Concierge: Towards Accuracy-Driven Bandwidth Allocation for Video Analytics Applications in Edge Network

Yuyang Huang*, Faishal Zharfan†, Hendrawan Hendrawan†, Haryadi S. Gunawi*, Junchen Jiang*
*Department of Computer Science
University of Chicago
Chicago, USA
{yuyangh, haryadi, junchenj}@uchicago.edu
†School of Electrical Engineering and Informatics
Bandung Institute of Technology
Bandung, Indonesia
18119002@telecom.stei.itb.ac.id, hend@itb.ac.id

***Abstract*—When performing inference on sensor data, edge video analytics applications may not always need high-fidelity data, since important information may not appear all the time. Consequently, each edge AI application's bandwidth demand is highly dynamic. Thus, a shared edge system should *dynamically* allocate more bandwidth to the applications in need to reach high accuracy at each moment. However, previous bandwidth allocators are ill-suited because they are agnostic to the time-varying impact of bandwidth on each application's accuracy.**

This short paper explores a new *accuracy-driven* approach to bandwidth allocation, which periodically re-allocates bandwidth across edge AI applications based on the *sensitivity* of each application's accuracy to its bandwidth share. To examine its practical benefit and technical challenges, we present a concrete accuracy-driven bandwidth allocator called Concierge, which exposes a simple yet efficient interface to estimate each application's sensitivity to a small change in its bandwidth share.

We run Concierge on state-of-the-art video-analytics applications with real video streams and show its early promise in greatly improving the inference accuracy of video analytics.

***Index Terms*—Resource Allocation, System for ML, ML at Edge, Video Analytic, Network**

I. Introduction

With the wide use of edge sensors in smart cities and intelligent manufacturing, the ability to perform highly accurate inference at edge systems using state-of-the-art deep neural nets (DNNs) has garnered significant interest in both research and industry. This has led to many *video analytics (VA) applications*. These applications use edge sensors (*e.g.* regular camera, LiDAR, infrared-camera, *et al.*) to collect and send the video stream through a shared edge network to edge servers, which run DNN-based inference on the video frames [3–5, 12, 16, 23, 27, 28, 30, 32, 33]. Therefore, the objective of a shared edge system is to *use the limited edge-network bandwidth to stream data at a low latency and enough fidelity such that the edge-side DNNs can obtain high inference accuracy* (defined in §II-A).

Like most network applications, VA applications perform better, *i.e.* are more accurate, with more available bandwidth. Therefore, an ideal edge system should allocate network bandwidth based on each application's *accuracy sensitivity*—how the allocated bandwidth affects the inference accuracy of a VA application (further defined in §II-C). The challenge, however, is that each VA application's accuracy sensitivity often *varies greatly over time*. This is because VA applications, are highly *adaptive to the variation of video content*. They tune a range of configurations including data-coding parameters (*e.g.* [5, 6, 12, 31]), video frame selection (*e.g.* [4, 14, 16]), or DNN architecture (*e.g.* [11, 12, 32]). When a video stream contains many small objects or fast-moving objects, more bandwidth allows an object-detection application to gather video in high resolution and frame rate to detect and track more objects of interest more accurately. In contrast, when the video content becomes simpler, allocating more bandwidth to the VA application might yield similar accuracy. As a result, giving more bandwidth to one VA application than another may lead to higher overall accuracy without increasing latency for both of them [26]. (See Figure 1 for a concrete example.) Therefore, even when all applications have the same priority, the accuracy sensitivity has substantial *temporal variability* for each VA application, and heterogeneity across VA applications.

Unfortunately, existing bandwidth allocators fall short of adapting to the dynamic accuracy sensitivity of each VA application. Traditionally, bandwidth allocators optimize bandwidth utilization rather than inference accuracy. They learn temporal patterns of applications' bandwidth usage [13, 19, 20] and allocate more bandwidth when the application's throughput falls below a threshold [8, 17]. Recently, emerging efforts are taken more directly at resource allocation for AI applications [7, 26, 32]. However, they assume a *static* (offline-profiled) relationship between bandwidth and VA application accuracy, so they are unable to capture the *temporal variability* of AI applications' sensitivity to bandwidth.

In this short paper, we explore a new framework of *accuracy-driven* bandwidth allocation, which *dynamically* allocates bandwidth based on the accuracy sensitivity of each edge application. At a high level, it periodically estimates the accuracy sensitivity of each application and uses the estimates to identify which applications are in need of more bandwidth to achieve higher accuracy.

