

Deep Neural Networks for youtube recommendations

0.Abstract

- deep learning을 통해 엄청난 성능 개선
- deep candidate generation model과 deep ranking model에 대해 설명
- 대규모 추천 시스템 구축 및 유지를 경험하며 생긴 실용적인 교훈과 통찰력을 제공

1.Introduction

- YouTube 추천은 10억 명 이상의 사용자가 계속 증가하는 동영상 콘텐츠에 대해 맞춤 콘텐츠를 추천하기 위한 미션이 있다
- YouTube 추천은 다음 세 가지 측면에서 극도로 챌린징하다
 - Scale
 - 작은 문제에 효과적으로 작동하는 많은 기존 추천 알고리즘들이 YouTube 규모에서 동작하지 않는다
 - YouTube의 막대한 사용자 규모와 동영상을 처리하기 위해서는 전문화된 분산 학습 알고리즘과 효율적인 서빙 시스템이 반드시 필요하다
 - Freshness
 - Youtube에는 매순간 많은 시간 분량의 동영상이 업로드 되며, 새로운 영상 콘텐츠에 따른 사용자의 반응도 실시간으로 발생한다
 - 추천 시스템에서는 이러한 실시간 반응 요소들을 적절하게 적용 및 반영할 수 있는 방법을 찾아야한다
 - 새로운 콘텐츠와 잘 알려진 동영상들을 균형 있게 다루는 것은 탐색(Exploration)과 활용(Exploitation) 관점에서 이해할 수 있다

- Noise

- 희소성과 다양한 관측 불가능한 외부 요인들 때문에 예측하기가 기본적으로 어렵다
 - 희소성 : 영상에 대해 직접적으로 평가하는 경우가 드물기 때문에
- 노이즈가 섞인 암시적 피드백 신호를 모델링
 - 동영상을 시청했다는 implicit feedback을 주로 사용
- 콘텐츠와 관련된 메타데이터는 잘 정의된 체계가 없어 구조화가 잘 되지 않는다

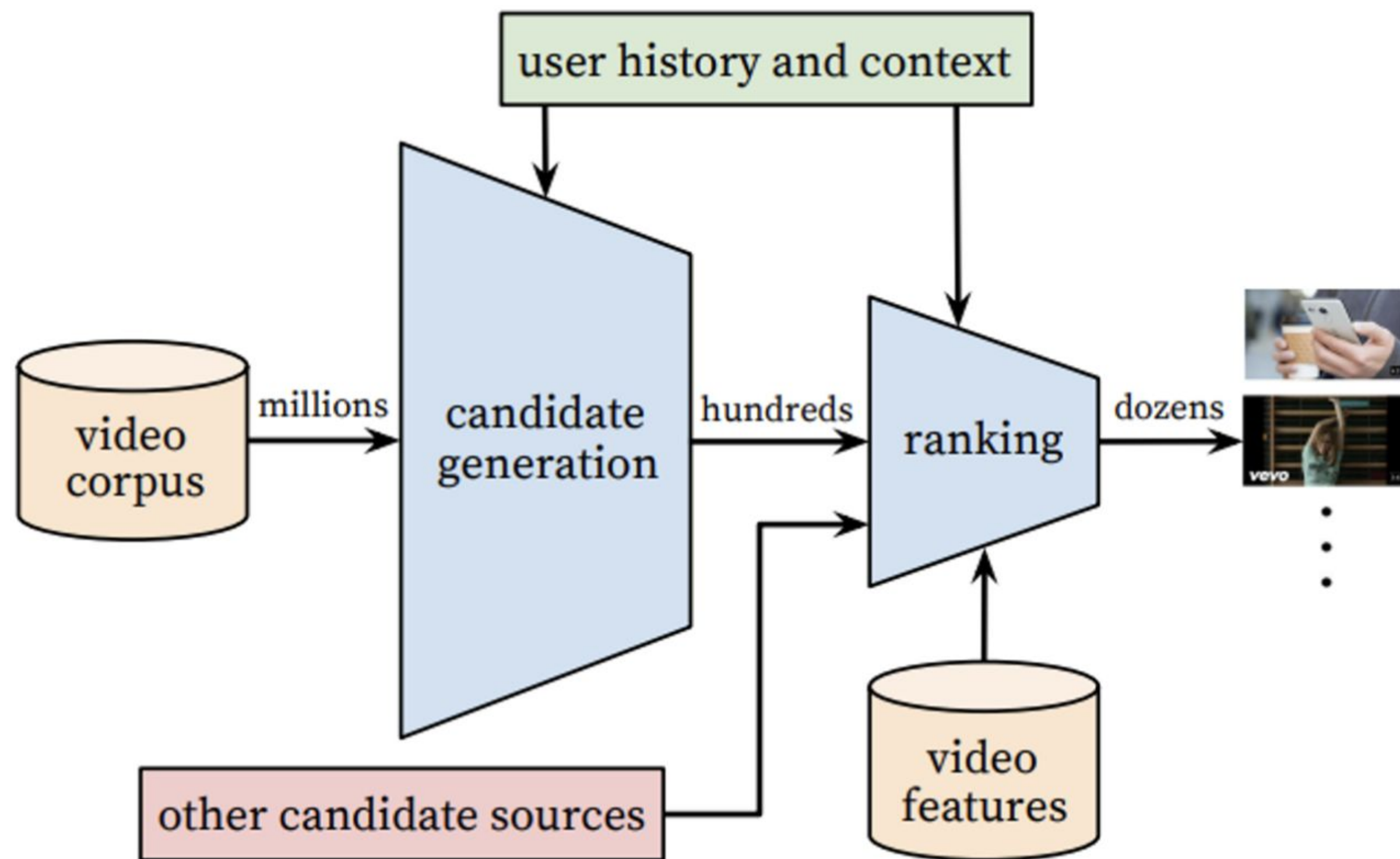
- Matrix Factorization에 비해 딥러닝을 이용한 추천에 대한 연구가 상대적으로 적었다

