When Locality is not enough: Boosting Peer Selection of Hybrid CDN-P2P Live Streaming Systems using Machine Learning

Zhejiayu Ma, Soufiane Roubia
Easybroadcast
Nantes, France
{ma.zhejiayu, rouibia}@easybroadcast.fr

Frédéric Giroire and Guillaume Urvoy-Keller
Université Côte d'Azur, CNRS
Sophia-Antipolis, France
{frederic.giroire, guillaume.urvoy-keller}@univ-cotedazur.fr

***Abstract*—Live streaming traffic represents an increasing part of the global IP traffic. Hybrid CDN-P2P architectures have been proposed as a way to build scalable systems with a good Quality of Experience (QoE) for users, in particular, using the WebRTC technology which enables real-time communication between browsers and, thus, facilitates the deployment of such systems. An important challenge to ensure the efficiency of P2P systems is the optimization of peer selection. Most existing systems address this problem using simple heuristics, e.g. favor peers in the same ISP or geographical region.**

We analysed 9 months of operation logs of a hybrid CDN-P2P system and demonstrate the sub-optimality of those classical strategies. Over those 9 months, over 18 million peers downloaded over 2 billion video chunks. We propose learning-based methods that enable the tracker to perform adaptive peer selection. Furthermore, we demonstrate that our best models, which turn out to be the neural network models can (i) improve the throughput by 22.7%, 14.5%, and 6.8% (reaching 89%, 20.4%, and 24.3% for low bandwidth peers) over random peer, same ISP, and geographical selection methods, respectively (ii) reduce by 18.6%, 18.3%, and 16% the P2P messaging delay and (iii) decrease by 29.9%, 29.5%, and 21.2% the chunk loss rate (video chunks not received before the timeout that triggers CDN downloads), respectively.

***Index Terms*—hybrid P2P, live streaming, peer selection, machine learning**

I. Introduction

Live streaming video traffic will represent 14 percents of all IP traffic by 2022, approximately 55.4 Exabytes per month [1]. A hybrid CDN-P2P architecture is commonly used to build live video streaming systems. This architecture aims at combining the QoE of CDN-based systems and the scalability of P2P-based systems. In this approach, the video is broken into chunks, similarly to the Video on demand (VoD) case, and a peer tries to obtain the next chunks from other peers, reverting to the CDN only if the chunks are not received within a given time interval, the P2P timeout, typically in the order of a few chunks duration (usually 8 seconds). This strategy alleviates the load and, thus, the cost, of the CDN servers.

The WebRTC technology [2] enables direct P2P communications between browsers with no need to install any third party software, easing users' adoption. A number of commercial solutions including Akamai [3], Easybroadcast [4], and CDNetworks [5] now rely on WebRTC to implement their CDN-P2P architectures.

A key parameter to reduce the load on the CDN is the peer selection algorithm. When a peer joins a channel, it contacts a tracker – a central controller under the control of the broadcaster that acts as rendezvous point for the new peers – that provides it a set of candidate neighbours. A classical strategy is to favour peers in the same ISP or geographical region [6].

We revisit the peer selection problem by analysing a large-scale hybrid CDN-P2P live streaming system. We collected and analysed a 9-month long operation log of a multi-channel IPTV package popular in Europe and North Africa. The tracker initially implemented a classical ISP pairing approach. In order to collect an unbiased sample, we modified this default policy to a purely random one. As video is a bandwidth-hungry service, we address the problem of identifying the best neighbors, in terms of throughput, of a peer. To this end, we compared a number of peer selection strategies starting from the initial random approach: ISP-based, city-based peer selection, linear regression, decision trees, and different neural networks approaches.

Our focus in this work is on the offline version of the problem[1]. It means that, based on our extensive and unbiased (see Section II-A) dataset, we evaluate the effectiveness of different peer selection strategies. It is a necessary step before modifying an operational platform like the one from which the dataset was collected. Our main contributions are summarized as follows. We found that (1) for some ISPs, choosing peers from external ISPs leads to a better throughput than that of intra-ISP; (2) some ISPs might have different median throughput for different cities, which indicates that geographical information also plays an vital role in deciding the throughput between peers; (3) ISP pairs' throughput is highly variable and the variation increases with time, while throughput is more predictable for inter-ISP pairs with recent logs, e.g. last week. These facts motivate us to propose

[1]We sketch in Section IV-F how the actual deployment of advanced peer selection algorithms could be done and how typical pitfalls, e.g. over-fitting, could be mitigated.

learning-based peer selection methods. Using our dataset, we demonstrated the efficiency of machine learning models, which can significantly improve the P2P throughput. Our best (neural network based) model improves the peers' average throughput with their selected neighbors by 22.7%, 14.5%, and 6.8% compared to the random, same ISP, and geographical peer selection, respectively. Moreover, the gain is significantly higher for peers experiencing low bandwidth, which arguably are the most important group of peers to help: it reaches 89%, 20.4%, and 24.3%, respectively, for the 10% slowest peers with respect to the initial random strategy.

As a by-product of throughput optimization, we demonstrate that these machine learning models also reduce the inter-peer delay and the chunk loss rate, which is the fraction of chunks not fully received before the timeout. In the latter case, the peer reverts to the CDN server, which enables to maintain the perceived quality but at a higher operational cost. Our best neural-network approach reduces P2P messaging delay by 18.6% and chunk loss rate by 29.9%, while locality-aware (geo-model or intra-ISP) solutions achieve gains of 14.9% and 7.2%, respectively.

The remainder of the paper is organized as follows: In Section II, the properties of the WebRTC-based hybrid CDN-P2P live streaming system are presented. We then discuss the design of the experiments in Section III. The evaluation and the comparison of the models are discussed in Section IV. We review related works on peer selection in Section V. At last, Section VI concludes the work and future directions are given.

II. Properties of the P2P system

A. System overview

In this section, we provide a high-level overview of our CDN-P2P live streaming system.

