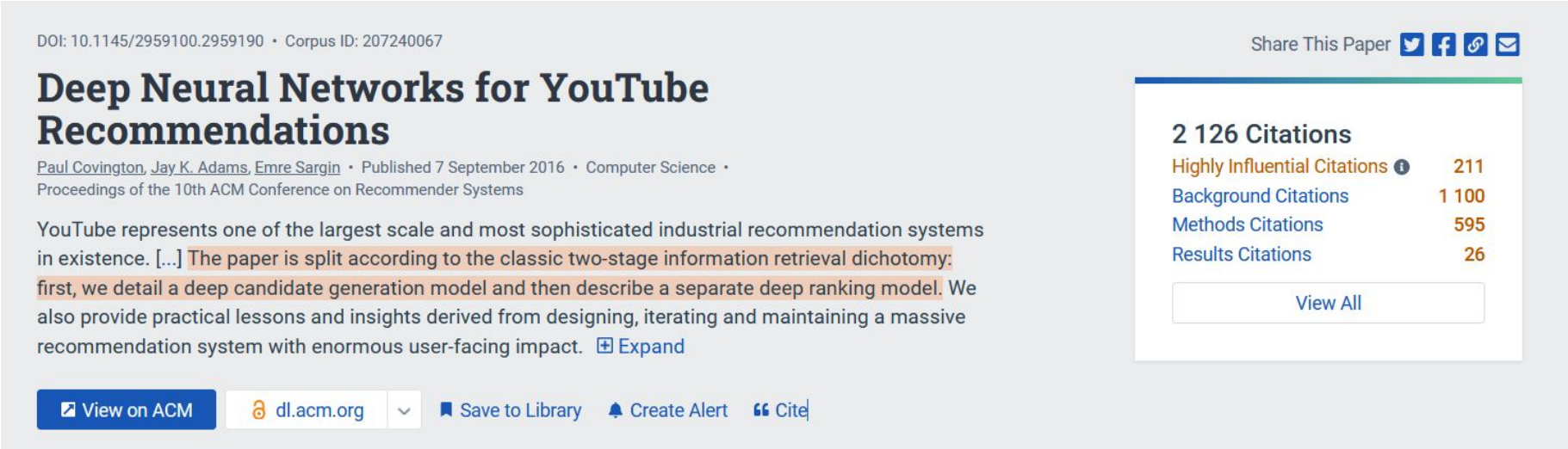
Deep Neural Networks for YouTube Recommendations

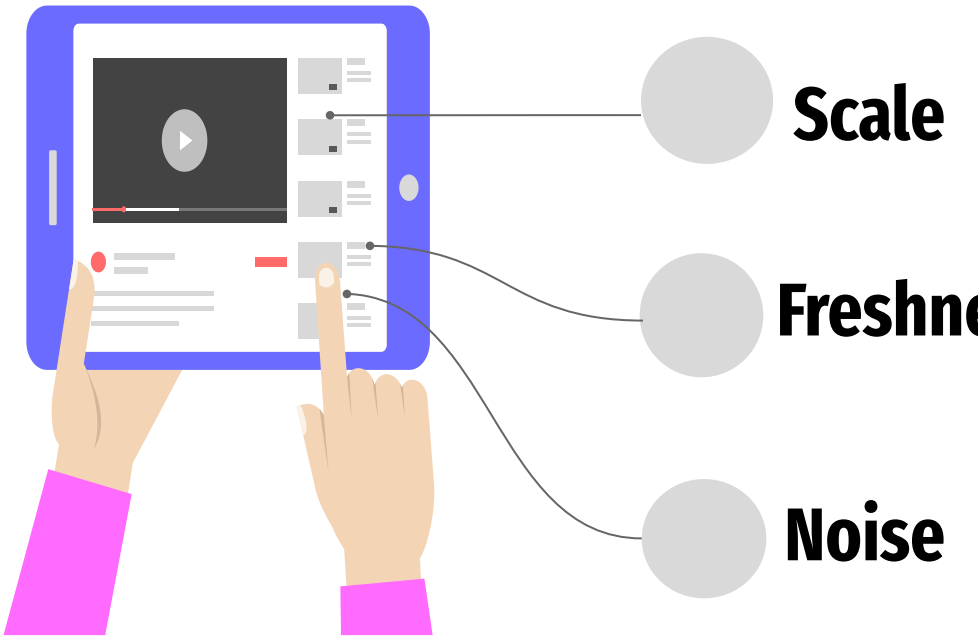
Authors:
Paul Covington,
Jay Adams,
Emre Sargin

