Distributed and Efficient Object Detection in Edge Computing: Challenges and Solutions

Ju Ren, Yundi Guo, Deyu Zhang, Qingqing Liu, and Yaoxue Zhang

Abstract

In the past decade, it was a significant trend for surveillance applications to send huge amounts of real-time media data to the cloud via dedicated high-speed fiber networks. However, with the explosion of mobile devices and services in the era of Internet-of-Things, it becomes more promising to undertake real-time data processing at the edge of the network in a distributed way. Moreover, in order to reduce the investment of network deployment, media communication in surveillance applications is gradually changing to be wireless. It consequently poses great challenges to detect objects at the edge in a distributed and communication-efficient way. In this article, we propose an edge computing based object detection architecture to achieve distributed and efficient object detection via wireless communications for real-time surveillance applications. We first introduce the proposed architecture as well as its potential benefits, and identify the associated challenges in the implementation of the architecture. Then, a case study is presented to show our preliminary solution, followed by performance evaluation results. Finally, future research directions are pointed out for further studies.

Introduction

By constantly capturing real-time pictures/videos of the interested area with a large number of high-resolution cameras, surveillance applications have been widely deployed in different fields ranging from traffic monitoring to national security. Since the captured real-time pictures/videos have very large sizes and high generation frequency, deploying dedicated fiber networks is a common solution to support high-throughput media data transmission for surveillance applications. Meanwhile, with the development of artificial intelligence, surveillance applications are evolving to integrate with modern machine learning mechanisms to provide enhanced functionalities, such as object detection and target tracking. In order to extract valuable knowledge from the huge amount of media data, cloud computing becomes an effective way for surveillance applications to preform massive-scale and complex data processing, In the traditional cloud-assisted architecture, end devices directly send a huge amount of real-time media data to the cloud via dedicated high-speed fiber networks, leading to a high network deployment cost. However, as the explosion of mobile devices and wireless services are driving us into the era of Internet-of-Things, this situation has gradually changed in recent years.

The amount of data generated by IoT devices has dramatically increased, posing great challenges on cloud computing in terms of response delay and communication burden. Limited by the capacity and increasing pressure of the Internet, transmitting all the data into the cloud for data processing is no longer a wise approach for delay-sensitive and large-scale surveillance applications. It consequently motivates edge computing to be a feasible complement of cloud computing to perform media data processing and analysis at the edge of the Internet [1]. By leveraging the data storage and processing capabilities of nearby infrastructures and devices, edge computing can provide distributed and timely media data analyzing services for surveillance applications [2]. Meanwhile, with the development of wireless communication technologies, transmitting bandwidth-hungry media resources via advanced wireless networks becomes a potential candidate to greatly reduce the network deployment cost of surveillance applications. These advantages have promoted the significance of leveraging edge computing for building next-generation surveillance applications.

Although edge computing can provide distributed and timely media data processing services, it also poses new challenges in the new computing paradigm and networking environment. The capacity-limited and unstable wireless link requires IoT devices to sufficiently compress the media data before transmitting it to edge servers. Meanwhile, data compression cannot sacrifice the accuracy of object detection on edge servers. Moreover, the applied data compression and object detection algorithms in IoT devices and edge servers should be lightweight enough but still achieve timely data processing under high media data generation rates. In addition, compared to cloud computing, distributed edge training and inference may lack global knowledge for object detection [3]. How to integrate the edge knowledge in the cloud and make the cloud interact with edge servers to update their locally trained models is another challenge in edge computing based object detection.

In this article, we propose an edge computing based object detection architecture and a preliminary implementation to achieve distributed and efficient object detection for surveillance applications. The reminder of this article is organized as follows. The next section introduces the proposed edge computing based object detection architec-

The authors are with Central South University, Changsha, China. Deyu Zhang is the corresponding author.