Architecture: Our system instantiates a mesh-pull architecture, which is renowned for its advantages in terms of robustness, scalability, and ease of deployment over other architectures such as IP multicast or tree-based overlay [7]. In mesh-pull systems, the overlay is constructed with the help of a tracker. WebRTC uses STUN[2] server for NAT-traversal. We do not consider the peers that are behind a symmetric NAT, because they usually need a TURN (Traversal Using Relays around NAT) server which limits scalability since all the traffic between peers has to be relayed by the server. Metrics related to each chunk exchange between peers, such as chunk loss events and the end-to-end delay are consolidated by the tracker in a log database.

The tracker is a server with 4 vCPU and 16 GBytes of RAM. Based on our experience, this configuration is able to support, in production, 30 000 simultaneous users when deploying random peer selection queries and WebRTC signaling functionalities.

Interactions: there are four major actions performed by the system:

[2]https://tools.ietf.org/html/rfc5389

Step 1: Registration of the peers. Peers submit to the tracker their key information, including the content URI, the video quality level, the codec, and the IP address which is mapped to an ISP and coarse grain geographical information, i.e., city, country, and continent.
Step 2: Peer selection. The tracker selects neighbours for the requesting peer from the *potential member set*, the pool of registered peers watching the same stream.
Step 3: Buffer map exchange. It allows for informing direct neighbours which chunks a peer can serve.
Step 4: Chunk exchange. When requesting a video chunk, the peer checks the buffer maps sent by its neighbours. A request is then sent to a neighbour having the chunk and the chunk exchange begins and hopefully ends before the timer expires. If not, the peer queries the chunk from a CDN server.

The **active neighbour size** is set to 10 for every peer. This means that a peer can simultaneously connect to at most 10 other peers from the potential member set. Our choice of this parameter is empirical. From the experience gained from the daily operation of the system, a higher neighbour size incurs a higher P2P overhead impairing the QoE, while a lower neighbour size is not effective enough, as, because of peer churn, the peers have fewer effective P2P links than the configured active neighbour size.

Our focus in this work is on peer selection. Our goal is to help the tracker performing a better peer selection using a machine learning model trained with the data collected on the statistics reported by the peers. We are able to perform a fair comparison of different peer selection strategies using our dataset as the tracker applies an **unbiased peer selection**. Indeed, the tracker randomly selects peers from the potential member set and, as a consequence, we have statistics on a variety of peer connection types (inter-ISP, intra-ISP, inter-city, intra-city, etc.) that we would not have obtained if e.g. a locality-aware peer selection had been chosen. We are thus able to explore different selection policies in this work.

As a side note, we would like to emphasize that we had to change the default tracker policy to carry out this study, which was not purely random but ISP-based initially. Changing the peer selection policy of a production system over a long time brings a unique value to the collected dataset, as operators usually are reluctant to change such an important system parameter. The decisive argument was that the study might determine the best selection strategy and, thus, lead to future operational gains.

B. P2P quality of service

We formally define here the key metrics that we consider when evaluating the various peer selection strategies.
Throughput: it is a key metric in a video distribution system, as video traffic is by definition bandwidth hungry. It is calculated per P2P link, i.e., for each tuple (p_1, p_2). We have a throughput sample per chunk exchange and we extract the median value as the throughput of the link.
Chunk loss rate: A chunk is declared lost if it is not received in due time by the requesting peer. In the latter case,

TABLE I: Overview of the dataset

Properties	Full dataset	Filtered
# chunk exchanges	2 083 975 863	1 381 884 394
# viewers	18 048 259	11 975 019
# ISPs involved	5 955	3 627
# cities involved	23 458	21 060
# countries involved	193	157

the peer falls back to the CDN to avoid playback freezing. The chunk loss rate is calculated as $(N_{\text{uploaded chunks}} - N_{\text{downloaded chunks}})/N_{\text{uploaded chunks}}$ for a given link.

End-to-End (e2e) delay is used to measure the RTT between two peers. We measured the delay with a specific type of WebRTC message containing an 8-byte length sequence number. The round trip time is the time elapsed between sending the ping message and receiving the pong one from the remote peer. The QoS of a hybrid CDN-P2P live streaming system is very sensitive to the e2e delay. Moreover, it also has an important impact on the system P2P ratio. Indeed, within the P2P timeout (few seconds), a peer needs to be quickly informed of the chunk availability (with a small size buffer-map exchange) so as to request and to fully receive the chunk. A high delay would presumably decrease the P2P ratio.

C. Dataset

Our dataset aggregates 9 months of logs[3] (Sept 2019-June 2020) for 7 channels of a national IPTV package of a North African country. IP addresses of clients were anonymized as each peer was assigned an UUID as a global identifier. We further have for each peer its ISP, city, and country. Table I provides a high-level overview of the dataset. While the channels originate from North Africa, we observe that viewers were spread over 193 countries, 23 458 cities, and 5 955 ISPs. More precisely, around 50% of users originate from North Africa, about 40% from different countries in Europe, and 10% worldwide. If we focus on the large majority of users, 823 cities and 60 ISPs are aggregating 90% of the users. Overall, more than 2 billion chunks were exchanged between more than 18 million peers over the period.

For our experiments on peer selection, we filtered the dataset to only retain peers that exchanged with at least 5 other peers during a session. Indeed, our models perform a selection of the 20% best neighbors (see Section III-A), and doing so we are always able to select at least 1 neighbour. The filtered dataset represents 66.4% of the data and also includes very diverse users as shown in Table I. We believe the value of our dataset goes beyond the live streaming application under study. Indeed, it can be seen as a large sampling of uplinks and downlinks of users performing WebRTC transfers, an open-source protocol gaining momentum to build over the top video services.

D. Preliminary evaluation of classical peer selection policies

As the classical peer selection approaches are based either on the ISP or geographical information, we highlight in this section the wide diversity of situations faced when using one criterion (ISP) or the other (here the city).

