SwitchingNet: Edge-Assisted Model Switching for Accurate Video Recognition over Best-Effort Networks

Florian Beye, Yasunori Babazaki, Ryuhei Ando, Takashi Oshiba, Koichi Nihei, and Katsuhiko Takahashi
NEC Corporation
Kawasaki, Japan
{f-beye, y_babazaki, ryu-ando, oshiba, knihei, katsuhiko.takahashi}@nec.com

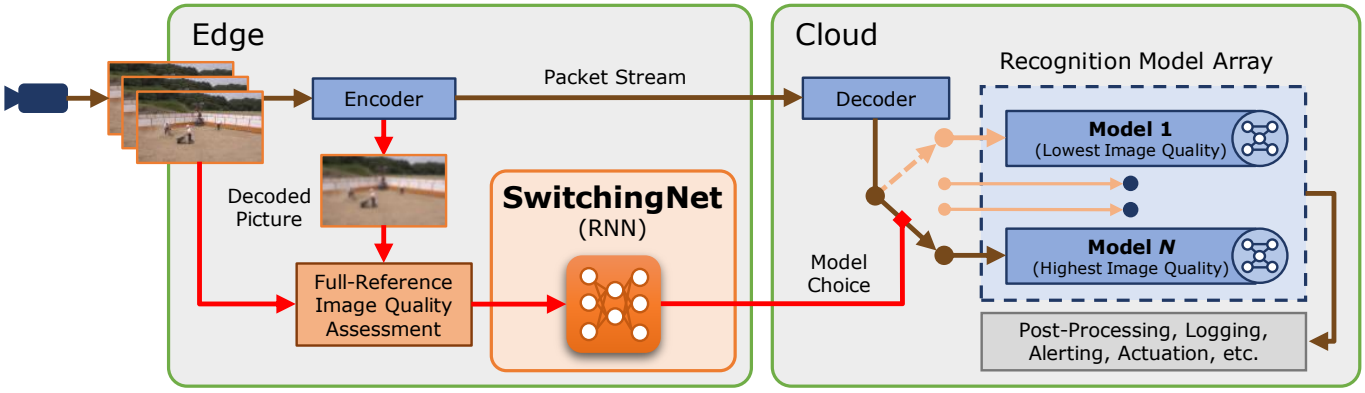

Fig. 1: The general system architecture surrounding SwitchingNet, which is trained to select the optimal inference model among an array of specialized recognition models, given a vector of full-reference image quality metrics.

***Abstract*—Despite the remarkable success of deep-learning in image and video recognition, constructing real-time recognition systems for computationally intensive tasks such as spatio-temporal human action localization is still challenging. As computational complexity of these tasks can easily exceed the capacity of edge devices, inference must be performed in remote (cloud) environments. But then, recognition accuracy is subject to fluctuating networking conditions in best-effort networks due to compression artefacts incurred from low-bitrate video streaming. To improve overall recognition accuracy under various networking conditions, we propose SwitchingNet, an edge-assisted inference model switching method. In SwitchingNet, we train multiple recognition models specialized towards different levels of image quality and a neural switching model for dynamically choosing among the specialized recognition models during system operation. Switching decisions are made at the edge given an image quality vector calculated from compressed and uncompressed frames. In the experiments, we show that our approach can on average sustain higher recognition accuracy than plain recognition systems under heavily fluctuating networking conditions. Also, our switching-based recognition approach is far less computationally intensive than competing ensemble methods and allows to significantly reduce cloud computing costs.**

I. Introduction

Deep learning has brought enormous advancements in the area of image and video recognition. Today, many openly available deep-learning-based implementations of recognition tasks such as object detection (e.g. [1]) have reached levels of accuracy which can be regarded sufficient for various real-world applications. Such applications include for example automated surveillance, with actual tasks ranging from e.g. license plate recognition [2] to livestock counting [3]. Furthermore, for simple recognition tasks and low image resolutions these applications have become practical, as inference complexity has arrived at levels at which smartphones, AI cameras or AI-enabled edge devices are sufficient for attaining execution speeds considered real-time.

Despite these successes, high inference complexity still remains an issue in many advanced cases. For example, one such advanced case occurs when a more sophisticated and computationally challenging recognition task such as spatio-temporal human action localization [4], [5] is demanded. In contrast to comparatively simple single-shot object detection (e.g. [1]), the complexity of human action detection often scales linearly with the number of persons in a scene (e.g. in an application of [6]). Moreover, the desire to increase image resolutions (for recognition at smaller scale and larger distance) or frame-rates further increases the computational cost.