Digital Object Identifier:
10.1109/MNET.2018.1700415

0890-8044/18/$25.00

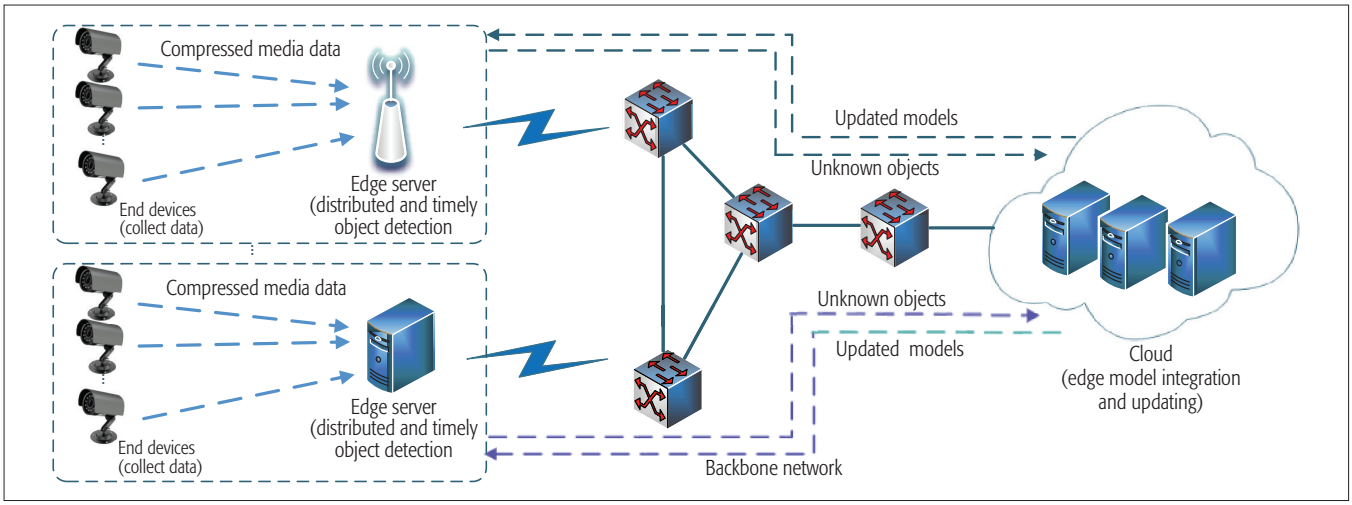

FIGURE 1. Edge computing based object detection architecture.

ture and its benefits. Following that, we clearly identify the key challenges in the proposed architecture. We then present a case study to show our preliminary implementation on edge computing based object detection and evaluate its performance. Finally, we conclude the article.

Edge Computing Based Object Detection: Architecture and Benefits

In this section, we briefly introduce the architecture and potential benefits of edge computing based object detection.

Edge Computing Based Object Detection Architecture

Edge computing refers to fully exploiting the data storage and processing capabilities of IoT terminals and their nearby infrastructures/devices to provide delay-efficient computing service at the edge of the Internet [4]. By leveraging these advantages, we aim to build an edge computing based object detection architecture to achieve distributed, prompt, and communication-efficient media data processing and object detection for real-time surveillance applications.

Figure 1 illustrates an overview of the proposed edge computing based object detection architecture. The architecture consists of three layers: the IoT device layer, the edge server layer and the cloud layer. The data flow goes as follows. Real-time media data is first captured by IoT devices and then transmitted to edge servers for object detection [5]. When the distributed edge servers find unknown objects (or detect the objects with low confidence), they will transmit the images to the cloud for further detection. Since the cloud has the global knowledge of the distributed edge servers and more powerful computing capability, it can assist the edge servers to detect unknown objects and update their inference models. In the following, we describe the detailed responsibilities of different components in the proposed architecture.

Media Data Compression in the End Device Layer: The end device layer is composed of a wide variety of monitoring devices, e.g., cameras, to capture real-time media data for object detection. Since these devices are generally connected to edge servers via speed-limited and unstable wireless communications, the captured media data should be sufficiently compressed before the transmission.

Distributed and Timely Object Detection in the Edge Layer: The edge server layer consists of numerous distributed facilities/infrastructures, for example, local servers or base stations, which are responsible to analyze the received compressed media data and provide timely object detection without transmitting them to the cloud via the backbone Internet [6]. Since edge servers are usually small-scale with limited computing capabilities, the object detection scheme should be lightweight without sacrificing detection accuracy. Moreover, edge servers work in a distributed way and each one only serves a number of nearby end devices.

Edge Model Integration and Parameters Updating in the Cloud Layer: The cloud layer is responsible for the integration of the edge model and the updating of parameters at edge servers. Since the performance of distributed model training and object detection in edge servers may be significantly degraded by their local knowledge, the cloud integrates the different well-trained edge models to achieve a global knowledge. When an edge server finds unknown objects (or detects the objects with low confidence), the cloud can leverage its powerful computing capability and global knowledge for further detection and assisting edge servers to update the inference models.

Benefits

With the proposed architecture, we can fully exploit the advantages of edge computing to support delay-sensitive object detection. Besides, it can also provide the following benefits.

Reduced Deployment Cost: Under the proposed architecture, real-time media data can be sufficiently compressed and transmitted to edge servers for object detection via wireless communication instead of deploying dedicated high-speed fiber networks. Hence, the total deployment cost of real-time monitoring applications can be significantly reduced.

