This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCC.2022.3142066, IEEE Transactions on Cloud Computing

Enabling Edge-Cloud Video Analytics for Robotics Applications

Yiding Wang[1], Weiyan Wang[1], Duowen Liu[1], Xin Jin[2], Junchen Jiang[3], Kai Chen[1]
[1]iSING Lab, Hong Kong University of Science and Technology
[2]Peking University [3]University of Chicago

Abstract—Emerging deep learning-based video analytics tasks demand computation-intensive neural networks and powerful computing resources on the cloud to achieve high accuracy. Due to the latency requirement and limited network bandwidth, edge-cloud systems adaptively compress the data to strike a balance between overall analytics accuracy and bandwidth consumption. However, the degraded data leads to another issue of poor *tail accuracy*, which means the extremely low accuracy of a few semantic classes and video frames. Autonomous robotics applications especially value the tail accuracy performance but suffer using the prior edge-cloud systems. We present Runespoor, an edge-cloud video analytics system to manage the tail accuracy and enable emerging robotics applications. We train and deploy a super-resolution model tailored for the tail accuracy of analytics tasks on the server to significantly improves the performance on hard-to-detect classes and sophisticated frames. During online operation, we use an adaptive data rate controller to further improve the tail performance by instantly adjusting the data rate policy according to the video content. Our evaluation shows that Runespoor improves class-wise tail accuracy by up to 300%, frame-wise 90%/99% tail accuracy by up to 22%/54%, and greatly improves the overall accuracy and bandwidth trade-off.

Index Terms—Cloud Computing, Super Resolution, Video Analytics.

1 Introduction

Deep learning-based video analytics applications are flourishing and powering a growing number of traditionally challenging applications for scene understanding. Such applications adopt computer vision (CV) techniques including multiple object detection [1], semantic segmentation [2], instance segmentation [3] and panoptic segmentation [4]. To obtain adequate inference accuracy, these tasks often require (i) computation-intensive deep learning (DL) models, (ii) powerful computation resources, and (iii) high-fidelity data. Also, to enable responsive and high-precision robotics applications such as autonomous vehicles and unmanned aerial vehicles (UAVs), video data should have both a high frame rate and high resolution.

Let us start with an example of robotics applications that can benefit from edge-cloud computing: autonomous delivery vehicles [5]. They are small-sized electric vehicles and designed to be deployed on a large scale. But during the COVID-19 lockdown when they are needed the most, "the technology is not ready at scale to deploy", said the president of a leading startup, and the service fee could be higher than delivery riders [6]. This is because unlike high-end autonomous cars that install expensive hardware to run complex DL inference locally [7], they are budget and battery constrained [8]. They run at slower speeds on sidewalks or bike lanes, and are monitored remotely by operators via cellular video streams and intervened if necessary [9], thus do not have a latency requirement as stringent as high-end autonomous cars. Nevertheless, they use the same fundamental technologies: autonomous delivery vehicles also heavily utilize cameras and deep neural networks to understand the surroundings for planning and control [10], [11]. Currently, autonomous delivery vehicles run DL video inference locally as full-size cars, which could raise some concerns for the future *performance* and *scalability*: as they perform more challenging tasks, DL computation also increases, so does the requirement for computation resources; in the meantime, they still need to keep the compactness and cost under budget to scale.

Off-loading the heavy DL inference tasks to the cloud is a reasonable and promising solution: it can significantly reduce the hardware requirements of edge devices, enable these applications to be deployed at scale at a reasonable cost, and reuse the bandwidth for remote monitoring. The cloud (datacenters and edge clusters) has abundant compute and network resources to provide high accuracy for fast video analytics tasks [12], [13], [14], [15], [16]. However, the downside of cloud is also apparent: the limited wide-area bandwidth between the cloud and the edge could cause long transmission delay [17] when streaming high-quality video or inferior inference performance with low resolution or frame rate. It is critical for autonomous delivery vehicles to make decisions fluently and identify obstacles on the road correctly. There is a trade-off between overall accuracy and bandwidth consumption [17], [18], [19].

However, we find that managing this classic trade-off is not sufficient to enable edge-cloud robotics applications since it tends to cause poor *tail accuracy* in practice. Although tuning degradation knobs (e.g., reducing video resolution) can balance bandwidth consumption and overall accuracy to some extent, it will also cause poor *class-wise tail accuracy* (i.e., low accuracy of specific semantic classes like traffic lights and motorcycles) and *frame-wise tail accuracy* (i.e., a few frames with extremely low accuracy over a period of time). This is because video data degradation will lead to the

2168-7161 (c) 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: Hong Kong University of Science and Technology. Downloaded on February 25,2022 at 02:00:33 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCC.2022.3142066, IEEE Transactions on Cloud Computing

loss of detailed information, thus hurt the accuracy of these hard-to-detect small regions (e.g., traffic signs and riders rather than sky and buildings) and sophisticated frames [20], [21], which leads to poor tail accuracy. Issues caused by poor tail accuracy, including the mislabeling of specific classes and short periods of low accuracy inference, would hurt real-world application performance and operations [22], [23]. However, in a conventional bandwidth-accuracy trade-off, the overall accuracy cannot reflect the low tail accuracy, because tail accuracy is covered by well-classified classes which are robust to data degradation (class-wise) and by the long periods of good operations (frame-wise) (§ 2.2).

Motivated by robotics applications, we argue that for such edge-cloud video analytics systems, the real challenge is the *three-way trade-off* between bandwidth consumption, overall accuracy, and especially *tail accuracy*. Runespoor addresses the tail accuracy challenge with two designs at both cloud and edge: (i) analytics-aware super-resolution (ASR) and (ii) content-aware adaptive controller (CAC).

Analytics-aware super-resolution extends super-resolution (SR), which is an effective DL technique that learns a mapping from low-resolution frames to high-resolution frames. We use the DL models for CV tasks to train ASR to reconstruct high-resolution frames tailored for the tail accuracy performance of video analytics tasks with augmented details from compressed low-resolution data on the server. ASR explicitly considers the DL inference performance in tail-related regions as the learning signals in addition to the conventional reconstruction similarity target (e.g., PSNR [24] and SSIM [25]) of SR. In this way, ASR significantly improves the accuracy of detailed regions, thus improves the accuracy of hard-to-detect classes and sophisticated frames (§ 3.1).

Content-aware adaptive controller specifically improves the inference accuracy of sophisticated frames that heavily relies on detailed information by sending specialized higher-resolution frames. In edge-cloud systems, reducing the data rate regarding the bandwidth constraints leads to the loss of detailed information. Moreover, the fast-changing scenes in robotics applications make it challenging to detect and fix tail accuracy during the online operation. Based on our tail-aware offline profiling, CAC learns to detect the frames that lead to low tail accuracy using the inference results of CV tasks and instantly decides the data configuration for subsequent frames to improve the frame-wise tail accuracy without extra DL computation or network overhead (§ 3.2).

We implement and evaluate Runespoor on two CV tasks for modern robotics applications: (i) road-driving semantic segmentation and (ii) drone-view object detection. Our evaluation shows that Runespoor significantly outperforms prior work on tail accuracy. Runespoor improves frame-wise 90% and 99% accuracy by 18%-22% and 35%-54% for semantic segmentation, respectively. Runespoor improves class-wise 50% to 100% accuracy averagely by 0.9%-79.4% for semantic segmentation, and 14%-300% for object detection. Runespoor with CAC is an efficient online end-to-end system that adapts to real-world changing scenes. Runespoor also improves overall accuracy and saves bandwidth consumption.

Our contributions are: (i) exposing and defining the important tail accuracy problem in edge-cloud video analytics; (ii) analytics-aware super-resolution (ASR) that fixes tail accuracy by focusing on detailed information reconstruction; (iii) content-aware adaptive controller (CAC) that adapts to fast-changing scenes with DL outputs in an end-to-end system.

This paper extends an earlier version which has been published in IEEE INFOCOM 2021 [26].

2 Motivation

Runespoor is motivated by video analytics tasks for emerging robotics CV applications, e.g., autonomous delivery robots and UAVs. They use fresh videos and are deployed on a large scale in complex scenes. They are equipped with high-definition cameras and require scene understanding capabilities for planning and control. They require powerful DL models, computation resources, and high-quality data [27] to ensure high inference accuracy and fast reaction. Due to limited computation resources at the edge, many applications send videos to the cloud to run heavy DL inference tasks [12], [17], [18]. State-of-the-art video analytics systems work efficiently on general tasks, but are limited for our target applications (§ 2.4).

We find that although current video degradation techniques in edge-cloud analytics systems (e.g., AWStream [17] and NAS [28]) can reduce the bandwidth consumption and maintain a relatively high overall inference accuracy, e.g., mIoU (mean of intersection over union) for semantic segmentation and mAP (mean average precision) for object detection, they cannot fix the poor *tail accuracy* issue, which is caused by data degradation and the requirement of detailed information in CV tasks, as further explained in § 2.2.

2.1 What is tail accuracy?

We conduct preliminary evaluations on data reconstruction methods in previous edge-cloud video analytics systems and find that the tail accuracy has two forms: (i) *class-wise* tail accuracy and (ii) *frame-wise* tail accuracy.

Class-wise tail accuracy refers to the huge accuracy loss of some semantic classes in the scene understanding DL inference tasks for scene understanding (e.g., semantic segmentation and object detection) on degraded data.

Let us illustrate class-wise tail accuracy with an example: under bandwidth constraints, the edge device (e.g., an autonomous delivery robot) sends downsampled 512×256 frames (from Cityscapes [27] validation set) to the cloud for semantic segmentation inference. Before inference, to fit the input resolution of the DL model, frames are upsampled to the original 2048×1024 resolution by (i) Bilinear algorithm and (ii) the state-of-the-art SR model [29]. Bilinear algorithm is the default upsampling method in major ML frameworks (e.g., TensorFlow and PyTorch [30]). SR proves useful in video systems [28], [31], [32]. In Figure 1, the class-wise accuracy is normalized to the fraction of the inference accuracy on original high-resolution frames, because we want to compare the accuracy loss caused by data degradation.