Inter- vs Intra-ISP exchanges. For each ISP, we computed the ratio between the average throughput of inter- vs intra-ISP chunk exchanges during a given day. Fig. 1(a) reports the cumulative distribution of the ratio over the whole period for the 6 top (anonymized) ISPs that serve around 52% of viewers and an additional one, ISP A, chosen to illustrate an extreme case. Each sample corresponds to averages of inter and intra ISP throughput computed over 5-minute intervals. For some ISPs like C, the classical policy to choose peers inside its own ISP is the policy of choice. However, for some other ones, it is in general more efficient to connect to peers outside the origin ISP to obtain a better throughput, e.g. 90% of the time for ISP A. For most of the ISPs, the efficiency of the policy is varying over time. For G and E, it is better to go outside for 35% and 5% of the time, respectively. Moreover, the advantage to connect outside of its own ISP may be important: a twice as large bandwidth is obtained 20% and 5% of the time for peers in ISPs A and G, respectively. An efficient policy with respect to ISP should thus be adapted over time if possible.

In fact, more efficient policies than inside/outside can be devised. We divide the inter-ISP traffic according to the target ISP to analyze it in more detail. We report the average ratios over the whole period for pairs of ISPs in the heatmap of Fig. 1(a). First, this confirms that ISP A should choose peers outside (line almost entirely red), but in priority in ISP C when possible. Second, by focusing on the red tiles, there are 4 out of 7 top ISPs in which peers should choose neighbours from external ISPs whenever possible. For instance, A should choose peers from B, C, D, E, and G. Similarly, F should choose peers from A, etc. We also want to point out the global asymmetry of the matrix. As an example, ISP D receives chunks from peers in C at the same bandwidth as the one from its fellow peers in ISP C, when ISP C receives at speed ten times lower from ISP D.

To sum up, (i) the fact that the bandwidth of transfers for a peer in a given ISP is often larger when exchanging with peers in another ISP than when exchanging within its own ISP, as well as (ii) the observation that the best ISP to connect with may change over time, hint that the classical strategy for a peer to choose neighbors of its own ISP is far from an optimal policy.

Intra- vs inter-city exchanges. We now evaluate the second most common peer selection policy: choosing peers which are geographically close. To this end, we consider the traffic within one ISP (to study comparable infrastructures) and we compute the ratio between the inter- vs intra-city chunk transfer throughput. We report the distribution of the ratio for a typical ISP (A, B, C, D, E) in Fig. 1b. Results for other ISPs are similar. As we see, even in the same ISP, it is not necessarily better to choose peers from the same city. It is better for peers in city I to select peers outside the city 40% of the time. The tracker might choose a peer, according to the heat map, from city L if available to obtain an improvement of

[3]See https://github.com/j-maz/p2p-live-streaming-dataset.git

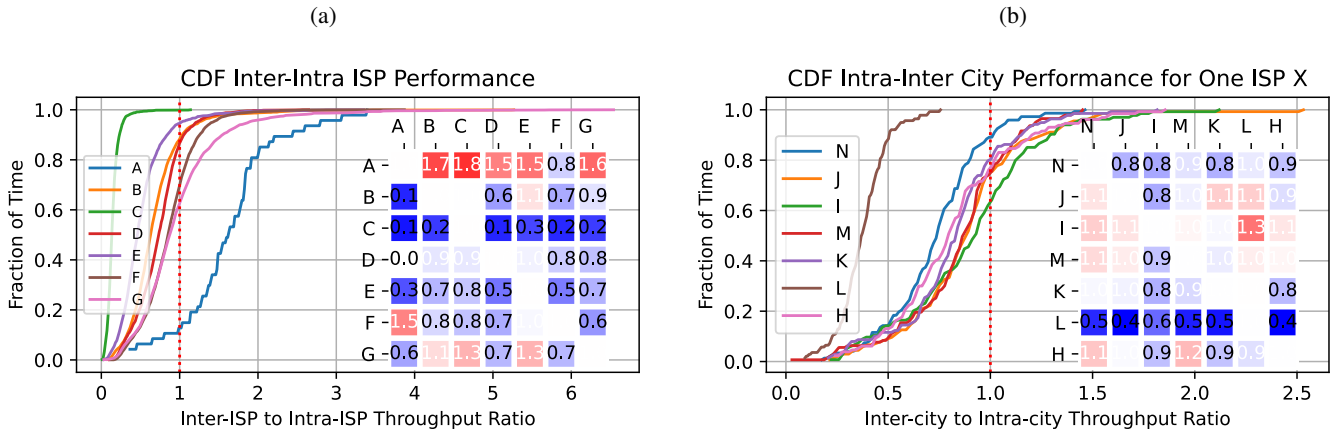

Fig. 1: Cumulative distributions of the ratios between (a) Inter-ISP vs Intra-ISP (b) Inter-City vs Intra-City chunk exchange throughput over the whole period of observation (one value per day) for one anonymized ISP. The matrices represent the average ratios for inter-ISP (a) and inter-city (b) exchanges for the considered ISPs and cities. A ratio greater than 1 means that it is more efficient for a peer to connect outside its ISP or City.

30% over the intra-city policy. In fact, for 4 cities over 7 in the study, it is better to systemically select peers from a different city over the whole period of time: from K, L, N and M for J, I, M and H respectively. We show in the following that in fact the peer selection can even be much better by using more complex evolutive policies.

In summary, the classical policies based on ISP or geographical information could enable a better peer selection, provided that they are based on a model that is able to learn the complex inter dependencies that exist between pairs of ISPs and pairs of cities. In addition, the best pairs of ISPs or cities vary over time. This is why we consider prediction models with varying amount of memory in our work.

III. Prediction Models

In this section, we first formalize the peer selection problem. We then describe different models to solve it in Section III-B and we compare their performances in Section IV.

A. Methodology

Problem: We formulate the *peer selection problem* as a *regression problem* in which the predicted outcome variable, to be used as the ranking criterion, is the throughput of a connection between two peers i and j at time t. The prediction is based on the following set of features: hour of the day and, for each peer, its ISP, city, country, and continent. The features are readily available when a peer registers to the tracker.

The predicted throughput is then used to rank peers, as the tracker of a P2P system should continuously select the best peers in the neighborhood. For a peer session s, we denote p_s the peer requesting chunks and $\mathbb{A}_s$ the set of peers (called *actual member set*) from which p_s has actually received chunks during the session. Note that, as we evaluate the algorithms on our dataset, the *prediction models can only be evaluated on the exchanges that actually occurred*. However, since the tracker implemented a purely random peer selection strategy, the $\mathbb{A}_s$ set is a *uniform random sample* of the actual neighborhoods of p_s during the session. For this reason, the results obtained from the experiment are representative of what would be observed if the strategy was actually implemented.

