

Real-time Fall Detection System for the Elderly Using Thermal Imaging and Deep Learning

Vetri Vel
Bangor High School
Bangor, ME, 04401

Abstract

Falls are the leading cause of fatal injury among older adults with over 12 million falls in the United States annually. For elderly people living alone, quick assistance after a fall is critical and can reduce hospitalization rate by 26% and death rate by 80%. While there are wearable medical alert systems, many elderly individuals prefer not to wear them or may be unable to call for help if injured after a fall. The research goal of this study was to develop an unobtrusive embedded system for real-time fall detection. A Raspberry Pi and a Forward-Looking Infrared (FLIR) thermal camera were utilized. Thermal imaging preserves privacy while detecting heat signatures of humans. Deep learning was used to analyze and classify images due to its effectiveness for complex pattern recognition. The system developed uses a set of thermal images to train a convolutional neural network to identify images of fallen individuals. The effects of factors including image resolution and number of training images on the accuracy of the created neural networks were investigated. An average accuracy of 99.2% (SD 0.62%, n=10) was achieved even in the presence of non-human heat sources such as pets. A functioning wall-mounted system has been developed to detect falls and immediately call for help, potentially increasing the safety and independence of elderly people living alone at home or in assisted living facilities. The embedded system can also be used for other applications such as monitoring the night-time activity and sleep patterns of autistic children, enhancing their safety.

Statement of Ethical Research

Research involving human subjects and vertebrate animals was conducted under the direct supervision of a qualified mentor with IRB approval which complies with local, state, and federal regulations for such research. Thermal image collection was non-intrusive and harmless.

Key Terms and Acronyms

FLIR – Forward-Looking Infrared

SPI – Serial Peripheral Interface

GPIO Pins – General Purpose Input/Output signal pins

CNN – Convolutional Neural Network

API – Application Programming Interface

Raspberry Pi – Single-board microcomputer

Deep Learning – Form of machine learning using neural networks

Table of Contents

1. INTRODUCTION	3
1.1 RESEARCH PROBLEM	3
1.1.1 <i>Falls Among the Elderly</i>	3
1.1.2 <i>Risk Factors for Falling</i>	3
1.1.3 <i>Costs of Falls</i>	3
1.2 FALL DETECTION SYSTEMS	3
1.2.1 <i>Wearable Devices</i>	4
1.2.2 <i>Stationary Monitoring Systems using Smart Tiles</i>	4
1.2.3 <i>Stationary Monitoring Systems using Radar</i>	4
1.2.4 <i>Stationary Monitoring Systems using Thermal Imaging</i>	5
1.3 MOTIVATION AND ENGINEERING GOAL	5
1.4 CRITERIA FOR PROPOSED SYSTEM	5
1.5 PROPOSED SYSTEM	6
2. MATERIALS	6
2.1 HARDWARE	6
2.2 SOFTWARE	7
3. METHODS	7
3.1 CONSTRUCTION OF EMBEDDED SYSTEM	7
3.2 ACQUISITION OF TRAINING AND TESTING DATA	8
3.2.1 <i>Reducing Image Resolution</i>	9
3.3 DEEP LEARNING	9
3.3.1 <i>Dense Neural Networks</i>	10
3.3.2 <i>Convolutional Neural Networks (CNNs)</i>	11
3.3.3 <i>Optimization of Neural Networks</i>	12
3.3.4 <i>Training Methods</i>	12
3.4 IMPLEMENTATION OF FUNCTIONING EMBEDDED SYSTEM	13
3.4.1 <i>Sending Alerts When a Fall is Detected</i>	14
4. RESULTS AND DISCUSSION	14
4.1 INFLUENCE OF NEURAL NETWORK ARCHITECTURE ON ACCURACY	14
4.1.1 <i>Dense Neural Network</i>	14
4.1.2 <i>Convolutional Neural Networks</i>	15
4.2 INFLUENCE OF A SLEEPING REGION ON ACCURACY	17
4.3 INFLUENCE OF NON-HUMAN HEAT SOURCES ON ACCURACY	17
4.4 RELATIONSHIP BETWEEN NUMBER OF TRAINING IMAGES AND ACCURACY	17
4.5 RELATIONSHIP BETWEEN IMAGE RESOLUTION AND ACCURACY	18
5. CONCLUSIONS	18
6. APPLICATIONS AND FUTURE WORK	19
ACKNOWLEDGEMENTS	19
REFERENCES	20

1. INTRODUCTION

1.1 Research Problem

1.1.1 Falls Among the Elderly

Annually, over a quarter of older individuals (above the age of 65) fall in the United States and 3 million older people are treated in emergency departments for fall injuries each year [1]. Every 15 seconds an elderly person is treated for a fall in an emergency department, and an elderly person dies from a fall every 29 minutes [2]. Additionally, falling once doubles a person's chance of falling again [1]. Deaths due to falls increased 30% from 2007 to 2016, as shown in Figure 1. If death rates continue to rise, 7 fall deaths will occur every hour by the year 2030 [1].

More than 95% of hip fractures are caused by falling and falls are the most common cause of traumatic brain injuries [1]. Over 20% of patients admitted to the hospital because of a fall had remained fallen for at least an hour, and even if there was no injury from the fall itself, their morbidity rates within 6 months were high [3]. However, getting quick assistance after a fall reduces the hospitalization rate by 26% and death rate by 80% [4].

If an elderly person living alone falls and nobody else is aware, they could remain fallen for hours, increasing their risk of injury and death.

1.1.2 Risk Factors for Falling

There are many factors that can increase a person's chance of falling, including difficulty with walking and balance, a weak lower body, use of medicines, especially those for sleeping or managing depression, and vision problems. Most falls are caused by a combination of risk factors. Having more risk factors correlates with a higher chance of falling [1].

1.1.3 Costs of Falls

Treating fall injuries is very costly. In 2015, the total medical costs for falls in the United States were more than \$50 billion. Out of the \$50 billion, \$12 billion was out of pocket costs for patients and the rest was covered by Medicare and Medicaid [1]. The average cost of a hospitalizing fall injury is over \$30,000 and treating fall injuries is more expensive for older individuals [1].

1.2 Fall Detection Systems

Falls can be detected by devices worn by the individual or by systems that monitor the living space for signs of them having fallen.

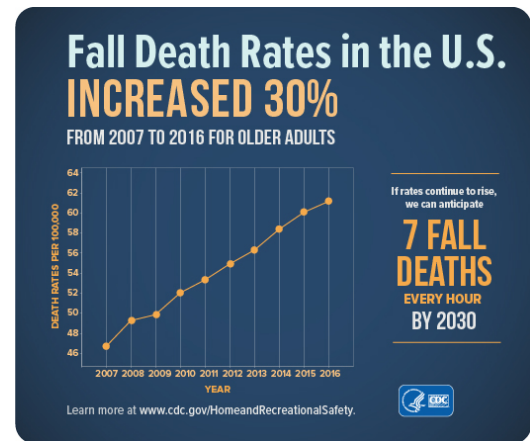


Figure 1. Graph by the CDC showing fall death rates per 100,000 people from 2007 to 2016 [1].

1.2.1 Wearable Devices

Wearable devices detect falls by sensing for movement patterns that likely correspond to falling. At their core, they are miniature sensor-based devices that are worn by the individual with or on top of clothing [3]. The majority of wearable fall detectors rely on the user pressing a button after falling or use accelerometers while some also use gyroscopes to get information about the wearer's position [3].

