INTERNATIONAL JOURNAL OF MULTIDISCIPLINARY RESEARCH AND ANALYSIS

ISSN(print): 2643-9840, ISSN(online): 2643-9875

Volume 06 Issue 06 June 2023

DOI: 10.47191/ijmra/v6-i6-14, Impact Factor: 7.022

Page No. 2457-2464

A Framework for Multi-Task Learning in Dynamic Adaptive Streaming Over HTTP

Koffka Khan

Department of Computing and Information Technology, The University of the West Indies, St Augustine, Trinidad and Tobago, W.I.

ABSTRACT: This paper presents a framework with a taxonomy for multi-task learning in the context of Dynamic Adaptive Streaming over HTTP (DASH). DASH is a widely used technology for video streaming, and multi-task learning has emerged as a promising approach to enhance the performance and user experience of DASH systems by jointly optimizing multiple related tasks. The framework provides a structured approach to design, train, and evaluate multi-task learning models in DASH, while the taxonomy categorizes the key components and approaches within the framework. The taxonomy includes task types, multitask learning approaches, input features, and training strategies. Task types encompass video quality adaptation, buffer management, bandwidth estimation, content pre-fetching, and resource allocation, representing the specific tasks involved in DASH. Multi-task learning approaches encompass methodologies such as shared representation learning, task-specific layers, multi-head architectures, knowledge distillation, and reinforcement learning, offering flexibility in model design and optimization. Input features cover video characteristics, network conditions, device capabilities, and user preferences, providing the necessary information for informed decision-making across tasks. Training strategies include joint training, alternate training, hierarchical training, task weighting, and task balancing, determining how the multi-task learning model is trained and optimized. By following the presented framework and taxonomy, researchers and practitioners can systematically approach the design, training, and evaluation of multi-task learning models in DASH. The framework enables the development of efficient and adaptive video streaming systems by leveraging the interdependencies among tasks. The taxonomy helps organize the components and approaches within the framework, aiding in a better understanding of the various aspects of multi-task learning in the DASH context. Overall, this framework and taxonomy provide a valuable resource for advancing the field of multitask learning in the dynamic and complex domain of video streaming over HTTP.

KEYWORDS: multi-task, learning, DASH, framework, video, quality, optimization, streaming

I. INTRODUCTION

Dynamic Adaptive Streaming over HTTP (DASH) [13), 14), 11)] has become a widely adopted technology for video streaming, allowing adaptive video quality delivery based on network conditions and device capabilities. In this context, multi-task learning has emerged as a powerful approach to enhance the performance and user experience of DASH systems by jointly optimizing multiple related tasks. To effectively apply multi-task learning in DASH, a framework with a taxonomy can provide a structured and comprehensive approach.

The framework presented here aims to guide researchers and practitioners in the implementation of multi-task learning in DASH systems, with a focus on video quality adaptation, buffer management, bandwidth estimation, content pre-fetching, and resource allocation. The taxonomy within the framework helps categorize and organize the various components and approaches involved in multi-task learning for DASH.

The taxonomy includes task types, multi-task learning approaches, input features, and training strategies. The task types categorize the specific tasks involved in DASH, such as video quality adaptation, buffer management, bandwidth estimation, content pre-fetching, and resource allocation. These tasks form the foundation for designing multi-task learning systems in the DASH context.

This paper consists of five sections. The key elements of Dynamic adaptive streaming over HTTP and how it relates to multitask learning is outlined in section two. The multi-task learning in DASH (MTLD) taxonomy with explanations is given in section three.

In section four the framework for multi-task learning in DASH (MTLD) is given. How the taxonomy is used in the MTLD framework is explained. Finally, the conclusion is given in section four.

II. DYNAMIC ADAPTIVE STREAMING OVER HTTP (DASH)

Dynamic Adaptive Streaming over HTTP (DASH) is a popular streaming protocol that enables the adaptive delivery of multimedia content over the internet. DASH allows for seamless streaming by dynamically adjusting the quality of the media based on the viewer's network conditions and device capabilities [12)]. There are several types of DASH implementations that differ in the way they handle content adaptation. There are three main types. Firstly, there is Segment-based DASH [26)], also known as "Chunked" DASH. It divides the multimedia content into small segments or chunks, typically a few seconds in duration. Each segment is encoded at different quality levels, creating multiple representations of the same content. The DASH client dynamically selects the appropriate segment quality based on the available bandwidth and device capabilities. It requests and downloads the segments in real-time, seamlessly switching between different quality levels as needed. This approach provides smooth playback and adapts to changing network conditions.