The machine learning based model first ranks all the peers in $\mathbb{A}_s$ according to the predicted throughput. As all the peers in $\mathbb{A}_s$ are not present and available during the whole session s due to churn and transfers with other peers, it would not be fair to always select the best estimated peer. To emulate this time effect, we considered that the *best available peer* when a chunk is requested is an *average over the best peers*. In the following, we consider the 20% best peers, but we tested other values (e.g. 10% or 30%) and got similar results for the model comparison. Formally, for a model m and a session s, the set of estimated (20%) best peers is denoted by $\mathbb{S}_s^m$.

The evaluation consists in comparing for different metrics the real exchanges within a peer session (with peers selected at random) with the ones of the best estimated peers in $\mathbb{S}_s^m$ (corresponding to the ones that would be chosen by a model) for different models m and over all peer sessions s of our dataset. Note that, for this final goal, the actual value of the prediction matters less as long as the model correctly ranks peers, i.e., is able to identify good and bad peers from a throughput perspective. We kept regression models even in this case in the training phase, as they had better performances than classification models trying to predict the best peers.

B. Studied Models

We considered the following peer selection strategies:

Random peer selection (random): this method randomly selects peers from the list of peers requiring the same video chunks. This model is used as the baseline for the evaluations.

ISP-based peer selection (isp): this method matches up peers that are in the same ISP. If there is no such peer available, the tracker performs a random peer selection.

Geo-based peer selection (geo): this method selects peers in the same city. If there are not enough peers in the same city, it falls back to select peers in the same country or alternatively

in the same continent. If there is no such peer available, the tracker performs a random peer selection.

Continuously-trained Decision tree based peer selection (con-dt-n): this model is trained with the previous n days of log and predicts the throughput for the next day using a decision tree. We report the results for $n = 1, 5, 7$.

Continuously-trained Linear regression based peer selection (con-lr-n): this models relies on linear regression trained with the previous n days of log and predicts the throughput for the next day. Results are given for $n = 1, 5, 7$.

Continuously-trained Neural Network (NN) based peer selection (con-nn-n): one specific NN model that is constructed according to the previous n days of log and predicts for the next day. We compared models with $n = 1, 5, 7$.

One-shot trained NN based peer selection (os-nn): one specific NN model that is trained only once and is evaluated for the remaining days. This enables to assess how fast the knowledge of such models decays over time.

Pandas [8] is the primary data processing library for data preparation. Scikit-learn [9] is used for *con-dt* and *con-lr* model training. The *os-nn* and *con-nn* ones are trained with TensorFlow [10].

Hyper-parameter tuning: in order to find the optimal hyper-parameters for different models without over-fitting the models, we took only 5 days of data for training and tuned with 1 extra day. We tuned the decision tree model to use *max_depth=20*, the neural network models to use *batch_size=2 180*, *epochs=47*, *learning_rate=0.04826*, *dropout_prob=0.1247*, *hidden_layers=7*, *hidden_layers_neurons=126*, and we varied the days for training, i.e., $n \in \{1, 5, 7\}$. This parameter is being appended to the model name, e.g. *con-nn-5* is the continuously-trained neural network model that is trained with last 5 days of data, etc.

Structure of the neural networks. We use a typical fully-connected neural network.The input (resp. hidden) layers use sigmoid (resp. ReLu) as activation functions as they generally lead to fast training [11]. Dropout layers [12] are inserted to the 6 hidden layers (with 126 neurons) to prevent over-fitting. The output value, the predicted throughput, is generated through the last layer with linear activation function.

C. Metrics

Let $\mathbb{M} = \{m|$ random, isp, geo, con-dt-*, con-lr-*, con-nn-*, os-nn$\}$ be the set of models. We evaluate model m over all peer sessions of the dataset. For the peer session s, the model computes $\mathbb{S}_s^m$, i.e., the estimated 20% best peers of the actual member set $\mathbb{A}_s$, see Section III-A. We then base our evaluation on the following metrics:

The **average throughput**, denoted as $AvgTP_{p_s}^m = \frac{1}{|\mathbb{S}_s^m|} \sum_{i \in \mathbb{S}_s^m} TP_{p_s,i}$ where $TP_{p_s,i}$ is the real throughput of peer p_s downloading from peer $i \in \mathbb{S}_s^m$.

The **average percentile rank**, denoted as $AvgPR_{p_s}^m = \frac{1}{|\mathbb{S}_s^m|} \sum_{i \in \mathbb{S}_s^m} PR_{p_s,i}$ where $PR_{p_s,i}$ is the real percentile rank of the throughput of peer p_s downloading from peer $i \in \mathbb{S}_s^m$.

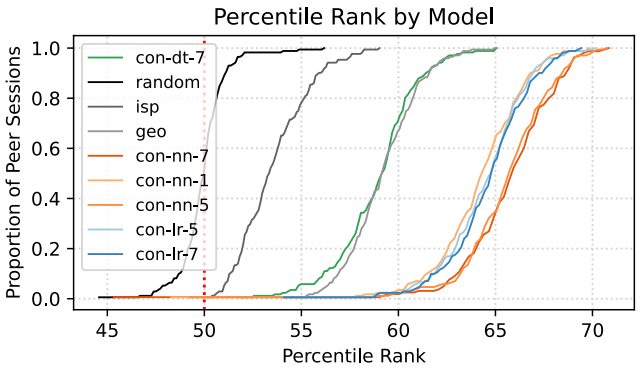

Fig. 2: Avg. percentile rank AvgPR pf all peer sessions in the "best" neighbor sets estimated by the prediction models.

The **average chunk loss rate**, denoted as $AvgCLR_{p_s}^m = \frac{1}{|\mathbb{S}_s^m|} \sum_{i \in \mathbb{S}_s^m} CLR_{p_s,i}$ where $CLR_{p_s,i}$ is the ratio of the number of chunk losses over the total number of chunks sent from selected peer i to the peer p_s.

