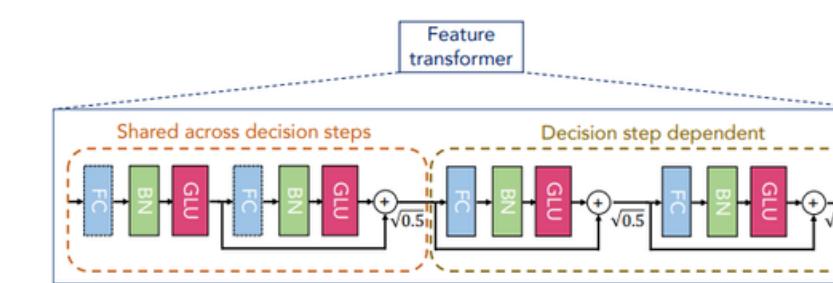
TabNet



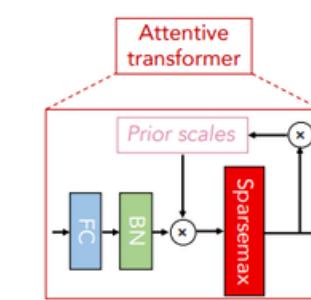
(a) TabNet encoder architecture



(b) TabNet decoder architecture

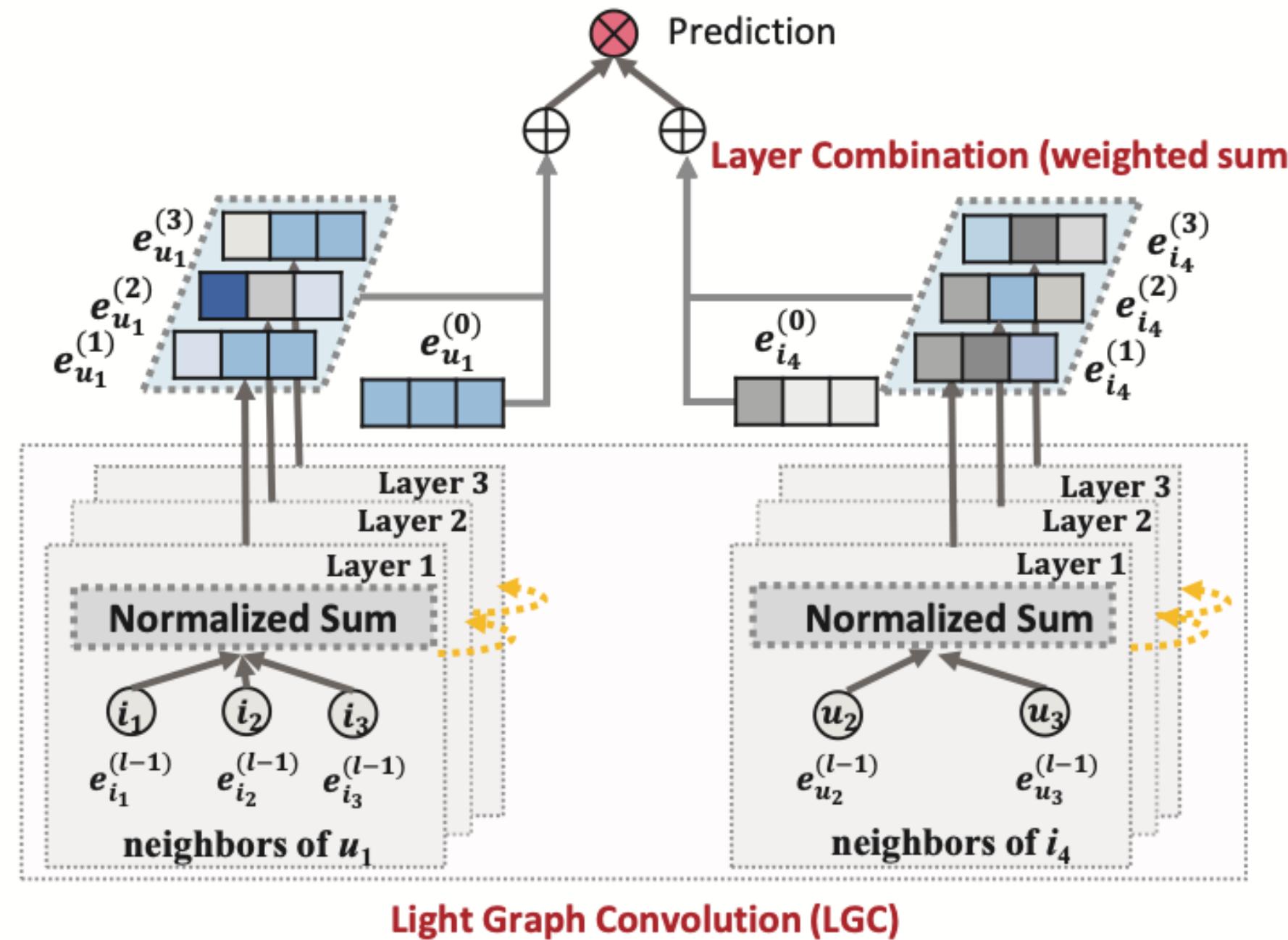


(c) Feature transformer



(d) Attentive transformer

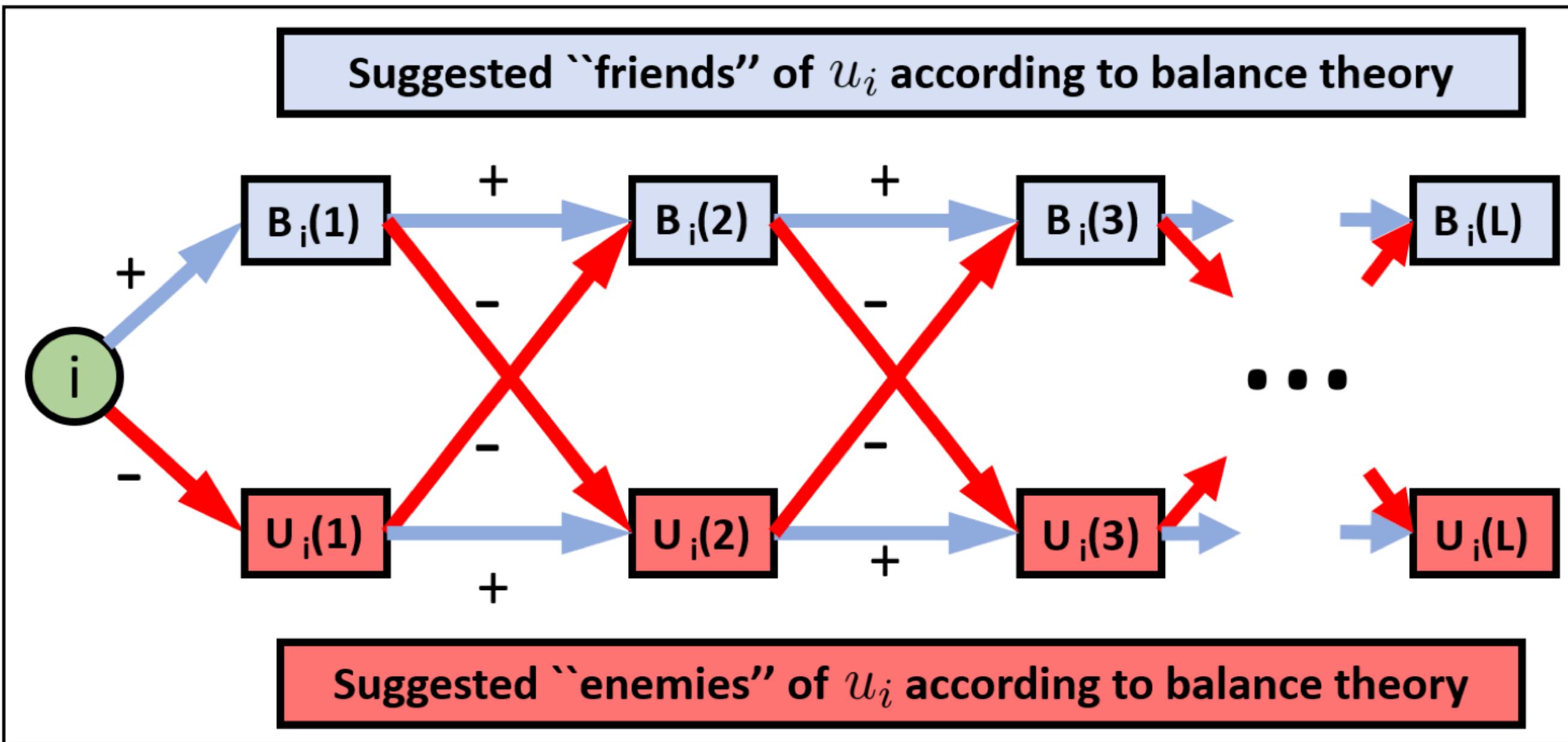
LightGCN(Graph)



