ECNet: A Fast, Accurate, and Lightweight Edge-Cloud Network System based on Cascading Structure

Libo Hu†
Waseda University
korikihaku@akane.waseda.jp

Tao Wang†
Waseda University
wangtao@akane.waseda.jp

Hiroshi Watanabe†
Waseda University
hiroshi.watanabe@waseda.jp

Shohei Enomoto‡
NTT *Software Innovation Center*
shohei.enomoto.ab@hco.ntt.co.jp

Xu Shi‡
NTT Software Innovation Center
xu.shi.ca@hco.ntt.co.jp

Akira Sakamoto‡
NTT Software Innovation
akira.sakamoto.ax@hco.ntt.co.jp

Takeharu Eda ‡
NTT Software Innovation Center
takeharu@acm.org

***Abstract*—The pervasiveness of "Internet-of-Things" in daily life has led to a recent surge in fog computing, encompassing a collaboration of cloud computing and edge intelligence. As a significant field of IoT, real-time detection and classification have a huge demand. Due to the insufficiency of computing power in mobile devices and the increment of network bandwidth, combination of edge devices and cloud servers would be an accessible orientation for real-time tasks. In this work, we present ECNet—an edge-cloud network system dealing with the balance between performance and time cost. We propose to transmit the output feature map of an exit point to the cloud with offload controller and quantizer deployed to minimize the transmission cost. ECNet is tested to reach a balance between processing time and accuracy performance with reducing transmission cost down to 25%. We also consider implementing an integrated feature map encoder to further reduce the bandwidth demand and meanwhile minimize the loss of accuracy. Additional achievements could be expected in our future work.**

Keywords—Edge-cloud system, ECNet, model cascading, classification, encoding.

I. Introduction

In recent years, edge and cloud cooperative approach for object detection has been proposed [1]. Thanks to advancements in both hardware and deep learning technology, even deeper networks which further improve the classification performance have been emerged. The integration of deep neural networks can greatly enhance the functions of edge device, however, the rapid increase in runtime and power for gains in accuracy may deepen the neural network which become less tractable in many real-time situations where latency and energy costs are important factors

The current state of deep learning systems on edge devices leaves an unsatisfactory result mainly because of the gap of computation power between edge devices and cloud servers. It is prone to sacrifice either processing time or inference accuracy. Besides, the step of offloading image data to a large model in the cloud will easily lead to associated communication costs, latency issues and privacy concerns [2]. When meeting a real-time task with a very high data rate, the challenge lies in the demand to achieve high image throughput with a limited transmission bandwidth.

To address these problems, we consider an edge-cloud system based on cascade structure, which combines a light weight neural network on edge devices with a high-accuracy network on cloud servers. The light weight model at the edge-side can quickly output feature extraction, and also complete the inference. The offload controller takes charge of determine whether the inference result from the edge-side is satisfactory or not. The feature data that is barely satisfactory should be transmitted to the cloud-side for further processing with powerful DNN model and relatively sufficient computational resources. The initial idea of our network designment is to achieve lower computing cost than that in a DNN model, and higher accuracy performance compared with a simple model on edge devices. Further improvement of edge-side and cloud-side network is fulfilled for compatibility. Additionally, data extraction and compression module is deployed to reduce communication cost, and achieve real-time nature of our system[3].

The major contributions of this paper are:

- **Designation and implementation of the edge-cloud architecture:** Specifically modified edge-side network processes majority of inference, and exits controllable parts of the feature maps. The whole system is mean to reduce computational costs, resulting in running time saving with achieving substantial overall performance on deep learning tasks.
- **System Regulation via offload-controller:** entropy of classification result is set as threshold that operated by users to control the offload rate of feature data, thus ensure ECNet to meet customized accuracy demand.
- **Feature data compressing and encoding:** We characterize the accuracy impact of different quantization while introducing several feature space encoding method with time cost analysis.

II. Approach

A. Edge-side and cloud-side designation

YOLOv3, as the most well used network for object detection tasks, requires large computing cost at the meantime. We considered YOLOv2, the earlier version of YOLOv3, operating as the edge-side network with the advantage of lightweight [4]. In the initial construction of the edge-cloud system, we applied edge-side with Darknet19(backbone of YOLOv2), cloud-side with DarkNet53 (backbone of YOLOv3), planning to set exit point at the edge-side. Owing to the different structure of both side network, feature maps extracted at the exit point could not be directly employed in further processing at cloud-side. Therefore, we intend to reconstruct head part of both networks with same structure. This kind of distributed approach, however, is challenging for several considerations including:

- The structure of DarkNet53 is built on numerous residual block, and each of residual block contains successive 3 × 3 and 1 × 1 convolutional layers connected by one shortcut connection [5]. This structure is aiming to solve the degradation problem on deep networks. Reconstructed front part of edge-side network should avoid dividing residual block to ensure its integrity.
- In detection tasks, YOLOv3 predicts boxes at 3 different scales. The cloud-side of ECNet extracts features from those scales using a similar concept as feature pyramid networks [6]. It has good performance on small objects that are to be recognized by the detector [7]. The location of offloading feature map to cloud-side should be before the layer where starting extracting features.
- To limit computing cost and processing time at edge-side, the depth of edge-side should not be too large.