Докладчик: Ковалева Илона
RecSys Reading Group https://t.me/ods_recommender_systems

Impact


DOI: 10.1145/2959100.2959190 • Corpus ID: 207240067
Deep Neural Networks for YouTube Recommendations
Paul Covington, Jay K. Adams, Emre Sargin • Published 7 September 2016 • Computer Science •
Proceedings of the 10th ACM Conference on Recommender Systems
YouTube represents one of the largest scale and most sophisticated industrial recommendation systems in existence. [...] The paper is split according to the classic two-stage information retrieval dichotomy: first, we detail a deep candidate generation model and then describe a separate deep ranking model. We also provide practical lessons and insights derived from designing, iterating and maintaining a massive recommendation system with enormous user-facing impact. Expand
View on ACM
dl.acm.org
Save to Library
Create Alert
Cite
Share This Paper
2 126 Citations
Highly Influential Citations 211
Background Citations 1 100
Methods Citations 595
Results Citations 26
View All

Challenges



Scale

- *Scale:* Many existing recommendation algorithms proven to work well on small problems fail to operate on our scale. Highly specialized distributed learning algorithms and efficient serving systems are essential for handling YouTube's massive user base and corpus.

Freshness

- *Freshness:* YouTube has a very dynamic corpus with many hours of video are uploaded per second. The recommendation system should be responsive enough to model newly uploaded content as well as the latest actions taken by the user. Balancing new content with well-established videos can be understood from an exploration/exploitation perspective.

Noise

- *Noise:* Historical user behavior on YouTube is inherently difficult to predict due to sparsity and a variety of unobservable external factors. We rarely obtain the ground truth of user satisfaction and instead model noisy implicit feedback signals. Furthermore, metadata associated with content is poorly structured without a well defined ontology. Our algorithms need to be robust to these particular characteristics of our training data.

2016: “relatively little work using deep neural networks for recommendation systems”

tion methods [19], there is relatively little work using deep neural networks for recommendation systems. Neural networks are used for recommending news in [17], citations in [8] and review ratings in [20]. Collaborative filtering is formulated as a deep neural network in [22] and autoencoders in [18]. Elkahky *et al.* used deep learning for cross domain user modeling [5]. In a content-based setting, Burges *et al.* used deep neural networks for music recommendation [21].

S. Sedhain, A. K. Menon, S. Sanner, and L. Xie. Autorec: Autoencoders meet collaborative filtering. In Proceedings of the 24th International Conference on World Wide Web, WWW ’15 Companion, pages 111–112, New York, NY, USA, 2015. ACM.

H. Wang, N. Wang, and D.-Y. Yeung. Collaborative deep learning for recommender systems. In Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD ’15, pages 1235–1244, New York, NY, USA, 2015. ACM.

A. van den Oord, S. Dieleman, and B. Schrauwen. Deep content-based music recommendation. In Advances in Neural Information Processing Systems 26, pages 2643–2651. Curran Associates, Inc., 2013

W. Huang, Z. Wu, L. Chen, P. Mitra, and C. L. Giles.A neural probabilistic model for context based citation recommendation. In AAAI, pages 2404–2410, 2015.