NGCF의 feature transformation matrix와
non-linear activation function 제거

**우리의 데이터는 User - item Sign edge를 가지지만,
LightGCN은 Edge의 부호를 고려하지 않잖아 !?
-> Edge의 부호를 고려한 모델을 사용해보자 !**

SGCN(Graph)



“Balance Theory 를 이용해, Negative Positive Edge를 학습하는 모델”

**생각해보니 추천 한정, Layer 에 붙는 activation function과,
node feature Transform weight는 학습에 방해가 된다고 했지 ?
그럼 두가지를 제거 한 Light SGCN을 만들어 보자 !**

SGCN + LightGCN

LightGCN의 Light Cone layer

-> feature transformation matirx 이 제거된 SGCNconv로 대체

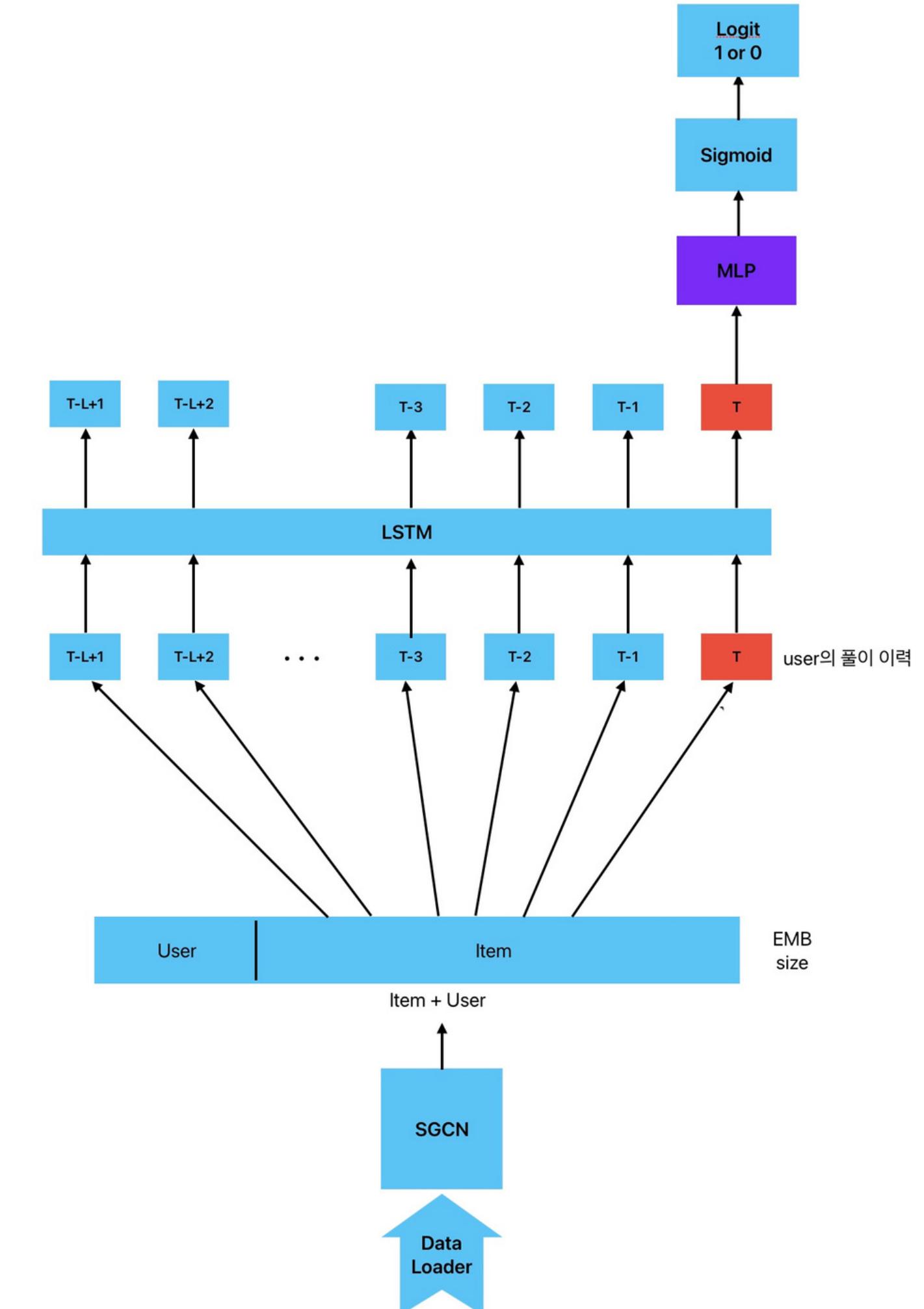
=> but 유의미한 성능 향상이 X
-> 폐기

사회적 네트워크에서 적용하는 Balance Theory가
적용되지 않는 듯 하다 (balance triangle을 세웠어야 하나 ..)

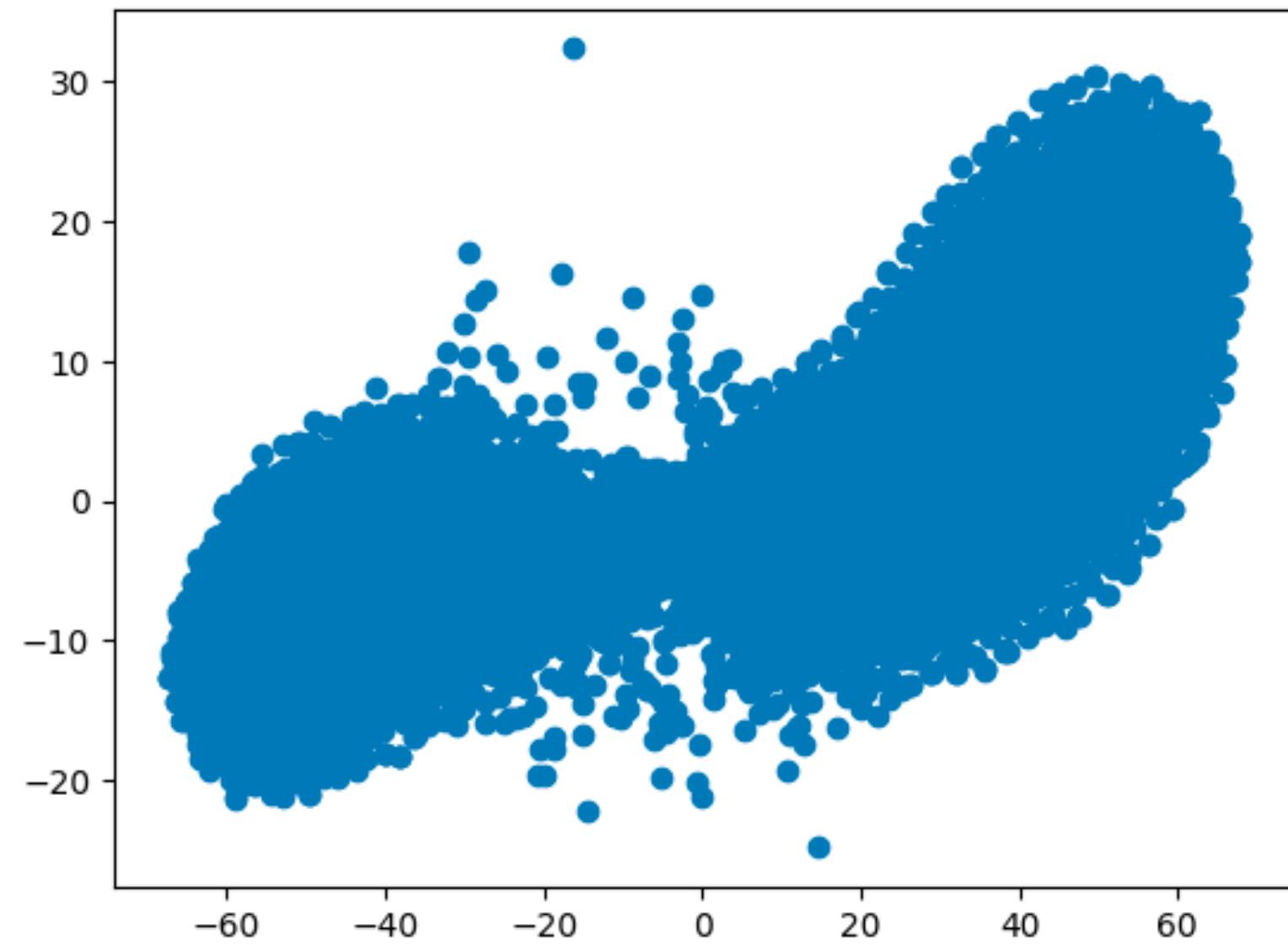
혹시 그래프도 시간 정보를 고려할 수 있지 않을까 !?

