AI

MDPI

Article

Memory-Efficient AI Algorithm for Infant Sleeping Death Syndrome Detection in Smart Buildings

Qian Huang [1,*], Chenghung Hsieh [2], Jiaen Hsieh [2] and Chunchen Liu [2]

[1] School of Architecture, Southern Illinois University, Carbondale, IL 62901, USA
[2] Kneron Inc., San Diego, CA 92121, USA; Samuel.hsieh@kneron.us (C.H.); Andy.hsieh@kneron.us (J.H.); albert@kneron.us (C.L.)
* Correspondence: qhuang@siu.edu

Abstract: Artificial intelligence (AI) is fundamentally transforming smart buildings by increasing energy efficiency and operational productivity, improving life experience, and providing better healthcare services. Sudden Infant Death Syndrome (SIDS) is an unexpected and unexplained death of infants under one year old. Previous research reports that sleeping on the back can significantly reduce the risk of SIDS. Existing sensor-based wearable or touchable monitors have serious drawbacks such as inconvenience and false alarm, so they are not attractive in monitoring infant sleeping postures. Several recent studies use a camera, portable electronics, and AI algorithm to monitor the sleep postures of infants. However, there are two major bottlenecks that prevent AI from detecting potential baby sleeping hazards in smart buildings. In order to overcome these bottlenecks, in this work, we create a complete dataset containing 10,240 day and night vision samples, and use post-training weight quantization to solve the huge memory demand problem. Experimental results verify the effectiveness and benefits of our proposed idea. Compared with the state-of-the-art AI algorithms in the literature, the proposed method reduces memory footprint by at least 89%, while achieving a similar high detection accuracy of about 90%. Our proposed AI algorithm only requires 6.4 MB of memory space, while other existing AI algorithms for sleep posture detection require 58.2 MB to 275 MB of memory space. This comparison shows that the memory is reduced by at least 9 times without sacrificing the detection accuracy. Therefore, our proposed memory-efficient AI algorithm has great potential to be deployed and to run on edge devices, such as micro-controllers and Raspberry Pi, which have low memory footprint, limited power budget, and constrained computing resources.

Keywords: SIDS; memory-efficient; convolutional neural networks; false detection

Citation: Huang, Q.; Hsieh, C.; Hsieh, J.; Liu, C. Memory-Efficient AI Algorithm for Infant Sleeping Death Syndrome Detection in Smart Buildings. *AI* **2021**, *2*, 705–719. https://doi.org/10.3390/ai2040042

Academic Editors: Antonio Fernaández-Caballero, Byung-Gyu Kim and Hugo Pedro Proença

Received: 8 November 2021
Accepted: 5 December 2021
Published: 8 December 2021

1. Introduction

Information technology, especially the Internet of Things (IoT) and artificial intelligence (AI), becomes increasingly popular in smart building applications, such as occupancy estimation for energy-efficient building operations [1,2], and demand-oriented air conditioners [3]. For example, with the help of distributed household IoT devices, AI algorithms have been widely used to model the energy consumption characteristics of smart buildings and find the optimal solutions of parameter thresholds and control parameters. AI algorithms have also been studied to intelligently interpret the visual contents of surveillance cameras and identify the number of residents and their locations in smart buildings. Thus, the operation of air conditioners is adjusted to provide "just-right" heating or cooling services. In addition, smart buildings can adjust the indoor thermal environment, such as temperature, humidity, or airflow, to improve the comfort of building occupants [4]. Large companies, such as IBM or Intel, are also committed to developing AI algorithms for building performance optimization.

In the United States, Sudden Infant Death Syndrome (SIDS) is one of the leading causes of sudden and unexpected death in babies under one year of old. Many studies point out that letting babies sleep on their stomachs can easily lead to SIDS [5,6]. Therefore, the American Academy of Pediatrics recommends that babies should sleep on their backs because this can keep the airway open. It is reported that babies have the lowest risk when sleeping on their backs, followed by sleeping on their sides. Sleeping on the stomach is at the highest risk, because it compresses a baby's chin, narrows the airway, and restricts breathing. However, in practice, it is difficult to let babies always sleep on their backs because it is easy for them to roll over to sleep on their stomachs.

In order to monitor sleeping status, a series of sensor-based wearable or touchable monitors have been developed [7–12]. For example, IoT-based smart posture detection systems have been developed in [10–12], where a pressure sensing mattress is used to collect body pressure data that are processed for posture recognition. Sleep experiments were conducted on an infant in [10], and the reported classification of baby sleep posture reached 88%. In [11], pressure sensors are placed in a sensing cushion, which is used to collect ten children's sitting pressure data. Although infant sleep posture is not involved to detect in [11], the average classification accuracy for children sitting posture is 95%. In [12], the authors proposed to recognize sleep positions with body pressure images and achieved a high recognition accuracy. When various sensors are placed on babies, these baby monitors can track the breathing, body temperature, heart rate of sleeping babies, etc. Then, if a baby monitor observes some abnormal activity, such as stopping breathing or slowing down heart rates, it will send a warning alert to parents. Although it sounds attractive, these wearable or touchable baby monitors suffer from two limitations. First, these monitor systems include various electrodes or sensors located on a crib mattress or the waist and feet of an infant's body. In order to collect reliable data, these pads and sensors need to fit or touch a baby's body well at any time of sleep. In fact, it is inconvenient for babies to always wear these pads or sensors correctly when they sleep. Second, these baby monitors often send out false alarms, which can increase the anxiety of many parents [13,14]. Therefore, parents are most likely to suffer from increased stress or even depression, which affects their sleep quality and emotion.

In order to get rid of the shortcomings of wearable sensor-based baby monitors, researchers began to investigate contactless camera-based monitoring, which detects sleeping postures through cameras and AI algorithms. The researchers in [15] predict that future research on sleep health will be data-driven and AI algorithms will play a critical role. Instead of using wearable body sensors or electrodes for signal collection, AI algorithms analyze the output of cameras and classify sleeping postures. Compared with sensor-based baby monitors, this approach is user convenient and cost-effective. In [16], infrared cameras and depth sensors were used to collect data, and then a convolutional neural network (CNN) classifies sleeping postures with an accuracy of 94%. However, this idea was only verified on a small dataset containing 1880 samples, and this approach has not been validated in the baby sleep scenarios. Furthermore, due to the use of depth sensors and infrared cameras, its hardware cost is expensive. Later, in order to reduce the system cost, the researchers in [17] used 4250 daytime baby sleep images from ordinary cameras to explore eight different CNN architectures. The highest classification accuracy of 87.8% is achieved in a CNN consisting of four convolutional layers and two dense layers. To further increase the classification accuracy, the researcher in [18] explored three CNN architectures. Inspired by GoogLeNets and ResNets, the researcher proposed to add skip connections to standard CNN architectures. Skipping effectively simplifies the network by using an average pool on each feature at the end, so keeps fairly low parameters. The dataset for baby sleep images is the same as [17]. Besides, in order to accommodate his CNN architectures in portable electronics, the researcher [18] proposed to reduce the number of feature maps. Thus, based on a ResNet network with 16 convolution layers and 3 dense layers, the corresponding classification accuracy is 89%. Recently, the researcher [19] proposed to use DenseNet-121 for baby sleep posture classification. DenseNet, also known as dense convolutional network, is a type of convolutional neural networks, in which each layer is connected to all subsequent layers. Since each layer in DenseNets receives collective knowledge from all preceding layers, the information flow among different layers is enhanced. Therefore, this type of network is thinner and more compact [20]. Compared with other AI algorithms, fewer parameters and higher accuracy can be potentially achieved through dense connection. As a result, DenseNet-121 tends to have fewer parameters and a smaller memory footprint. Unfortunately, the researcher [19] only demonstrated his AI algorithm function very well with baby doll pictures, but did not evaluate the classification accuracy using real infant sleep images. Moreover, in [21], a CNN architecture (i.e., Inception-v3 [22]) with transfer learning is used for sleep posture classification. As a widely used image recognition model, Inception-v3 improves the computational efficiency and meanwhile keeps fewer parameters. Although the classification of adult sleep scenes shows an accuracy of around 90% on a dataset with only 1200 non-baby sleep images, the effectiveness of this Inception-v3 architecture has not been tested on real infant sleep datasets.