The key challenge facing accuracy-driven bandwidth allocation is how to estimate accuracy sensitivity, both efficiently (*i.e.*minimum extra compute and communication) and reliably (*i.e.*closely approximate the true accuracy sensitivity). A strawman solution would let VA applications run extra profiling periodically, but this can greatly increase the compute demand of each application (*e.g.*by more than 60% [12]). We tackle this challenge in the context of CONCIERGE, a concrete accuracy-driven bandwidth allocator. The key insight of CONCIERGE is that if a VA application has a high accuracy sensitivity, then it will have significantly different inference outputs on the same input when it is allocated with a different amount of bandwidth. Using this insight, CONCIERGE only requires the application to rerun a small number of inferences. We empirically show that this seemingly simple interaction between the VA application and the bandwidth allocator already reveals enough information to gradually re-allocate more bandwidth to the VA applications with higher accuracy sensitivity. To our best knowledge, CONCIERGE is the first bandwidth allocator to account for both heterogeneity and temporal variability of the accuracy sensitivity of VA applications.

To show the potential of CONCIERGE, we apply it to optimize bandwidth allocation for multiple instances of DDS [5], a representative state-of-art VA application [29]. And We show that CONCIERGE can improve accuracy by upto 7.5% compared to fair allocator and 5.4% when compared to VideoStorm [32] when reallocating bandwidth every 5s.

Taking a step back, CONCIERGE's design is not optimal, but it nonetheless sheds light on the gap between the traditional bandwidth allocator and AI applications such as VA, whose accuracy sensitivity varies over time.

II. MOTIVATION

A. Background

Edge VA applications: We consider a typical edge video-analytics (VA) system in which multiple distributed video sensors share an edge network to transmit the captured video frames to an edge server. Equipped with GPUs, the edge server runs inference on these frames by computer-vision models, such as object detection DNNs, semantic segmentation DNNs, *et al.*. We use *VA application* to refer to running a DNN on the frames sent from one video sensor. Following one of the common designs of VA applications (*e.g.* [29]), we assume that each sensor only encodes and transmits video frames and the DNN inference is performed per frame and is executed on the edge server (no DNN inference on sensors).

Objective of bandwidth allocation: When the edge network has infinite bandwidth, all VA applications can stream their video data at the highest fidelity to perform accurate inference. However, in practice, each application may have a limited bandwidth and must lower data fidelity which affects its *inference accuracy* (or accuracy for briefty). Thus, the goal of an edge network system is to allocate network bandwidth so that *each VA application achieves a high accuracy and has low latency (i.e. no backlog).*[1] Our definitions of accuracy and latency follow recent work in this space (*e.g.* [4, 5, 12, 32]):

- We define an application's *accuracy* as the difference between the DNN inference output under a limited bandwidth share and that under an unlimited bandwidth share.
- We define the *latency* as time elapsed from the moment sensors begin encoding data to the point when the VA applications receive the output from the DNN.

Of course, the accuracy of a VA application is affected by not only the bandwidth allocator but also how the application utilizes its allocated bandwidth. As elaborated later, this interaction between the bandwidth allocator and VA applications is an important feature of our envisioned framework.

B. Related Work

Video-analytics (VA) applications: Numerous prior works have studied VA applications (*e.g.* [4, 5, 11, 12, 14, 16, 29, 32]). A key feature of VA applications is that depending on the available bandwidth and the video content, they can adjust the internal *configurations* such as video resolution and quality parameters (QP) [5, 12], or frame selection [4, 14, 16], or DNN selection [11, 12, 32] to maximize accuracy at a specific target latency. For example, when a VA application has more available bandwidth, it can adjust the video encoding to use a higher resolution, without increasing the latency, and show smaller objects with more details to make inference more accurate. Many techniques have proposed to quickly adapt these configurations, including offline profiling (*e.g.* [29]) or periodically reprofiling (*e.g.* [30]) the relationship between the configuration settings and its inference accuracy, and cheap heuristics to derive a good setting of configuration from input video content (*e.g.* [12]).

Edge bandwidth allocation: Many bandwidth allocators (*e.g.* [2, 9, 10, 13, 18–22, 24, 27, 28]) are designed to dynamically resize each application's allocated bandwidth to make sure that the applications are not under or over-utilizing the bandwidth. There are two general approaches. First, the bandwidth allocators reallocate bandwidth based on the applications' bandwidth utilization [8, 17] or throughput[13, 19, 20]. This could be done by setting a static threshold [8, 17] or recognizing the pattern in the time series of applications' bandwidth utilization. Second, some emerging bandwidth allocators also aim to maximize the inference accuracy of VA applications [7, 26]. They typically construct a static model mapping the relationship between the inference accuracy and the resource consumption based on some datasets containing a large number of videos.

[1]To focus our discussion on bandwidth allocation, we leave the transport-related questions (such as how to deal with packet loss) to future work.

C. Sensitivity of Accuracy to Bandwidth

To see why existing bandwidth allocation schemes are inadequate, we first understand how bandwidth allocation affects a VA application's accuracy.