In these computationally demanding cases, inference computation is inevitably pushed from the edges to remote (e.g. cloud) environments, which implies streaming compressed video between edge and remote environment over communication networks [7].

Then, however, application QoE (i.e. recognition accuracy) is greatly influenced by networking conditions. This is because as video is streamed from edge devices to the cloud, it is

compressed by video codecs such as H.265 [8] to a given bitrate that is limited by network throughput. Compression deteriorates image quality (due to compression artefacts), and image quality deterioration causes recognition errors [9], [10] which become more severe as bitrate decreases. Hence, as in many real-world cases networking includes wireless segments and QoS is only *best-effort*, drops in bitrate and thus also recognition accuracy must be expected.

One approach to reduce the influence of the networking conditions on recognition accuracy is to train recognition models such that they become more robust against compression artefacts. For example, the recognition model's training data can be augmented with images artificially distorted by video codecs at lower bitrates. This improves recognition accuracy at lower bitrates because the model becomes accustomed to compression artefacts.

However, doing so adversely affects recognition accuracy at higher bitrates, as overall accuracy is limited by the recognition model's capacity (i.e. the size of the neural network).

To overcome this, increasing the model capacity is a valid approach. For example in [11], an ensemble model combining recognition models specialized for different bitrate regions is proposed.

However, increasing recognition model capacity also increases inference complexity, and hence incurs additional cloud and networking costs. For example, the ensemble approach in [11] requires at least twice the amount of computation of a single recognition model. Depending on the use-case such a rise in cost can be detrimental to profitability and hence using larger models often ruled out as an option.

Therefore, improving recognition accuracies over a wide range of video streaming bitrates without excessively increasing computational costs necessitates other approaches.

To this end, we propose SwitchingNet, which is an edge-assisted inference model switching method (the general system architecture is shown in Fig. 1). SwitchingNet assumes an array of recognition models, where each model is specialized towards a specific image quality range by training with video data distorted at different compression strengths. During system operation, we dynamically switch between models on a frame-by-frame basis. To do so, we first calculate an image quality vector by comparing distorted and undistorted video frames. As this calculation requires access to the uncompressed frame it must necessarily be performed at the edge device. Then, we employ a recurrent neural network (RNN) for selecting a specific recognition model from the array given the calculated image quality vector. This RNN is trained towards outputting the optimal choice of recognition model by means of an unsupervised method inspired from reinforcement-learning.

Due to our model switching approach, inference becomes more robust against compression artefacts at various degrees and recognition accuracy is improved overall. In the experiments, we show this by evaluating recognition accuracy in a scenario where bitrate fluctuates heavily, simulating the effect of changing networking conditions.

Furthermore, in contrast to ensemble approaches such as [11], our approach has the advantage that only a single recognition model has to be evaluated at a time, and hence inference complexity is greatly reduced. However, in comparison to a plain inference system (only a single model and no switching), computational complexity is increased by image quality calculation and RNN inference. However, this additional amount of computation is small, and as we show in the experiments, can easily fit within AI-enabled or GPU-enabled edge devices.

The paper is structured as follows. In Section II we discuss other works related to this paper. Our method and system is then described in Section III, and the results of our experimental evaluation are presented in Section IV. Section V concludes our paper.

II. Related Work

Many previous works consider the impact of video compression and networking in remote recognition systems, and several methods for improving overall recognition accuracy have been proposed.

Most methods for improving accuracy can be classified into two categories: (a) controlling or enhancing compression with recognition-awareness in order to use available bandwidth resources more efficiently without impairing accuracy, and (b) making recognition more robust against the issues caused by compression or otherwise exploiting information from compressed video streams.

The former category of methods includes region-of-interest (RoI) based approaches such as [12]–[15]. These approaches for example perform fast object detection on an edge device and reduce image quality and bitrate outside these detected regions, which turns out to not be detrimental to accuracy for some tasks. Another approach [16] proposes a recognition-aware image preprocessing stage which alters image content before compression. Furthermore, approaches such as those in [17]–[19] propose replacing or augmenting traditional codecs such as H.265 with recognition-aware neural networks in order to improve accuracy. Other methods such as [20], [21] try to maximize overall recognition accuracy in the presence of multiple video streams by improving the allocation of bandwidth resources among streams in a recognition-aware manner. Another approach [22], optimizes encoder bitrates and resolutions dynamically based on video content in order to improve accuracies.