Table 1 summarizes the accuracy and disadvantages of these existing AI algorithms. From the above discussion and Table 1, it is clear that these existing contactless camera-based baby sleep monitoring studies have not fully met the requirements of AI for edge computing in smart buildings [23–25]. To date, two major bottlenecks are preventing AI from detecting potential infant sleep hazards in smart buildings. First, current datasets of baby sleep posture are not large or diverse. Generally speaking, the performance of AI algorithms is improved by adding more training samples [26], and a high-diversity dataset can maximize the information contained [27]. However, the researchers [16] use 1880 data samples, the researchers [17,18] use 4250 daytime baby sleep images, the researchers [21] use 1200 data samples, while the researcher [19] uses baby doll pictures to approximate real baby sleep images. Although babies sleep at night most of the time, existing datasets do not contain night-vision sleep images. Therefore, it is necessary to generate a large and diverse baby sleep posture dataset to train and evaluate AI algorithms. Second, as stated in [18,19], memory constraint is a major challenge for using deep learning AI algorithms in edge computing systems. AI algorithms must not only fit in the program memory of edge computing systems (such as micro-controllers, Raspberry Pi), but also leave space in the memory so that operating systems or CPU kernels can run smoothly. For example, under the Raspberry Pi 3 A+'s maximum memory constraint of 512 MB, it can run lightweight programs and scripts. Therefore, if an AI algorithm requires several hundred Megabytes of memory, it may not be able to run on edge computing systems. To deal with this challenge, AI algorithms must be optimized to a small memory footprint for real-time operations on edge systems.

Table 1. Summary of Existing AI Algorithms for Contactless Camera-Based Sleep Posture Detection.

Existing Work	Year	AI Algorithm	Accuracy	Disadvantages
[16]	2016	CNN with 3 convolution and 2 dense layers	94%	1. Expensive due to using a combination of depth sensors and infrared camera; 2. A small dataset (1880 samples) 3. Does not consider minimizing memory footprint
[17]	2020	CNN with 4 convolution and 2 dense layers	88%	1. A small dataset (4250 samples) 2. Diversity issue: all samples are daytime images 3. A large memory footprint (275 MB)
[18]	2021	ResNet with 16 convolution and 3 dense layers	89%	1. A small dataset (4250 samples) 2. Diversity issue: all samples are daytime images 3. A large memory footprint (147 MB)
[19]	2021	DenseNet-121	N/A	1. A small dataset with baby doll images 2. Lack of real baby sleep images 3. A large memory footprint (58.2 MB)
[21]	2021	Inception-V3	90%	1. A small dataset (1200 samples) 2. Lack of real baby sleep images 3. A large memory footprint (175.7 MB)
This work	2021	CNN with post-training weight quantization	90%	1. Post-training weight quantization may cause a slight decrease in accuracy

In order to solve the aforementioned two research bottlenecks, in this work, we investigate and propose an optimized AI algorithm for infant sleep posture classification. Regarding the contribution of the body of knowledge, this work makes the following two contributions: (1) we have generated a large and diverse dataset for training and evaluating AI algorithms. This dataset contains 10,240 day and night-vision baby sleep images. (2) We propose a new AI algorithm and use the post-training weight quantization technique to minimize memory usage. In this way, the data type of weight parameters in our AI algorithm is converted from 32-bit floating points to 8-bit integers. Thus, these quantized weights are easy to store and run in many edge computing devices (e.g., 8-bit ATmega328P micro-controller). In order to evaluate the proposed AI algorithm, we have implemented it in a Python program and run it on TensorFlow and Keras platforms. The experimental results show that with a very small memory footprint of 6.4 MB, a classification accuracy of about 90% is obtained. Compared with the state-of-the-art AI algorithms in the literature, the proposed idea achieves a comparable detection accuracy, while the memory footprint decreases from at least 58.2 MB to 6.4 MB, a reduction of at least 9 times. Therefore, our proposed memory-efficient AI algorithm has great potential to be deployed and to run on edge devices, such as micro-controllers and Raspberry Pi, which have low memory footprint, limited power budget, and constrained computing resources.

2. AI Algorithm

2.1. Proposed AI Algorithm—Convolutional Neural Networks (CNNs)

In this work, we choose CNN because it is the simplest neural network architecture that transforms input images into output classification results. CNN has been utilized to recognize heavy construction equipment [28], safety hardhats [29], and baby movement tracking [30]. Figure 1 shows the proposed AI algorithm for infant sleep posture classification. This CNN consists of a series of stacked layers, including convolutional layers, pooling layers, and dense layers. Input images are infant sleeping pictures with a dimension of $256 \times 256 \times 3$, which indicates a width of 256 pixels, a height of 256 pixels, and 3 color channels of RGB. In this AI algorithm, input images undergo multiple convolutions to obtain advanced features for binary classification.

Next, let us describe the network layers of this CNN. Generally speaking, these convolution and pooling layers are for feature extraction, while these dense layers are for object classification. Convolution refers to sliding convolution kernels (i.e., filters) on inputs to sweep over the full inputs and perform linear matrix multiplications. As shown in Figure 1, in our algorithm, the size of each filter is 3×3, and the number of filters is 16, 32, 64, and 64, respectively. Since the convolution layer is the core building block, these four filters involve a lot of matrix multiplications to extract image features. Pooling layers are used to effectively down sample the output dimension of the prior layer (i.e., convolution layer). As a result, the pooling operation reduces the number of parameters to be trained as well as the amount of computation performed by the following layer. In our algorithm, we choose a pooling size of 2×2 to take the maximum value in a 2×2 pooling window. In this way, the most present features after convolution are retained and highlighted. As listed in Table 2, the pooling operation halves each dimension. For example, the output shape of the first convolution layer is $256 \times 256 \times 16$, in contrast, the output shape of the first pooling layer is $128 \times 128 \times 16$. The purpose of batch normalization is to normalize the outputs of the previous layer. Specifically, values are normalized by subtracting the mean and then dividing by the standard deviation. In this way, all the values lie on a common scale between 0 and 1. Thus, the gradient explosion problem is alleviated by using batch normalization [31,32], and extreme gradients accumulated from the previous convolution and pooling layers are eliminated. Batch normalization helps to make our algorithm more stable during training, thereby accelerating the training process and reducing the overfitting phenomena [33]. The flatten layer converts input data into a one-dimensional array to facilitate the next layer (i.e., dense layer). Here, the output of the last pooling layer is transformed into a single long feature vector. The dense layer, also known as a fully-connected layer, serves the actual classification function. In our algorithm, there are three dense layers with 384, 128, and 1 neuron, respectively. Each dense layer multiplies the input by a weight matrix and then is adjusted by adding a bias vector. Since there are two possible classification results for output prediction, the sigmoid function is used to map any input to an output ranging from 0 to 1. In our algorithm, the loss function is binary_crossentropy, which computes the cross-entropy loss between true labels and predicted labels. In short, through these stacked layers, this CNN converts pixel values of input images layer by layer to final classification.

