Poster: Can Recommenders Compensate for QoS?

Mateus Nogueira,
Daniel Menasché
Federal Univ. of Rio de Janeiro,
Brazil

Pavlos Sermpezis
Aristotle University of
Thessaloniki, Greece

Thrasyvoulos
Spyropoulos
EURECOM, France

ABSTRACT

Content recommendation systems (recommenders) are pervasive. Although recommenders have been primarily devised to account for users interests with respect to the content catalog, it is a matter of fact that mobile users are typically served by unreliable networks, subject to losses and low QoS. Can content recommenders compensate for low QoS? We conducted measurements over the Internet, and verify that making requests a bit trendier can hit much closer content, suggesting conditions under which recommenders can compensate for low QoS, at zero costs for the operator.

ACM Reference Format:
Mateus Nogueira, Daniel Menasché, Pavlos Sermpezis, and Thrasyvoulos Spyropoulos. 2021. Poster: Can Recommenders Compensate for QoS?. In *Proceedings of ACM IMC'21.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Content recommenders and caches are two fundamental pillars in the Internet ecosystem. While recommenders such as those used by Netflix influence a significant portion of users demands, caches such as deployed by Akamai serve a vast amount of content to end users. Caches reduce the load on custodians, decrease the latency for users, and benefit network infrastructure reducing the traffic over bottlenecks.

Given the benefits of caching, a significant effort has been invested in order to improve cache performance. In particular, similarity caching [1] and cost-aware caching [8] are some of the various recent developments in that domain [2, 5, 7]. Such advances, in turn, suggest novel opportunities but also pose new challenges in the realm of content distribution.

The basic idea behind similarity caching and cost-aware caching consists in determining both the similarity between contents and the cost to serve and/or retrieve a content, and then make decisions about which content to store and/or serve based on such assessments. The multiple dimensions involved in the problem are intertwined, and optimal decisions are non trivial. In particular, a user consuming a content not stored in a local cache may experience low quality of service (QoS), and may prefer to rely on a content recommender to find a title that suits its expectations both in terms of content as well as in terms of QoS, motivating our research question: *can a recommender compensate for low QoS?*

In this poster, we report results on Internet measurements indicating the feasibility of characterizing the QoS at which different items can be served. Given such characterization, and information about the content recommendation graph, we present findings to support conditions under which we have an affirmative answer to our main question.

Related work. There is a large body of work on recommenders and QoS, and on the interplay between the two [3, 6]. However, to the best of our knowledge there are no measurements to *(a)* quantify the extent at which QoS degrades as a function of the trendiness of contents, and *(b)* assess the extent at which recommenders can compensate for QoS. Our goal is to fill that gap, presenting measurements of the content recommendation graph and of the delays incurred to access those contents.

2 METHODOLOGY

Popular content is known to be cached close to users. In YouTube, a list of trending contents is presented to users at their home page. Beyond such remarkably trending contents, which other contents are cached closer to users? How do different features, such as number of views, impact closeness? To answer those questions, we rely on YouTube recommender and network measurements. We use YouTube API to access the recommendation system and generate recommendation graphs for each trending content by performing a Breadth-First Search (BFS) through the network of recommendations [3]. Then, we emulate a request towards each of the videos, and determine the server providing that video.

This research is co-financed by Greece and the European Union (European Social Fund- ESF) through the Operational Programme "Human Resources Development, Education and Lifelong Learning" in the context of the project "Reinforcement of Postdoctoral Researchers - 2nd Cycle" (MIS-5033021), implemented by the State Scholarships Foundation (IKY).

ACM IMC'21, November, Hosted online
© 2021 Association for Computing Machinery.
ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

We measured network level features, using ping and traceroute towards video servers, and group the videos into two clusters based on those metrics. Let C denote an indicator variable, equal to 1 if our measurements suggest that the video is cached close to our measurement vantage point, and 0 otherwise. Then, we computed correlations between C and metrics related to the recommenders, as detailed below.

Videos distances from recommender viewpoint. To determine the distance between videos, we consider a walk in the recommendation tree, where nodes and edges correspond to videos and video recommendations. The children of a node are ordered based on their position in the recommendation list, which is also referred to as the node *width*. Correspondingly, the distance between any node and the root is the node *depth*. In case of repeated recommendations, the first one from the root is taken into account for the purpose of computing the above metrics (see [3] for details).

Recommender and network correlation. Let W and D be the video width and depth, respectively. We observed negative correlations of -0.19 and -0.23 between C and the above two metrics, respectively. Such correlations quantify the tendency that videos closer to the root of the recommender graph are closer in the cache network, and motivate the analysis that follows.

