Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.Doi Number

Machine Learning Algorithm for Occupancy Detection Using a Thermal Image Dataset

Ryan P. Wood1 and Second B. Author­1

1Department of Mechanical and Aerospace Engineering, University of Virginia, Charlottesville, VA 22903 USA

Corresponding author: Ryan P. Wood (e-mail: rpw6fx@virginia.edu).

This paragraph of the first footnote will contain support information, including sponsor and financial support acknowledgment. For example, “This work was supported in part by the U.S. Depart­ment of Com­merce under Grant BS123456.”

ABSTRACT As energy costs continue to rise, energy efficiency within buildings is becoming increasingly important to monitor in order to save money on lighting, heating, and air conditioning. Many algorithms have been tested to accurately count the number of people in a room or building. However, the vast majority of these are not acceptable to use, as they are either high cost, are unable to count the number of people in an area, or are incapable of reaching a satisfactory accuracy of occupancy detection in a room. In the suggested algorithm, thermal images are used to precisely detect the presence and count the number of people in an area. Initially, images are taken with a camera module connected through wires to a Raspberry Pi 4. The images are then sorted into folders based on the number of individuals in the frame. Using a combination of Python code and a TensorFlow machine learning algorithm, a model is developed to precisely fit the data. Results reveal that the algorithm reaches a training accuracy of around 98% and a validation accuracy of around 99% after the training and testing processes.

INDEX TERMS Algorithm training, algorithm validation, machine learning, people counting, TensorFlow model, thermal imaging

1. INTRODUCTION

Across the United States, 30% of energy used in commercial buildings is wasted [1]. Considering the U.S. spends $190 billion on energy each year powering its businesses, it loses over $50 billion due to the squandering of energy in its commercial buildings. It can be assumed that a significant portion of the energy loss occurs when the internal heating, cooling, and lighting of the building are in use while no one is occupying it. Turning off lights when someone exits a room conserves electricity and money. Using air conditioning in warmer months and heating in colder months keeps occupants of the building comfortable from the outside elements; however, turning the climate control systems off once the room is empty can help limit extra spending on energy. Yet, finding a way to make the environment of a room enjoyable when the individual returns to it should be considered as well. As important as it is to conserve as much energy as possible, having a space that is unbearable to work in, because the heating or air conditioning does not start until an individual is in the room, should not be the case. Creating a device that can conserve energy in unused times, but still have the room comfortable when a person arrives can help to solve the problem of energy overuse in the United States.

1. *PREVIOUS HUMAN COUNTING DEVICES*

Different methods have attempted to solve the problem of energy overuse in recent studies. Each method has been developed and is able for use in public areas, but each of the following has its drawbacks that make it unsuitable.

Carbon dioxide sensors can make predictions for the number of people occupying a room if the building conditions with nobody inside are known. CO2 concentration increases linearly with the number of people inside a room, so knowing the base concentration and current concentration gives an accurate value for the number of people. However, given dynamic conditions inside varying buildings, such as windows or doors being left open for extended periods of time, the sensors can be proved inaccurate when the external scenario is altered [2]. Another drawback of CO2 sensors is the level of carbon dioxide in the air after a person leaves the room. The level does not decrease for quite some time after the individual exits, so the readings and predictions that were correct when people entered are incorrect once they leave.

Radio Frequency Identification (RFID) can yield incredible picture quality. This allows the algorithm to accurately represent the number of people in the area. However, given the picture quality, as well as the fact that each RFID tag is associated with a single person, privacy can become a serious concern.

Like RFID, image cameras have clear picture quality and the ability to easily recognize the number of people in an open room [3]. Yet, the drawback to image cameras lies in the number of positions the camera can be placed. The cameras must be able to see the room, so objects that are not humans can easily crowd the picture, causing the data to be inaccurate. Image cameras also yield privacy concerns, as the clear pictures can expose individual’s identities within the frame of the images.