As expected in Figure 1, upsampling the compressed frames achieves higher inference accuracy with the same received data and bandwidth consumption. What is more, there exists significant class-wise accuracy inequality on all three series. The accuracies of classes like road, wall, sky,

2168-7161 (c) 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: Hong Kong University of Science and Technology. Downloaded on February 25,2022 at 02:00:33 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCC.2022.3142066, IEEE Transactions on Cloud Computing

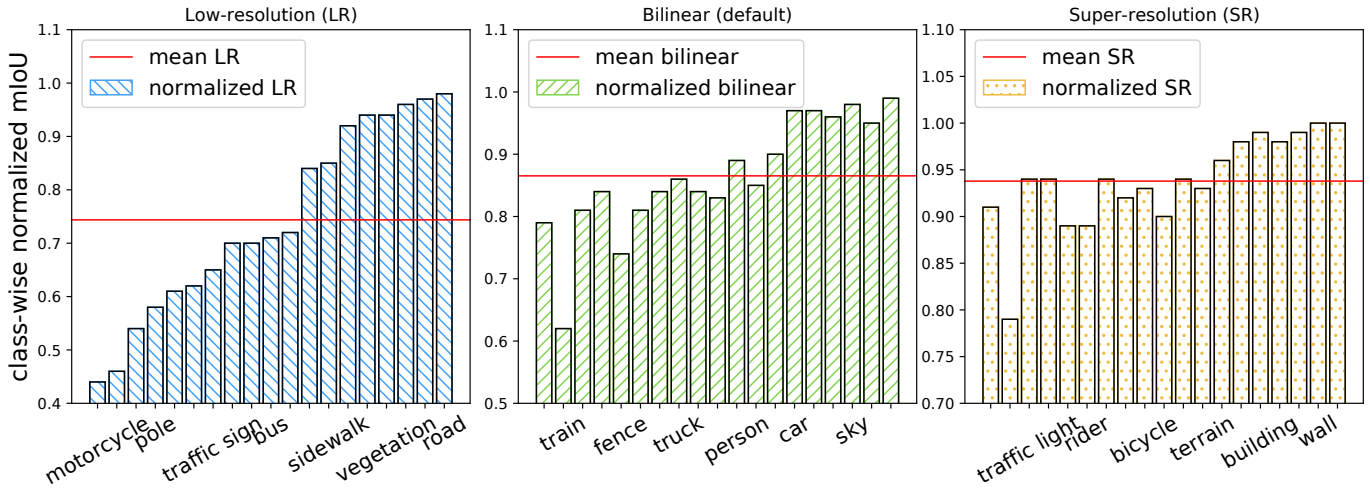


Fig. 1: Normalized semantic segmentation class-wise accuracy after compression. The normalization baseline is the accuracy on the original data. Bilinear (mid) and super-resolution (right) still obtain degraded tail accuracy, although they improve overall accuracy. Subfigures have different Y-axises. The 19 X-axis class labels are printed in Round-Robin for space limitation. (Complete ordered labels please see Figure 9.)

and buildings are higher than the overall accuracy (mIoU, three red lines) and close to 1, which means almost no loss on degraded data. However, other classes (e.g., motorcycles, bicycles, riders, traffic lights/signs, and buses) suffer from much more accuracy loss. Overall accuracy (e.g., mIoU and mAP) does not reflect the huge accuracy loss of those classes (tail), because most regions and classes are robust to the video resolution degradation (e.g., sky, road, cars, and buildings).

Frame-wise tail accuracy refers to the extremely low tail accuracy over a temporal series of video frames in inference tasks. Emerging applications such as autonomous vehicles require fluent decision-making. Every frame matters for such applications, but some frames suffer from more accuracy loss on degraded data. Figure 2 uses the same setting as Figure 1. It shows the accuracy distribution of frames that are processed by Bilinear algorithm and the standard analytics-agnostic SR model. The accuracy of the hardest frames (90% and 99% accuracy) can be 19% to 37% worse than the mean accuracy of frames. The accuracy distribution indicates that during a period of operation, some frames can suffer from significantly low accuracy, which is not pleasant for applications. However, the overall accuracy of all frames over a long period of time cannot capture the tail accuracy.

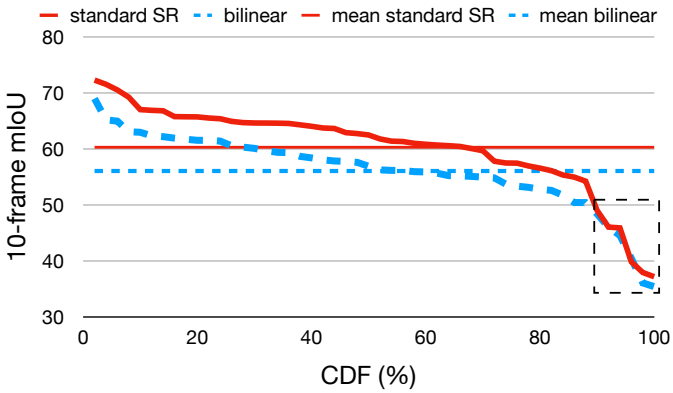


Fig. 2: Frame-wise accuracy distribution (averaged on every 10 frames). Using SR does substantially improve the overall accuracy over Bilinear (two horizontal lines), but they both suffer from poor *tail accuracy* (the highlighted 90+ percentiles).

2.2 Why does tail accuracy exist?

The fundamental cause of the poor tail accuracy is that the detailed information, including small regions/objects and boundaries, is more sensitive to the degraded input data (downsampled frames) in DL inference tasks. The smaller or the lower resolution a region is, the harder it is to detect its semantic label by a neural network [33], as shown in the evaluation of class-wise inference accuracy and region size in Table 2. This is why scene understanding applications require high-resolution videos to ensure high accuracy. This has been well discussed in the CV community [20], [21], but ignored in designing edge-cloud systems for video analytics.

As shown in Figure 1, upsampling low-resolution frames does improve the overall accuracy, and even achieves 94% of the best mIoU results with SR. However, the accuracy of those tail classes are still not satisfactory because they are generally small and sensitive to detailed quality loss.

The same reason also causes frame-wise tail accuracy. During a time series of video frames of moving robotics applications, the content of the video scenes constantly shifts, unlike fixed traffic cameras [18]. On the frames that are sensitive to detailed information, the inference accuracy of those classes will drop drastically on degraded data, and the frame-wise accuracy will drop accordingly. Naturally, the tail frames occur when the scene changes. The worst frames are not frequent, but they lead to poor tail accuracy performance.

2168-7161 (c) 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: Hong Kong University of Science and Technology. Downloaded on February 25,2022 at 02:00:33 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCC.2022.3142066, IEEE Transactions on Cloud Computing

2.3 Why does tail accuracy matter?

The tail accuracy issue would result in two problems in real-world scenarios: (i) Class-wise tail accuracy would harm the accuracy of some important classes, which are critical to the application performance. (ii) Frame-wise tail accuracy would hurt fluent decision-making and impair operations.

Class-wise tail accuracy is an important issue because those most affected classes are usually critical to robotics applications. The importance of an object often appears in small sizes in videos in practice [22]. For example, autonomous vehicles should pay more attention to the regions that are closely related to driving safety than those that are not crucial for vehicle operation. In other words, the system should focus on major obstacles/risks (e.g., pedestrians, riders, vehicles, and bicycles) and traffic information (e.g., traffic signs/lights). It does not need to pay the same attention to processing less critical regions such as the sky, vegetation, and buildings. It would be very dangerous to mis-detect a person or rider in the scene. In Figure 1, it is clear that some of the tail classes are more important for applications.

Frame-wise tail accuracy in a time series of frames is critical to robotics applications for two reasons. First, the operation of autonomous systems relies on the DL inference results of scene understanding CV tasks [10]. Robotics applications require correct DL inference to support the following decision-making. It is important to avoid extremely low tail accuracy on degraded data in edge-cloud systems. Second, the impact of tail accuracy would be magnified and significant for the overall service quality during a long time of operation and under a large-scale deployment.

2.4 Missing pieces in prior work

We argue that the real challenge for such edge-cloud video analytics systems is to handle the three-way trade-off between bandwidth consumption, overall accuracy, and tail accuracy, as shown in Figure 1. Let us review how prior work handles this challenge.

1) Skipping non-relevant and stale frames at the edge can reduce the bandwidth and computation load for fixed cameras with many similar frames [18], [19], [34]. However, robotics applications require a fast reaction within the stopping distance to the constantly changing traffic conditions [7]. We analyze our video datasets and find that 88%-94% of frames are essential to applications: they contain vehicles, people, traffic signs and actively change. So the room for saving bandwidth by skipping frames is limited here.
2) Splitting the CV model to both the edge and the cloud and only sending the intermediate CNN output of partial layers to the cloud [35], [36] can reduce the bandwidth consumption, compared to sending raw videos. This method uses the computation resource of edge devices to save bandwidth. However, robotics applications require complex CNN models on powerful hardware to provide high inference accuracy. Constrained edge devices can only run cheaper models to extract features, which may not achieve the best accuracy. For example, for semantic segmentation, using the edge-friendly MobileNetV3_Small [37] as the backbone takes only 1% computation of the heavy Xception-71 [38], but loses 14% accuracy [39], which is crucial for demanding applications. Also, the rapid development of CNN often causes backbone architectures to change, which will break the edge-cloud splitting configuration [40]. We do not dispute the work on the edge and DL accelerators [41], but present a new design considering these limitations.
3) The systems using compression/downsampling [17] and quality recovery with SR [28], [32] reduce the bandwidth consumption with acceptable overall accuracy, and utilize the computation resource on the cloud. However, such data degradation methods lead to the *tail accuracy* issue.

system design	3 requirements of edge-cloud systems		
	tail accuracy	overall accuracy	bandwidth
resolution compression [17], [28]	no	yes	yes
edge-cloud splitting [35], [36]	yes	no	yes
stale frame skipping [18], [19], [34]	yes	yes	no
high-quality video (baseline only)	yes	yes	no
Runespoor	yes	yes	yes

TABLE 1: The three requirements of edge-cloud video analytics for robotics applications.

3 Design

Runespoor is an edge-cloud video analytics framework especially for robotics applications. We propose two major components: *analytics-aware SR* (ASR) and *content-aware adaptive controller* (CAC). Figure 3 illustrates the workflow of Runespoor. The edge device adaptively compresses the video captured by the high-definition camera with CAC and streams it to the cloud server via the network. On the server, it first reconstructs the received data with ASR, then runs DL inference for CV tasks. CAC uses the DL inference results for the edge-side data rate control.

3.1 Analytics-aware super-resolution

Inspired by applying SR in video systems [28], [31], [32], we find that SR can be an effective method to strike a balance between bandwidth consumption and overall inference accuracy, however the tail accuracy issue still remains (§ 2.4). An SR model trained on the same datasets as the vision analytics tasks (NAS [28]) is naturally an enhancement for downsampling-based systems (e.g., AWStream [17]) and a strong baseline.

To address the three-way trade-off between bandwidth consumption, overall accuracy, and tail accuracy, especially the limitations of video compression techniques that lead to the poor tail accuracy issue (§ 2.2), Runespoor uses analytics-aware SR (ASR), which focuses on detailed information reconstruction for analytics tasks. ASR improves the DL inference accuracy on reconstructed frames, especially on the small regions and objects that are quality-sensitive to be correctly detected.

