DNN 4 Youtube

Domain	RecSys
tag	Candidate gen & Ranking, Surrogate problem
Conference / Journal	ACM RecSys
Publish year	2016
정리 날짜	@2024년 2월 1일
AI summary	Youtube의 추천 시스템에 DNN을 적용한 방법을 제안한다. 이를 위해 두 개의 신경망을 사용하여 후보 생성과 랭킹을 수행한다. 후보 생성 단계에서는 사용자 히스토리와 행동 패턴을 고려하여 수백 개의 후보 동영상을 선정하고, 랭킹 단계에서는 후보 동영상에 점수를 매겨 최종 추천을 수행한다. 이 논문은 대규모 데이터셋에서 개인화된 추천을 가능하게 하며, 다른 소스에서 생성된 후보 동영상도 함께 사용할 수 있다는 장점을 갖고 있다.
AI key info	DNN, Youtube RecSys, Scale, Freshness, Noise, Candidate generation, Ranking, Pros, Offline metric, A/B testing, Recommendation as Classification, Negative sampling, Model architecture, Heterogeneous Signals, Label and Context Selection, Surrogate problem, Impression data, Feature Representation, Embedding Categorical Features, Normalizing Continuous Features, Modeling Expected Watch Time, Experiment

Summary

DNN을 이용하여 Youtube video recommendation을 함

- Candidate generation & ranking으로 나눠서 접근
- 많은 signal들을 이용해 예측 정확도 향상

Recommendation을 Classifying a future watch의 문제로 변형해 접근

Background & Motivation

DNN 적용한 Youtube RecSys를 해보자!

Youtube RecSys의 challenge

- Scale: user base와 video가 너무 많음
  - Highly specialized distributed learning algorithm & efficient serving system 필요
- Freshness: dynamic corpus: 새로운 video가 계속 추가됨
  - RecSys가 responsive 하고 new/old content를 balancing 해야 함
- Noise: ground truth 추출의 어려움
  - sparsity & unobservable external factors 때문에 prediction 어려움
  - Implicit feedback noise 많음
  - 잘 정의한 framework 없이는 Content의 metadata가 poorly structured

Methodology

Overview

Two neural network로 구성

- Candidate generation
  - 사용자간 유사성 고려해 유저의 행동 패턴 파악
  - User history input으로 받아 수 백개 가량의 video를 candidate으로 선정
    - User와 관련성 높은 video로 선정
  - Collaborative filtering

        - 개별 사용자 뿐 아니라 다른 유저들과의 유사성 정보 활용함
            - Individual personalization아니라 broad personalization 함
- Ranking
    - Candidate에 점수 매겨 상위 비디오 최종 추천
        - Fine-level representation으로 Candidate 사이의 relative importance 파악
- Pros
    - Large corpus(집단) handle 가능
        - 큰 규모의 동영상 set에서 personalized된 적은 영상 추천 가능
    - Blending candidate generated by other sources
        - 다른 방법으로 만든 candidate 또한 사용 가능
- Offline metric + A/B testing
    - offline metric 으로 성능 평가하며 개발했지만, Online A/B testing으로 성능 추가 검증함