Timely and Accurate Object Detection: Compared to training data and detecting objects in the cloud, the proposed architecture can leverage the computing capability of end devices and edge servers to perform real-time object detection in

the vicinity, without frequently interacting with the cloud via the backbone network. Moreover, the object detection algorithm deployed at the edge can cooperatively and adaptively work with the data compression algorithm of the end devices to provide high detection accuracy on compressed data.

Distributed Detection with Global Knowledge Sharing: Distributed data training and object detection is a double-edged sword for real-time surveillance applications. Although it can serve real-time data processing, each edge server lacks the global model for object detection. The proposed architecture can exploit the centralized cloud to integrate the local knowledge from distributed edge servers to achieve a global model that is in turn used to improve the local detection accuracy.

Key Challenges

In order to fully exploit the benefits of edge computing based object detection, we still face several challenges in its implementation that significantly impede the flourish of the related applications.

Trade-off Between Compression Ratio and Detection Accuracy

Since the data generation rate in media applications far exceeds the bandwidth of wireless channels, IoT terminals have to compress the raw media data before transmission [7]. However, the compressed data may be distorted, which can impact the accuracy of object detection at the edge servers. Thus, the first challenge comes from the trade-off between compression ratio and detection accuracy, especially when we use conventional lossy compression methods such as JPEG. Using such a method will significantly reduce image fidelity, thus deteriorating detection accuracy. As we observe, only a small region of the raw image contains the critical information to detect the object, which is called the region-of-interest (RoI). As an example, consider an application to identify a boy running on a grassland in an image. The object of interest, i.e., the boy, possibly takes less than 15 percent of the image, while the background takes the rest of it. In this case, given that the clarity of RoI is guaranteed, we can highly compress the background without impacting the accuracy of object detection.

The end devices send the compressed data to the edge servers through wireless links for object detection. Different from the dedicated fiber connections between the conventional cameras and data centers, the conditions of wireless links are highly dynamic and unstable. When the channel is in deep fading, the delay caused by data transmission becomes the key obstacle for real-time applications, especially for the case that the RoI takes a large portion of the image. For example, for an application to identify the IDs of people from an image of a workshop, the outline of all people will be RoIs, which can take 80-90 percent of the whole image. In this case, the RoI has to be compressed to a certain extent to reduce the size of the media data. To guarantee detection accuracy, the edge-computing based architecture requires adaptive data compression at the IoT terminals to carefully strike the balance between data compression rate and transmission delay. To this end, the edge servers can collaborate with the IoT terminals to fine-tune the setting of data compression. Furthermore, the object detection performed at the edge servers can exploit the correlation of the media flow over time to detect the object in a compressed image, which provides extra possibility to improve the accuracy of detection.

The objects exhibit features of a large variety, ranging from static features, such as colors and shapes, to dynamic features, such as gait and gestures. Even for the powerful cloud server, the computation complexity to train such a global model is still considerable.

Inference Model Integration and Updating

The architecture consists of a massive number of edge servers, each of which maintains a local inference model. The local model is trained to detect several specific objects that constitute a detectable object pool. To obtain the global knowledge, the cloud server integrates the local models into a much larger global model. The detectable pool of the global model is the union of the pools of all edge servers, which can includes thousands of objects. The objects exhibit features of a large variety, ranging from static features, such as colors and shapes, to dynamic features, such as gait and gestures. Even for the powerful cloud server, the computation complexity to train such a global model is still considerable. Therefore, the efficiency to integrate the global model poses a significant challenge.

Furthermore, object detection at the edge server cannot identify new objects out of its detectable pool. Although we can put the global model at the edge server to enable the detection of all objects, the small-scale edge servers may not be able to afford it due to the computation capability. Take the surveillance of a wildlife center as an example, in which the edge-computing architecture is deployed to detect the daily activities of animals. One of the edge servers controls several camera sensors to monitor the gathering places of antelopes and tigers. To improve the efficiency of inference, the detectable pool of the edge server includes only the two animals. Upon the presence of unknown animals, such as monkeys, the local inference model fails. To enable the edge server for unknown object detection, the cloud server needs to update the original local model to describe the feature of a new object. Furthermore, the size of the updating should also be constrained, for the sake of efficiency. As such, we need to gradually update the local inference model, which poses another challenge even with the assistance of the global knowledge at the cloud server.

A Preliminary Implementation of Edge Computing based Object Detection

In this section, we present a preliminary implementation of the proposed object detection architecture, aiming to address the aforementioned challenges.

