Wide & Deep

Domain	RecSys
tag	
Publish year	2016
AI summary	The paper proposes the **Wide & Deep Learning** framework, combining the strengths of both linear models and deep neural networks for recommendation systems. The aim is to achieve a balance between memorization (effective in capturing particular feature interactions) and generalization (capable of generalizing to unseen feature combinations).

Summary

Memorization에 특화된 linear wide model + generalization에 특화된 non-linear deep model

Background & Motivation

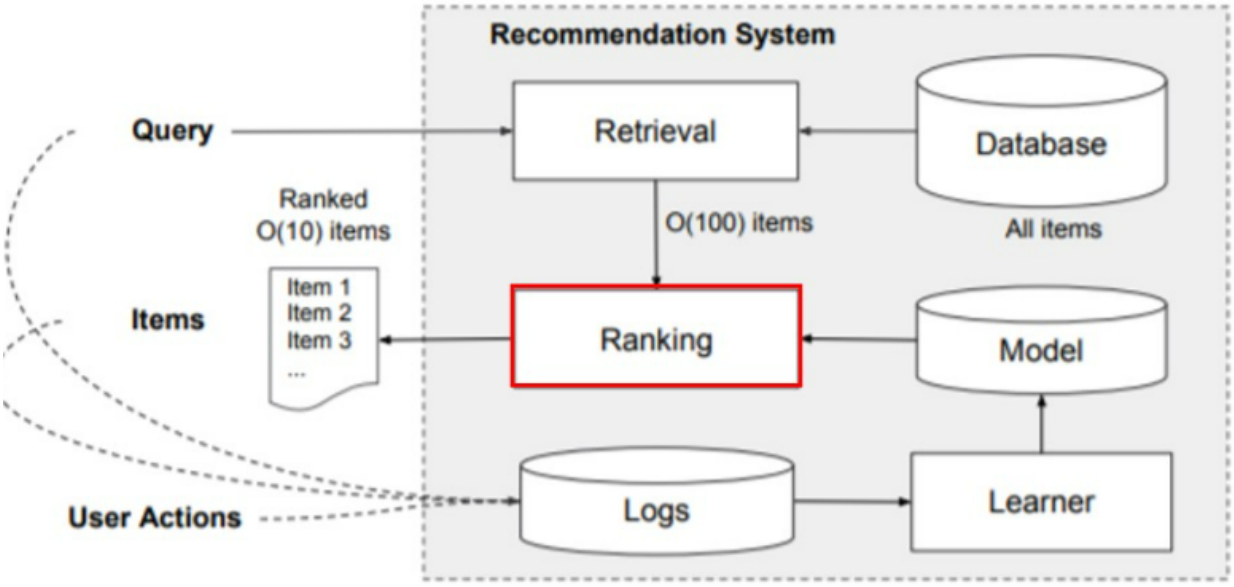
Linear model

- Regression or classification에 많이 쓰임
- feature간의 cross-product가 데이터 특징을 기억하는데 효과적
- 하지만, generalization에는 많은 feature engineering 필요

Deep neural network

- Unseen feature combination의 generalization에 효율적
    - Low-dimensional dense embedding으로 sparse feature 학습