3 RECOMMENDERS AGAINST LOW QOS

To answer our main research question, we conduct experiments simulating scenarios in which a user randomly starts selecting one of the most trending videos, followed by videos from subsequent recommendation lists. These lists are presented in 2 ways: *(i)* the original order, i.e., like they would be in YouTube and *(ii)* according to an algorithm that prioritizes cached videos, the Cache-Aware & BFS-related Recommendations (CABaRet) [3]. CABaRet recommendations replace some non-cached videos by cached counterparts, and order the videos in a way that cached videos are preferably presented at the top of the recommendation lists.

To determine whether a video is cached or not, we leverage our measurements as described in the previous section. In particular, we use the inferred indicator C to determine whether a video is cached, and to eventually increment hit counts. While the authors of the CABaRet algorithm assumed, shrewdly, that the top 50 trending videos were cached, our measurements allow the application of CABaRet algorithm with more information regarding the network conditions of the media servers.

We compare the cache-hit ratio (CHR) produced by YouTube's lists of recommendations against the lists generated by CABaRet. To that aim, we vary the mechanism through which users select a video from a recommendation list, considering two alternatives: uniform and Zipf. The uniform distribution assumes users select videos uniformly at random, whereas the Zipf distribution captures a preference towards videos ranked in top positions [4]. We also vary the two main CABaRet parameters: maximum depth ($\widetilde{D}$) and maximum width ($\widetilde{W}$). Larger values correspond to broader searches for cached contents in the recommendation graph, providing more flexibility for recommenders to compensate for QoS. In particular, when $\widetilde{D} = 1$ CABaRet only reorders the recommendations, whereas for $\widetilde{D} = 2$ it also replaces some non-cached videos by cached alternatives.

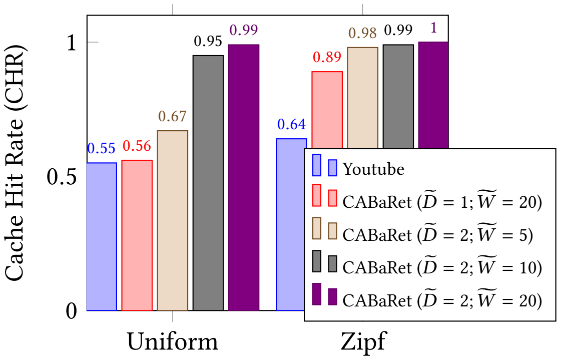

Figure 1: CHR varying workloads and recommenders

Figure 1 shows that CABaRet easily achieves a higher CHR than Youtube baseline. When the request workload is uniform, CABaRet requires larger $\widetilde{D}$ and $\widetilde{W}$ to show its benefits. This is because both replacements and reorderings of recommendations affect CHR under the Zipf workload, whereas the uniform workload is insensitive to reorderings.

Summary. Combining recommender and network measurements, we learned that recommendation reorderings are sufficient to increase CHR from 0.64 to 0.89 under a Zipf workload. Diminishing returns are gained by allowing for replacements of recommendations, in addition to reorderings. In particular, allowing for replacements of videos that are at most two hops away in the recommendation graph suffices to reach a CHR of 0.98.

REFERENCES

[1] Garetto, M., Leonardi, E., and Neglia, G. Similarity caching: Theory and algorithms. In *Proc. IEEE INFOCOM* (2020).

[2] Giannakas, T., et al. Show me the cache: Optimizing cache-friendly recommendations for sequential content access. In *WoWMoM* (2018).

[3] Kastanakis, S., Sermpezis, P., et al. Cabaret: Leveraging recommendation systems for mobile edge caching. In *SIGCOMM workshop* (2018).

[4] Krishnappa, D. K., Zink, M., Griwodz, C., and Halvorsen, P. Cache-centric video recommendation. *ACM TOMM 11*, 4 (2015), 48.

[5] Sermpezis, P., et al. Soft cache hits: Improving performance through recommendation and delivery of related content. *JSAC* (2018).

[6] Sermpezis, P., Kastanakis, S., Pinheiro, J. I., et al. Towards QoS-aware recommendations. *arXiv:1907.06392* (2020). CARS (RecSys).

[7] Sermpezis, P., Spyropoulos, T., Vigneri, L., and Giannakas, T. Femto-caching with soft cache hits. In *GLOBECOM* (2017), pp. 1–7.

[8] Zheng, Z., et al. QoS-aware web service recommendation by collaborative filtering. *IEEE Transactions on services computing 4*, 2 (2011).