The **average end-to-end delay**, denoted as $AvgEED_{p_s}^m = \frac{1}{|\mathbb{S}_s^m|} \sum_{i \in \mathbb{S}_s^m} EED_{p_s,i}$ where $EED_{p_s,i}$ is the round trip time for the ping/pong message from peer p_s to peer i.

IV. Evaluation Results

A. Overall models performance

We first evaluated the ability of the model to correctly rank the neighbours for a given peer. In other words, we evaluate their ability to select the neighbours with highest rank in reality. Fig. 2 presents the CDFs of the average rank AvgPR of the peers in $\mathbb{S}_s^m$ over all peer sessions s for the different models $m \in \mathbb{M}$. As expected, Random peer selection produces AvgPR values around the 50th percentile. All the other methods improve the efficiency of the system compared to the random one by selecting the neighbours of peers that are in higher percentile rank.

The classical strategies, namely ISP-based and geo-based models, only slightly improve the result over the random method. In contrast, the linear regression model already leads to significant improvements over those classical methods. The best performance is obtained with the neural network model trained over one week of data.

Next, we computed and reported in Table II the key metrics (throughput, e2e delay and chunk loss rate) for each model. We indicate each time the (i) average value and (ii) average performance gain with respect to the random model. As we know the ground truth, i.e., the 20% best neighbors of each peer, we also report it with the name *ideal*. Note that reaching the performances of ideal would suppose a complete knowledge of the network conditions and of the peer characteristics, allowing to always select the best available peers. It is not attainable, but is given as an upper bound of the gain.

We confirm that con-nn-7 is the model reaching the highest bandwidth, 3 782.60 Kbit/s (corresponding to an increase of 22.7% over random model). We also observe that while the

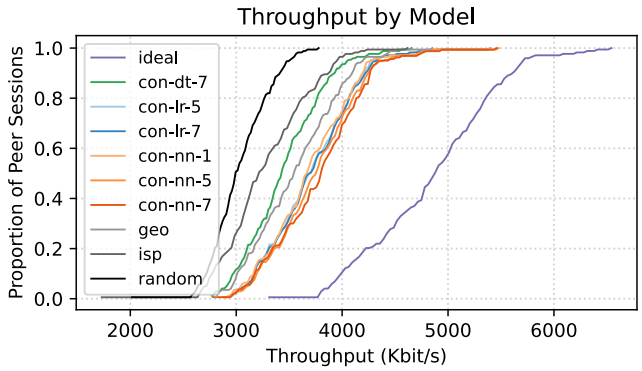

Fig. 3: Throughput distributions as obtained for each model.

TABLE II: Performance gains over the random peer selection for each model and for each considered metric.

	avgPR		avgTP		avgEED		avgCLR	
model_name	value	ratio	value	ratio	value	ratio	value	ratio
ideal	3.92	1.962	4 828.74	1.567	390.06	0.750	0.08	0.479
random	1.99	1.000	3 081.97	1.000	520.38	1.000	0.17	1.000
isp	2.15	1.076	3 305.31	1.072	518.18	0.996	0.16	0.963
geo	2.35	1.177	3 540.32	1.149	504.24	0.969	0.15	0.889
con-dt-1	2.33	1.167	3 439.25	1.116	464.40	0.892	0.14	0.859
con-dt-5	2.34	1.172	3 452.63	1.120	465.72	0.895	0.14	0.856
con-dt-7	2.33	1.169	3 447.99	1.119	468.11	0.900	0.14	0.857
con-lr-1	2.55	1.278	3 679.47	1.194	435.57	0.837	0.13	0.784
con-lr-5	2.57	1.285	3 700.55	1.201	429.85	0.826	0.13	0.779
con-lr-7	2.57	1.287	3 705.38	1.202	428.77	0.824	0.13	0.779
con-nn-1	2.56	1.284	3 691.71	1.198	434.38	0.835	0.13	0.781
con-nn-5	**2.61**	**1.305**	3 739.48	1.213	**423.50**	**0.814**	0.13	0.765
con-nn-7	2.60	1.303	**3 782.60**	**1.227**	428.52	0.823	**0.12**	**0.701**

models seek to optimize the throughput, this translates also in a gain in terms of e2e delay and chunk loss rate. This is not entirely surprising as the TCP throughput is known to be correlated with the RTT of a connection [13]. Also, the higher the throughput, the more likely a peer is to receive the requested chunk before the timeout expires. The best performance, with respect to these two metrics is con-nn-5 with **18.6% smaller end-to-end delay** and con-nn-7 with **29.9% smaller chunk loss rate**. The good performance directly translates into higher efficiency of the hybrid system.

B. Performance gain breakdown

We next turn our attention to the performance of individual peers, as the results in the previous section are aggregated results, and, thus, only provide a global tendency.

Fig. 3 reports the distribution of throughput for each model. The key observation here is that we have no overlap between the curves, which suggests that the improvements, from one model to the other, are for every peer.

To further investigate this question, we computed the gain with respect to the initial average throughput (over all its exchanges with other peers) of a peer. We grouped peers in categories (deciles) with respect to their initial throughput given by the random model. Intuitively, one expects the gain to decrease with increasing peer initial throughput, as the higher the initial throughput, the more difficult it is to discover better neighbors. This is indeed what we observe in Fig. 4, where we see that the throughput gain can reach 89% for the con-nn-7 model for the 10% peers with the lowest initial throughput. This is an important key performance indicator for the content provider as this means that the models will boost the performance of all peers (or at least not decrease it for the peers with highest bandwidth), especially the ones that had initially the worst performance, arguably the ones needing the most an improvement.

C. Time evolution of neural network models

As con-nn-7 offered the best performance, either globally or on a peer basis, we decided to further focus on the evolution of its performance over time. Fig. 5 reports the performance over time of the con-nn-7, os-nn, and ideal model. Recall that the os-nn model is trained only once, when con-nn-7 is continuously retrained with the last 7 days of data. In the plot, each point is the average of the AvgPR values of the session for each day. We further report the absolute difference between os-nn and con-nn-7. We observe that the accuracy of os-nn gradually decays over time as compared to con-nn-7. The continuous training of con-nn-7 enables to maintain consistent performance over time, i.e., the distance to ideal remains roughly the same over time. This is in line with Section II-D, where we observed that a peer selection strategy based on ISPs or cities needs to account for time variations.

