Smart Fire Alarm System with Person Detection and Thermal Camera

Yibing Ma[1](✉), Xuetao Feng[1], Jile Jiao[1], Zhongdong Peng[1], Shenger Qian[1], Hui Xue[1], and Hua Li[2]

[1] Alibaba Group, Beijing, China
{yibing.myb, xuetao.fxt, jile.jjl, zhongdong.pzd, shenger.qse, hui.xueh}@alibaba-inc.com

[2] Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China
lihua@ict.ac.cn

Abstract. Fire alarm is crucial for safety of life and property in many scenes. A good fire alarm system should be small-sized, low-cost and effective to prevent fire accidents from happening. In this paper we introduce a smart fire alarm system used in kitchen as a representative scenario. The system captures both thermal and optical videos for temperature monitoring and person detection, which are further used to predict potential fire accident and avoid false alarm. Thermal videos are used to record the temperature change in region-of-interests, for example, cookware. YOLOv3-tiny algorithm is modified for person detection and can be iteratively improved with the hard examples gathered by the system. To implement the system on an edge device instead of a server, we propose a high-efficiency neural network inference computing framework called TuringNN. Comprehensive rules enable the system to appropriately respond to different situations. The proposed system has been proved effective in both experiments and numerous cases in complex practical applications.

Keywords: Fire alarm system · Thermal camera · Person detection · TuringNN

1 Introduction

Fire prevention is crucial for the safety of life and property in residential and public scenes such as restaurants, hotels and factories. According to statistics [1, 2], cooking is the leading cause of hotel fires (46%) and one out of every 12 hotels reports a structural fire per year in US. A fire accident usually happens when the cookware is kept heated on the stove without supervision. Although fire regulations have been made to prevent fire accidents, they are sometimes ignored or disobeyed. The earlier the accidents can be detected, the easier they can be dealt with and the fewer damages can be caused. Therefore, it is necessary to build a smart alarm system for fire prevention. The system needs two essential modules, potential fire sensing and proper disposal strategy.

V. V. Krzhizhanovskaya et al. (Eds.): ICCS 2020, LNCS 12143, pp. 353–366, 2020.
https://doi.org/10.1007/978-3-030-50436-6_26

There are various methods for fire sensing such as by detecting smoke, carbon monoxide or flame. However, these methods work only when the fire has broken out. In contrast, monitoring the temperature of combustibles makes it possible to prevent fire before it occurs. There are three necessary ingredients for most fires, i.e. fuel, oxygen and heat. It is feasible to predict potential fire accidents when the temperature rises too quickly or near the ignition point. Temperature measurement sites in application scenario, e.g. stoves in kitchen, are usually numerous and irregularly distributed. Consequently, it is more practical to deploy remote and wide-ranged thermal cameras than single-point thermometers to monitor temperature.

Nevertheless, merely monitoring temperature may lead to the problem of false alarm when someone is working in the scene, for example a chef in the kitchen. As a correction, it is necessary to judge the presence of persons. The task can be fulfilled by person detection. With the development of computer vision and deep learning, person detection has been widely researched and various algorithms have been proposed. Surveillance cameras have become common in public places, providing clear and real-time videos. But there are still challenges for person detection in different sites. For instance, chefs in kitchen usually wear white suits with masks and tall hats, which are rare in public datasets and difficult to detect for existing models.

Moreover, the system should be reliable, low-cost and easy to deploy. Transmitting all the videos to one computing server is a waste of network bandwidth. Instead, person detection should be implemented on distributed mobile processors on an edge device and only detection results are sent to communication server. To ensure the speed of person detection, a device-specific deep learning acceleration framework is necessary.