Implementation Overview

According to the proposed object detection architecture and its underlying challenges, the implementation should address four key problems:

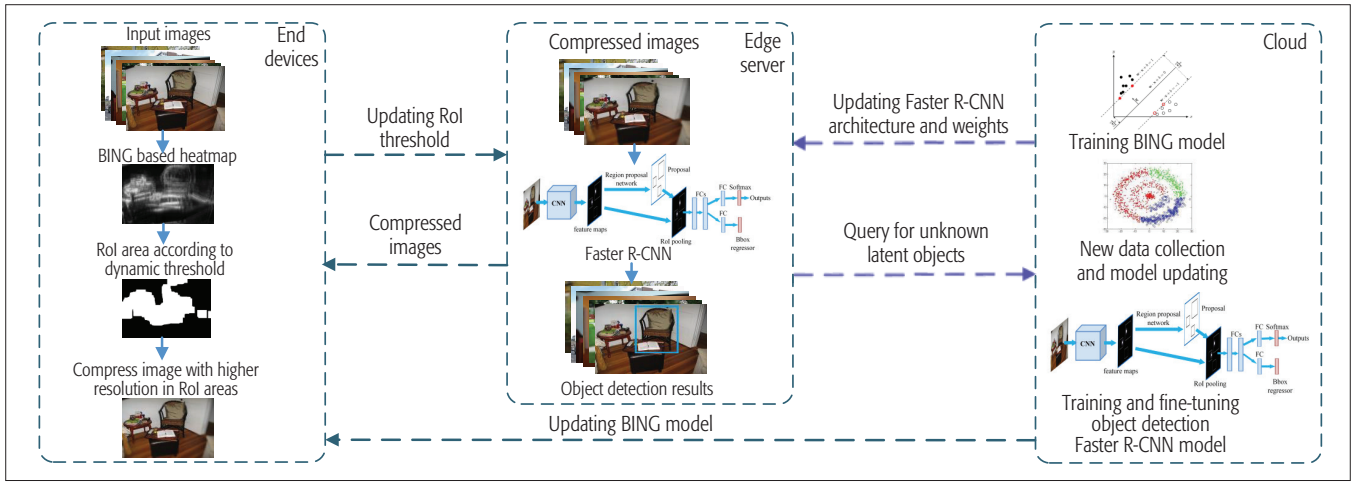

FIGURE 2. Implementation overview.

- How to compress data at end devices.
- How to design lightweight and distributed object detection algorithms for edge servers.
- How to integrate distributed models of edge servers and assist unknown object detection in the cloud.
- How to efficiently perform interactions among end devices, edge servers and the cloud to improve the performance of the whole system.

Figure 2 shows the overview of our preliminary implementation on the proposed architecture. In the end devices, we extract the Regions-of-Interest (RoI) from each image by employing the binarized normed gradients (BING [8])-based RoI extraction method, named BRE, such that the areas of the interested targets in each image can be well identified and compressed with a low compression ratio to keep more details, while the other areas can be regarded as background and sufficiently compressed without caring about information loss. Edge servers adopt Faster R-CNN for model training and object detection [9], and can also adaptively control end devices to adjust the size of RoI areas according to their detection accuracy. The cloud maintains the labeled training data and inference models of different edge servers. It can assist edge servers to detect unknown objects and update inference models and adjust the BING models of end devices based on the powerful computing capability and integrated knowledge.

Implementation Details

In this subsection, we describe the details of our implementation, in terms of the different algorithms applied in end devices, edge servers and the cloud, respectively.

In order to transmit rich media data over the air, the captured real-time images should be sufficiently compressed with a high speed. Most traditional compression algorithms, such as JPEG [10], can achieve acceptable compression rate and efficiency. However, since these algorithms perform indiscriminate compression on the whole image, the accuracy of object detection may be sacrificed due to losing valuable information of the targeting areas (i.e., the areas with objects) [11]. Therefore, we design an RoI based image compression method, named BRE, where the potential targeting areas are clearly identified and lightly compressed to keep their original information for object detection while the background areas are sufficiently compressed to achieve efficient data transmission over wireless links. To optimally balance the detection accuracy and data compression rate, the key problem is to design an accurate RoI detection algorithm.

Limited by the computing capability of end devices, the RoI detection algorithm should be lightweight enough. The binarized normed gradients (BING) model proposed by M. M. Cheng *et al.* [8] is a data-driven object proposal generation algorithm, that can find latent object bounding-boxes at 300 fps on a lightweight terminal. We propose a BING RoI Extraction algorithm, which improves BING through the following mechanisms. In BING, the proposal generation method provides a series of latent object boxes sorted by their object probability scores, where a higher score indicates that the area of the chosen box has a higher probability of containing an object. However, these chosen boxes cover most of the image area for high recall, which is not applicable for our RoI based image compression task. Since BING calculates the probability score in different sizes of reshaped feature maps, and the score of each pixel indicates an 8×8 region on left top of it, we can reshape all these score matrices back to the original image size and integrate them together, such that we can create a heatmap to identify the RoI based on the integrated score matrix. Moreover, we define an adaptive threshold to bound the RoI area according to the created heatmap. Only if the score of the pixel in the heatmap exceeds the threshold, will the pixel be included in the RoI area. After that, we can use JPEG 2000 to compress the chosen RoI areas and the background areas with different compression ratios, respectively.