System overview

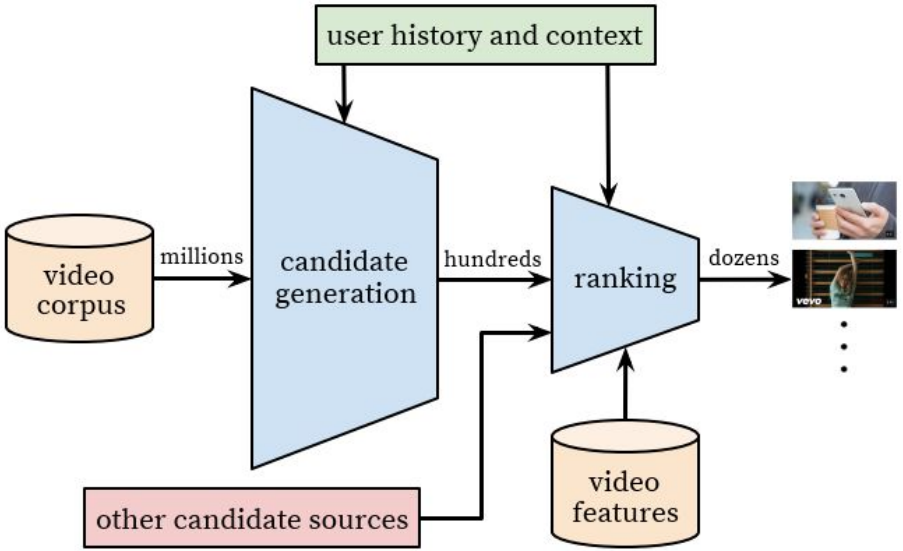


Figure 2: Recommendation system architecture demonstrating the "funnel" where candidate videos are retrieved and ranked before presenting only a few to the user.

CANDIDATE GENERATION

3.1 Recommendation as Classification

We pose recommendation as extreme multiclass classification where the prediction problem becomes accurately classifying a specific video watch w_t at time t among millions of videos i (classes) from a corpus V based on a user U and context C,

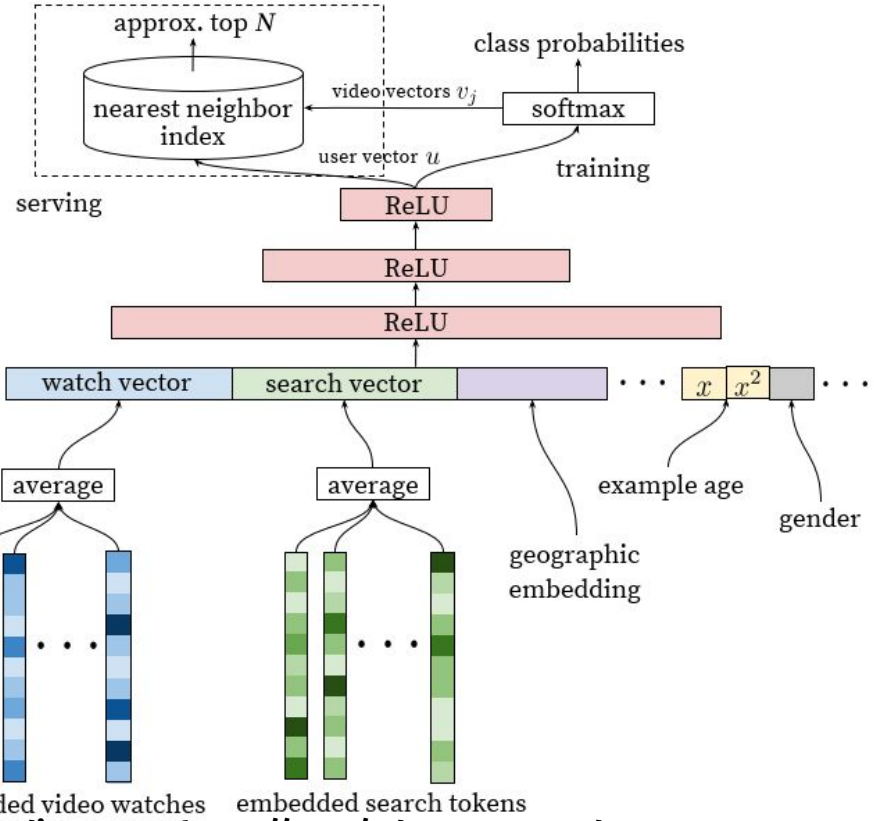
$$P(w_t = i|U, C) = \frac{e^{v_i u}}{\sum_{j \in V} e^{v_j u}}$$

where $u \in \mathbb{R}^N$ represents a high-dimensional "embedding" of the user, context pair and the $v_j \in \mathbb{R}^N$ represent embeddings of each candidate video. In this setting, an embedding is simply a mapping of sparse entities (individual videos, users etc.) into a dense vector in $\mathbb{R}^N$. The task of the deep neural network is to learn user embeddings u as a function of the user's history and context that are useful for discriminating among videos with a softmax classifier.