In this paper, we propose a smart fire alarm system with bi-spectrum camera and embedded Industry Personal Computer (IPC). The system is applied in kitchens and the diagram is shown in Fig. 1. The camera has an optical lens and a thermal lens to capture visible videos and monitor the temperature of cookware respectively. The processor performs person detection with YOLOv3-tiny [3] model. The algorithm is modified to deal with special appearance and occlusions of chefs in kitchen, and the inconsistency of aspect ratio between different training datasets. To implement the system on a mobile processor, we propose a high-performance neural network inference computing framework called TuringNN. Person detection and temperature analysis are completed in real time on the IPC. When the temperature reaches preset thresholds and no person is detected, staff in charge are alarmed with buzzers in the kitchen, phone calls or instant messages from mobile applications. Meanwhile, the IPC reports an alarm along with current temperature, video frames and detection results to the server. In severe cases when temperature gets too high or rises too fast, the IPC sends a signal to controller to shut off the stove. As the system runs, incorrect person detection cases are collected to improve the model, and disposal rules can be updated according to feedback. The system has been applied for kitchen fire alarm in Freshhema [4], a combination of fresh food supermarket and restaurant with over 130 chain stores all over China. By adjusting detection model and disposal rules, we can apply the system to various application situations.

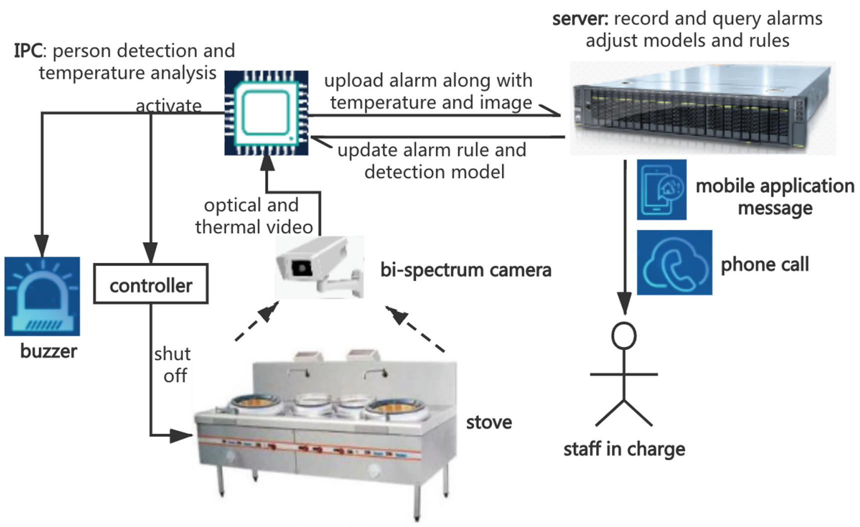

Fig. 1. The diagram of our system.

The remainder of the paper is organized as follows. Section 2 reviews the related work on fire alarm systems and person detection. Section 3 presents the details of the proposed system. Section 4 introduces the experiments and shows the respective results. Section 5 concludes the whole paper and gives some directions for future work.

2 Related Works

2.1 Fire Alarm System and Thermal Camera

Fire safety solutions have been widely researched for years. A fire alarm system has a number of devices working together to detect fire and warn people through visual and audio appliances when smoke, flame, carbon monoxide or other emergencies occur. These alarms may be activated automatically from smoke detectors and heat detectors or be activated via manual fire alarm activation devices like manual call points. Siemens [5] proposed a solution of fire detection in kitchens with fire detector which consists of smoke, carbon monoxide and flame detectors. However, this solution only works after fire has happened. Moreover, although Siemens claims that selectable parameter settings make the detectors immune to deceptive phenomena, they are not well adaptive to environment changes. Cheng et al. [6] proposed a fire safety device for stove-top burner. It senses the temperature of the cooking utensil and automatically shuts off the flow of electricity or gas to the burner when the temperature of the cooking utensil exceeds threshold. However, the device does not consider the presence of persons, which may cause false alarms. There are some other fire alarm solutions such as [19–21], which are similar to [5] and [6]. The alarms are made after the fire happens, which lead to delayed handling.

Thermal camera (also called infrared camera) is a device that creates an image using infrared radiation. All objects with temperature higher than absolute zero (−273.15 °C) radiate infrared constantly. Infrared thermal imager is a device that receives infrared radiation from objects and converts it into visible-light image. This thermal image corresponds to the temperature distribution field on the surface of the object, which is essentially the infrared radiation distribution map of all parts of the object. Different from the 400–700 nm range of the visible light camera, thermal cameras are sensitive to wavelengths from about 1 μm to about 14 μm. When sensitive enough, infrared camera can capture accurate temperature of every position in an image, and further calculate the temperature change rate in a video.

