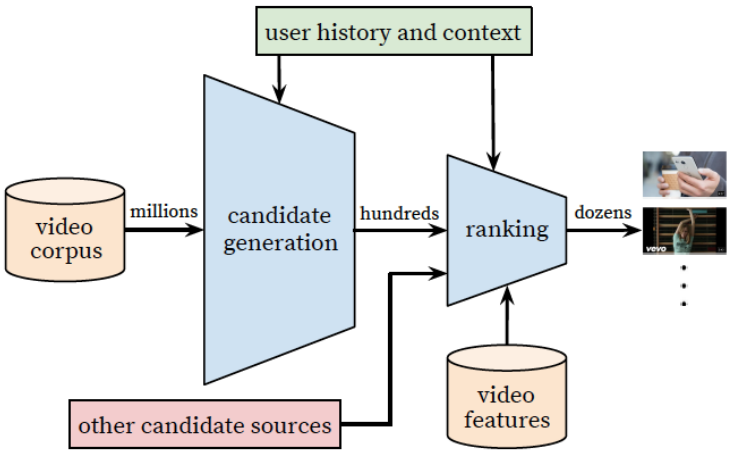
Deep Neural Networks for Youtube Recommendations

1. Intro

- 挑战主要来自于三方面
  - Scale: Youtube的用户和数据集规模需要高度专业的分布式学习算法和高效的serving系统。
  - Freshness: 变化非常快的数据集，每秒都会有数小时的新视频上传。平衡新老视频是一个EE问题
  - Noise: 基本无法获得用户满意度的Ground Truth, 只能获得有噪声的隐式反馈信号. 内容的元数据是非结构化的。
- 基于TF搭建，10亿参数，千亿样本规模

2. System overview



- 两部分组成：candidate generation和ranking
- Retrieval: 用户的历史行为作为输入，产出youtube视频集合的一个子集. 选出的候选集为和用户大致相关.
- Ranking: 对Retrieval产出的候选集排序
- two-stage的方法还有个好处: 可以混入从其他来源产生的candidates

3. Candidate generation

- 之前用的是matrix factorizaion+rank loss.
- 现在已经使用NN替代, **NN可以看作是factorization方法的非线性泛化**.

3.1 Recommendation as Classification

- 分类 t时刻，观看一个视频w_t的概率。$u \in \mathbb{R}^N$表示用户的embedding，$v_j \in \mathbb{R}^N$表示视频的embedding。