Temporal variability of accuracy sensitivity: As explained earlier, VA applications adapt to the variations of both available bandwidth and video content. Since video content is not always static, increasing the bandwidth of a VA application could result in greater accuracy improvements at one moment compared to increasing its bandwidth share a few seconds earlier or later. For instance, when the video contains small objects or fast-moving objects, more bandwidth allows an object-detection application to gather video in high resolution and frame rate to detect and track more objects of interest more accurately.

This shows the **temporal variability** of the accuracy's sensitivity to bandwidth—for the same VA application, its accuracy might be *more sensitive* to allocated bandwidth at certain times than others. As different VA applications see different videos and different temporal variation of content, the temporal variability naturally leads to different levels of accuracy improvement when allocating the same amount of bandwidth to different applications. At one time point, VA application A may experience a greater increase in accuracy compared to another B when given the same additional bandwidth. That is, A is more sensitive to changes in allocated bandwidth than B. At another time point, B may be more sensitive to changes in allocated bandwidth than A.

Existing bandwidth allocators fail to capture the temporal variability of how bandwidth affects each application's accuracy. The most related prior works (*e.g.* [32]) do treat different applications differently, but they takes spatial heterogeneity into account while assuming each application has a *static* sensitivity to bandwidth.

Accuracy Sensitivity: To formally capture this intuition, we define the *Accuracy Sensitivity*, or *AccSen* for short, of VA applications with respect to bandwidth as the ratio of the change in the highest achievable inference accuracy (*i.e.*the highest accuracy that can be achieved by the VA application under the currently allocated bandwidth) to the change in the VA applications' allocated bandwidth, δ. This δ can be either negative or positive to determine the VA applications' *AccSen* when we provide more bandwidth (upward *AccSen*) or less bandwidth (downward *AccSen*).

To further illustrate these two properties, we use DDS as an example (a recent VA application that intelligently decides which regions to send in high quality or which do not). Based on the original DDS implementation, we implement DDS+ adding a periodical profiling procedure (*i.e.* a procedure to understand the impact of bandwidth similar to the one proposed in [30]) since the original one does not. With the DDS+, under the 50 Kbps allocated bandwidth and set the fixed target latency, we measure the DDS+'s accuracy sensitivity across two distinct (one highway traffic and one city traffic) traffic videos.

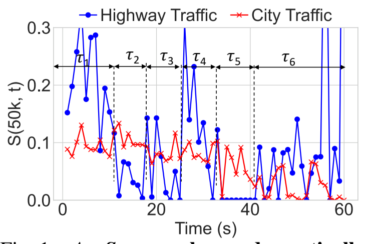

Fig. 1. ***AccSen* can change dramatically across the video.** *The y-axis shows the AccSen of two videos across when bitrates are adjusted to be 50 Kbps and δ to be 10 Kbps.*

We observe that during τ_1, the scene in the highway traffic video contains a handful of vehicles. If the DDS+ on the highway traffic video sends slightly more regions in higher quality (resulting in a small increase in video size), even though it leads to a small increase in the number of true positive bounding boxes, the accuracy can be significantly improved since we have a small denominator when calculating the accuracy. But with a large number of vehicles in the city traffic scene, the other DDS+ needs to encode significantly more regions in high quality (much higher bandwidth consumption) to achieve a similar accuracy improvement. Consequently, the DDS+ processing of the highway traffic video exhibits higher sensitivity in comparison to one of the city traffic videos, highlighting the spatial heterogeneity across different VA applications.

However, at a different time, τ_2, the video content changes, and the highway traffic contains no vehicles. In this case, regardless of the amount of bandwidth allocated to the DDS+, the accuracy remains as one (since there are no objects to detect), leading to zero accuracy sensitivity. As a result, the sensitivity of the DDS+ analyzing the highway traffic video falls below the sensitivity of the one on the city traffic video, illustrating temporal variability within the same VA application.

III. Design Principal

Overview: The discussion so far has shown a strong need for a better bandwidth allocation framework that is driven by tempoeral variability of the accuracy sensitivity (*AccSen*) of each edge application. We present the design principal of a *AccSen*-driven dynamic bandwidth allocator. We envision it consists of three logical components:

- Interface allowing the communication between the bandwidth allocator and VA applications,
- Logic to calculate *AccSen* based on the information the bandwidth allocator obtains from the application interface,
- Logic to determine the new bandwidth allocation across VA applications.

Requirements: We view such bandwidth allocator should meet the following three requirements:

- **RQ1:** It should periodically update its knowledge of the *AccSen* for the video analytics (VA) applications, as the *AccSen* can vary with video content, as shown in §II-C,

- **RQ2:** It should not introduce significant overhead for the VA applications, ensuring that they can maintain real-time analysis for most of the time,
- **RQ3:** It should only require VA applications to expose minimal necessary information (*i.e.*grey box modeling).

Key Challenge: A challenge facing accuracy-driven bandwidth allocation is how to periodically update *AccSen*.