- 논문의 구성

- Section 2: 시스템에 대한 간단한 개요
- Section 3: candidate generation model에 대한 구체적인 설명
- Section 4: ranking model에 대한 구체적인 설명

+ :: ◦ Section 5: 결론 및 시사점

2. System Overview



- 크게 candidate generation과 ranking으로 나뉜다
- candidate generation
 - user history and context를 input으로 한다
 - 유저가 관심있을 법한 수백 개의 동영상 콘텐츠를 output으로 한다
 - 협업 필터링을 사용하며, 사용자들 간의 유사성은 동영상 시청 기록의 ID, 검색어 토큰 및 인구 통계 정보와 같은 고수준 특징들을 기반으로한다
- ranking
 - 비디오와 사용자를 설명하는 다양한 feature들을 사용하여 각 비디오에 점수를 할당
 - 가장 높은 점수를 받은 비디오들이 사용자에게 제시된다
- 개발 과정에서는 offline metrics(precision, recall, ranking loss 등)을 활용하여 시스템을 반복적으로 개선
- 알고리즘 또는 모델의 효과를 최종적으로 결정하기 위해서는 라이브 실험을 통한 A/B 테스트에 의존한다
- 라이브 실험에서는 CTR, 시청 시간 및 사용자 참여를 측정하는 여러 다른 메트릭들의 변화를 측정할 수 있다
- 이는 라이브 A/B 결과가 오프라인 실험과 항상 상관관계가 있는 것은 아니기 때문에 중요

3. Candidate Generation

- 엄청난 양의 콘텐츠 중에서 유저와 관련성이 있는 수백 개의 콘텐츠로 범위를 좁힘
- 이전에는 rank loss를 기반으로 matrix factorization이 사용
- Shallow Network를 통해 Factorization 진행
- Factorization 기술을 non-linear으로 일반화했다고 볼 수 있음