Acoustic recognition is a low cost, fairly accurate way of determining the count of people in a room. Similar to the carbon dioxide sensors, however, external factors have a large effect on the results of the data collection [4]. If a person works in a room while making no sound, the sensor does not detect them. Also, areas with lots of noise pollution, like shopping malls, make it complicated for the sensor due to the large variety of different voices and non-human sounds coming from a single area.

Finally, Wi-Fi probes can be utilized to compute the number of people in a room. These are extremely accurate if certain conditions are met. The targeted building must have a Wi-Fi network that each individual is connected to. Due to the constant checking of Wi-Fi strength on the device, privacy concerns develop. Moreover, each person must have a device connected to that network on their person wherever they go. If a person leaves a room without their device, the Wi-Fi will still sense them being in that room, leading to inaccurate results from the data collection.

1. *PROPOSED SOLUTION*

This paper communicates a solution that can solve the ongoing problem of energy overuse around the world. The proposed setup counts the occupancy of rooms, allowing users to make decisions about how to best use the HVAC systems in a building. Connecting Raspberry Pi computers with a camera model allows for a constant stream of video to be seen and utilized. Setup of the Raspberry Pi and camera combination is straightforward and can be replicated on any Pi module. All files of code are stored on the single Raspberry Pi, meaning no external storage space is required aside from that which is included in the components of the system. Through running a single line of code, the camera captures the current frame, saving it to the file directory for use. The entire process, taking the picture and examining it, can be accomplished without a network connection, allowing the module to be widespread and used almost anywhere. The use of thermal images allows the identity of each individual to be preserved, eliminating the concern of privacy leaks within the system. This procedure can be duplicated on any module; therefore, the scaling of the project can be easily accomplished to allow users around the world to improve the use of energy in shared spaces. The Raspberry Pi 4 starter kit used costs $140, the entire camera module attachment costs $182, an SD card costs about $25, and jumper wires cost less than a dollar, so the entire system can be built for a little under $350. While other occupancy detection systems designed might be cheaper, this architecture is extremely competitive due to the increase in accuracy and potential for energy savings for a building in the future.

The data flow in Fig. 1 shows the proposed system setup in the experiment. In Fig. 2, the process of transforming input code and thermal images into a Portable Gray Map (PGM) file as an output is described. The process will be further detailed later in the paper.

1. *FLOW OF THE PAPER*

Three main sections in the process of discussing the study’s operation are the system architecture, the dataset, and the algorithm and associated results.

The system architecture section describes the overall process involved in this study. The paper begins by explaining the crucial components of the system. It is followed by a description of the thermal camera used in experimentation. The section is finalized with details of the connection between the Raspberry Pi module and the thermal camera.

The dataset section initially illustrates the specific scenario in which the dataset is created for. It explains why the chosen method of data collection is thermal images. Next, the paper analyzes the images used, how they are sorted, and the specific details of the thermal images. It is followed by an explanation of the dataset architecture and how the directories are organized for best ease of access and convenience during the experimentation. Finally, the section describes the difficulties in creating a cumulative dataset and how those problems are overcome.

The algorithm and results sections relate the proposed algorithm in Python code to the associated results after training and testing. The algorithm section begins by explaining how the data was split. Furthermore, the image processing operation is explained. Moreover, the section details the variables in the Python code and why the variables are set to a specific value. The method in which the model is developed and what filters are used to transform the raw data are also illustrated. After the training, the results section evaluates the outcomes of both the training and validation processes. Additionally, it explains likely sources of error within the experiment and how they can be fixed for future studies. The fifth section ends with a conclusion, as well as a discussion of where future experimentation might be heading.