Secondly, there is Layered DASH [23)], also known as "Scalable" DASH. It uses layered coding techniques to adapt the content based on available network conditions. Instead of dividing the content into separate quality levels, layered DASH encodes the content as a base layer and additional enhancement layers. The base layer provides a lower quality version of the content, while the enhancement layers contain additional data to improve the quality. The DASH client can selectively download and combine the base and enhancement layers to adapt the content quality. This approach allows for fine-grained adaptation by enabling the client to choose which layers to download based on the network conditions. Thirdly, there is Adaptive Bitrate Streaming (ABR) [16)]. It is a form of DASH that focuses on dynamically adjusting the bitrate of the content rather than using separate representations or layers. ABR encodes the content at different bitrates and divides it into small chunks. The DASH client continuously monitors the network conditions and selects the appropriate bitrate for each chunk. It requests and downloads the chunks at the selected bitrate, adapting to the available bandwidth in real-time. ABR is widely used by popular streaming platforms and provides smooth playback even in fluctuating network conditions.

Dynamic Adaptive Streaming over HTTP (DASH) optimizes streaming by employing various techniques to adapt the delivery of multimedia content based on the viewer's network conditions and device capabilities. There are many ways in which DASH optimizes streaming. The first way is Adaptive Quality Selection [21)] where DASH divides the multimedia content into small segments or chunks, each encoded at different quality levels. The DASH client dynamically selects the appropriate segment quality based on the viewer's available bandwidth and device capabilities. By continuously monitoring the network conditions, DASH adjusts the quality in real-time, ensuring smooth playback and minimizing buffering or stuttering issues [15)]. The second way is Bitrate Adaptation where DASH utilizes adaptive bitrate streaming (ABR) techniques to adjust the bitrate of the content dynamically. By encoding the content at multiple bitrates and dividing it into small chunks, DASH selects the appropriate bitrate for each chunk based on the viewer's available bandwidth. This adaptive bitrate selection ensures that the viewer receives the highest quality possible without experiencing interruptions due to limited bandwidth.

The third way is Buffer Management [10)] where DASH employs intelligent buffer management techniques to optimize streaming performance. It maintains a playback buffer on the client-side, preloading and buffering segments ahead of time. By doing so, DASH can compensate for temporary fluctuations in the network conditions. The buffer allows for smoother playback by ensuring a continuous supply of segments, even if there are short-lived drops in network bandwidth. The fourth way is Network Awareness [5)] where DASH incorporates network-awareness mechanisms to monitor the available bandwidth and latency in real-time. It uses this information to make informed decisions about which segment quality or bitrate to request and download. By adapting to the current network conditions, DASH optimizes the streaming experience by avoiding quality levels that would result in buffering or playback interruptions.

The fifth way is Content Caching [18)] where DASH can take advantage of content caching at various levels, such as content delivery networks (CDNs) or local caches. Caching frequently accessed segments closer to the viewer reduces the latency and improves the overall streaming performance. By minimizing the distance and network hops between the viewer and the content, DASH optimizes the delivery of segments, leading to faster startup times and smoother streaming. The sixth way is Multi-Server Adaptation [4)] where in some scenarios, DASH can distribute the content across multiple servers or CDNs. This approach, known as multi-server adaptation, allows for load balancing and efficient utilization of network resources. By dynamically selecting the best server or CDN for each segment, DASH optimizes the streaming experience by minimizing congestion and maximizing the available bandwidth. Overall, DASH optimizes streaming by adapting the content delivery based on real-time network conditions, employing adaptive quality selection, bitrate adaptation, buffer management, network awareness, content caching, and multi-server adaptation techniques. These optimizations collectively ensure smooth playback, reduce buffering, and provide the best possible streaming experience for viewers.

