# Deep Neural Networks for YouTube Recommendations

유튜브 추천 알고리즘의 원리에 대해 설명
ㄴ Deep Learning을 통한 추천 성능 향상

정보 검색(Information Retrieval)에 대해 두 가지 관점으로 접근

**1) 심층 후보 생성 모델**

**2) 개별 심층 랭킹 모델**

## 1. Introduction

- 해당 논문에서는 최근 YouTube 동영상 추천 시스템에 딥러닝이 미치는 영향에 중점을 두고 있음
- 해당 연구는 Google Brain 연구에 기반

### YouTube 동영상을 추천 시의 어려움

1. **규모**
    - 기존 추천 알고리즘을 대규모에서 작동시키기 어려움
    - 대규모 사용자 및 동영상 데이터 코퍼스를 처리하기 위해서는 고도로 특화된 분산 학습 알고리즘과 효율적인 서빙 시스템이 필수적임
2. **신선도**
    - YouTube는 초당 많은 시간의 동영상이 업로드되는 매우 동적인 코퍼스를 가지고 있음

      ⇒ 추천 시스템은 새로 업로드된 콘텐츠와 사용자의 최신 작업을 빠르게 모델링할 수 있어야 함

- 새로운 콘텐츠와 잘 알려진 동영상을 탐험/활용 관점에서 균형을 맞추는 것이 중요

3. **잡음**
    - YouTube의 과거 사용자 행동은 희소성과 여러 가지 관측할 수 없는 외부 요인으로 예측하기가 어려움

      ⇒ 사용자 만족도의 정확한 기준을 거의 얻지 못하고 대신에 노이지 함시적 피드백 신호를 모델링
    - 또한, 콘텐츠와 관련된 메타데이터는 명확한 온톨로지 없이는 이해하기 어렵게 구조화되어 있음

⇒ 위의 어려움들에 robust한 알고리즘을 구축하고자 함

# 2. System Overview

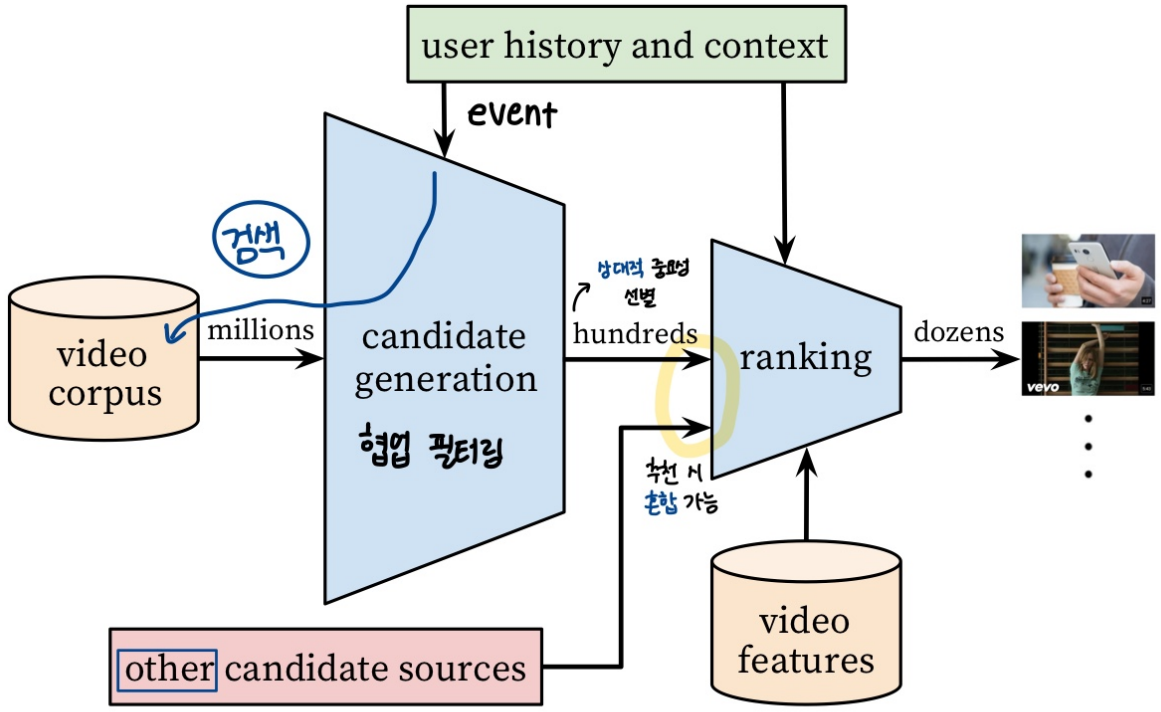



**Figure 2: Recommendation system architecture demonstrating the "funnel" where candidate videos are retrieved and ranked before presenting only a few to the user.**