1. SYSTEM ARCHITECTURE, SETUP OF ALGORITHM­­
2. *ARCHITECTURE OF THE PROPOSED SYSTEM*

All elements of the system architecture have a distinct purpose; understanding what each part does allows the user to connect all components meaningfully. The central component is the Raspberry Pi 4. In this study, the Raspberry Pi module contains 8 GB of Random Access Memory (RAM), and the SD card inserted into the Raspberry Pi contains 128 GB of internal storage. The thermal camera module used in this study is the FLIR Lepton® 2.5. To connect the thermal imager with the Raspberry Pi, the FLIR Lepton® Breakout Board v2.0 and female to female jumper wires are utilized. Although not necessary to detail in more than a sentence, a USB-C power adapter, keyboard, mouse, computer monitor, and other cords necessary to connect the Raspberry Pi to the computer monitor are required for operation.

After the thermal camera is operational, a series of commands in Python code, as well as pointing the camera to take a snapshot of the specified target, produces a PGM image, as shown in Fig. 3. This PGM output is simplified in several ways. Any given pixel in the image has only one or two bytes of data, depending on if it is an 8-bit or 16-bit image. A typical colored image uses three bytes per pixel, one for each shade of red, green, and blue in an RGB file. A PGM file is much simpler and faster for a computer to process. Additionally, this study employs a machine learning algorithm where the computer is looking for patterns in the images; therefore, being able to see a colored image does not carry any benefits over its grayscale counterpart.

1. *DETAILS OF THE THERMAL CAMERA*

Thermal cameras are unlike visible light cameras. Visible light cameras work similarly to human eyesight. When visible light energy reflects off a surface, it bounces back to the target, which then turns the reflected light into an image [5]. Both human eyes and cameras can perceive everyday environments in the full color spectrum using this technique. Thermal imagers use the same process. However, instead of using light waves, thermal cameras use heat waves to detect the presence of an object.

Heat waves and light waves are both contained within the electromagnetic spectrum. Visible light waves, as detected by the human eye, range from 380 to 700 nanometers. Infrared waves, on the other hand, range from 700 nanometers to 1 millimeter. Types of waves with wavelengths shorter than visible light are, in order from longest to shortest, ultra-violet (UV), x-rays, and gamma rays, which are as short as 100 picometers in length [6]. Types of waves with wavelengths longer than infrared waves are, in order from shortest to longest, are microwaves, TV waves, and radio waves, which reach over 100 kilometers in length [7]. The more thermal energy an object emits, the hotter the sensor recognizes the object to be [5]. Because thermal cameras recognize heat, not light, they can operate in complete darkness, which allows for versatility in different environments. Each different temperature value of a thermal imager is represented by a different color, with shades of blue, purple, and other darker colors signifying colder temperatures, whereas brighter colors, like red, orange, and yellow, designate warmer temperatures.

Typically, the resolution of thermal cameras is less than that of visible light cameras. Because heat energy has larger wavelengths that visible light, each sensor in a thermal camera must be slightly larger than that of a visible camera because each needs to take in slightly more information [5]. Therefore, a thermal camera has a much lower resolution and fewer pixels than an average visible camera when both are the same mechanical size.

The thermal camera used in this study is the FLIR Lepton® 2.5. A necessary goal of the experiment is to ensure the privacy of individuals captured by the camera. Some other studies in the past have struggled to maintain a person’s privacy when using their detection device or people counting mechanism. The resolution of the thermal camera used in this study is 80x60 pixels, with each pixel being 17 micrometers on each side. This resolution has a clear enough picture to see the individual in the frame distinctly. It also gives the computer program a quality image to train and validate the dataset on. Most importantly, the privacy of the person being seen in the frame is maintained, allowing for users to find comfort that their identity is not being released.

With a 50° horizontal field of view and 63.5° diagonal field of view, the Lepton 2.5 is able to capture a wide perspective of a space if placed in a useful position, such as the top corner of a room [8]. The camera has a frame rate of 8.6 Hz, allowing for nearly constant updating of the frames seen in the thermal video. The spectral range of the Lepton 2.5 ranges from 8 to 14 micrometers, falling in the long-wave infrared (LWIR) region. Using Wien’s Law [9], the spectral range corresponds to visualizing temperatures between -87.1 and 192.3 degrees Fahrenheit, easily covering the range of temperatures maintained in typical daily environments.