Multi-task learning can be applied to DASH (Dynamic Adaptive Streaming over HTTP) to improve its performance and optimize the streaming experience. Multi-task learning refers to training a single model to simultaneously learn and perform multiple related tasks. In the context of DASH, multi-task learning can be employed in the following ways. The first is in Quality Prediction and Adaptation [7)]. One task in DASH is predicting the quality level or bitrate that should be selected for a given segment based on the network conditions. Another task is adapting the content delivery to the viewer by selecting the appropriate quality level. By jointly learning these tasks, a multi-task learning approach can improve the accuracy of quality prediction and enable more effective content adaptation. The model can learn to identify patterns and relationships between network conditions and the optimal quality level, leading to better decision-making during content adaptation.

The second is in Buffer Management and Bitrate Selection [8)]. Buffer management involves deciding how many segments to buffer in advance and when to request new segments to maintain a continuous playback experience. Bitrate selection determines the appropriate bitrate for each segment based on the available network bandwidth. Multi-task learning can integrate these two tasks, enabling the model to learn the optimal buffer management strategy while considering the bitrate selection. By jointly optimizing these tasks, the model can improve the buffer utilization and ensure smoother playback without interruptions. The third is in Quality Evaluation and Reinforcement Learning [20)]. Multi-task learning can incorporate a quality evaluation task, where the model learns to assess the quality of the streaming experience based on user feedback or objective metrics. This quality evaluation can be used as a reward signal for reinforcement learning. By combining quality evaluation with reinforcement learning techniques, the model can learn to make better decisions during content adaptation, optimizing the streaming experience based on user preferences and quality feedback.

The fourth is in Content Delivery Optimization [17)]. Multi-task learning can be applied to optimize the content delivery process in DASH. Tasks such as content caching, segment scheduling, and server selection can be jointly learned to maximize the utilization of network resources, minimize latency, and improve overall streaming performance. By considering multiple related tasks simultaneously, the model can make more informed decisions and optimize the content delivery strategy dynamically. Overall, multi-task learning in DASH can enhance various aspects of the streaming process, including quality prediction and adaptation, buffer management and bitrate selection, quality evaluation and reinforcement learning, and content delivery optimization. By jointly learning multiple tasks, DASH can improve its performance, adaptability, and overall streaming experience for viewers.

III. MULTI-TASK LEARNING IN DASH (MTLD) TAXONOMY

The following taxonomy enables one to understand and categorize the different dimensions and techniques involved in MultiTask Learning as applied to DASH. It categorizes the different aspects of Multi-Task Learning (MTL) in DASH based on task types, MTL approaches, input features, and training strategies. Each category is shown as a bullet point, followed by the subcategories indented using vertical bars. Each level is represented by a vertical bar (|), and subcategories or subtopics are indented with two underscores (__). This representation gives an overview of the taxonomy structure and the relationships between different categories in a hierarchical manner.

```
- Task Types:
  |--- Video Quality Adaptation
  |--- Buffer Management
  |--- Bandwidth Estimation
  |--- Content Pre-fetching
  |--- Resource Allocation

- Multi-Task Learning Approaches:
  |--- Shared Representation Learning
  |--- Task-Specific Layers
  |--- Multi-Head Architectures
  |--- Knowledge Distillation
  |--- Reinforcement Learning

- Input Features:
  |--- Video Characteristics
  |--- Network Conditions
  |--- Device Characteristics
  |--- User Preferences

- Training Strategies:
  |--- Joint Training
  |--- Alternate Training
  |--- Hierarchical Training
  |--- Task Weighting
  |--- Task Balancing
```

A description of each category follows. The first category is Task Types. This category outlines the specific tasks or objectives in DASH that can benefit from MTL. It includes tasks such as Video Quality Adaptation, Buffer Management, Bandwidth Estimation, Content Pre-fetching, and Resource Allocation. These tasks represent different aspects of optimizing the streaming experience. The second category is Multi-Task Learning Approaches. This category focuses on the approaches or techniques used for MTL in DASH. It includes Shared Representation Learning, where a common representation is learned and shared among tasks; Task-Specific Layers, where separate layers are used for each task; Multi-Head Architectures, which involve taskspecific branches in the model; Knowledge Distillation, which transfers knowledge from a teacher model to a student model; and Reinforcement Learning, which employs reinforcement learning techniques for joint task optimization.