LightGCN + LSTM

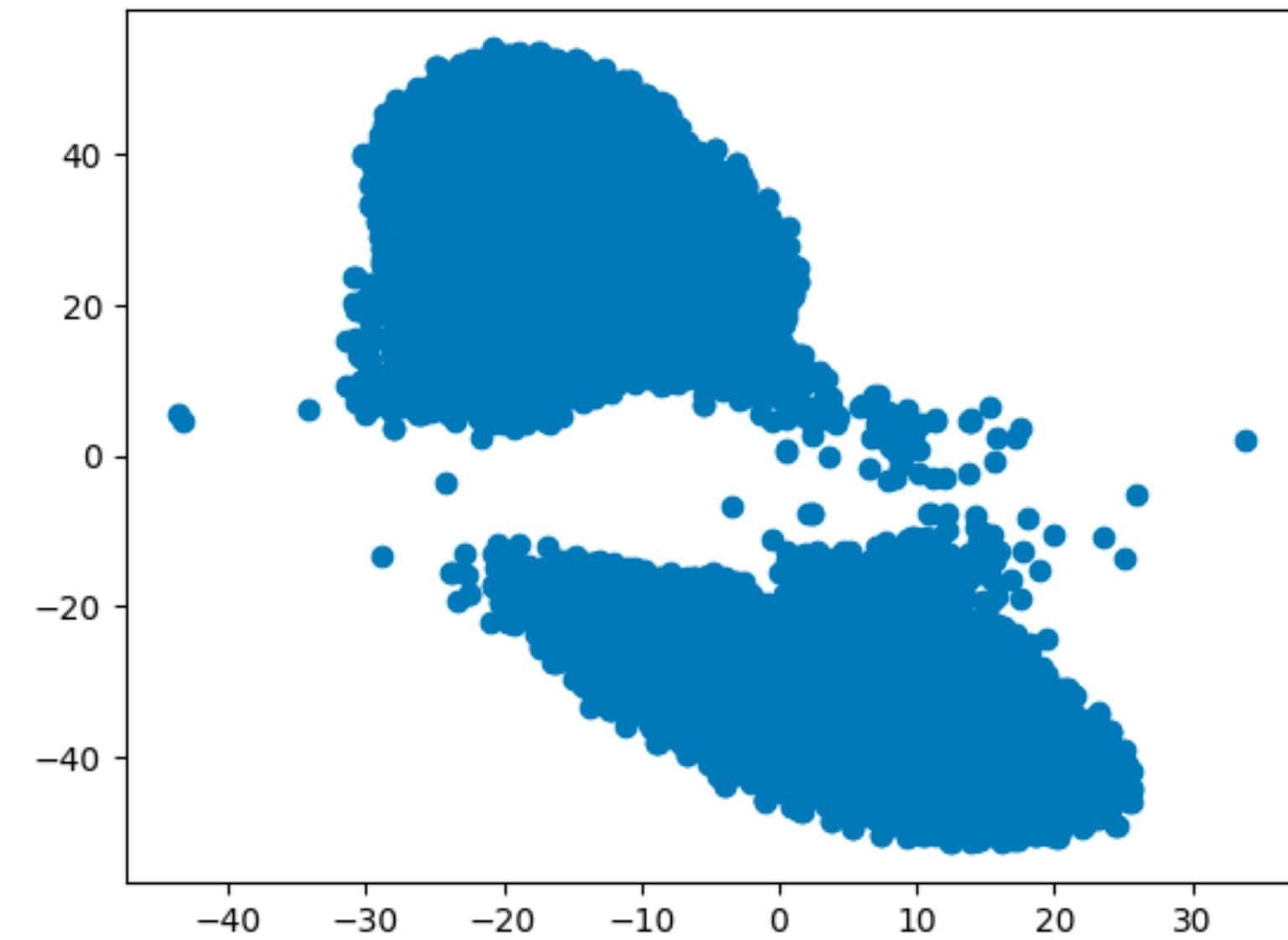
만들어진 Item Node Feature를
User로 그룹화 하여
Sequence Data로 변경 후
LSTM에 태운 후, MLP에 태우자!



생각보다, Node 클러스터링이 안 되네 ..
T-sne로 찍어보니 그룹이 나눠지지 않아 ..