2.2 Person Detection and Deep Learning Inference Framework

In kitchens, cooking is similar to fire accident when only monitoring heat, flame or smoke. Therefore, person detection is an essential task in an intelligent fire alarm system to prevent false alarms. Compared with general object detection, there are more challenges in person detection, such as various clothing, appearance, poses and different viewpoints of the camera. With the development of computer vision and deep learning, various object detection algorithms are proposed such as Faster R-CNN [7], SSD [8] and YOLO [9]. Although new algorithms are constantly proposed and better performances have been achieved, they usually require lots of computing resources and long inference time. In application scenarios, deep learning models should be economical and efficient on mobile platforms with limited computing power.

Numerous techniques have been proposed to accelerate deep learning models on low-power devices. For example, model quantization refers to the process of reducing the number of bits that represent a number [22]. In the context of deep learning, the predominant numerical format used for research and for deployment has so far been 32-bit floating point. However, the pursuit for reduced bandwidth and computing requirements of deep learning models has driven research into using lower-precision numerical formats. It has been proved that weights and activations can be represented using 8-bit integers without incurring significant loss in accuracy. By combining various acceleration techniques, researchers have built deep learning inference frameworks for mobile devices, such as TensorFlow Lite [11] and NCNN [12]. The frameworks make deep learning models faster and smaller while maintain or even improve the performance by combining multiple optimization methods. Despite the convenience of public frameworks, there are still adaption problems for specific devices and network structures, and there is still space for further optimization.

3 The Proposed System

The workflow of our system is demonstrated in Fig. 2. It consists of four modules, (a) temperature measurement, (b) person detection, (c) TuringNN and (d) disposal rules.

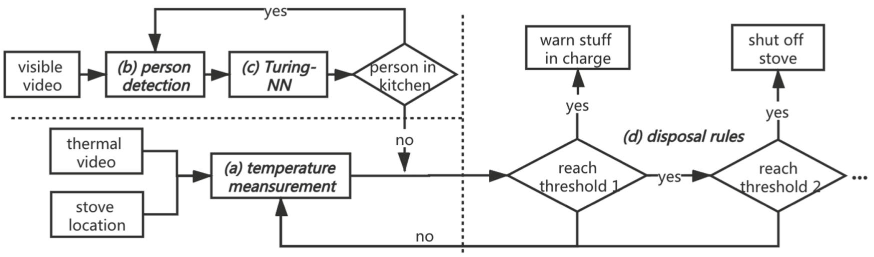

Fig. 2. The workflow of our system.

3.1 Temperature Measurement

The temperature is measured with thermal videos. First of all, the locations of stoves are manually annotated in thermal videos, as shown in Fig. 3 right. The temperature of target locations is sampled with an adjustable time interval. The heating curve is fit with least square method according to the discrete temperature data points in observation period. The slope of the curve, namely current temperature ascent rate R, is calculated as

$$R = \frac{n\sum_{i=1}^{n} ity_i - \sum_{i=1}^{n} it \sum_{i=1}^{n} y_i}{n\sum_{i=1}^{n} (it)^2 - \left(\sum_{i=1}^{n} it\right)^2} \tag{1}$$

where the last n temperature data points are used for calculation and n is set to 20 in our system, t denotes the temperature check time interval and is set according to current temperature as shown in Table 1, y_i represents the values of the n temperature data points. If the difference between the current temperature and the previous temperature is more than 20 °C, the current temperature is ignored as a noise.

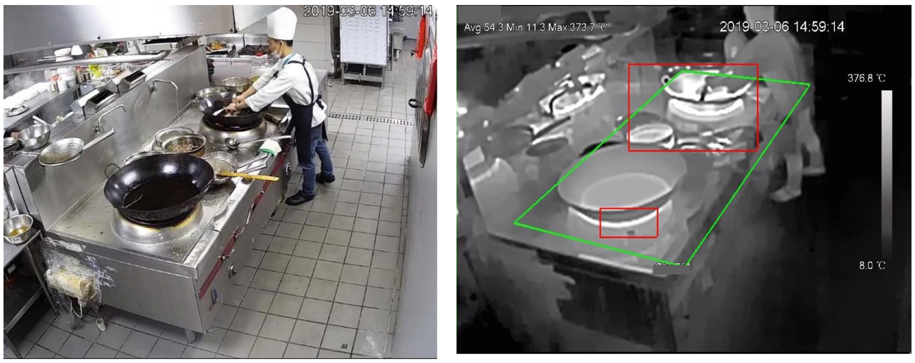

Fig. 3. Optical video (left) and thermal video (right) of the same site. The temperature measurement locations are annotated as red boxes. (Color figure online)