Acceleration data is collected during falls using a tri-axial accelerometer within the wearable device. For acceleration-based fall detection, the two ways to determine whether a fall has occurred are threshold based methods and machine learning methods [3].

As the name suggests, in threshold based methods falls are reported when features such as peaks on the graph of acceleration vs. time reach set thresholds [3]. This can be a problem if the accelerometer is placed on the individual's wrist, because it could misclassify running and other activities as falls [3]. Machine learning methods help with this problem because the algorithm that detects falls can be trained beyond just finding high points in the acceleration graph [3]. These algorithms are better at discerning between the acceleration patterns generated when running and the acceleration patterns generated when falling [3].

The main disadvantage of wearable devices is that elderly individuals may find them uncomfortable to wear constantly, they may forget to wear them, or they may not be able to press a button to call for help after a fall [3].

1.2.2 Stationary Monitoring Systems using Smart Tiles

Stationary systems monitor the environment in which the individual lives for signs of them having fallen. Examples include using Smart Tiles (floor tiles with built in accelerometers and force sensors) to track the movements of people and detect activities such as walking, standing, sitting, and most importantly falling [4]. Each smart tile has an accelerometer in the center and four force sensors at each corner. Also, it takes a total of 104 tiles to cover the floor of a small apartment [4].

The Smart Tile method has drawbacks that prevent it from being used commercially, such as how expensive and involved it would be to purchase and install hundreds of Smart Tiles, each with intricate sensors, and the complexity of transmitting data from each tile to a centralized processing unit that can detect falls.

1.2.3 Stationary Monitoring Systems using Radar

There are other stationary systems for fall detection such as the commercially available Walabot HOME [5]. The Walabot HOME is a wall-mounted device that continuously monitors a room using radar technology [5]. It uses machine learning for fall detection and is apparently four times more accurate than other automatic fall alert systems [5]. The Walabot HOME automatically contacts up to four emergency contacts in the case of a fall [5].

Unfortunately, the research that was used to develop the Walabot HOME is not publicly available, and the part designs and programs are not open source. Additionally, the available packages of one,

two, or three Walabot HOME units cost \$99, \$94, and \$89 per unit, and have a monthly cost of \$9.99 for each of the packages.

1.2.4 Stationary Monitoring Systems using Thermal Imaging

Preliminary work has been done by researchers on using thermal imaging for posture detection. Thermal imaging works by sensing the increased levels of infrared light emitted by objects at higher temperatures. Thermal imaging can detect heat from humans during the night, in dark conditions, when the probability of falling is higher [6]. The cost of thermal cameras is decreasing, which will make them even more viable in the future [6]. Another advantage of thermal imaging is that it protects privacy. An individual can't be identified by facial features in a low resolution thermal image [5,6]. This makes it implementable in bathrooms, where 80% of falls occur among the elderly [1].

Vadivelu et al. [6] used a thermal video stream and a preexisting video dataset for elderly fall detection. They used an optical flow based technique and a Support Vector Machine model to discriminate fall from non-fall scenarios. Adolf et al. [7] used an 8×8 pixel thermal camera that was placed on the ceiling and pointed directly down to monitor a 3×3 m area on the floor. Their system used a neural network architecture from the field of computer vision to differentiate between simple cases such as nothing in the frame, an object, a person standing, a person sitting, and a person lying on the ground. They achieved a classification accuracy of about 89% [7]. They state that while their system works reasonably well, it is not reliable enough to be used for monitoring the elderly. One of the drawbacks of their system is that requires a PC connected to a thermal camera to perform all the computations. In addition, it can only monitor a small region directly under the camera. Currently, there does not appear to be an unobtrusive, low-cost embedded system that can be used for fall detection with high accuracy.

1.3 Motivation and Engineering Goal

The world's older adult population is growing at an unprecedented rate. In 2016, 8.5% (617 million) of the world population was over 65 years old, but the proportion is projected to increase to 17% of the world population by 2050, or 1.6 billion people. The 65-and-over population in the US is projected to nearly double from 48 million to 88 million by 2050 [8]. With such a rapidly growing elderly population, it is essential to make sure there is technology to help take care of them. With this in mind the research done and presented in this paper was undertaken.

The engineering goal was to create an unobtrusive system for the real-time fall detection of elderly individuals. The envisioned product is a wall mounted system positioned like a security camera that monitors a space for signs of a fall and automatically calls for help.

1.4 Criteria for Proposed System

The engineering constraints and objectives for the proposed system are listed in Table 1. The system should be low cost, maintain the privacy of individuals, be capable of detecting falls at night, and automatically call for help. Objectives include the system being an embedded system and having a high accuracy of fall detection while being easy to use and robust, meaning it can work in

a variety of environments.

Constraints	Objectives
1. Maintain privacy	1. Embedded system
2. Work in the absence of light	2. Fall detection with high accuracy
3. Can call for help	3. Easy to use and robust
4. Low cost	

Table 1. Criteria for proposed solution.

1.5 Proposed System

A thermal camera was used in conjunction with a Raspberry Pi to create an embedded system that uses deep learning to analyze images, monitor individuals, and call for help if they fall. Table 2 lists each component used and the objectives and constraints from Table 1 that it satisfies or allows for. Using a thermal camera helps to preserve privacy because humans are not identifiable in thermal images, and it allows the system to work in the absence of light. A Raspberry Pi, shown in Figure 3, was used to control the embedded system and allows the system to be low cost. Since it can connect to the internet, it meets the constraint of being able to call for help. Deep learning was used because it is form of machine learning and allows for high accuracies of pattern recognition. All programs were written in Python since it is an open-source programming language, and many libraries have been developed in Python that allow the Raspberry Pi to interface with thermal cameras and run code for deep learning; this helps meet the objective of being easy to use.

Component	Objectives and Constraints Satisfied
Thermal camera	Constraints 1 and 2 (privacy and works in the absence of light)
Raspberry Pi	Constraints 3 and 4 (low cost and can call for help) and Objective 1 (embedded system)
Deep learning	Objective 2 (high accuracy)
Python code	Objective 3 (easy to use)

Table 2. Components of the proposed system and the objectives/constraints they satisfy

2. MATERIALS

2.1 Hardware

A FLIR Lepton long wave infrared micro thermal camera module, shown in Figure 2, was used to capture thermal images. The images have resolution 80×60, and the lens has a 50° horizontal field of view. The camera consumes only 150 mW when operating (capturing thermal images) and 4 mW when on standby. It is connected to a breakout board that has pins to connect to microcomputers such as the Raspberry Pi [9]. The FLIR Lepton uses Serial Peripheral Interface (SPI)

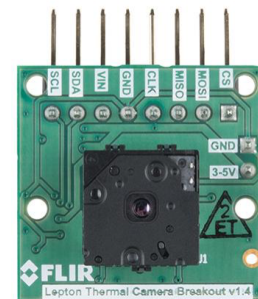


Figure 2. FLIR Lepton [9]

communication protocol to transmit the thermal images in real time to a microcomputer and an I²C communication protocol for the control interface.

The Raspberry Pi 4, shown in Figure 3, was used in this research. It is a single-board microcomputer with a Quad core Cortex-A72 64-bit processor at 1.5 GHz and 4 GB of RAM. It costs \$55 with 4 GB of RAM, but only \$35 with 1GB of RAM [10].



Figure 3. Raspberry Pi 4 [10]

2.2 Software

All the programs, such as ones to capture thermal images or implement deep learning were written in Python 3. TensorFlow is an advanced platform for deep learning, and can be run on Python, so it was used to implement deep learning and train neural networks. While TensorFlow is versatile and well developed, it is involved and complex to use, so a Python library called Keras was used to interface with TensorFlow [11].