- 해당 시스템은 후보 생성과 순위를 위한 **두 개**의 신경망으로 이루어져 있음

    1. 후보 생성 신경망

- 사용자의 활동 이력에서 이벤트를 받아 대규모 데이터 코퍼스에서 소량의 동영상을 검색하여 사용자에게 관련성 높은 후보를 제공

2. 순위 신경망
    - 동영상과 사용자의 풍부한 특징을 사용하여 각 동영상에 점수를 할당하고 사용자에게 가장 높은 순위의 동영상을 제시

⇒ 매우 큰 동영상 코퍼스에서 개인화된 추천을 제공하면서 사용자에게 매력적인 동영상이 기기에 표시될 확신을 얻을 수 있음

⇒ 이전에 기술된 다른 소스에서 생성된 후보를 혼합하는 것도 가능

# 3. Candidate Generation

- 후보 생성 단계에서, 거대한 YouTube 코퍼스는 수백 개의 동영상으로 축소되어 사용자와 관련이 있을 것으로 생각되는 동영상들이 선택
- 순위 손실에 기반한 행렬 분해로 문제에 접근

  ⇒ 이를 비선형적 행렬 분해 기술로 일반화

## 3-1. Recommendation as Classification

- 추천을 극도의 다중 클래스 분류 작업으로 간주

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

  - **task)** 사용자 U와 컨텍스트 C를 기반으로 하는 시점 t에서 수백만 개의 비디오 i(클래스) 중 특정 비디오 시청 $w_t$를 정확하게 분류하는 것
  - **임베딩)** 희소 엔터티(개별 비디오, 사용자 등)를 $R^n$의 밀집 벡터로 매핑하는 작업
  - DNN과 softmax(→ 각 클래스에 속할 가능성을 확률로써 제시) classifier를 사용하여 비디오 클래스를 구분하는 데 유용한 사용자 임베딩 u를 학습
- 명시적 피드백 메커니즘과 암묵적 메커니즘을 동시에 활용
  - 명시적) 좋아요/싫어요, 제품 내 설문 조사 등
  - 암묵적) 사용자가 비디오를 완전히 시청하는 것 → 긍정적 예제

- 수백만 개의 클래스로 이루어진 모델을 효율적으로 훈련시키기 위해 후보 샘플링 기술을 사용하였고, 이를 중요성 가중치로 보정하였음
  - 소프트맥스에 비해 100배 이상의 속도 향상
- 서빙 시간에는 수백만 개의 항목에 대한 점수를 계산해야 함
  - 소프트맥스의 대안으로 해싱과 유사한 방법을 사용
  - 점수화 문제를 점곱 공간에서의 최근접 이웃 검색으로 단순화
  - A/B 테스트 결과 ⇒ 최근접 이웃 검색 알고리즘의 선택에 큰 영향을 받지 않았음