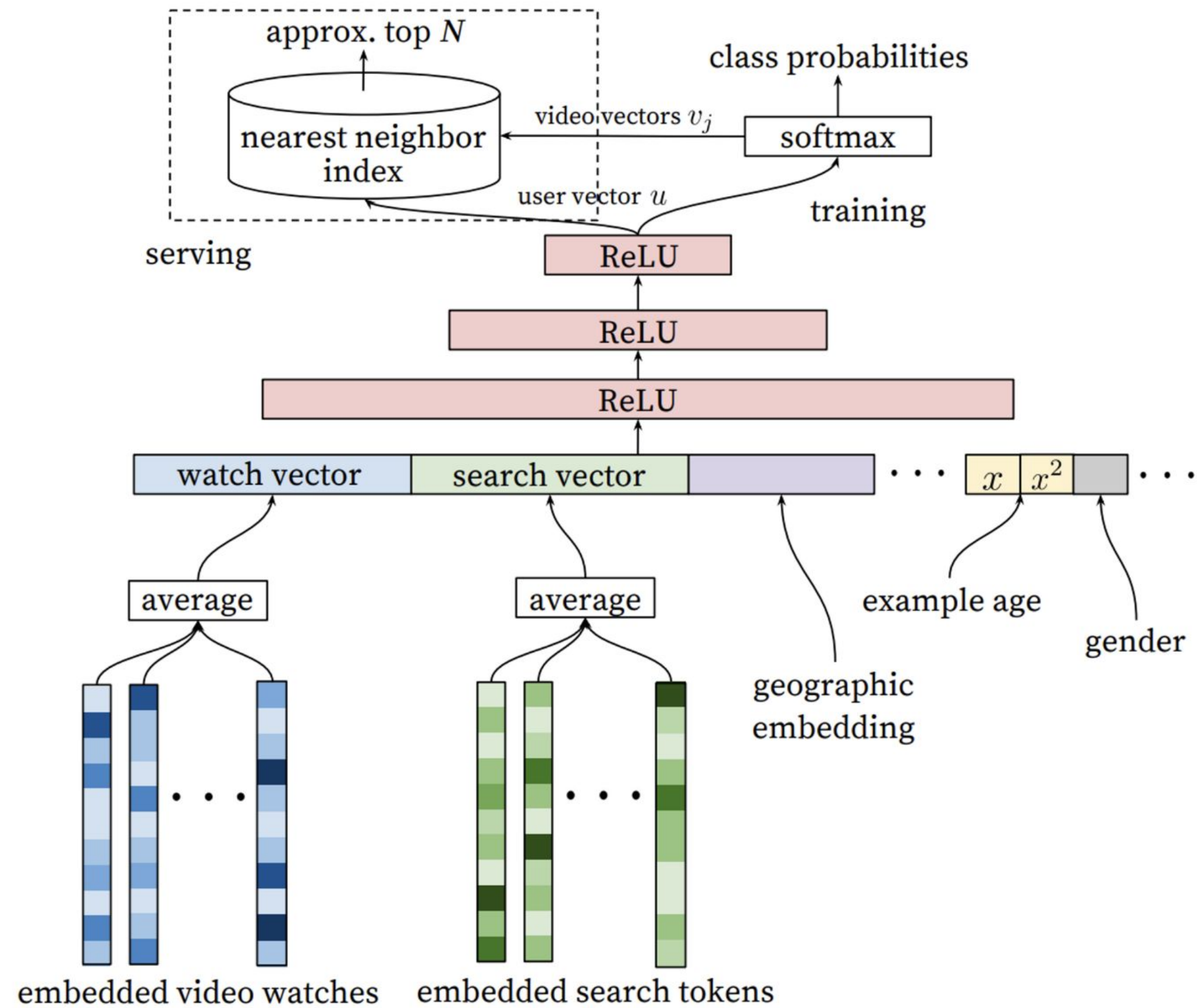
3-1. Recommendation as Classification

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

- 사용자(U)와 Context(C)를 기반으로 특정 시간(t)에서 수백만개의 아이템(V) 중 각 아이템(i)의 시청 w_t 를 예측
 - u 는 user-context pair의 고차원 임베딩
 - v_j 는 각각의 후보 동영상의 임베딩
 - 임베딩 벡터들은 단순히 희소 entities 를 매핑하여 얻어진 dense 한 벡터
 - deep neural networks는 사용자의 히스토리와 맥락을 기반으로 사용자 임베딩 u 를 학습
 - 사용자 임베딩을 활용하여 소프트맥스 분류기를 통해 다양한 동영상들을 분류하는데 사용됩니다.

- explicit feedback(thumbs up/down)이 존재하지만, 사용자가 시청 완료한 영상 콘텐츠를 positive로 분류하여 implicit feedback을 학습함
 - implicit feedback이 explicit feedback보다 훨씬 더 많이 존재
 - implicit feedback을 활용하면 explicit feedback이 부족한 tail부분의 정보를 보완하여 더 다양하고 정확한 추천을 생성할 수 있게 된다
- Efficient Extreme Multiclass
 - Softmax classification 에서 클래스의 갯수가 늘어날때 계산량이 기하급수적으로 증가
 - negative sampling을 통해 기존 softmax 보다 약 100배 가량 속도를 향상했다

3-2. Model Architecture



- sparse한 ID들로 이루어진 가변 길이의 특징들을 처리하기 위해 임베딩(embedding)을 사용
 - 사용자의 시청 기록은 비디오 ID들로 구성된 가변 길이의 시퀀스로 표현되며, 이러한 비디오 ID들은 임베딩을 통해 밀집 벡터 표현으로 매핑
- 임베딩들을 평균화하여 고정된 길이의 벡터로 변환한 뒤 concatenate. 이렇게 함으로써, 모델은 가변 크기의 입력을 고정된 크기의 벡터로 변환하여 히든 레이어에 입력으로 사용할 수 있다
 - 여러 전략중에서 임베딩의 평균을 사용하는 것이 가장 좋은 결과를 냈다
- Hidden layers: Deep candidate generation model은 fully connected된 여러 개의 hidden 레이어를 가진다.
- 훈련 단계에서는 cross-entropy loss가 사용되며, 샘플링된 softmax(output of the sampled softmax)에 대한 gradient descent를 이용하여 loss를 최소화한다
- Serving: 실제 서비스에서의 운영 시, approximate nearest neighbor lookup을 수행하여 수백 개의 후보 동영상 추천을 생성한다