Guided by aforementioned considerations, the structure of edge-cloud network is designed after several times of trials and simulations.

B. *Measure of confidence score at edge-side exit point*

In the task of classification, we use entropy of a classification result (e.g., by softmax) as measure [8]. The entropy works as a confidence score during the simulation. When the classification result stands with high confidence, the value of entropy will be small, otherwise the value of entropy will be large. The edge-side network will transmit feature maps to the cloud when the output entropy is above a threshold set by users. Entropy is defined as

$$\text{entropy}(\boldsymbol{y}) = -\sum_{c\in C} y_c \log\, y_c$$

where $\boldsymbol{y}$ is a vector containing computed probabilities for all possible class labels and C is a set of all possible labels.

C. *ECNet*

As shown in Fig.1, the general framework of ECNet is mainly consists of Edge operation and Cloud operation. Feature maps extracted from edge-side will be transferred to cloud-side determined by offload controller. Due to the limitation of network bandwidth, the feature map should be compressed before transmission. Quantization is a popular method to accelerate neural network. Offload rate can be determined by the threshold to fit the accuracy requirement.

III. Experiment and Result

In this section, we provide additional analysis on key aspects of ECNet. We use 10 classes Imagenet dataset for experiment (10000 images for training and 3000 images for testing).

A. *Classification performance of the edge-side network*

To ensure the classification performance of the optimized edge-side network, we made evaluation and comparison with darknet19 and darknet53.

TABLE I. Classification Performance Comparation

	Rank-1(%)	***Rank-5(%)***	***Processing Time(s/frame)***
Edge-side	68.5	81.8	0.013
Darknet19 (YOLO v2)	64.3	76.4	0.006
Darknet53 (YOLO v3)	81.2	98.2	0.023

The result shows that the designed Edge-side network has considerable classification performance and the less processing time per frame than Darknet53, which is reasonable.

B. *Entropy distribution*

Entropy of each sample from our test dataset is counted to confirm the entropy distribution as shown in Fig.2. Most of feature maps outputted from edge-side has less entropy than 0.5, which confirmed the relatively reliable classification performance of our designed edge-side network. The part above 0.5 tends to be sparse distributed, which shows better classification ability is expected on the cloud-side network.

