Edge Computing with Embedded AI: Thermal Image Analysis for Occupancy Estimation in Intelligent Buildings

Aly Metwaly
almetw@utu.fi
University of Turku
Turku, Finland

Jorge Peña Queralta
jopequ@utu.fi
University of Turku
Turku, Finland

Victor Kathan Sarker
vikasar@utu.fi
University of Turku
Turku, Finland

Tuan Nguyen Gia
tunggi@utu.fi
University of Turku
Turku, Finland

Omar Nasir
omar.nasir@helvar.com
Helvar Oy Ab
Espoo, Finland

Tomi Westerlund
tovewe@utu.fi
University of Turku
Turku, Finland

ABSTRACT

With the rise of the IoT, there has been a growing demand for people counting and occupancy estimation in Intelligent buildings for adapting their heating, ventilation and cooling systems. This can have a significant impact on energy consumption at a global scale as such systems consume about 40% of electricity and create about 36% of the CO2 emissions in Europe. Previous approaches to occupancy estimation either utilize methods that do not ensure people's privacy when obtaining high accuracy estimations, such as RGB cameras, or utilize thermal or radar sensors with lower accuracy. Thermal vision for people detection has several advantages. It protects people's privacy while being less affected by changes in the environment. In addition, most of the previous image processing approaches rely on streaming the data to the cloud to be analyzed. However, with the development of the more distributed network paradigms edge and fog computing, there has been a trend in moving computation towards the edge of the network. This process of embedding intelligence into end-devices enables more efficient energy consumption and network load distribution. In this work, we present an embedded algorithm for room occupancy estimation based on a thermal sensor with accuracy over the state-of-the-art. We study the performance of a variety of deep learning models on different embedded processors. We achieve a prediction accuracy of 98.9% for people counting estimation with minimal 2 KB RAM utilization. Furthermore, the proposed algorithm has very low latency achieving execution times under 14 ms.

CCS CONCEPTS

• **Computer systems organization → Embedded software**; • **Computing methodologies → Scene understanding**; **Machine learning algorithms**; • **Hardware → Sensors and actuators**; *Digital signal processing*; *Sensor applications and deployments.*

KEYWORDS

Edge Computing; IoT; Embedded Intelligence; Embedded AI; Thermal Imaging; Intelligent Buildings;

ACM Reference Format:
Aly Metwaly, Jorge Peña Queralta, Victor Kathan Sarker, Tuan Nguyen Gia, Omar Nasir, and Tomi Westerlund. 2019. Edge Computing with Embedded AI: Thermal Image Analysis for Occupancy Estimation in Intelligent Buildings . In *INTESA '19: INTelligent Embedded Systems Architectures and Applications, Co-Located with ES WEEK 2019 - October 13–18, 2019, NY, USA.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

In the Industry 4.0 era, cutting-edge technologies such as the IoT and AI are emerging rapidly [7]. These technologies have the potential to impact our daily lives through applications in smart cities, smart homes or intelligent buildings [28]. Different industries are adopting these technologies and transforming them into market opportunities. One promising application is people counting and occupancy estimation in buildings. The acquired information can be utilized for more efficient planning and intelligent space management in smart workplaces. Furthermore, information about the occupancy in buildings and individual rooms can have a significant impact on energy consumption. Buildings are considered the largest energy consumer in Europe using approximately 40% of the total energy and creating about 36% of the total carbon dioxide emissions [8]. Similarly, heating, ventilation, and air conditioning (HVAC) systems in buildings were liable for 38.9% of the total energy consumption in 2017 in the USA [25]. By acquiring reliable information on the occupancy of the buildings, energy consumption can be drastically reduced if HVAC systems in the building are adjusted automatically [13].

Within the IoT, there is a recent trend in more distributed network architectures, in contrast with traditional cloud-centric computing [20, 24]. Edge and fog computing paradigms involve moving computational power and data analysis closer to where the data originates [32, 33]. Combined with artificial intelligence algorithms running at the local network level, this approach enables lower-latency and reduced network load [16, 19, 29]. In this work, we explore the case of embedded artificial intelligence, in which the data analysis runs directly on the sensor node itself.