The error in temperatures in the frame ranges from 5 to 10 degrees Celsius (9 to 18 degrees Fahrenheit), or 5 to 10 percent of the overall temperature, whichever is greater. However, this error only affects the collective image, not singular pixel values. Therefore, since the computer in this study uses only the patterns displayed in the image to train, not the overall accuracy of temperature values, the training is unaffected by this deviation.

1. *DETAILS OF THE CAMERA CONNECTIONS*

Connecting the Raspberry Pi 4 with the FLIR Lepton® Breakout Board v2.0 is the single most important step in allowing the camera to work. If the connections are made incorrectly, the thermal camera will not function as originally intended. Worst of all, the General Purpose Input/Output (GPIO) pins of the Raspberry Pi 4 can be fried and make the computer unusable unless replaced. By connecting four wires to the ports on the Raspberry Pi 4, as well as 8 GPIO pins on the Pi module and Breakout Board, the system can be carried out in a manner that does not harm any components involved.

To connect the Raspberry Pi with the power supply, monitor, keyboard, and mouse, 4 ports or the Pi are needed. Two of the USB ports are used to plug in the computer’s keyboard and mouse, allowing them to help navigate around the desktop and terminal once the Raspberry Pi is started. The computer monitor is connected using a micro-HDMI to HDMI cable. The HDMI side can be plugged in anywhere on the monitor. The micro-HDMI plug should be attached to the Raspberry Pi 4 via the HDMI0 port on the side. This connection allows the user to use the monitor once the power supply is turned on. The power supply cable has an on/off button, allowing it to be triggered when the user wants to provide power to the machine. One side of the plug is plugged into any outlet or source of electricity. The other side of the cord is a USB-C plug. It should be connected to the USB-C Power port on the side of the Raspberry Pi 4. When the power is triggered on, all systems of the Raspberry Pi should operate, and the computer desktop and functionality should be visualized on the monitor.

General Purpose Input/Output pins are useful in performing a variety of tasks on a computer, provided they are positioned correctly by the user. On the Raspberry Pi 4, there are four different types of pins: power, ground, GPIO, and special purpose [10]. Connected devices that do not have their own power source are energized using power pins. These pins provide voltage at one of two magnitudes: either 3.3 V or 5 V. Ground pins, on the other hand, do not output a voltage. They ground the circuit for safety reasons to allow stray voltage to discharge, preventing power surges or other catastrophic events while using the device [11]. GPIO pins are configured to send and receive electrical signals. They can be used for data transfer between the connected devices in a setup. Special purpose pins can serve many motives. They are flexible in their use, as they can be placed in a variety of combinations to achieve different results in the system based on the corresponding GPIO pin they are functioning for.

To get the FLIR Lepton® 2.5 functioning, eight different connections must be made between the GPIO pins of the Raspberry Pi 4 and the FLIR Lepton® Breakout Board v2.0. Each jumper wire connected on the Raspberry Pi must also be connected to its corresponding counterpart on the Breakout Board [12]. Initially, the ground pins on both must be connected. When connecting two circuits together, as done with the Pi module and thermal camera, the ground pins must be used. The ground pins also give a common reference for all circuits. Next, the voltage pins with 3.3V must be connected. These power pins provide a source of electricity for the external component, the thermal camera. The voltage pins do not power the Raspberry Pi, as that is done with the power port on the side of the Pi module. The third and fourth connections made are between the SDA pins and SCL pins. Both connections relate to the Inter-Integrated Circuit (I2C) bus interface of the circuit. The main purpose of the SDA pin is to exchange data between the thermal camera and the Raspberry Pi. The SCL pin is a clock on the I2C. Based on when the clock triggers, the data can be exchanged between the two modules. The final four connections made with jumper wires are the Master Out, Slave In (MOSI), Master In, Slave Out (MISO), CLK, and Chip Select (CS) connections. All four connections are included in the Serial Peripheral Interface (SPI). SPI functions much like I2C; however, it can run faster due to the configuration of more wires. The MOSI connection sends data from the master (Raspberry Pi 4) to the slave (breakout board and thermal camera). Conversely, the MISO connection sends data the other direction, from the slave to the master. The CLK pin is simply a clock for the SPI. It functions similarly to the SCL pin in the I2C interface. When the clock is triggered, data can be exchanged between the master and slave. Finally, the CS pins must be connected. For each slave on the circuit, there must be one CS pin connection. Because only one external module is connected, the proposed setup only needs the one Chip Select connection.