Candidate generation

Recommendation as Classification

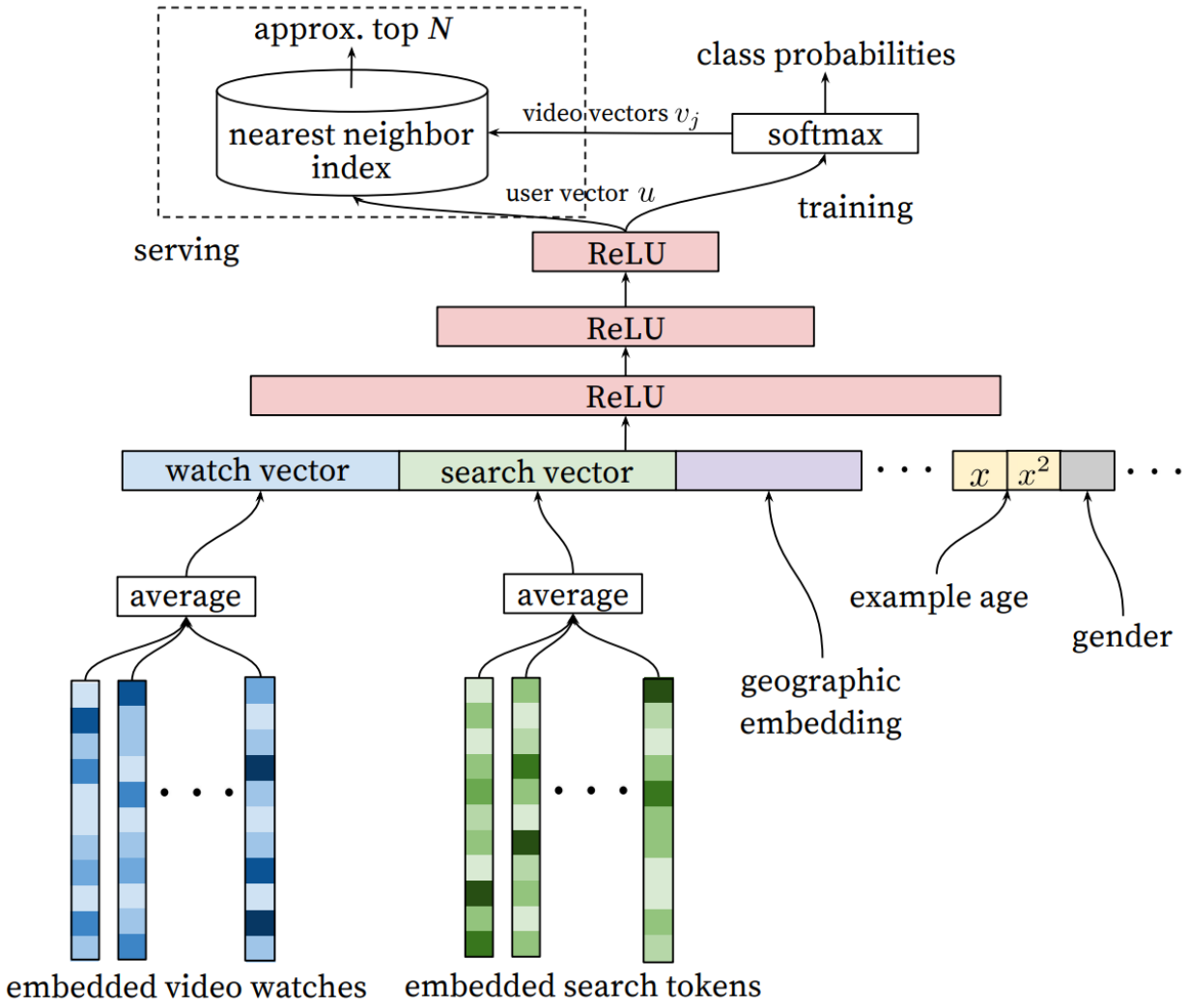
- Multiclass classification으로 추천함
    - 다음에 뭘 볼지 예측: User&context / candidate embedding u, v를 학습

$$P(w_t = i|U, C) = \frac{e^{v_i u}}{\sum_{j \in V} e^{v_j u}}$$

        - w_t watch: time t에 video i를 봄
        - under user U and context C
    - Embedding이 sparse entity와 dense vector의 mapping
        - $u \in \mathbb{R}^N$: high-dim embedding of user & context pair

        - $v_j \in \mathbb{R}^N$: high-dim embedding of candidate video
    - implicit data로만 학습
        - explicit이 너무 sparse해서
- Negative sampling
    - Positive or Negative sample만 사용해서 전체 class를 다 계산하지 않아 속도 증가
    - 가능한 많은 sample한다면 성능에 문제 X