3.2 Person Detection

Person detection is deployed on optical video because the color and texture of persons are more obvious in optical video compared with thermal video, as shown in Fig. 3. In order to achieve accurate and efficient person detection, the small model of YOLOv3-tiny is utilized in our system. YOLOv3-tiny is pruned from YOLOv3 [3] with fewer layers and scales. Compared to the large model of YOLOv3, YOLOv3-tiny is 5 times faster with small accuracy loss. The network structure of YOLOv3-tiny is shown in Fig. 4. The model extracts feature map (FM) of input image with a 13-layer backbone network. YOLOv3-tiny predicts bounding boxes at two different scales (with stride 8 and 16) with the idea of Feature Pyramid Network for Object Detection (FPN) [10]. Two detection headers, which are separately built on the top of two feature maps with different scales, are responsible for detecting objects with different sizes. As shown in Fig. 4(b), each grid in the detection header is assigned with K different anchors, and thus predicts K detections that consist of 4 bounding box offsets, 1 objectiveness and C class predictions. In our case, K is set to 3 and C is 1 to only detect person. The final result tensor of detection header has a shape of $N \times N \times (K(4 + 1 + C))$ where $N \times N$ denotes the spatial size of the last convolutional feature map. Three modifications are proposed to make it more suitable for our system.

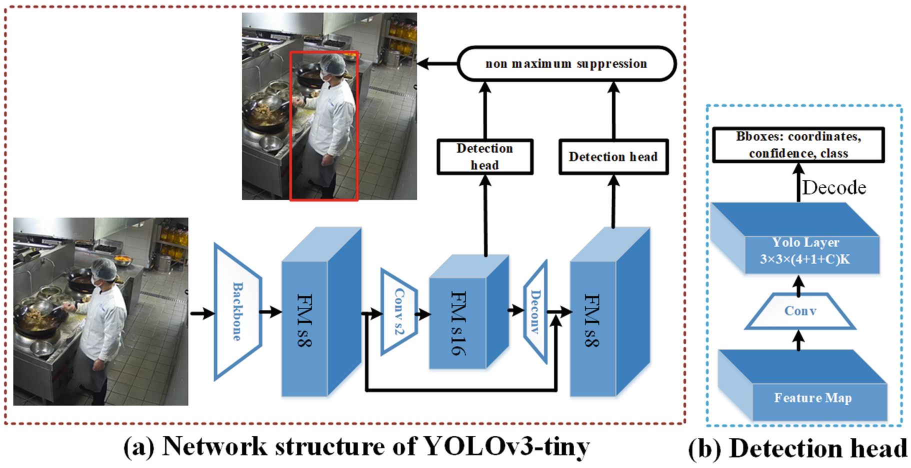

Fig. 4. The network structure of YOLOv3-tiny. K denotes the number of anchors in each scale and is set to 3 in YOLOv3-tiny.

Wide augmentation, which means data augmentation while keeping aspect ratio. To deal with the difference of person appearance between public datasets and kitchen and to guarantee the generalization ability of the model, we train it with both public datasets and application dataset of surveillance videos. As shown in Fig. 5(a) and Fig. 5(b), public datasets have various aspect ratios while surveillance videos have fixed aspect ratio of 16:9. When training the model, if we directly resize all images into the same aspect ratio, some ground truths will suffer from severe distortion, as illustrated in Fig. 5(c). Therefore, we keep the aspect ratio of images unchanged by adding borders to the image in data augmentation, as demonstrated Fig. 5(d).