2168-7161 (c) 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: Hong Kong University of Science and Technology. Downloaded on February 25,2022 at 02:00:33 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCC.2022.3142066, IEEE Transactions on Cloud Computing

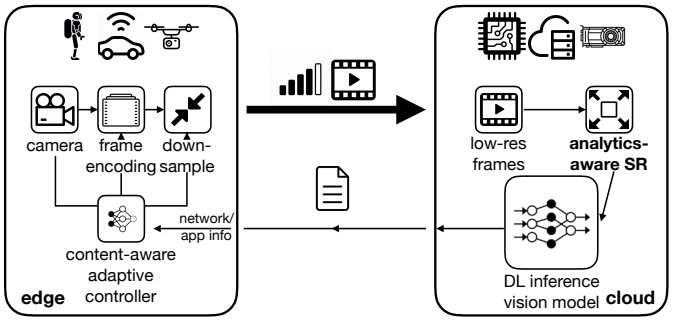


Fig. 3: Runespoor framework overview. The two design components are cloud-side ASR and edge-side CAC.

To train ASR, we first train a base SR model using the dataset for CV tasks (e.g., semantic segmentation and object detection), then use the *analytics-aware loss function* to fine-tune the SR model to improve the DL inference accuracy. We use this two-step arrangement instead of training from scratch because of the different nature of SR and DL tasks. First, data labeling is prohibitively labor-intensive [42], thus Cityscapes dataset only labels 1 out of 30 frames. Analytics-aware training requires data annotation for DL tasks, while the base SR training can use much more unlabeled frames besides the training dataset. Second, it allows us to apply different training techniques. For example, SR and DL tasks both use randomly cropping as data augmentation for generalization [43]. We can use small patches (64×64) for the base SR training to reconstruct small details, and large patches (720×720) for analytics-aware fine-tuning to identify regions of any scale to improve the training quality.

The key design of ASR is the *analytics-aware loss function* that maximizes tail accuracy. Equation 1 illustrates its general idea. l_s denotes the structural similarity loss function, which measures the frame reconstruction quality and stabilizes loss convergence. NAS [28] only uses l_s to train the standard SR. l_a denotes the task analytics loss, which measures the accuracy of DL inference tasks. *hr* and *lr* are the original high-resolution and downsampled low-resolution frames from the same datasets for training video analytics tasks. `asr_model` is the ASR model to train. `inf_model` denotes the video analytics model, which is pre-trained on high-resolution datasets for CV tasks. Runespoor uses SR as a pluggable module in the application workflow and does not require any customization on `inf_model` for generality in the real-world deployment. We compute gradients of the loss function on both `asr_model` and `inf_model` whose parameters are frozen. α and β are weight parameters. Through experiments, we recommend a larger α for robust convergence. In this way, we utilize the powerful `inf_model` architecture to make `asr_model` analytics-aware.

$$\begin{aligned}\text{loss} = \alpha \cdot l_s(\text{asr\_model}(lr), hr) + \\ \beta \cdot l_a(\text{asr\_model}, \text{inf\_model}, lr).\end{aligned} \tag{1}$$

We compare three potential designs of ASR and Equation 1, and explain the rationale behind ASR.

1) Train the SR model from scratch with the analytics-aware loss function (the standard way).
2) Train a base SR model, then fine-tune it with the analytics-aware loss to minimize the difference between inference results of post-SR and HR frames (similar to CloudSeg [31]).
3) Train a base SR model, then fine-tune it with the analytics-aware loss to minimize the final analytics error on detailed regions (our choice).

We find the first method that applying the analytics-aware loss at the beginning of ASR training is not an efficient way because of the different nature of SR and DL tasks. First, analytics-aware training requires data annotation for DL tasks, while the standard SR training needs no annotation (illustrated in Figure 4 and 6). Data labeling is prohibitively labor-intensive and time-consuming [23], [42], thus the Cityscapes dataset only labels 1 out of 30 frames. Standard SR training can use more easy-to-get unlabeled frames. Second, they use different training techniques. For example, SR and DL tasks both use randomly cropping as data augmentation for adaptability and generalization [43]. Training SR uses small patches (e.g., 64×64) to reconstruct small details. The scene understanding DL models are trained with large patches (e.g., 720×720) to identify regions of any scale. In ASR joint training, feeding the small patches produced by the SR model to the DL task model cannot obtain meaningful semantic results.

To implement Equation 1, another alternative method is using l_a to minimize the difference between the inference results of *hr* and reconstructed *lr*, as shown in Equation 2 and Figure 4. CloudSeg [31], which is a preliminary exploration of SR for video analytics, uses this method. Its rationale is treating HR (and the inference results on HR) as the target, which is intuitively inherited from the traditional SR training. We find this method inefficient, because a well-trained DL model for CV tasks can still make wrong predictions and brings inaccurate supervisory signals to the analytics-aware loss and affects `asr_model`. Figure 5 is an extreme example. Figure 5a is a sophisticated frame. Its inference accuracy is only 42.09% mIoU (Figure 5d) with HR data, which makes it a tail frame. It achieves only 5%-25% mIoU on tail classes like motorcycles, riders, and buses. We reconstruct this frame from ×4 downsampled data using the ASR trained with Equation 2. The output is Figure 5b, which gives a failed inference result as Figure 5e. This method limits the accuracy performance (evaluated in § 5.1.1), and even makes tail accuracy worse by augmenting false predictions, as we find in evaluations.

$$\begin{aligned}\text{loss} = \alpha \cdot l_s(\text{asr\_model}(lr), hr) + \\ \beta \cdot l_a(\text{inf\_model}(\text{asr\_model}(lr)), \text{inf\_model}(hr)).\end{aligned} \tag{2}$$

Instead of the CloudSeg method, to minimize tail accuracy, we use the classification error of the DL inference tasks on small regions as the analytics loss l_a, as shown in Equation 3 and Figure 6. We bring the label of the analytics tasks to ASR training, which CloudSeg does not utilize. More importantly, l_a (e.g., cross entropy loss for semantic segmentation) in Equation 3 concentrates on the loss signals from small/detailed regions instead of the whole frame to improve tail accuracy by targeting the root cause (§ 2.2). We select small or detailed (`dtl.`) regions by the relative size of a region to the frame. The threshold is different for each

2168-7161 (c) 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: Hong Kong University of Science and Technology. Downloaded on February 25,2022 at 02:00:33 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCC.2022.3142066, IEEE Transactions on Cloud Computing

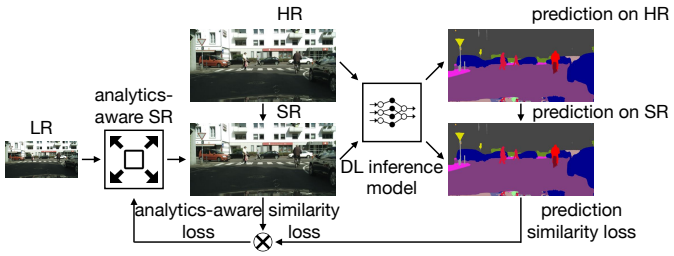


Fig. 4: Using inference results on HR as target (Equation 2).

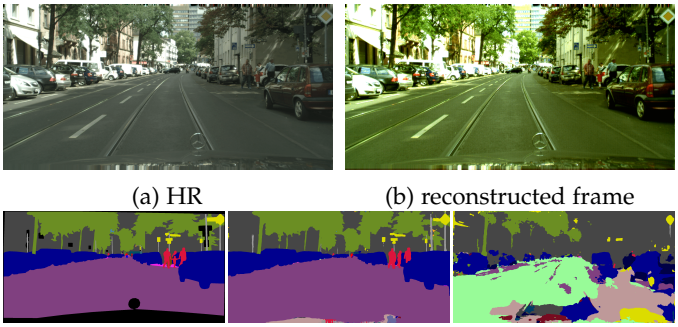


Fig. 5: An extreme example shows that training ASR with Equation 2 may hurt tail accuracy.

dataset. As a result, the large and stable areas like the sky are not considered in l_a.

$$\begin{aligned}\text{loss} = \alpha \cdot l_s(\text{asr\_model}(lr), hr) + \\ \beta \cdot l_a(\text{inf\_model}(\text{asr\_model}(lr))_{\text{dtl.}}, \text{label}_{\text{dtl.}}).\end{aligned} \tag{3}$$

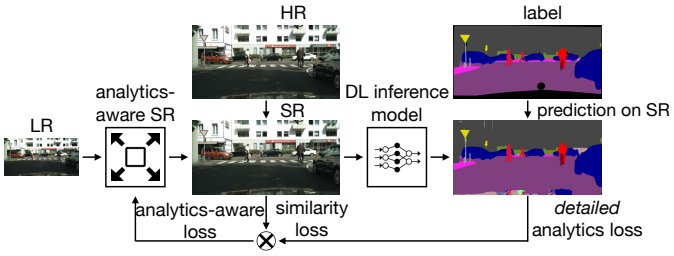


Fig. 6: We use DL inference results of detailed regions as the analytics-aware optimization targets (Equation 3).

To summarize the full design of ASR, we first train the standard SR model with a large volume of unannotated data, then fine-tune it with task-specific labeled data and the analytics-aware loss function (Equation 3) that focuses on the DL inference results of detailed regions so that it can handle the low tail accuracy issue on degraded data in edge-cloud systems. ASR models are deployed on the server, improving the resolution of the received compressed data and feeding processed frames to the DL analytics model.

3.2 Content-aware adaptive controller

Towards the bandwidth constraint in the online operation of edge-cloud systems, conventional wisdom [17], [44] suggests three knobs to adjust data rate: resolution, frame rate, and encoding quality and use a data rate controller to adapt to network conditions. For example, AWStream [17] learns the knob configurations for tasks by *offline profiling* and periodically updates the controller to handle *model drift* (less efficiency when the environment changes) by *online profiling*. Its online profiling works as steps 1 to 3 in Figure 7: (i) periodically profiling the current DL inference performance with extra raw data, and (ii) updating the data rate strategy, to (iii) make the corresponding adjustments in video knobs.

However, we find that for video analytics in robotics applications, the conventional data rate controller [17] is incapable of handling the tail accuracy issue. First, the conventional controller learns the data rate policy for a dataset of similar scenes under varying bandwidth constraints. Its design goal is to handle network changes instead of *frequent scene changes*. Thus it does not detect and react to the few frames that lead to low tail accuracy (§ 2.2). Second, periodically online profiling (step 1) requires extra bandwidth consumption (to send raw data as a reference) and DL computation (for baseline accuracy). Thus it takes tens of seconds for AWStream [17] to react to significant scene changes only like camera movement. Such a slow process cannot catch "tail frames" in robotics applications because the newly-updated policy has already become stale.