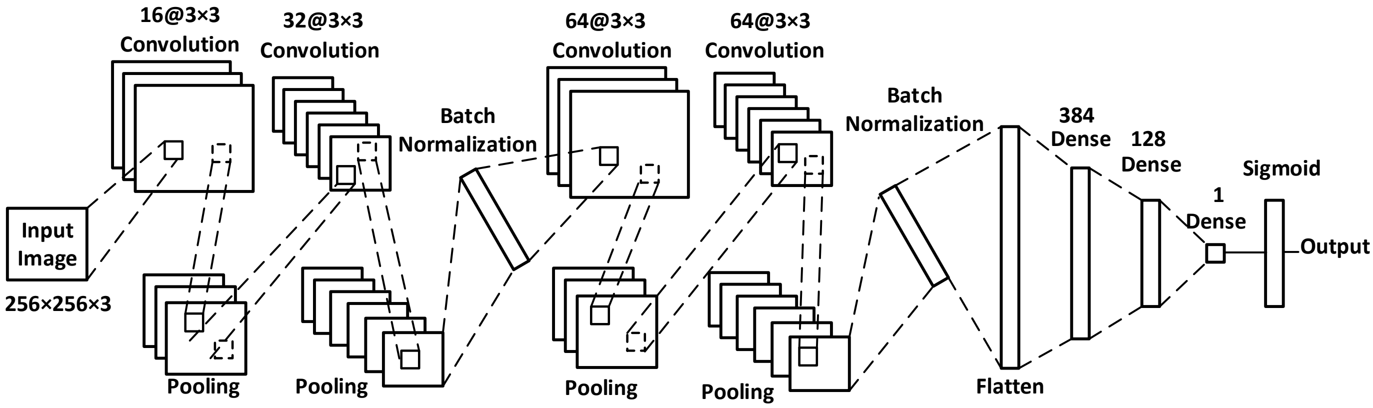

Figure 1. Proposed AI algorithm for infant sleep posture classification.

Table 2 lists detailed information on the model layer, type, number of filters, output shape, and number of parameters. The total number of trainable parameters is about 6.4 million. We can see that most of the parameters are located at the big fully connected layer after the flatten layer (about 6.3 million of the total 6.4 million parameters). Compared with the existing CNN algorithm [17], this CNN is composed of fewer filters and greatly reduces the size of dense layers, thereby reducing the memory footprint. In addition, batch normalization layers are proposed to mitigate the gradient explosion problem and improve classification accuracy.

Table 2. Summary of existing AI algorithms for contactless camera-based sleep posture detection.

Layer Name	Layer Type	Number of Filters	Output Shape	Number of Parameters
Conv2d	Conv2D	16	(256, 256, 16)	448
Max_pooling2d	MaxPooling2D		(128, 128, 16)	0
Conv2d_1	Conv2D	32	(128, 128, 32)	4640
Max_pooling2d_1	MaxPooling2D		(64, 64, 32)	0
Batch_normalization	Batch Normalization		(64, 64, 32)	128
Conv2d_2	Conv2D	64	(64, 64, 64)	18,496
Max_pooling2d_2	MaxPooling2D		(32, 32, 64)	0
Conv2d_3	Conv2D	64	(32, 32, 64)	36,928
Max_pooling2d_3	Maxpooling2D		(16, 16, 64)	0
Batch_normalization	Batch Normalization		(16, 16, 64)	256
Flatten	Flatten		16,384	0
Dense	Dense		384	6,291,840
Dropout	Dropout		384	0
Dense_1	Dense		128	49,280
Dense_2	Dense		1	129

2.2. Post–Training Weight Quantization

When the 6.4 million floating-point parameters in Table 2 are implemented in edge systems, they occupy 51.3 MB of memory space, which is expensive and often is not available to use. In order to make the proposed CNN run smoothly in memory-constrained edge systems, we propose to apply post-training weight quantization to our pre-trained AI algorithms. There are two benefits of weight quantization: (a) reducing memory footprint to save parameters, and (b) accelerating computation to enable smooth and fast running AI algorithms on edge systems.

Figure 2a shows a traditional convolution operation that does not involve weight quantization, so the weight defaults to 32-bit floating-point data. As illustrated in Figure 2b, instead of the 32-bit floating-point type, each weight of the pre-trained AI algorithm is converted to an 8-bit integer type. Thus, it is estimated that the memory usage of AI algorithms can be reduced a lot through weight quantization.

As quantization noise occurs when a continuous random variable is converted to a discrete one, quantization noise reduces the precision of weights, it may lead to a decrease in classification accuracy. Fortunately, researchers have found that weight precision is not very sensitive for deep learning AI algorithms. As a result, deep AI algorithms can get along well with small changes in weights due to quantization. Prior studies have reported that 8-bit post-training weight quantization may slightly reduce the accuracy of the model, while significantly improving the hardware computation latency [34,35].

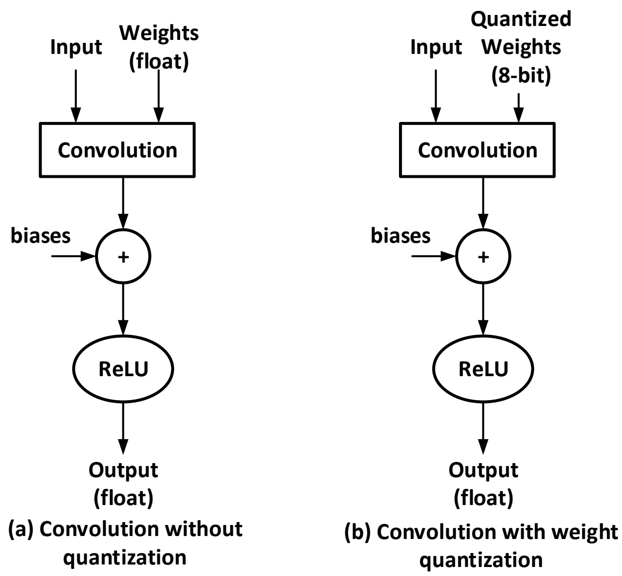

Figure 2. An example illustrating the post-training weight quantization process of the proposed AI algorithm.

3. Experiments and Discussion

3.1. Datasets Generation

Datasets are critical in deep learning since AI algorithms rely heavily on data [36,37]. The rule of thumb is that a sufficient dataset needs to contain at least 10 times the number of trainable parameters in an AI algorithm. To meet this condition, we generated three datasets (i.e., daytime dataset, night-vision dataset, and mixed dataset in Table 3). The mixed dataset is a large and diverse dataset containing 10,240 day and night vision samples. As illustrated in Figure 3, these night-vision images are converted from daytime images, so this mixed dataset covers both daytime and night-vision scenes. Each dataset is randomly spitted into 70% for the training set, 20% for the validation set, and 10% for the testing set. The training and validation sets are used for training, tuning, and evaluation of AI algorithms, while the testing set is used to estimate the final prediction performance after completing the training phase.

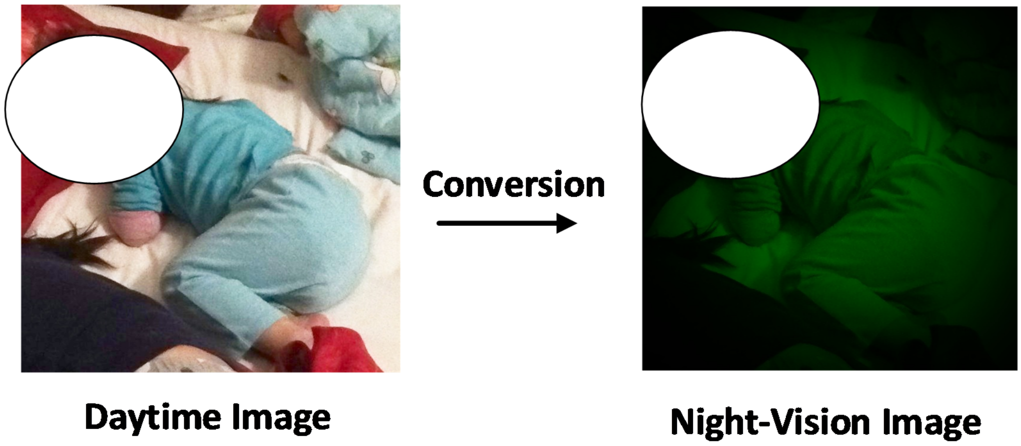

Figure 3. An example of converting a daytime image into a night-vision image for infant sleep posture detection. The child's face is hidden for privacy.

Table 3. Summary of three datasets generated for AI algorithms in this work.