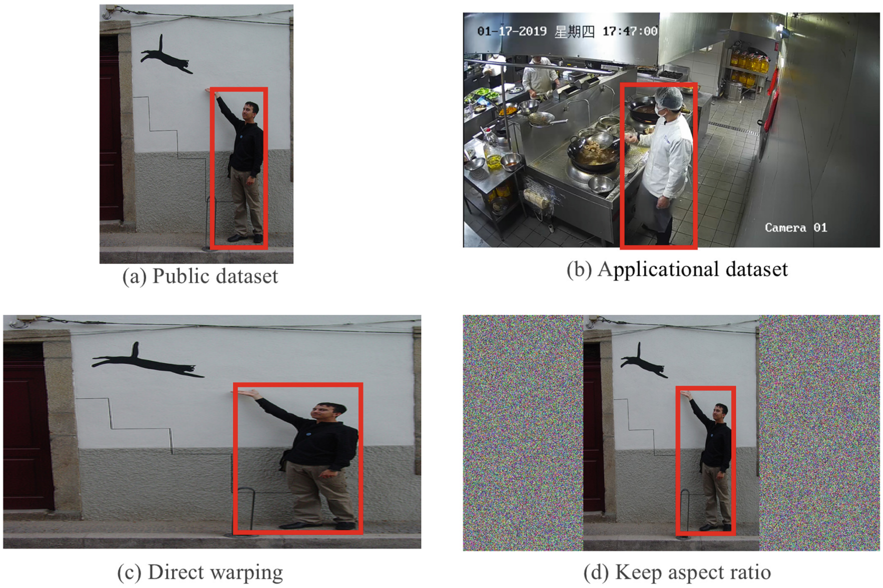

Fig. 5. Different ways to change image size.

Ignore Label. The original YOLO distinguishes positive and negative objects with an Intersection-Over-Union (IOU) threshold. However, the ground truth is ambiguous in some cases such as crowd, heavy occlusion and blur. To deal with this issue, we introduce “ignore” label in annotation. The object is annotated with ignore label if it is a crowd of persons, a part of a person or when we are not sure if the object is a person or not. In training, if the IOU between a box proposal and any ground truth box with ignore label is larger than threshold, the proposal is ignored in loss calculation. In this way the noise of unclear targets is avoided.

Iterative Improvement. With the operation of the system, the incorrect person detection instances are accumulated along with the false fire alarms. The instances can be used as hard examples (images that cause the model make errors) to improve the detection model. The new model is finetuned from current model with new instances as well as original training data, which effectively avoid overfitting. Each time the model is updated, the accumulation of hard examples and the next iteration becomes slower. Finally, the model turns stable and accurate.

3.3 Neural Network Inference Acceleration with TuringNN

TuringNN is a novel deep learning inference framework optimized for mobile heterogeneous computing on Android, iOS, Linux and Windows devices. It has optimization solutions for most neural network models, operating systems and mobile processors. When developing TuringNN, we set goals in various aspects, such as customization, resource-saving, high performance and easy-using. To achieve all these goals, comprehensive technical solutions and tricks are considered.

Data Rearrangement. Rearrange data for the underlying hardware to reduce the bottleneck of memory reading. For example, the technique of Im2col is demonstrated in Fig. 6. A straightforward implementation of convolution is not well suited for fast execution on CPU. When dealing with convolutional layers in a neural network, TuringNN rearranges the 2×2 kernel matrix into a 4×1 array. Meanwhile the $N \times M$ input feature map (image) is reshaped into a $P \times 4$ matrix by putting the elements of 2×2 neighborhood into a row, where $P = (N - 2 + 1)(M - 2 + 1)$. In this way, the operation of convolution is translated to matrix multiplication. The CPU can then take advantage of special parallel hardware (SSE or MMX) to speed up convolution substantially.

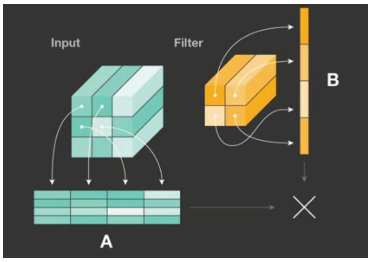

Fig. 6. Im2col-based implementation of convolution.