Previous works on building occupancy estimation or people counting have used RGB cameras [9, 26], motion sensors [17, 18, 34], and, more recently, thermal arrays [2–4, 11]. Motion sensors, such as passive infrared (PIR) sensors, have the drawback of being inaccurate as the number of people increases, as well as limited range [13, 23]. RGB cameras are able to produce high accuracy occupancy estimations, but require computationally intensive image processing [12].

Thermal imaging is one of the most promising sensing technologies. It has been frequently used in smart-city applications [1]. The advantages of the thermal cameras over the RGB cameras are that they are not light-dependent and can work in dark environments. However, thermal cameras with low and medium resolution usually cannot recognize characteristics of the detected person. Therefore, they are unable to identify features of the people in the scene.

In this paper, we propose novel solutions based on Deep Neural Networks (DNNs) to use images from generic, low-resolution, thermal cameras to reliably detect occupancy and count number of people with high accuracy. Compared to previous works, our models achieve higher occupancy prediction accuracy and enable faster image processing than other micro-controller-based implementations. Our approach provides an almost error-free prediction in the case of no occupancy and otherwise matches the number of people in the room. For the tests included in the paper, we have trained the model with a dataset where the occupancy ranged from 0 to 5 persons. However, the same model can be retrained with a more varied dataset to provide a wider range of inference occupancy estimation outputs.

The main contributions of this work are: (i) the design of multiple deep learning models for estimating room occupancy based on thermal images; (ii) the implementation of these models on Arm Cortex M4 and M7 micro-controllers for real-time analysis of thermal images; (iii) the analysis of the performance and impact on the micro-controller computing resources of the proposed models; and (iv) the comparison of our work with the state-of-the-art showing an improved accuracy and reduced computation time.

The rest of the paper is organized as follow: Section 2 reviews existing works in occupancy estimation. Section 3 introduces the concept of embedded AI and describes the types of neural networks utilized in this work. Section 4 describes the data acquisition process, the hardware platforms utilized for testing and the evaluated machine learning models. In Section 5, we demonstrate the superior performance of our algorithm when compared to the state-of-the-art and provide an overview of the models which produced the best results. Finally, Section 6 concludes this work and outlines the directions for future work.

2 RELATED WORK

Oosterhout et al. introduced a head-detection system based on stereo cameras for counting people from video streams [30]. The method is robust and provides high accuracy ranging from 90-95 % for different scenarios. In contrast, we rely on thermal cameras in order to preserve people's privacy and enable fast embedded image processing. In addition, we are able to achieve higher accuracy. Other early approaches which do preserve people's privacy use PIR sensors with some limitations. Wahl et al. presented an approach for people counting for office environments [13] which use distributed PIR sensors enhanced with algorithms to interpret the sensors' information. They explored the performance of two people counting algorithms on this experimental setup with different simulation scenarios. Their approach required a larger number of sensors and the accuracy decreased with an increasing number of people.

Beltran et al. presented a system for estimating occupancy [2] called ThermoSense based on a thermal sensor array and a PIR sensor. It can detect occupancy with an RMS error of approximately 0.35 persons. More recently, Gomez et al. developed a people counting algorithm on thermal images-based on CNN [3]. The used CNNs fit in less than 500 KB of memory and operated on Cortex-M4 MCU. The CNN algorithm could provide an error-free detection accuracy of 53.7% while using 308 KB of the MCU memory. The resolution of the thermal sensors used is 80x60 which shows some features of the people involved in the scene. The execution time for one image is 63 seconds. In our work, we aim for a high error-free accuracy in an office environment using a thermal sensor of 24x32 pixels which cannot detect any features of the people.

Griffiths et al. used a thermal imager with 60x80 pixel resolution [11]. The algorithm used is based on the individuals' height differences for presence detection. Further, the algorithm detects the movement direction of the individuals. Similarly, Tyndall et al. proposed a low-pixel thermal imager system for occupancy estimation [4] and used a classification algorithm. The system is based on Thermosense [2] but is different from Thermosense in the choice of the thermal sensor, positioning of the sensor and the classification algorithm. In our work, we are able to achieve better real-time occupancy estimation accuracy while embedding the AI models in low power microprocessors.