By adopting Faster R-CNN as the local model, the edge server detects objects in the compressed image in three steps. First, it uses a convolutional neural network that takes the compressed image as input and extracts a series of feature maps. Then Region Proposal Network (RPN) is employed to obtain a set of rectangular object

proposals. Second, the RoI pooling layer converts the features inside the proposals into small feature maps with fixed size. Finally, the softmax layer and the bounding box regression layer serve as the output layers of the Faster R-CNN. The former outputs a discrete probability distribution over potential detectable objects. The latter outputs the bounding box of the detected objects. The object detection method used by the edge servers can be changed to any latest solution, given that the method can maintain the interactions among layers to achieve distributed and communication-efficient object detection.

The cloud server has powerful computing capacity, so it trains the BING model and updates the local model to detect new objects. BING uses binary approximation [12] of a linear support vector machine (SVM) [13] to efficiently evaluate whether there is an object in an image window. To update the local model at edge servers, the cloud server modifies the output layer of the edge model to classify the new object. To train this modified model, the cloud server uses part of the data corresponding to the previously detectable objects and all data corresponding to the new object as training data. In such a way, the modified model can detect the new object without sacrificing the accuracy to detect the previously detectable objects.

Interactions among End Devices, Edge and Cloud

As a whole system, the three layers of the proposed architecture, i.e., end devices, edge servers and the cloud, interact with each other to adaptively adjust their algorithms/models to improve the performance of object detection.

Interactions between End Devices and Edge Servers for Adjusting RoI Threshold: In BRE, we use an adaptive threshold to extract the RoI areas from the created heatmap based on BING. The determination of the threshold is significantly impacted by the specific type of objects and the learned knowledge from the training data. For example, for an end device monitoring human traffic in a crowded mall, the outline of all people in its monitored area will be RoIs, which can take 80–90 percent of the whole image. Comparatively, for an end device monitoring a tracked animal in the wild, the RoI would only take a small portion of the captured image. If the inference model on edge servers is well trained through a huge amount of training data, we can confidently detect an object even without its full details in the compressed image. Otherwise, we should keep a relative large RoI area to improve the detection accuracy. To this end, we adaptively adjust the value of the threshold in BRE, according to the detection accuracy of the Faster R-CNN model on edge servers during the training phase.

	Device	Hardware	Software
End device	Xiaomi 6	Snapdrago 835 (2.45 GHz), 6 GB, Andreno 540	BRE algorithm
Edge server	PC	i7 6700 (4 GHz), 32 GB, GTX 980Ti	Fast R-CNN on tensorflow
Cloud server	Work station	E5-2683 V3 (2.00 GHz), 128 GB, GTX TitanXp*4	Fast R-CNN on tensorflow and BRE training

TABLE 1. Settings of the experimental platform.

Interactions between the Cloud and End Devices for BING Parameter Modification: Since end devices are usually too resource-limited to run complex data training algorithms, the training of the BING model is performed offline in the cloud and applied to end devices. Meanwhile, during online object detection, the cloud will occasionally get new images from the end devices and retrain the BING model according to the increasing dataset. The parameters of the updated BING model are then transmitted to end devices for RoI area extraction.

Interactions between Edge Servers and Cloud for Unknown Object Detection: In our implementation, each edge server trains its Faster R-CNN model in a distributed way, but the cloud has a global training dataset and can integrate the distributed edge models. Intuitively, the cloud can share a global model with edge servers to enable them to recognize as many objects as possible. However, since edge servers are usually much more lightweight than cloud servers, it is impractical to run a global model on the edge servers.