Model architecture

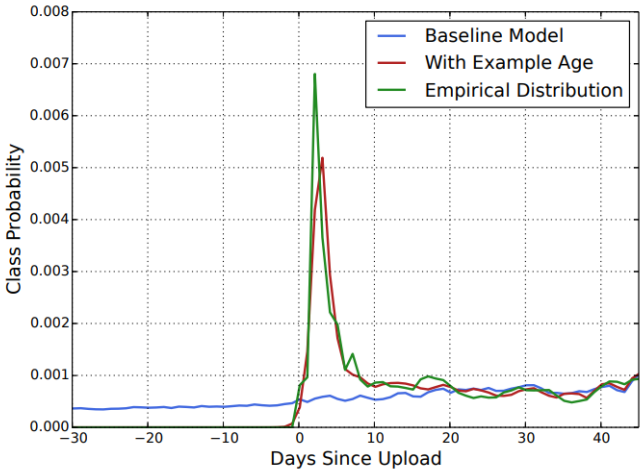


- Embedding input

- Feature(video의 tag, keyword 등)들을 각각 fixed vocabulary의 embedding으로 나타냄
- 각 종류별로 average 때려서 Feedforward (sum, max 등보다 좋은 성능)
- GD Backpropagation 통해 embedding과 model parameter동시 학습
  - Embedding vector은 각 user, video 등을 표현하는 vector
  - Model parameter는 embedding 바탕으로 추천 생성하는 등의 역할로서 학습

Heterogeneous Signals

- Continuous / categorical feature들이 model에 쉽게 추가될 수 있음
  - 검색 기록을 unigram, bigram(한 단어, 두 단어)로 토큰화되어 embedding됨
  - 이 토큰들이 Average되면 각각이 summarized dense search history를 나타냄
  - Demographic(인구통계학), geographic, device 정보 등도 같이 합쳐짐
    - 새 user 추가시 유용함
- Example age
  - 유튜브는 새 영상을 추천해야함
    - ML은 기존 데이터에 기반해 prediction하는 경향이 있어 문제 발생
      - Bootstrapping: 새로운 사용자는 cold-start 겪음
      - Propagating viral content: 인기 급상승 동영상과 덜 알려진 영상 간 balancing
  - Age feature를 같이 추가해서 성능 개선

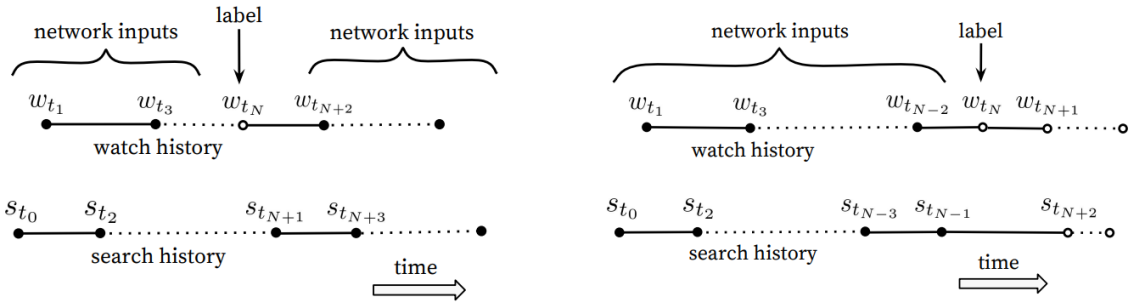


Label and Context Selection

- Surrogate problem
  - 추천이라는 추상적 문제를 어떻게 구체적 문제로 정의하냐?
    - Accurately predict rating으로 정의함
  - 이 방법대로 검증하기 위해 online A/B test가 중요함
    - offline은 검증 어려움: 실제 유저가 추천받은걸 누르냐로 평가해야 하니까
- 추천 외 시청영상 propagate
  - recommendation이 아닌 경로 통해 시청한 영상을 CF로 다른 user들에게 propagate
- 헤비 유저 제한
  - User당 training example 수 제한하여 일부 user가 dominate하지 않도록
- Classifier에 너무 많은 정보 X
  - Classifier에 정보량을 조절해 surrogate problem에 overfitting 방지
    - e.g. 최근 검색 정보를 이용하도록 하면 단순히 검색 내용을 반복 추천할 수 있음
  - 순서 정보 제거하고, 검색 쿼리를 순서가 없는 토큰 집합으로 표현
    - Classifier가 label의 출처를 직접 알 수 없게 됨