A naïve design of CONCIERGE that meets **RQ1** could periodically request VA applications to calculate inference accuracies under two allocated bandwidths and determine the difference between these accuracies to obtain *AccSen*. However, this approach fails to satisfy **RQ2**. During runtime, the ground truth is often obtained by running inference on lossless videos using the most computationally expensive DNN model by convention [5, 30]. This process can be extremely costly, as the size of lossless videos can be 10x to 100x larger than their encoded and compressed counterparts, resulting in substantial end-to-end delay overhead for the VA applications. Therefore, the key to meeting **RQ2** lies in obtaining *AccSen* with reasonable overhead.

IV. CONCIERGE: ACCURACY-DRIVEN BANDWIDTH ALLOCATION

To show that accuracy-driven bandwidth allocation is feasible with reasonable overhead in practice, we now present the design of CONCIERGE, a prototype of accuracy-driven bandwidth allocator.

A. *AccSen Estimator*

Assumptions: For the CONCIERGE to work correctly, we need to first make a few reasonable assumptions on VA applications. First, each VA application has a fixed latency target (*e.g.* no backlog), and its goal is to maximize the total overall accuracy over application (*i.e.* the relative importance among applications is pre-determiend). Second, when the allocator changes the available bandwidth of an application, it will explicitly inform the application, *i.e.* each application does not need to use probe bandwidth by themselves. Third, each application can adapt its internal configurations when available bandwidth changes, as explained in §II-A.

Key insight behind *AccSen* estimation: While it seems impractical to periodically update the *AccSen* at run-time, we can still *estimate AccSen* based on a crucial insight: a VA application with high accuracy sensitivity will have significantly different inference results on the same input when it is allocated with a different amount of bandwidth. Effectively, this suggests that the accuracy sensitivity of a VA application can be estimated by letting the application run two inferences on the same input under two different amounts of bandwidth.

To illustrate this insight, we use three 5-second video segments obtained by using the "traffic video" and "drone video" as the keywords to search YouTube in incognito mode (following the practice in [5]). For each video segment, we

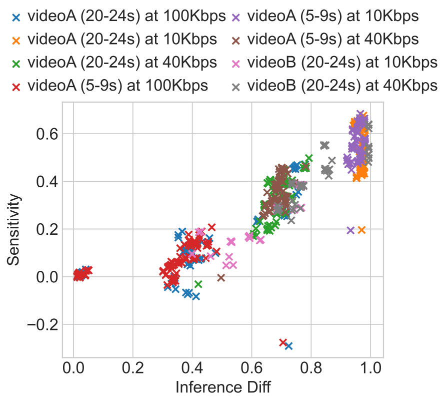

Fig. 2. ***InferDiff* correlates linearly with *AccSen* for various videos at different allocated bandwidths.**

run inference using the DDS+ with two levels of allocated bandwidth, A and B where $B > A$, and calculate the difference between their actual inference *accuracies* and the difference between their inference *outputs*, defined by the F1 score of the results under A with the result under B as the ground truth. Thus, the accuracy sensitivity *AccSen*, $S_i^{B-A}(A,t)$, is the former when divided by $B-A$, and we define the latter when divided by $B-A$ as the *inference difference* or *InferDiff*, denoted by $\mathcal{D}(A,B,t)$. Similar to *AccSen*, depending on the magnitude of B, there is a upward *InferDiff* when $B > A$ and downward *InferDiff* when $B < A$.

Figure 2 shows that for these video segments under test three choices of A and varying B, there is a decent correlation between accuracy sensitivity *AccSen*, $S_i^{B-A}(A,t)$ and *InferDiff*, $\mathcal{D}(A,B,t)$. We do not claim that this correlation will be universally true (for instance, when bandwidth is too low, the output between two bandwidth levels may be equally bad, while *InferDiff* can still be non-negligible). That said, the correlation makes intuitive sense and these empirical results suggest a promising path to the cheap estimation of *AccSen*.

B. *InferDiff-based Bandwidth Allocation*

Now, if we ignore **RQ3** for now and assume that VA application i can (1) store the current inference results under the allocated bandwidth A at video time t, (2) identify the optimal configuration under the new bandwidth limit B, (3) execute the inference on time t again using this new configuration, and (4) compute the *InferDiff* $\mathcal{D}_i(A,B,t)$, which correlates with the VA application's downward accuracy sensitivity (*AccSen*) at the current bandwidth allocation (and this process needs to repeat for the upward *AccSen*). CONCIERGE can maximize total accuracy change

$$\sum_i \Delta BW_i \times \left(\mathbf{1}_{\Delta BW_i>0}\mathcal{D}_i^{+}(A,B,t) + \mathbf{1}_{\Delta BW_i<0}\mathcal{D}_i^{-}(A,B,t)\right),$$

where the $\mathcal{D}_i^{\pm}(A,B,t)$ is the upward or downward *InferDiff* and $\mathbf{1}$ is the indicator function. Hence, CONCIERGE only