We observe that the video analytics performance is highly related to the semantic content of the video. If a frame contains fewer small regions or objects, then it is more likely to become a tail frame because the accuracy of hard-to-detect classes is easily affected by wrong predictions, as illustrated in Figure 8. In addition, video analytics applications also have significant content correlations [45], [46]. Easy-to-analyze frames or tail frames typically appear as a sequence because of their similar contents.

Considering these two issues and our observation, we propose *content-aware adaptive controller* (CAC) to use the video analytics results to instantly detect the frames that could lead to extremely low accuracy on degraded data ("tail frames") and apply specialized video knobs configurations to improve tail accuracy. Figure 7 shows the workflow of CAC. Two key designs of CAC are: (i) *tail frame detection* with scene understanding predictions; and (ii) *tail-aware offline profiling* to obtain tail-specific data rate configurations.

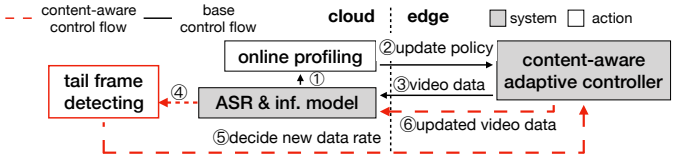


Fig. 7: Online workflow of content-aware adaptive controller (CAC). Steps 4 to 6 are content-aware. Steps 1 to 3 follow AWStream [17].

Tail frame detection requires a *low latency* and a *high recall rate* to catch all potential tail frames in fast changing environments. Lower precision, i.e., mis-detecting simple frames as tail frames, will not hurt the final accuracy because simple frames are quality-robust for DL tasks (§ 2.2) under the same bandwidth constraint, as evaluated in § 5.3. We use a threshold of the ratio of small regions to detect tail frames: checking the DL inference result (e.g., semantic segmentation)

2168-7161 (c) 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: Hong Kong University of Science and Technology. Downloaded on February 25,2022 at 02:00:33 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCC.2022.3142066, IEEE Transactions on Cloud Computing

of a frame, if the size ratio of all small regions in the frame is below the threshold, we consider it very likely to be a tail frame, based on our previous observation and experiments. CAC applies specialized data configurations to send the subsequent frames to improve the DL inference accuracy of these tail frames. The small region here is defined the same as in Equation 3.

Finding this threshold is done together with *tail-aware offline profiling*. For generalization, we use a linear regression model to determine the threshold for tail frame detection using the DL profiling results of a dataset. The threshold we obtain for the Cityscapes dataset is 1%. We analyze the tail frame detection results on the Cityscapes validation dataset in Figure 8. The accuracy (mIoU) is calculated per frame, thus lower than the cumulative overall accuracy. The red crosses depict 95% to 100% tail accuracy. Using the 1% threshold of the ratio of small regions to detect tail frames, the recall rate is over 90%, which means we can catch almost all tail frames. The precision is between 25% and 35% on three scales, but we do not require high precision. By utilizing DL inference results of analytics tasks, the online computation overhead of CAC is finding small regions by depth-first search on downsampled prediction maps (segmentation) or directly adding small bounding boxes (detection), which takes <1ms (unnoticeable) per frame. The detection is accurate even with low-resolution (8×) frames.

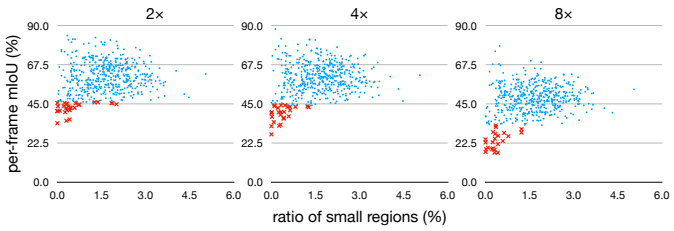


Fig. 8: Using a 1% threshold of small region ratio catches tail frames with a high recall rate. 8× is the lowest resolution.

Tail-aware offline profiling in CAC extends prior work AWStream [17] to optimize tail accuracy. For a CV task, offline profiling on the training dataset with different knobs (resolutions and encoding qualities) processed by ASR will find the best data rate configurations for high accuracy regarding different bandwidth constraints. We separate offline profiling to find (i) the *tail-specific* knob configurations on frames with 90% to 100% accuracy (tail frames) and (ii) the standard configurations on the rest of the dataset. As discussed before, tail frames need a higher resolution to be correctly detected. The tail-specific configurations we obtain have a higher resolution, lower encoding quality, and the same data size. This is because tuning encoding quality within a range can reduce the data rate without hurting accuracy [17], [44]. This brings a significant tail accuracy improvement and almost no loss for other frames. Offline profiling is a one-time process, which takes tens of minutes for the Cityscapes dataset with dozens of configurations.

In summary, CAC in Figure 7 works as follows. At the server, in addition to periodically online profiling as AWStream (steps 1 to 3) for standard performance monitoring, CAC gathers the video analytics results to detect tail frames (step 4). If tail frames are detected, CAC will adaptively apply a specialized data configuration (step 5) learned at the tail-aware offline profiling. At the edge, CAC instantly (~500ms, § 5.3) applies the specialized configurations for the subsequent similar frames (step 6) to improve tail accuracy, compared with ~10s of AWStream. During the online operation, CAC will adjust data rate configurations according to the video content, video analytics and network performance. CAC requires no extra DL computation and raw data to obtain accuracy baselines online, and has the same offline profiling workload as AWStream.

4 Implementation

Framework. We implement the end-to-end Runespoor prototype as Figure 3. We build video frames and control messages transmission with ZeroMQ [47] between edge and cloud, and DL inference with PyTorch [30] at the cloud. To achieve real-time video analytics, we use *per-frame inference* [48], and implement CV and encoding operations (e.g., downsampling) with OpenCV. Multiple ASR models are loaded via Flask at the cloud for different resolutions of received data. ASR models are not memory-consuming comparing with DL inference models. We build two applications which are widely used in real-world autonomous systems [49]: semantic segmentation and object detection.

Dataset. We implement semantic segmentation on the urban road driving dataset Cityscapes [27], and object detection on the drone-view dataset VisDrone2019 [50]. They are both scene understanding tasks for real-world autonomous robotics applications [49] and demand accuracy performance.

Cityscapes dataset consists of 8-bit RGB video frames in 2048×1024 resolution and a 17 fps frame rate, which are captured by the front camera of vehicles driving in the cities. It contains 2975 training frames, 500 validation frames, and more unlabeled video clips which are used in ASR training and CAC evaluation.

VisDrone2019 image dataset consists of frames collected by drones, mostly in cities, and the resolution is up to 2000×1500. We use 1000 high-quality frames with a minimum resolution of 1920×1080 to provide high inference accuracy, especially for small objects. They are split into 700 and 300 images for ASR training and object detection evaluation, respectively.

To evaluate the DL inference accuracy, we use standard metrics: intersection over union (IoU) for semantic segmentation, and average precision (AP) for object detection. Regarding the bandwidth constraints, CAC adaptively sends compressed video frames, and ASR processes the frames for inference. The video knobs include resolution, frame rate, and encoding quality.

Training ASR and CAC. We implement ASR with PyTorch [30] based on CARN [29], which is an efficient SR architecture. The base ASR model (CARN) is not exclusive. To train the base SR models (§ 3.1), we use the training datasets plus unlabeled frames. The base training uses standard techniques including image patches (64×64) by randomly cropping and flipping, and the default L1 loss function to evaluate the difference between post-SR and HR images. After the base training, we get a set of SR models as a strong

2168-7161 (c) 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: Hong Kong University of Science and Technology. Downloaded on February 25,2022 at 02:00:33 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCC.2022.3142066, IEEE Transactions on Cloud Computing

baseline of NAS [28] for the trade-off between bandwidth consumption and overall accuracy.

We use SwiftNet [2] and YOLOv3 [1] to implement our analytics-aware training for semantic segmentation and object detection, respectively. The CV models are also not exclusive. We choose these two because they both consume high-resolution data to achieve state-of-the-art inference accuracy in real-time (§ 5.3). We use the model weights provided by authoritative implementations [51], [52] which are pre-trained on the target datasets. They will not be modified in ASR training.

The input to SwiftNet is in 2048×1024, and YOLOv3 input is set to 832×832 to exploit the high-resolution data. To train ASR with the analytics-aware loss function (Equation 3), we set α and β to 0.95/0.05 and 0.9/0.1 through experiments. For semantic segmentation, we use cross entropy as the analytics loss l_a. For object detection, the analytics loss l_a includes cross entropy and focal loss [53], which calculates the classification and bounding-box location error.

To find the standard and tail-specific knob configurations in CAC, offline profiling includes resolutions from ×2 to ×8, JPEG encoding quality from 40 to 90, and processed by ASR regarding different bandwidth targets. Usually, tail-specific configurations have one level higher of resolution. We use depth-first search (DFS) to find small regions on downsampled prediction maps of scene understanding models, which is accurate and gives no noticeable overhead.

5 Evaluation

We show the evaluation results of Runespoor from:

§ 5.1 Runespoor handles both class-wise and frame-wise *tail accuracy*. ASR improves 50% to 100% class-wise accuracy by up to 300%, and frame-wise 90% and 99% accuracy by up to 22% and 54%, respectively.

§ 5.2 For the traditional goal of edge-cloud analytics systems, ASR reduces *bandwidth consumption* and also improves *overall inference accuracy*.

§ 5.3 During *online operation*, CAC instantly reacts to the hard-to-detect and complex frames and further improves *tail accuracy*. The end-to-end Runespoor system is efficient.

5.1 Tail accuracy improvements

Edge-cloud video analytics systems send compressed data to save bandwidth but lead to low tail accuracy. We compare ASR of Runespoor with three baselines which improve the accuracy of scene understanding DL inference tasks: (i) Bilinear algorithm (used in AWStream [17]) and (ii) the standard SR model (used in NAS [28]). (iii) CloudSeg [31] described in § 3.1. Bilinear is the default image resizing algorithm of PyTorch [30]. The standard SR models are well trained and achieve their optimal performance in PSNR [24], which shows the reconstruction performance to keep similarity. We compare with CloudSeg, a previous version of this work, which only focuses on the overall accuracy of semantic segmentation and is not an end-to-end system, on class-wise tail accuracy of segmentation. As a reference, we show the DL inference results on the original high-resolution data. The original resolution of Cityscapes validation dataset for semantic segmentation is 2048×1024 (HR), and ×2 means 1024×512 and so on. The VisDrone validation dataset has original frames in 1920×1080 and 2000×1500.