A high accuracy method for estimating room occupancy with thermal array sensors was proposed by Abedi *et al.* [21]. The authors presented a real-time monitoring system which was only able to detect the presence of people in a room giving a binary output. They achieved an accuracy of over 99%. The authors rely on cloud computing for image processing and their model is unable to estimate the exact number of people in the room. In our work, we achieve a similar accuracy while estimating the number of people and embed the algorithms so that it is not required to send raw data to the cloud for processing.

3 EMBEDDED AI

With the increasing pervasiveness of the IoT in all aspects of our daily lives, it is expected that billions of edge devices will be connected to the internet in the near future. These devices will be producing extremely large amounts of data. In the traditional cloud-centric approach, all data acquired at the edge devices is sent to the cloud to be crunched and processed. Then, the results of the analysis and commands are sent back to the edge devices. As the most important information resides on the data analysis results, the process of sending raw data to the cloud can be avoided if part of the computation is moved towards the edge of the network. Within the edge and fog computing paradigms, embedded AI refers to embedding artificial intelligence algorithms into low-power and computationally-constrained devices. Jägare reflects on the benefits of moving data analysis from cloud-centric architectures towards embedded systems for given applications in a recent work [31]. These benefits include (i) reduced latency, increased reliability, and safety in time-critical applications; (ii) overall energy-efficiency and reduced cost with a reduced impact to network traffic and cloud server load; and (iii) enhanced privacy and security, with a lower risk of raw data being exposed, and natural support for applications where privacy is paramount and raw data cannot be shared.

In summary, applying AI at the edge instead of the cloud achieves a more reliable low latency response. Also, it has the potential of providing a better user experience with enhanced security and privacy. However, applying AI algorithms on embedded devices can present significant challenges. Embedded systems are resource-constrained devices because of their low computational power, low memory, and low power consumption requirements. In the rest of this section, we overview the basic concepts for the neural networks that have been studied in this work. Each of these networks has a different impact on system requirements (RAM, Flash) and execution time.

3.0.1 Feedforward Neural Networks (FNNs). also known as Deep FNNs are the basic deep learning models. It is called feedforward because the information flow is only in the forward direction. In other words, there is no feedback connection from the output that is fed to the model [14]. FNNs form the basis of many other significant neural networks such as the convolutional networks. In addition, it is an essential step on the path to the recurrent networks [14]. FNNs is composed of different functions that are chained together. Each function is called a layer and the overall length of the chain is called the depth of the model. The training data shows only the overall output of the whole network which specifying the output of each layer, that's why they are called the hidden layers. Each of the hidden layers is vector-valued and their dimension determines the width of the model which is measured in the number of neurons [14].

3.0.2 Convolutional Neural Networks (CNNs). Convolutional neural network (CNN) employs the convolution mathematical operation instead of general matrix multiplication in at least one of their layers. The CNN is enhanced from the FNN by overcoming some of the FNN disadvantages: sparse connectivity is used in CNNs to reduce the number of weights. On the other hand, Parameter sharing is used to decrease the memory required for neural models. It also reduces the complexity of the model at a given accuracy, which is called the statistical efficiency [5]. A sliding window called kernel is required to perform the convolution process. When convolution is applied in machine learning, the input is usually a multidimensional array and the kernel is usually a multidimensional array of parameters (tensors) that are adjusted by the learning algorithm. In the case of a 2D image, the input would be a frame matrix of the number of pixels and the kernel would be a 2D convolution sliding window [14]. Each layer in CNN has neurons arranged in 3 dimensions: width, height, and depth. The depth is the number of channels (filters) for the layer.

3.0.3 Recurrent Neural Networks (RNNs). RNN is a special form of the FNN with internal states and loops. The fundamental difference is that the FNN neurons are not accessed twice whereas in RNN the neurons can be accessed more than once through the loops in back-propagation. This allows the information to persist in a time-series. This feature makes RNNs used widely in speech recognition and video processing [5]. The RNNs are one of the families that are used for sequential data. It is specialized in processing sequential data in time series. However, RNNs can be applied to 2-dimensional data such as images which is the case in this work [14]. The look-back of the RNN is the number of previous inputs that the network will keep before it performs the back-propagation process. This is a fundamental process that makes the RNN able to keep a time-series of the inputs. Therefore, without back-propagation, each input to the network is treated independently.

In this work, Gated Recurrent Unit (GRU) is used as a recurrent neural network because the GRU has low complexity and high performance in comparison to other variants of the RNNs [15].