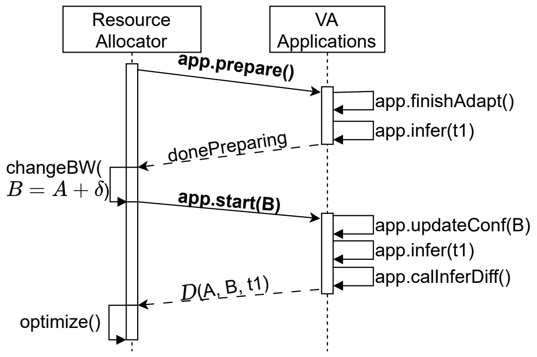

Fig. 3. **CONCIERGE's Interface.** *Workflow of the interaction between a VA application and* CONCIERGE. *The* ***functions in bold*** *indicate the public API the VA applications need to implement. Other functions are either provided by* CONCIERGE *or has been implemented by VA applications even if they do not use* CONCIERGE.

requires VA applications to run two extra inferences on the current video segment it is already running.

C. Allocator-Application Interface

With **RQ3** (*i.e.*applications only expose minimal required information) in mind, we present the final design of CONCIERGE. To integrate into CONCIERGE, the VA applications need to inherit an abstract class, App, and implement the consisted 2 public abstract API.

- App.prepare(): This interface should be implemented by the VA application. Upon return, the VA application guarantees it 1) has no running adaption logic, and 2) runs and stored the best achievable inference results as well as the timestamp for content it runs.
- App.start(newBW): This interface should be implemented by the VA application. This interface is called by CONCIERGE to notify the new bandwidth limit. In this API, the VA application will change its internal configuration according to the mapping obtained during the App.prepare(), analyze the video at the stored timestamp with the new configuration, and return the *InferDiff* between the two inference results. The function to calculate the *InferDiff*, calInferDiff(), is implemented by CONCIERGE and provided as a private function in the App abstract class for the VA applications to call as needed.

By this design, the VA application only exposes the *InferDiff* to the CONCIERGE, and all the video content, as well as the inference results, are stored locally in the VA applications satisfying **RQ3**.

Figure 3 visualizes how CONCIERGE works at run-time, ensuring the order to two control loops and keeping the information exchanged minimal.

V. PRELIMINARY EVALUATION

A. Experiment Setup

Baselines: To understand the performance of CONCIERGE, we compare it to two baselines. First, we compare it with a fair allocator, which allocates the available network bandwidth evenly across each VA application. Second, we compare CONCIERGE with an existing scheme called VideoStorm [32], which relies on the resource-accuracy trade-off of each type of VA application before allocating resources to them. Hence, VideoStorm is more intelligent than accuracy-agnostic allocators like the fair allocator.
Application: We use DDS+ [5] introduced in §II-A as the VA applications in the experiments. We run total of 3 instances of DDS+ and all instances run object detection as their VA tasks. We configure the applications' target latency to be such that the applications should finish the current video segment before the next one is received by the server-side, i.e., with no backlogging. The overhead brought by CONCIERGE will also be included in the target latency. This means that when applications are required to profile for *InferDiff*, if CONCIERGE brings significant overhead, the latency budget for the current video segment will decrease significantly, eventually reflected in a possible decrease in transmission quality and the accuracy of such video segment.

At runtime, CONCIERGE will start with fair allocation and re-allocate bandwidth every 5 seconds by default unless specified otherwise. We use the F1 score to evaluate applications' accuracy, which is a common metric for evaluating object detection tasks.
Dataset: Following the prior works' conventions [5, 30], we use keywords like "security camera", "traffic camera", *et al.*. to randomly collect the videos on YouTube to form our evaluation dataset. To further understand the performance of CONCIERGE under various settings, we categorize the collected videos based on their temporal variabilities in *AccSen* using heuristics like the number of objects or the size of the objects. We use *Type I* to denote videos with relatively high temporal variabilities and *Type II* to denote the others.
Hardware setup: The server-side of all the applications will be situated on the same machine equipped with an Nvidia RTX6000 GPU [1], while their client-sides will be hosted on another CPU-only machine. We ensure that network bandwidth is the sole bottleneck between the servers' and clients' machines. We constrain the network bandwidth between the servers' and clients' machines using the Linux *tc* [25] command to regulate the ingress traffic on the server's listening port for each application.