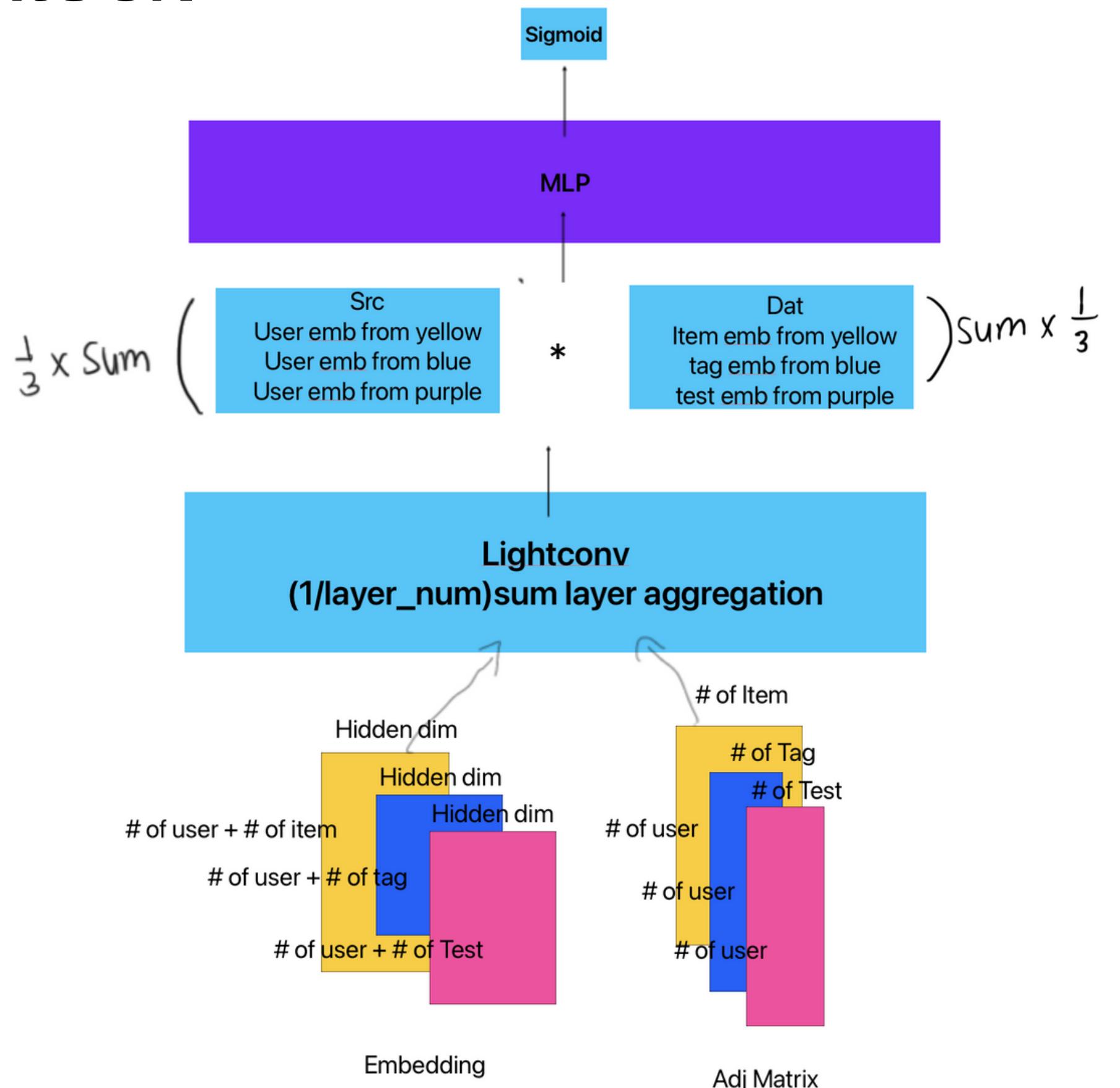
Item Node T-sne



User Node T-sne

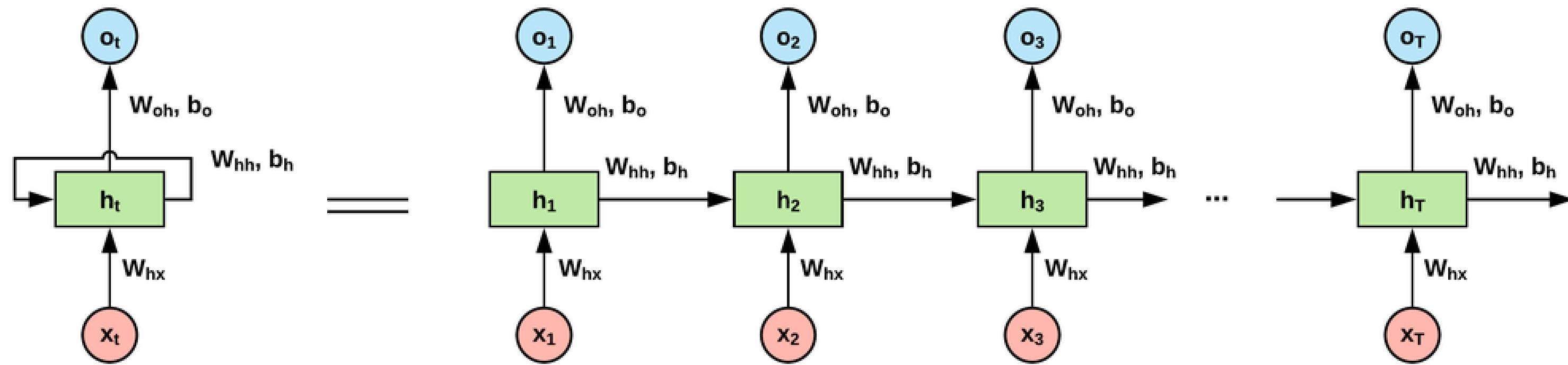
**차라리, Edge Type을 3개를 가지고,
Node type을 4개를 가지는
heterogeneous 한 그래프 모델을 만들까?**