The next category is Input Features. This category identifies the different types of input features that can be used in MTL for DASH. It includes Video Characteristics, such as resolution, encoding parameters, or content complexity; Network Conditions, which encompass bandwidth, latency, and other network-related metrics; Device Characteristics, such as device capabilities or available resources; and User Preferences, which capture user-specific preferences or requirements. The final category is Training Strategies. This category focuses on the strategies used during the training phase of MTL in DASH. It includes Joint Training, where all tasks are trained simultaneously; Alternate Training, where tasks are trained in an alternating or sequential manner; Hierarchical Training, where tasks are organized hierarchically, with some tasks supervising others; Task Weighting, where different weights are assigned to each task during training; and Task Balancing, which aims to achieve a balance between tasks in terms of training focus and optimization. In general, this taxonomy provides a comprehensive framework for understanding and categorizing MTL in DASH, considering the specific tasks involved, the approaches employed, the input features utilized, and the training strategies applied. It helps organize the different aspects of MTL in DASH, facilitating the development and exploration of MTL techniques for optimizing streaming performance.

A explanation for the various subcategories within the taxonomy on Multi-Task Learning in Dynamic Adaptive Streaming over HTTP (DASH) is now given:

Task Types:

a. Video Quality Adaptation: This task focuses on selecting the appropriate video quality (bitrate) for a given network condition and device capabilities.
b. Buffer Management: This task involves managing the playback buffer to ensure smooth video playback and minimize interruptions.
c. Bandwidth Estimation: This task aims to estimate the available network bandwidth to make informed decisions regarding video quality and buffer management.
d. Content Pre-fetching: This task involves predicting future video segments that are likely to be requested and proactively Fetching them to reduce latency.
e. Resource Allocation: This task deals with allocating system resources such as network bandwidth, CPU, and memory among different tasks and components of the streaming system.

Multi-Task Learning Approaches:

a. Shared Representation Learning [3), 9)]: This approach involves training a single model with shared parameters to perform Multiple tasks simultaneously. The model learns a shared representation that captures common features and relationships across tasks.

b. Task-Specific Layers [**Error! Reference source not found.**, **Error! Reference source not found.**]: In this approach, a model consists of shared layers for common features and separate task-specific layers for individual tasks. The shared layers capture shared information, while task-specific layers capture task-specific knowledge.

c. Multi-Head Architectures [2), 24)]: This approach uses multiple output heads in a neural network, with each head dedicated to a specific task. The shared layers extract shared features, and each head learns task-specific representations and predictions.

d. Knowledge Distillation [1), 27)]: This approach involves training a primary model (teacher) on multiple tasks and using it to Guide the training of secondary models (students) that focus on specific tasks. The secondary models learn from the knowledge of the primary model.

e. Reinforcement Learning [6), 19)]: This approach combines reinforcement learning with multi-task learning to optimize the Streaming system's performance. The reinforcement learning agent learns to make decisions for different tasks based on rewards and system objectives.

Input Features:

a. Video Characteristics: Features related to the video content, such as resolution, frame rate, spatial and temporal complexity, and encoding parameters.

b. Network Conditions: Features related to the network environment, including available bandwidth, round-trip time (RTT), Packet loss rate, and congestion indicators.

c. Device Characteristics: Features describing the device capabilities, such as processing power, battery level, screen size, and Display resolution.

d. User Preferences: Features representing user preferences, including history, context, QoS preferences, and user behavior Patterns.

Training Strategies:

a. Joint Training: All tasks are trained together using a single loss function that combines the objectives of different tasks. The Model learns to jointly optimize multiple objectives.

b. Alternate Training: The model is trained in an alternating manner, focusing on one task at a time while freezing the Parameters related to other tasks. The training process iterates between different tasks until convergence.

c. Hierarchical Training: Tasks are trained in a hierarchical manner, where lower-level tasks are trained first, and their outputs are used as inputs for higher-level tasks. This approach captures dependencies and allows for the transfer of knowledge between tasks.

d. Task Weighting: Different tasks can be assigned different weights or importance during training to reflect their relative Significance or system objectives.

e. Task Balancing: Techniques such as curriculum learning or data balancing can be employed to ensure that the model Receives sufficient training examples for each task and avoids bias towards dominant tasks.

Note that the taxonomy provided above is a general overview of multi-task learning in the context of Dynamic Adaptive Streaming over HTTP (DASH). Actual implementations and research in the field may vary, and new approaches may emerge over time.