Table 1: STM32F401RE (F4) and STM32F722ZE (F7) specs.

	STM32F401RE	STM32F722ZE
Clock	84 MHz	216 MHz
Flash	512 KB	512 KB
SRAM	96 KB	256 KB
Pipeline Stages	3	6 (dual-issue)
Cache	No	8 KB/I&D
I^{2}C	3	3

Table 2: Distribution of samples in the training and test sets.

Dataset		Labels					
		0	1	2	3	4	5
Original	Training	3540	196	229	201	74	125
	Test	881	39	59	66	14	33
Augmented	Training	3540	1568	1832	1608	592	1000
	Test	881	312	472	528	112	264

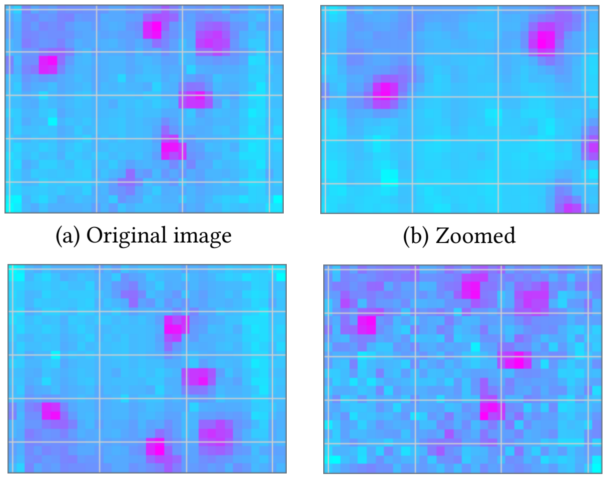
(a) Original image (b) Zoomed
(c) Vertical Flip (d) Added Noise

Figure 1: Different types of data augmentation.

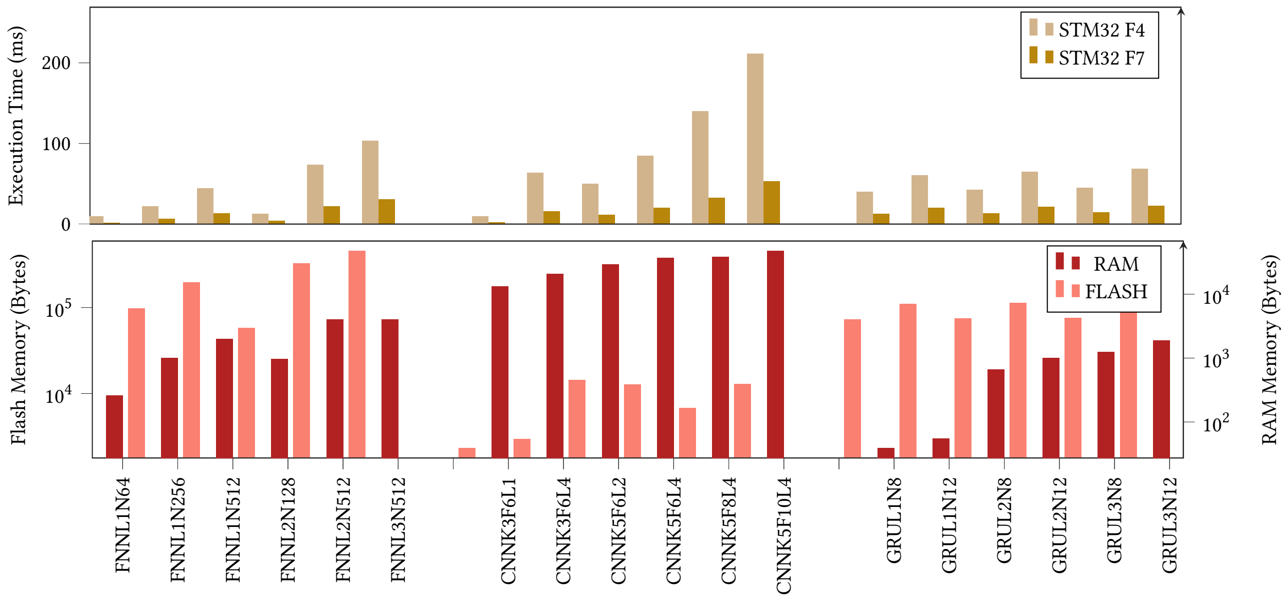

Figure 2: Comparison of execution time, flash and RAM usage for the different models tested.