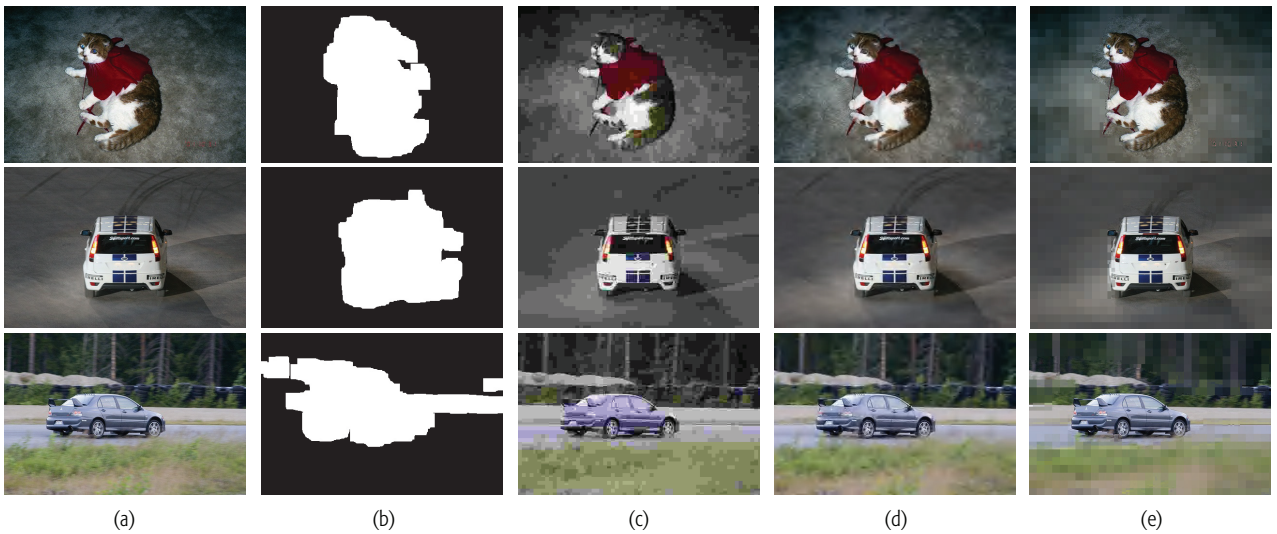

FIGURE 3. Comparison of different image compression methods: a) original images; b) RoI area; c) JPEG; d) JPEG2000; d) RoI based.

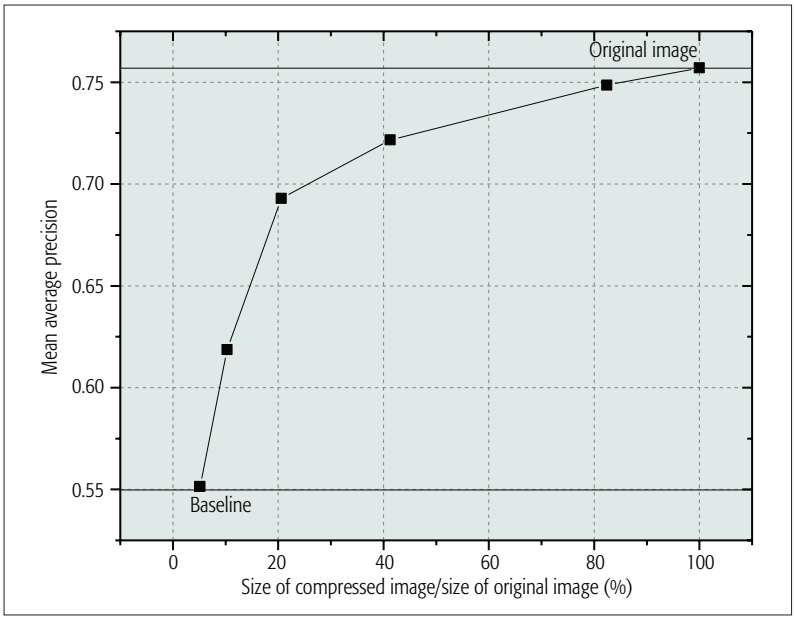

FIGURE 4. Object detection accuracy vs. image compression ratio.

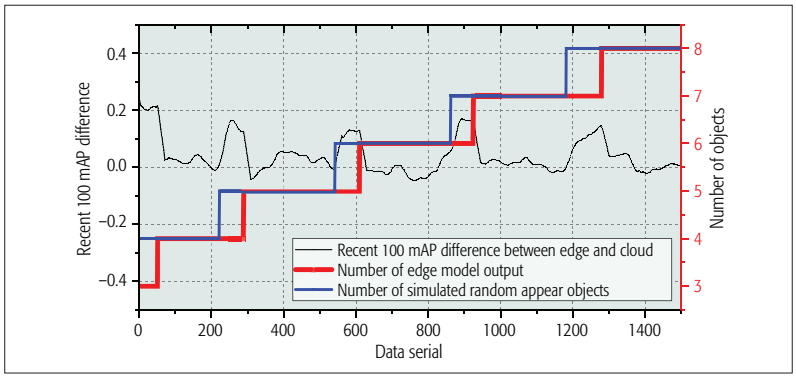

FIGURE 5. Detection accuracy of edge server for new objects.

Moreover, since each edge server only processes the media data from a number of nearby end devices, the number of frequently appeared objects are usually very limited, such that a local inference model would be sufficient.