An Application Programming Interface (API) called Twilio was used to send alerts from the Raspberry Pi in the form of text messages. Twilio is a cloud communications platform which allow programs to send and receive phone calls and text messages [12]. The Twilio library was used in a Python script to call for help when a fall was detected.

3. METHODS

3.1 Construction of Embedded System

The embedded system consists of a Raspberry Pi 4 connected to the FLIR Lepton thermal camera and an RGB LED. Initially, the FLIR Lepton was wired on breadboard using a T-cobbler to connect the Raspberry Pi to the breadboard, as shown in Figure 4(a). In the next stage of development, the pins on the Lepton breakout board were wired directly to corresponding GPIO pins on the raspberry Pi as shown in Figure 4(b).

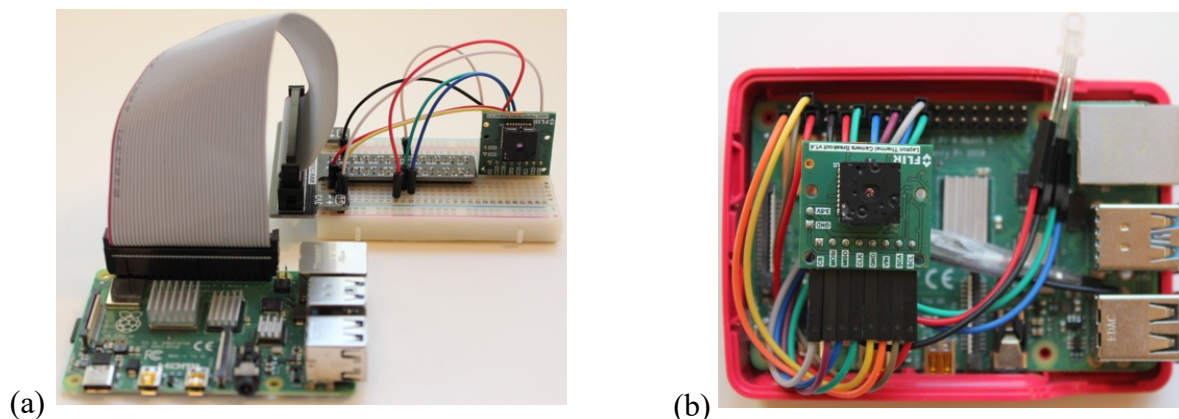


Figure 4. Pictures showing the development of the embedded system in which the FLIR Lepton is (a) wired with a T-cobbler, (b) directly connected to the Raspberry Pi GPIO pins along with an LED

The FLIR Lepton breakout board has 8 pins that connect to corresponding pins on the Raspberry Pi.

The pins are referred to as VIN, GND, SDA, SCL, MOSI, MISO, CLK, and CS. The VIN pin powers the module, and the GND pin grounds it, allowing current to flow. The SDA and SCL pins are used for the I²C communication protocol used to control the thermal camera from the Raspberry Pi. The CLK (clock signal), MOSI (Master Output Slave Input), MISO (Master Input Slave Output), and CS (chip select) pins are used for reading thermal images using SPI.

The Raspberry Pi communicated with the FLIR Lepton using Python libraries to capture images. Once the Lepton had been connected to the Raspberry Pi, this library allows for programs to take images via the thermal camera and store the pixel values in matrices.

The completed system, shown in Figure 5, is mounted high up on a wall, and the wiring and Raspberry Pi are enclosed in the case. The FLIR Lepton is on the outside of the case and angled slightly downwards to achieve an optimal field of view. The only cord needed in the final embedded system is the power supply.



Figure 5. Final wall-mounted embedded system

3.2 Acquisition of Training and Testing Data

In order to train the neural network and then test its accuracy, thermal images were collected of participant 1 in varying postures and positions in the field of view. Approximately $\frac{1}{4}$ of images were used for testing and the others were for training. In the case of normal thermal images containing two people, images were taken of participants 1 and 2, each in normal postures such as standing or sitting. Thermal images were collected of a dog, cat, and hot object for the study on how non-human heat sources affect the accuracy of fall detection.

A set of testing images were picked manually so that they covered different positions in the field of view and were representative of each case in Table 3. The testing images were not used when training the network but rather to test for pattern recognition and the accuracy of image classification.

Case	Classification	Number for Training	Number for Testing
Empty Room	Normal	16	6
1 Person Standing	Normal	58	20
1 Person Sitting	Normal	23	7
1 Person Sleeping	Normal	13	5
2 People	Normal	31	10
Dog	Normal	59	19
Cat	Normal	35	12
Hot Objects	Normal	46	15
1 Person Fallen	Abnormal/Fallen	52	18
Total Images	-	287	97

Table 3. Case, classification, and the number of images used for training and testing.

Representative thermal image for select cases such as standing, sitting, and sleeping are shown in Figure 6. Image 6(c) is classified as sleeping (normal) because a region on the floor was marked to be a bed, and images where a participant was lying in that region were classified as normal; this was to later train the system to not detect a fall when someone is lying in a sleeping region. Image 6(d) contains an example of a non-human heat source, namely a dog. Image 6(f) shows someone lying on the floor and not in a sleeping region as if the person had fallen. No identifying features are discernable in the thermal images, and the similar heat signatures of the two people in image 6(e) shows that thermal imaging preserves the privacy and identity of individuals.

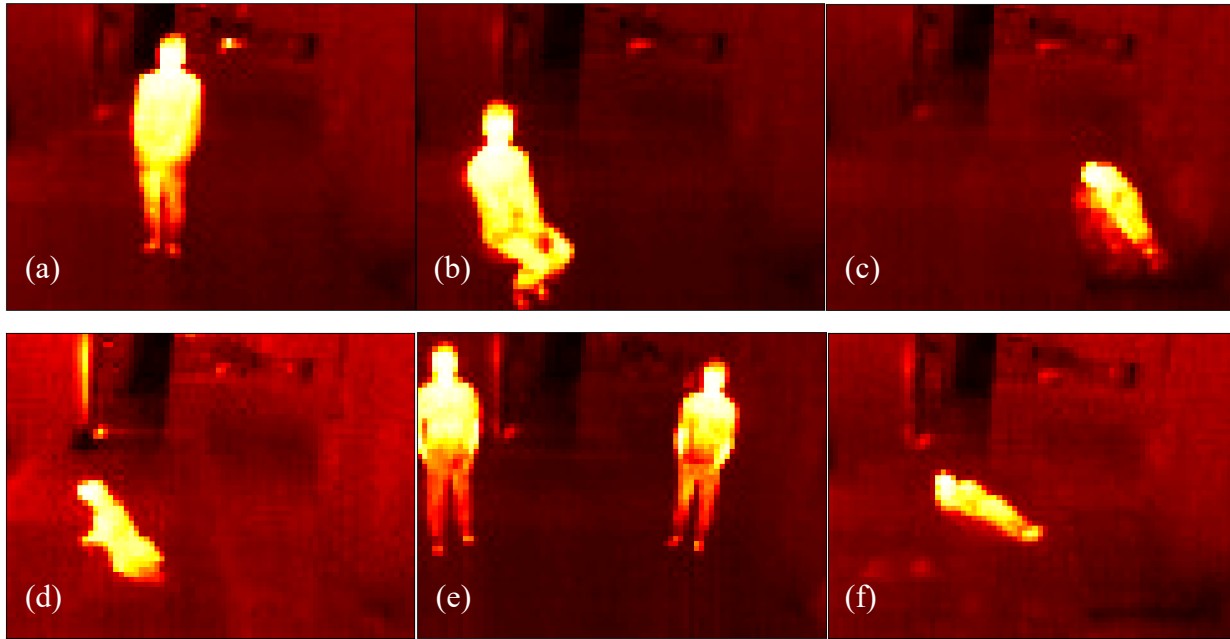


Figure 6. Representative thermal images for different cases, such as (a) standing, (b) sitting, (c) sleeping, (d) non-human heat sources (dog), (e) two people standing, and (f) fallen.