3 - stacked LightGCN

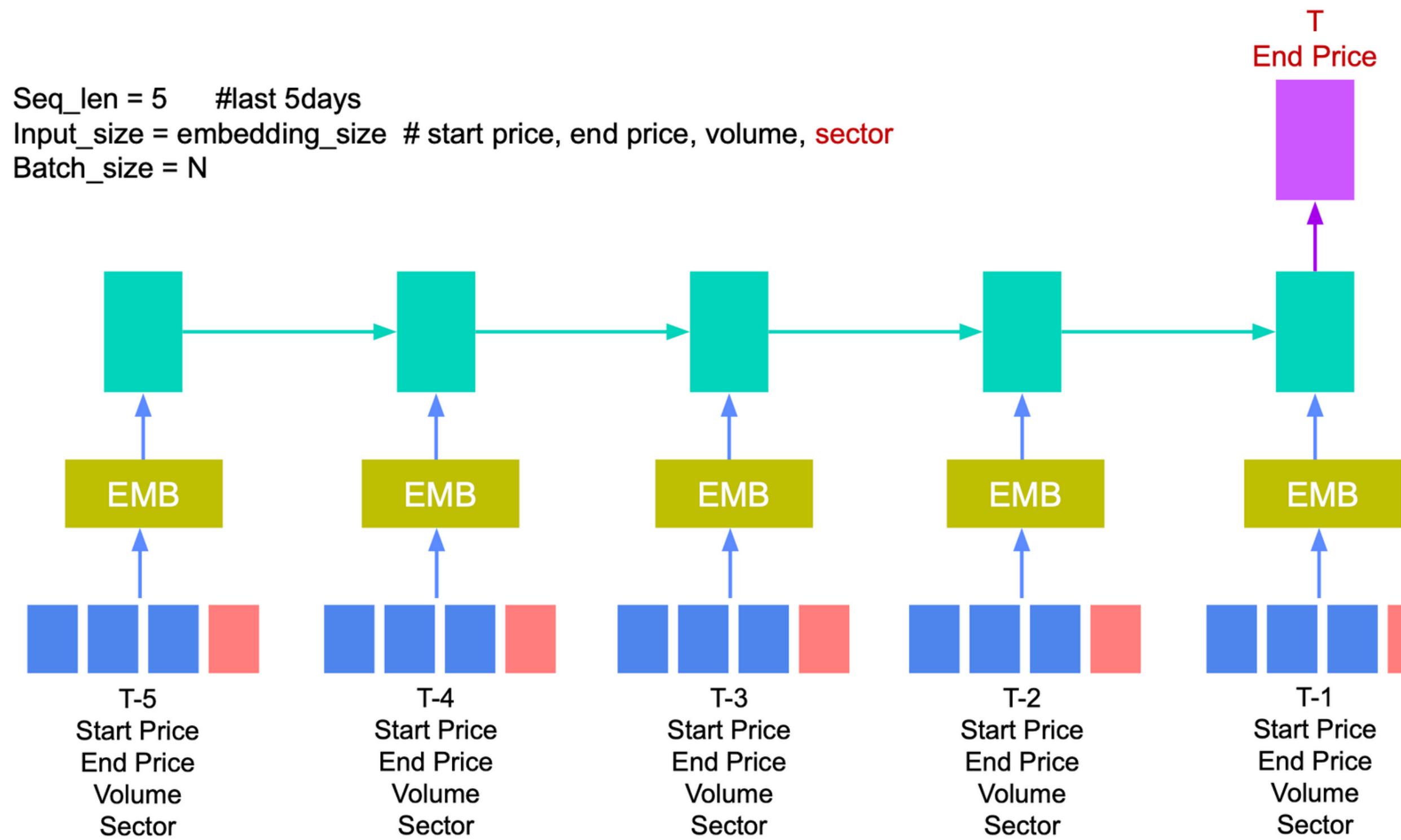


Sequential Data

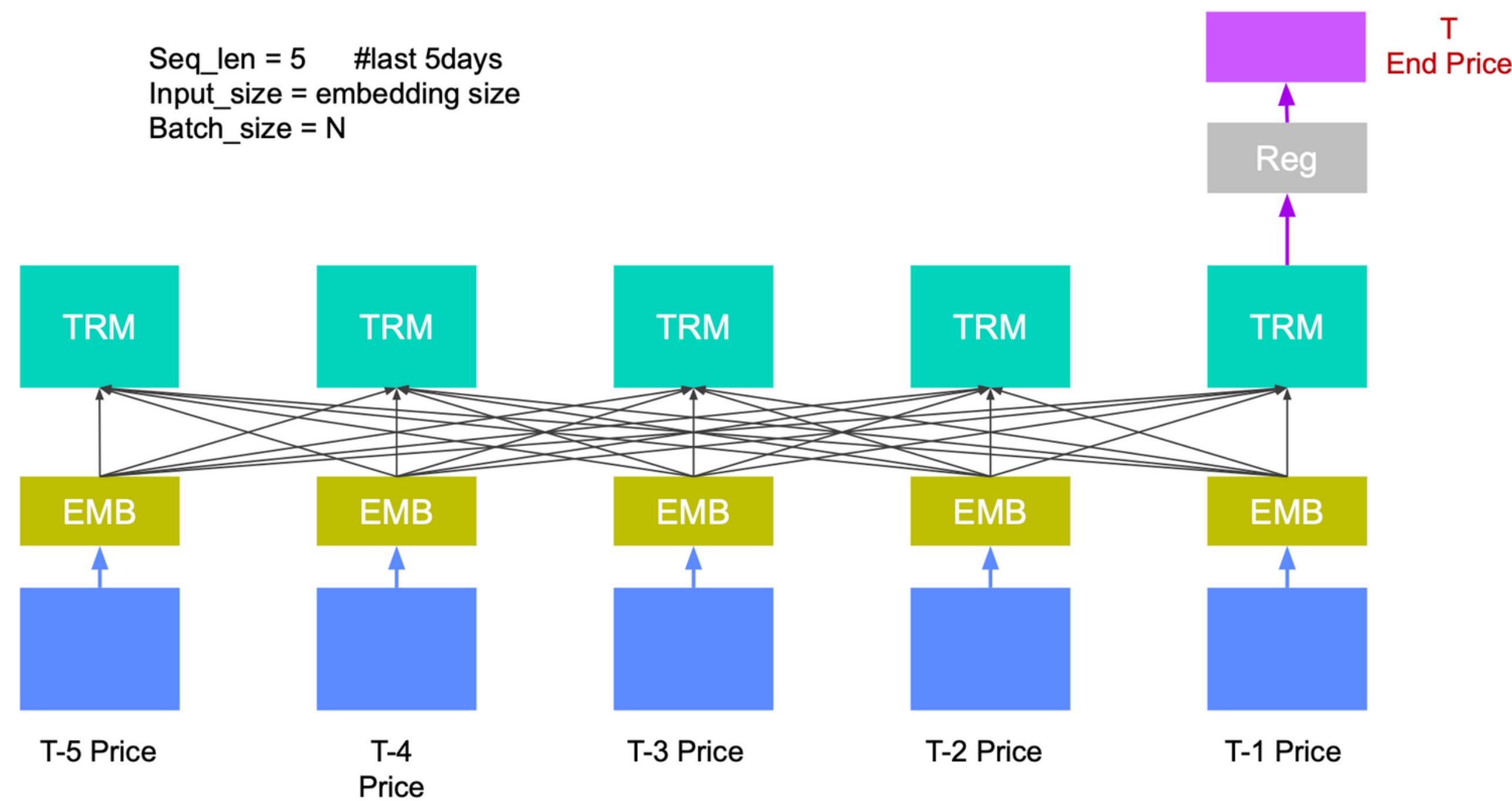
RNN



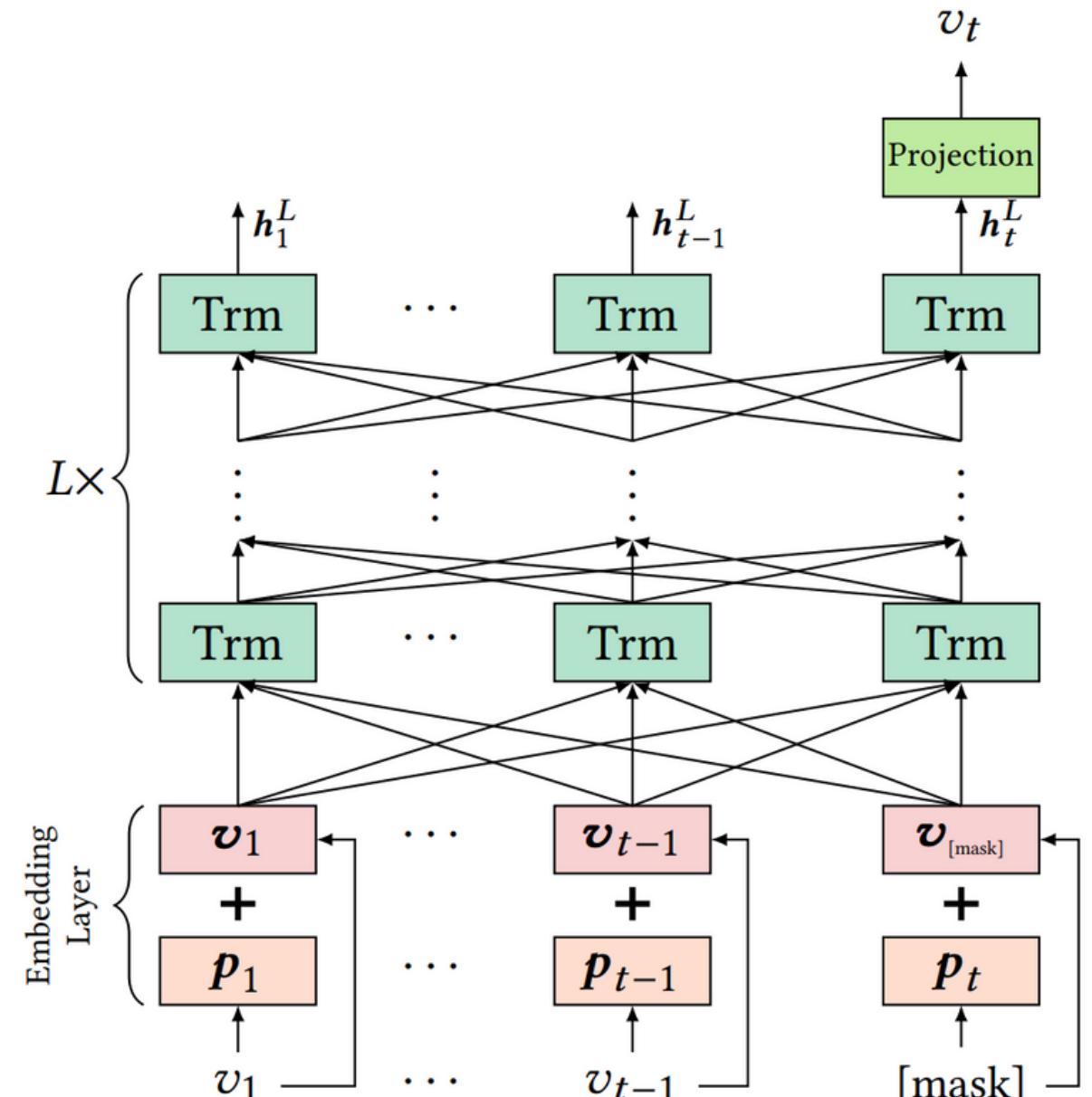
LSTM



Transformer



BERT4Rec



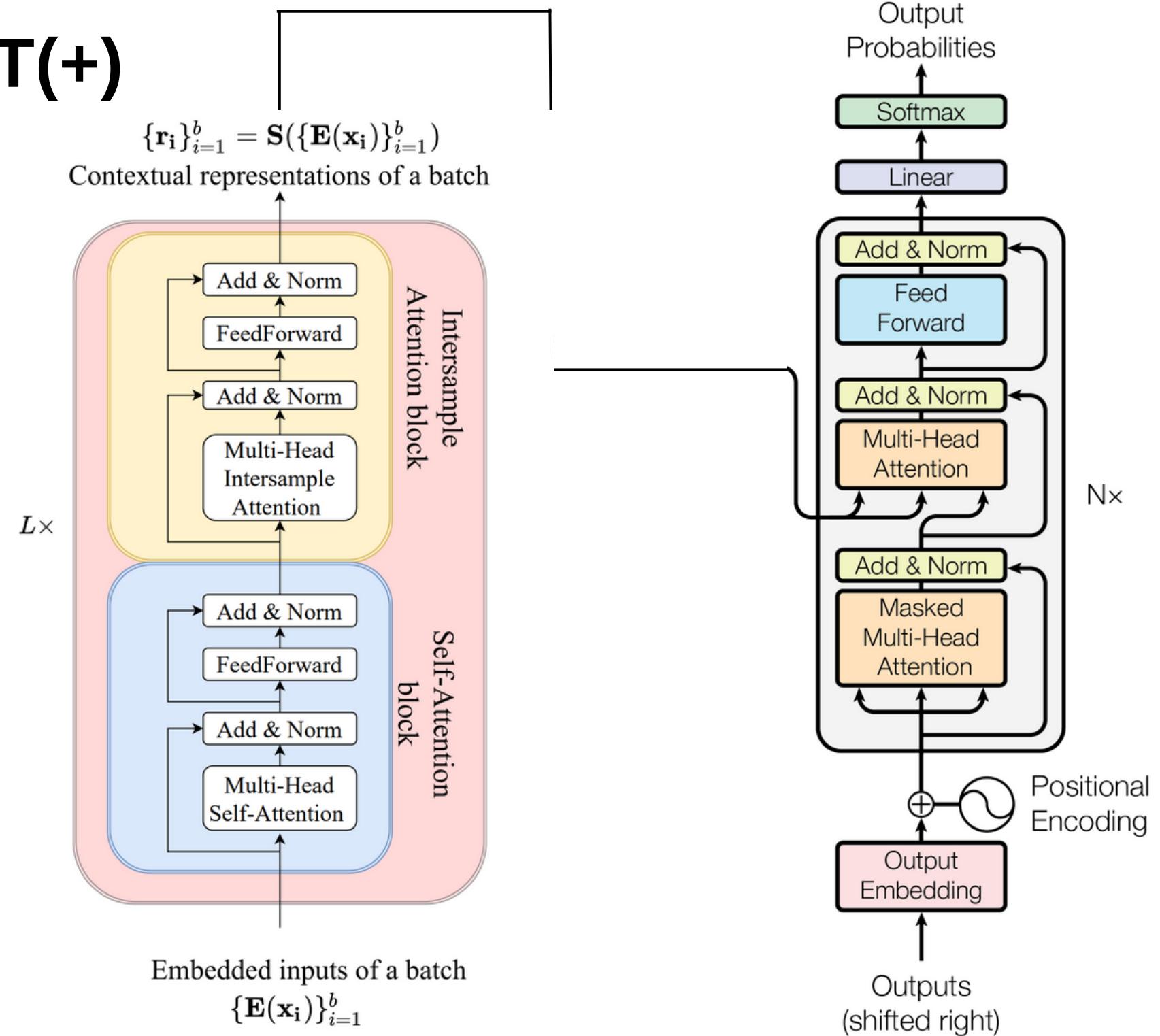
(b) BERT4Rec model architecture.

BERT4Rec은 여러 변수를 고려
양방향 학습을 통해 sequence data에서
사용자 특성 위주로 학습

✓ 사용자 특성을 고려하면
정답 여부 예측을 더 잘할 수 있지 않을까?

하지만 교육 분야에서 예측을 위해 사용하는 모델들은
K번째 문제를 풀 때 K-1의 문제 정보만 알고 있다고 가정
-> 단방향 학습
∴ 폐기 !!