IV. MULTI-TASK LEARNING IN DASH (MTLD) FRAMEWORK

The framework presented below provides a comprehensive description of the steps involved in applying multi-task learning to Dynamic Adaptive Streaming over HTTP (DASH), along with a taxonomy that categorizes the key components and approaches within the framework. In our framework and we use the taxonomy derived in the previous section to categorize the techniques within that framework. The Framework for Multi-Task Learning in Dynamic Adaptive Streaming over HTTP (DASH) is as follows:

Problem Formulation:

a. Define the tasks: Identify the specific tasks involved in DASH, such as video quality adaptation, buffer management, bandwidth estimation, content pre-fetching, and resource allocation.
b. Define the objectives: Clearly define the objectives for each task, considering system performance metrics, user experience, and network conditions.
c. Identify input features: Determine the relevant input features that capture video characteristics, network conditions, device capabilities, and user preferences.

Data Collection and Preparation:

a. Collect training data: Gather a diverse dataset that includes examples of different network conditions, video content, and User behaviors.

b. Preprocess the data: Clean and preprocess the collected data, extract relevant features, and perform any necessary data Transformations or normalization.

Model Architecture:

a. Choose a suitable architecture: Select a neural network architecture that can accommodate multi-task learning, such as Shared representation learning, task-specific layers, or multi-head architectures.
b. Design the input layers: Define the input layers of the model to accept the relevant features identified in the problem Formulation stage.
c. Define shared layers: Determine the layers that capture shared representations and common features across tasks.
d. Specify task-specific layers: Incorporate task-specific layers that specialize in capturing task-specific knowledge and making Task-specific predictions.
e. Configure output layers: Configure the output layers corresponding to each task, ensuring that they produce the desired Predictions or decisions.

Training:

a. Loss function: Define a composite loss function that combines the objectives of different tasks, considering their relative importance or system objectives.
b. Optimization algorithm: Choose an appropriate optimization algorithm, such as stochastic gradient descent (SGD) or adaptive optimization methods like Adam or RMSprop.
c. Training strategy: Decide on the training strategy, such as joint training, alternate training, hierarchical training, or task weighting, based on the problem requirements and model architecture.
d. Train the model: Iterate through the training data, feeding the inputs into the model, computing the loss, and updating the model parameters using backpropagation and the chosen optimization algorithm.

Evaluation and Validation:

a. Evaluation metrics: Select appropriate evaluation metrics to assess the performance of each task, considering system-level performance, user experience, and other relevant criteria.
b. Validation set: Set aside a portion of the collected data as a validation set to monitor the model's performance during training and tune hyperparameters.
c. Validate the model: Periodically evaluate the model on the validation set to assess its performance, identify potential overfitting, and make necessary adjustments to the model or training process.

Testing and Deployment:

a. Test the model: Evaluate the trained model on unseen test data to measure its performance on real-world scenarios and validate its effectiveness.
b. Deployment considerations: Consider the deployment environment and any constraints or requirements specific to the target system or application.
c. Integration: Integrate the trained multi-task model into the DASH system, ensuring compatibility and appropriate interfaces with other components.
d. Monitor and fine-tune: Continuously monitor the performance of the deployed model and make necessary adjustments or ine-tuning to optimize its performance over time.

The taxonomy fits into the framework for multi-task learning in DASH as follows. The first is in Problem Formulation. The taxonomy helps in defining the tasks by identifying specific tasks involved in DASH, such as video quality adaptation, buffer management, bandwidth estimation, content pre-fetching, and resource allocation. It also aids in defining objectives for each task and identifying the relevant input features based on video characteristics, network conditions, device capabilities, and user preferences. The second is in Model Architecture. The taxonomy aligns with the model architecture stage by providing options for suitable MTL approaches, such as shared representation learning, task-specific layers, multi-head architectures, knowledge distillation, or reinforcement learning. It also guides the design of input layers, shared layers, task-specific layers, and output layers based on the identified tasks and input features.