Dataset	Subset	Number of Samples	Percentage
Daytime dataset (5120 daytime images)	Training Set	3584	70%
	Validation Set	1024	20%
	Testing Set	512	10%
Night-vision dataset (5120 night-vision images)	Training Set	3584	70%
	Validation Set	1024	20%
	Testing Set	512	10%
Mixed dataset (10,240 daytime and night vision images)	Training Set	7168	70%
	Validation Set	2048	20%
	Testing Set	1024	10%

3.2. Experimental Environment and Setup

In this work, we use TensorFlow and Keras to train and evaluate our proposed AI algorithm. TensorFlow is an open-source software platform for machine learning [38], and it supports a variety of attractive programming features in deep learning, such as the efficient execution of tensor operations on GPUs. Keras is an open-source application programming interface (API) written in Python that can run on the TensorFlow platform. By providing a user-friendly interface and functions, Keras facilitates us to explore the potential and scalability of TensorFlow. In this study, TensorFlow and Keras API run on a hardware computing system, which consists of a 64-bit Ubuntu operating system and four NVIDIA TITAN XP graphics processors (GPUs). The memory size of each GPU is 12 GB and the operating frequency is 1582 MHz. Therefore, the device memory bandwidth reaches 578 GB/s, which is enough to support 12.15 TFLOPs of full-precision floating-point (32-bit) computing performance. To accelerate deep learning and model training, we use NVIDIA CUDA toolkit 11.0 that includes GPU-accelerated libraries, compilers, runtime libraries for debugging and optimization. Figure 4a shows the traditional AI training and evaluation process without weight quantization. The AI algorithm is trained with the stochastic gradient descent (SGD) optimizer on the training and validation sets. Then, the well-trained AI model is evaluated with the testing set to obtain the classification accuracy. In contrast, as shown in Figure 4b, the weight post-quantization process is added. Through it, the well-trained AI model becomes a quantized AI model for evaluation. In this work, Google's TensorFlow Lite tool is adopted to perform the post-training weight quantization process.

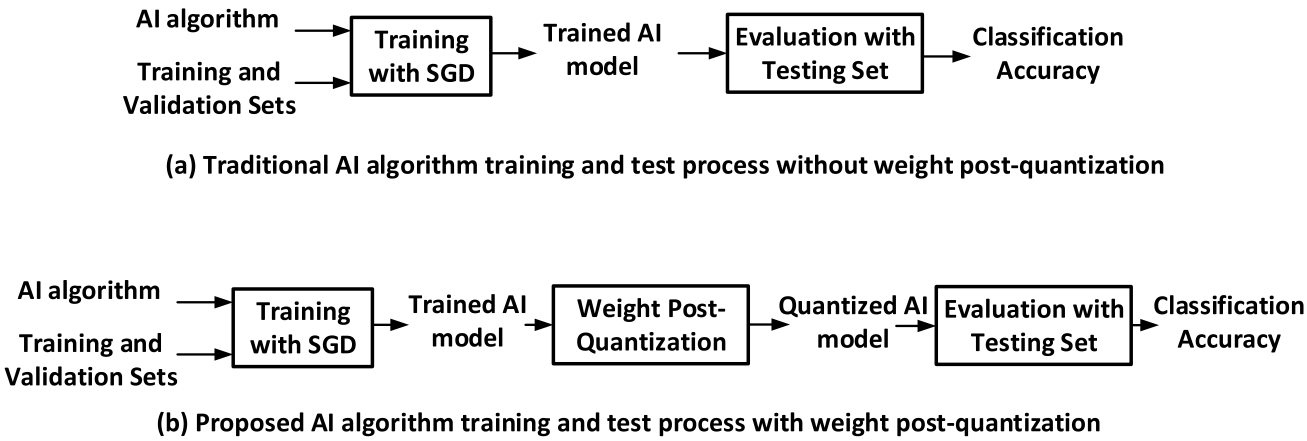

Figure 4. (**a**) Traditional AI training and evaluation process without weight quantization, and (**b**) Proposed AI training and evaluation process with post-training weight quantization.

3.3. Experimental Results and Discussion

All the weights of the AI algorithm are trained by the SGD optimizer [39,40]. The learning rate is a hyper-parameter that controls the speed at which the SGD optimizer updates weights to their best values. Therefore, the learning rate is viewed as an important hyper-parameter to tune for training deep neural networks. If the learning rate is fixed, a large learning rate may learn faster, but there is a risk of reaching sub-optimal weight values. Although the training process is slow, it is necessary to set a very small learning rate so that weights are stable at their global optimal values. In contrast, learning rate decay can dynamically adapt learning steps to reduce training time and help the network converge near the minimum [41]. Deep network training usually starts from a relatively large learning rate in the beginning, then decreases the learning rate during training to allow more fine-grained weight updates. Therefore, in this work, we use the exponential decay function in Keras to gradually reduce the learning rate over time. We set up an initial learning rate of 0.001 and a learning rate decay parameter of 10^{-6}. In our experiments, we found the initial learning rate and its decay parameter are not sensitive to the datasets. Moreover, the number of epochs is set to 250, which is long enough to allow network training to converge well. Instead of training individual input images, a mini-batch size of 64 is selected so that 64 input images are learned as a group.

Let us first look at the experimental results of our AI algorithms before applying weight quantization. In order to check if our generated datasets are good, we run experiments on the daytime dataset, night-vision dataset, and mixed dataset, respectively. Figure 5 shows the experimental results of the loss and classification accuracy on the training and validation sets. We can see that the validation loss has an initial high value and then gradually decreases. After 200 epochs, the validation loss becomes very flat, which indicates the AI model fits well without overfitting. In addition, the training accuracy and validation accuracy are finally stabilized at around 0.99 and 0.92, respectively. Note that the validation set is used to fine-tune the weights of the AI algorithm for accuracy improvement. Figures 6 and 7 show similar loss curves and accuracy curves. The validation accuracy on the night-vision dataset is around 0.9, slightly lower than the training on the daytime dataset.

Next, we perform weight quantization on these well-trained AI models. Then, we run experiments on the testing sets to obtain the final test accuracy after completing the training phase. Since the existing works in the literature only provide experimental results on datasets containing daytime baby sleep images, we compare our AI algorithm on the daytime dataset with them for a fair comparison. As shown in Table 4, the memory footprints in [17–19,21] are 275 MB, 174 MB, 58.2 MB, and 175.7 MB, respectively. Due to memory limitations, these AI algorithms may not fit on edge computing systems, such as microcontrollers, smaller FPGAs, or low-end Raspberry PIs. Without applying weight quantization, our proposed AI algorithm results in a memory footprint of 51.3 MB, which saves at least 12% of memory space compared to these existing works. Thanks to the weight quantization, the memory footprint can be further decreased by 44.9 MB, which means another 88% reduction. Meanwhile, due to weight quantization, the test accuracy is slightly improved from 90.8% to 91.6%. As a result, our proposed AI algorithm only consumes 6.4 MB of memory.

Table 4. Comparison with existing contactless camera-based AI algorithms for baby sleep posture detection on the daytime dataset.