B. Experiment Results

Experiment #1: In this experiment, we evaluate the performance of CONCIERGE when handling three DDS+ applications running on videos with different temporal variabilities in *AccSen*. Hence, for each run, when we randomly decide which videos each application will run, we make sure at least one Type I and one Type II video are presented in the combination. The results in Figure 4 show that CONCIERGE consistently outperforms both of the baselines in a range of

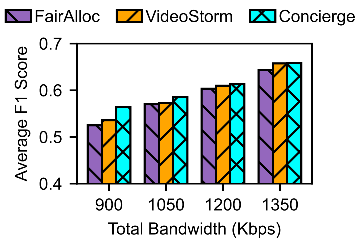

Fig. 4. **CONCIERGE performance when *AccSen*'s temporal variabilities are different across VA applications.** *The seemingly marginal accuracy improvement is de facto non-trivial. This could means more objects are correctly detected or, with the same detected objects, more objects are correctly labeled.*

total bandwidths, with up to 7.5% improvement in F1 score compared to fair allocator and 5.4% improvement compared to VideoStorm at 900Kbps total bandwidth.

We also observe that the improvement diminishes as we increase the total bandwidth. However, this is expected since the applications' sensitivity will drop and eventually reach zero at high bandwidth (e.g., there is enough bandwidth such that all applications can use their highest configurations). Hence, this leaves limited space for CONCIERGE to improve as all the applications already reach the highest accuracy they can achieve.

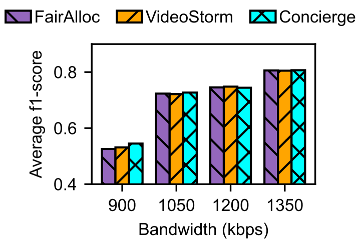

Fig. 5. **CONCIERGE performance when VA applications have similar temporal varibility in *AccSen*.** *The experiment indicates the expected behavior and results for* CONCIERGE *as the temproal variabilities are similar across videos.*

Experiment #2: In this experiment, we test the performance of CONCIERGE when the temporal variabilities in sensitivity are similar across all three DDS+. Hence, when we randomly assign videos, we make sure the videos are either all Type I or all Type II. As shown in Figure 5, CONCIERGE achieves similar performance compared to VideoStorm even when started with a fair allocation but later gradually reached the optimal.

Experiment #3: In this experiment, we evaluate the effectiveness of *InferDiff* by comparing the performance of CONCIERGE when using *InferDiff* and that when using *AccSen*. In order to obtain *AccSen* at runtime, we provide the ground truth of the inferring videos to applications (*i.e.* oracles) so that they can calculate the accuracy and therefore *AccSen* with overhead comparable to calculating *InferDiff*. Results shown in Figure 6 indicates that CONCIERGE using *InferDiff* achieves similar accuracy compared with one leverage *AccSen*. This further proves the effectiveness of *InferDiff* in estimating *AccSen* with reasonable overhead.

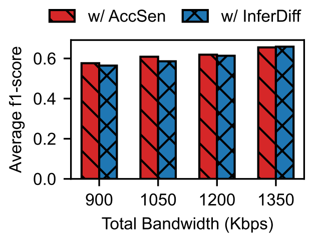

Fig. 6. **CONCIERGE's performance comparison when using *InferDiff* and *AccSen* (oracle).** *The experiment shows the effectiveness of the InferDiff as in that* CONCIERGE *with InferDiff produces the similar F1 score compared to the* CONCIERGE *using AccSen with ground truth provided (i.e., oracle).*

VI. LIMITATION AND ASSUMPTION OF CONCIERGE

CONCIERGE as-is has the following limitations, which will be addressed in future work:

- CONCIERGE considers the VA applications have a fixed target latency such that they only make trade-off between bandwidth consumption and accuracy.
- CONCIERGE will only work if the VA application is self-adaptive and be able to tune internal parameters according to the change of video content at runtime. If not, for the same video content, CONCIERGE will estimate the *InferDiff* to be zero for such VA application since the internal parameters and the resulting inference output do not change.
- CONCIERGE does not have a self-correct logic hence assuming the VA applications will use the provided function to calculate the *InferDiff* and will not attempt to give false information about the *InferDiff* (*i.e.* no VA application will lie about its *InferDiff* to obtain more allocated bandwidth).

VII. CONCLUSION

As video analytics is widely used, this short paper argues that the recent self-adaptive and temporal varying VA applications pose significant challenges to prior bandwidth allocation systems which focus largely on applications' throughput or is based on a static understanding of allocated bandwidth's impact on the VA applications. In response, we propose a new bandwidth allocation framework capable of updating the estimation of VA applications' accuracy sensitivity online to capture the temporal variability of VA applications. The preliminary results show that it greatly improves the accuracy when multiple VA applications share limited bandwidth with reasonable overhead.

VIII. ACKNOWLEDGMENTS

We thank the anonymous reviewers for their tremendous feedback and comments. This material was supported by funding from NSF (grant Nos. CCF-2119184, CNS-2027170, CNS-2146496, CNS-1901466, CNS-2313190, and CNS-2131826). The experiments in this paper were performed in Chameleon [15] testbed supported under NSF grants Nos. CNS-2027170.