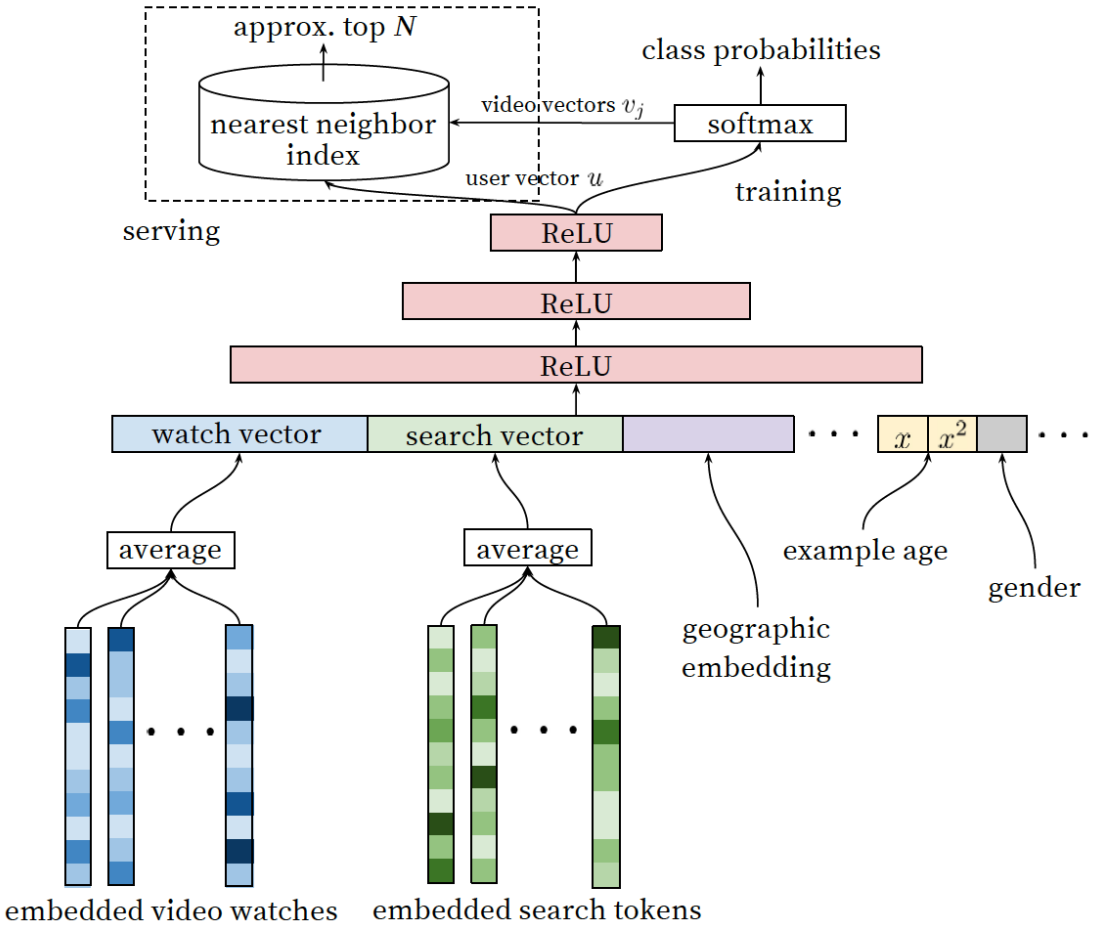
$$P(w_t = i|U, C) = \frac{e^{v_i u}}{\sum_{j \in V} e^{v_j u}}$$

- 使用隐式反馈标注，因为显式反馈(like)太稀疏

Efficient Extreme Multiclass

- 几百万个类的softmax + negative sampling
- 尝试过hierarchical softmax，效果并不好
- serving time: 用hashing方法选出最有可能的N个类. 选哪种最近邻查找方法效果都差不多

3.2 Model Architecture



3.3 Heterogeneous Signals

- 使用DNN代替MF，连续和离散特征都能够加入
- 离散特征都是通过embedding方式加入

"Example Age" Feature

- 模型偏向于过去的内容，视频欢迎程度 训练数据上的建模反应的是几星期的训练窗口内的平均的观看的可能性
- 修正方法是使用训练数据的age作为特征，serving的时候此特征为0或负数