3-3.Heterogeneous Signal

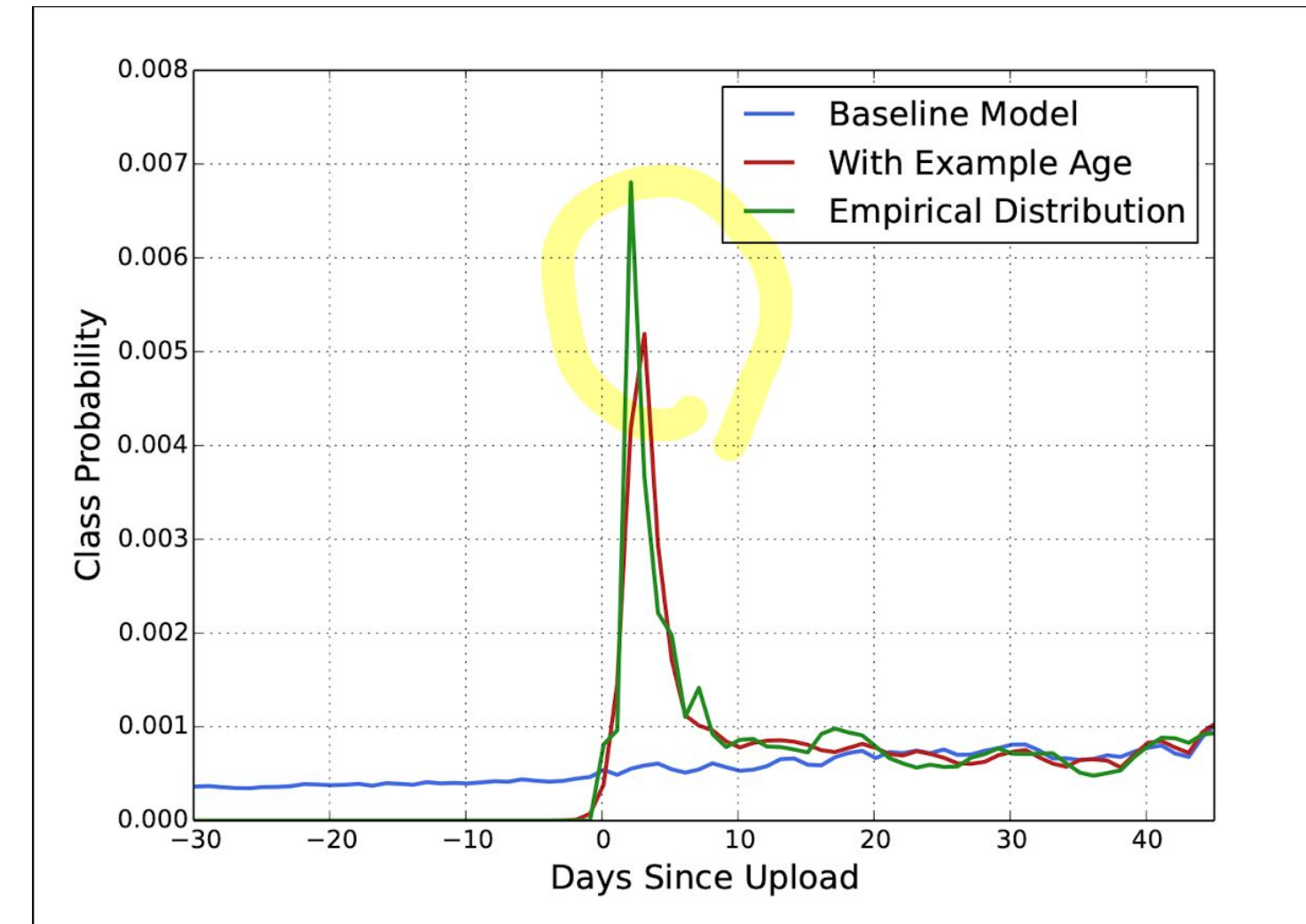
Heterogeneous Signals (추가 feature)

- generalization of Matrix Factorization처럼 neural network를 사용한 이점은 다양한 continous feature와 categorical feature들을 쉽게 사용할 수 있다는 점이다.
- 추천시스템이 해결하고자 하는 **cold start** 문제는 **demographic** 정보를 통해 해결을 한다고 한다.
 1. 검색 기록은 시청 기록과 유사하고, unigrams나 bigrams를 활용하여 embedding을 적용할 수 있다.
 2. demographic(인구관련정보) 정보들은 새로운 유저들에 대한 예측을 위해 중요하게 여겨진다.
 - 유저의 지질학적 위치와 device 정보가 embedding을 거친 후 concat된다.
 - 성별, 로그인 장소, 나이 등등 간단한 수치 정보도 정규화 후에 모델에 사용된다.

3-3.Heterogeneous Signal

3. Example Age 특징

- 유튜브는 계속 새로운 아이템들이 업로드된다.
 - 사용자들은 그 아이템을 보통 선호하는 것으로 관측되기 때문에 추천에 있어 새로운 아이템은 굉장히 중요하다
- 하지만 머신러닝 시스템에서는 과거 데이터를 바탕으로 미래의 행동을 예측한다.
 - 종종 과거에 있는 아이템에 대한 결과를 보여주는 경향이 종종 있다고 한다.
- 시청 기록에 있어 아이템 벡터의 평균을 적용하기 때문에 시간적 요소에 대한 sequence가 반영되지 않는다.
 - 이를 보정하기 위해서 아이템의 나이, Age에 해당하는 feature를 추가하게 되었다.
- 'Example Age'를 적용하게 된 모델의 성능을 보면 baseline 모델보다 엄청난 성능 향상을 보이는 것을 알 수 있다.



3-4. Label and Context Selection

- 추천 시스템은 가끔 surrogate problem을 해결하며 그 결과를 특정 context로 바꾸는 과정을 포함하고 있다.

- surrogate problem



개발한 추천 엔진에 대한 평가는 직접 serving하여 사용자로부터 피드백을 받는 것이지만, 이러한 형태의 평가가 불가능하기 때문에 RMSE, MAP 같은 성능 지표를 사용하여 모델을 평가한다.