The latter category of methods includes approaches such as those described in [11], [23]–[25] which improve recognition accuracies at model side for example by using additional information which can be obtained from the decoder (such as motion vectors), or by ensemble techniques and data augmentation methods. Among these works, in particular [23], [24] target the spatio-temporal action recognition task, which we also adopt in our experiments.

III. Method and System

In this work we consider the remote recognition system architecture shown in Fig. 1. It contains functionality common

to video streaming (encoding and decoding) and recognition, but also implements our model switching approach which is broken down into three parts:

1) Image quality assessment which generates an image quality vector.
2) Computing a model choice based on the generated image quality vector using an RNN. This RNN is trained towards making optimal selection decisions among N recognition models using an unsupervised method inspired by reinforcement-learning.
3) Transmitting the model choice from edge to cloud alongside the packet stream of the encoded video, and using the chosen model for inference.

Besides what is shown in Fig. 1, we also assume that video streaming is adaptive, i.e. that encoder bitrate is controlled as to not surpass the maximum achievable throughput. This can be achieved for example by predicting throughput using methods such as the one proposed in [26].

In the following sections we describe in detail our approaches to image quality assessment, neural recognition model selection and unsupervised training.

A. Image quality assessment

In this work, we propose using full-reference image quality metrics as basis for model switching. In general, image quality can be assessed using full-reference or no-reference metrics. In the context of compression, full-reference metrics such as PSNR (peak signal-to-noise ratio) measure image quality by comparing the compressed image with the uncompressed image. In contrast to this, no-reference metrics such as the one proposed in [27] determine image quality by looking at the compressed image only. While in our context, no-reference metrics would seem to be advantageous because they can be evaluated in the cloud only without putting computational load on the edge device, these metrics themselves make use of large neural networks and thus negatively affect cloud costs. Therefore, we resort to full-reference metrics and calculate them on the edge device.

Furthermore, in order to allow the RNN to make better decisions, a vector consisting of multiple full-reference metrics capturing several aspects of image quality is calculated and passed to the RNN. In particular, given the uncompressed image x_{cij} and the compressed image $\tilde{x}_{cij}$ (c denotes the RGB color component, i and j denote the vertical and horizontal pixel indices ranging from $1 \dots H$ and $1 \dots W$, respectively), we first consider the root mean squared error (RMSE):

$$\mathrm{RMSE}(\tilde{x}, x) = \sqrt{\frac{1}{3 \cdot H \cdot W} \sum_{cij} (\tilde{x}_{cij} - x_{cij})^2} \,. \tag{1}$$

The RMSE serves as a basic metric for measuring the deviation between uncompressed and compressed images.

Furthermore, to quantify spatial variation of image quality within an image, we propose the following LRMSE metric, which is calculated by first applying Laplacian filters to the compressed and uncompressed images, and then calculating the RMSE:

$$\mathrm{LRMSE}(\tilde{x}, x) = \mathrm{RMSE}(L(\tilde{x}), L(x)) \,, \tag{2}$$

where L denotes a Laplacian filter.

Finally, to capture image quality across different scales we downscale the images to respectively $1/2$, $1/4$, $1/8$ and $1/16$ of the original resolution, and calculate RMSE and LRMSE also for these respective scales. This leads to an image quality vector I_m of in total 10 components, and captures multiple aspects of image quality.

B. Stateful model switching

In order to optimally select a recognition model from an image quality vector, we use a *recurrent* (stateful) neural network model (henceforth called switching model). This is because many advanced video recognition models are also stateful by design. For example, a video recognition model may consider a sliding window of video frames as input for each time step, or consist of recurrent neural network models by itself. Hence, considering image quality on a frame-by-frame basis is not sufficient as evidence to make optimal model selection decisions, as recognition accuracy will to some extent also depend on the history of image qualities.

To this end, we propose feeding the ten-dimensional image quality vector calculated as described in the previous Section into a three-layer stacked LSTM (Long Short Term Memory) with 64 feature and output dimensions. The outputs of the LSTM are further fed into a three layer MLP (multi layer perceptron) with 64 feature dimensions and softmax activations in the final layer, in order to produce the final output of the model at each time step. This final output is a N-dimensional vector assigning a probability p_n to each of the N recognition models.

Then, during system operation, we choose the model n with highest probability p_n for video recognition inference.

Finally, note that when the recognition models are stateful the state itself must be shared between the N models. When the recognition model architecture is for example based on sliding windows of input frames this sharing can be straightforward. When the architecture however contains hidden (learned) states then extra care must be taken during training in order to ensure states from different models are compatible with each other.