Quantitative Calculation. 8-bit data is calculated for the underlying hardware to increase data multiplication and throughput. The quantitative calculation chart is shown in Fig. 7. Firstly, TuringNN runs a model of 32-bit floating point (FP32) with a calibration dataset and records the maximum and minimum of the feature map in each layer. During inference, the weights and input in each layer are converted into 8-bit integers (INT8) with the recorded maximum and minimum, and then the output is turned back to FP32. The quantization and dequantization formula are noted in Eq. (2), where F and Q are the FP32 and INT8 values, while F_{max} and F_{min} are the maximum and minimum FP32 values of each layers. As shown in Fig. 7 left, the dequantization of the previous operation and the quantization of the next operation will cancel each other out. Therefore, TuringNN combines the dequantization and quantization before each operation when there are multiple quantified operations in succession, as illustrated in Fig. 7 right.

$$Q = \frac{255(F - F_{min})}{F_{max} - F_{min}} - 128, \quad F = \frac{(Q + 128)(F_{max} - F_{min})}{255} + F_{min} \tag{2}$$

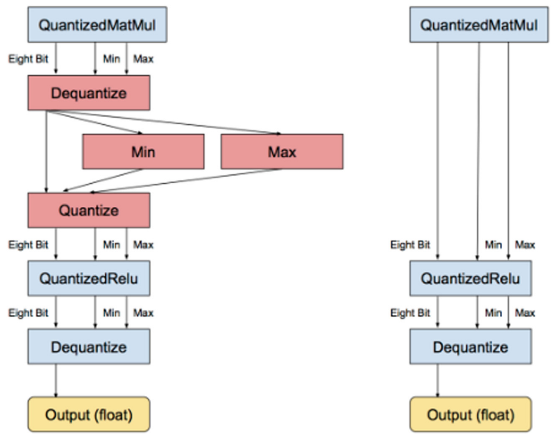

Fig. 7. The quantitative calculation charts. Dequantization are canceled out by quantization of the next layer.

Multithreading. TuringNN makes full use of the ability of multithreading. When the compiler supports OpenMP, we set the concurrent number according to the hardware device, and the concurrent time-consuming cycle will use multi-core to complete the operation. On Apple's platform, we use Grand Central Dispatch (GCD) to dispatch the time-consuming cycle, and the system dynamically allocates the concurrent number, so as to obtain better performance than OpenMP. On Android and Linux platform, using thread pool for multithreading management and control can make more efficient use of multi-core and large-small-core hardware resources.

Memory Layout. Different data arrangement formats are designed for different operators. For example, operators such as convolution and pool run on the memory layout of batch number/channel/height/width (NCHW), while operators such as post and split use the memory layout of NHWC. When the memory layouts of adjacent operators are inconsistent, conversion operators are inserted.

Fusing. When transforming models, we merge layers that do not need to exist independently to reduce the amount of calculation. For example, by merging batch normalization into convolution, the cost of batch normalization is saved while the operation result is not affected.

3.4 Disposal Rule and Devices

The disposal rules of industrial personal computer (IPC) in our system are demonstrated in Fig. 2. The optical lens and thermal lens focus on the same view of the kitchen. Person detection is continuously run in optical videos. If no person is detected, current temperature of cookware is measured from thermal videos. When food is fried, the oil temperature rises fast and then slows down. According to this rule, different temperature thresholds are set for temperature analysis, as listed in Table 1. As the temperature rises, the check interval becomes shorter, the accent rate threshold gets smaller and the disposal action becomes more urgent. When the temperature or the ascent rate reaches a threshold, the corresponding disposal actions are triggered. Each time the system sends an alarm, the frames of the optical and infrared videos are saved along with person detection results and temperature data. Incorrect person detection results will be fed back to improve person detection algorithm.

Table 1. Different thresholds in different temperature ranges.

Temperature range (°C)	Temperature threshold (°C)	Check interval (s)	Ascent rate threshold (°C/s)	Disposal actions
180–220	220	1.2	3	Buzzer
220–250	250	1	2.4	The above and IM app
250–300	300	0.8	1.8	The above and phone call
Above 300	350	0.6	1.5	The above and shut off stove

The bi-spectrum camera in our system (Fig. 8) is DH-TPC-BF2221-T manufactured by Dahua Technology [13]. It has an optical lens and a thermal lens to capture optical videos and monitor temperature. The thermal lens has many advantages such as fast response (latency less than 40 ms), wide temperature measurement range (−20 °C to 550 °C) and small error (less than 5 °C). The type of IPC (Fig. 9) is UNO-2372G produced by Advantech [14]. It has Intel ® atom Baytrial E3845 quad core processor up to 1.91 GHZ and 4 GB DDR3L memory.