$$ExampleAge = t_max - t_N$$

t_max是训练数据中的最大时间戳

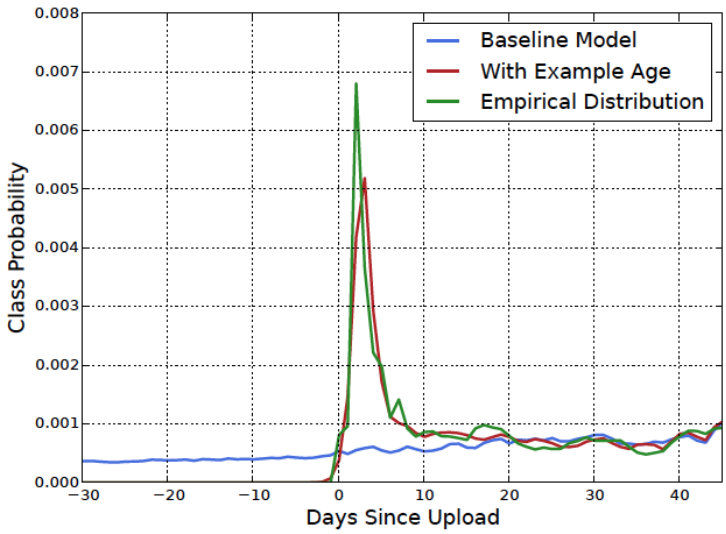
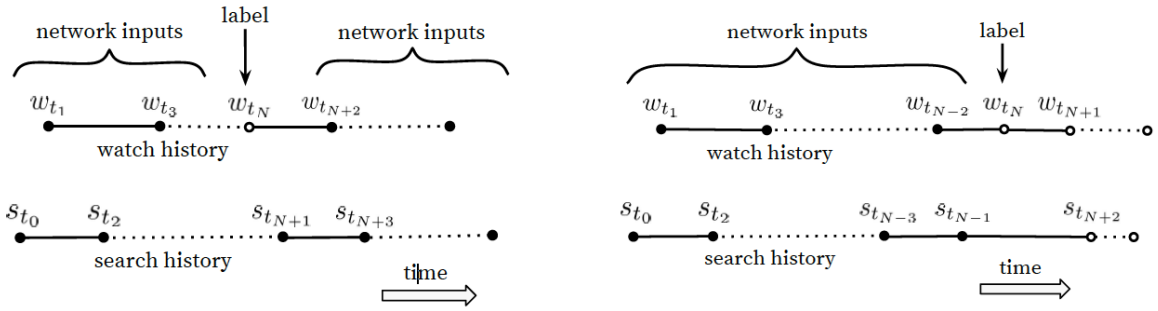


Figure 4: For a given video [26], the model trained with example age as a feature is able to accurately represent the upload time and time-dependant popularity observed in the data. Without the feature, the model would predict approximately the average likelihood over the training window.

3.4 Label and Context Selection

- 选择代理问题(surrogate problem)对效果影响非常大，但这种影响不容易离线评估，在A/B test中评估效果非常明显。
- 在这个模型中选择的代理问题是预估下一个有可能看的视频，类似于CBOW
- 训练使用全部的youtube watches而不是recommendor产生的，否则容易倾向于exploitation.
- 每个用户至多保留固定数量的样本，防止loss被那些超级活跃的用户主导
- 需要特别小心防治模型过拟合到代理问题。比如刚搜过Taylor Swift，就推荐talor swift的视频，效果很差。在本模型中解决方法是丢弃掉序列信息，采用unordered bag of tokens.
- label采用predicting future watch而不是random hold-out watch。因为人的观看模式导致视频都被观看的条件概率是不对称的。这个和CBOW的训练是有区别的



(a) Predicting held-out watch

(b) Predicting future watch

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.

3.5 Experiments with Features and Depth

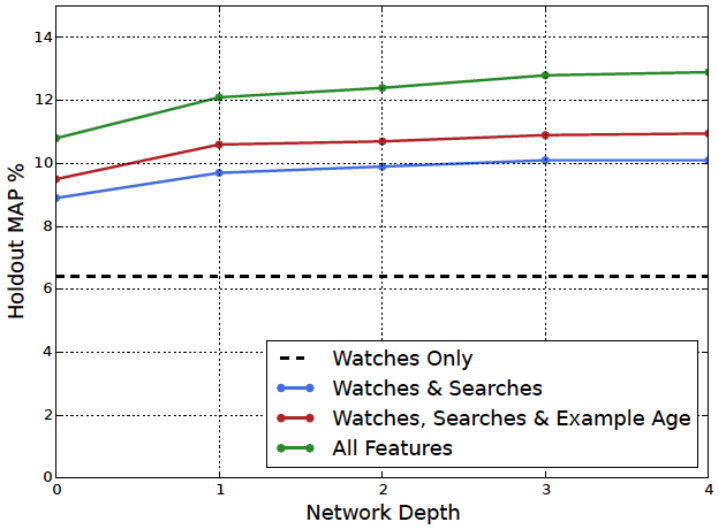


Figure 6: Features beyond video embeddings improve holdout Mean Average Precision (MAP) and layers of depth add expressiveness so that the model can effectively use these additional features by modeling their interaction.