- 영화 추천에서 볼 수 있는데 영화의 평점을 예측하는 문제를 풀어 높은 평점을 가진 영화를 추천하는 것이 성공적인 추천으로 이어지는 경우가 있다.
- 본 논문에서는 분류 예측 모델이기 때문에 예측을 잘한다는 것은 추천을 잘하는 모델이라고 해석한다.
- 모델에 사용되는 비디오들은 추천된 context들뿐만 아니라 유튜브에서 나온 모든 데이터들 중에서 만들어진다.
 - 데이터의 모집단을 전체 유튜브로 설정하지 않으면, bias가 생길 수 있어 새로운 아이템 추천이 어렵기 때문이다.
 - 추천 시스템이 아닌 다른 수단을 통해 비디오들을 찾았다면, 이 데이터를 활용해 collaborative filtering으로 학습에 사용한다.
- 모델의 성능지표에 큰 역할을 준 또 다른 방법은, 모든 사용자의 가중치를 동일하게 유지하기 위해서 사용자마다 학습 데이터의 수를 고정한 점이다.
 - 너무 활동적인 유저들에게 영향받을 수 있는 문제를 방지하고자

3-4. Label and Context Selection

- 학습 데이터를 선택하는 방법으로 어떤 사용자의 무작위로 선정된 아이템을 예측하는 것보다 다음 아이템을 예측하도록 데이터를 구성하는 것이 효과적이라고 한다.



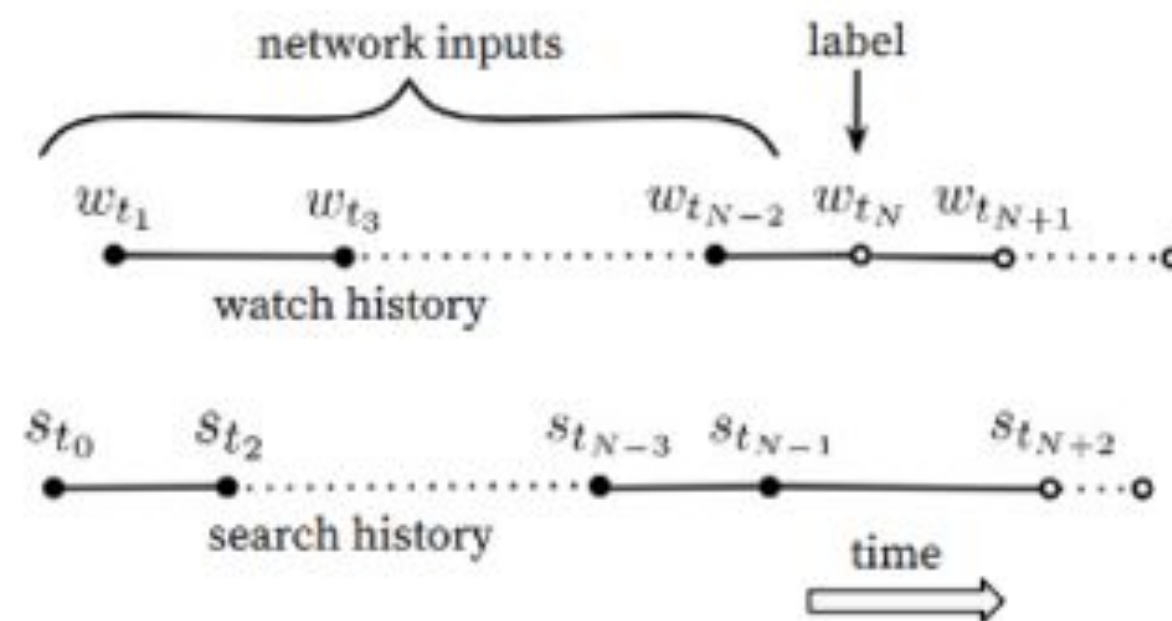
예를 들어 'Taylor Swift' 를 검색했을 때를 살펴보겠다.

추천의 목적은 **다음에 볼 영상을 추천** 하는 것이다. 그렇다면 다음에 볼 영상으로 Taylor Swift 관련 영상을 추천하는 것이 무조건적으로 옳을지에 대한 답변은 '아니다'이다.

마지막 검색 기록을 기반으로 추천 영상을 찾는 방법보다 **이전 검색 기록** 과 시청 기록을 기반으로 다음 영상을 예측할 수 있도록 학습하는 것이 적절한 방법이다.

이런 방법이 더 적절한 이유는 사용자의 소비 패턴은 일관적이지 못하기 때문이다.

특정 구간들을 input, 그리고 그외 구간들 중 특정 구간을 예측하는 random 방식보다는, 특정 구간을 예측할 수 있도록 이전 사용자 소비 기록을 학습하는 것이 더 적합하다고 한다.



(b) Predicting future watch

3-5. Experiments with Features and Depth

- feature들과 depth를 추가할 수록, precision과 데이터 검증의 성능이 높아지는 것을 볼 수 있다.
- 실험에서 1만개의 비디오와 1만개의 검색 토큰들을 256개의 float형태로 embedding 시킨 후, 각 최근 50개 검색 토큰과 시청 비디오를 갖고 있다.