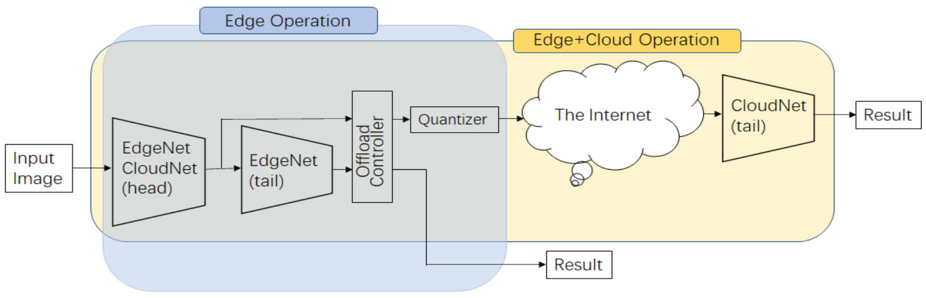

Fig.1. General framework of ECNet

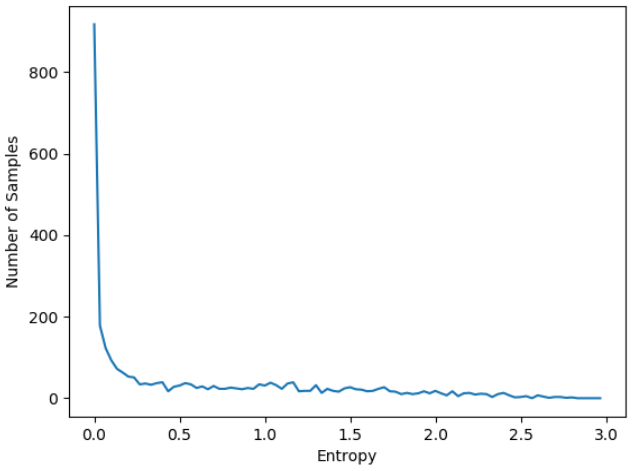

Fig. 2. Entropy distribution of feature maps

C. Accuracy impact with different quantization bits

We evaluated the overall accuracy of ECNet for varying entropy threshold with different quantization bits [9]. As shown in Fig.3 (overlap existed between the green dotted line and the original result), the int8 quantization has the least accuracy drop, the int6 and int4 quantization lead to a serious accuracy drop. The solid line in blue shows how operating of threshold affects offload rate. The threshold and offload rate satisfy negative correlation. The accuracy is tend to have little improvement when the offload rate reaches to 0.5. Thresholds should be chosen when it satisfies the latency requirement while maintaining accuracy requirement.

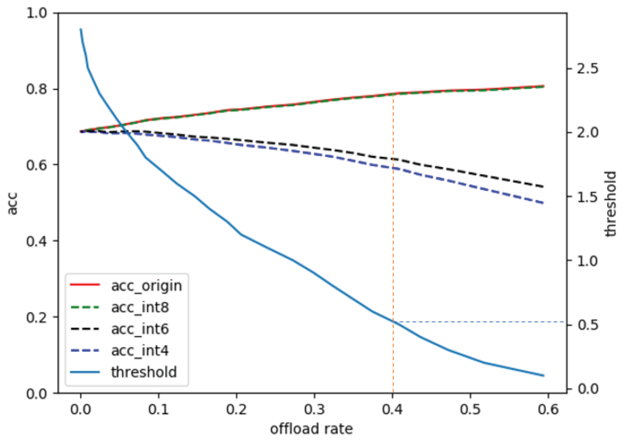

Fig. 3.Total accuracy with different quantization bits and offload control by threshold operating

D. Feature map compression attempts

Thanks to the int-8 quantization operation and offload controlling based on threshold operating, we can limit the transmission burden with little accuracy loss comparing with the original network. Since most real-time tasks have strict transmission demand, we still consider the offload performance could be unsatisfactory for real scenario, which lead to the idea of compression on feature maps after quantization to further improve energy-efficiency and throughput [10].

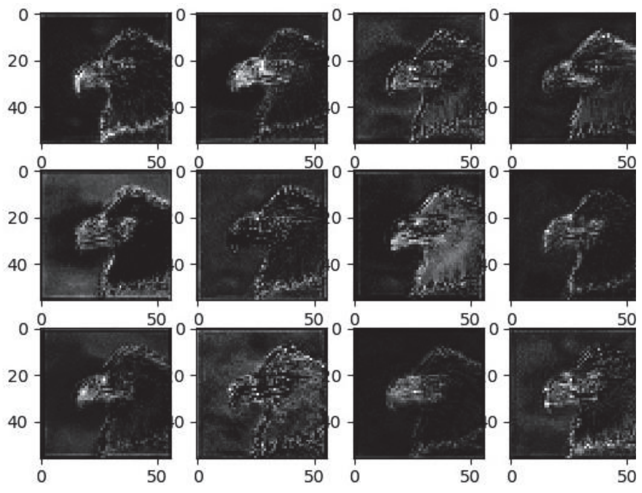

Fig. 4.Visualization of feature maps at exit point by channels

According to the visualization of the feature maps that output from the edge-side as Fig.4 shows, we can conspicuously lead to a conclusion that it seems to be less likely to apply differential pulse-code modulation (DPCM) solution in our task, since the degree of sparsity and similarity of feature data hardly meet our need, not only between channels but also between raw or column in channel. Besides, we analyze the data variation of the quantized feature maps.

TABLE II. COMPARISON OF INFORMATION VOLUME

	Raw data	*DPCM between channels*	*DPCM in channel*
Number of symbol	98.4	171.8	151.4
Entropy	4.85	5.73	4.90

TABLE Ⅱ gives the result of data analysis of quantized feature map, feature map after DPCM applied between channels and feature map after DPCM applied in channel. We use 1200 images for calculation and applying DPCM in channel for the element of each column. The result shows that DPCM operation cannot lead to a reduction of the entropy which means it cannot help reduce the amount of information in feature map.

