Predicting the Session of a P2P IPTV User through Support Vector Regression (SVR)

Muhammad Ali
Department of Computer Science & IT
University of Balochistan
Quetta, Pakistan
mali.mscs@uob.edu.pk

Ihsan Ullah
Department of Computer Science & IT
University of Balochistan
Quetta, Pakistan
ihsanullah@um.uob.edu.pk

Waheed Noor
Department of Computer Science & IT
University of Balochistan
Quetta, Pakistan
waheed.noor@um.uob.edu.pk

Ahthasham Sajid
Department of Computer Science
BUITEMS
Quetta, Pakistan
ahthasham.sajid@buitms.edu.pk

Abdul Basit
Department of Computer Science & IT
University of Balochistan
Quetta, Pakistan
drabasit@uob.edu.pk

Junaid Baber
Department of Computer Science & IT
University of Balochistan
Quetta, Pakistan
junaidbaber@ieee.org

Abstract—**Scalability and ease of implementation make Peer-to-Peer (P2P) infrastructure an attractive option for live video streaming. Peer end-users or peers in these networks have extremely complex features and exhibit unpredictable behavior, i.e. any peer may join or exit the network without prior notice. Peers' dynamics is considered one of the key problems impacting the Quality of Service (QoS) of the P2P based IPTV services. Since, peer dynamics results in video disruption to consumer peers, for smooth video distribution, stable peer identification and selection is essential. Many research works have been conducted on stable peer identification using classical statistical methods. In this paper, a model based on machine learning is proposed in order to predict the length of a user session on entering the network. This prediction can be utilized in topology management such as offloading the departing peer before its exit. Consequently, this will help peers to select stable provider peers, which are the ones with longer session duration. Furthermore, it will also enable service providers to identify stable peers in a live video streaming network. Results indicate that the SVR based model performance is superior to an existing Bayesian network model.**

Keywords-P2P IPTV; user behavior; machine learning; SVR; Bayesian network; session prediction

I. INTRODUCTION

The advances in Internet speed and bandwidth over the past decade and the growing acceptance of peer-to-peer (P2P) applications have changed the culture of networking. The idea of P2P has opened a new era for the future development of human society, and particularly the digital video industry, by creating platforms for people to cooperate in exchanging data such as files, pictures, videos and music. Video streaming over the Internet is challenging since initially the Internet was not designed for such services. Video streaming can involve a very large number of users and it requires huge amounts of bandwidth as compared to other services. Live video streaming adds to the challenges as it imposes stringent playback deadlines and uses small buffers. To respond to the need of enabling live streaming over the Internet, a number of solutions have emerged. These are client/server, IP multicast, CDN and P2P based architectures. In contrast to other architectures, P2P solution, also called P2P IPTV, is potentially self-scalable and can be easily deployed with low cost over the existing infrastructure due to its implementation at application layer. In P2P approach, end-hosts become peers and they are placed in an overlay, virtual, self-organized network. Hosts in a P2P network receive stream and relay it to the other peers in the overlay, thus joining peers not only consume but also share resources. Nonetheless, implementing real-time video sharing applications through P2P networks poses many challenges due to the heterogeneity of the end-points and access networks, peer dynamics, and other IP network-inherited issues. Peer dynamics in a P2P network, when peers join or leave the overlay network without warning, is considered a major problem. When a user leaves the network the availability of the stream to all its dependents peers is disrupted. The dependent peers need some time to reconnect to other stream provider peers to receive the stream. Similarly, when a user joins a P2P system, the selection of a stable peer is needed. The majority of the existing approaches to find that stable peer are based on classical statistical methods which generalize the problem to a greater extent making it inappropriate for such a scenario which involves users. As user behavior is a complex issue, changing from one user to another, such models are not very efficient to observe the stability of an individual peer controlled by a user [1, 2]. A few latter works such as [3], attempt to model user behavior through Bayesian Network, which also considers contextual information. We chose the same approach and aimed at reducing the prediction error.