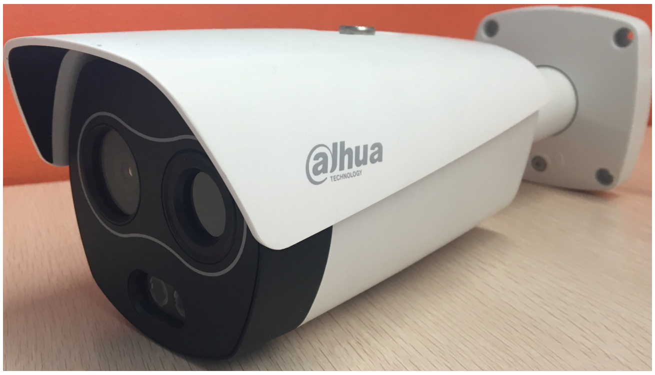

Fig. 8. The bi-spectrum camera in our system.

Fig. 9. The IPC in our system.

4 Experiments and Analyses

To guarantee the generalization of the person detection model, we build the training and testing dataset with our application dataset and multiple public datasets including Pascal VOC (only person) [15], COCO2017 (only person) [16], KITTI [17] and CityPersons [18]. There are 60000 and 80000 images in application dataset and public datasets respectively, and 20% of images in each dataset are randomly chosen as test set and the remainder for training. The results of the following experiments are based on the test set.

The results of different person detection algorithms are listed in Table 2. Compared with original YOLOv3-tiny, the model with ignore label improves the precisions of both public and application datasets, showing its general effectiveness. As a contrast, wide augmentation significantly improves the performance on application dataset while not affects the performance on public datasets. The reason is that wide augmentation adapts the model to the highly concentrated aspect ratio of the application dataset (16:9) with input size of 384 $\times$ 224 instead of the original 288 $\times$ 288. By combining ignore label and wide augmentation (IL $\times$ WA), the model achieves better scores on application dataset, showing the coordination of the two modifications. The last line indicates the model finetuned with 500 hard examples. It further improves the precision on application dataset, proving the effectiveness of iterative model improvement.

Table 2. Performances of different person detection models. AP stands for average precision [23].

Models	Input size	Public datasets			Application dataset		
		AP	AP50	AP75	AP	AP50	AP75
Original Yolov3-tiny	288 $\times$ 288	0.28	0.632	0.252	0.237	0.514	0.176
Ignore label	288 $\times$ 288	0.365	0.724	0.339	0.319	0.613	0.262
Wide augmentation	384 $\times$ 224	0.361	0.708	0.326	0.355	0.684	0.281
IL + WA	384 $\times$ 224	**0.363**	**0.709**	**0.327**	0.362	0.718	0.306
500 hard examples	384 $\times$ 224	0.361	0.706	0.326	**0.367**	**0.739**	**0.309**

We speed up the person detection model above with different model acceleration frameworks, and the inference time is listed in Table 3. Darknet [3] is the original framework, and all the models have the same precision. The inference time is calculated with the same processor of our system. It is clear that all the acceleration frameworks have large speedup over original Darknet. Moreover, TuringNN shows the advantage in efficiency over the other open-source frameworks. It is 17.4% faster than NCNN [12], 48.5% faster than Tensorflow Lite [11] and 370% faster than Darknet, indicating the effectiveness of the optimizations in TuringNN.

Table 3. Speed of different model optimization frameworks.

Framework	Inference time (ms)	Frame per second
Darknet	620	1.61
Tensorflow Lite	196	5.1
NCNN	155	6.45
TuringNN	**132**	**7.58**

In summary, the features of existing fire alarm systems and the proposed system are compared in Table 4. It is obvious that our system has advantage over existing ones in accuracy, speed, cost and so on. The proposed fire alarm system has been applied for kitchen safety in Freshhema for over 6 months. An alarm is defined as effective if the oil temperature has risen into the ranges in Table 1 and nobody is near the stove, i.e. both the temperature measurement and person detection are correct. For each store, the system makes over 100 effective alarms per month. The alarmed cases do not necessarily lead to fire accident, but the warnings do reduce potential fire risks and improve the safety awareness of the staff.

Table 4. Comparison of existing fire alarm systems and the proposed system.