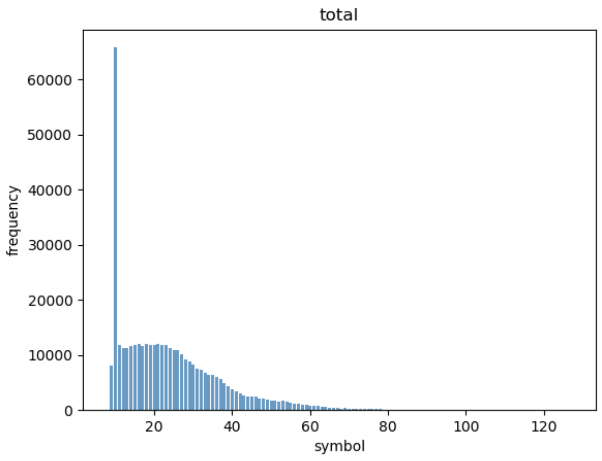

Fig. 5.Data variation of the feature map in single channel

Fig.5 shows the analysis of data variation of quantized feature maps on 100 images. Due to the int-8 quantization function and the standard attributes of the Imagenet datasets we use during our simulation, the symbol of '10' has the highest amount of frequency while other symbol satisfying a Gaussian-like distribution as shown in Fig.5. According to this data distribution factor, we propose to apply lossless compression method on whole channels like Huffman coding and ZIP compression after quantization operation [11] [12].

TABLE III. COMPRESSION PERFORMANCE EVALUATION

	Origin	*Huffman coding*	*Optimized Huffman coding*	*ZIP*
Compression ratio	1	1.62	1.50	1.68
Time cost(s/frame)	0.001	0.98	0.35	0.009

The improvement of optimized Huffman coding is using prior Huffman tree which calculate sufficient samples instead of setting Huffman tree for every feature map. Since the operation of Huffman coding is not based on matrix operation which means the encoder and decoder have to run on the whole dataset at least once a time, the time cost is increasing dramatically. As shown in TABLE III, ZIP compression method has better performance both on compression ratio (compression ratio is defined as the ratio between uncompressed size and compressed size) and time cost factors. The ZIP compression could reach to 1.68 compression ratio with 0.009 second each frame, however, the time cost is still a major concern.

Besides the method we have mentioned, we also continuing making research on other compression method such as principal component analysis (PCA) before coding, compression in neural network and JPEG compression for monochrome images. We expect further improvement on feature map compression task and completely settle the problem of time cost limitation.

IV. CONCLUSION

In this paper, we proposed ECNet with designed edge-side and cloud-side network. The edge is able to transmit quantized feature maps to the cloud, administrated by the offload controller with entropy as threshold. The improvement of ECNet is leveraged by reaching a balance between processing time and accuracy performance with reducing transmission cost down to 25%. This system has been evaluated on classification tasks and chose proper quantization bit based on experiments. For future work, we plan to adapt the ECNet to detection tasks, and make evaluation in actual scenarios.

REFERENCES

[1] Choi, H., & Bajić, I. V. Deep feature compression for collaborative object detection. In 2018 25th IEEE International Conference on Image Processing (ICIP) (pp. 3743-3747). IEEE, Oct. 2018

[2] 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," ACM Workshop on Hot Topics in Networks (HotNets2018), pp.50-56, Nov. 2018

[3] Ko, J. H., Na, T., Amir, M. F., & Mukhopadhyay, S. Edge-host partitioning of deep neural networks with feature space encoding for resource-constrained internet-of-things platforms. In 2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS) (pp. 1-6). IEEE. Nov. 2018

[4] Redmon, J., & Farhadi.A, YOLO9000: better, faster, stronger. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 7263-7271). 2017

[5] He, K., Zhang, X., Ren, S., & Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778). 2016

[6] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," arXiv preprint arXiv:1804.02767, 2018.

[7] T.-Y. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature pyramid networks for object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 2117–2125, 2017.

[8] S. Teerapittayanon, B. McDanel, and H.-T. Kung, "Branchynet: Fast inference via early exiting from deep neural networks," in IEEE International Conference on Pattern Recognition (ICPR), pp. 2464–2469. 2016

[9] Cao, S., Ma, L., Xiao, W., Zhang, C., Liu, Y., Zhang, L., ... & Yang, Z. Seernet: Predicting convolutional neural network feature-map sparsity through low-bit quantization. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 11216-1122). 2019

[10] Ko, J. H., Na, T., Amir, M. F., & Mukhopadhyay, S. Edge-host partitioning of deep neural networks with feature space encoding for resource-constrained internet-of-things platforms. In 2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS) (pp. 1-6). IEEE. Nov.2018

[11] Chmiel, B., Baskin, C., Banner, R., Zheltonozhskii, E., Yermolin, Y., Karbachevsky, A., ... & Mendelson, A. Feature map transform coding for energy-efficient cnn inference. arXiv preprint arXiv:1905.10830. 2019

[12] Cavigelli, L., Rutishauser, G., & Benini, L. EBPC: Extended bit-plane compression for deep neural network inference and training accelerators. IEEE Journal on Emerging and Selected Topics in Circuits and Systems, 9(4), 723-734. 2019