In this paper, an SVR based machine learning model is proposed to predict the session duration of a newly joined peer based on his past data. Such an outcome can be used in a

Corresponding author: Ihsan Ullah

number of ways in topology management of P2P IPTV systems for improved quality of streaming. For example, identification of a stable peer can be helpful to design and maintain stable topology resulting in better delivery of video stream. We also compare this model on the same data set with an existing Bayesian network model.

II. RELATED WORK

User behavior is playing a vital role in the performance of P2P systems. In order to improve the performance of P2P live video streaming systems, many researchers have modeled different aspects of user behavior. These works can be broadly divided into two types, namely the classical statistics-based methods and machine learning based methods.

Users have daily patterns in arrivals and session length in a music streaming service. Therefore, the user's first session can predict the following sessions and downtime in the service [4]. Similarly, the selection of stable peers enhances the streaming quality of peers and reduces playback interruptions [5]. To study user dynamics in live video streaming services, authors in [6] analyze how user join and leave the streaming service. They divide the user behavior based on the elapsed time and analyze the impact of stream disruption on stability. For this they intentionally put a disruption into a streaming session for 200 seconds and observe the number of remaining users in the session during this period. They notice that some of the users wait until 80 seconds, while others quit the session after 30 seconds. This clearly shows the importance of streaming quality for peer stability. To identify stable peers in the network, authors in [1] use the statistical correlation-based method which reveals that the time spent in the current session is positively correlated to the remaining time in the same session. They also propose a network topology where these so-called stable peers are placed near the root node. Another approach [2] that creates a stable backbone and then allows unstable peers to connect to the backbone uses the same principle for the identification of stable peers, which is a core component of the whole approach since the reliability of the system depends upon the stability estimation of the peer. However, stability of peers comes from user behavior and its estimation only from peer's age needs improvement since user behavior involves several metrics such as time of the day and day of the week [22]. A major contribution in this area appears in [7], in which survival analysis techniques are used to find the impacting factors of stability. They conjecture that stability is impacted by popularity, daytime and streaming quality. This work paves the way to consider models that involve more variables instead of the sole consideration of elapsed time. Towards this, authors in [8] first compare the observations of numerous measurements performed over the behavior in live video streaming systems. They extract user behavior metrics, which are studied commonly. They also extract the relationships of these behaviors with the external environment and network performance variables. All this information is collected in the form of a causal graph which can be used in user behavior models.

Concerning the application of machine learning to IPTV systems, a deep neural network-based model is proposed in [9] to choose the appropriate bit-rate to enable adaptive bit-rate streaming. Similarly, authors in [10] propose a collaborative filtering method based on transfer learning, which predicts the neighbor peer having best QoS by using his past usage experiences of a small number of other peers who have evaluated this node. To restrict the cloud rental cost to a desired QoS level, authors in [11] use reinforcement learning in a cloud assisted P2P streaming system. Fuzzy logic is used in [12] to construct a P2P IPTV overlay. The authors define priority to choose a parent peer which is a combination of peer's age, upload bandwidth, and utilization. Fuzzy logic is used to determine the overall priority of a peer for parent selection. Prediction of the next channel in IPTV is used to pre-stream the likely channels to users they may choose to watch next. For this purpose an agent-based model attempts to predict the next channel in [13]. The efficiency of different classifiers over the prediction of the next channel has been evaluated in [14]. For session duration prediction, authors in [15] first analyze session length in mobile based online services and then propose a Gradient Boosted Tree based algorithm to predict the length of the users' sessions. They use this prediction for video recommendation. Video disruption has a large impact on video streaming performance by creating high peer dynamics. So minimizing video disruption can improve streaming performance in P2P systems. To minimize video disruption due to user dynamics, one solution is to identify stable peers in the network. For stable peer identification, authors in [3] employ a contextual approach which not only considers users' past sessions but also includes the impacting variables of the session. They use a Bayesian network model in global and local scenarios. In the global scenario, a Bayesian network is trained and tested on mixed data of all users while in the local case they train and test the model on data of individual users. Their results show that accuracy is far better in the local case. We consider this latter approach to improve its performance further by reducing the prediction error.