Existing Work	Dataset	Weight Quantization	Memory Footprint	Test Accuracy
[17]	4250 daytime images	No	275 MB	88%
[18]	4250 daytime images	No	174 MB	89%
[19]	Baby doll pictures instead of real baby pictures	No	58.2 MB	N/A
[21]	1200 non-baby sleep images	No	175.7 MB	90.2%
This work	Daytime dataset (5120 images)	No	51.3 MB	90.8%
		Yes	6.4 MB	91.6%

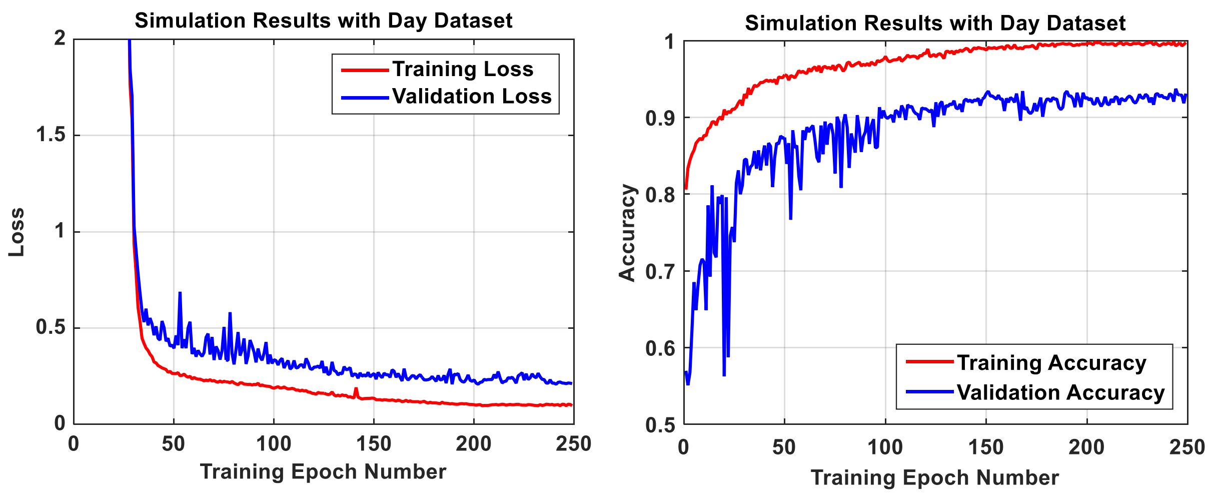

Figure 5. Simulation results of our AI algorithm on the daytime dataset before weight quantization.

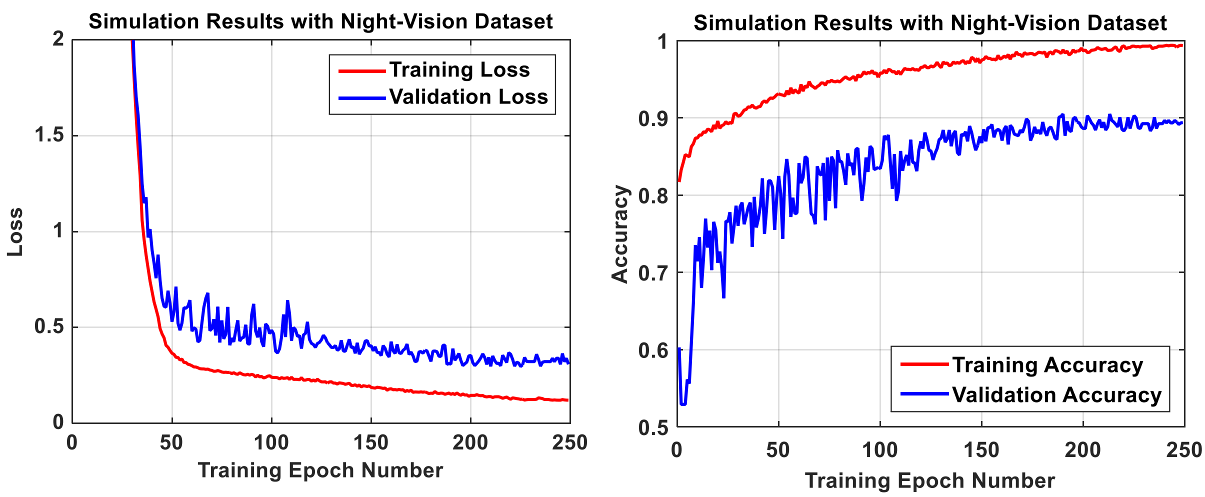

Figure 6. Simulation results of our AI algorithm on the night-vision dataset before weight quantization.

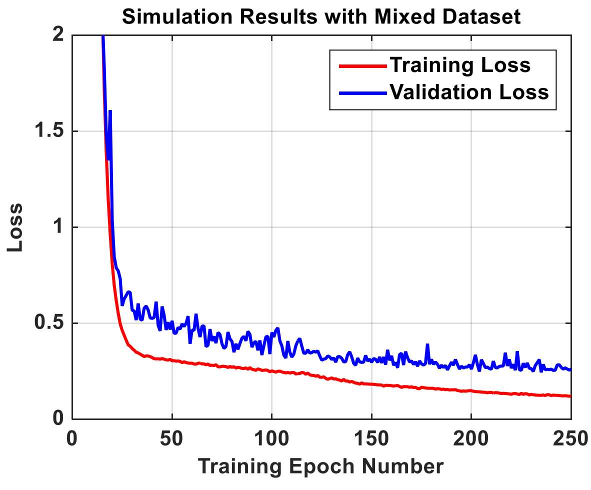

Figure 7. Simulation results of our AI algorithm on the mixed dataset before weight quantization.

The researchers [19,21] only reported that their AI algorithms are DenseNet-121 and Inception-v3, but did not evaluate the classification accuracy using real baby sleep images. Therefore, in order to make a fair comparison, we evaluated these existing AI algorithms [17–19,21] on the same dataset (i.e., the mixed dataset), plotted their experimental results in Figures 8–11, and listed their performance results in Table 5. We can see that the best test accuracy of 91.0% corresponds to [21]. Compared with these existing AI algorithms, this work reduces memory footprint by at least 89%, while maintaining similar classification accuracy. In our proposed AI algorithm, the use of weight quantization leads to a negligible degradation in test accuracy (i.e., 0.2%). The experimental results in Table 5 are also plotted and visualized in Figure 12, where the x-axis and y-axis represent memory usage and test accuracy, respectively. Compared with the existing work [18], this work has achieved a 27-fold reduction in memory footprint and a 0.7% improvement in test accuracy. Compared with the existing work [19], this work has achieved a 9-fold reduction in memory usage with a 1.3% drop in test accuracy.

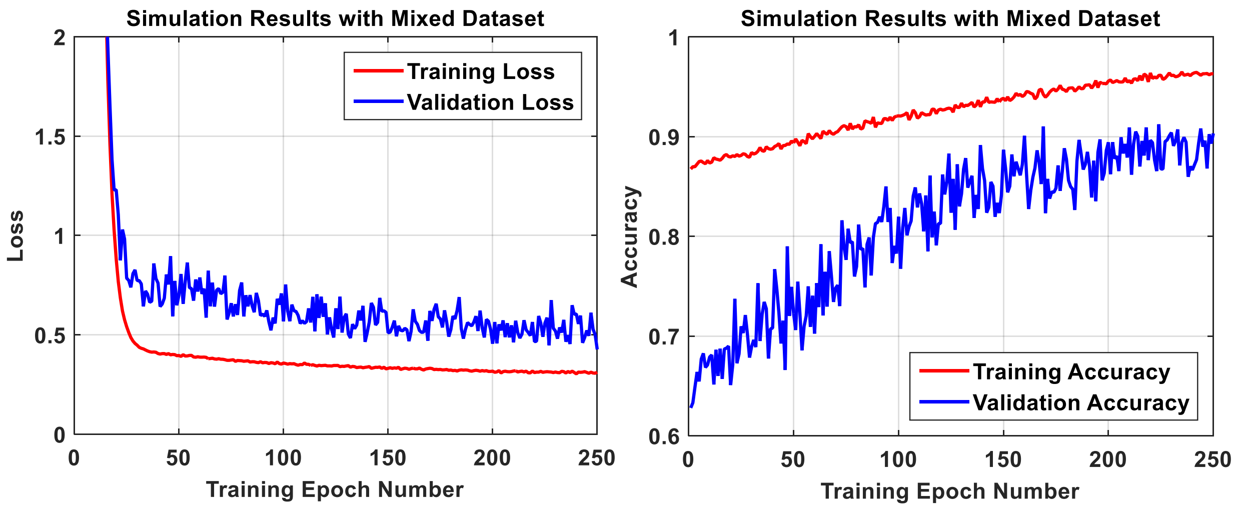

Figure 8. Simulation results of the AI algorithm [17] on the mixed dataset.