The third is in Training. The taxonomy complements the training stage by assisting in the selection of appropriate training strategies, such as joint training, alternate training, hierarchical training, task weighting, or task balancing. It also helps in defining the loss function that combines the objectives of different tasks and in choosing the optimization algorithm for training the model. The fourth is in Evaluation and Validation. The taxonomy supports the evaluation and validation stage by suggesting relevant evaluation metrics for assessing the performance of each task, considering system-level performance, user experience, and other criteria. It also encourages the use of a validation set to monitor the model's performance during training and make necessary adjustments. The fifth is in Testing and Deployment. The taxonomy aligns with the testing and deployment stage by emphasizing the need to test the trained model on unseen data and evaluate its performance in real-world scenarios. It also highlights the importance of considering deployment considerations, integration with the DASH system, and monitoring the model's performance for continuous optimization.

By incorporating the taxonomy within the multi-task learning framework, it provides a structured and systematic approach to designing, training, evaluating, and deploying multi-task models in DASH. The taxonomy helps in organizing the different aspects of multi-task learning and ensures that important considerations specific to DASH are addressed at each stage of the framework.

V. CONCLUSIONS

In conclusion, the taxonomy presented within the framework for multi-task learning in DASH provides a structured and comprehensive approach to developing, training, and deploying multi-task models. The taxonomy categorizes different aspects of multi-task learning in DASH, including task types, MTL approaches, input features, and training strategies. By following this taxonomy, the framework enables researchers and practitioners to effectively tackle the challenges in DASH and optimize the streaming experience. The taxonomy helps in defining the specific tasks involved in DASH, such as video quality adaptation, buffer management, bandwidth estimation, content pre-fetching, and resource allocation. It guides the selection of appropriate MTL approaches, such as shared representation learning, task-specific layers, multi-head architectures, knowledge distillation, or reinforcement learning. It also aids in identifying relevant input features, such as video characteristics, network conditions, device capabilities, and user preferences. Moreover, the taxonomy assists in determining suitable training strategies, including joint training, alternate training, hierarchical training, task weighting, or task balancing. It supports the design of model architectures that incorporate shared layers, task-specific layers, and appropriate output layers for each task. It helps in defining loss functions that combine the objectives of different tasks and in selecting optimization algorithms for training the models. Furthermore, the taxonomy suggests evaluation metrics and emphasizes the use of validation sets to monitor and validate the performance of multi-task models during training. It also highlights the importance of testing the models on unseen data and considering deployment considerations and integration with the DASH system. Finally, the taxonomy within the multi-task learning framework for DASH provides a systematic and organized approach to tackle the complex challenges in adaptive streaming. By following this taxonomy, researchers and practitioners can effectively design, train, evaluate, and deploy multitask models that optimize the streaming experience, improve system performance, and enhance user satisfaction in DASH applications.

REFERENCES

1) Antaris, S., Rafailidis, D., & Girdzijauskas, S. (2020, December). EGAD: Evolving graph representation learning with selfattention and knowledge distillation for live video streaming events. In 2020 IEEE International Conference on Big Data (Big Data) (pp. 1455-1464). IEEE.
2) Barbalau, A., Ionescu, R. T., Georgescu, M. I., Dueholm, J., Ramachandra, B., Nasrollahi, K., ... & Shah, M. (2023). Ssmtl++: Revisiting self-supervised multi-task learning for video anomaly detection. Computer Vision and Image Understanding, 229, 103656.
3) Bengio, Y., Courville, A., & Vincent, P. (2013). Representation learning: A review and new perspectives. IEEE transactions on pattern analysis and machine intelligence, 35(8), 1798-1828.
4) Bruneau-Queyreix, J., Lacaud, M., & Négru, D. (2017, October). A Hybrid P2P/Multi-Server Quality-Adaptive LiveStreaming Solution Enhancing End-User's QoE. In Proceedings of the 25th ACM international conference on Multimedia (pp. 1261-1262).
5) Chang, Z., Zhou, X., Wang, Z., Li, H., & Zhang, X. (2019, April). Edge-assisted adaptive video streaming with deep learning in mobile edge networks. In 2019 IEEE Wireless Communications and Networking Conference (WCNC) (pp. 1-6). IEEE.
6) Chen, X., Proietti, R., Liu, C. Y., & Yoo, S. B. (2021). A multi-task-learning-based transfer deep reinforcement learning design for autonomic optical networks. IEEE Journal on Selected Areas in Communications, 39(9), 2878-2889.
7) Doshi, K., & Yilmaz, Y. (2022). Multi-task learning for video surveillance with limited data. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 3889-3899).
8) Elgendy, I. A., Zhang, W. Z., He, H., Gupta, B. B., & Abd El-Latif, A. A. (2021). Joint computation offloading and task caching for multi-user and multi-task MEC systems: reinforcement learning-based algorithms. Wireless Networks, 27(3), 2023-2038.
9) Hussein, A., & Hajj, H. (2022). Domain adaptation with representation learning and nonlinear relation for time series. ACM Transactions on Internet of Things, 3(2), 1-26.
10) Jiang, Z., Zhang, X., Huang, W., Chen, H., Xu, Y., Hwang, J. N., ... & Sun, J. (2019). A hierarchical buffer management approach to rate adaptation for 360-degree video streaming. IEEE Transactions on Vehicular Technology, 69(2), 21572170.