- User의 장기적인 선호나 다양한 상호작용 통해 얻은 정보로 더 정교화된 추천 할 수 있도록

- 유저의 비대칭적 시청 패턴
  - 일반적으로 유저들은 시청시 랜덤 시청보다 한 주제의 영상을 쭉 시청함
  - 기존 방법들은 랜덤하게 추천해줬음(a)
  - 이 논문은 과거의 어느 시점으로 dataset을 “rollback”해서 이후 데이터를 지우고 예측(b)



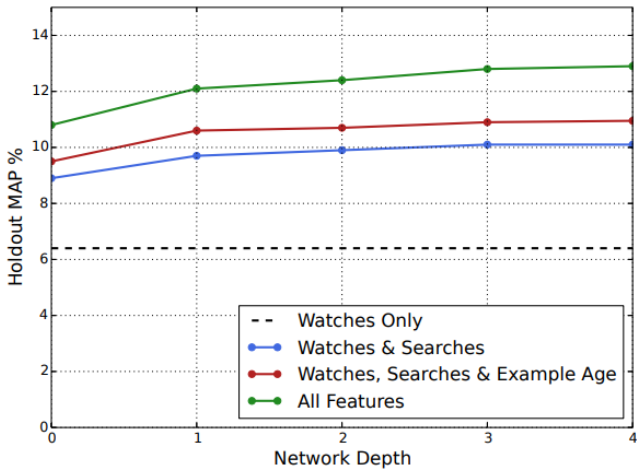
(a) Predicting held-out watch

(b) Predicting future watch

Experiments with Feature and Depth

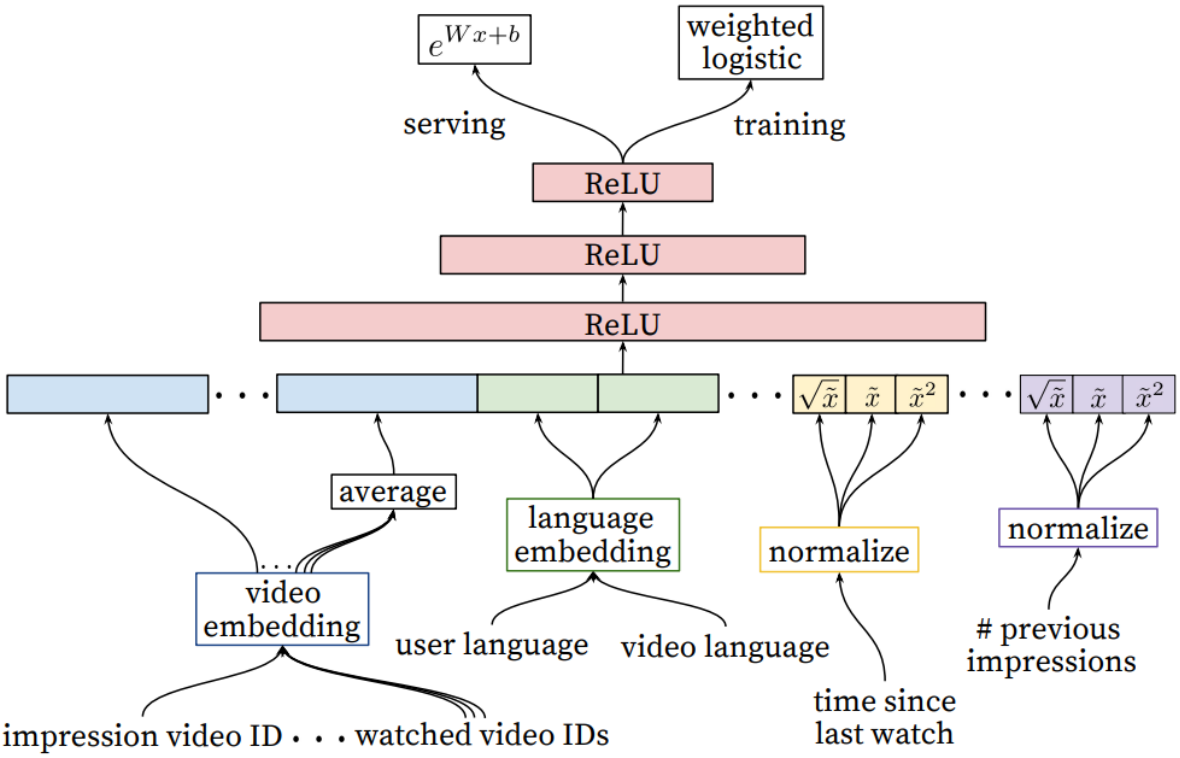
- 충분한 Feature와 Depth를 쌓는 것이 성능 향상을 보임

- Depth 0: A linear layer simply transforms the concatenation layer to match the softmax dimension of 256
- Depth 1: 256 ReLU
- Depth 2: 512 ReLU → 256 ReLU
- Depth 3: 1024 ReLU → 512 ReLU → 256 ReLU
- Depth 4: 2048 ReLU → 1024 ReLU → 512 ReLU → 256 ReLU



- Tower pattern: 각 layer가 input이 가장 크고, 다음 layer가 절반이 되는 타워(피라미드) 형태의 구조 network
    - Output으로 embedding vector이 나옴

Ranking



특정 UI에서 Impression data를 추가적으로 사용해서 score가 같은 candidate끼리의 scoring

- video가 이제 몇백 개밖에 안되니까 더 많은 feature을 사용할 수 있음
- Impression data