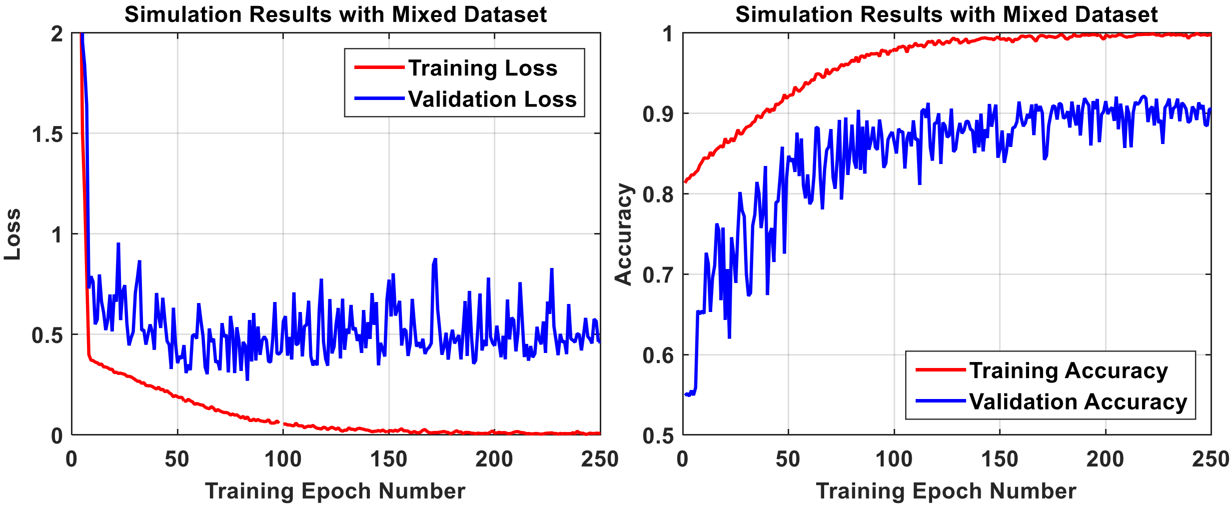

Figure 9. Simulation results of the AI algorithm [18] on the mixed dataset.

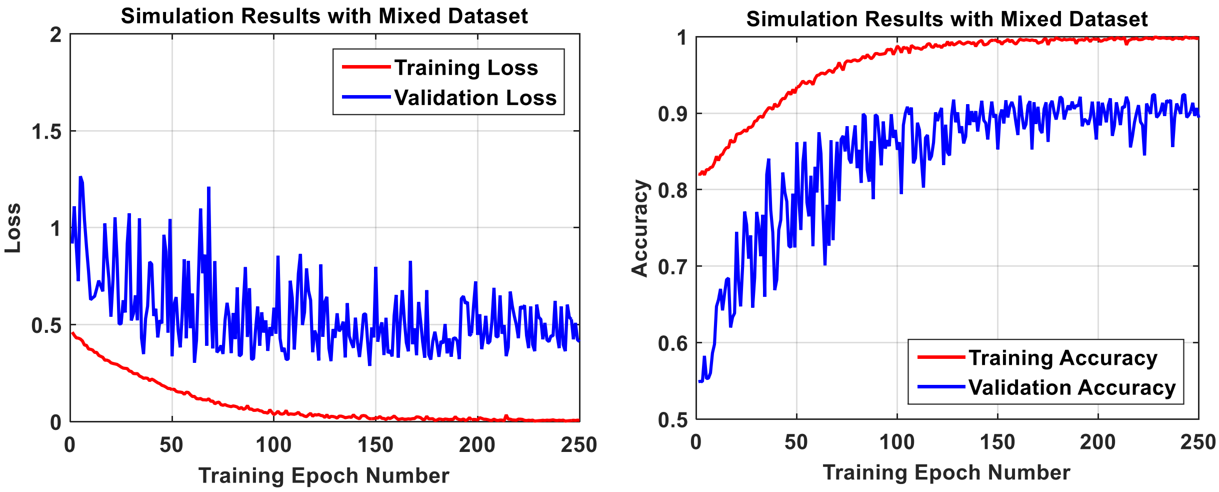

Figure 10. Simulation results of the AI algorithm [19] on the mixed dataset.

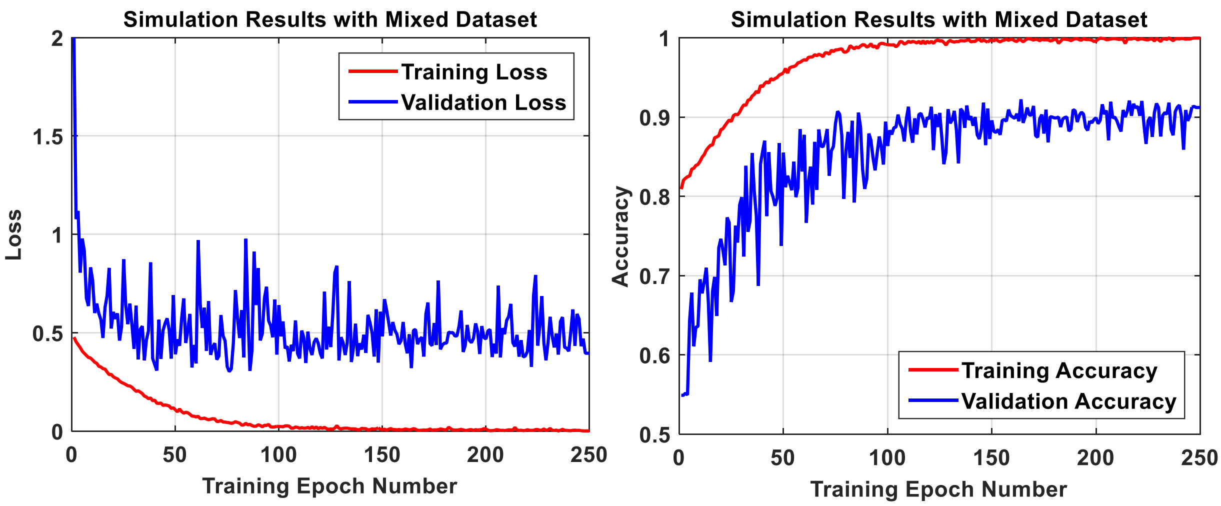

Figure 11. Simulation results of the AI algorithm [21] on the mixed dataset.

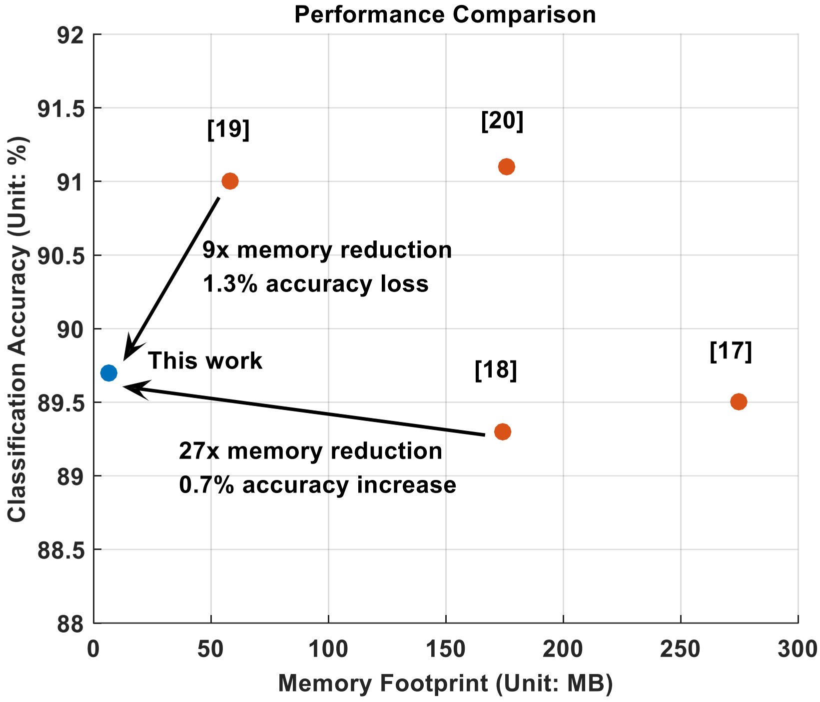

Figure 12. Performance comparison of test accuracy vs. memory footprint between this work and the existing state-of-the-art works in the literature.