5.1.1 Class-wise tail accuracy

Figure 9 shows the class-wise DL inference results breakdown when receiving 512×256 (×4) frames for semantic segmentation, using three baselines and ASR. The per-class accuracy is normalized to the HR inference accuracy. ASR improves explicitly the accuracy of those semantic classes that perform poorly on degraded data, e.g., motorcycles, riders, buses, and fences, while CloudSeg [31] shows little or even negative improvements. Some classes having normalized accuracies that are higher than 100% (which is equivalent to ASR having higher accuracies than HR in Table 2) are reasonable. It is because (1) training ASR essentially employs larger DNN architectures of SR plus the powerful segmentation model and more iterations on labeled data, making ASR a data augmentation method to the video analytics tasks; and (2) ASR improves under-trained classes (which seldom appear in the dataset like fence and motorcycle) in the pre-trained CV model whose accuracies are already low with HR. With Runespoor, all 19 classes achieve higher accuracy than the overall accuracy of other baselines (right). This shows that ASR is effective to fix the poor class-wise tail accuracy issue.

In detail, we compare with prior work on the average of 50% to 100% class-wise tail accuracy, which includes the worst 9 classes ("tail classes") out of 19 in Figure 9. Table 2 shows the tail accuracy breakdown of semantic segmentation. Comparing with the standard SR (NAS [28]), Runespoor exceeds on the average of 50% to 100% class-wise tail accuracy by 8.4% (from 61.73% to 66.92%), which is larger than the overall accuracy improvement (5.5% from 71.15% to 75.04%). Comparing with the default Bilinear algorithm (AWStream [17]), Runespoor exceeds on the average of 50% to 100% class-wise tail accuracy by 23.7% (from 54.11% to 66.92%), which is also larger than the overall improvement(13.9%). While CloudSeg [31] generally improves the accuracy, it is not always effective on tail accuracy and sometimes even makes some classes worse.

class IoU(%)	Bilinear	standard SR	CloudSeg	**ASR**	HR	size(%)
train	45.45	58.12	68.95	70.41	70.41	0.11
fence	41.83	50.09	55.49	61.12	56.20	0.82
motorcycle	45.96	53.38	53.22	59.02	58.44	0.08
traffic light	54.54	63.80	63.82	65.01	67.65	0.20
rider	47.34	51.64	52.87	55.96	58.28	0.22
bus	70.69	76.71	78.33	82.66	85.17	0.39
pole	52.73	59.05	60.54	61.24	62.91	1.48
bicycle	63.57	70.78	70.23	72.25	75.77	0.71
traffic sign	64.91	71.96	71.84	74.60	76.95	0.67
tail classes	54.11	61.73	62.72	**66.92**	68.30	4.68
normalized	0.79	0.90	0.92	**0.98**	1.00	in total
overall mIoU(%)	65.87	71.15	72.46	**75.04**	75.35	—

TABLE 2: Class-wise tail accuracy breakdown at ×4 of semantic segmentation. ASR largely improves the class-wise tail accuracy performance comparing with prior work.

ASR of Runespoor works for all resolution inputs, especially for those experiencing more accuracy loss. Table 3

2168-7161 (c) 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: Hong Kong University of Science and Technology. Downloaded on February 25,2022 at 02:00:33 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCC.2022.3142066, IEEE Transactions on Cloud Computing

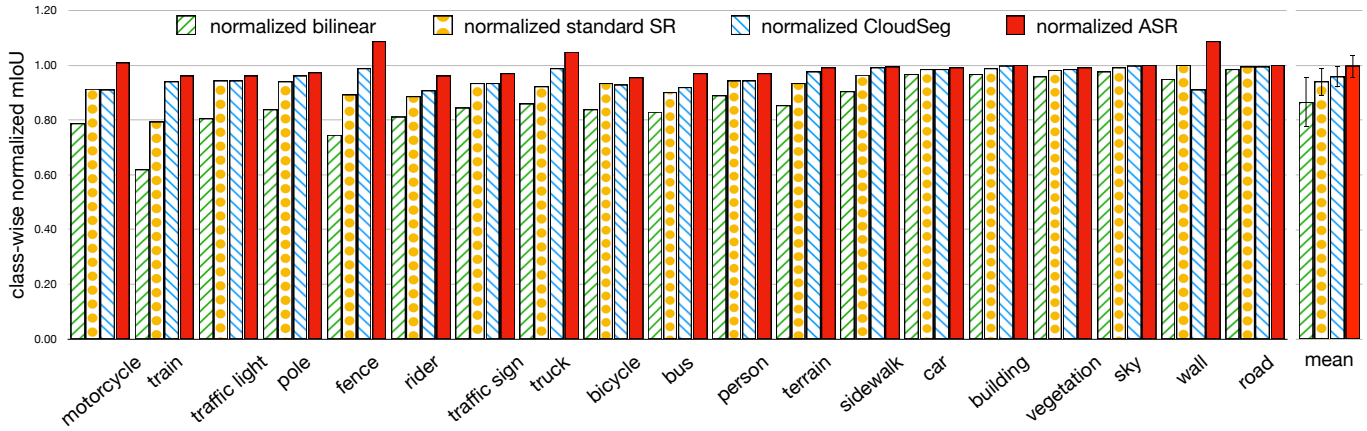


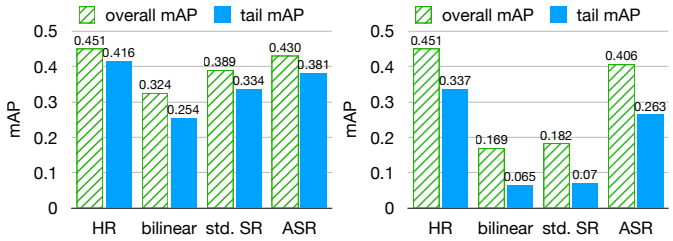
Fig. 9: ASR effectively improves the class-wise tail accuracy, and has the lowest standard deviation with a higher mean accuracy.

shows that comparing with Bilinear and the standard SR, ASR improves the average of 50% to 100% class-wise tail accuracy by 4.5%/0.9%, 23.7%/8.4%, 58.2%/20.3% and 79.4%/65.9% respectively on ×2, ×4, ×6 and ×8 compressed inputs. These results show that Runespoor can effectively manage the low class-wise tail accuracy issue in edge-cloud systems, which is important to the operation of robotics applications.

mIoU(%)		**tail classes**	overall
HR 2048×1024		68.30	75.35
×2 1024×512	bilinear	64.60	73.11
	standard SR	66.91	74.66
	ASR	**67.52**	**75.12**
×4 512×256	bilinear	54.11	65.87
	standard SR	61.73	71.15
	ASR	**66.92**	**75.04**
×6 341×170	bilinear	38.17	53.13
	standard SR	54.77	65.14
	ASR	**61.24**	**70.81**
×8 256×128	bilinear	25.36	40.49
	standard SR	27.42	45.40
	ASR	**45.50**	**60.32**

TABLE 3: Overall and class-wise tail accuracy of all scales for semantic segmentation.

We report the class-wise tail accuracy performance of object detection in Figure 10. ASR of Runespoor beats the two baselines at ×2 and ×4 inputs and obtains good performance compared with original data. The 50% to 100% tail classes of the VisDrone dataset include tricycles, bicycles, awning-tricycles, motors, and people, which are all small and hard to detect in the view of drones. ASR improves the class-wise tail accuracy of object detection by 14% to 300%, and overall accuracy by 11% to 140%. Let us take a close look. In this task, ×4 downsampled data hurts the accuracy significantly. For example, accuracy (mAP) of tricycles detection on ×4 data with the standard SR only achieves 10% normalized accuracy. Impressively, ASR improves it to 91%. Figure 11 shows results on a real complex frame; Figure 12 enlarges the parking and pedestrian areas to show more details. We can see that the frame reconstruction of ASR which focuses on the detailed information lets the DL inference model detect the compact parked cars (left) and people on the street (right) much more accurately than the standard SR.



(a) ×2 downsampling (b) ×4 downsampling

Fig. 10: Average 50% to 100% class-wise tail accuracy of the object detection task. ASR improves class-wise tail accuracy especially on ×4 downsampled input.

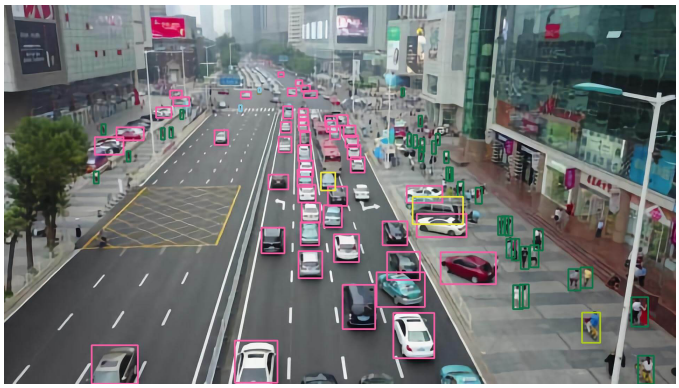
5.1.2 *Frame-wise tail accuracy*

Mitigating high-percentile accuracy over time is important to the responsiveness and operation of robotics applications, as discussed in § 2. In this part, we show that ASR of Runespoor can improve temporal frame-wise tail accuracy comparing with the standard SR in semantic segmentation. We show the 90% and 99% accuracy on the Cityscapes validation dataset of 500 frames for semantic segmentation in Figure 13. Each data point is an average accuracy on an episode of 10 frames. The horizontal lines for mean mIoU are different from the overall mIoU in Figure 9 because the overall mIoU is accumulated on the whole dataset (i.e., 500-frame mIoU), while the mean mIoU in Figure 13 is the average accuracy of all frame episodes.

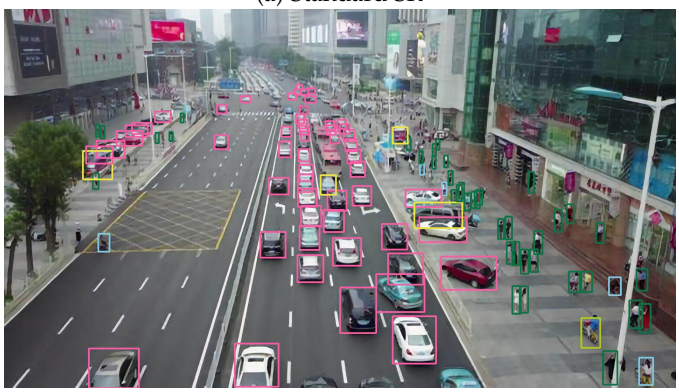
Figure 13 shows the frame-wise accuracy distribution with ×4 and ×6 inputs for semantic segmentation. ASR improves all frames in general, especially the tail ones with low accuracy (more obvious in Figure 13a), thanks to its focus on detailed information. Table 4 shows the tail accuracy in detail. ASR improves 90% accuracy by 18% to 22%, and 99% accuracy by 35% to 54% comparing with the standard SR (NAS).