If any of the jumper wires are connected improperly, the circuit and camera will not function. Most importantly, it is crucial to connect the ground and voltage pins together in the correct manner; if these connections are wrong, the machine will be fried and will no longer be able to function at all. The specific numbers associated with each pin in the GPIO of the Pi module and breakout board are shown in Table 1. Once everything is connected sufficiently, the camera allows for the collection of images and creation of a thermal image dataset.

1. CREATING THE DATASET

Exploring the overuse of electricity and energy for heating and air conditioning is not a new topic to study. By turning off lights and temperature control functions while a room does not have an active presence in it, the owner of the building can save drastic amounts on electricity, heating, and air conditioning bills. Many solutions have been tried, yet very few use thermal imaging in an attempt to conserve energy.

In this study, a new tactic, thermal imaging, is put into place to make a more usable and practical method to saving money in commercial buildings. Previous studies have used visible light cameras, which remove an individual’s privacy by revealing their facial features. Thermal cameras have a low enough resolution where a person’s facial features cannot be made out; however, the resolution is high enough that a person can be detected in an open space. In addition to a thermal image camera, a machine learning algorithm is tested in order to discover the accuracy of the people counting algorithm used in the study. While people are in a room, the infrared display in the thermal camera allows it to view the internal body temperature of each one of the individuals. After testing and various experimentation, a value representing the ideal difference between internal temperature and room temperature can be found. With this value, a room containing any number of people can have its temperature adjusted to create the closest-to-ideal temperature for each person in the area. Once the room is unoccupied, the temperature control can be turned off, allowing for the saving of energy in a room.

In order to begin development of the algorithm and start training a model for use in commercial buildings, 1055 thermal images were taken using the FLIR Lepton® 2.5. As shown in Fig. 4, each image was labeled with the number of people contained in the frame. After each was labeled, they were placed into folders based on the number of people in the frame. In this study, only images with four or less people are used. Images with more than four people can be trained and validated in future work. In total, there are 101 images with zero people, 149 images with one person, 344 images with two people, 263 images with three people, and 198 images with four people, totaling 1055 images across the five categories.

Each image taken by the camera is 80 pixels wide by 60 pixels high, meaning each frame contains 4800 pixels. Each pixel contains a value, corresponding to the temperature of the object contained in the pixel. Hotter temperatures, like a light that is turned on, or a human body, will have higher values. Colder temperatures, like a table, chair, or floor, will have lower values. Higher values are represented by brighter colors like reds and oranges. Lower values are shown as darker colors like blues and purples. While the frame of the image is shown in the full RGB color spectrum, the code utilized saves each image as a PGM file. Each PGM file only uses one or two bytes of data to represent the color, a decrease from three, which is the number used in an RGB image. The decrease in byte usage allows for quicker processing of each image. Given that the proposed algorithm only focuses on the patterns created by different temperatures in the image, there is no need for extra bytes to be wasted creating a color image for the algorithm.

In creating the dataset, difficulties arose and had to be solved. When initially sorting the images, several frames had fractions of a person in them. In a real-life scenario, someone can be partially obscured, either by another person or by some object, like a cubicle wall, in a room. However, that does not mean that the individual is not in the room at the time the frame was analyzed. Therefore, it was decided that any fraction of a person in an image would be considered a whole person. Some of the study’s error can be attributed to this solution, as the model trains its accuracy slowly, and a series of images containing only parts of people can lead to the model training poorly, like thinking a full person is in fact two halves of separate people.