  - e.g. user에게 표시된 UI에서 어떤 video를 클릭(안)했는지, 얼마나 봤는지 등의 정보
  - 클릭보다는 watch time이 더 효과적임
  - Impression당 얼마만큼의 watch time을 가지냐

Feature Representation

- Feature 구분
  - Categorical & Continuous 로 구분
    - Categorical feature마다 여러 feature가 있을 수 있고, 1~여러 개 value를 가질 수 있음

        - e.g. login같은 binary ~ search query같은 많은 possible valeu / select or multiselect
    - Properties of item (impression)/ Properties of user & context (query)
- Feature engineering
    - Deep learning 이지만, raw data input에 바로 넣을 수 없어 engineer 해야함
    - Useful feature 뽑아 넣어보자
        - 유저 action sequence를 반영하기 & 이걸 어떻게 점수 매기냐가 main point
    - User-item의 과거 interaction impression정보가 중요함
        - e.g. 이 채널에서 어떤 영상을 봤는가 등 유저의 과거 설명해주는 continuous feature
        - 다른 item간에도 generalization을 잘 해주므로
    - Candidate generation에서 사용한 정보도 ranking에서도 사용해야
    - 과거 video impression의 노출 정도 또한 중요
        - 추천 떴는데 안보면 평가절하
        - 최신 impression 제공이 중요함