SAINT(+)



들어가는 feature:

- Exercise ID
- Exercise category
- position Embedding
- ✓ 문제 및 시험 정보

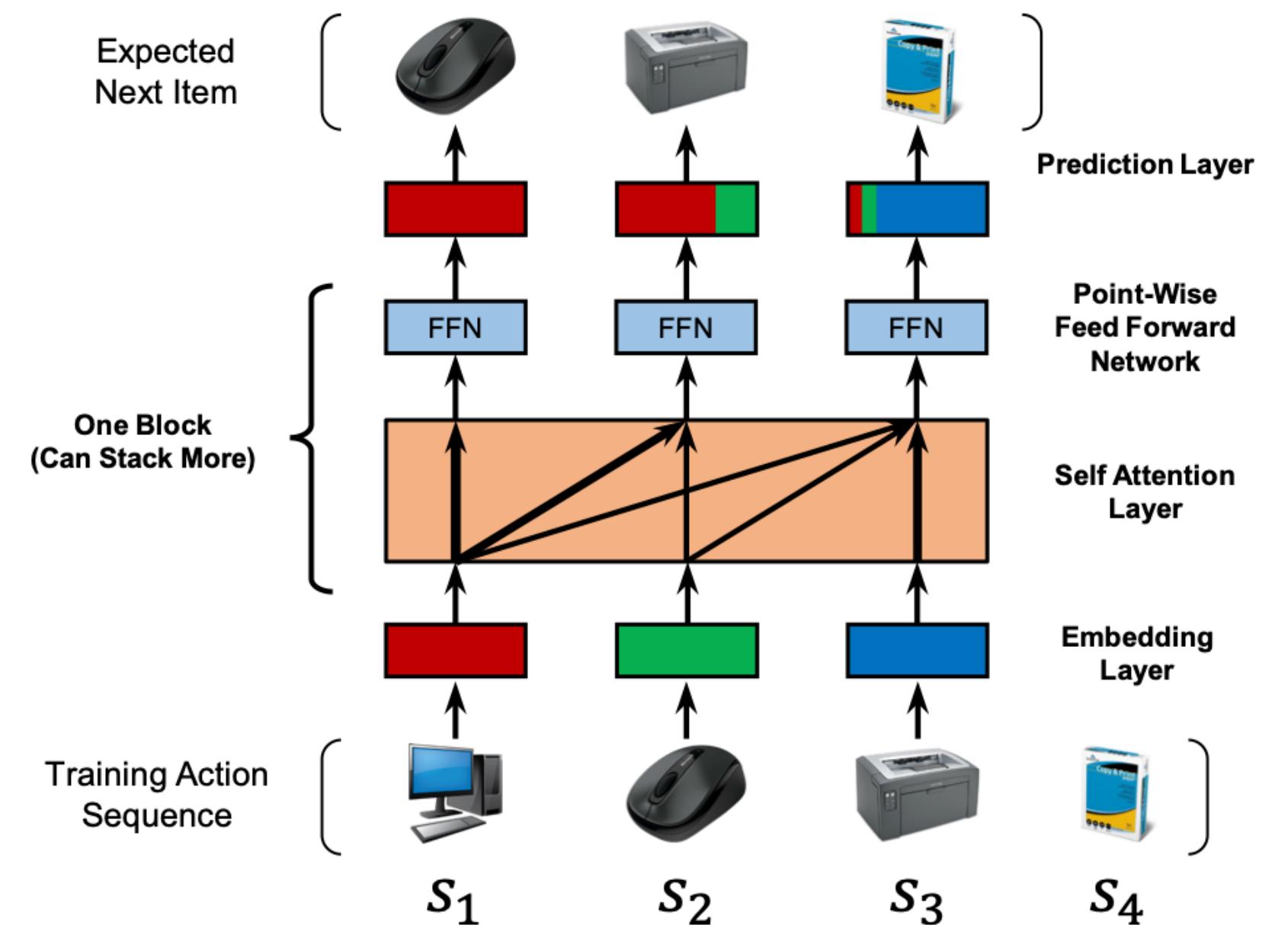
들어가는 feature:

- position Embedding
- Response
- ✓ 사용자 및 시간 정보

- ✓ K번째 문제를 풀 때 K-1개의 문제 관련된 정보만 알고 있음
- ✓ 현재 지점(K)는 앞의 Sequence(K-1)에만 의존하도록 제한
- ✓ 기존 Transformer와 다르게 Batch에 대해 Intersample Attention 적용

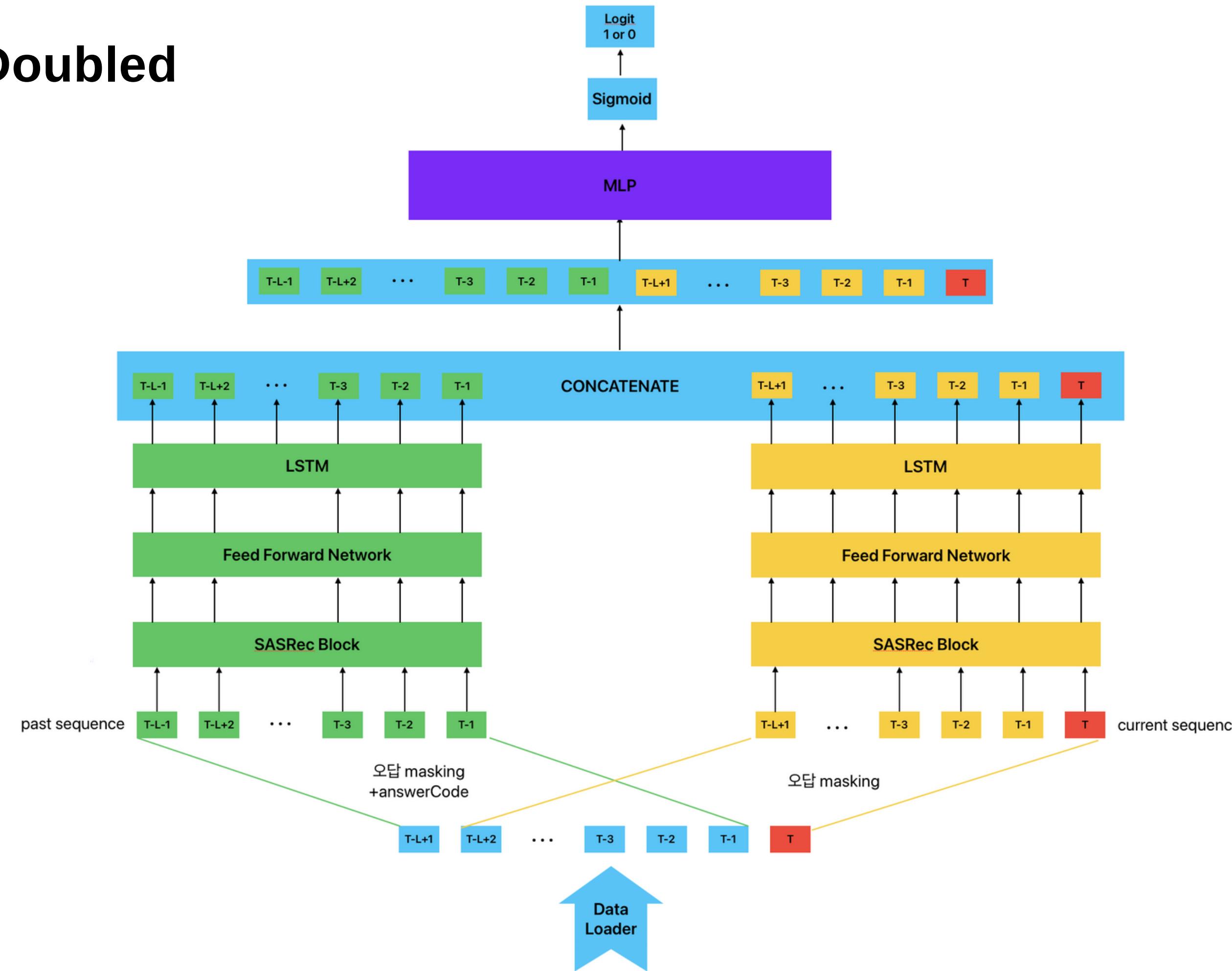
--> OOM 으로 사용 불가 !