4 METHODOLOGY

In this work, cloud instances are used to train the models according to the aforementioned needs and CPU instances are used to train the GRUs model due to lower parallelism. The CPU instances run on 4 Intel Xeon Scalable Processors (Cascade Lake) with a turbo clock frequency of 3.6 GHz. Also, GPUs are used to train the FNN and XNN models. GPU instances provide one NVIDIA Tesla K80 Accelerator which runs a pair of NVIDIA GK210 GPUs providing a total of 2496 parallel processing cores. Also, the instance has 4 GPUs of Intel's Broadwell microarchitecture running at 2.7 GHz.

The actual implementation of the embedded intelligence has been carried out with two 32-bit MCUs from ST-Microelectronics. We have used the STM32F401 and the STM32F722 from the Arm Cortex M4 and M7 families respectively for running the DNNs as these have sufficient resources. Two MCUs are used to provide a more extensive evaluation and to overcome some of the limitations that might occur. In the process, the deep learning algorithms are applied first to the MCUs for realizing the proof of concept. The features and available resources of the two MCUs used in the experiments are listed in Table 1.

For applying the deep learning models, an expansion package named X-CUBE-AI is used which helps in applying deep learning algorithms and is capable of converting trained neural networks and generating STM32-optimized library. In addition, the package supports various deep learning frameworks such as Keras which is used in our trained models [27].

4.1 Data Acquisition and Analysis

In this work, a fully calibrated 24x32 pixels FIR thermal sensor array MLX90640 from Melexis is used. This is a medium resolution camera and therefore the images from it are not sufficient to identify features which can help in revealing a person's identity. This conforms to our research requirements of ensuring the privacy of individuals. Moreover, it has integrated sensors to measure the supply voltage (VDD) and ambient temperature (Ta) of the chip. The measurement outputs stored in the internal RAM are accessed through the I^2C interface [22]. In addition, there are two FOVs of the thermal sensor array- 55x35 and 110x75 degrees of which the wider one is used in our experiments. The output is a thermal image where heat signatures are represented by the intensity of the colors.

In our experiments, the MLX90640 is installed in an indoor office environment. For such a contained environment, it can be assumed that people are constantly warmer than the ambient or room temperature [6]. An additional RGB camera is set up to cross-check the total number of people from the results of our experimental setup. This serves as ground truth for model validation and benchmarking. In this work, we have collected data for 2 days and 9 hours resulting in a total of 5457 data samples from the thermal sensor.

The experiments involved zero to five people in the office room. The experiments included one or more person(s) entering and exiting the room sequentially and simultaneously. Moreover, people in the room were sitting, standing or walking.

The total pool of collected samples is divided into 4365 (80%) for the training-set and 1092 (20%) for the test-set. The training set was further subdivided into training and validation sets, with a ratio of 4:1. The case distribution of the training-set and the test-set are shown in Table 2. Here, the case value refers to the ground-truth of the number of people in the room.

4.2 Error Analysis

Hyper-parameter tuning for optimization is an important process in ML which defines a set of optimal parameters for a learning algorithm. These parameters are typically not adjustable or cannot change during the training process. For example, in a DNN, the number of layers and neurons are hyper-parameters.

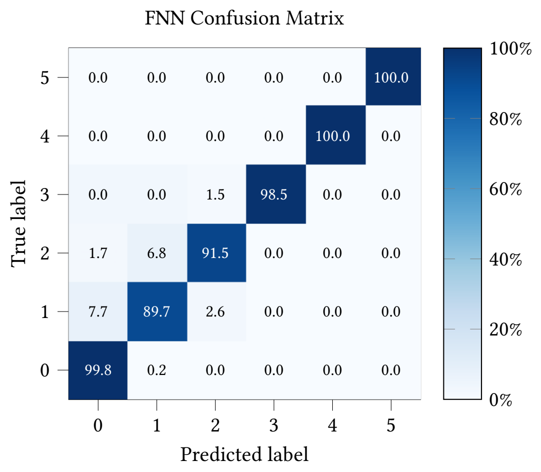

Figure 3: Thermal sensor FNN_L1_N512 original data confusion matrix.