3.2.1 Reducing Image Resolution

The FLIR Lepton captures images with 80×60 pixel resolution. To test the effect reducing image resolution on the accuracy of the neural network, the resolution of images in the training and testing image sets were artificially reduced from 80×60 pixels while maintaining the aspect ratio. The images were converted to resolutions such as 40×30 , 20×15 , and 16×12 using bicubic interpolation to simulate them having been taken by a thermal camera of lower resolution.

By testing the accuracy of the system when trained and tested on lower resolution images, it was possible to see the minimum resolution needed to reach a sufficiently high accuracy. Once the minimum resolution was determined, thermal cameras with resolutions higher than the minimum resolution may be used instead of the FLIR Lepton. This would lower the cost of thermal camera, and therefore the total cost of the embedded system.

3.3 Deep Learning

Deep learning is a more recent machine learning technique. Deep learning is the process of training

neural networks to perform tasks such as handwriting recognition that machines would normally not be able to do [11]. Details of the neural networks used in this project and their structures are explained in sections 3.3.1 and 3.3.2. Sections 3.3.3 and 3.3.4 explain how the neural networks are trained/optimized, then implemented in real-time fall detection system.

3.3.1 Dense Neural Networks

Figure 7 shows a diagram of a dense neural network as applied in this research. The image shown in the figure is an actual thermal image captured using the FLIR Lepton. The gridlines on the input image show that the image is a two-dimensional array of greyscale pixel values. First, the input thermal images is “flattened,” meaning each greyscale pixel value is input into a node in the input layer. Each node in a neural network holds only one scalar value. Each node is connected to the nodes in the previous layer.

In addition to an input layer, a deep neural network consists of multiple hidden layers and an output layer that are connected in a sequential manner as shown in Figure 7. In a dense neural network, values of the nodes in each layer are the weighted sums of the nodal values in the previous layer plus a bias (a scalar value). For example, the values of the nodes in the first hidden layer depend on the nodal values in the input layer, as described in the following equation,

$$y_i = a \left(\sum_{j=1}^n w_{ij} x_j + b_i \right),$$

where the index i varies from 1 to m and w_{ij} and b_i are weights and biases. Here, y_i are the nodal values of the first hidden layer and x_j are the values of the input layer. $a(x)$ is the activation function, which modifies the weighted sum to “squish” the result between 0 and 1 (softmax activation function) or map negative values to 0 while leaving positive values unchanged (ReLU activation function). The final layer of the neural network, called the output layer, gives predictions/scores between 0 and 1 that correspond to classifications of normal or fallen.

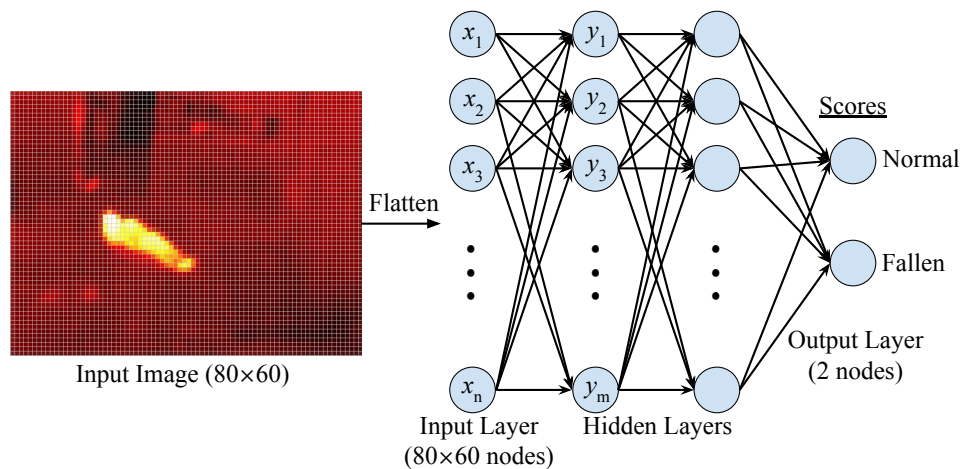


Figure 7. Dense neural network.

All the weights and biases in the neural network are variables that can be adjusted. Adjusting the weights and biases affects the nodal values in each layer including the output layer given a fixed

input. Deep learning as used in this research is the process of optimizing these weights and biases to correctly classify images.

3.3.2 Convolutional Neural Networks (CNNs)

CNNs first apply convolutional filters to an image before putting them through a dense neural network. Convolutional filters can be used to detect features in images such as horizontal and vertical edges. Figure 8 shows the resulting matrices, called feature maps, when specialized filters are applied across the input thermal input image of a standing person. When the vertical edge filter is applied to the input image, the resulting feature map has more pixels at high values than when the horizontal edge filter is applied. This makes sense because thermal images with a standing person have more vertical edges than horizontal edges between the person and the background.

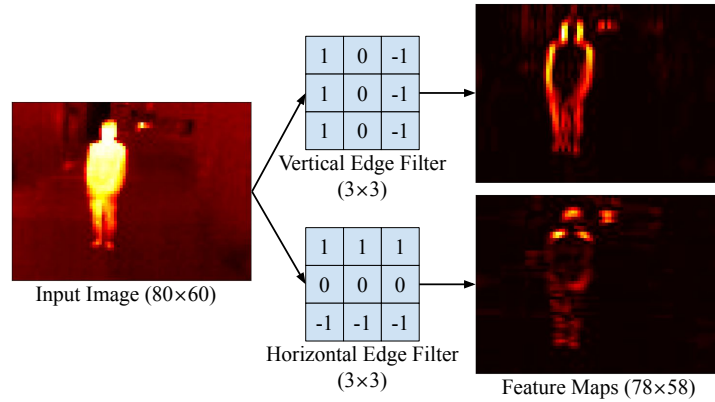


Figure 8. Feature maps generated after specialized filters are applied across an image

Figure 9 shows a model of a CNN. In Figure 9, four 3×3 convolutional filters are applied across the input image to yield four feature maps, but any number of convolutional filters of any resolution could be used. The “flatten” process (denoted by an arrow) takes each pixel value of the four feature maps and assigns a node in the dense layer that value. Since there are four feature maps of resolution 78×58 , the dense layers has to have $4 \times 78 \times 58 = 18096$ nodes. It is possible to use pooling, which is a technique to reduce the size of the feature maps before feeding them into the dense layers. Since the thermal images were only 80×60 , it wasn’t necessary to use pooling to reduce of the size of the feature maps before being fed into the first dense layer. No hidden layers were used, so the dense layer of $4 \times 78 \times 58$ nodes connects directly to the output layer consisting of 2 nodes.