SASRec

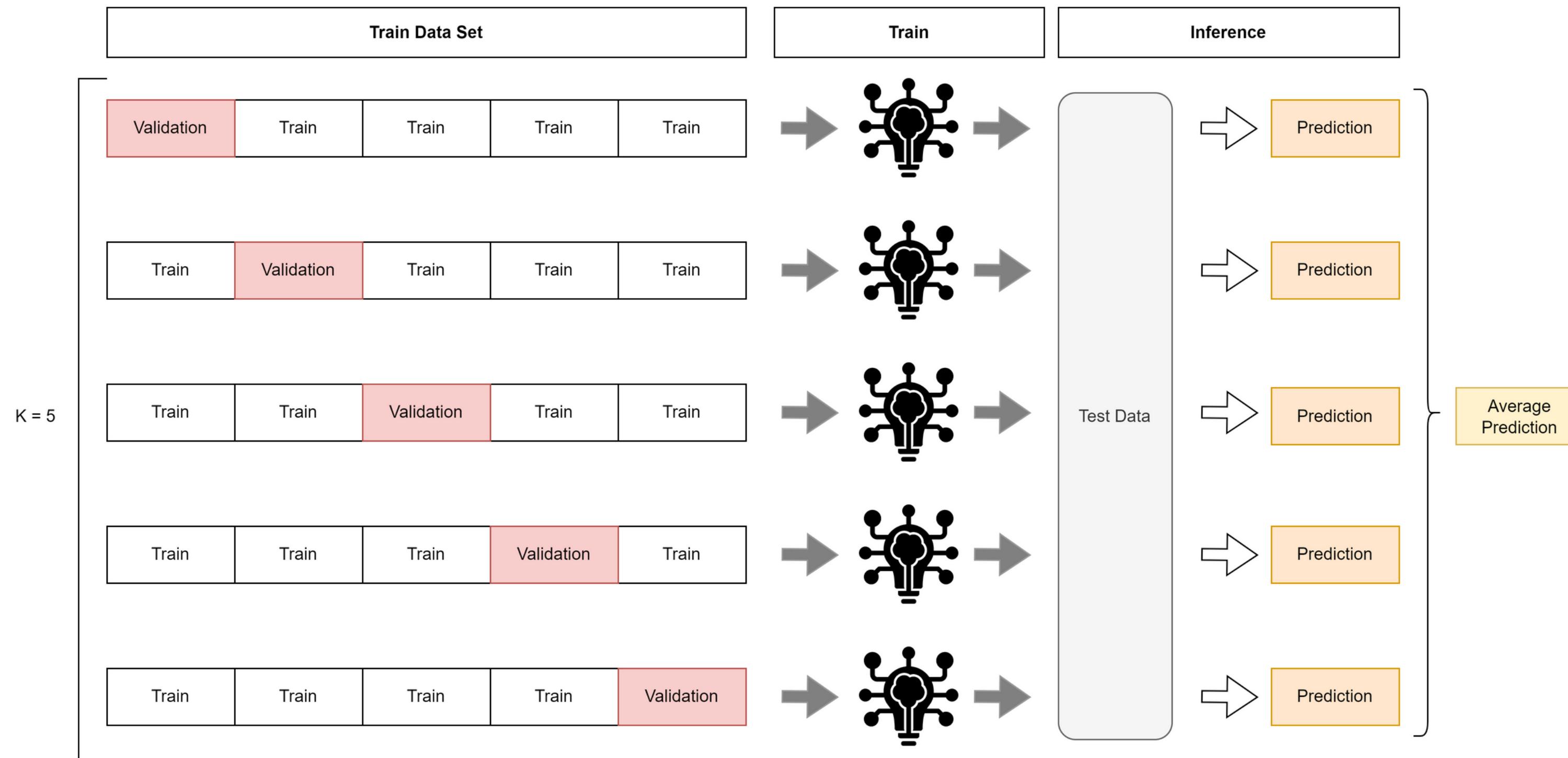


뒤 문항을 마스킹 처리하고 학습하니, 성능이 꽤 괜찮네 ?

SASRec - Doubled



Out Of Fold



단일 모델 학습 결과

Model	Public		Private	
	AUROC	ACC	AUROC	ACC
SASRec_v3	0.8314	0.7554	0.8544	0.7849
SASRec_v1	0.8261	0.7688	0.8546	0.7796
SASRec_v2	0.8304	0.7554	0.8581	0.7796
Transformer	0.8056	0.7366	0.8436	0.7715
MF	0.7973	0.8432	0.7177	0.7715
SAINT(-)	0.7772	0.8160	0.7339	0.7312
SASRec + 오답패턴	0.8105	0.7339	0.8407	0.7473
TabNet	0.7720	0.7417	0.7016	0.6989
LSTM	0.7197	0.7406	0.6398	0.6774
Bert	0.6875	0.7506	0.6559	0.6747

Model	Public		Private	
	AUROC	ACC	AUROC	ACC
CatBoost	0.8114	0.7581	0.8017	0.7392
LightGBM	0.8111	0.7446	0.8134	0.7419
XGBoost	0.8055	0.7339	0.8065	0.7392
AutoGluon	0.8064	0.7446	0.8258	0.7500
LightHgcn	0.7861	0.7043	0.8113	0.7312
LightGcn	0.6817	0.5941	0.6250	0.5672
SGCN	0.7644	0.7508	0.6962	0.6935
LightSGCN	Nan	Nan	Nan	Nan

*SASRec_v1: feature selection1 cat emb : num emb = 1 : 1

*SASRec_v2: feature selection2 cat emb : num emb = 1 : 1

*SASRec_v3: feature seleciton2 cat emb: num emb = 1 : 2

양상블 결과

양상블	Public		Private	
	AUROC	ACC	AUROC	ACC
SASRec_v3(0.395) + SASRec_v1(0.2325) + Catboost(0.1775) + LGBM(0.065) + HLGCN(0.065) + MF(0.065)	 0.8330	0.7581	0.8550	0.7957
SASRec_v3(0.4) + SASRec_v1(0.15) + CatBoost(0.15) + LGBM(0.1) + HLGCN(0.1) + MF(0.1)	0.8327	0.7608	0.8553	0.7984
SASRec_v3(0.5) + SASRec_v2(0.5)	0.8316	0.7473	0.8568	0.7823
SASRec_v3(0.5) + SASRec_v1(0.5)	0.8315	0.7634	0.8564	0.7876
SASRec_v2(0.3) +SASRec_v1(0.3) + CatBoost(0.2) + LGBM(0.2)	0.8307	0.7634	0.8523	0.7957
SASRec_v2(0.3) +SASRec_v1(0.3) + CatBoost(0.4)	0.8306	 0.7715	0.8488	0.7849
SASRec_v3(0.3) + SASRec_v1(0.3) + CatBoost(0.25) + LGCN(0.15)	0.8304	0.7581	0.8554	0.7930

양상블 결과

양상블	Public		Private	
	AUROC	ACC	AUROC	ACC
SASRec_v3(0.395) + SASRec_v1(0.2325) + Catboost(0.1775) + LGBM(0.065) + HLGCN(0.065) + MF(0.065)	0.8330	0.7581	0.8550	0.7957
SASRec_v3(0.4) + SASRec_v1(0.15) + CatBoost(0.15) + LGBM(0.1) + HLGCN(0.1) + MF(0.1)	0.8327	0.7608	0.8553	0.7984
SASRec_v3(0.5) + SASRec_v2(0.5)	0.8316	0.7473	0.8568	0.7823
SASRec_v3(0.5) + SASRec_v1(0.5)	0.8315	0.7634	0.8564	0.7876
SASRec_v2(0.3) +SASRec_v1(0.3) + CatBoost(0.2) + LGBM(0.2)	0.8307	0.7634	0.8523	0.7957
SASRec_v2(0.3) +SASRec_v1(0.3) + CatBoost(0.4)	0.8306	0.7715	0.8488	0.7849
SASRec_v3(0.3) + SASRec_v1(0.3) + CatBoost(0.25) + LGCN(0.15)	0.8304	0.7581	0.8554	0.7930