A grid-search is an approach to choose the hyper-parameters where all the possible combinations of hyper-parameters are tried from a grid of parameters values. In this work, we adopted this approach for tuning hyper-parameters. The model is trained on the training subset and its performance is evaluated on the validation subset using minimum squared errors (MSE) as the loss function. For all training epochs, the model state with the lowest validation error is selected as the best representative for a particular set of hyper-parameters. The training itself is performed with an early stopping manner in which the process is terminated if the change in validation error is less than 0.1% for 10 consecutive epochs. The data-set is preprocessed before being fed to the neural network by centering the mean to 0 and scaling to unit variance. Moreover, each layer is augmented with appropriate dropout and *Adam* is used for gradient descent optimization with tuned learning rate value [10]. After finding the best hyper-parameters, the best model is tested with the test-set to measure its prediction accuracy.

5 EXPERIMENTATION AND RESULTS

In this section, the experimental results are presented and analyzed. The purpose of these experiments is to evaluate the chosen MCUs while applying the trained people counting algorithms. This is achieved by running a set of different DNN models on the MCUs in inference mode. That will be followed by an accuracy analysis for the models that work on the MCUs. We utilize the original data-set and an augmented data-set or training and testing the models.

The accuracy of the DNNs trained with the original data-set was remarkably high. Consequently, we decided to make it more challenging for the neural networks by augmenting the data with more corner cases that are expected to be harder to process by the algorithms. The data augmentations used are (i) cropping, (ii) flipping upside down, (iii) flipping left to right, (iv) zooming out, (v) adding random noise, (vi) rotating the image, and (vii) blurring the image. A subset of the different data augmentation techniques utilized is shown in Figure 1. The FNN & GRU models are labeled as *FNN_xL_yN* where x is the depth and y is the width of the model. The CNN models are denoted as *CNN_Kx_Fy_Lz* where x is the kernel size, y the number of filters and z the number of layers.

The total number of samples for the thermal sensor data-set after augmentation is 12709 the same division method mentioned earlier for separating into training, validation and test sets is followed. The augmented data-set samples are divided to 10140 (80%) for the training-set and 2569 (20%) for the test-set. The distribution os the augmented training and the test sets are shown in Table 2.

5.1 Results

The DNNs provide robust and accurate results with the thermal sensor data-sets. The data quality is suitable to result in high accuracy even with the relatively simpler FNNs. The algorithm was able to learn when to detect a temperature signature as a human or another heat source. In Table 3, a side by side comparison of the best performing models is presented. As shown, there are three different novel solutions for the thermal sensor. The solutions have different resources requirements. This allows the possibility of tailoring the algorithm based on the resources available in the MCU. The resources required are shown in Table 3.

The Flash and RAM requirements of the models, together with the execution time for the different models, are presented in Figure 2. The three DNN models that have been used are different in their structure and thus cannot be directly compared. However, we compare them against the targeted application of people counting. An example of the structural difference between network types is that the number of neurons and layers are lower in the GRUs than in the FNNs. This is mainly because the GRUs depends on the look-back and do not need a high width or depth. In consequence, similar MSEs were achieved with a lower number of layers and neurons in GRUs. The Flash requirements for the networks is reported after compression using the X-CUBE-AI package.

In terms of resource utilization, the CNN models have the highest impact on processor resources. Because of the convolution layers, CNNs require larger RAM usage and longer computation times. On the opposite side, FNNs are the simplest models in terms of network structure, and this has a direct relation regarding the memory usage and computation time. GRUs are situated in a middle point. Nonetheless, because of the lower number of neurons and layers in GRU models, their RAM requirements are also lower.

The best accuracy obtained with each of the models is very similar, ranging from 97.27% to 98.90%. The best FNN model achieves a prediction accuracy of 98.90%, which considerably improves the state-of-the-art results in people counting from thermal images. The confusion matrix illustrating the performance of this model is shown in Figure 3. Only the work from Abedi *et al.* [21] achieves higher accuracy. However, in their case, the authors only detect whether the room is empty or not, with a binary output. Moreover, in that work, the machine learning analysis runs on cloud servers. Implementing the models in embedded processors enables a more robust design with lower latency. A comparison with other previous works is summarized in Table 4. Within the works utilizing thermal cameras and estimating the exact number of people in the image, our prediction accuracy is over 15% better than the previous work by Tyndall *et al.* [4]. We also achieve the best accuracy within embedded AI algorithms for any type of thermal or PIR sensor.