When a new object that is not pre-trained on an edge server appears, the edge server will send the image to the cloud for further detection. The regions containing potential objects will be checked by analyzing the probability scores using the BING-based heatmap. If the probability score exceeds a threshold, the contained object will be identified as a latent one. Since the cloud has the global information of all edge servers, it can quickly identify the unknown object. If the edge server frequently sends the same type of unknown objects to the cloud, the cloud can retrain the Fast R-CNN model of the edge server by adding an output to detect this new object. Afterward, the new neural network architecture and weights will be transmitted to the edge server and replace old ones. In such a way, the edge server can recognize the unknown object and detect it in future images.

Performance Evaluation

In this section, we evaluate the performance of our preliminary implementation. The settings of the experimental platform are shown in Table 1. We use the labeled training dataset of Pascal VOC2007 [14] to train the distributed Faster R-CNN models on edge servers, and the testing dataset of Pascal VOC2007 to evaluate the performance.

Figure 3 compares the image quality under different image compression algorithms. It can be seen from the figure that the proposed RoI based image compression algorithm can accurately identify the areas with latent objects. Moreover, under the same compression ratio, our proposed solution can keep more details of the latent objects for object detection than JPEG and JPEG 2000 without RoI coding, which conducts the quantization and coding on the regions of interest with higher resolution while on other parts with lower resolution.

Figure 4 shows the detection accuracy of our implementation under different image compression ratios, where we use the mean Average Precision (mAP) [15] as the metric to evaluate the accuracy. The top line represents the detection accuracy of edge servers using original images, while the baseline represents the detection accuracy using the compressed images with all the background removed. As shown in the figure, the detection accuracy is only reduced 2.5 percent under the image compression ratio of 60 percent, indicating that our implementation can significantly improve image transmission efficiency with little impact on detection accuracy. Moreover, according to the comparison between our algorithm and the baseline, we find that the background information can help edge servers improve the detection accuracy of objects.

Figure 5 illustrates the performance of the detection of unknown objects on the edge server by adaptively updating its model with the assistance of the cloud. The blue line denotes the time slots when a specific type of unknown object begins to frequently appear in the monitoring areas of the edge server. The figure shows that the inference model output of the edge server can be quickly updated by the cloud's detection of unknown objects after it appears for a certain rounds. The mAP difference between the cloud and the edge server shows that the model integration and updating can improve the detection accuracy of the edge servers with local knowledge.

Conclusion

Moving data processing from the cloud to the edge has been a significant trend in the era of Internet-of-Things. In this article, we have introduced an edge computing based object detection architecture for surveillance applications. The benefits and underlying challenges of the proposed architecture are clearly identified. We have also presented a preliminary implementation of the proposed architecture and evaluated its performance. We hope that this article can provide insights into how to leverage edge computing to build more efficient surveillance applications and attract continuous research efforts.

Acknowledgments

This research work is supported by National Natural Science Foundation of China (61702561, 61702562); the Innovation-Driven Project of Cen-

tral South University (No. 2016CXS013); the International Science and Technology Cooperation Program of China (No. 2013DFB10070); and the China Hunan Provincial Science and Technology Program (No. 2012GK4106).

References

[1] Y. Zhang *et al.*, "A Survey on Emerging Computing Paradigms for Big Data," *Chinese J. of Electronics*, vol. 26, no. 1, 2017, pp. 1–12.
[2] Q. Xu *et al.*, "Secure Content Delivery with Edge Nodes to Save Caching Resources for Mobile Users in Green Cities," *IEEE Trans. Industry Informatics*, 2017, to appear, DOI: 10.1109/TII.2017.2787201.
[3] Z. Fadlullah *et al.*, "State-of-the-Art Deep Learning: Evolving Machine Intelligence toward Tomorrows Intelligent Network Traffic Control Systems," *IEEE Commun. Surveys & Tutorials*, 2017, DOI: 10.1109/COMST.2017.2707140.
[4] J. Ren *et al.*, "Serving at the Edge: A Scalable IoT Architecture Based on Transparent Computing," *IEEE Network*, vol. 31, no. 5, 2017, pp. 96–105.
[5] D. Zhang *et al.*, "Utility-Optimal Resource Management and Allocation Algorithm for Energy Harvesting Cognitive Radio Sensor Networks," *IEEE JSAC*, vol. 34, no. 12, 2016, pp. 3552–65.
[6] T. G. Rodrigues *et al.*, "Hybrid Method for Minimizing Service Delay in Edge Cloud Computing through VM Migration and Transmission Power Control," *IEEE Trans. Computers*, vol. 66, no. 5, 2017, pp. 810–19.
[7] J. Ren *et al.*, "Dynamic Channel Access to Improve Energy Efficiency in Cognitive Radio Sensor Networks," *IEEE Trans. Wireless Commun.*, vol. 15, no. 5, 2016, pp. 3143–56.
[8] M.-M. Cheng *et al.*, "BING: Binarized Normed Gradients for Objectness Estimation at 300fps," *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2014.
[9] S. Ren *et al.*, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," *Advances in Neural Information Processing Systems 28*, Curran Associates, Inc., 2015, pp. 91–99.
[10] G. K. Wallace, "The JPEG Still Picture Compression Standard," *IEEE Trans. Consumer Electronics*, vol. 38, no. 1, 1992, pp. xviii–xxxiv.
[11] C. Guo and L. Zhang, "A Novel Multiresolution Spatiotemporal Saliency Detection Model and Its Applications in Image and Video Compression," *IEEE Trans. Image Processing*, vol. 19, no. 1, 2010, pp. 185–98.
[12] S. Hare, A. Saffari, and P. H. Torr, "Efficient Online Structured Output Learning for Keypoint-Based Object Tracking," *Proc. 2012 IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2012, pp. 1894–1901.
[13] R.-E. Fan *et al.*, "Liblinear: A Library for Large Linear Classification," *J. Machine Learning Research*, vol. 9, no. Aug. 2008, pp. 1871–74.
[14] M. Everingham *et al.*, "The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results," http://www.pascal-network. org/challenges/VOC/voc2007/workshop/index. html.
[15] M. Zhu, "Recall, Precision and Average Precision," Department of Statistics and Actuarial Science, University of Waterloo, Waterloo, vol. 2, p. 30, 2004.