C. Unsupervised training

In order to train the parameters of our switching model (denoted by θ), we use an actor-critic method inspired by the A3C algorithm [28]. The training goal is to make the output probabilities p_n maximise an expectation value involving a recognition accuracy metric. Similar to A3C which is based on policy gradients, our method tries to improve a reward-based loss function $\mathcal{L}(\theta)$ using gradient descent updates.

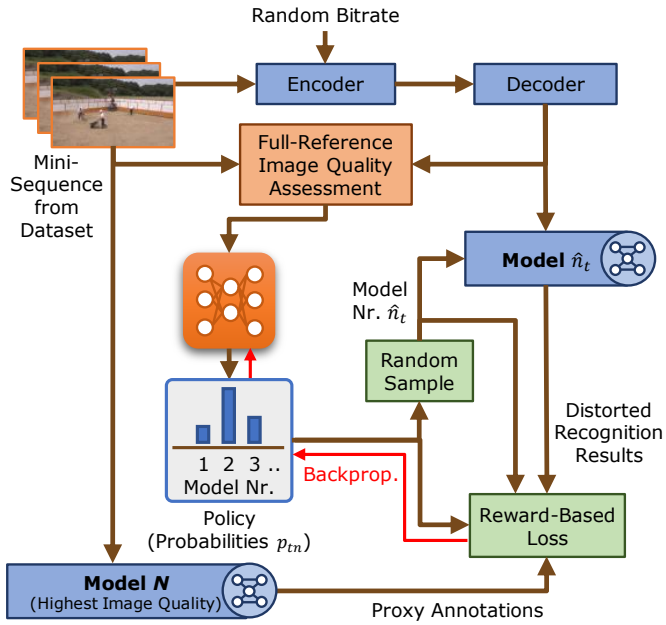

Fig. 2: Reinforcement training apparatus used for automated tuning of the SwitchingNet RNN model.

Below, we describe the calculation of this loss function, which is given by:

$$\mathcal{L}(\theta) = - \mathop{\mathbb{E}}_{x,\tilde{x},\hat{n}} \sum_t (R_t(x,\tilde{x},\hat{n}|\phi^*) - V_t(x,\tilde{x}|\phi^*)) \log p_{t,\hat{n}_t}(\theta) \,. \tag{3}$$

A simplified computational flow for evaluating $\mathcal{L}$ is schematically depicted in Fig. 2.

First, in Eqn. 3, the expectation $\mathbb{E}$ is over triples $(x,\tilde{x},\hat{n})$, where x and $\tilde{x}$ denote uncompressed and compressed video sequences and $\hat{n}$ denotes a time series of model choices (selections), respectively. The expectation and is approximated by a fixed number (the batch size) of samples which are generated as follows.

1) First a short uncompressed sequence x_{tcij} (with t denoting time) is sampled from a training dataset.
2) Then, this uncompressed sequence is fed into an encoder/decoder compression pipeline which outputs the compressed sequence $\tilde{x}_{tcij}$. In order to generate compressed video sequences at various qualities, we set the bitrate target parameter to a random value sampled from a predefined distribution.
3) Next, a time series I_{tm} of image quality vectors is calculated from the video sequences x_{tcij} and $\tilde{x}_{tcij}$ as described in Section III-A.
4) The switching model is applied to I_{tm}, giving a time series of probability vectors p_{tn} (the selection policies).
5) From these probability vectors, we sample the time series of model choices $\hat{n}_t$.

Next, we explain the calculation of the returns R_t and the value estimation V_t from the sampled triples. The returns are defined as a discounted sum of the rewards r_t.

$$R_t = \sum_{s=t}^{T-1} \gamma^{s-t} r_s + \gamma^{T-t} V_T \,, \tag{4}$$

where $0 < \gamma < 1$ is a factor reducing the influence of future rewards on the training of the model choice at time t, and T is the sequence length.

Now, to calculate the rewards r_t at each time step t, first the uncompressed video sequence x is fed through the N-th recognition model, which is assumed to have been *trained for the highest image quality*. This gives a time series of recognition results which we treat as a proxy for ground truth annotations. This makes our training algorithm *unsupervised*, i.e. manual ground truth annotations are not required, which facilitates tuning the switching model towards new data in an automated fashion. We also found that when the N-th recognition model is accurate enough, then using these proxy annotations instead of manual ground truth annotations led to similar results accuracy-wise.