III. MACHINE LEARNING MODELS FOR SESSIONS

As discussed above, very few existing works have attempted to include all important variables to model user behavior, so our work aims to consider all established variables for which data are available. These variables have been taken from [8] which has synthesized user behavior measurements for extraction of these variables. These are time-of-day, content type, arrival rate, departure rate, session duration, and popularity. Furthermore, we target a local model which learns and predicts the behavior of an individual user. Such a model needs to be trained on the data of user activities. To test the model, we will provide it with other available information except the current session duration which it needs to predict. Since we need to predict the session duration, which is a regression type problem, we chose to use Support Vector Regressor (SVR) model for this problem. To the best of our knowledge, SVR has not been used before for this particular problem but it is used for many other regression problems. Authors in [23] used SVR machine learning models to improve the accuracy of short-term forecasting algorithms based on past electrical load data. To observe the efficiency of SVR over this problem, we also re-use a Bayesian network model from [3] and compare the results of SVR and Bayesian network.

A. Support Vector Regressor (SVR)

As mentioned above, our problem is one of regression type, which is a generalization of the classification problem. In regression, the output of a model is a continuous value in contrast to an output belonging to a finite set. SVR is a generalization of Support Vector Machine (SVM) which is aimed at binary classification problems. SVMs formulate the binary classification problems as convex optimization problems which require finding the maximum margins that separate the hyperplane. The hyperplane is represented through support vectors. Introduction of an insensitive region around the function generalizes SVM to SVR. This region is termed as tube. SVR attempts to solve the optimization problem of approximating the continuous valued function and in the meanwhile balancing the complexity of the model and prediction error. It means that first a convex insensitive loss function is minimized and the flattest tube (containing most of the training instances) is found. In the next step, the convex optimization problem is solved, which has a unique solution, through appropriate numerical algorithms for optimization. The support vectors represent the hyperplane which consists of training samples lying outside the tube boundary. The most influential instances affecting the shape of the tube are contained in the support vectors [16]. To build and test the SVR model for our problem, we use the Python machine learning library's built-in SVR model. The SVR model has a large number of parameters, which are configurable according to the data nature. To maximize support vector regression efficiency we need to pick the best values for these parameters for the model. Below, these parameters are described in detail and the selected value for our model.

- Kernel: Our selected Python machine learning library provides several kernel functions for SVR. The kernel functions, which are widely utilized are Linear, Polynomial, Sigmoid, and Radial Basis. One needs to select the right kernel when applying the SVR technique. Kernel selection is not trivial as it involves optimization strategies to get the best combination. We applied each kernel one by one and analyzed the results. In the developed SVR model, we select the Radial Basis Function (RBF) kernel [17]. RBF kernel is the most commonly used kernel with a strong learning efficiency. RBF kernel functions are valid regardless of the conditions.
- Gamma: The gamma parameter intuitively describes the degree of which the influence of a particular sample of training extends, with low values meaning "far" and high values meaning "near." The gamma parameters can be interpreted as the opposite of the sample influence radius selected by the model as support vectors. We set different values for gamma parameter during experiments and select 0.1 as the best value for our developed model.
- Epsilon (ε): It specifies the epsilon-tube within which there is no penalty associated with predicted points within a epsilon distance from the actual value in the training loss function. Parameter ε controls the width of the epsilon-tube, used to fit the training data. The ε value can influence the number of support vectors used to construct the regression function. The greater the ε, the fewer support vectors are chosen. However, greater ε values lead to more flat estimates. We set different values for epsilon and analyzed the results. We chose number 0.1 for epsilon, which gives higher accuracy than other values.
- C: C is a regularization parameter that manages the tradeoff between obtaining a low training error and a low test error, which is the opportunity to generalize a model to unseen data. We set different values for C during the experiments and selected 100 as the best value for our developed model.