4 Ranking

- 要解决的问题是根据曝光的场景对某个特定用户调整候选集。比如视频非常相关，但是缩略图不吸引该用户可能导致用户不点击
- Ranking的对于一次请求的量级缩减到百个视频，所以可以使用更多的特征
- Ranking模型对于整合不同来源的候选集非常重要，否则它们的score是没法对比的
- objective: 视频观看时间的简单函数，ctr可能导致clickbait问题

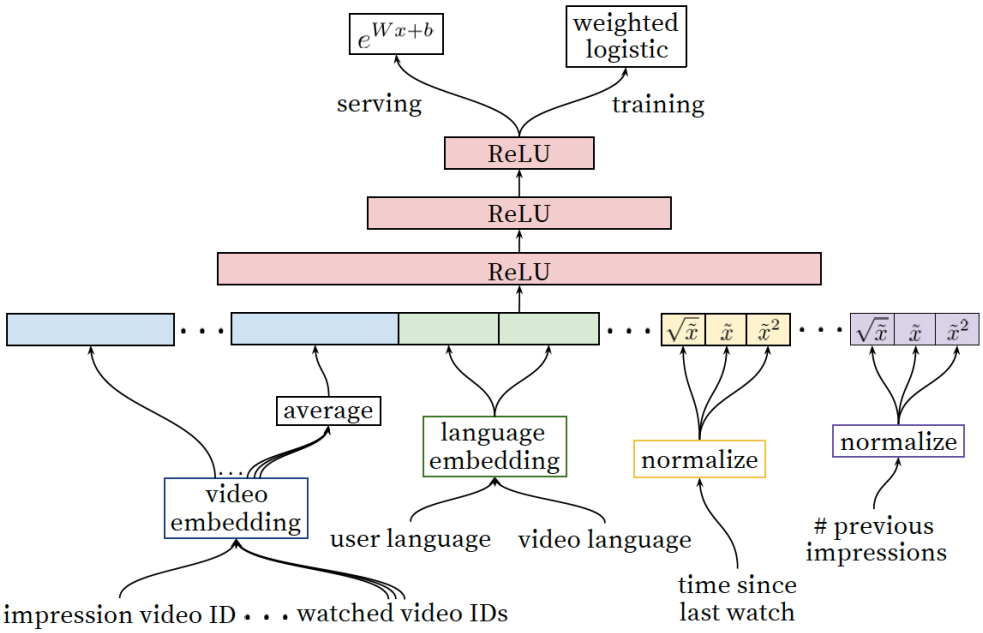


Figure 7: Deep ranking network architecture depicting embedded categorical features (both univalent and multivalent) with shared embeddings and powers of normalized continuous features. All layers are fully connected. In practice, hundreds of features are fed into the network.

4.1 Feature Representation

- 单值离散特征和多值离散特征：例如当此观看的视频和历史上观看过的视频
- 使用了请求级别和展现级别的特征

Feature Engineering

- 最有用的信息是那些描述用户过去和该视频或类似视频的交互信息。比如用户在某个频道的观看次数、用户上次观看该主题视频的时间
- 之前视频推荐展现的频次对于引入"churn"很重要，能够使的前后两次请求不会返回完全相同的结果
- 秒级的曝光、观看历史信息是工程挑战

Embedding Categorical Features

- 离散特征使用embedding，而且相同ID空间的embedding共享。OOV 映射到零embedding
- 大部分参数是embedding，1M ID的32维的embedding大约是2048个结点的全连接层参数的7倍

Normalizing Continuous Features

- 连续特征归一化到[0,1)
- 加入特征的平方和开方等特征以获得更多表达能力

4.2 Modeling Expected Watch Time

- 输出层使用logistic regression和cross entropy loss。
- 计算loss时将正样本按照观看时间加权，负样本权重为1，相当于按照观看时长复制正样本，那么在正样本占比很小的时候，logit就近似于观看时间的期望
- 预测时的输出需要调整为e^{wx+b}

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.