## 3-2. Model Architecture

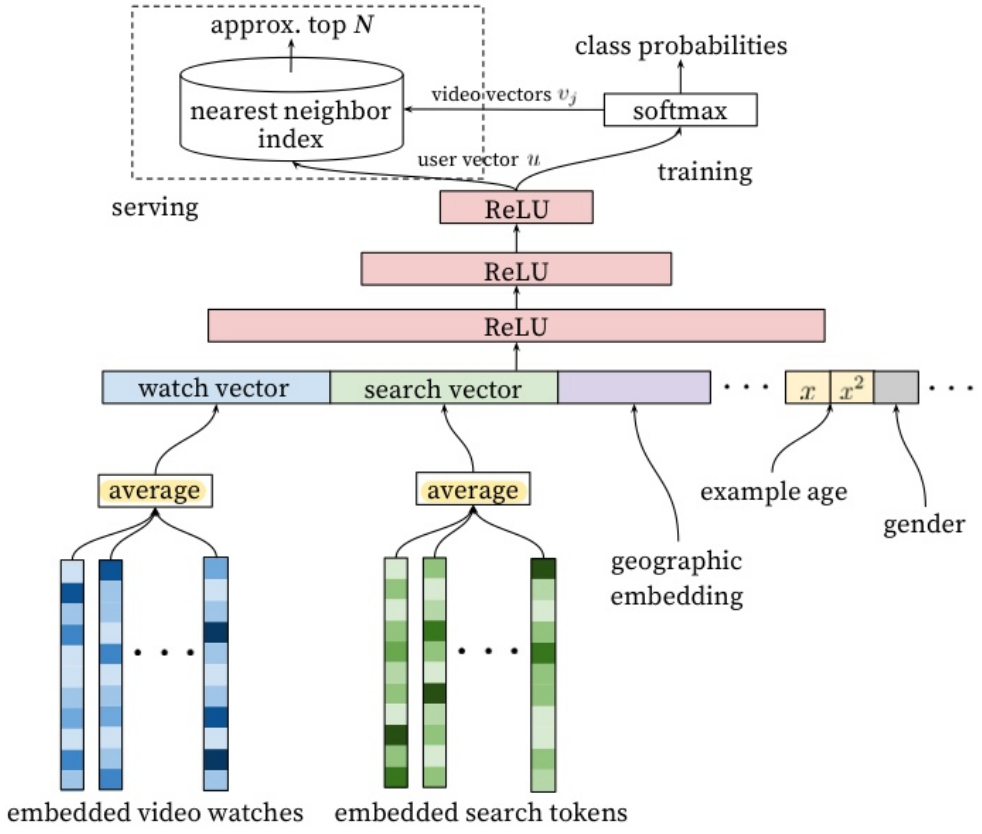



**Figure 3: Deep candidate generation model architecture showing embedded sparse features concatenated with dense features. Embeddings are averaged before concatenation to transform variable sized bags of sparse IDs into fixed-width vectors suitable for input to the hidden layers. All hidden layers are fully connected. In training, a cross-entropy loss is minimized with gradient descent on the output of the sampled softmax. At serving, an approximate nearest neighbor lookup is performed to generate hundreds of candidate video recommendations.**

- 고정된 어휘 크기를 갖는 각 비디오에 대한 고차원 임베딩을 학습하고 이러한 임베딩을 피드포워드 신경망에 입력

- 사용자의 시청 이력은 희소한 비디오 ID의 가변 길이 시퀀스로 나타내어지며, 이를 임베딩을 통해 밀집 벡터 표현으로 매핑

- 신경망은 고정 크기의 밀집 입력을 필요로 하며, 임베딩을 단순히 평균 내는 것이 여러 전략 중 가장 성능이 우수했음
- 이러한 임베딩은 일반적인 경사 하강 역전파 업데이트를 통해 모든 다른 모델 매개변수와 함께 공동으로 학습됨
  - 특징들은 첫 번째 레이어에 연결됨
  - 이때 몇 개의 fully-connected ReLU 레이어가 이어짐

## 3-3. Heterogeneous Signals

- DNN을 행렬 분해의 일반화로 사용하는 주요 이점 중 하나는 임의의 연속 및 범주형 특징을 모델에 쉽게 추가할 수 있다는 점임
  - 검색 이력은 시청 이력과 유사하게 처리되며, 각 쿼리는 유니그램 및 바이그램으로 토큰화되고 각 토큰은 임베딩됨
  - 평균을 내면 사용자의 토큰화된 임베딩된 쿼리는 요약된 밀집 검색 이력을 나타냄
- "예제 연령" feature

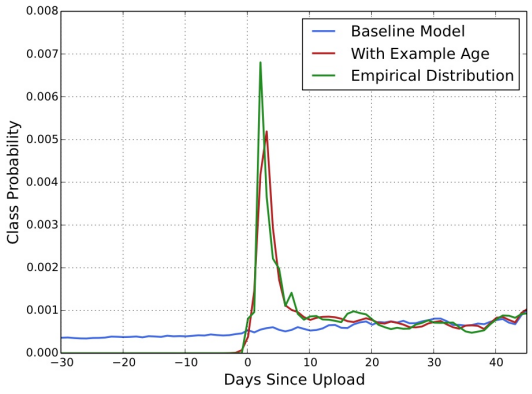



**Figure 4: For a given video [26], the model trained with example age as a feature is able to accurately represent the upload time and time-dependant popularity observed in the data. Without the feature, the model would predict approximately the average likelihood over the training window.**