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

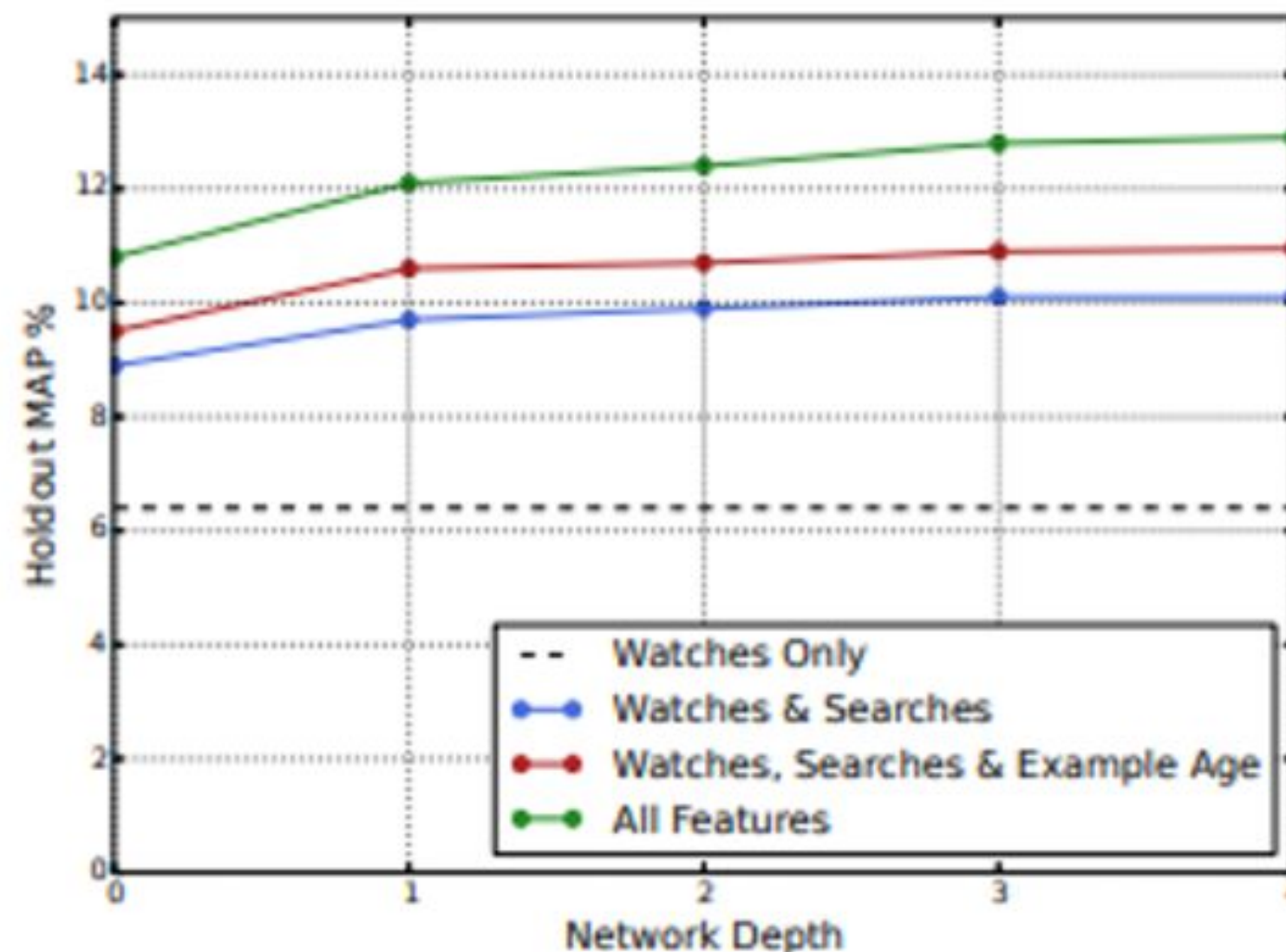
Depth 3: 1024 ReLU → 512 ReLU → 256 ReLU

Depth 4: 2048 ReLU → 1024 ReLU → 512 ReLU → 256 ReLU

- 결과적으로 위 Depth와 같이 depth를 깊게 할 수록 성능이 높아지는 것을 알 수 있다.