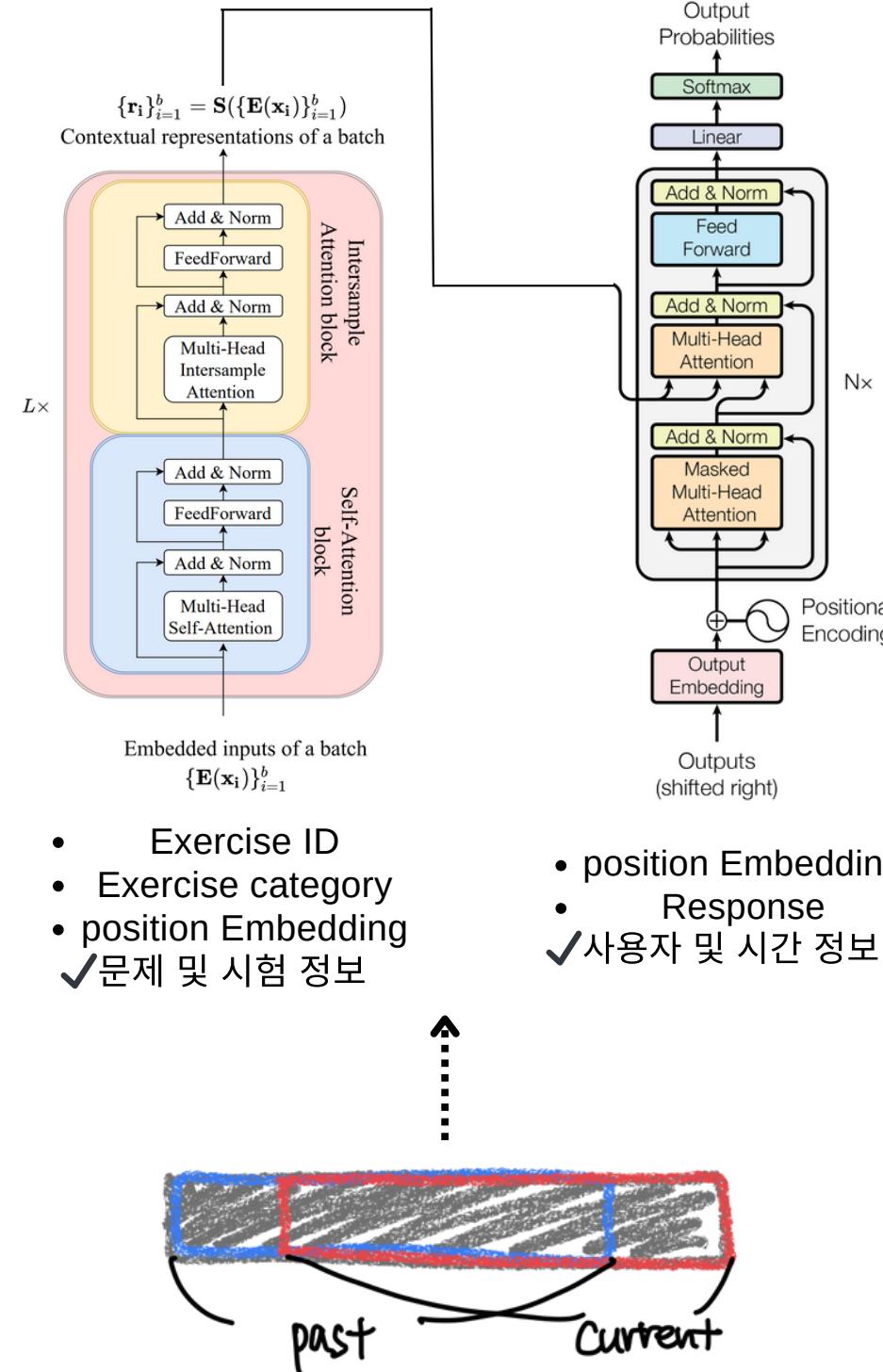
Public

1	RecSys_02조		0.8330	0.7581	143	7h
---	------------	---	--------	--------	-----	----

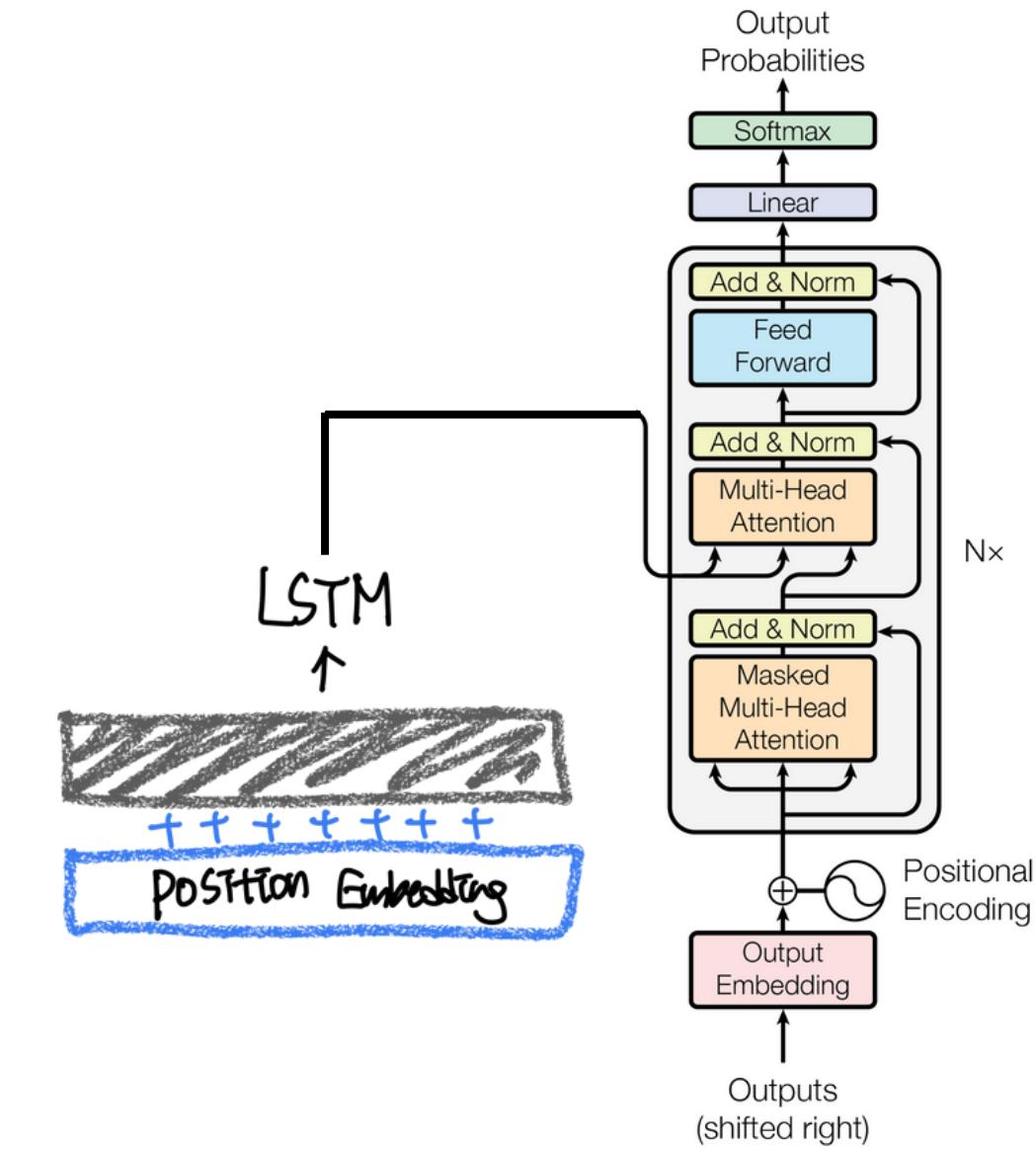
Private

1	RecSys_02조		0.8550	0.7957	143	7h
---	------------	---	--------	--------	-----	----

추가로 구상한 모델덜 ... (빛을 보지 못하고 크윽..)



SAINT | 에 position embedding
빼고 past/current 나눠 학습?



문제 및 시험 정보와 사용자간의 상관관계를 학습할 수
있지 않을까?

- 문제 및 시험 정보로 만들어지는 K, V: sequence
에 주목하도록 position emb을 더하고 LSTM을
태우자!
- 사용자 및 시간 정보는 Q로!

+ α 이지만 다 쓰기 힘들어요...

잘한 점 & 아쉬웠던 점 & 배운 점

잘한 점

- 깃허브를 이용한 이슈 기반 협업을 원활히 진행하였다.
- 여러 모델들을 사용하였다.
- decesion tree model에 의존하지 않았다.

아쉬웠던 점

- 그래프 오버피팅을 잡지 못해서, 그래프의 끝을 보지 못한 것 같다.
- 강의에 소개된 내용을 모두 적용해보지 못해 아쉬움이 있다.
- 제출시 Description 작성하는 것으로 실험들을 단순하게 정리 하였지만, 제출수가 많아지고 나서부터 구분하기가 쉽지 않아지면서, 이후 과정에는 노션 템플릿을 반드시 사용하여 실험관리를 좀더 체계적으로 한다면 더 좋았을 것이다.

배운 점

- 의외로 간단하지만 성능이 좋은 모델들이 많다!
- 백지장도 맞들면 낫다!(양상을 학습)

궁금한 점

그래프 모델

- 오버피팅이 해결되지 않아 Val score와 public score의 점수 차이가 컸는데, 이걸 해결하는 방법이 있을까요?
 - 시도해본 것
 - 엣지 드랍
 - 드랍 아웃(레이어)
 - 얼리스탑
 - schedular

데이터 증강

- 유저별로 seq 길이를 15, window size를 10을 두고 데이터를 증강 하였더니, 오버피팅이 났습니다. 증강에 적절한 파라미터는 실험으로 찾아야 하는 걸까요?

피처 엔지니어링

- 특정 피처 추가 시, Val score 와 제출시 public score의 점수 차이가 컸는데 흔히 발생하는 원인이 있을까요?
- 대회에서는 피처 생성 및 선택 시 실험적으로 처리를 하였는데, 현업에서는 상당한 개수의 피처를 주로 어떻게 생성하고 선택하나요?

피치

바니

꾸눅

삥삥

미미

라쿤



탱크 보리 망치