  - 사용자는 신선한 콘텐츠를 선호하지만, 그 중요성을 유지하면서 관련성을 유지하려 함
  - 머신 러닝 시스템은 역사적 예시에서 미래 행동을 예측하기 위해 훈련되므로 과거에 대한 암묵적인 편향을 보일 수 있음

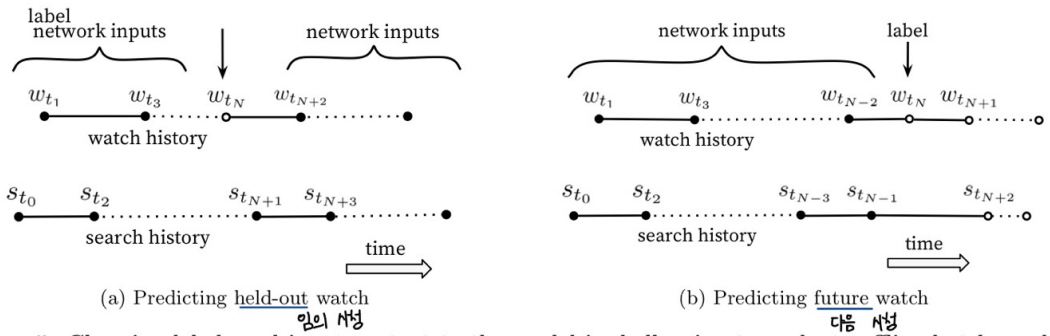
- 비디오 인기의 분포는 매우 불안정하지만, 추천 시스템이 생성하는 코퍼스에 대한 다항 분포는 몇 주 동안의 훈련 창에서의 평균 시청 가능성을 반영할 것임

  ⇒ 이를 보정하기 위해 훈련 중에 훈련 예제의 연령을 기능으로 입력

## 3-4. Label and Context Selection

- 추천 시스템은 종종 대리 문제를 해결하고 결과를 특정 맥락으로 이전하는 것이 중요
  - 예를 들면, 영화 평점을 정확하게 예측하는 것이 효과적인 영화 추천으로 이어진다는 가정이 있음
  - 추천 학습 문제의 선택이 A/B 테스트의 성능에 큰 영향을 미치지만 오프라인 실험에서 측정하기는 어렵다는 점을 강조
- 모든 YouTube 시청(다른 사이트에 임베드된 시청 포함)에서 생성된 교육 예제를 사용하여 새로운 콘텐츠가 나타나는 것이 어려워지지 않도록 함
  - 이렇지 않으면 추천이 과도하게 이용되어 새로운 콘텐츠가 나타나기 어려워짐
- 또한 사용자가 추천 이외의 수단으로 비디오를 발견하는 경우, 협업 필터링을 통해이 발견을 빠르게 전파하고자 함
  - 각 사용자 당 고정된 수의 교육 예제를 생성하여 손실 함수에서 사용자를 동등하게 가중치를 부여
  - 이로써 손실이 높은 소수의 활발한 사용자가 지배하는 것을 방지
- 모델이 사이트 구조의 구조를 이용하고 대리 문제에 과적합하는 것을 방지하기 위해 분류기에 정보를 숨기는 것이 중요
  - 예를 들어, 사용자가 "테일러 스위프트"에 대한 검색 쿼리를 수행한 경우, 해당 정보를 사용하면 모델이 "테일러 스위프트"에 대한 검색 결과 페이지에 표시된 동영상을 가장 시청될 확률이 높은 것으로 예측
  - 이를 방지하기 위해 순서 정보를 버리고 검색 쿼리를 토큰의 순서 없는 집합으로 나타냄으로써 레이블의 출처를 직접 인식하지 않도록 함
- 동영상의 자연적인 소비 패턴은 일반적으로 비대칭적인 공동 시청 확률을 유발
  - 에피소드 시리즈는 일반적으로 순차적으로 시청되며 사용자는 종종 가장 인기 있는 장르의 아티스트를 먼저 찾은 후 작은 치수에 중점을 두게 됨

    ⇒ 사용자의 다음 시청을 예측하는 것이 무작위로 유보된 시청을 예측하는 것보다 훨씬 성능이 우수




(a) Predicting held-out watch 임의 시청 (b) Predicting future watch 다음 시청

**Figure 5: Choosing labels and input context to the model is challenging to evaluate offline but has a large impact on live performance. Here, solid events • are input features to the network while hollow events ○ are excluded. We found predicting a future watch (5b) performed better in A/B testing. In (5b), the example age is expressed as $t_{\max} - t_N$ where $t_{\max}$ is the maximum observed time in the training data.**