Next, we feed the compressed video sequence $\tilde{x}$ through the series of models specified by $\hat{n}$, which gives a time series of (distorted) recognition results. These recognition results are compared to the proxy annotations by means of a recognition accuracy measure suiting the task (we give an example in Section IV-C) evaluated on a frame-by-frame basis, and these per-frame accuracies are treated as the rewards r_t.

Furthermore, in Eqn. 3 an estimation V_t of the *value* $\mathbb{E}[R_t]$ is subtracted from the returns R_t. According to common theory in reinforcement learning, subtracting the expectation of R_t leads to variance reduction and improves training convergence. Here, in order to estimate the expectation as $V_t(x,\tilde{x}|\phi^*)$, we employ a combination of a convolutional neural network (CNN), LSTM and MLP which inputs x and $\tilde{x}$ and is parametrized by ϕ. These networks are only used during training and are not relevant for actual model switching. The parameters ϕ are trained simultaneously with θ using gradient descent updates of the L2 loss function $\mathcal{L}_V(\phi)$ which is defined as:

$$\mathcal{L}_V(\phi) = \mathop{\mathbb{E}}_{x,\tilde{x},\hat{n}} \sum_t (V_t(x,\tilde{x}|\phi) - R_t(x,\tilde{x},\hat{n}|\phi^*))^2 \,. \tag{5}$$

In the above, ϕ^* is not subject to gradient computation and denotes the value of ϕ obtained from the previous gradient descent iteration.

With the given calculation of above loss functions, gradient descent updates are performed by means of backpropagation. In order to further speed up training convergence, we additionally employ the ADAM algorithm [29].

IV. Experiments

A. Recognition task and dataset

In our experiments, we consider the recognition task of spatio-temporal human action localization. This task includes detecting and tracking persons in video, as well as classifying the actions (activities) of each tracked person. In particular,

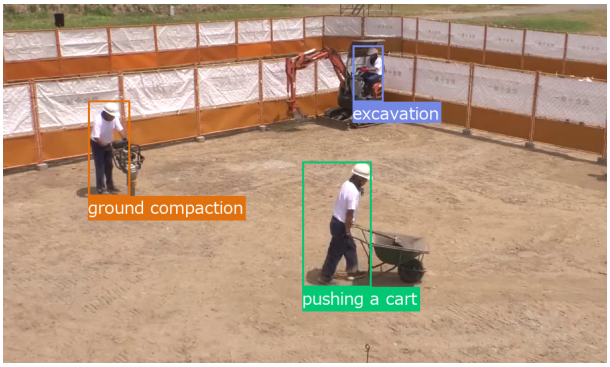

Fig. 3: Example image from our dataset with localized human actions.

we consider detecting the following activities commonly performed at construction sites: pushing a cart, ground leveling, distance measurement, excavation and ground compaction.

For recognition model training and for training our proposed switching model, we create a private dataset of surveillance videos showing persons performing above actions (up to 5 persons simultaneously). The dataset contains over 1 hour of annotated high-quality 1080p material recorded at 30 frames per second, where annotations cover localization (bounding boxes), tracking (person IDs) and classification (action labels). For the purpose of final testing we reserve 10 video sequences totalling 6 minutes, and use the remaining material for training.

As an example, we show an image from the dataset with localized actions in Fig 3.

B. Accuracy metrics

We have to define accuracy metrics suited to our recognition task for the purpose of performance evaluation and comparison, but also for the purpose of defining the rewards r_t used during unsupervised training of the switching model.

In the following, we make use of *recall* and *precision* as accuracy metrics. They are defined as:

$$\text{Recall} = \frac{\text{TP}}{\text{TP}+\text{FN}}, \qquad \text{Precision} = \frac{\text{TP}}{\text{TP}+\text{FP}}. \tag{6}$$

Here, TP denotes the number of correctly detected (i.e. localized and classified) actions, FN counts the actions which are present in the annotations but undetected, and FP is the number a detected actions which are not present in the annotations.

In addition, to take care of the edge cases, we augment the definitions in Eqn. 6 when the denominators vanish. Then, recall and precision are set to one.

C. Model training

Our approach requires an array of specialized recognition models suitable for our task at hand. To this end, we build upon a model architecture for spatio-temporal action localization based on the work in [6]. This model architecture internally breaks action localization down into the subtasks of human detection and tracking, detection of objects relevant to the actions (such as excavators, carts etc.), feature extraction and action classification. After detection and tracking, the contents of the detected bounding boxes are cropped from the input frames. Then, after extracting features from these crops, actions are classified by what is mostly an LSTM operating on two-second temporal sliding windows of the extracted features.