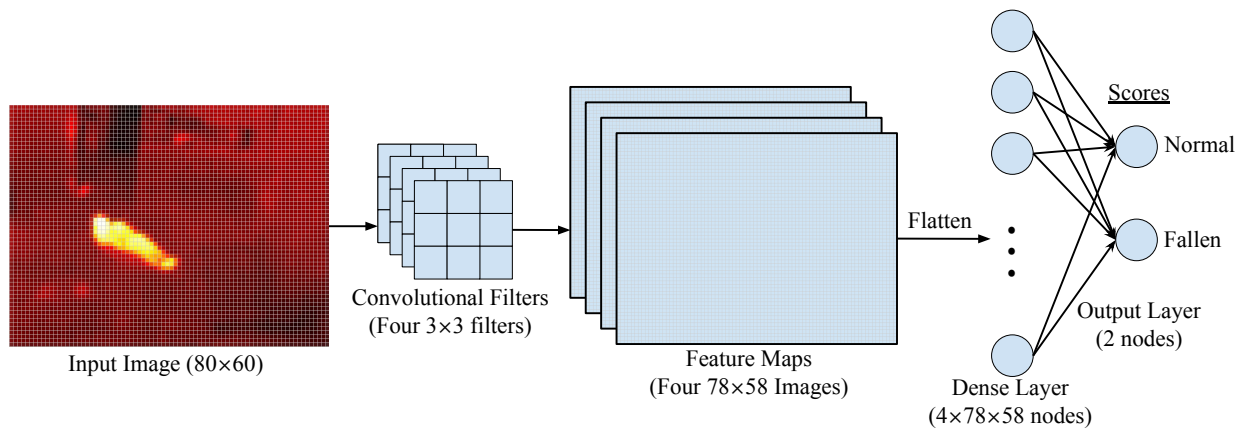


Figure 9. Convolutional Neural Network.

3.3.3 Optimization of Neural Networks

The weights and biases for each layer of a neural network, as well the filter matrices in the case of a CNN, are optimized using gradient descent. Figure 10 shows an example of gradient descent. In Figure 10, there are only two variables and the loss function is the z-value. A loss function measures the amount by which a neural network deviates from the correct classifications of images. In a neural network, there would be thousands of independent variables for the weights and biases that need to be optimized.

Gradient descent is used because it is almost impossible for a computer to “graph” the whole loss function and find the absolute minimum since there are so many independent variables that can be varied. In gradient descent, the gradient vector, or a vector containing the partial derivatives of the loss function with respect to each variable is calculated. The gradient vector points in the direction of steepest ascent, so by multiplying the gradient vector by a negative scalar, the resulting vector points in the direction of steepest descent. The variables are repeatedly incremented in the direction of steepest descent along the loss function, which causes the loss to reach a local minimum [13].

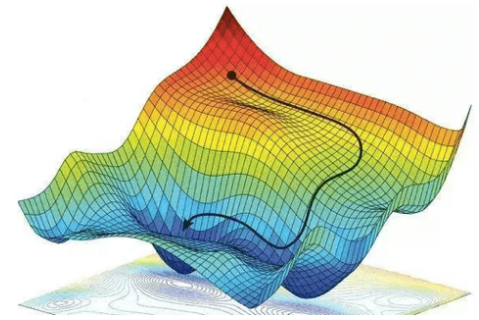


Figure 10. The path followed when using gradient descent [14]

3.3.4 Training Methods

First, a neural network is trained using a set of images that have been manually classified as normal or abnormal. The whole process typically takes a few minutes on a Raspberry Pi. Figure 11 shows the flowchart used in this research for training neural networks using a database of thermal images, then saving the defining information of the neural network to a file so that it can be read by the real-time fall detection program.

An epoch is when the entire training dataset is passed through the neural network once. In each epoch, shown in Figure 11, first the training images are input into the neural network and predictions are output. The predictions are compared with the target outputs for each training image, and the cumulative loss is computed and input into the optimizer, which uses gradient descent to increment the weights and biases. This process is iterated over several epochs to minimize the loss and maximize accuracy. The trained network’s weights and biases are then saved to a file which will be read by the real-time fall detection algorithm.

To implement a bed region where someone lying in that region would still be classified as normal, thermal images of someone lying down in a bed region were added to the training and testing data under the classification of normal. This trained the neural network to not classify images as fallen when the person is lying down in a specific location in the image.

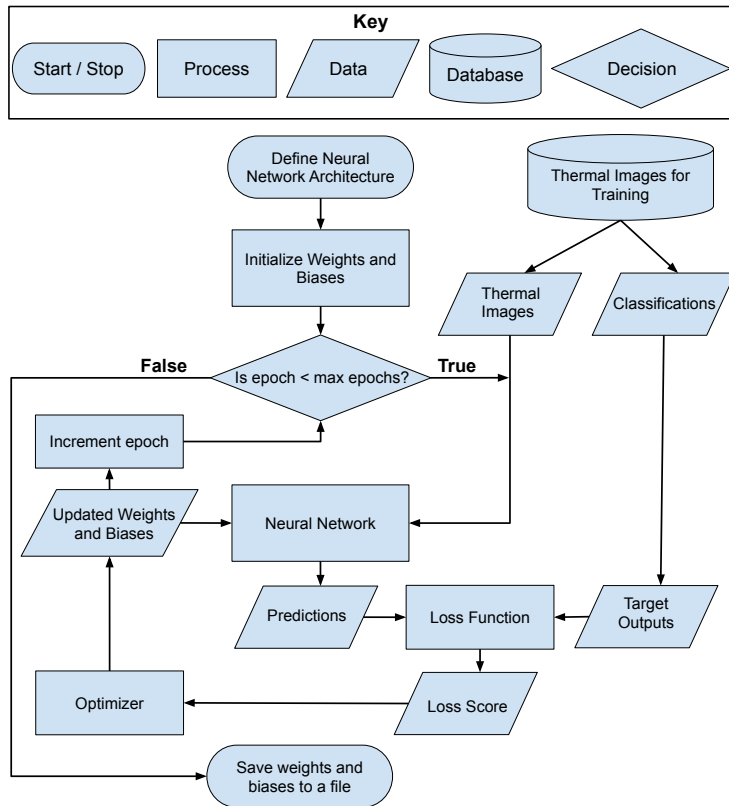


Figure 11. Flowchart for training a neural network with key listing the meaning of each shape.

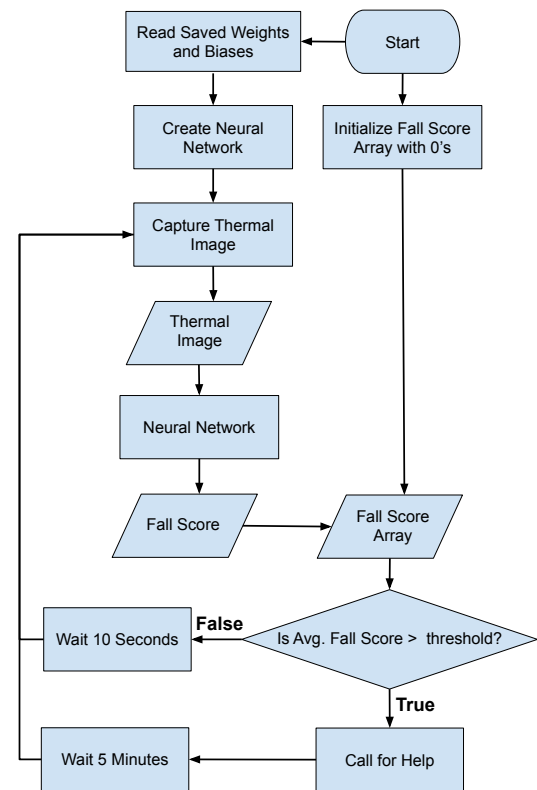


Figure 12. Flowchart for using a trained neural network for real-time fall detection.