References

[1] Nvidia RTX 6000 GPU. Online, 2018. Accessed on June 11, 2023.

[2] Yahya Al-Dhuraibi, Fawaz Paraiso, Nabil Djarallah, and Philippe Merle. Elasticity in cloud computing: state of the art and research challenges. *IEEE Transactions on Services Computing*, 11(2):430–447, 2017.

[3] Romil Bhardwaj, Zhengxu Xia, Ganesh Ananthanarayanan, Junchen Jiang, Nikolaos Karianakis, Yuanchao Shu, Kevin Hsieh, Victor Bahl, and Ion Stoica. Ekya: Continuous learning of video analytics models on edge compute servers, 2020.

[4] Tiffany Yu-Han Chen, Lenin Ravindranath, Shuo Deng, Paramvir Bahl, and Hari Balakrishnan. Glimpse: Continuous, real-time object recognition on mobile devices. In *Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems*, pages 155–168, 2015.

[5] Kuntai Du, Ahsan Pervaiz, Xin Yuan, Aakanksha Chowdhery, Qizheng Zhang, Henry Hoffmann, and Junchen Jiang. Server-driven video streaming for deep learning inference. In *Proceedings of the Annual conference of the ACM Special Interest Group on Data Communication on the applications, technologies, architectures, and protocols for computer communication*, pages 557–570, 2020.

[6] Kuntai Du, Qizheng Zhang, Anton Arapin, Haodong Wang, Zhengxu Xia, and Junchen Jiang. Accmpeg: Optimizing video encoding for accurate video analytics. *Proceedings of Machine Learning and Systems*, 4:450–466, 2022.

[7] Apostolos Galanopoulos, Jose A Ayala-Romero, Douglas J Leith, and George Iosifidis. Automl for video analytics with edge computing. In *IEEE INFOCOM 2021-IEEE Conference on Computer Communications*, pages 1–10. IEEE, 2021.

[8] Hamoun Ghanbari, Bradley Simmons, Marin Litoiu, and Gabriel Iszlai. Exploring alternative approaches to implement an elasticity policy. In *2011 IEEE 4th International Conference on Cloud Computing*, pages 716–723. IEEE, 2011.

[9] Zhenhuan Gong, Xiaohui Gu, and John Wilkes. Press: Predictive elastic resource scaling for cloud systems. In *2010 International Conference on Network and Service Management*, pages 9–16. Ieee, 2010.

[10] Judy Hoffman, Sergio Guadarrama, Eric S Tzeng, Ronghang Hu, Jeff Donahue, Ross Girshick, Trevor Darrell, and Kate Saenko. Lsda: Large scale detection through adaptation. *Advances in neural information processing systems*, 27, 2014.

[11] Kevin Hsieh, Ganesh Ananthanarayanan, Peter Bodik, Shivaram Venkataraman, Paramvir Bahl, Matthai Philipose, Phillip B Gibbons, and Onur Mutlu. Focus: Querying large video datasets with low latency and low cost. In *13th USENIX Symposium on Operating Systems Design and Implementation (OSDI 18)*, pages 269–286, 2018.

[12] Junchen Jiang, Ganesh Ananthanarayanan, Peter Bodik, Siddhartha Sen, and Ion Stoica. Chameleon: scalable adaptation of video analytics. In *Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication*, pages 253–266, 2018.

[13] Vasiliki Kalavri, John Liagouris, Moritz Hoffmann, Desislava Dimitrova, Matthew Forshaw, and Timothy Roscoe. Three steps is all you need: fast, accurate, automatic scaling decisions for distributed streaming dataflows. In *13th USENIX Symposium on Operating Systems Design and Implementation (OSDI 18)*, pages 783–798, 2018.

[14] Daniel Kang, John Emmons, Firas Abuzaid, Peter Bailis, and Matei Zaharia. Noscope: optimizing neural network queries over video at scale. *arXiv preprint arXiv:1703.02529*, 2017.

[15] Kate Keahey, Jason Anderson, Zhuo Zhen, Pierre Riteau, Paul Ruth, Dan Stanzione, Mert Cevik, Jacob Colleran, Haryadi S Gunawi, Cody Hammock, et al. Lessons learned from the chameleon testbed. In *2020 USENIX annual technical conference (USENIX ATC 20)*, pages 219–233, 2020.

[16] Yuanqi Li, Arthi Padmanabhan, Pengzhan Zhao, Yufei Wang, Guoqing Harry Xu, and Ravi Netravali. Reducto: On-camera filtering for resource-efficient real-time video analytics. In *Proceedings of the Annual conference of the ACM Special Interest Group on Data Communication on the applications, technologies, architectures, and protocols for computer communication*, pages 359–376, 2020.