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

w_t - просмотр видео, в период времени t принадлежащего классу i
U - пользователь
C - контекст
u - эмбеддинг пары пользователь - контекст
v_j - эмбеддинг видео

CANDIDATE GENERATION: Model Architecture



"Example Age" Feature

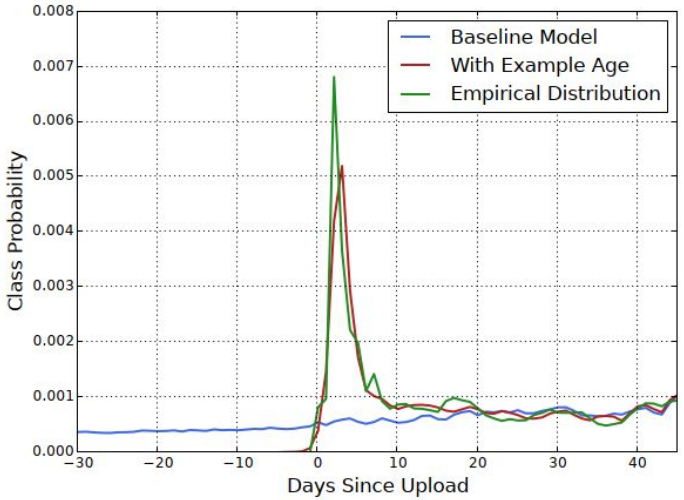


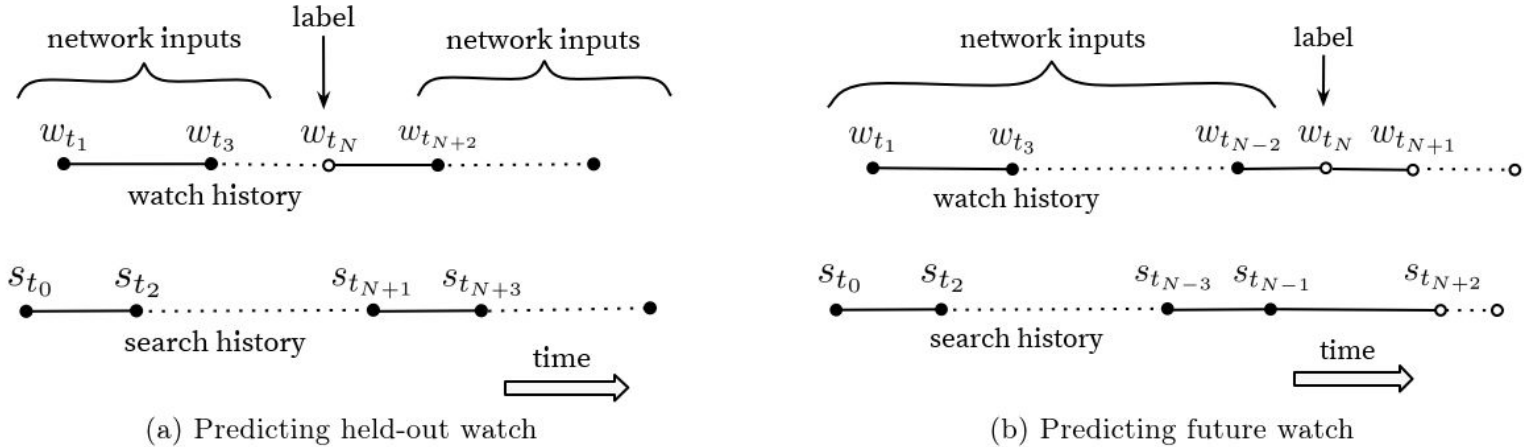
Figure 4: For a given video [26], the model trained with example age as a feature is able to accurately represent the upload time and time-dependant popularity observed in the data. Without the feature, the model would predict approximately the average likelihood over the training window.

Label and Context Selection: key points

- **Training examples are generated from all YouTube watches** (even those embedded on other sites)
- **Generate a fixed number of training examples per user,** effectively weighting our users equally in the loss function.
- By **discarding sequence information** and representing **search queries** with an unordered bag of tokens.

Label and Context Selection

"Much **better performance predicting the user's next watch**, rather than predicting a randomly held-out watch"



(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.