2168-7161 (c) 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: Hong Kong University of Science and Technology. Downloaded on February 25,2022 at 02:00:33 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCC.2022.3142066, IEEE Transactions on Cloud Computing



(a) Standard SR



(b) ASR

Fig. 11: Object detection on ×4 inputs. The detector finds more details on post-ASR frames.

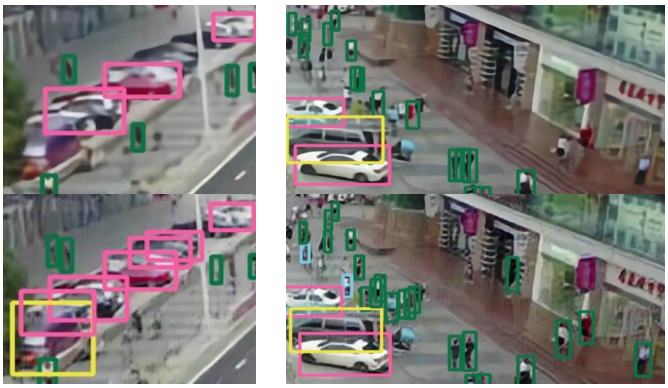
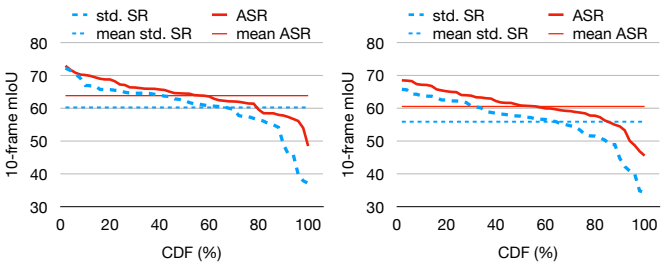


Fig. 12: Enlarged details of object detection on ×4 downsampling with standard SR (top) and ASR frames (down).

5.2 Overall accuracy and bandwidth saving

As we focus on the tail accuracy performance, ASR naturally improves the overall accuracy at the same bandwidth consumption, which is also important for edge-cloud systems.

Table 5 compares the overall accuracy of semantic segmentation across these schemes. We can see that ASR improves the overall accuracy by 3% to 49% compared with Bilinear algorithm (AWStream [17]) and 1% to 33% compared with the standard SR (NAS [28]). Figure 14 shows the semantic segmentation accuracy-bandwidth comparison of Runespoor, NAS, and AWStream. From a bandwidth saving perspective, to achieve the optimal overall accuracy



(a) ×4 downsampling (b) ×6 downsampling

Fig. 13: Frame-wise accuracy distribution of semantic segmentation. ASR especially improves tail accuracy performance.

10-frame mIoU(%)		90%	99%	mean
×4 512×256	standard SR	49.01	37.93	60.29
	ASR	**57.83**	**54.03**	63.88
×6 341×170	standard SR	44.75	34.86	55.93
	ASR	**54.51**	**46.92**	60.58

TABLE 4: 90% and 99% frame-wise tail accuracy for semantic segmentation.

(75% mIoU), ASR outperforms NAS by 3.5× and AWStream by 14.3×. To achieve 70% mIoU, ASR can still save 2.1× and 6.9× bandwidth consumption comparing with prior work.

scale	metric	Bilinear	standard SR	**ASR**	HR
×2	mIoU(%)	73.11	74.66	**75.12**	75.35
	PSNR	36.72	40.13	39.91	—
	SSIM	0.981	0.991	0.989	
×4	mIoU(%)	65.87	71.15	**75.04**	75.35
	PSNR	31.79	35.34	35.06	—
	SSIM	0.946	0.972	0.968	
×6	mIoU(%)	53.13	65.14	**70.81**	75.35
	PSNR	29.34	33.30	32.75	—
	SSIM	0.917	0.955	0.950	
×8	mIoU(%)	40.49	45.40	**60.32**	75.35
	PSNR	27.81	30.62	30.57	—
	SSIM	0.897	0.929	0.924	

TABLE 5: Overall accuracy comparison for semantic segmentation. Runespoor achieves higher accuracy than other baselines while exhibiting similar visual quality.

To show the image reconstruction quality, we use PSNR (peak signal-to-noise ratio) [24] and SSIM (structural similarity index) [25]. A higher PSNR or SSIM shows that the reconstructed images are more close to the original ones. In Table 5, ASR shows a slightly lower reconstruction quality compared with the standard SR, because the standard SR is trained only towards the similarity target, while ASR is not. We design ASR to sacrifice some image quality (e.g., clouds in the sky) for higher DL inference accuracy on those details that are critical to the tail accuracy performance.

For the object detection task, Runespoor also achieves a better accuracy-bandwidth performance. In Table 6 (same mAP data as in Figure 10), ASR improves the overall accuracy by 33% to 140% comparing with the bilinear

2168-7161 (c) 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: Hong Kong University of Science and Technology. Downloaded on February 25,2022 at 02:00:33 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCC.2022.3142066, IEEE Transactions on Cloud Computing

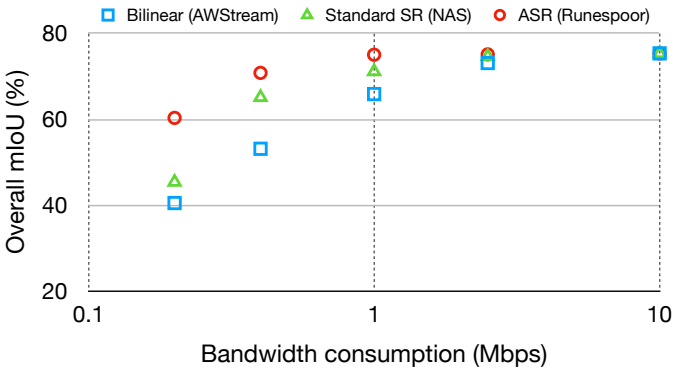


Fig. 14: Accuracy and bandwidth consumption at a same high-quality encoding. X-axis is in log scale. Runespoor achieves higher accuracy with the same bandwidth.

algorithm (AWStream [17]), and 11% to 123% compared with the standard SR (NAS [28]), at ×2 and ×4 (bandwidth) respectively. To achieve a high object detection accuracy (0.4 mAP), Runespoor saves 3.4× bandwidth consumption than the standard SR. The PSNR and SSIM metrics on the VisDrone dataset show the same results as the Cityscapes dataset.

scale	metric	bilinear	standard SR	ASR	HR
×2	mAP	0.324	0.389	**0.430**	0.451
	PSNR	26.75	28.54	28.16	—
	SSIM	0.849	0.893	0.882	
×4	mAP	0.169	0.182	**0.406**	0.451
	PSNR	24.52	25.04	24.54	—
	SSIM	0.772	0.784	0.767	

TABLE 6: Overall performance comparison for object detection. Runespoor achieves higher accuracy than other baselines especially on ×4 downsampled data.

5.3 End-to-end performance with CAC

Content-aware adaptive controller (CAC) works in the online operation of edge-cloud video analytics of robotics applications. We use a Linux desktop as the edge device and a local server with 4 NVIDIA K40m GPUs as the cloud. We use the Linux `tc` utility to modify the outgoing bandwidth of the edge device to emulate the edge-cloud environment. Cityscapes dataset only labels 1 frame in every 30 frames (1.8s video clip) because annotation is costly, so does VisDrone, as discussed in § 3.1. We choose the Cityscapes dataset to evaluate CAC for its diversity of scenes. To simulate the real-world online operation, we use a consecutive 20-second 17-fps *unlabeled* video clip (340 frames) to evaluate CAC on frame-wise tail accuracy handling. We use the DL predictions on original HR as the baseline to evaluate the relative frame-wise accuracy. We process the baseline and mark the ignored areas to make it in Cityscapes label format. We also mark the ignored areas (black area in Figure 5c like the vehicle itself) in this evaluation to make it more accurate. Figure 15 shows the average relative accuracy of every second.

We initially set the outgoing bandwidth to 10 Mbps (2048×1024, 17 fps, 90 encoding quality), which is an abundant 4G uploading bandwidth [54]. There are four stages in our evaluation: (i) before t=5s, we set the bandwidth to 10 Mbps; (ii) at t=5s, we limit the bandwidth to 5 Mbps, which is typical for uploading; (iii) at t=12s, we reduce the bandwidth to 2 Mbps; (iv) at t=19s, we reset the bandwidth to 10 Mbps. During 14s to 16s, the frames that lead to low tail accuracy occur.

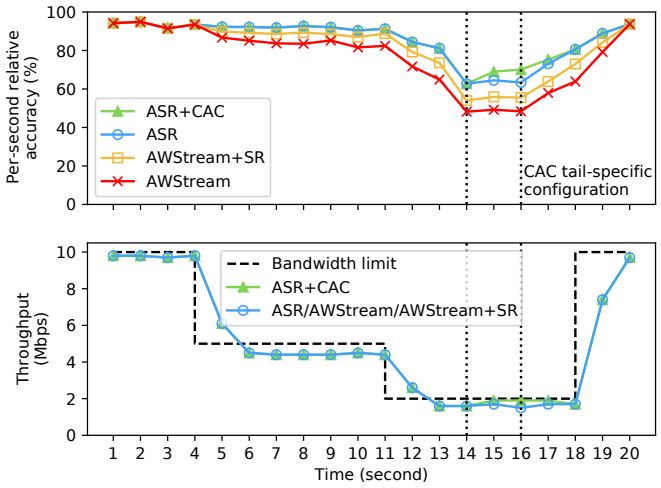


Fig. 15: CAC improves tail accuracy in the real-world video stream. Accuracy is relative to performance on HR.

Figure 15 shows that CAC takes ~500ms of latency to detect and switch to the tail-specific knob configuration (higher resolution and lower encoding quality) for the subsequent frames, comparing with the ~10 seconds of delay to detect scene changes in AWStream [17]. At around t=15s, Runespoor with CAC achieves a higher tail accuracy than ASR only. At t=17s, CAC switches back to the standard configuration. CAC, including the tail-specific configuration (14s-16s), has a similar throughput as other baselines, as shown in the lower figure. This is because we learn these configs through offline profiling regarding the same bandwidth targets, as discussed in § 3.2.