	Existing fire alarm systems	The proposed system
Sensor	Flame, smoke, carbon monoxide detectors	Regular and thermal bi-spectrum camera
Timeliness	After fire happens	Before fire happens
False alarm prevention	None	Person detection
Warning mode	Buzzer, phone call to fire station, delayed response	Buzzer, IM app, phone call to staff, shut off stove, quick response
System upgrade	Device replacement, difficult	Model or parameter change, easy

5 Conclusion

In this paper we propose a smart kitchen fire alarm system. The proposed system prevents fire accidents by monitoring the temperature of cookware with thermal cameras. Moreover, the system achieves quick response and avoids false alarm via person detection. By using YOLOv3-tiny algorithm and model quantification framework TuringNN, we are able to implement the system on small-scale low-cost edge devices for practical situations. Experiments demonstrate the effectiveness of our system. Future research will focus on person detection on thermal videos and improving the precision and recall of the system with the detections of both optical and thermal videos.

References

1. National Fire Protection Association: Structure Fires in Hotel and Motels. Report, September 2015
2. U.S. Fire Administration: Hotel and Motel Fires (2014–2016). Topical Fire Report Series, vol. 19, no. 4 (2018)
3. Redmon, J., Farhadi, A.: YOLOv3: an incremental improvement (2018). arXiv preprint arXiv:1804.02767. https://pjreddie.com/darknet/yolo/
4. Freshhema. (Chinese). https://www.freshhema.com/. Accessed 05 Jan 2020. Introduction in English. https://equalocean.com/retail/20190324-freshhema-future-of-alibaba. Accessed 05 Jan 2020
5. Siemens Switzerland Ltd.: Fire detection in kitchens. https://www.downloads.siemens.com/download-center/Download.aspx?pos=download&fct=getasset&id1=A6V10430602. Accessed 05 Jan 2020
6. Cheng, Y., Cheng, L.: Fire Safety Device for Stove-top Burner. Patent, US5945017A, United Sates (1999)
7. Ren, S., He, K., GirShick, R., Sun, J.: Faster R-CNN: towards real-time object detection with region proposal networks. In: Advances in Neural Information Processing Systems (NIPS) (2015)
8. Liu, W., et al.: SSD: single shot multibox detector. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9905, pp. 21–37. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46448-0_2
9. Redmon, J., Divvala, S., Girshick, R., Farhadi A.: You only look once: unified, real-time object detection. arXiv:1506.02640 (2015)
10. Lin, T., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature pyramid networks for object detection. arXiv:1612.03144 (2016)
11. TensorFlow Lite. https://www.tensorflow.org/lite. Accessed 05 Jan 2020
12. NCNN Github. https://github.com/Tencent/ncnn. Accessed 05 Jan 2020
13. Dahua Technology. https://us.dahuasecurity.com/. Accessed 05 Jan 2020
14. Advantech. https://buy.advantech.com/?country=United%20States. Accessed 05 Jan 2020
15. Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The PASCAL visual object classes challenge 2012 (VOC2012) results
16. Lin, T.-Y., et al.: Microsoft COCO: common objects in context. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8693, pp. 740–755. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10602-1_48

17. Geiger, A., Lenz, P., Urtasun, R.: Are we ready for autonomous driving? The kitti vision benchmark suite. In: CVPR (2012)
18. Zhang, S., Benenson, R., Schiele, B.: CityPersons: a diverse dataset for pedestrian detection. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 4457–4465 (2017)
19. Koorsen Fire & Security, Inc.: Installation, service, and inspection for a range of systems. https://www.koorsen.com/products-services/fire-protection/fire-alarm-systems/. Accessed 09 Apr 2020
20. Nearby Engineers Inc.: Fire Suppression Systems for Commercial Kitchens. https://www.ny-engineers.com/blog/fire-suppression-systems-for-commercial-kitchens. Accessed 09 Apr 2020
21. Encore Fire Protection Inc.: Protecting Your Kitchen: Restaurant Fire Safety Systems. http://blog.encorefireprotection.com/blog/protecting-your-kitchen-restaurant-fire-safety-systems. Accessed 09 Apr 2020
22. Krishnamoorthi, R.: Quantizing deep convolutional networks for efficient inference: A whitepaper. arXiv:1806.08342 (2018)
23. Lin, T.-Y., et al.: Microsoft COCO: common objects in context. arXiv:1405.0312 (2014)