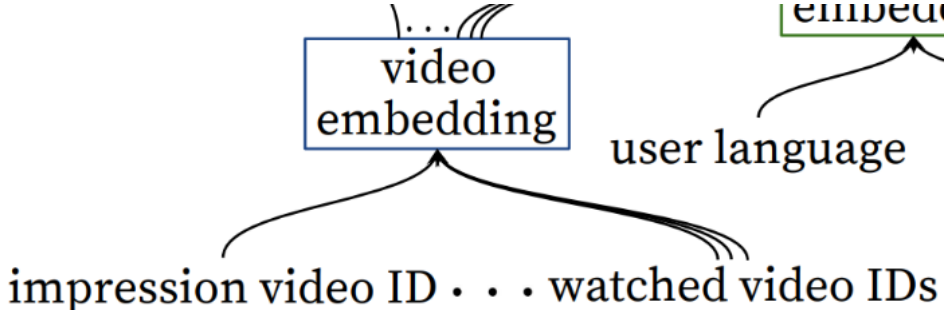
Normalizing Continuous Features

- Neural network는 scale과 input의 distribution에 민감함
    - e.g. 각 feature의 scale이 다를 경우, 작은 쪽이 무시될 수 있음
    - e.g. 가정한 분포(정규분포)와 다를 경우 잘못 학습될 수 있음
- Normalization을 통해 해결
    - Cumulative distribution $\tilde{x} = \int_{-\infty}^{x} df$을 통해 [0,1)로 정규화
    - $\tilde{x}^2$나 $\sqrt{\tilde{x}}$ 를 network에 넣어서 super- or sub-linear 한 함수에 대한 설명력 증가 가능
        - linear한 것보다 더 빠르게 혹은 느리게 증가하는 함수

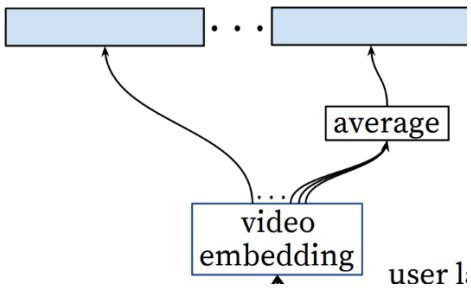
Embedding Categorical Features

Candidate generation 때처럼 sparse categorical feature를 dense representation으로 매핑

- Unique ID space (vocabulary)는 separate learned embedding 가짐



  - 같은 ID space를 사용하는 categorical feature은 embedding을 공유함
    - e.g. 위 그림처럼 video ID를 impression과 watched video ID 등에서 공유하는 경우
    - 공유하더라도 각 feature는 각각 따로 network에 feed함



      - feature당 specialized representation 학습할 수 있게
    - 공유 통해 generalization, speed up, reduce memory
- Cardinality (집합 크기)
  - ID space의 크기가 큰 경우, imbedding click기준 top-N을 추려서 embedding을 넣어줌
  - 이 후, N에 해당 않는 Out-of-voca value의 경우 zero embedding으로 매핑
  - multivalent categorical feature는 average 때려서 feedforward net에 넣음

Modeling Expected Watch Time

- Expected watch time을 weighted logistic regression으로 예측
  - Positive impression은 weight으로 observed watch time
  - Negative impression은 watchtime이 0이므로 unit weight를 줌
- Expected watch time = $\frac{\Sigma\, T_i}{N-k}$
  - positive impression이 작으므로 approximately $E[T](1+P)$가 됨
    - Ti: ith impression의 시청 시간
    - P: 클릭 확률
    - k: positive impression 수
  - P가 작으므로 $E[T]$로 간주 가능

Experiment

Loss: total amount mispredicted watch time

- negative impression이 positive보다 높은 경우 positive의 watch time이 mispredicted watch time

Hidden layer의 넓이가 넓어질수록 성능이 좋아짐

Hidden layers	weighted, per-user loss
None	41.6%
256 ReLU	36.9%
512 ReLU	36.7%
1024 ReLU	35.8%
512 ReLU → 256 ReLU	35.2%
1024 ReLU → 512 ReLU	34.7%
1024 ReLU → 512 ReLU → 256 ReLU	34.6%

Youtube RecSys의 변화

1. The YouTube Video Recommendation System
    a. 시청한 video와 유사도가 높은 video set 매핑하여 점수 상위 N개 추천
2. DNN 4 YouTube
    a. Candidate generation에 초점
3. Recommending what video to watch next, a multitask ranking system
    a. Wide & Deep model을 확장함
        i. Wide 파트에 선형 모델 대신 간단한 DNN인 shallow tower model을 적용
    b. Ranking 모델에 초점
        i. impression data인 클릭, 시청 시간, 좋아요 등 피드백으로 학습
        ii. 기존 impression data + Multimodal feature를 활용
            1. MMoE(Multi-gate Mixture-of-Experts)라는 각각 다른 종류 패턴, 데이터를 학습하는 데 특화된 딥러닝 모델 여러 개 묶어서 사용함

전부 두 단계로 나눠서 추천 알고리즘을 설계했다는 공통점

Questions