- 기존의 협업 필터링 시스템은 종종 사용자 이력에서 무작위 항목을 유보하고 사용자 이력의 다른 항목에서 이를 예측함으로써 레이블과 맥락을 선택
  - 이는 미래 정보를 누출하고 비대칭적인 소비 패턴을 무시
  - 반면에 우리는 사용자의 이력을 "롤백"하여 임의의 시청을 선택하고 유보된 레이블 시청 전에 사용자가 취한 동작만 입력으로 사용

## 3-5. Experiments with Features and Depth

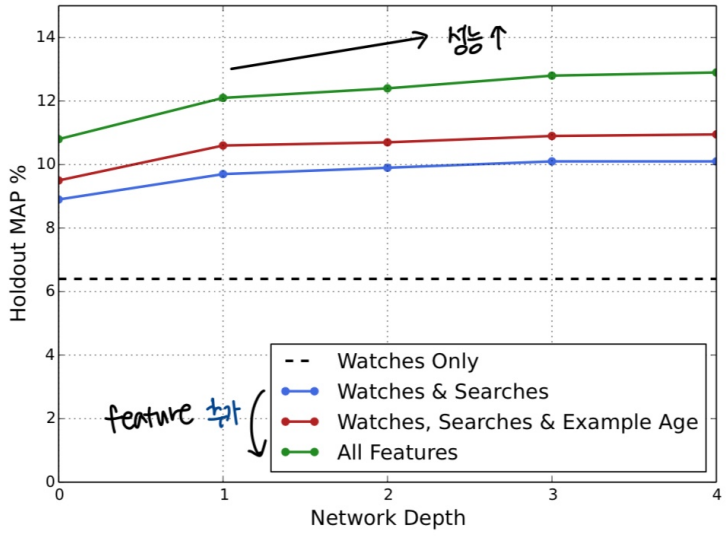



**Figure 6: Features beyond video embeddings improve holdout Mean Average Precision (MAP) and layers of depth add expressiveness so that the model can effectively use these additional features by modeling their interaction.**

- 기능과 깊이를 추가함으로써 정확도가 크게 향상되었음

- 해당 실험에서는 1백만 개의 비디오 어휘와 1백만 개의 검색 토큰이 사용되었으며, 각각 256 차원의 임베딩을 가진 최대 50개의 최근 시청 및 50개의 최근 검색이 포함된 가방 크기로 설정되었음
- 소프트맥스 레이어는 1백만 개의 비디오 클래스에 대한 256 차원의 다항 분포를 출력하며, 이러한 모델은 YouTube 사용자 전체에 대해 수렴될 때까지 교육되었음
- 추가된 너비와 깊이는 점진적인 이익이 감소하고 수렴이 어려워질 때까지 이루어졌음