[17] Tania Lorido-Botrán, José Miguel-Alonso, and Jose Antonio Lozano. Auto-scaling techniques for elastic applications in cloud environments. *Department of Computer Architecture and Technology, University of Basque Country, Tech. Rep. EHU-KAT-IK-09*, 12:2012, 2012.

[18] Michael Maurer, Ivona Brandic, and Rizos Sakellariou. Enacting slas in clouds using rules. In *European Conference on Parallel Processing*, pages 455–466. Springer, 2011.

[19] Hiep Nguyen, Zhiming Shen, Xiaohui Gu, Sethuraman Subbiah, and John Wilkes. {AGILE}: Elastic distributed resource scaling for {Infrastructure-as-a-Service}. In *10th International Conference on Autonomic Computing (ICAC 13)*, pages 69–82, 2013.

[20] Haoran Qiu, Subho S Banerjee, Saurabh Jha, Zbigniew T Kalbarczyk, and Ravishankar K Iyer. {FIRM}: An intelligent fine-grained resource management framework for {SLO-Oriented} microservices. In *14th USENIX Symposium on Operating Systems Design and Implementation (OSDI 20)*, pages 805–825, 2020.

[21] Jia Rao, Xiangping Bu, Cheng-Zhong Xu, Leyi Wang, and George Yin. Vconf: a reinforcement learning approach to virtual machines auto-configuration. In *Proceedings of the 6th international conference on Autonomic computing*, pages 137–146, 2009.

[22] Krzysztof Rzadca, Pawel Findeisen, Jacek Swiderski, Przemyslaw Zych, Przemyslaw Broniek, Jarek Kusmierek, Pawel Nowak, Beata Strack, Piotr Witusowski, Steven Hand, et al. Autopilot: workload autoscaling at google. In *Proceedings of the Fifteenth European Conference on Computer Systems*, pages 1–16, 2020.

[23] Haichen Shen, Lequn Chen, Yuchen Jin, Liangyu Zhao, Bingyu Kong, Matthai Philipose, Arvind Krishnamurthy, and Ravi Sundaram. Nexus: A gpu cluster engine for accelerating dnn-based video analysis. In *Proceedings of the 27th ACM Symposium on Operating Systems Principles*, pages 322–337, 2019.

[24] Zhiming Shen, Sethuraman Subbiah, Xiaohui Gu, and John Wilkes. Cloudscale: elastic resource scaling for multi-tenant cloud systems. In *Proceedings of the 2nd ACM Symposium on Cloud Computing*, pages 1–14, 2011.

[25] The Linux Documentation Project. *tc - tool for controlling traffic in Linux*. The Linux Documentation Project, 2021.

[26] Can Wang, Sheng Zhang, Yu Chen, Zhuzhong Qian, Jie Wu, and Mingjun Xiao. Joint configuration adaptation and bandwidth allocation for edge-based real-time video analytics. In *IEEE INFOCOM 2020-IEEE Conference on Computer Communications*, pages 257–266. IEEE, 2020.

[27] Yiding Wang, Weiyan Wang, Junxue Zhang, Junchen Jiang, and Kai Chen. Bridging the {Edge-Cloud} barrier for real-time advanced vision analytics. In *11th USENIX Workshop on Hot Topics in Cloud Computing (HotCloud 19)*, 2019.

[28] Zhujun Xiao, Zhengxu Xia, Haitao Zheng, Ben Y Zhao, and Junchen Jiang. Towards performance clarity of edge video

analytics. *arXiv preprint arXiv:2105.08694*, 2021.

[29] Zhujun Xiao, Zhengxu Xia, Haitao Zheng, Ben Y Zhao, and Junchen Jiang. Towards performance clarity of edge video analytics. In *2021 IEEE/ACM Symposium on Edge Computing (SEC)*, pages 148–164. IEEE, 2021.

[30] Ben Zhang, Xin Jin, Sylvia Ratnasamy, John Wawrzynek, and Edward A Lee. Awstream: Adaptive wide-area streaming analytics. In *Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication*, pages 236–252, 2018.

[31] Dongmei Zhang, Huadong Ma, Liang Liu, and Dan Tao. Eaar: An approach to environment adaptive application reconfiguration in sensor network. In *Mobile Ad-hoc and Sensor Networks: First International Conference, MSN 2005, Wuhan, China, December 13-15, 2005. Proceedings 1*, pages 259–268. Springer, 2005.

[32] Haoyu Zhang, Ganesh Ananthanarayanan, Peter Bodik, Matthai Philipose, Paramvir Bahl, and Michael J Freedman. Live video analytics at scale with approximation and {Delay-Tolerance}. In *14th USENIX Symposium on Networked Systems Design and Implementation (NSDI 17)*, pages 377–392, 2017.

[33] Tan Zhang, Aakanksha Chowdhery, Paramvir Bahl, Kyle Jamieson, and Suman Banerjee. The design and implementation of a wireless video surveillance system. In *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking*, pages 426–438, 2015.