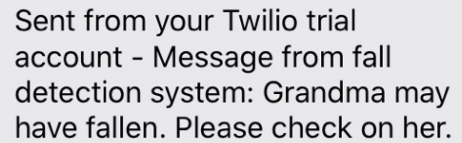
3.4 Implementation of Functioning Embedded System

In the functioning embedded system for fall detection, thermal images are taken periodically and given a fall score by the neural network which has already been trained. A running array of the previous few (such as 10) fall scores are kept, and if the average value of the array exceeds a preset threshold (such as 0.7), the system decides that someone has fallen and calls for help. The system then waits 5 minutes before taking another image, and if it still detects a fall, it would call for help again. Having a running array of fall scores helps ensure that the system doesn't call for help after detecting just one fallen image and compensates for images that may be misclassified by the neural network.

A flowchart of the Python program developed for the real-time fall system is shown in Figure 12. The program uses the previously trained neural network shown in Figure 11 to classify images. Classifying an image using a trained neural network shown takes less than 0.1 seconds on a Raspberry Pi.

3.4.1 Sending Alerts When a Fall is Detected

A script was written in Python to allow the Raspberry Pi to send a customized text message using Twilio when connected to the internet. Figure 13 is a screenshot of such a text message sent to the author's phone when a fall was detected.

A screenshot of a text message in a light blue bubble. The text reads: "Sent from your Twilio trial account - Message from fall detection system: Grandma may have fallen. Please check on her." The background is a light gray gradient.

Sent from your Twilio trial account - Message from fall detection system: Grandma may have fallen. Please check on her.

Figure 13. Example text message sent when a fall is detected

4. RESULTS AND DISCUSSION

Since there were many variables and aspects of the neural networks and training data that could be changed, studies were conducted on the effects of each variable separately. First, neural network architecture was studied to determine if convolutional neural networks achieved higher accuracy than dense neural networks, and the optimal number of convolutional layers and filters to use.

Once the architecture of the neural network was optimized, images where there was a sleeping area (where someone lying down would be classified as normal) were introduced to the training and testing data to test the effect on accuracy. This was to test if the neural network could be trained to not classify an image as fallen when a person is lying in a specific location in the image. Images with non-human heat sources such as pets and hot objects (space heater, stove, etc.) were also introduced into the training and testing data to test the effect on accuracy. This was to test if the neural network would incorrectly classify images because of non-human heat sources.

The effects of the number of training images and the resolution of the images were also studied by holding one fixed, then varying the other. It is important to know what resolution camera is needed for the neural network to perform well, and how many training images are needed to be taken.

4.1 Influence of Neural Network Architecture on Accuracy

In this section, the accuracies of dense neural networks and convolutional neural networks are compared. This study was done first so that it could be determined whether using convolutional layers improved the accuracy of the neural network. The normal images in the training and testing sets were limited to empty room, 1 person standing, and 1 person sitting since this study was to determine the optimal neural network architecture for recognizing key patterns. The effects of adding sleeping images and images with non-human heat sources to the training and testing sets are presented in sections 4.2 and 4.3.

4.1.1 Dense Neural Network

Since the thermal images had 80×60 pixels, the input layer for the neural network had 4800 nodes, one for each pixel value. The output layer always has 2 nodes for a normal and fallen classifications.

Figure 14 is the learning graph for a dense neural network with dense layers of 4800, 80, then 2 nodes. As shown in Figure 14, the accuracy of the dense neural network during the training process stays at a constant low accuracy. The loss decreases, but still remains high. This means that the network is not learning to recognize patterns. Validation accuracy and loss referenced in Figure 14

are the accuracy and loss of the neural network when classifying testing images.

The learning graph is for a neural network with one hidden layer of 80 nodes, but the number of hidden layers and the number of nodes in each layer can be varied. Table 4 shows the performance of dense neural networks with 1, 2, and 3 hidden layers. Only 5 measurements ($n=5$) were taken for each row of Table 4 because the training times were high. The number of nodes in each consequent layer decrease since the input layer has 4800 nodes and the output layer has only 2. Unfortunately, the accuracies of the networks always stayed at 0.647. This indicates that the dense networks just classified every image as normal when trying to minimize the loss (33/51 of the testing images are normal).

Number of Hidden Layers and Nodes in each Layer	Avg. RMS Loss	Avg. Accuracy
1 (80)	0.561 (SD 0.080)	0.647 (SD 0.000)
2 (160, 20)	0.642 (SD 0.004)	0.647 (SD 0.000)
3 (320, 40, 10)	0.652 (SD 0.014)	0.647 (SD 0.000)

Table 4. Average accuracy and RMS losses for dense neural networks with varying numbers of hidden layers and nodes in each.

4.1.2 Convolutional Neural Networks

Figure 15 shows a learning graph for a convolutional neural network with 1 convolutional layer comprised of four 3×3 convolutional filters. The convolutional neural network learns much faster than the dense neural network, and its validation accuracy even reaches a value of 1.0, meaning that it classifies every image in the testing set correctly. This showed that the network had learned to recognize patterns and did not overfit to the training images, which is the desired outcome.

Table 5 lists the average accuracies and loss values of convolutional neural networks with 1, 2, or 3 convolutional layers, each containing 4 3×3 filters. It shows that neural networks with 1 convolutional layer performed the best over 10 trials ($n=10$).

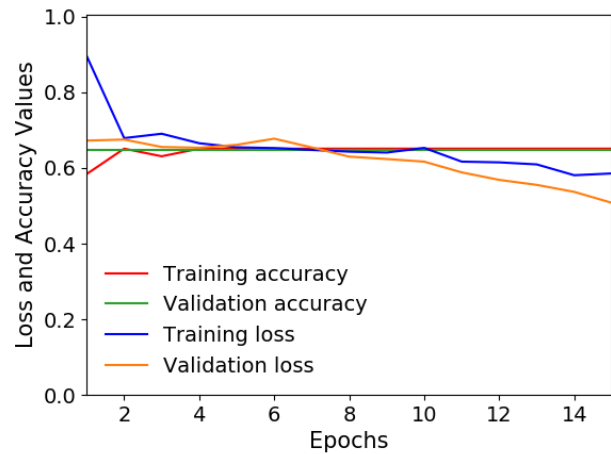


Figure 14. Learning graph for a dense neural network

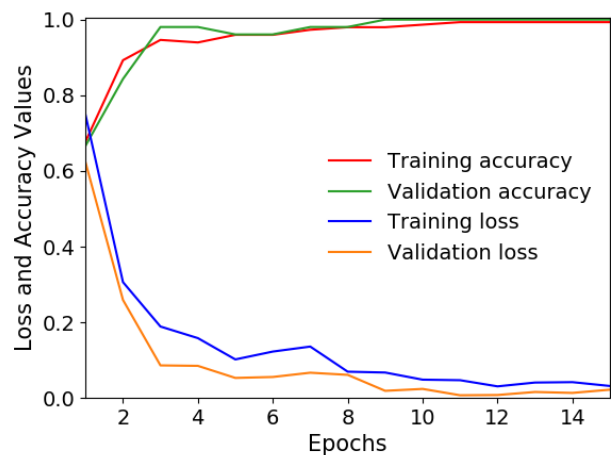


Figure 15. Learning graph for a convolutional neural network with 1 convolutional layer containing four 3×3 filters.