Recommendation system

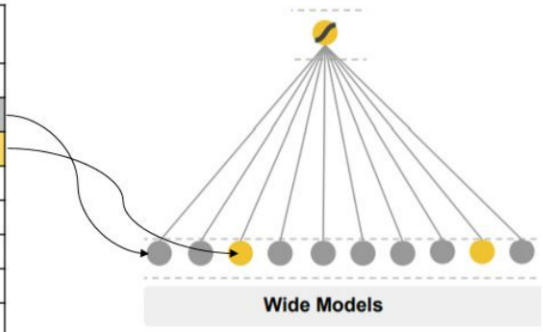


1. User search query → DB에서 적합한 후보 item retrieval
2. Ranking통해 정렬
    a. Wide & Deep 쓰임

Methodology

Wide component

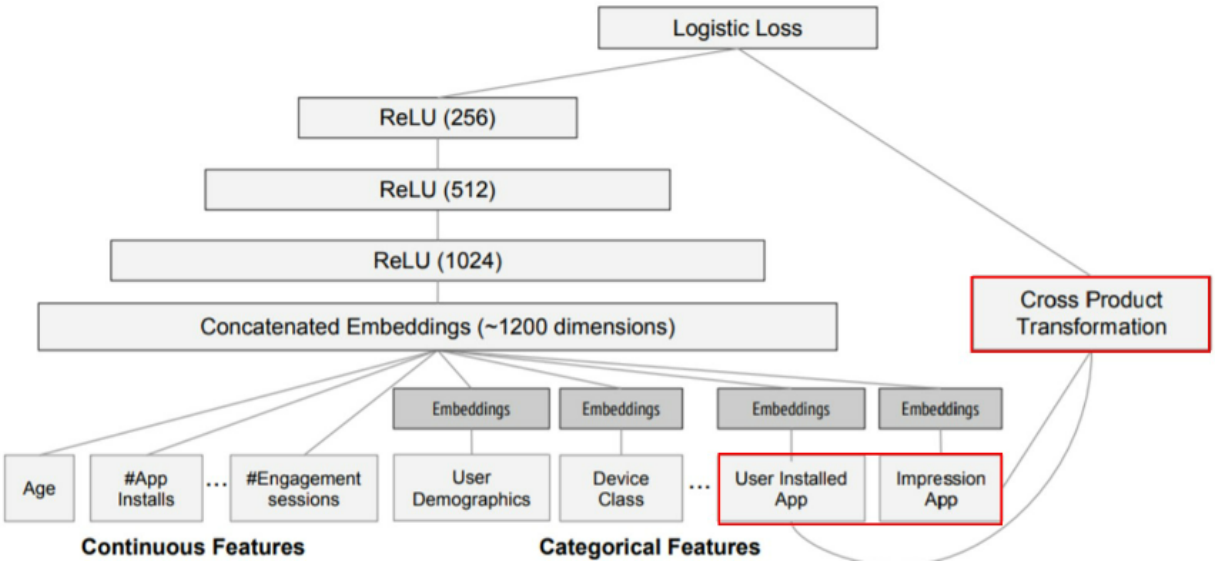
Install	Impression	(Install, Impression)	Install x Impression
A	A	(1,1)	1
A	B	(1,0)	0
A	C	(1,1)	1
B	A	(1,1)	1
B	B	(1,0)	0
B	C	(1,1)	1
C	A	(0,1)	0
C	B	(0,0)	0
C	C	(0,1)	0



- 유저가 설치한 앱 feature, 열람한 앱 feature를 input으로 사용

$$\text{user\_install\_app} = [A, B]$$
$$\text{user\_impression\_app} = [A, C]$$

- 설치한 앱과 열람한 앱의 cross-product로 interaction을 표현
  - e.g. $(A, C) = (1, 1)$
  - 앱이 3가지일 경우, 총 9가지의 경우의 수, 1이되는 경우 4가지
- 1이되는 모든 경우를 학습하기에 memorization, 유저 취향 반영된 niche combination 학습 강함
- 0이 되는 pair는 학습 불가