11) Khan, K., & Goodridge, W. (2018). Bandwidth Estimation Techniques for Relative'Fair'Sharing in DASH. International Journal of Advanced Networking and Applications, 9(6), 3607-3615.
12) Khan, K., & Goodridge, W. (2018). Future DASH applications: A survey. International Journal of Advanced Networking and Applications, 10(2), 3758-3764.
13) Khan, K., & Goodridge, W. (2018). QoE in DASH. International Journal of Advanced Networking and Applications, 9(4), 3515-3522.
14) Khan, K., & Goodridge, W. Markov Decision Processes for bitrate harmony in adaptive video streaming. In 2017 Future Technologies Conference (FTC), Vancouver, Canada, unpublished.
15) Koffka, K., & Wayne, G. (2018). A DASH Survey: the ON-OFF Traffic Problem and Contemporary Solutions. Computer Sciences and Telecommunications, (1), 3-20.
16) Kua, J., Armitage, G., & Branch, P. (2017). A survey of rate adaptation techniques for dynamic adaptive streaming over HTTP. IEEE Communications Surveys & Tutorials, 19(3), 1842-1866.
17) Li, J., Tian, Y., Huang, T., & Gao, W. (2010). Probabilistic multi-task learning for visual saliency estimation in video. International journal of computer vision, 90, 150-165.
18) Li, L., Shi, D., Hou, R., Chen, R., Lin, B., & Pan, M. (2020). Energy-efficient proactive caching for adaptive video streaming via data-driven optimization. IEEE Internet of Things Journal, 7(6), 5549-5561.
19) Liu, Z., Tian, J., Cai, Q., Zhao, X., Gao, J., Liu, S., ... & Gai, K. (2023, April). Multi-Task Recommendations with Reinforcement Learning. In Proceedings of the ACM Web Conference 2023 (pp. 1273-1282).
20) Long, L., Huang, F., Yin, Y., & Xu, Y. (2022). Multi-task learning for collaborative filtering. International Journal of Machine Learning and Cybernetics, 1-14.
21) Mao, H., Netravali, R., & Alizadeh, M. (2017, August). Neural adaptive video streaming with pensieve. In Proceedings of the conference of the ACM special interest group on data communication (pp. 197-210).
22) Tang, H., Liu, J., Zhao, M., & Gong, X. (2020, September). Progressive layered extraction (ple): A novel multi-task learning (mtl) model for personalized recommendations. In Proceedings of the 14th ACM Conference on Recommender Systems (pp. 269-278).
23) Uzakgider, T., Cetinkaya, C., & Sayit, M. (2015). Learning-based approach for layered adaptive video streaming over SDN. Computer Networks, 92, 357-368.
24) Vandenhende, S., Georgoulis, S., Van Gansbeke, W., Proesmans, M., Dai, D., & Van Gool, L. (2021). Multi-task learning for dense prediction tasks: A survey. IEEE transactions on pattern analysis and machine intelligence, 44(7), 3614-3633.
25) Wallingford, M., Li, H., Achille, A., Ravichandran, A., Fowlkes, C., Bhotika, R., & Soatto, S. (2022). Task adaptive parameter sharing for multi-task learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 7561-7570).
26) Zhang, Y., & Yang, Q. (2021). A survey on multi-task learning. IEEE Transactions on Knowledge and Data Engineering, 34(12), 5586-5609.
27) Zhang, Y., & Yang, Q. (2021). A survey on multi-task learning. IEEE Transactions on Knowledge and Data Engineering, 34(12), 5586-5609.