Table 5. Comparison with existing contactless camera-based AI algorithms for baby sleep posture detection on the mixed dataset.

Dataset		Weight Quantization	Memory Footprint	Test Accuracy	Comments
Mixed dataset (10,240 images)	[17]	No	275 MB	89.5%	Compared with these existing AI algorithms, this work reduces memory footprint by at least 89%, while maintaining similar classification accuracy.
	[18]	No	174 MB	89.3%	
	[19]	No	58.2 MB	91.0%	
	[21]	No	175.7 MB	91.1%	
	This work	No	51.3 MB	89.9%	
		Yes	6.4 MB	89.7%	

The confusion matrix of each AI algorithm is also plotted in Table 6. The probability for true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN) are listed in this table. We can see that our algorithm leads to a much lower false-negative rate (i.e., 11%), while the false-negative rate of other algorithms is at least 13%. Note the FN error is a test result that incorrectly indicates baby sleep hazard does not hold, but in fact the baby's sleep posture is not safe. That means no threat is observed even though a threat exists. In baby sleep monitoring applications, it is desirable to have a lower false-negative rate, because a lower false-negative rate allows parents to trust the detection performance of our AI algorithm more, and thus helps parents reduce fear, worry, and anxiety.

Table 6. Confusion matrix comparison with existing contactless camera-based AI algorithms for baby sleep posture detection on the mixed dataset.

[17]	Negative (predicted)	Positive (predicted)
Negative (actual)	TN = 0.93	FP = 0.07
Positive (actual)	FN = 0.13	TP = 0.87

Table 6. *Cont.*

[18]	Negative (predicted)	Positive (predicted)
Negative (actual)	TN = 0.93	FP = 0.07
Positive (actual)	FN = 0.15	TP = 0.85
[19]	Negative (predicted)	Positive (predicted)
Negative (actual)	TN = 0.95	FP = 0.05
Positive (actual)	FN = 0.14	TP = 0.86
[21]	Negative (predicted)	Positive (predicted)
Negative (actual)	TN = 0.94	FP = 0.06
Positive (actual)	FN = 0.14	TP = 0.86
This work	Negative (predicted)	Positive (predicted)
Negative (actual)	TN = 0.92	FP = 0.08
Positive (actual)	FN = 0.11	TP = 0.89

To our best knowledge, this work is the first to construct a night-vision baby sleep dataset and use the daytime and night-vision hybrid dataset to train AI algorithms. We expect this work will promote parent-child interaction. Thanks to the smaller memory requirement and higher detection accuracy, the proposed AI algorithm can be easily integrated into a baby monitor, which usually supports two-way audio transmission. Since the proposed design automatically monitors baby sleep posture and sends warnings or captured baby images to the parents' mobile phones, parents do not need to stand next to their baby, especially at night, to understand their baby's sleep status. As a result, parents can relieve stress and even depression to a large extent, thereby improving their sleep quality and mood.

4. Conclusions

In order to deal with Sudden Infant Death Syndrome (SIDS), it is desirable to develop and optimize AI algorithms for contactless camera-based infant sleep posture detection. In this work, we generate a large and diverse dataset for AI training and evaluation. This dataset contains 10,240 day and night-vision baby sleep images. In addition, we propose a CNN AI algorithm and use the post-training weight quantization technique to minimize memory usage. In this way, the data type of weight parameters in our AI algorithm is converted from 32-bit floating points to 8-bit integers. Experiments demonstrate that the proposed AI algorithm achieves high classification accuracy with a small memory footprint. Compared with the existing state-of-the-art works in the literature, our proposed memory-efficient AI algorithm supports comparable test accuracy of around 90% and consumes only 6.4 MB memory, which means at least a 9-fold memory reduction compared to other existing AI algorithms.

Although post-weight quantization can significantly reduce memory footprint, some information is lost when floating-point weights are converted into integer weights during the quantization process. In order to reduce this negative impact, in future work, we plan to integrate quantization-aware training into our proposed AI algorithm. Quantization-aware training is to simulate the quantization behavior and save integer parameters in training, and use quantized weights for output inference. Therefore, generally speaking, quantization-aware training is prone to higher detection accuracy than post-weight quantization in this work. Furthermore, since our proposed AI algorithm does not require substantial memory space or computing capacity, in future work, we plan to implement and run our developed AI algorithm in edge computing devices, such as micro-controllers or Raspberry Pi.

Author Contributions: Conceptualization, Q.H. and C.L.; methodology, Q.H. and C.L.; software, C.H. and J.H.; validation, Q.H.; formal analysis, Q.H.; investigation, Q.H.; resources, Q.H.; data curation, Q.H.; writing—original draft preparation, Q.H.; writing—review and editing, Q.H.; visualization, Q.H.; supervision, Q.H.; project administration, Q.H.; funding acquisition, Q.H. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Acknowledgments: The authors would like to thank Shadman Rashik (the author of [18])'s advisor at Southern Illinois University Carbondale for providing his algorithm for repeating experimental results on the newly developed datasets.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Huang, Q. Review: Energy-Efficient Smart Buildings Driven by Emerging Sensing, Communication, and Machine Learning Technologies. *Eng. Lett.* **2018**, *26*, 320–332.
2. Huang, Q.; Rodrigeuz, K.; Whestone, N.; Habel, S. Rapid Internet of Things (IoT) Prototype for Accurate People Counting Towards Energy Efficient Buildings. *J. Inf. Technol. Constr.* **2019**, *24*, 1–13. [CrossRef]
3. Huang, Q.; Hao, K. Development of CNN-based visual recognition air conditioner for smart buildings. *J. Inf. Technol. Constr.* **2020**, *25*, 361–373. [CrossRef]
4. Huang, Q.; Lu, C.; Chen, K. Smart Building Applications and Information System Hardware Co-Design. In *Big Data Analytics for Sensor-Network Collected Intelligence*; Elsevier BV: London, UK, 2017; pp. 225–240.
5. Gilbert, R.; Salanti, G.; Harden, M.; See, S. Infant sleeping position and the sudden infant death syndrome: Systematic review of observational studies and historical review of recommendations from 1940 to 2002. *Int. J. Epidemiol.* **2005**, *34*, 874–887. [CrossRef] [PubMed]
6. Alfleesy, O. Right-Side Sleeping Position Prevents Sudden Infant Death Syndrome a Literature Review. *J. Forensic. Sci. Criminol.* **2016**, *4*, 204. [CrossRef]
7. Zhu, Z.; Liu, T.; Li, G.; Li, T.; Inoue, Y. Wearable Sensor Systems for Infants. *Sensors* **2015**, *15*, 3721–3749. [CrossRef] [PubMed]
8. Bonafide, C.; Localio, A.; Ferro, D.; Orenstein, D.; Lavanchy, C.; Foglia, E. Accuracy of Pulse Oximetry-Based Home Baby Monitors. *J. Am. Med Assoc.* **2018**, *320*, 717–719. [CrossRef]
9. Hasan, M.; Negulescu, I. Wearable Technology for Baby Monitoring: A Review. *J. Text. Eng. Fash. Technol.* **2020**, *6*, 112–120. [CrossRef]
10. Boughorbel, S.; Bruekers, F.; Breebaart, J. Baby-Posture Classification from Pressure-Sensor Data. In Proceedings of the 2010 20th International Conference on Pattern Recognition, Istanbul, Turkey, 23–26 August 2010; pp. 556–559.
11. Kim, Y.M.; Son, Y.; Kim, W.; Jin, B.; Yun, M.H. Classification of Children's Sitting Postures Using Machine Learning Algorithms. *Appl. Sci.* **2018**, *8*, 1280. [CrossRef]
12. Liu, Z.; Wang, X.; Su, M.; Lu, K. A Method to Recognize Sleeping Position Using an CNN Model Based on Human Body Pressure Image. In Proceedings of the 2019 IEEE International Conference on Power, Intelligent Computing and Systems (ICPICS), Shenyang, China, 12–14 July 2019; pp. 219–224. [CrossRef]
13. Malik, A.; Ehsan, Z. Media Review: The Owlet Smart Sock—A "must have" for the baby registry? *J. Clin. Sleep Med.* **2020**, *16*, 839–840. [CrossRef]
14. Moon, R.Y.; Syndrome, T.F.O.S.I.D. SIDS and Other Sleep-Related Infant Deaths: Evidence Base for 2016 Updated Recommendations for a Safe Infant Sleeping Environment. *Pediatrics* **2016**, *138*, e20162940. [CrossRef] [PubMed]
15. Perez-Pozuelo, I.; Zhai, B.; Palotti, J.; Mall, R.; Aupetit, M.; Garcia-Gomez, J.M.; Taheri, S.; Guan, Y.; Fernandez-Luque, L. The future of sleep health: A data-driven revolution in sleep science and medicine. *NPJ Digit. Med.* **2020**, *3*, 42. [CrossRef]
16. Grimm, T.; Martinez, M.; Benz, A.; Stiefelhagen, R. Sleep position classification from a depth camera using Bed Aligned Maps. In Proceedings of the 2016 23rd International Conference on Pattern Recognition (ICPR), Cancun, Mexico, 4–8 December 2016; pp. 319–324. [CrossRef]
17. Huang, Q.; Hao, K. The Development of Artificial Intelligence (AI) Algorithms to Avoid Potential Baby Sleep Hazards in Smart Buildings. *ASCE Constr. Res. Congr.* **2020**, 278–287. [CrossRef]
18. Shadman, R. The Development of Neural Network Architectures for Image Classification to Prevent Sudden Infant Death in Smart Buildings. Master's Thesis, Southern Illinois University Carbondale, Carbondale, IL, USA, 2021.
19. Khan, T. An Intelligent Baby Monitor with Automatic Sleeping Posture Detection and Notification. *Artif. Intell. (AI)* **2021**, *2*, 290–306. [CrossRef]
20. Huang, G.; Liu, Z.; Maaten, L.; Weinberger, K. Densely Connected Convolutional Networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 4700–4708.