B. Bayesian Network Model

A Bayesian network (BN) is a pair (G, P), where $G = (V, E)$ is a Directed Acyclic Graph (DAG) containing vertices V and edges E [18]. Vertices/nodes represent random variables and directed edges show informational or causal dependencies among variables [20]. Let us consider a set of random variables, $U = X_1, \dots, X_n$. Then P is a set of conditional probability functions $P(X_i | X_j . X_j \in Parents(X_i))$. A directed link in G from X_j to X_i means X_j is the parent of X_i. Parent node has a direct influence on a child node through which the dependency relationships between the two variables is modeled. Variables having no parent are independent. Bayesian networks are a direct representation of the real world, not of the reasoning process. Although the directed edges in a network show direct dependency relationship between the variables, the reasoning process can propagate the information in any direction depending on the required result. Conditional independence makes a Bayesian network efficient to find the joint distribution. It states that a variable is conditionally independent from all its descendants given the states of its parents. For a given Bayesian network, the joint probability distribution over U can be computed as shown in (1):

$$P(U) = \prod_{i=1}^{n} P\left(X_i \middle| X_j, X_j \in Parents(X_i)\right) \quad (1)$$

The network structure can either be manually constructed by a domain expert or it can be developed from data through learning algorithms [19]. Similarly, parameters of each node which are needed to characterize each variable can be assigned manually to each node or learned from data [20]. Such a model naturally suits the user behavior model in P2P IPTV systems since session duration has dependency relationships with other variables such as content type and popularity. As shown in [3] the Bayesian network has been manually constructed and parameters are learned from data. This Bayesian network is a mix of continuous and discrete variables. Time-of-day and content type are independent variables while the other variables are dependent ones. Bayesian network allows estimating any variable in the model, in our case this variable is the session duration which will be estimated in the presence of other variables. We further explain these variables in terms of the dataset in the next section.

IV. EXPERIMENTS

A. Data Description

The dataset generated through the established model of [21] was used. This model generates a synthetic dataset and it was used because a real dataset for such an experiment was not available. All the datasets we have analyzed either lack major variables or they do not allow tracing individual users over different sessions. Our dataset consist of 462,825 instances. The dataset is divided into two subsets, the training set and the test set using Python library [18]. The train_test_split function of the library was used to divide the data. The random_state parameter was set to 5, which will ensure the same random number sequence is produced every time the code is run. The test_size parameter was set to 0.3, which divides the dataset into 70% for training and 30% for testing. We build our model using the training subset and evaluated it with the testing subset.

B. Feature Selection

- Join Time τ_j: This field includes a number from 1 to 1440 which reflects the user's joining day time in minutes. This field is available in the dataset.
- Departure Time τ_d: Like joining time, departure time shows the time at which a user leaves a channel. This field is also available in the dataset.
- Session Duration: The length of time between joining time and departure time which is essentially the time a user spends online. It is calculated from τ_j and τ_d.
- Arrival Rate: Arrival rate is the number of joins in a specific minute of the day. It is deduced from the dataset.
- Departure Rate: It is calculated like the arrival rate by counting the number of leaves/quits in a particular minute.
- Population: Population is the online user count at a given minute of the day. It is also called popularity.
- Content Type: This feature shows the type of the content broadcasted at each minute of the day. There are three types of content, which are fiction, sports, and reality. These types are labeled as 1, 2, and 3 respectively.

All these variables were provided to our chosen machine learning models except departure time, since [8] does not mention any impact of departure time on session duration.

C. Efficiency Measurement

To measure the performance of our model, we used Mean Absolute Error (MAE), Mean Squared Error (MSE) and Root Mean Squared Error (RMSE).

MAE is an average of absolute differences between the actual target values and the model predicted values. The MAE is a linear score which means that all the individual differences are weighted equally in the average:

$$MAE = \left(\frac{1}{n}\right) \sum_{i=1}^{n} |y_i - \hat{y}_i| \quad (2)$$

where, n denotes the number of records in the data set and y_i and $\hat{y}_i$ are the predicted and target value respectively the for i^{th} record.