In our experiments, we consider the special case of using a pair of specialized action localization models, which are trained as follows:

- *High-quality model*. This model is trained using our dataset as-is, i.e. without any artificial distortion using compression.
- *Low-quality model*. For this model, we retrain the action classification part by making use of compressed videos. To generate these videos, we use a software encoder [30] with a random quantization parameter (QP) chosen uniformly from the range 37 to 51. Note, that for this encoder QP = 1 produces highest quality and QP = 51 produces lowest quality, so our generated videos are on the lower end of the quality spectrum.

Furthermore, we train our switching model as described in Section III-C using above dataset for a total of 8000 gradient descent iterations. During training, we sample video sequences with a length of 5 seconds from the dataset. Rewards are defined to be the negative of the computed recall value, and the discount factor is set to $\gamma = 0.8$. As batch size we choose 32, and ADAM is used with learning rate 0.001.

TABLE I: Recall and precision of our method and the baselines in a simulated scenario of dynamically changing bitrate.

Method	Recall (%)	Precision (%)
Low-Quality Model	85.3	87.6
High-Quality Model	83.0	85.1
SwitchingNet (Ours)	**87.3**	**89.7**

D. Evaluation setup

All testing is done using the built-in H.265 hardware encoder of a 20W-class edge device of type Jetson Xavier NX. This encoder allows us to set a bitrate value and dynamically change it at arbitrary points in time.

In the evaluations, we use this encoder for distorting our test set in two different ways. First, we generate 15 compressed datasets by encoding the test set with 15 different constant bitrate settings approximately covering the range from 200 kbps to 4000 kbps. Second, we consider a simplified simulation of a best-effort network (such as the uplink direction of an LTE network) where the bitrate setting dynamically transitions between two states of approximately 300 kbps and 3000 kbps according to a Markov chain. In this case, we simulate the Markov chain only once, and compress all 10 videos in the test set according to the same simulation result to generate the compressed dataset.

E. Accuracy evaluation and comparison

First, we examine the recognition performance of the generated recognition models at constant bitrate using the 15

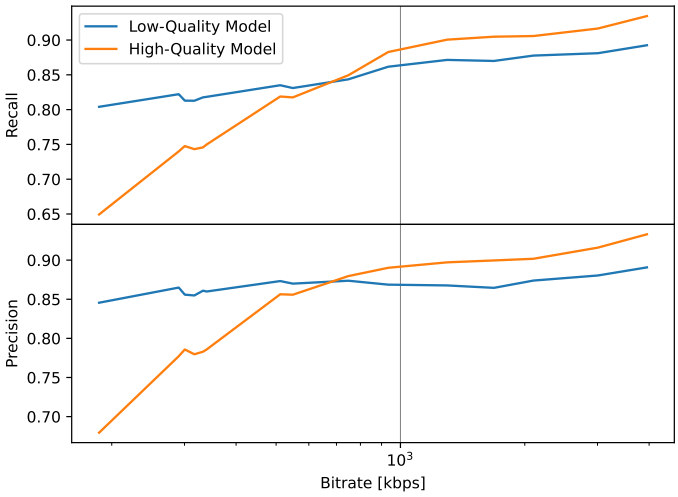

Fig. 4: Recall and precision of the baseline low-quality and high-quality models at different constant bitrate settings.

compressed datasets generated as described in Section IV-D. From the result shown in Fig. 4, it is evident that on average each model is specialized to a specific range of bitrates, which is exactly what we intended.

Next, we compare our approach to the baseline approaches, i.e. to each single model without switching. For this we use the dataset compressed at dynamically changing bitrate. The results shown in Table I indicate that our method overall performs better than the baselines in both recall and precision. Also, when we examine the evolution of precision and recall over time as shown in Fig. 5 (the evolution is averaged over the 10 sequences in the test set), we can see that our method performs best or close to best most of the time. Especially when compared with the high-quality model, our method mostly causes less severe accuracy drops when in the low-bitrate state.

Also, note, that in Fig. 5 the specialized models are not always best in the regime where they are meant to. This is because the result in Fig. 4 is only valid *on average* over all video sequences and time steps, but generally not for each individual frame. In theory, such reversals of speciality occur most frequently in the boundary region between the low-bitrate and high-bitrate regime. While this result may sound like an obstacle for our approach it actually also allows our model to surpass the average performance of both baseline models. This is apparent around seconds 10-12 and 20-22, where our model performs best in comparison to the baselines. Finally, note that our model is never worse than both baseline models at the same time.