# 4. Ranking

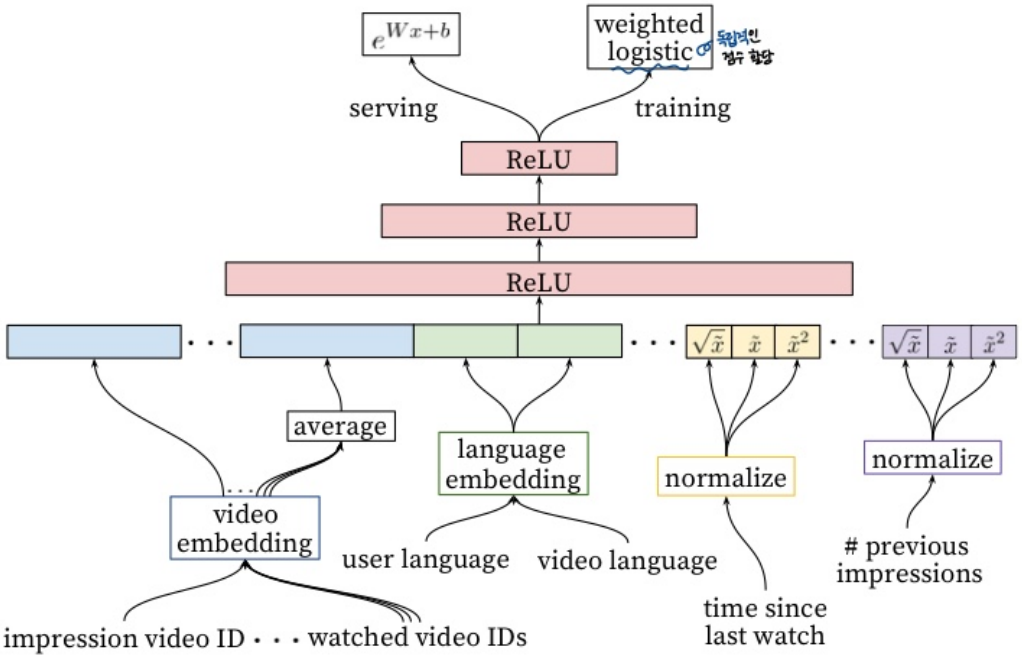



Figure 7: Deep ranking network architecture depicting embedded categorical features (both univalent and multivalent) with shared embeddings and powers of normalized continuous features. All layers are fully connected. In practice, hundreds of features are fed into the network.

- 랭킹의 주요 역할은 인상 데이터를 활용하여 특정 사용자 인터페이스에 맞게 후보 예측을 특수화하고 보정하는 것
  - 랭킹은 수백 개의 비디오에만 점수를 부여하는 반면, 후보 생성에서는 수백만 개의 비디오에 점수를 매기기 때문에 더 많은 비디오 및 사용자 관련 기능에 액세스

    → 더 세부적인 부분에 집중하겠다~
  - 랭킹은 또한 직접적으로 비교할 수 없는 여러 후보 소스를 결합하는 데 중요
    - DNN을 사용하여 각 비디오 인상에 독립적인 점수를 할당하고 이를 기반으로 사용자에게 정렬된 비디오 목록을 반환

- 최종 랭킹 목표는 일반적으로 인상당 예상 시청 시간을 기준으로 조정되며, 클릭 스루율보다는 시청 시간이 참여를 더 잘 나타내는 경향이 있음

## 4-1. Feature Representation

- 다양한 성격의 feature를 활용하여 추천을 수행
- 특징은 항목("인상")의 속성 또는 사용자/맥락("쿼리")의 속성을 기준으로 분류됨
- 쿼리 특징은 요청 당 한 번 계산되며, 인상 특징은 각 항목의 점수가 매겨질 때마다 계산됨

### Feature Engineering

- 랭킹 모델에서는 수백 개의 특징을 사용
  - 범주형과 연속적인 특징을 균등하게 분할
  - 이진 특징(예: 사용자의 로그인 여부), 수백만 개의 가능한 값을 가지는 특징(예: 사용자의 최근 검색 쿼리) 등 매우 다양한 성격을 지님
- 딥러닝이 특징 엔지니어링 부담을 줄이겠다는 약속에도 불구하고, 원시 데이터의 특성으로 인해 특징을 유용한 형태로 변환하는 데 많은 엔지니어링이 필요
  - 사용자 작업의 시간적 순서를 모델링하는 것이 주요 도전이며, 사용자의 이전 상호작용과 비디오 시청 히스토리가 중요한 신호로 작용
- 특히, 후보 생성에서 랭킹으로 정보를 전파하는 것이 중요하며, 과거 비디오 인상의 빈도를 고려하여 추천 모델을 민첩하게 유지하는 것이 권장됨

### Embedding Categorical Features

- 희소한 범주형 특징을 신경망에 입력하기 위해 임베딩을 사용
  - 각 고유 ID 공간에는 학습된 임베딩이 있으며, 이 임베딩은 고유 값 수에 비례하여 차원이 증가
- 어휘는 교육 시작 전에 데이터를 통과하여 조회 테이블을 구축하며, 큰 기수 ID 공간은 빈도를 기준으로 상위 N개만을 유지
- 특징의 임베딩은 여러 값을 가진 경우 평균화되며, 동일한 ID 공간의 범주형 특징은 공통의 임베딩을 공유
  - 이 공유는 일반화를 향상시키고 교육 속도를 높이며 메모리 요구 사항을 줄임
  - 매개 변수의 대다수는 고차원 cardinality 임베딩 공간에 위치하며, 이는 더 적은 매개 변수로 높은 효율성을 제공