Experiments with Features and Depth

- 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

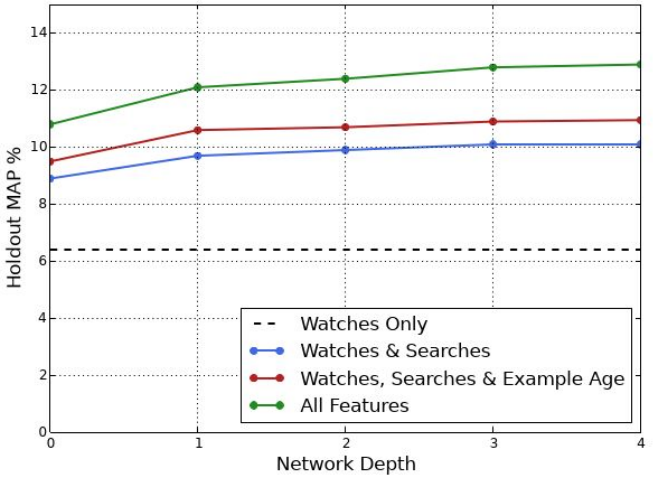


Figure 6: Features beyond video embeddings improve holdout Mean Average Precision (MAP) and layers of depth add expressiveness so that the model can effectively use these additional features by modeling their interaction.

RANKING

N - is the number of training examples,
k - is the number of positive impressions
T_i - is the watch time of the ith impression

$$\frac{\sum T_i}{N-k}$$

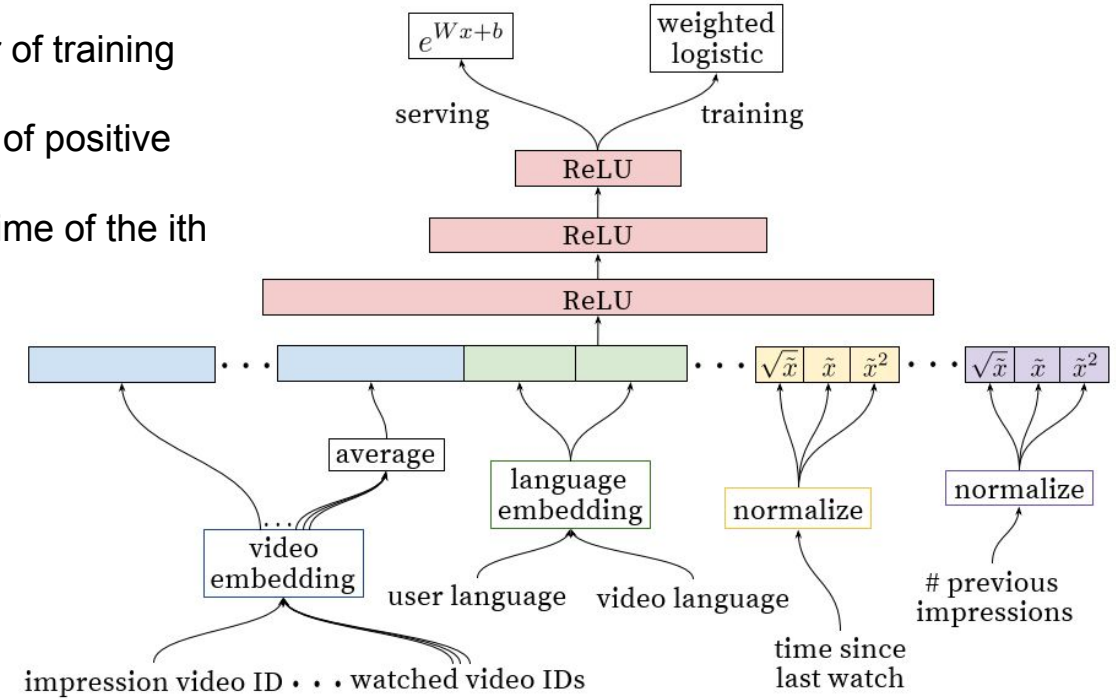


Figure 7: Deep ranking network architecture depicting embedded categorical features (both univalent and multivalent) with shared embeddings and powers of normalized continuous features. All layers are fully connected. In practice, hundreds of features are fed into the network.

Assuming the fraction of positive impressions is small, the learned odds are approx. **E[T](1 + P)**,
P - is the click probability
E[T] - is the expected watch time of the impression. Since P is small, this product is close to E[T].

RANKING

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.

- normalized continuous features without their powers increased loss by 0.2%
- positive and negative examples are weighted equally - increased 4.1%

Спасибо за внимание!