We show the breakdown of end-to-end latency per frame in Table 7. Each application has 410-530ms and 330-420ms latency. DL computations are distributed over multiple GPUs. Runespoor is efficient to support our target applications. Note that CloudSeg [31] only implements an SR model and has no end-to-end implementation.

time (ms)	transmission+processing				ASR	inference
	×2	×4	×6	×8		
segmentation	144	40	20	14	68-80	316
detection	133	37	—	—	66-70	220

TABLE 7: Breakdown of end-to-end latency per frame.

6 Related Work

6.1 Edge-cloud video analytics systems

In many real-time video analytics applications, it is fundamentally challenging to co-locate expensive computation resources with high-fidelity video data due to scalability, cost, and energy restrictions. With more edge devices being

2168-7161 (c) 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: Hong Kong University of Science and Technology. Downloaded on February 25,2022 at 02:00:33 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCC.2022.3142066, IEEE Transactions on Cloud Computing

deployed in geographically distributed locations, it is difficult to send the videos collected at the edge to the cloud for analytics under limited bandwidth.

A common approach is to reduce the video quality at the edge, e.g., via pixel-level (spatial) and frame-level (temporal) downsampling. The degradation must ensure that sufficient information is retained so that the cloud can run video analytics on downsampled videos to produce sufficiently accurate results. For example, AWStream [17] learns the Pareto-optimal policy for a task, and adaptively selects a video degradation strategy to achieve the best accuracy at the given WAN bandwidth. DDS [55] designs an iterative workflow driven by the server-side DNN feedback to stream the video. It can react to the real-time inference results by sending the low-quality video first then encoding the feedback regions in a higher quality. It achieves a better bandwidth-accuracy trade-off and lower end-to-end delay.

FilterForward [18], along with other filter-based frameworks [19], [34], selectively filters video frames at the edge with a small neural network to reduce bandwidth consumption. The insight is that for fixed-view surveillance applications, there are a lot of non-relevant frames that can be skipped in DL video analytics. Reducto [46] dynamically adapts its filtering decisions to the changing features and video content.

Neurosurgeon [35] and Split-Brain [36] split the DL computation between the edge and the cloud: they process the frames at the edge and only send the output of partial CNN layers (features) to reduce the bandwidth consumption compared with sending raw videos. Prior work [56] also studies the cloud offloading policy for edge devices to improve the analytics accuracy and the focus is still edge computing. Brainwave [41] and DeepCPU [57] focus on efficient DNN accelerators.

6.2 Super-resolution for video analytics

Our ASR is based on the recent advancements in super-resolution techniques. SR reconstructs a high-resolution image from a low-resolution image, by inferring details based on information in the LR image. Recently, CNN-based SR models have significantly improved the performance [29].

Prior work in both system and CV has shown that SR is a promising approach to improving video streaming quality [28] and analytics accuracy [31], [32], [58] when only low-resolution videos are available. NAS [28] trains content-aware SR models and deploys them on the client device to improve the video streaming QoE with limited bandwidth. NAS uses scalable DNN [59] of SR which can take multiple input resolutions regarding resource restrictions because it is deployed at the client instead of the cloud server. However, directly applying NAS to video analytics systems does not address the tail accuracy issue (§ 2.4).

6.3 Robust performance in CV/ML

Achieving robust edge-cloud video analytics performance motivates Runespoor because this is critical to real-world applications (§ 2). There are many related works in the CV/ML community on the tail accuracy issue, or generally on the robust performance.

For *frame-wise* performance, stability training [60] stabilizes the CV model against small input distortions to enable a higher performance on noisy visual data. An adversarial network approach [61] trains the model with the adversarial noise using a generative network (GAN) to stabilize the performance. ACE [23] synthesizes new data (e.g., winter and dawn) to train semantic segmentation to adapt to changing environments over time.

For *class-wise* performance, IAL [22] introduces the notion of class importance for autonomous driving and trains an importance-aware semantic segmentation model that favors the accuracy of some important classes. WBA [62] acknowledges the class-wise imbalanced distribution including importance and rareness. They propose an evaluation framework and an accuracy metric that to manage the arbitrary skews in class cardinalities and importances. Note that comparing with prior CV/ML work addressing the tail accuracy or robustness issue, Runespoor does not modify or retrain the backend CV inference model for generality.

6.4 Networking systems for machine perception

Runespoor aims for better analytics performance, rather than high quality in human perception. Today a large proportion of videos are consumed by computers for analytics instead of being watched by human. There is a growing trend to treat machine perception as a key design factor in edge-cloud video analytics systems. Task-aware encoding [40] explores specialized video encoding schemes for high inference accuracy with reinforcement learning. DeepN-JPEG [63] explores image compression methods that are favored by image recognition tasks.

7 Conclusion

For modern edge-cloud video analytics applications, the tail accuracy performance is critical but often ignored when handling the bandwidth consumption and overall accuracy trade-off. We propose Runespoor to enable demanding applications (e.g., autonomous robotics vehicles) to take advantage of the cloud and conduct advanced video analytics efficiently with edge-cloud computing. Runespoor improves *tail accuracy* with *analytics-aware SR* (ASR) to reconstruct detailed information from compressed data, and *content-aware adaptive controller* (CAC) to instantly adapt to fast-changing scenes. Evaluations show that Runespoor successfully manages the trade-off between bandwidth consumption, overall and tail accuracy.

Acknowledgement: This work is supported in part by the Hong Kong RGC TRS T41-603/20-R and GRF-16215119 and GRF-16213621. Kai Chen is the corresponding author of this paper.

References

[1] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," *arXiv Preprint arXiv:1804.02767*, 2018.

[2] M. Orsic, I. Kreso, P. Bevandic, and S. Segvic, "In defense of pre-trained imagenet architectures for real-time semantic segmentation of road-driving images," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 12 607–12 616.

2168-7161 (c) 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: Hong Kong University of Science and Technology. Downloaded on February 25,2022 at 02:00:33 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCC.2022.3142066, IEEE Transactions on Cloud Computing

[3] M. Teichmann, M. Weber, M. Zoellner, R. Cipolla, and R. Urtasun, "MultiNet: Real-time joint semantic reasoning for autonomous driving," in *2018 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2018, pp. 1013–1020.

[4] A. Kirillov, K. He, R. Girshick, C. Rother, and P. Dollár, "Panoptic segmentation," *arXiv Preprint arXiv:1801.00868*, 2018.

[5] M. Simon and A. Pardes. (2019) The prime challenges for Scout, Amazon's new delivery robot - WIRED. [Online]. Available: https://www.wired.com/story/amazon-new-delivery-robot-scout/

[6] A. Marshall. (2020) Delivery robots aren't ready when they could be needed most - WIRED. [Online]. Available: https://www.wired.com/story/delivery-robots-arent-ready-when-needed-most/

[7] S.-C. Lin, Y. Zhang, C.-H. Hsu, M. Skach, M. E. Haque, L. Tang, and J. Mars, "The architectural implications of autonomous driving: Constraints and acceleration," in *Proceedings of the Twenty-Third International Conference on Architectural Support for Programming Languages and Operating Systems*. ACM, 2018, pp. 751–766.

[8] J. Condliffe. (2018) Why sidewalk delivery robots still need safety drivers - MIT Technology Review. [Online]. Available: https://www.technologyreview.com/f/610107/why-sidewalk-delivery-robots-still-need-safety-drivers-too/

[9] I. Boudwayn. (2020) Delivery robot operators are also working from home - Bloomberg. [Online]. Available: https://www.bloomberg.com/news/articles/2020-05-13/delivery-robot-operators-are-also-working-from-home

[10] L. Fridman, D. E. Brown, M. Glazer, W. Angell, S. Dodd, B. Jenik, J. Terwilliger, J. Kindelsberger, L. Ding, S. Seaman *et al.*, "MIT autonomous vehicle technology study: Large-scale deep learning based analysis of driver behavior and interaction with automation," *arXiv Preprint arXiv:1711.06976*, 2017.

[11] M. Burns. (2019) "Anyone relying on lidar is doomed," Elon Musk says - TechCrunch. [Online]. Available: https://techcrunch.com/2019/04/22/anyone-relying-on-lidar-is-doomed-elon-musk-says/

[12] H. Zhang, G. Ananthanarayanan, P. Bodik, M. Philipose, P. Bahl, and M. J. Freedman, "Live video analytics at scale with approximation and delay-tolerance." in *NSDI*, vol. 9, 2017, p. 1.

[13] K. Hsieh, G. Ananthanarayanan, P. Bodik, S. Venkataraman, P. Bahl, M. Philipose, P. B. Gibbons, and O. Mutlu, "Focus: Querying large video datasets with low latency and low cost," in *13th USENIX Symposium on Operating Systems Design and Implementation (OSDI 18)*, 2018, pp. 269–286.

[14] H. Shen, L. Chen, Y. Jin, L. Zhao, B. Kong, M. Philipose, A. Krishnamurthy, and R. Sundaram, "Nexus: A gpu cluster engine for accelerating dnn-based video analysis," in *Proceedings of the 27th ACM Symposium on Operating Systems Principles*, 2019, pp. 322–337.

[15] S. Hu, W. Bai, G. Zeng, Z. Wang, B. Qiao, K. Chen, K. Tan, and Y. Wang, "Aeolus: A building block for proactive transport in datacenters," in *Proceedings of the 2020 Conference of the ACM Special Interest Group on Data Communication*, 2020, pp. 422–434.

[16] J. Zhang, W. Bai, and K. Chen, "Enabling ecn for datacenter networks with rtt variations," in *Proceedings of the 15th International Conference on Emerging Networking Experiments And Technologies*, 2019.

[17] B. Zhang, X. Jin, S. Ratnasamy, J. Wawrzynek, and E. A. Lee, "AWStream: Adaptive wide-area streaming analytics," in *Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication*. ACM, 2018, pp. 236–252.

[18] C. Canel, T. Kim, G. Zhou, C. Li, H. Lim, D. G. Andersen, M. Kaminsky, and S. R. Dulloor, "Scaling video analytics on constrained edge nodes," in *2nd Conference on Systems and Machine Learning (SysML)*, 2019.

[19] T. Y.-H. Chen, L. Ravindranath, S. Deng, P. Bahl, and H. Balakrishnan, "Glimpse: Continuous, real-time object recognition on mobile devices," in *Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems*. ACM, 2015, pp. 155–168.

[20] H. Ding, X. Jiang, A. Q. Liu, N. M. Thalmann, and G. Wang, "Boundary-aware feature propagation for scene segmentation," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 6819–6829.