Number of Convolution Layers	Avg. RMS Loss	Avg. Accuracy
1	0.015 (SD 0.011)	1.000 (SD 0.000)
2	0.015 (SD 0.026)	0.994 (SD 0.013)
3	0.037 (SD 0.043)	0.978 (SD 0.023)

Table 5. Accuracies of CNNs with varying numbers of convolutional layers.

In the convolutional layer, the number of convolutional filters and dimension of each filter can be varied. Table 6 displays the average accuracies of neural networks with varying numbers of convolutional filters and filter dimensions. It shows that neural networks with six 3x3 convolutional filters performed the best because they had the lowest average RMS loss and an accuracy of 1.0 over 10 trials ($n=10$), meaning they classified every image in the testing set correctly 10 times in a row.

Number of Convolutional Filters	Filter Dimensions	Avg. RMS Loss	Avg. Accuracy
2	2x2	0.080 (SD 0.078)	0.984 (SD 0.032)
2	3x3	0.038 (SD 0.039)	0.988 (SD 0.019)
4	2x2	0.021 (SD 0.020)	0.998 (SD 0.006)
4	3x3	0.025 (SD 0.046)	0.9901 (SD 0.031)
6	2x2	0.008 (SD 0.006)	1.000 (SD 0.000)
6	3x3	0.006 (SD 0.007)	1.000 (SD 0.000)

Table 6. Accuracies of CNNs with 1 convolution layer containing varying numbers of filters with varying dimensions.

Figure 16 shows trained two of the trained 3x3 convolutional filter matrices and grayscale visualizations of the matrices. Interestingly, the second trained filters has similarities with the horizontal edge detector in Figure 8.

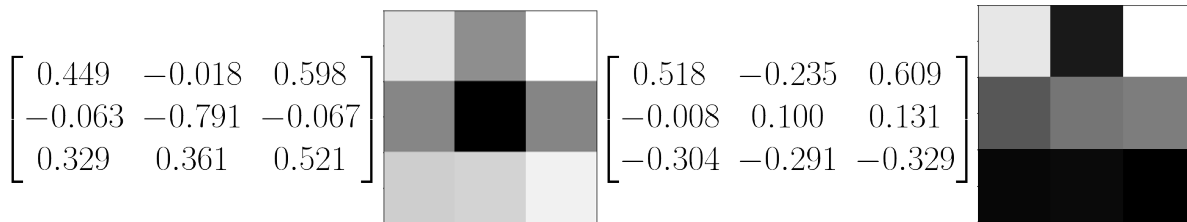


Figure 16. Trained 3x3 convolutional filter matrices and visualizations.

CNNs had much higher accuracies than dense neural networks because they were able to detect patterns using layers of convolutional filters. One convolutional layer with six 3x3 convolutional filters was found to be optimal and will be used going forward.

4.2 Influence of a Sleeping Region on Accuracy

Images of a person lying down in a specific region as if sleeping were introduced to the training and testing data. The accuracy of the neural networks with and without these added images is displayed in Table 7. The sample size is $n=10$ for each row in the table. Table 7 shows that introducing a sleeping region into the testing and training data doesn't significantly reduce the accuracy of fall detection.

Sleeping Region	Avg. RMS Loss	Avg. Accuracy
No	0.006 (SD 0.007)	1.000 (SD 0.000)
Yes	0.028 (SD 0.045)	0.989 (SD 0.028)

Table 7. Average accuracy and loss with and without a bed/sleeping region.

4.3 Influence of Non-Human Heat Sources on Accuracy

Table 8 shows the how the accuracy of the neural network changes as thermal images containing non-human heat sources such as hot objects and pets are added to the training and testing data. Even with hot objects and pets in the thermal images, the average accuracy is still very close to 1, at 0.998. The sample size is $n=10$ for each row in the table.

Non-Human Sources	Avg. RMS Loss	Avg. Accuracy
None	0.006 (SD 0.007)	1.000 (SD 0.000)
Hot Objects	0.023 (SD 0.031)	0.991 (SD 0.016)
Pets	0.013 (SD 0.009)	0.998 (SD 0.005)
Hot Objects and Pets	0.008 (SD 0.006)	0.998 (SD 0.004)

Table 8. Accuracies with and without non-human heat sources in the training and testing data.

4.4 Relationship Between Number of Training Images and Accuracy

The influence of the number of training images on the accuracy of the system is studied in this section. The motivation for this study was to identify the minimum number of images needed to train the system and obtain a high accuracy of fall detection. The system would be easier to use if fewer training image are needed.

Figure 17 is a plot of the number of images used to train the system vs the average accuracy and average RMS loss of the neural network on a testing set of images. The error bars show standard error, and $n=10$ for each data point. Interestingly, the accuracy doesn't start to

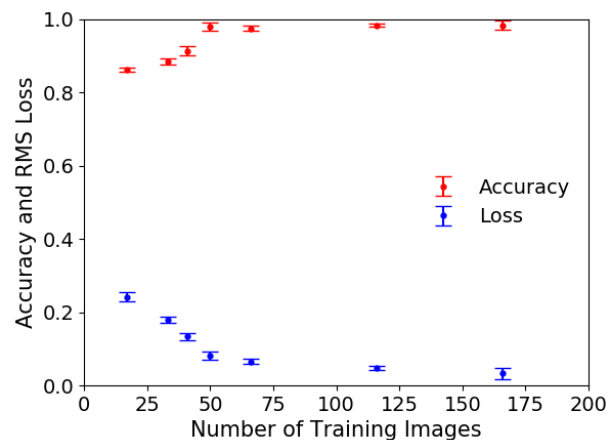


Figure 17. Accuracy and RMS loss of the neural networks when trained with varying numbers of training images.

significantly decline until the number of training images falls below 50.

4.5 Relationship Between Image Resolution and Accuracy

The influence of the image resolution on the accuracy of the system is studied in this section. The motivation for this study was to identify the minimum resolution needed to train the system while maintaining a high accuracy of fall detection. The cost of the system could be reduced if a lower resolution thermal camera is sufficient.

Figure 18 is a plot of the number of pixels in the training images vs the average accuracy and average RMS loss of the neural network on a testing set of images. The error bars show standard error with sample size $n=10$ for each data point. Interestingly, the accuracy doesn't start to significantly decline until the number of pixels in the images falls below 192 (16×12 pixels). The learning graphs for the neural networks when using lower resolution images showed that it took more than 15 epochs for the accuracy to reach a higher and more stable value. Therefore, 40 epochs were used to train the neural networks for each data point in Figure 18. The average accuracy with 80×60 images increased to 1.0 because the neural network had more epochs to stabilize at an accuracy of 1.0 during each trial. An accuracy of 1.0 means that the neural network correctly classified all of the testing images, although the testing set of images is finite.

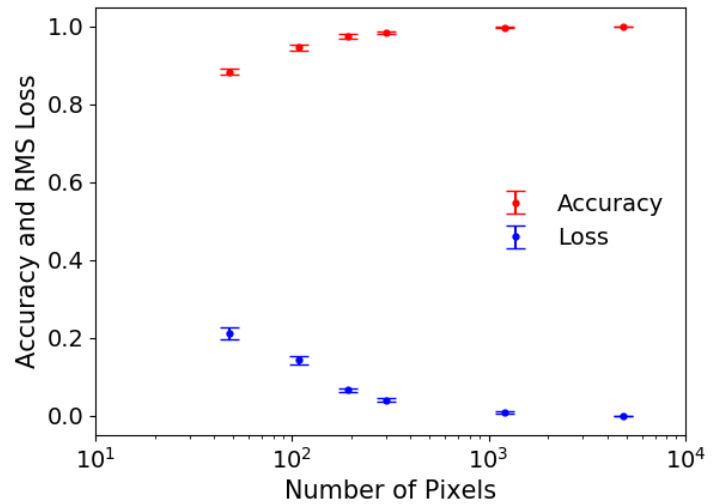


Figure 18. Graph of the accuracy and RMS loss of the neural networks when trained with images of varying resolutions.