D. Impact of the choice of objective metric

The best results were obtained with the con-nn-7 model in the previous sections. We used throughput as the objective metric and observed the impact of optimizing this metric on other key metrics such as chunk loss rate (avgCLR) and end-to-end delay (avgEED). One could argue that chunk loss rate is the metric to optimize instead of throughput, as loosing a chunk is detrimental both to the overall efficiency of the content delivery system and to the individual resources of the peer. Note, however, that the two metrics are a priori correlated as aiming for a high throughput should reduce the chances that a chunk is not fully received before the deadline and, thus, considered as lost.

To investigate this scenario, we took the best model con-nn-7 and trained a copy of it on loss rate. We denote this model as M_{lr}. We let the model pick up peers and we log the throughput of the constructed neighbourhood. The result of this experiment are highly correlated with the one trained with throughput, i.e. M_{tp}. Indeed, we observed a Pearson coefficient of correlation of 0.8 for the throughput values per peer chosen by each model. The scatterplot of those values can be seen in Fig. 6, in which we report the pairs (x, y), where x and y are the throughputs of neighbourhood constructed with M_{tp} and M_{lr}, respectively.

E. Impact of peer selection on node degree

The various strategies we studied are able to improve the quality of the neighborhood of a peer. This benefit could

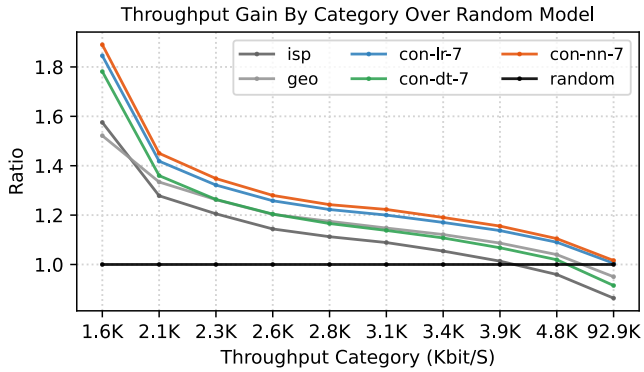

Fig. 4: Achieved throughput gains over random model as a function of the initial peer throughput.

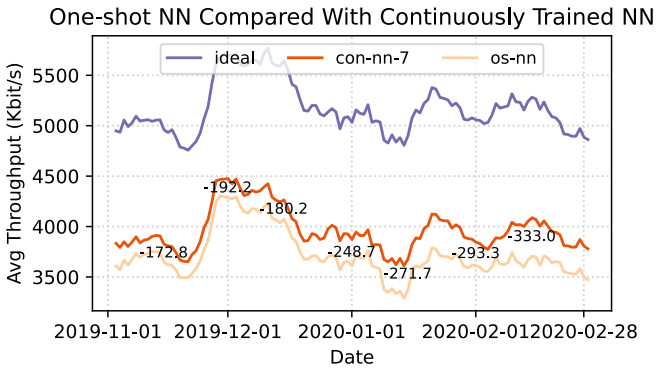

Fig. 5: The time evolution of the average throughput of con-nn-7 and os-nn with their absolute differences. The performance of os-nn (trained only once) decreases over time.

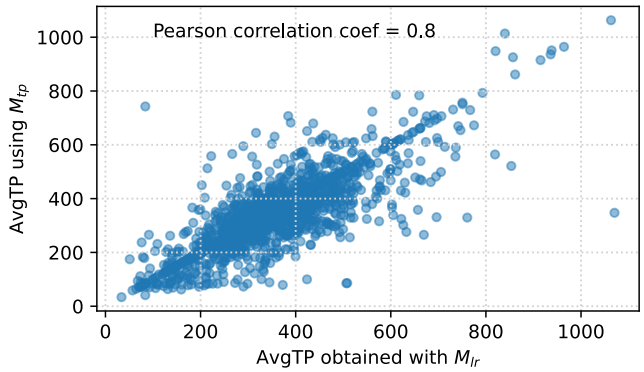

Fig. 6: Comparison of the performances obtained using neural networks trained on lost chunks versus on throughput. We observe a strong correlation.

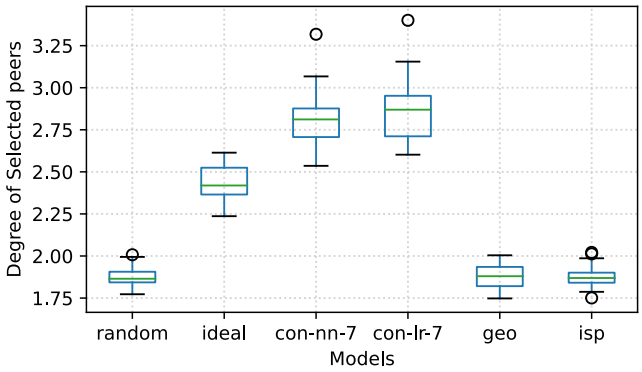

Fig. 7: Degree distribution of peers selected as neighbors for each model.

however be deemed artificial if the rewiring of connections proposed by a strategy leads to overload the peers providing the best bandwidth. Indeed, as a peer is servicing without any preference all the peers requesting chunks, an overload scenario would be similar to a server operated under the Processor Sharing policy that services too many clients: the service capacity per client drastically decreases as the number of clients increases. Note, however, that a safeguard mechanism exists that should prevent reaching such an extreme scenario in our system as a peer cannot serve more than 3 peers simultaneously in our protocol.

To assess the impact of the various peer selection strategies on the node degrees, we sampled 20 days at random. For each day, we calculated the number of times each peer was selected as neighbor. We call it its degree. We report the distribution of selected peer degrees for each model in Fig. 7.

We observe that the random peer selection and geo (isp) model have lowest degrees, with in general fewer than 2 peers connecting to any selected peers. In contrast, the con-lr-7 and the con-nn-7 have higher degrees. However, the node degree increase is quite limited (by only about one new connection). We can thus conclude that the benefit provided by the advanced peer selection strategies we studied is not outweighed by the overload of some peers.