This dataset is a tool created prior to the training and validation of the proposed model, which is explained in detail in Section IV. The dataset has been made public; therefore, anyone is able to add to the existing database of images. Consequently, in future research, a wider variety of images can be used, allowing for a more complete and thorough training to be provided to the model. Instead of just over one thousand images being used, the goal is to have tens of thousands of images used by the model.

Thermal imaging has many perks over using visible light cameras. Its ability to preserve the privacy of an individual while observing allows for use in a wide range of situations. Additionally, its ability to easily distinguish living and non-living objects simply by observing their heat emitted allows for more consistency than visible light imaging. All in all, the thermal imaging dataset and model for people counting has much more versatility and ability to be consistent than other proposed solutions at this time, authorizing it to be pursued heavily in future studies.

1. ALGORITHM TRAINING AND VALIDATION

Once the images in the dataset are appropriately sorted, the process of developing the people counting algorithm can begin. Through a series of processes, the model can effectively train and validate itself on the complete set of images.

The proposed algorithm in this study involves many steps to transform the set of raw images into an organized collection that can be trained and validated by the suggested algorithm.

In preparation, one must organize the images into folders that clearly label the data with the corresponding number of people in the frame. To begin, a split size for the data must be selected. The allocation of images into both the training image directory and validation image directory can drastically affect results of the study. By convention, developers use 80% of their data for training and 20% for validation. As proved in [13] and shown in Fig. 5, the training and validation split ratios can be estimated to an ideal fraction of the whole, where *n* is the number of adjustable parameters in the code. In the proposed model, there are 24 adjustable parameters, meaning the proportion of data in the validation set should be approximately 0.204, whereas the fraction of data in the training set is about 0.796. These values are extremely close to the 80/20 rule; therefore, the study uses 80% of the images for training and the other 20% for validation.

A function in the code then randomly selects images to go into the training and testing sets, based on the 80:20 ratio of training and validation. Each image is then moved to its appropriate folder into a training or testing directory, and then into a further folder corresponding to the number of people in the frame of the image. Once sorted, the algorithm loops through each PGM image. The return of the function is a 1D array of length 4800. Each value in the array represents the associated value of the pixel in the image after the original image is flattened to a 1x4800 array. Subsequently, a value equivalent to the number of individuals shown in the image is appended to the front of the array, changing its size to a 1x4801. Finally, that array is appended to a corresponding array of arrays, one for training, and one for testing. Once all images are placed into the correct array of arrays, the training and testing arrays are saved as comma-separated value (CSV) file.

Furthermore, the algorithm reads in and splits each successive line of the CSV file into one of two arrays; one accounts for the number of people in the image, and the other contains the values of each pixel in the image. The pixel values are then reshaped into the original 80x60 format for use by the model.

The Python TensorFlow library contains a function called ImageDataGenerator. The arguments of the function allow the user to adjust how the images appear to the model, all to cover as many scenarios to train on as possible.