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%

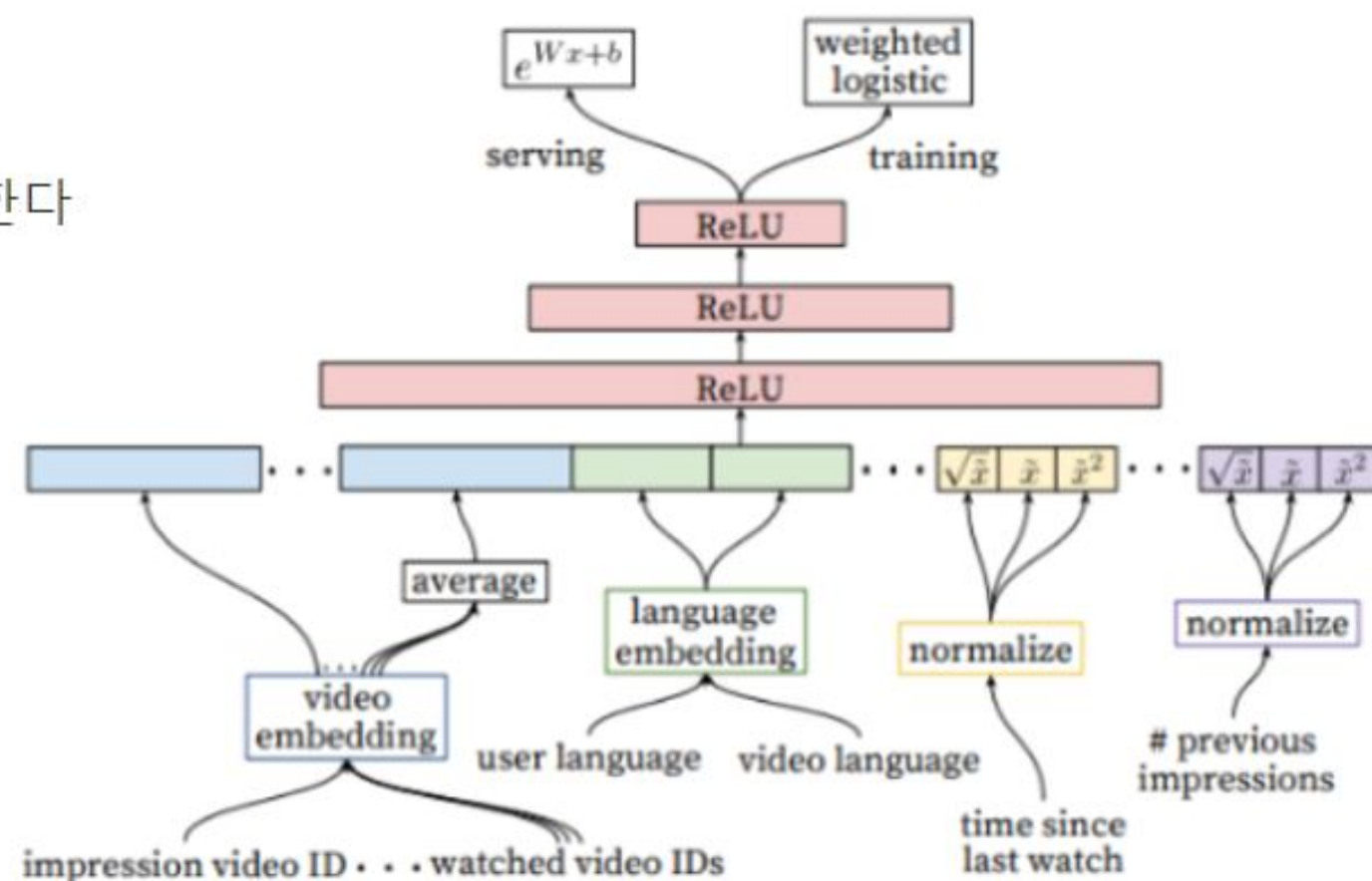
Table 1: Effects of wider and deeper hidden ReLU layers on watch time-weighted pairwise loss computed on next-day holdout data.



4. Ranking 모델

Ranking 모델

- 앞선 과정에서 생성한 후보군(높은 확률로 클릭할) 영상들을 모두 볼 수도 있지만, 특정 썸네일에 따라 안 누를 수도 있기때문에 랭킹모델을 통해 수십개로 줄여준다.
 - 따라서 Input으로 들어가는 영상의 갯수가 줄어들었기 때문에 추가적으로 유저 & 영상간 관계 파악하기 위해 피쳐들을 추가해준다.
- 그림에서 볼 수 있듯 네트워크의 레이어는 Candidate Generation Model 과 비슷하다. 하지만 사용하는 피쳐가 수백개 정도라 한다. 해당 논문에서는 단순 CTR 을 통해 랭킹을 계산하지 않는다고 한다.
 - 그 이유는 썸네일과 같은 이유로 Click bait 가 일어날 수 있는 영상이 많을텐데 이는 proper 한 metrics가 아니게 된다.
 - 그래서 저자들은 impression 에 따른 시청 시간 을 예측하는 식으로 랭킹을 계산한다고 한다



4-1. Feature Representation

Feature Representation

- feature들은 데이터의 형태에 따라 categorical과 continuous로 나뉜다.
 - categorical feature들은 binary 데이터, 마지막 검색 기록, 점수 부여 ID, 마지막 시청 N개 아이템

Feature Engineering

- ranking 모델에서는 여러 개의 feature들을 사용한다.
- DL이 feature engineering의 어려움을 줄여주지만, feature size와 어떤 feature들을 사용할지 고민해야 한다.
 - 사용자의 행동에 대한 sequence를 반영하는 것, 이런 행동을 점수로 어떻게 연관지을지 등등을 고민해야 한다.
 - 해당 채널에서 시청한 영상 개수, 해당 이슈와 관련된 영상을 시청한 마지막 시간 등과 같은 연속적인 feature는 아이템과 연관된 사용자의 과거 행동이기에 매우 효과적이다.