Table 3: Summary of prediction performance and system requirements for the best model of each network type.

	Pred. Acc.	F4 Exec. Time	F7 Exec. Time	Alloc. Flash	Alloc. RAM	Test MSE	Valid. MSE
FNN_L1_N512	98.90%	44.141 ms	13.269 ms	196.07 KB	2.01 KB	0.0137	0.004
CNN_K3_F8_L3	98.26%	77.075 ms	18.435 ms	8.46 KB	22.13 KB	0.0174	0.039
GRU_L1_N12	97.27%	60.7 ms	20.097 ms	109.88 KB	0.055 KB	0.030	0.017

Table 4: Comparison of system setup, data analysis technique and results of our method with the state-of-the-art.

		Sensor	Placement	Output	Platform	Processing	Accuracy
Beltran et al.	[2]	PIR+Thermal	Ceiling	Numbered	Tmote Sky	Custom	NA
Gomez et al.	[3]	Thermal	Wall	Numbered	Cortex M4	CNN	53.7%
Tyndall et al.	[4]	PIR+Thermal	Ceiling	Numbered	Arduino	K* algorithm	82.56%
Abedi et al.	[21]	Radar+Thermal	Ceiling	Binary	Cloud	CNN	99%
Zappi et al.	[23]	PIRs	Wall	Numbered (0-3)	GT60 MCU	Custom	89%
Ours		**Thermal**	**Ceiling**	**Numbered (0-5)**	**STM32F**	**FNN**	**98.90%**

6 CONCLUSION AND FUTURE WORK

Knowing the number of people can help manage resources in smart buildings and places where automation can improve management and dramatically reduce the total consumption of electricity hence effectively decreasing greenhouse emissions. In this paper, we presented a novel solution for people counting with high prediction accuracy. The proposed algorithms have low computational, power and memory requirements making those suitable for resource-constrained devices used in IoT-based applications. It is observed that the thermal imaging technique is promising for counting people and more effective than other approaches such as the ones based on RGB cameras. Among the tested algorithms, FNN_L1_N512 resulted in the highest accuracy of 98.90%. The two MCUs are able to run the FNN_L1_N512 algorithm in inference mode. The algorithm utilized 4% of CPU processing cycles on the STM32F401 MCU while using 37% of its flash memory and less than 2.1% of total available RAM. In our experiments, two other models (CNN_K3_F8_L3 and GRU_L1_N12) resulted in similar prediction accuracy, offering various choices for the flash memory and RAM and hence can be tailored according to the available resources of the MCU. In this work, we have focused on novel solutions of embedded AI enhanced thermal sensor for counting people. In future work, we will extend the dataset to include more cases, as well as study the impact of the camera location and distance to subjects on the prediction accuracy.

REFERENCES

[1] A. Anjomshoaa *et al.* 2018. City scanner: Building and scheduling a mobile sensing platform for smart city services. *IEEE Internet of Things Journal* (2018).

[2] A. Beltran *et al.* 2013. Thermosense: Occupancy thermal based sensing for hvac control. In *ACM BuildSys Workshop*. ACM.

[3] A. Gomez *et al.* 2018. Thermal image-based CNN's for ultra-low power people recognition. In *ACM International Conference on Computing Frontiers*. ACM.

[4] A. Tyndall *et al.* 2016. Occupancy Estimation Using a Low-Pixel Count Thermal Imager. *IEEE Sensors Journal* (2016).

[5] B. Moons *et al.* 2018. *Embedded Deep Learning: Algorithms, Architectures and Circuits for Always-on Neural Network Processing* (1st ed.). Springer.

[6] B. Thomas *et al.* 2016. Thermal Imaging Systems for Real-Time Applications in Smart Cities. *Aalborg Universitet* (2016).

[7] C. J. Bartodziej. 2017. The concept industry 4.0. In *The Concept Industry 4.0*. Springer, 27–50.