Biographies

Intuitively, the cloud can share a global model with edge servers to enable them to recognize as many objects as possible. However, since edge servers are usually much more lightweight than cloud servers, it is impractical to run a global model on the edge servers.

Ju Ren [S'13, M'16] (renju@csu.edu.cn) received the B.Sc. (2009), M.Sc. (2012), Ph.D. (2016) degrees, all in computer science, from Central South University, China. From August 2013 to September 2015, he was a visiting Ph.D. student in the Department of Electrical and Computer Engineering, University of Waterloo, Canada. Since August 2016, he has been a tenure-tracking professor with the School of Information Science and Engineering, Central South University, China. His research interests include Internet-of-Things, transparent computing and edge computing. In these related research areas, he has published over 30 papers in prestigious international journals and conferences, including IEEE TIFS, TWC, TVT, TII and IEEE INFOCOM, among others. He serves/has served as an associate editor for *Peer-to-Peer Networking and Applications*, a leading guest editor for *IEEE Network*, and a Technical Program Committee member of many international conferences including IEEE INFOCOM'18, Globecom'17, WCNC'17, and WCSP'16, among others. He also served as a co-chair for the "Vehicular Communications, Networks, and Telematics" track of IEEE VTC'17 Fall, a co-chair of the "Transparent Media Computing" special session of IEEE VCIP'17, and an active reviewer for over 20 international journals. He is a member of IEEE and ACM.

Yundi Guo (csgrandeur@csu.edu.cn) received the B.S. degree in computer science from Central South University, China, in 2012. Currently, he is pursuing his Ph.D. degree in computer science at Central South University, China. His research interests include deep learning and image processing.

Deyu Zhang [S'14, M'17] (zdy876@csu.edu.cn) (corresponding author) received the B.Sc. degree (2005) in communication engineering from PLA Information Engineering University, China, and the M.Sc. degree (2012) from Central South University, China, also in communication engineering. He received his Ph.D. degree in computer science from Central South University, China, in 2016. He is now an assistant professor with the School of Software and a postdoc fellow with the Transparent Computing Lab in the School of Information Science and Engineering, Central South University, China. He was a visiting scholar with the Department of Electrical and Computer Engineering, University of Waterloo, Canada, from 2014 to 2016. His research interests include stochastic resource allocation transparent computing, edge computing and IoT. He is a member of IEEE and CCF.

Qingqing Liu (liuqingqing@csu.edu.cn) received the B.Sc. degree in automation from Central South University in 2016. Currently, she is pursuing her M.Sc. degree in control science and engineering from Central South University, China. Her research interests include machine learning, edge computing and IoT.

Yaoxue Zhang (zyx@csu.edu.cn) received his B.Sc. degree from Northwest Institute of Telecommunication Engineering, China, in 1982, and his Ph.D. degree in computer networking from Tohoku University, Japan, in 1989. Currently, he is a professor with the School of Information Science and Engineering, Central South University, China, and also a professor with the Department of Computer Science and Technology, Tsinghua University, China. His research interests include computer networking, operating systems, ubiquitous/pervasive computing, transparent computing, and big data. He has published over 300 technical papers in international journals and conferences, as well as nine monographs and textbooks. He is a fellow of the Chinese Academy of Engineering and the editor-in-chief of the *Chinese Journal of Electronics*.