5. CONCLUSIONS

An unobtrusive embedded system has been developed for the accurate real-time fall detection of elderly individuals. The wall-mounted system uses a Raspberry Pi, a FLIR thermal camera and deep learning to capture and classify thermal images. The thermal images were able to clearly distinguish humans from the background due to their distinct heat signatures while at the same time preserving their privacy. Deep learning is able to analyze and classify images with a high degree of accuracy due to its effectiveness for complex pattern recognition. A set of thermal images is used to train a convolutional neural network to detect falls. Once trained, the system continuously monitors the living space of elderly individuals to detect falls and immediately call for help, potentially increasing the safety and independence of elderly people living alone at home or in assisted living facilities. The developed system is easy to use and may be sufficiently robust to be used for elderly fall detection after further testing.

It was found that CNNs performed better than dense neural networks and were able to achieve an

accuracy of 98.8% (SD 2.6%) even in the presence of non-human heat sources on the set of testing images. The system was robust and could accommodate a sleeping region without a reduction in accuracy. It is found that approximately 50 training images is sufficient to obtain an average accuracy of 97.9% (SD 3.2%). In addition, a resolution of 16x12 pixels still allowed for an accuracy of 97.5% (SD 1.7%).

The Raspberry Pi 4 (\$55) was chosen for the embedded system since it was powerful enough to run the deep learning models. An 80x60 FLIR Lepton camera (\$200) was used to ensure that the resolution of the thermal images was sufficient. The cost of the resulting system, along with the case and electronic components, is \$270. Based on the speed of the embedded system and having investigated the influence of image resolution on accuracy, it is found that a Raspberry Pi zero (\$10) and a 32x24 MLX90640 IR Thermal Camera (\$60) should have sufficient speed and resolution for a real-time fall detection system. As such, it is possible to reduce the cost of the system from \$270 to \$85 which would make it much more affordable.

6. APPLICATIONS AND FUTURE WORK

In research done by Miguel et al. on fall detection using visible light cameras, they found that a type of apartment commonly associated with the elderly consisting of two bedrooms, a living room, a kitchen, a bathroom and a hallway could be covered using six cameras [15]. Although Miguel et al. used visible light cameras, thermal cameras could also be used to cover an apartment. Using the price estimate for a Raspberry Pi zero and a low resolution thermal camera, this would cost \$510. However, in assisted living facilities or nursing homes, where each elderly individual has a living space of 1-2 rooms, the cost would be much lower (\$85-\$170) per individual. This will increase the safety of elderly individuals in these facilities, which tend to be short-staffed [16].

The embedded system can be used for many applications that requires monitoring using thermal imaging, such as monitoring the safety of autistic children at night. Some autistic children may wander at night when they are unsupervised and could find themselves in dangerous situations [17]. The system could be trained with thermal images of a person in a sleeping area classified as normal and no person in the image or a person outside of the sleeping area classified as abnormal instead of the cases in Table 3.

ACKNOWLEDGEMENTS

I would like to thank Mr. Cary James (Retired Bangor High School Science Department Chair) for his guidance, mentorship, and pushing me to do my best work. I would also like to thank my parents for their support.

REFERENCES

1. CDC. (2019, March 4). *Older Adult Falls*. Retrieved July 20, 2019, from <https://www.cdc.gov/homeandrecreationalafety/falls/index.html>
2. National Council on Aging. State Policy Toolkit for Advancing Fall Prevention Select Resources. Retrieved February 4, 2020, from <https://www.ncoa.org/wp-content/uploads/State-Policy-Toolkit-for-Advancing-Fall-Prevention-Select-Resources.pdf>
3. Igual, R., Medrano, C., & Plaza, I. (2013). Challenges, issues and trends in fall detection systems. *BioMedical Engineering OnLine*, 12(1), 66. doi:10.1186/1475-925x-12-66
4. Daher, M., Diab, A., Najjar, M. E., Khalil, M. A., & Charpillet, F. (2017). Elder Tracking and Fall Detection System Using Smart Tiles. *IEEE Sensors Journal*, 17(2), 469-479. doi:10.1109/jsen.2016.2625099
5. Walabot Fall Alert System: No Emergency Button. No Cameras. No Wearables. (n.d.). Retrieved July 22, 2019, from <https://walabot.com/walabot-home>
6. Vadivelu, S., Ganesan, S., Murthy, O. V., & Dhall, A. (2017). Thermal Imaging Based Elderly Fall Detection. *Computer Vision – ACCV 2016 Workshops Lecture Notes in Computer Science*, 541-553. doi:10.1007/978-3-319-54526-4_40
7. Adolf, J., Macas, M., Lhotska, L., & Dolezal, J. (2018). Deep neural network based body posture recognitions and fall detection from low resolution infrared array sensor. 2018 IEEE International Conference on Bioinformatics and Biomedicine (BIBM). doi:10.1109/bibm.2018.8621582
8. He, W., Goodkind, D., Kowal, P. (2016). *An Aging World: 2015*, International Population Reports. Retrieved January 25, 2020, from <https://www.census.gov/content/dam/Census/library/publications/2016/demo/p95-16-1.pdf>
9. FLIR Lepton. (n.d.). Retrieved February 1, 2020, from <https://lepton.flir.com/>
10. Raspberry Pi Foundation. (n.d.). Raspberry Pi 4 Model B - 4G RAM. Retrieved January 3, 2020, from <https://www.raspberrypi.org/products/raspberry-pi-4-model-b/>
11. Chollet, F. (2018). *Deep learning with Python*. Shelter Island, NY: Manning Publications.
12. Twilio Docs: API Reference, Tutorials, and Integration. (n.d.). Retrieved January 1, 2020, from <https://www.twilio.com/docs>
13. Arora, R.K., (2015) *Optimization Algorithms and Applications*, Chapman and Hall.
14. Amini, Alexander & Soleimany, Ava & Karaman, Sertac & Rus, Daniela. (2018). Spatial Uncertainty Sampling for End-to-End Control. *Neural Information Processing Systems (NIPS), Workshop on Bayesian Deep Learning 2017*.
15. Miguel, K. D., Brunete, A., Hernando, M., & Gambao, E. (2017). Home Camera-Based Fall Detection System for the Elderly. *Sensors*, 17(12), 2864. doi: 10.3390/s17122864
16. Harrington, C., Schnelle, J. F., McGregor, M., & Simmons, S. F. (2016). Article Commentary: The Need for Higher Minimum Staffing Standards in U.S. Nursing Homes. *Health Services Insights*, 9. doi: 10.4137/hsi.s38994
17. Anderson, C., Law, J. K., Daniels, A., Rice, C., Mandell, D. S., Hagopian, L., & Law, P. A. (2012). Occurrence and Family Impact of Elopement in Children With Autism Spectrum Disorders. *Pediatrics*, 130(5), 870–877. doi: 10.1542/peds.2012-0762