[21] S. Zhao, Y. Wang, Z. Yang, and D. Cai, "Region mutual information loss for semantic segmentation," in *Advances in Neural Information Processing Systems*, 2019, pp. 11 115–11 125.

[22] B. Chen, C. Gong, and J. Yang, "Importance-aware semantic segmentation for autonomous vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 1, pp. 137–148, 2018.

[23] Z. Wu, X. Wang, J. E. Gonzalez, T. Goldstein, and L. S. Davis, "ACE: Adapting to changing environments for semantic segmentation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[24] A. Hore and D. Ziou, "Image quality metrics: PSNR vs. SSIM," in *2010 20th International Conference on Pattern Recognition*. IEEE, 2010, pp. 2366–2369.

[25] Z. Wang, A. C. Bovik, H. R. Sheikh, E. P. Simoncelli *et al.*, "Image quality assessment: From error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.

[26] Y. Wang, W. Wang, D. Liu, X. Jin, J. Jiang, and K. Chen, "Enabling edge-cloud video analytics for robotic applications," in *IEEE INFOCOM 2021-IEEE Conference on Computer Communications*. IEEE, 2021.

[27] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 3213–3223.

[28] H. Yeo, Y. Jung, J. Kim, J. Shin, and D. Han, "Neural adaptive content-aware internet video delivery," in *13th USENIX Symposium on Operating Systems Design and Implementation (OSDI 18)*, 2018, pp. 645–661.

[29] N. Ahn, B. Kang, and K.-A. Sohn, "Fast, accurate, and lightweight super-resolution with cascading residual network," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 252–268.

[30] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in PyTorch," 2017.

[31] Y. Wang, W. Wang, J. Zhang, J. Jiang, and K. 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.

[32] D. Dai, Y. Wang, Y. Chen, and L. Van Gool, "Is image super-resolution helpful for other vision tasks?" in *Applications of Computer Vision (WACV), 2016 IEEE Winter Conference on*. IEEE, 2016, pp. 1–9.

[33] D. Guo, L. Zhu, Y. Lu, H. Yu, and S. Wang, "Small object sensitive segmentation of urban street scene with spatial adjacency between object classes," *IEEE Transactions on Image Processing*, vol. 28, no. 6, pp. 2643–2653, 2018.

[34] T. Zhang, A. Chowdhery, P. Bahl, K. Jamieson, and S. Banerjee, "The design and implementation of a wireless video surveillance system," in *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking*, 2015, pp. 426–438.

[35] Y. Kang, J. Hauswald, C. Gao, A. Rovinski, T. Mudge, J. Mars, and L. Tang, "Neurosurgeon: Collaborative intelligence between the cloud and mobile edge," *ACM SIGARCH Computer Architecture News*, vol. 45, no. 1, pp. 615–629, 2017.

[36] J. Emmons, S. Fouladi, G. Ananthanarayanan, S. Venkataraman, S. Savarese, and K. Winstein, "Cracking open the dnn black-box: Video analytics with dnns across the camera-cloud boundary," in *Proceedings of the 2019 Workshop on Hot Topics in Video Analytics and Intelligent Edges*, 2019, pp. 27–32.

[37] A. Howard, M. Sandler, G. Chu, L.-C. Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang, V. Vasudevan *et al.*, "Searching for MobileNetV3," *arXiv Preprint arXiv:1905.02244*, 2019.

[38] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1251–1258.

[39] TensorFlow, "TensorFlow DeepLab Model Zoo," https://github.com/tensorflow/models/blob/master/research/deeplab/g3doc/model_zoo.md.

[40] S. P. Chinchali, E. Cidon, E. Pergament, T. Chu, and S. Katti, "Neural networks meet physical networks: Distributed inference between edge devices and the cloud," in *Proceedings of the 17th ACM Workshop on Hot Topics in Networks*. ACM, 2018, pp. 50–56.

[41] J. Fowers, K. Ovtcharov, M. Papamichael, T. Massengill, M. Liu, D. Lo, S. Alkalay, M. Haselman, L. Adams, M. Ghandi *et al.*, "A configurable cloud-scale DNN processor for real-time AI," in *2018 ACM/IEEE 45th Annual International Symposium on Computer Architecture (ISCA)*. IEEE, 2018, pp. 1–14.

2168-7161 (c) 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: Hong Kong University of Science and Technology. Downloaded on February 25,2022 at 02:00:33 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCC.2022.3142066, IEEE Transactions on Cloud Computing

[42] B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman, "Labelme: A database and web-based tool for image annotation," *International Journal of Computer Vision*, vol. 77, no. 1-3, pp. 157–173, 2008.

[43] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *Journal of Big Data*, vol. 6, no. 1, p. 60, 2019.

[44] P. M. Grulich and F. Nawab, "Collaborative edge and cloud neural networks for real-time video processing," *Proceedings of the VLDB Endowment*, vol. 11, no. 12, pp. 2046–2049, 2018.

[45] S. Jain, X. Zhang, Y. Zhou, G. Ananthanarayanan, J. Jiang, Y. Shu, P. Bahl, and J. Gonzalez, "Spatula: Efficient cross-camera video analytics on large camera networks," in *2020 IEEE/ACM Symposium on Edge Computing (SEC)*. IEEE, 2020, pp. 110–124.

[46] Y. Li, A. Padmanabhan, P. Zhao, Y. Wang, G. H. Xu, and R. 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*, 2020, pp. 359–376.

[47] "Zeromq," https://zeromq.org/, 2020.

[48] Y. Liu, C. Shen, C. Yu, and J. Wang, "Efficient semantic video segmentation with per-frame inference," *arXiv Preprint arXiv:2002.11433*, 2020.

[49] Autopilot Tesla. [Online]. Available: https://www.tesla.com/autopilotAI

[50] P. Zhu, L. Wen, X. Bian, H. Ling, and Q. Hu, "Vision meets drones: A challenge," *arXiv Preprint arXiv:1804.07437*, 2018.

[51] M. Orsic, "SwiftNet," https://github.com/orsic/swiftnet, 2019.

[52] P. Zhang, "SlimYOLOv3," https://github.com/PengyiZhang/SlimYOLOv3, 2019.

[53] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2980–2988.

[54] "USA Mobile Network Experience Report, July 2019," https://www.opensignal.com/reports/2019/07/usa/mobile-network-experience, 2019.

[55] K. Du, A. Pervaiz, X. Yuan, A. Chowdhery, Q. Zhang, H. Hoffmann, and J. 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*, 2020, pp. 557–570.

[56] S. Chinchali, A. Sharma, J. Harrison, A. Elhafsi, D. Kang, E. Pergament, E. Cidon, S. Katti, and M. Pavone, "Network offloading policies for cloud robotics: A learning-based approach," *arXiv Preprint arXiv:1902.05703*, 2019.

[57] M. Zhang, S. Rajbhandari, W. Wang, and Y. He, "DeepCPU: Serving rnn-based deep learning models 10x faster," in *2018 USENIX Annual Technical Conference (USENIX ATC 18)*, 2018, pp. 951–965.

[58] M. Haris, G. Shakhnarovich, and N. Ukita, "Task-driven super resolution: Object detection in low-resolution images," *arXiv Preprint arXiv:1803.11316*, 2018.

[59] T. Bolukbasi, J. Wang, O. Dekel, and V. Saligrama, "Adaptive neural networks for efficient inference," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 527–536.

[60] S. Zheng, Y. Song, T. Leung, and I. Goodfellow, "Improving the robustness of deep neural networks via stability training," in *Proceedings of the ieee conference on computer vision and pattern recognition*, 2016, pp. 4480–4488.

[61] H. Wang and C.-N. Yu, "A direct approach to robust deep learning using adversarial networks," *arXiv Preprint arXiv:1905.09591*, 2019.

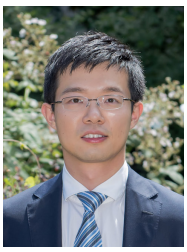
[62] A. Gupta, N. Tatbul, R. Marcus, S. Zhou, I. Lee, and J. Gottschlich, "Class-weighted evaluation metrics for imbalanced data classification," *arXiv preprint arXiv:2010.05995*, 2020.

[63] Z. Liu, T. Liu, W. Wen, L. Jiang, J. Xu, Y. Wang, and G. Quan, "DeepN-JPEG: A deep neural network favorable jpeg-based image compression framework," in *Proceedings of the 55th Annual Design Automation Conference*. ACM, 2018, pp. 1–6.

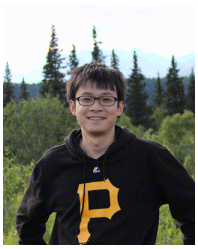
Yiding Wang is currently a Ph.D. candidate from the Department of Computer Science and Engineering, the Hong Kong University of Science and Technology. He received the B.E. degree in instrument science from Shanghai Jiao Tong University in 2017. His research interests include improving the performance of machine learning systems using machine learning techniques.

Weiyan Wang is currently a Ph.D. candidate from the Department of Computer Science and Engineering, the Hong Kong University of Science, advised by Prof. Kai Chen. Before that, he received the M.Phil. degree on Computer Software and Theory from Institution of Software, Chinese Academy of Sciences and the B.Eng. degree in Computer Science and Technology from Huazhong University of Science and Technology.

Duowen Liu is a research postgraduate student supervised by Prof. Kai Chen from the Hong Kong University of Science and Technology. He received his M.Phil. and Bachelor's degree both in Computer science in 2021 and 2017 respectively. His research interests include machine learning systems and data center networking.



Xin Jin is an Associate Professor in the Department of Computer Science and Technology at Peking University. His research is in computer systems, with a focus on hardware-software co-design, programmable networks and machine learning systems. Before joining Peking University, he was a Postdoctoral Researcher at UC Berkeley, and was an Assistant Professor at Johns Hopkins University. He received his BS in computer science and BA in economics from Peking University in 2011, and his MA and PhD in computer science from Princeton University in 2013 and 2016.



Junchen Jiang is currently an Assistant Professor at the University of Chicago. He received his PhD degree from the Computer Science Department at Carnegie Mellon University in 2017. Before that, he received Bachelor degree in Computer Science from Yao Class at Tsinghua University in 2011. His research applies state-of-the-art machine learning techniques to drastically improve the performance and reliability of large-scale networked systems.

Kai Chen is an Associate Professor with Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Hong Kong. He received his Ph.D. degree in Computer Science from Northwestern University, Evanston, IL, USA in 2012. His current research interests include data center networking, machine learning systems and privacy-preserving computing.

2168-7161 (c) 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: Hong Kong University of Science and Technology. Downloaded on February 25,2022 at 02:00:33 UTC from IEEE Xplore. Restrictions apply.