21. Tang, K.; Kumar, A.; Nadeem, M.; Maaz, I. CNN-Based Smart Sleep Posture Recognition System. *IoT* **2021**, *2*, 119–139. [CrossRef]
22. Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J.; Wojna, Z. Rethinking the inception architecture for computer vision. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 2818–2826.
23. Zhou, Z.; Chen, X.; Li, E.; Zeng, L.; Luo, K.; Zhang, J. Edge Intelligence: Paving the Last Mile of Artificial Intelligence with Edge Computing. *Proc. IEEE* **2019**, *107*, 1738–1762. [CrossRef]
24. Deng, S.; Zhao, H.; Fang, W.; Yin, J.; Dustdar, S.; Zomaya, A.Y. Edge Intelligence: The Confluence of Edge Computing and Artificial Intelligence. *IEEE Internet Things J.* **2020**, *7*, 7457–7469. [CrossRef]
25. Nasr, M.; Islam, M.; Shehata, S.; Karray, F.; Quintana, Y. Smart Healthcare in the Age of AI: Recent Advances, Challenges, and Future Prospects. *IEEE Access* **2021**, *9*, 145248–145270. [CrossRef]
26. Althnian, A.; AlSaeed, D.; Al-Baity, H.; Samha, A.; Bin Dris, A.; AlZakari, N.; Elwafa, A.A.; Kurdi, H. Impact of Dataset Size on Classification Performance: An Empirical Evaluation in the Medical Domain. *Appl. Sci.* **2021**, *11*, 796. [CrossRef]
27. Gong, Z.; Zhong, P.; Hu, W. Diversity in Machine Learning. *IEEE Access* **2019**, *7*, 64323–64350. [CrossRef]
28. Fang, W.; Ding, L.; Zhong, B.; Love, P.E.; Luo, H. Automated detection of workers and heavy equipment on construction sites: A convolutional neural network approach. *Adv. Eng. Inform.* **2018**, *37*, 139–149. [CrossRef]
29. Wu, J.; Cai, N.; Chen, W.; Wang, H.; Wang, G. Automatic detection of hardhats worn by construction personnel: A deep learning approach and benchmark dataset. *Autom. Constr.* **2019**, *106*, 102894. [CrossRef]
30. Airaksinen, M.; Räsänen, O.; Ilén, E.; Häyrinen, T.; Kivi, A.; Marchi, V.; Gallen, A.; Blom, S.; Varhe, A.; Kaartinen, N.; et al. Automatic Posture and Movement Tracking of Infants with Wearable Movement Sensors. *Sci. Rep.* **2020**, *10*, 1–13. [CrossRef] [PubMed]
31. Bjorck, J.; Gomes, C.; Selman, B.; Weinberger, K. Understanding Batch Normalization. In Proceedings of the International Conference on Neural Information Processing Systems, Montreal, QC, Canada, 3–8 December 2018; pp. 7705–7716.
32. Yang, G.; Pennington, J.; Rao, V.; Dickstein, J.; Schoenholz, S. A Mean Field Theory of Batch Normalization. In Proceedings of the International Conference on Learning Representations, New Orleans, LA, USA, 6–9 May 2019; pp. 1–15.
33. Ioffe, S.; Szegedy, C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In Proceedings of the International Conference on Machine Learning, Lille, France, 6–11 July 2015; pp. 448–456.
34. Banner, R.; Hubara, I.; Hoffer, E.; Soudry, D. Scalable Methods for 8-bit Training of Neural Networks. In Proceedings of the International Conference on Neural Information Processing Systems, Montreal, QC, Canada, 2018, 3–8 December; pp. 5151–5159.
35. Wu, S.; Li, G.; Chen, F.; Shi, L. Training and Inference with Integers in Deep Neural Networks. In Proceedings of the International Conference on Learning Representations, Vancouver, BC, Canada, 30 April–3 May 2018; pp. 1–14.
36. Zhu, X.; Vondrick, C.; Fowlkes, C.C.; Ramanan, D. Do We Need More Training Data? *Int. J. Comput. Vis.* **2016**, *119*, 76–92. [CrossRef]
37. Zheng, J.; Lu, C.; Hao, C.; Chen, D.; Guo, D. Improving the Generalization Ability of Deep Neural Networks for Cross-Domain Visual Recognition. *IEEE Trans. Cogn. Dev. Syst.* **2021**, *13*, 607–620. [CrossRef]
38. Abadi, M.; Barham, P.; Chen, J.; Chen, Z.; Davis, A.; Dean, J.; Devin, M.; Ghemawat, S.; Irving, G.; Isard, M.; et al. TensorFlow: A System for Large-Scale Machine Learning. In Proceedings of the ACM USENIX Conference on Operating Systems Design and Implementation, Savannah, GA, USA, 2–4 November 2016; pp. 265–283.
39. Baldi, P. Gradient descent learning algorithm overview: A general dynamical systems perspective. *IEEE Trans. Neural Netw.* **1995**, *6*, 182–195. [CrossRef]
40. LeCun, Y.; Bottou, L.; Bengio, Y.; Haffner, P. Gradient-based learning applied to document recognition. *Proc. IEEE* **1998**, *86*, 2278–2324. [CrossRef]
41. Senior, A.; Heigold, G.; Ranzato, M.; Yang, K. An empirical study of learning rates in deep neural networks for speech recognition. In Proceedings of the 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, Vancouver, BC, Canada, 26–31 May 2013; pp. 6724–6728. [CrossRef]