MSE calculates the square difference between the model predicted value (session duration) and the actual target value for each point, and then averages those values. For a perfect model this value would be zero and the distance of its value from zero denotes the error. MSE can be computed as:

$$MSE = \left(\frac{1}{n}\right) \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \quad (3)$$

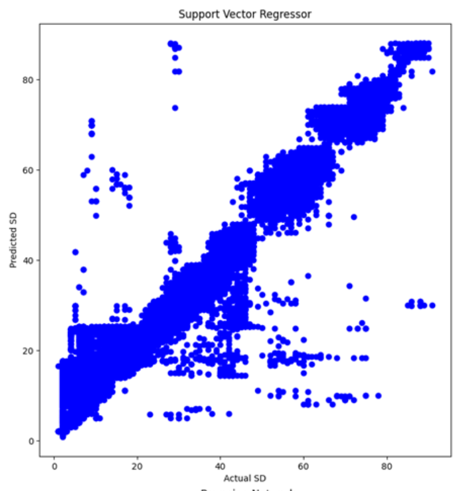
RMSE is a metric similar to MSE. It is actually measured in two stages. First, normal mean error is taken. The square root is added to allow the error ratio to be the same as the targets number. RMSE is computed by (4):

$$RMSE = \sqrt{\left(\frac{1}{n}\right) \sum_{i=1}^{n} (y_i - \hat{y}_i)^2} \quad (4)$$

D. Results

To develop the SVR model, Radial Basis Function kernel in Python was used. Similarly, for the BN model the Bayes Net Toolbox for MATLAB [24] was utilized. Furthermore, Conditional Gaussian inference was used as this model involves continuous variables and to learn the parameters we used the maximum likelihood learning algorithm. We trained the SVR model on the training data subset making the session duration the target variable. Then we fed the testing subset to our model to predict the session duration. For a fair comparison, we trained the BN model on the same training subset and evaluated it on the same test subset. A scatter plot of actual versus predicted sessions can be seen in Figure 1 for both SVR and BN models.

(a)



(b)

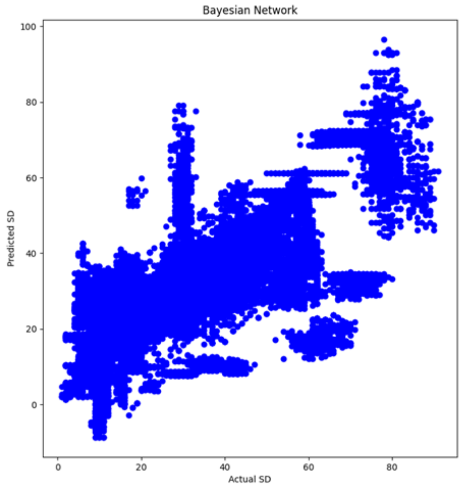


Fig. 1. Actual versus predicted sessions (in minutes).

In such a plot the model with higher accuracy exhibits more points near the slope while their spread is small. It is obvious from these two plots that for SVR the points are aligned in such a way that it shows a slope and furthermore their distribution is uniform for shorter and longer sessions. By contrast, in the case of BN, the spread is larger which shows larger errors and furthermore for certain lengths the error is further larger. This shows an inconsistency by the BN model and larger error than SVR. For a concrete comparison, MAE, MSE and RMSE were calculated for the two models and were plotted in Figure 2. Considering the three parameters, SVR performance is far superior than BN's. Since the unit of session duration is the minute in our experiment, it is evident that the MAE of 1.26 is a small error produced by SVR.

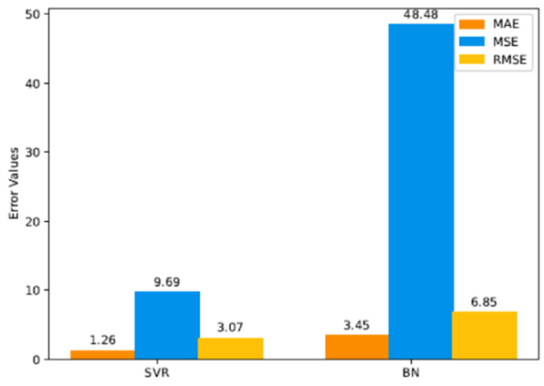


Fig. 2. MAE, MSE, and RMSE of SVR and BN.

To conclude the discussion on results, the plots in Figures 1-2 clearly indicate the superior accuracy of SVR model over the BN model for this problem on the considered dataset.