F. Computational overhead

To show that the computational overhead incurred by our approach is low enough to fit the edge device described above, we empirically measure the total execution time incurred by image quality assessment and switching model inference. The result is that execution time is on average 17 milliseconds per

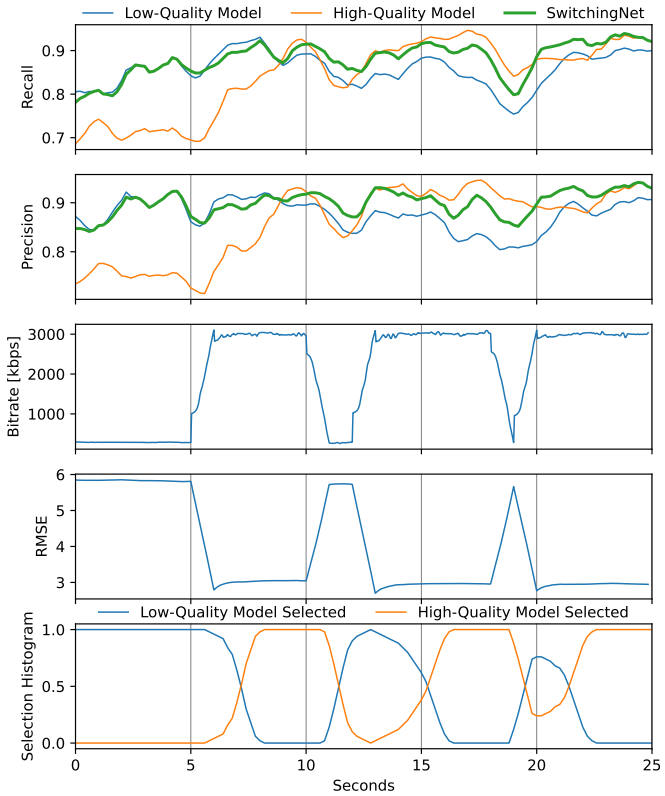

Fig. 5: Recall, precision, bitrate, RMSE, and the selection histogram measured over time (one second sliding window average to reduce variance) but averaged over the 10 sequences in the test set. Recall and precision are shown for our approach and the baseline approaches. The selection histogram indicates the relative frequency in our approach.

1080p frame, which leaves some room on the edge device for other AI tasks even when our approach is applied at 30 frames per second. Furthermore, note, that all neural networks in this paper are implemented using plain PyTorch [31], and no special inference optimizations are used, so there is still room for improvement.

V. Conclusion

In this work, we considered the interaction between video compression and video recognition in a remote recognition system. To improve the accuracy of the AI inference in the cloud, we proposed an edge-assisted method which switches between different inference models trained for different video compression qualities. In comparison to existing methods which rely on training data augmentation and increasing model sizes, our method does not raise cloud costs in exchange for a small computational overhead at the edge device.

Finally, in our experiments, we show that in a scenario simulating throughput drops in best-effort networks, overall our method surpasses the baseline methods which do not involve model switching.

REFERENCES

[1] A. Bochkovskiy, C. Wang, and H. M. Liao, "Yolov4: Optimal speed and accuracy of object detection," *CoRR*, vol. abs/2004.10934, 2020.

[2] Q. Zhang, H. Sun *et al.*, "Edge video analytics for public safety: A review," *Proceedings of the IEEE*, vol. 107, no. 8, pp. 1675–1696, 2019.

[3] B. Xu, W. Wang *et al.*, "Automated cattle counting using mask r-cnn in quadcopter vision system," *Computers and Electronics in Agriculture*, vol. 171, p. 105300, 2020.

[4] Y. Zhu, X. Li *et al.*, "A comprehensive study of deep video action recognition," *CoRR*, vol. abs/2012.06567, 2020.

[5] C. Feichtenhofer, H. Fan *et al.*, "Slowfast networks for video recognition," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[6] C.-Y. Ma, A. Kadav *et al.*, "Attend and interact: Higher-order object interactions for video understanding," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[7] S. Duan, D. Wang *et al.*, "Distributed artificial intelligence empowered by end-edge-cloud computing: A survey," *IEEE Communications Surveys & Tutorials*, vol. 25, no. 1, pp. 591–624, 2023.

[8] G. J. Sullivan, J. Ohm *et al.*, "Overview of the high efficiency video coding (HEVC) standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1649–1668, 2012.