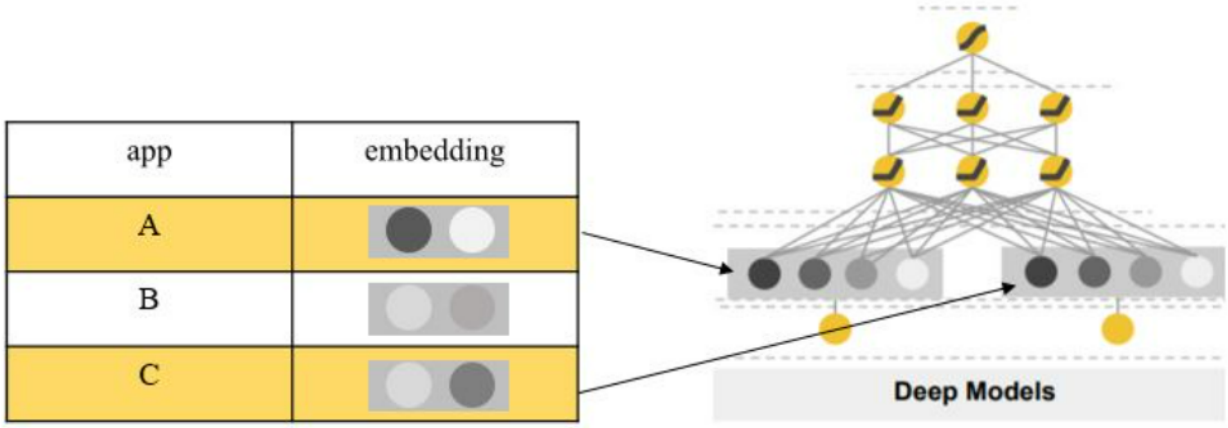


- user installed app, impression app을 cross-product하여 input으로 사용

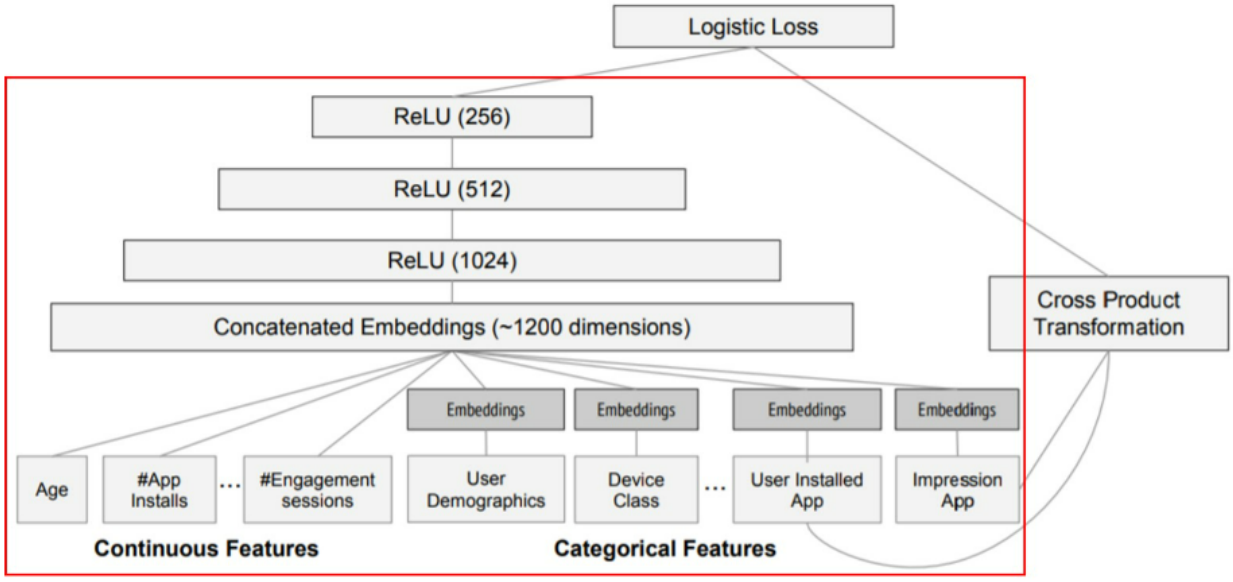
$$y = \mathbf{w}^T \mathbf{x} + b$$

$$where, \quad \mathbf{x} = [x_1, x_2, \ldots, x_d]$$
$$\mathbf{w} = [w_1, w_2, \ldots, w_d]$$
$$b = \text{bias}$$