[8] European Commission. 2002. European union directive on the energy performance of buildings (EPBD). *European Commission, Tech. Rep. 2002/91/EC* (2002).

[9] D. B. Yang *et al.* 2003. Counting people in crowds with a real-time network of simple image sensors. In *Proceedings Ninth IEEE International Conference on Computer Vision*. 122–129 vol.1.

[10] D. P. Kingma *et al.* 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[11] E. Griffiths *et al.* 2018. Privacy-preserving Image Processing with Binocular Thermal Cameras. 1, 4 (2018).

[12] F. Jazizadeh *et al.* 2018. Personalized thermal comfort inference using RGB video images for distributed HVAC control. *Applied Energy* (2018).

[13] F. Wahl *et al.* 2012. A Distributed PIR-based Approach for Estimating People Count in Office Environments. *15th IEEE CSE and 10th IEEE/IFIP EUC*, 640–647.

[14] I. Goodfellow *et al.* 2016. *Deep Learning*. MIT Press.

[15] J. Chung *et al.* 2014. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. *CoRR* (2014), 1–9.

[16] J. Peña Queralta *et al.* 2019. Edge-AI in LoRabased healthcare monitoring: A case study on fall detection system with LSTM Recurrent Neural Networks. In *2019 42nd International Conference on Telecommunications, Signal Processing (TSP)*.

[17] J. Yun *et al.* 2014. Human movement detection and identification using pyroelectric infrared sensors. *Sensors (Switzerland)* 14 (2014).

[18] K. Hashimoto*et al.* 1997. People count system using multi-sensing application. In *Transducers 97*.

[19] L. Qingqing *et al.* 2019. Edge Computing for Mobile Robots: Multi-Robot Feature-Based Lidar Odometry with FPGAs. In *12th ICMU, IEEE*.

[20] L. Qingqing *et al.* 2019. Visual Odometry Offloading in Internet of Vehicles with Compression at the Edge of the Network. In *12th ICMU, IEEE*.

[21] M. Abedi *et al.* 2019. Deep-learning for Occupancy Detection Using Doppler Radar and Infrared Thermal Array Sensors. In *ISARC*, Vol. 36. IAARC Publications.

[22] Melexis. [n.d.]. MLX90640 32x24 IR array. Datasheet.

[23] P. Zappi *et al.* 2007. Enhancing the spatial resolution of presence detection in a PIR based wireless surveillance network. 295 – 300.

[24] R. Mahmud, *et al.* 2018. Fog computing: A taxonomy, survey and future directions. In *Internet of everything*. Springer, 103–130.

[25] S. Koebrich *et al.* 2017. 2017 Renewable Energy Data Book Including Data and Trends for Energy Storage and Electric Vehicles Acknowledgments. (2017), 142.

[26] S. Lu *et al.* 2018. Dynamic HVAC Operations with Real-time Vision-based Occupant Recognition System. In *2018 ASHRAE Winter Conference, Chicago*.

[27] STM. 2019. User manual Getting started with X-CUBE-AI Expansion Package for Artificial Intelligence (AI). January (2019), 1–62.

[28] T. K. L. Hui *et al.* 2017. Major requirements for building Smart Homes in Smart Cities based on Internet of Things technologies. *FGCS* (2017).

[29] T. Nguyen Gia *et al.* 2019. Edge AI in Smart Farming IoT: CNNs at the Edge and Fog Computing with LoRa. In *2019 IEEE AFRICON*.

[30] T. V. Oosterhout *et al.* 2011. Head Detection in Stereo Data for People Counting and Segmentation. 2003 (2011), 620–625.

[31] U. Jägare. 2019. *Embedded Machine Learning Design FD Arm Special Edition*. John Wiley & Sons, Inc. 30 pages.

[32] V. K. Sarker *et al.* 2019. Offloading SLAM for Indoor Mobile Robots with Edge-Fog-Cloud Computing. In *ICASERT*.

[33] V. K. Sarker *et al.* 2019. A Survey on LoRa for IoT: Integrating Edge Computing. In *Int. Workshop on Smart Living with IoT, Cloud and Edge Computing*.

[34] Y. Agarwal *et al.* 2010. Occupancy-driven Energy Management for Smart Building Automation. In *ACM BuildSys '10 Workshop*. ACM, 1–6.