V. CONCLUSION

P2P networking is a suitable choice for broadcasting live videos, thanks to its scalability and ease of deployment. Since the building blocks in these networks are the peers, they exhibit the behavior of their users and this network becomes very dynamic. The leaving of a peer may disrupt streaming to consumer peers, which is the key problem for QoS of the P2P network.

To handle this issue we proposed in advance prediction of the session duration of each peer. This will enable each consumer peer to know the provider peer's quit time. Therefore, P2P network can be managed in such a way that stable peers function as provider peers avoiding the stream disruption due to abrupt departures. Towards this, an SVR model was proposed, which was compared with an existing BN model. We trained both models on the same training data subset and then evaluated them on the same testing data subset. Our results show that SVR model performs better than BN and reduces the RMSE from 6.85 to 3.07 which is a significant improvement. As a future perspective, other machine learning models will be studied and will be evaluated on real datasets. Furthermore, we plan to propose a complete framework for topology design and management through the predictions of machine learning models.

REFERENCES

[1] Y. Tang, L. Sun, J.-G. Luo, S.-Q. Yang, and Y. Zhong, "Improving Quality of Live Streaming Service over P2P Networks with User Behavior Model," in *Advances in Multimedia Modeling*, T.-J. Cham, J. Cai, C. Dorai, D. Rajan, T.-S. Chua, and L.-T. Chia, Eds. Berlin, Heidelberg: Springer, 2006, pp. 333–342, doi: 10.1007/978-3-540-69429-8_34.

[2] F. Wang, Y. Xiong, and J. Liu, "mTreebone: A Collaborative Tree-Mesh Overlay Network for Multicast Video Streaming," *IEEE Transactions on Parallel and Distributed Systems*, vol. 21, no. 3, pp. 379–392, Mar. 2010, doi: 10.1109/TPDS.2009.77.

[3] I. Ullah, G. Doyen, G. Bonnet, and D. Gaïti, "A Bayesian approach for user aware peer-to-peer video streaming systems," *Signal Processing: Image Communication*, vol. 27, no. 5, pp. 438–456, May 2012, doi: 10.1016/j.image.2012.02.007.

[4] B. Zhang *et al.*, "Understanding user behavior in Spotify," *Proceedings of the IEEE INFOCOM 2013 (Turin, Italy, April 14-19, 2013)*, pp. 220–224, 2013, doi: 10.1109/INFCOM.2013.6566767.

[5] S. Budhkar and V. Tamarapalli, "An overlay management strategy to improve peer stability in P2P live streaming systems," in *2016 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS)*, Nov. 2016, pp. 1–6, doi: 10.1109/ANTS.2016.7947813.

[6] K. Park, D. Chang, J. Kim, W. Yoon, and T. Kwon, "An Analysis of User Dynamics in P2P Live Streaming Services," in *2010 IEEE International Conference on Communications*, Cape Town, South Africa, May 2010, doi: 10.1109/ICC.2010.5502009.

[7] Liu, Z., Wu, C., Li, B., Zhao, S., 2009. Distilling Superior Peers in Large-Scale P2P Streaming Systems, in: IEEE INFOCOM 2009. Presented at the IEEE INFOCOM 2009, IEEE, Rio de Janeiro, Brazil, pp. 82–90. https://doi.org/10.1109/INFCOM.2009.5061909

[8] I. Ullah, G. Doyen, G. Bonnet, and D. Gaiti, "A Survey and Synthesis of User Behavior Measurements in P2P Streaming Systems," *IEEE Communications Surveys Tutorials*, vol. 14, no. 3, pp. 734–749, Third 2012, doi: 10.1109/SURV.2011.082611.00134.

[9] A. Lekharu, K. Y. Moulii, A. Sur, and A. Sarkar, "Deep Learning based Prediction Model for Adaptive Video Streaming," in *2020 International Conference on COMmunication Systems NETworkS (COMSNETS)*, Jan. 2020, pp. 152–159, doi: 10.1109/COMSNETS48256.2020.9027383.