Deep component



- 각 앱을 동일한 embedding 공간에 표현
  - pair가 없는 관계도 표현되어 학습 가능함
  - Niche combination의 경우 다른 유저에서 등장 적어 충분한 표현 정보 붗족
    - 이 앱이 다른 앱들과의 관계를 제대로 표현하지 못한 embedding vector 가질 가능성 큼
- 두 feature간 interaction이 multi-layer의 non-linear 공간에서 표현
  - generalization에 강함
  - 희소한 앱들은 학습 어려워 관계없는 아이템 추천 가능

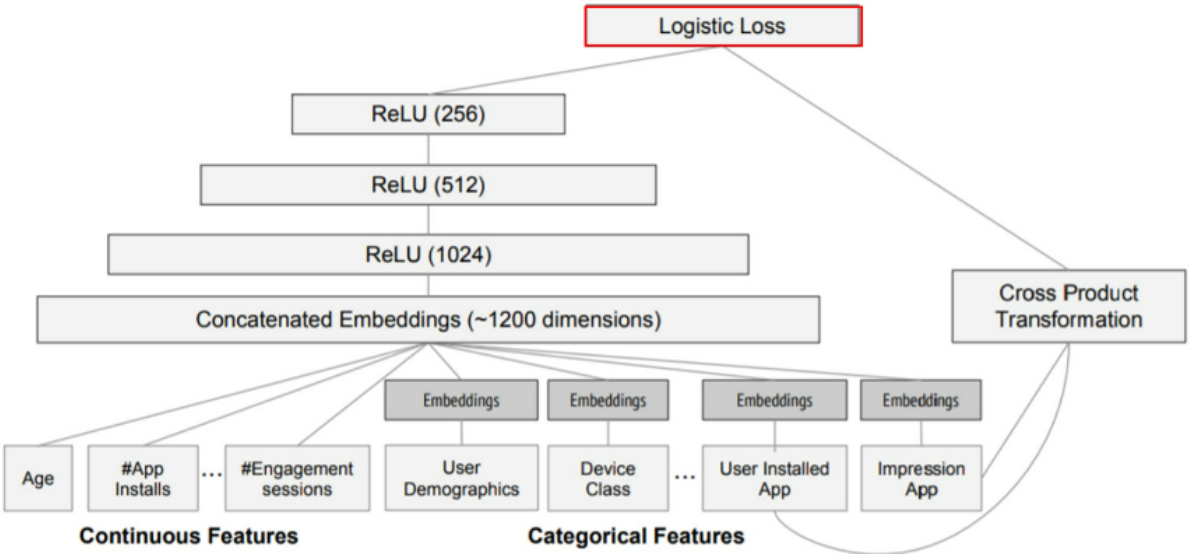


- Continuous feature, embedded categorical feature을 concat한 결과를 input으로
- l번째 layer가 다음 계산 수행

$$\mathbf{a}^{(l)} = f(W^{(l-1)}\mathbf{a}^{(l-1)} + \mathbf{b}^{(l-1)})$$

  - f: activation function (ReLU)
  - W: weight matrix
  - b: bias

Wide & Deep



Joint training

- Ensemble과 다르게 output의 gradient를 wide와 deep에 모두 동시에 propagation하여 학습
- Prediction

$$p(Y=1|\mathbf{x}) = \sigma(\mathbf{w}_{wide}^T[\mathbf{x}] + \mathbf{w}_{deep}^T a^{(l_f)} + b)$$

Questions

1. joint 방식 학습을 어떻게 하는거임? joint에서 동시에 각 wide와 deep 파트에 대해서 backpropagate하는데, 이러면 각각 wide와 deep의 학습이 서로 영향을 줄 수 있는거임? 이게 ensemble 방식이랑 무슨 차이가 있는지 모르겠

- 쉬운 질문인데 이해 안가서
- ensemble은 학습을 개별적으로 해주고, 추론할때 합쳐 쓰고,
- joint는 동시에 둘을 backprogate
  - Loss 값을 구할 때 둘 다 사용하는 것 같음

- 확률 구할때 둘 값의 합을 쓰니까?
- 이게 맞는듯