### Normalizing Continuous Features

- 신경망은 입력의 스케일링과 분포에 대해 매우 민감하며, 대안적인 방법(예: 의사 결정 트리의 앙상블)은 개별 특징의 스케일링에 불변
- 연속적인 특징의 적절한 정규화는 수렴에 매우 중요
  - 분포 f를 가진 연속 특징 x는 해당 특징 값의 누적 분포를 사용하여 [0; 1)에서 특징이 균일하게 분포되도록 값을 축소하여 $\tilde{x}$로 변환됨
- 훈련이 시작되기 전에 데이터를 한 번 통과하여 계산된 특징 값의 분위수에서 선형 보간으로 근사
  - 정규화된 원시 특징 x뿐만 아니라$x^2$ 및 $\sqrt{x}$의 제곱을 추가로 입력하여 특징의 super- and sub-linear function을 쉽게 형성할 수 있도록 함
  - 연속 특징의 거듭제곱을 공급하면 오프라인 정확도가 향상될 수 있었음

## 4-2. Modeling Expected Watch Time

- 목표) 훈련 예제에서 양성(클릭된 비디오 인상) 및 음성(클릭되지 않은 인상)을 고려하여 예상 시청 시간을 예측하는 것
- 가중 로지스틱 회귀 기술을 사용하여 모델을 훈련하고, 양성 인상은 관찰된 시청 시간에 따라 가중치를 받음
  - $odds = \frac{\sum T_i}{N-k}$
    - $N$: # of training examples
    - $k$: # of positive impressions
    - $T_i$: watch time of the ith impression
  - 이를 통해 학습된 로지스틱 회귀의 오즈는 양성 인상의 시청 시간을 고려하여 추정
  - 최종적으로 지수 함수를 사용하여 예상 시청 시간을 근사화

## 4-3. Experiments with Hidden Layers

| 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.**

- 은닉 레이어의 너비와 깊이를 늘리면 결과가 향상됨
  - 이에 대한 희생은 추론에 필요한 서버 CPU 시간의 증가임
- 최상의 성능을 제공하는 구성은 1024-너비 ReLU, 그 뒤에 512-너비 ReLU, 그리고 256-너비 ReLU로 이루어진 것으로, 서빙 CPU 예산 내에 남을 수 있었음
- 정규화된 연속 특징의 제곱을 사용하지 않고 훈련하는 경우에 모델 성능이 어떻게 변하는지를 확인하였음
  - 손실이 0.2% 증가
- 추가로, 양성 및 음성 예제에 대해 동일한 가중치를 적용하여 모델을 훈련한 경우 시청 시간 가중치 손실이 4.1% 크게 증가하였음

# 5. Conclusions

- YouTube 동영상 추천을 위한 두 가지 주요 문제인 후보 생성과 랭킹을 해결하기 위한 딥 뉴럴 네트워크 아키텍처를 소개함
- 우리의 딥 협업 필터링 모델은 다양한 신호를 효과적으로 통합하며 깊은 레이어를 사용하여 지난 행렬 인수 분해 방법을 능가했음
- 추천 시스템에서 성능을 측정하는 데 있어서는 대체로 예술적인 측면이 많이 관여하며 미래 시청을 예측하는데 있어 비대칭한 상호작용을 고려하고 미래 정보 누출을 방지하기 위한 효과적인 대리 문제를 선택하는 것이 중요

    - 특히 특성을 분류기로부터 잘 보호하여 모델이 대리 문제에 오버피팅되지 않도록 하는 것이 핵심
    - 훈련 예제의 나이를 특성으로 사용하여 과거 편향을 줄이고 모델이 시간에 따라 인기 있는 동영상의 행동을 잘 반영하도록 했음

  ⇒ 이로써 오프라인 정확도가 향상되었고 A/B 테스트에서 최근 업로드된 동영상에 대한 시청 시간이 크게 증가

- 또한, 랭킹 문제에서는 딥 러닝 접근 방식이 기존의 선형 및 트리 기반 방법을 능가
    - 추천 시스템에서는 사용자의 이전 행동과 관련된 특성을 효과적으로 활용할 수 있도록 했으며, 딥 뉴럴 네트워크에서는 범주형 및 연속적인 특성을 임베딩 및 분위수 정규화를 통해 특별한 표현으로 변환
    - 이러한 방법을 통해 클릭률 예측보다 시청 시간을 중점으로 한 랭킹 평가 지표에서 우수한 결과를 얻을 수 있었음