[10] W. Ma, Q. Zhang, C. Mu, and M. Zhang, "QoS Prediction for Neighbor Selection via Deep Transfer Collaborative Filtering in Video Streaming P2P Networks," *International Journal of Digital Multimedia Broadcasting*, vol. 2019, Jan. 2019, doi: https://doi.org/10.1155/2019/1326831, Art no. 1326831.

[11] M. Sina, M. Dehghan, and A. M. Rahmani, "CaR-PLive: Cloud-assisted reinforcement learning based P2P live video streaming: a hybrid approach," *Multimedia Tools and Applications*, vol. 78, no. 23, pp. 34095–34127, Dec. 2019, doi: 10.1007/s11042-019-08102-1.

[12] K. Pal, M. C. Govil, and M. Ahmed, "FLHyO: fuzzy logic based hybrid overlay for P2P live video streaming," *Multimedia Tools and Applications*, vol. 78, no. 23, pp. 33679–33702, Dec. 2019, doi: 10.1007/s11042-019-08010-4.

[13] A. Ghaderzadeh, M. Kargahi, and M. Reshadi, "ReDePoly: reducing delays in multi-channel P2P live streaming systems using distributed intelligence," *Telecommunication Systems*, vol. 67, no. 2, pp. 231–246, Feb. 2018, doi: 10.1007/s11235-017-0336-x.

[14] I. Basicevic, D. Kukolj, S. Ocovaj, G. Cmiljanovic, and N. Fimic, "A Fast Channel Change Technique Based on Channel Prediction," *IEEE Transactions on Consumer Electronics*, vol. 64, no. 4, pp. 418–423, Nov. 2018, doi: 10.1109/TCE.2018.2875271.

[15] T. Vasiloudis, H. Vahabi, R. Kravitz, and V. Rashkov, "Predicting Session Length in Media Streaming," presented at the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2017, Tokyo, Japan, Aug. 2017, pp. 977–980.

[16] M. Awad and R. Khanna, "Support Vector Regression," in *Efficient Learning Machines: Theories, Concepts, and Applications for Engineers and System Designers*, M. Awad and R. Khanna, Eds. Berkeley, CA: Apress, 2015, pp. 67–80.

[17] F. Pedregosa *et al.*, "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, no. 85, pp. 2825–2830, 2011.

[18] G. Chen and Z. Ge, "Robust Bayesian networks for low-quality data modeling and process monitoring applications," *Control Engineering Practice*, vol. 97, Apr. 2020, doi: 10.1016/j.conengprac.2020.104344, Art no. 104344.

[19] T. Talvitie, R. Eggeling, and M. Koivisto, "Learning Bayesian networks with local structure, mixed variables, and exact algorithms," *International Journal of Approximate Reasoning*, vol. 115, pp. 69–95, Dec. 2019, doi: 10.1016/j.ijar.2019.09.002.

[20] L. Azzimonti, G. Corani, and M. Zaffalon, "Hierarchical estimation of parameters in Bayesian networks," *Computational Statistics & Data Analysis*, vol. 137, pp. 67–91, Sep. 2019, doi: 10.1016/j.csda.2019.02.004.

[21] G. Bonnet, I. Ullah, G. Doyen, L. Fillatre, D. Gaïti, and I. Nikiforov, "A Semi-Markovian Individual Model of Users for P2P Video Streaming Applications," in *2011 4th IFIP International Conference on New Technologies, Mobility and Security*, Paris, France, Feb. 2011, doi: 10.1109/NTMS.2011.5721042.

[22] A. Tanovic, I. Androulidakis, and F. Orucevic, "Analysis of IPTV Channels Watching Preferences in Bosnia and Herzegovina," *Engineering, Technology & Applied Science Research*, vol. 1, no. 5, pp. 105–113, Oct. 2011.

[23] N. T. Dung and N. T. Phuong, "Short-Term Electric Load Forecasting Using Standardized Load Profile (SLP) And Support Vector Regression (SVR)," *Engineering, Technology & Applied Science Research*, vol. 9, no. 4, pp. 4548–4553, Aug. 2019.

[24] "Google Code Archive - Long-term storage for Google Code Project Hosting." https://code.google.com/archive/p/bnt/ (accessed Jul. 01, 2020).