Embedding Categorical Features

- 아이템(영상)의 ID와 검색 기록을 임베딩하여 모델의 input으로 사용한다.
 - impression 클릭 빈도수에 따라 오로지 상위 N개를 사용하여 임베딩을 진행하고, N개에 포함되지 않은 Out-of-vocabulary 들은 zero-embedding 으로 매핑이 된다고 한다.
 - 동일한 영상에 관해서는 그 영상에 대한 global embedding 을 활용한다고 한다.
 - 위 그림에서 볼 수 있는 video embedding 같은 경우, impression 영상ID, 마지막으로 유저가 본 영상ID, 해당 추천을 해준 seed 영상ID 를 활용하여 만들어진 것이다.

4-1.Feature Representation

Normalizing Continuous Features

- continuous feature의 경우에는 0~1로 scaling해주며 super/sub-linear한 특징을 배우기 위해 x^2 , $x^{(0.5)}$ 의 데이터 또한 input으로 넣어준다.

✓ super-linear

입력과 출력 간의 수학적 관계에서 출력이 입력보다 빠르게 증가하는 함수입니다. 다시 말해, 입력이 증가함에 따라 출력은 더 빠른 속도로 증가합니다. 수학적으로 함수 $f(x)$ 가 초선형 함수라면, 그 증가율이 선형보다 큰 것을 의미합니다. 이는 함수의 변화율(기울기)이 증가하는 것을 의미합니다.

초선형 함수 예시: $f(x) = x^2$

sub-linear

입력과 출력 간의 수학적 관계에서 출력이 입력보다 느리게 증가하는 함수입니다. 다시 말해, 입력이 증가함에 따라 출력은 더 느린 속도로 증가합니다. 수학적으로 함수 $g(x)$ 가 부선형 함수라면, 그 증가율이 선형보다 작은 것을 의미합니다. 이는 함수의 변화율(기울기)이 감소하는 것을 의미합니다.

부선형 함수 예시: $g(x) = \sqrt{x}$

super-/sub-linear 를 허용함으로써, 신경망은 비선형적인 패턴을 더 잘 포착하고 주어진 입력 데이터를 기반으로 더 정교한 예측을 할 수 있게 됩니다. 이러한 증가된 표현 능력은 특히 복잡하고 복잡한 데이터셋을 다룰 때 유용합니다

4-2. Modeling Expected Watch Time

Modeling Expected Watch time

- 본 저자의 목적은 좋아요 혹은 impression이 담겨있는 비디오든 아니든 선택된 training example 들을 활용해서 예상된 시청 시간을 예측하는 것이 목표다.
- positive인 비디오들에 대해서는 annotation (기록/주석)을 추가한다.

✓ 이 section에서 말하는 positive의 기준은 노출된 비디오에 대해 클릭을 했는지에 대한 여부다.

- 해당 watch time을 예측하기 위해서, weighted logistic regression을 사용한다.
 - cross entropy loss를 사용해서 regression을 학습한다.
 - positive video 같은 경우 비디오의 시청시간을 기반으로 가중치가 조정된다.
 - Negative 비디오들은 unit weight로 지정된다.
 - logistic regression이 학습하는 odds는 $\text{sum}(\text{Time}) / (N - k)$ 이다.
 - N: 전체 training example
 - k: positive 개수
 - $\text{sum}(\text{Time})$: positive 한 비디오들의 시간 합

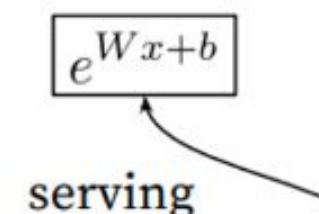
- 실제로 odds를 정리하면 아래와 같다.

$$E[T](1 + P)$$

- $E[T]$: positive video 시청 평균 시간
- P : click 한 확률

✓ 그런데 P의 값이 거의 작기 때문에 $E[T]$ 에 근사하다고 한다.

- Inference에는 지수함수를 activation function으로 사용한다.



e^{Wx+b}

serving

- odd 값이 제일 시청 시간 (T)에 근사해질 수 있게 하기 위해서 사용한다.

5. Conclusion

1. 피쳐 엔지니어링을 통해 중요한 피쳐를 활용하여 기존 linear 모델에서는 찾지 못한 non-linear 한 특성을 DL 을 통해 찾아낸 것이라고 볼 수 있다.
2. "Example Age"가 성능을 크게 개선 시켰다.
 - Freshness가 중요한 유튜브 생태계에 크게 기여했다고 한다.
 - 특히 온라인 A/B 테스트에서 시청시간 예측을 향상시켰다고 한다.
3. 모든 것을 딥러닝으로 하기는 쉽지 않다.
 - 파이프라인 구축하는 방법, 임베딩 방법 등이 중요하다는 것을 알 수 있다.
4. positive data는 감상 시간에 가중치를 주고, negative data는 unity(통일성)에 가중치를 준 것도 개선점이 컸다.
 - 이를 통해 클릭률 prediction보다 시청시간예측에서 더 중요하다는 것을 유추할 수 있다.