[9] M. Poyser, A. Atapour-Abarghouei, and T. P. Breckon, "On the impact of lossy image and video compression on the performance of deep convolutional neural network architectures," in *2020 25th International Conference on Pattern Recognition (ICPR)*, 2021, pp. 2830–2837.

[10] A. Otani, R. Hashiguchi *et al.*, "Performance evaluation of action recognition models on low quality videos," *IEEE Access*, vol. 10, pp. 94 898–94 907, 2022.

[11] J. Lee, "Deep learning ensemble with data augmentation using a transcoder in visual description," *Multimedia Tools and Applications*, vol. 78, no. 22, pp. 31 231–31 243, Nov. 2019.

[12] Y. Shinohara, H. Itsumi *et al.*, "Video compression estimating recognition accuracy for remote site object detection," in *2020 International Wireless Communications and Mobile Computing (IWCMC)*, 2020, pp. 285–290.

[13] S. Milani, R. Bernardini, and R. Rinaldo, "A saliency-based rate control for people detection in video," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2013, pp. 2016–2020.

[14] H. Choi and I. V. Bajic, "High efficiency compression for object detection," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 1792–1796.

[15] L. Galteri, M. Bertini *et al.*, "Video compression for object detection algorithms," in *2018 24th International Conference on Pattern Recognition (ICPR)*, 2018, pp. 3007–3012.

[16] S. Suzuki, M. Takagi *et al.*, "Image pre-transformation for recognition-aware image compression," in *2019 IEEE International Conference on Image Processing (ICIP)*, 2019, pp. 2686–2690.

[17] F. Beye, H. Itsumi *et al.*, "Recognition-aware deep video compression for remote surveillance," in *2022 IEEE International Conference on Image Processing, ICIP 2022*. IEEE, 2022, pp. 1986–1990.

[18] Z. Yang, Y. Wang *et al.*, "Discernible image compression," in *Proceedings of the 28th ACM International Conference on Multimedia*. Association for Computing Machinery, 2020, p. 1561–1569.

[19] Y. Tian, G. Lu *et al.*, "A coding framework and benchmark towards compressed video understanding," *CoRR*, vol. abs/2202.02813, 2022.

[20] F. Beye, Y. Shinohara *et al.*, "Recognition-aware bitrate allocation for ai-enabled remote video surveillance," in *2023 IEEE 20th Consumer Communications & Networking Conference (CCNC)*, 2023, pp. 158–163.

[21] S. G. Davani and N. J. Sarhan, "Experimental analysis of optimal bandwidth allocation in computer vision systems," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 31, no. 10, pp. 4121–4130, 2021.

[22] T. Murad, A. Nguyen, and Z. Yan, "Dao: Dynamic adaptive offloading for video analytics," in *Proceedings of the 30th ACM International Conference on Multimedia*, ser. MM '22. Association for Computing Machinery, 2022, p. 3017–3025.

[23] C.-Y. Wu, M. Zaheer *et al.*, "Compressed video action recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[24] Z. Shou, X. Lin *et al.*, "Dmc-net: Generating discriminative motion cues for fast compressed video action recognition," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 1268–1277.

[25] H. Itsumi, F. Beye *et al.*, "Edge cloud ensemble with motion vectors for object detection in wireless environments," in *ICC 2021 - IEEE International Conference on Communications*. IEEE, 2021, pp. 1–6.

[26] H. Yoshida, K. Satoda, and T. Murase, "Constructing stochastic model of tcp throughput on basis of stationarity analysis," in *2013 IEEE Global Communications Conference (GLOBECOM)*, 2013, pp. 1544–1550.

[27] J. C. Mier, E. Huang *et al.*, "Deep perceptual image quality assessment for compression," in *2021 IEEE International Conference on Image Processing (ICIP)*, 2021, pp. 1484–1488.

[28] V. Mnih, A. P. Badia *et al.*, "Asynchronous methods for deep reinforcement learning," in *Proceedings of The 33rd International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, M. F. Balcan and K. Q. Weinberger, Eds., vol. 48. PMLR, 20–22 Jun 2016, pp. 1928–1937.

[29] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations, ICLR 2015*, 2015.

[30] Videolan. (2021) x265, the free h.265/hevc encoder. [Online]. Available: https://www.videolan.org/developers/x265.html

[31] A. Paszke, S. Gross *et al.*, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle *et al.*, Eds. Curran Associates, Inc., 2019, pp. 8024–8035.