F. Towards a real-world deployment of advanced peer selection algorithms

The results gathered from the offline analysis of our dataset suggest that a neural network-based approach can improve the overall performance of a hybrid CDN-P2P live system. The next step is to deploy the peer selection strategy in the live system. We discuss hereafter essential points to address during actual deployment.

First, we have to consider where to deploy the peer selection algorithm. The tracker acting as a rendezvous point for all peers is the natural target for the deployment. The load at the tracker, e.g. in case of flash crowds will have to be anticipated.

Second, we want to evaluate the models in production continuously. We compare different policies (e.g., isp and con-nn-7) or variants of the same policy, such as con-nn-7 with additional features or different hyper-parameters. We thus intend to rely on a classical A/B testing approach where the set of clients will be dynamically split into two groups, where each set benefits from a different policy. We will have to account for the load variation for channels with a high time-varying load, e.g., channels broadcasting popular sports events.

Third, we want to avoid the over-fitting problem that might appear here when we retrain the model with samples imposed

by the policy under study. It is vital to find the right balance between exploiting the current dataset that partly results from the previous decisions of the algorithm and exploring new options. It is akin to the Monte-Carlo method applied in Reinforcement Learning. The agent should perform a random sampling of the action space to improve the overall expected return to converge to the best action. BitTorrent relies on the notion of random peer to sample new possible neighbors [14]. In a first stage, we will consider two non-mutually exclusive solutions to introduce randomness in the peer selection process: the tracker can 1) sample the potential neighbors set randomly before evaluating the pairs, or 2) randomize the peer selection with a certain probability which would be a system parameter (e.g. 20%).

V. Related work

While a host of works exists on P2P streaming, see e.g. [15] for a survey, there exists fewer studies on CDN-P2P systems. In [16] a theoretical framework based on fluid modeling is presented for such systems and demonstrates that the associated P2P network enables to serve more users with a higher quality. A decade ago, ChinaCache performed a large scale measurement study of their CDN-P2P live streaming system [17]. While both CDNs and peers are organized in trees, little detail is given on the exact procedure for a peer to select its neighbors and to create those P2P trees.

The P2P overlay construction can be mainly categorized into tree-based [18]–[21], mesh-based [22], and hybrid methods [23]. The mesh-based overlay construction is being used by most commercial products [22] (including ours) for its simplicity and robustness against peer churn. In order to form a mesh network, [24] and [25] developed random peer selection strategy. [26] and [27] incorporated contribution score and location of peers, as well as the age of a peer in order to distinguish the stable ones. In most recent works, [28] introduced AStream which incorporated location and upload capacity information into the overlay construction consideration, outperforming the previous methods. Similarly, [29] considers geographical location, upload bandwidth, the age, and the utilization of a peer to prioritize the peers in the overlay construction phase with the help of a fuzzy logic system. The authors in [30] propose a two stage process to build neighborhoods: first, the tracker groups the peers based on their upload capacity and, then, peers organize themselves based on more precise measurements such as end-to-end delay. This approach focuses only on each peer upload capacity, which is not easy to obtain, especially for mobile devices connected to 3G/4G networks, while we leverage pair exchange statistics, which are readily available.

In the above stream of work, overlay construction and peer selection can be mixed. Some works have focused specifically on peer selection. In [31], the authors propose a two-step peer selection strategy which leverages an ISP's Oracle service and a gossip method for peer selection to find a balance between locality and QoS. The Oracle is used to rank the peers at the tracker. This requires a deep collaboration with ISPs, which is not readily available for most of the live streaming providers. Other works focused on discouraging free-riders [32], [33], maximizing utilization of peers with high upload-bandwidth [30], [34], [35], or anticipating the future resource demands in VoD context [36].

A number of works advocated the importance of limiting inter ISP traffic, either for pure P2P [6], [31] or hybrid CDN-P2P networks [37], [38]. As we observed with our measurements, favoring intra ISP links at the expense of inter ISP fails to capture the reality of bandwidth distribution in the wild. We can also argue that limiting inter-ISP traffic is less a of concern nowadays as the Internet is becoming flatter with more direct peering links between ISP thanks to IXPs [39].

VI. Conclusion

In this paper, we followed an empirical approach to assess the extent to which the peer selection strategy of a hybrid CDN-P2P live video streaming system could be improved.

We first outlined that the legacy policies that group peers based on their ISP or geographical relation fail to capture the complex relations that exist between (and also inside) ISPs or cities. This justifies the need for machine learning models to capture these complex relations as well as accounting for possible time effects that can modify those relations.

Out of the various machine learning approaches we considered, the best performance were achieved with a neural network prediction model with a memory horizon of one week. On our dataset, we demonstrated that it can increase by more than 22.7% (resp. 6.8% and 14.5%) the throughput over the random (resp. geo and ISP) peer selection method for an average peer. Moreover, the gain reaches 89% for the 10% peers initially featuring the smallest download throughput, which are the peers which would benefit the most from an improvement of their chunk exchanges. Other key points of our best neural network model are (i) its temporal stability as its distance to the ideal model remains roughly constant over time and (ii) that it does not overload the peers featuring the best throughput. Improving the throughput has for (positive) side effect to decrease by 18.6% the e2e delays between peers —a good point for the transfer of control messages between the peers— and also the chunk loss rate by 29.9%. The latter enables to further maximise the benefit of the P2P strategy and reduce the cases where the peers have to revert to the CDN to download a video chunk.

The methods proposed can be easily adapted to every WebRTC-based hybrid CDN-P2P live streaming system to have an instant improvement to the P2P performance, as they entail only modifications at the central tracker.

In future work, we will focus on studying the behavior of the models in the live system. We will pay attention to how these policies influence the fairness of the P2P network. Another path would be to propose more advanced machine learning models, e.g., reinforcement learning as the tracker could be modeled as an intelligent agent performing according to its environment.

REFERENCES

[1] "Cisco visual networking index: Forecast and trends, 2017–2022," p. 38, 2018.

[2] W3C, "Webrtc 1.0: Real-time communication between browser," 2020. [Online]. Available: https://www.w3.org/TR/webrtc/

[3] (2020) Akamai. [Online]. Available: https://www.akamai.com/fr/fr/

[4] (2020) EasyBroadcast. [Online]. Available: https://www.easybroadcast.fr/

[5] (2020) Cdnetworks. [Online]. Available: https://www.cdnetworks.com/

[6] X. Haiyong *et al.*, "P4p: Provider portal for applications," *ACM SIGCOMM Computer Communication Review*, 2008.

[7] M. Zhang *et al.*, "Optimizing the throughput of data-driven peer-to-peer streaming," *IEEE Transactions on Parallel and Distributed Systems*, vol. 20, no. 1, pp. 97–110, 2009.

[8] W. McKinney *et al.*, "Data structures for statistical computing in python," in *Proceedings of the 9th Python in Science Conference*, vol. 445. Austin, TX, 2010, pp. 51–56.

[9] F. Pedregosa *et al.*, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[10] M. Abadi, *et al.*, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/

[11] Y. LeCun *et al.*, "Deep learning," vol. 521, no. 7553, pp. 436–444, 2015.

[12] N. Srivastava *et al.*, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, 2014.

[13] Q. He *et al.*, "Prediction of TCP throughput: formula-based and history-based methods," in *SIGMETRICS*. ACM, 2005, pp. 388–389.

[14] B. Cohen, "Incentives build robustness in bittorrent," in *Workshop on Economics of Peer-to-Peer systems*, 2003.

[15] U. Ihsan *et al.*, "A survey and synthesis of user behavior measurements in P2P streaming systems," *IEEE Commun. Surv. Tutorials*, 2012.

[16] M. Ahmed *et al.*, "Analysis of adaptive streaming for hybrid CDN/P2P live video systems," in *IEEE ICNP*, 2011, pp. 276–285.

[17] Y. Hao *et al.*, "Design and deployment of a hybrid CDN-P2P system for live video streaming: experiences with livesky," in *International Conference on Multimedia*. ACM, 2009.

[18] D. A. Tran, K. A. Hua, and T. Do, "Zigzag: an efficient peer-to-peer scheme for media streaming," in *IEEE INFOCOM 2003*, 2003.

[19] X. Jin *et al.*, "Cloud assisted p2p media streaming for bandwidth constrained mobile subscribers," in *2010 IEEE 16th International Conference on Parallel and Distributed Systems*, 2010, pp. 800–805.

[20] M. Castro *et al.*, "Splitstream: High-bandwidth multicast in cooperative environments," *ACM SIGOPS Oper. Syst. Rev.*, 2003.

[21] V. N. Padmanabhan *et al.*, "Resilient peer-to-peer streaming," in *IEEE ICNP*, 2003.

[22] G. Gao *et al.*, "Collaborative caching in p2p streaming networks," vol. 27, no. 3, pp. 815–836.

[23] M. Sina *et al.*, "Car-plive: Cloud-assisted reinforcement learning based p2p live video streaming: a hybrid approach," *Multimedia Tools and Applications*, vol. 78, no. 23, pp. 34 095–34 127, 2019.

[24] Xinyan Zhang *et al.*, "Coolstreaming/donet: a data-driven overlay network for peer-to-peer live media streaming," in *IEEE INFOCOM*, 2005.

[25] N. Magharei *et al.*, "Prime: Peer-to-peer receiver-driven mesh-based streaming," *IEEE/ACM Transactions on Networking*, 2009.

[26] H. Liu *et al.*, "Metree: A contribution and locality-aware p2p live streaming architecture," in *24th IEEE International Conference on Advanced Information Networking and Applications*, 2010.

[27] C. Hammami *et al.*, "Hybrid live p2p streaming protocol," *Elsevier Procedia Computer Science*, 2014.

[28] B. U. Maheswari *et al.*, "An improved delay-resistant and reliable hybrid overlay for peer-to-peer video streaming in wired and wireless networks," *IEEE Access*, vol. 6, pp. 56 539–56 550, 2018.

[29] K. Pal *et al.*, "Flhyo: fuzzy logic based hybrid overlay for p2p live video streaming," *Springer Multimedia Tools and Applications*, 2019.

[30] S. Budhkar *et al.*, "Two-tier peer selection strategy to minimize delay in p2p live streaming systems," in *NCC*, 2016.

[31] Z. Shen *et al.*, "ISP-friendly peer selection in p2p networks," in *ACM international conference on Multimedia - MM '09*, 2009.

[32] C.-W. Lo *et al.*, "Contribution-guided peer selection for reliable peer-to-peer video streaming over mesh networks," vol. 22, pp. 1388–1401, 2012.

[33] A. Habib *et al.*, "Service differentiated peer selection: an incentive mechanism for peer-to-peer media streaming," vol. 8, no. 3, pp. 610–621, 2006, conference Name: IEEE Transactions on Multimedia.

[34] N. Madeshan *et al.*, "An efficient super-peer selection for peer-to-peer live streaming networks over video-on demand service," vol. 13, no. 7, pp. 4606–4613, 2016.

[35] E. Kim *et al.*, "Efficient neighbor selection through connection switching for p2p live streaming," vol. 10, no. 4, pp. 1413–1423, 2019.

[36] T. Rohmer *et al.*, "Priori knowledge guided approach for optimal peer selection in p2p VoD systems," vol. 11, no. 3, pp. 350–362.

[37] Z. Jian *et al.*, "Locality-aware streaming in hybrid p2p-cloud CDN systems," *Peer Peer Netw. Appl.*, vol. 8, no. 2, pp. 320–335, 2015.

[38] M. Rodrigues *et al.*, "On traffic locality and qoe in hybrid CDN-P2P networks," in *ACM Annual Simulation Symposium (ANSS)*, 2011.

[39] B. Ager, N. Chatzis, A. Feldmann, N. Sarrar, S. Uhlig, and W. Willinger, "Anatomy of a large european ixp," in *Proceedings of the ACM SIGCOMM 2012 conference on Applications, technologies, architectures, and protocols for computer communication*, 2012.