The rescale argument adjusts the values of all cells by a specified fraction. Typically, it is used to modify all values in a set of cells to be between zero and one. Because the images used in this study are 16-bit, each image was scaled down by a factor of 65,535, the maximum value a 16-bit integer can be. The rotation\_range argument designates a range of values, in degrees, that an image is rotated before training or validation. A thermal image of a person standing up straight can look like a person laying down horizontally if rotated 90 degrees. This function allows for a variety of layouts of images to be tested by simply rotating a single image. In the proposed study, however, almost everyone is standing or sitting vertically in reference to the thermal camera, so the rotation\_range is set to five. The width\_shift\_range and height\_shift\_range parameters adjust how an image is translated, as a fraction of total width and height, before testing. Because the people in the images are spread out across the frame, the value in the proposed study was set to 0.05 for both the width and height shift ranges. Therefore, no person is moved off the frame after a translation, which negatively affects the overall results. The shear\_range parameter adjusts how far, as a fraction of the overall width, an image should be sheared before testing. This effect creates a parallelogram instead of a rectangular image frame. Human silhouettes look relatively similar in shape; thus, it was decided that the human proportions should not be drastically adjusted in the images, so the value was set to 0.05. Next, the zoom\_range argument of the function can be altered. This value represents the percent an image is magnified, about the center, before training. In the proposed study, the Raspberry Pi module is placed in the upper corner of a room. Since the camera is fairly far away from the subjects in the room, most of the people are similar size with respect to the image frame. Therefore, the value was set at 0 for the generator. The horizontal\_flip parameter mirrors an image around the vertical axis before training. As it can reenact a person facing right in frame to facing left, it can create variety and help to fill some gaps in the machine learning algorithm. Hence, the horizontal\_flip Boolean value was set to true. Finally, the fill\_mode parameter can be altered. When an image has a combination of rotation, shear, and translation, it distorts into a non-rectangular shape. The fill\_mode parameter decides how to fill in the new pixels to make a rectangular 80x60 image again. In the proposed model, the ‘nearest’ type was utilized, as the pixel neared to the blank one in the altered image will fill in the gap.

Outside the ImageDataGenerator function, the batch size and number of epochs the model trains for can be adjusted.

Batch size is an often overlooked parameter of the model training. A larger batch size increases the speed at which the model trains. Smaller batch sizes increase the accuracy while training the model due to passing over the data more times per epoch. Typical batch sizes are a power of 2; it is more efficient on the GPU to run a set of full batches than have some processors waiting for others to finish at the end of batching [14]. Therefore, the proposed model began with a batch size of 32; after trying different batch sizes ranging from 8 to 128, a size of 16 was selected.

In addition to batch size, the number of epochs the model trains for is an important feature to consider. If trained for too many epochs, the model overfits and is too specific to the set of training data used. This can backfire when outside data is utilized that is slightly different than that used for training and validation. On the other hand, too few epochs keeps the model from training to its potential, causing there to be gaps when outside data is tested. Beginning with 50 epochs, values from 50 to 750 were tested, with an eventual final number of epochs for the proposed model settling at 250.

As shown in Fig. 6, the model proposed in the study involves many steps to accurately fit the dataset. Any image picked from the dataset begins with 4800 pixels. For a computer model to be able to accurately learn how to distinguish between different numbers of people in images, it would take an extremely long time. Therefore, convolutional layers and pooling are executed in order to minimize the number of parameters the model must understand to train itself. A convolutional layer contains a number of filters, again a number that is a power of two. Each filter is a matrix with a specified size and center at the current pixel. The values represent the weight of each pixel once the filter is multiplied by the original image values. Once all filters are applied, certain features in an image stand out, allowing the computer model to visualize better the characteristics of the image. Because a convolutional layer cannot be 1x1, the outer border of the image is lost with each convolution, causing the image to lose some pixels of width and height each time a convolution is applied. In this experiment, 3x3 convolutional layers are used, causing just the outer layer of pixels to be lost in a convolutional node. Additionally, the first of the three convolutional layers must specify the input shape. Since the proposed study begins with and 80x60 image containing only one layer, the input shape specified is (80, 60, 1). For all the convolutional layers, an activation is also identified. The Rectified Linear Activation Unit (ReLU) is a piecewise activation function used in training. An activation function details how an input is transformed into an output through various nodes in a layer of a network [15]. ReLU specifically returns the input if it is positive; otherwise, it returns zero [15].

In pooling, a subset of an image is taken. In max pooling, the maximum value from that subset is spread across the entire subset. Hence, features with high values will continue to survive when pooling occurs. In the context of this study, an area with pixels of low value can be assumed to contain no living presence. Therefore, a computer model can detect humans in an area the same way a human would looking through a thermal imager. In the proposed algorithm, max pooling with an size of 2x2 was utilized. Therefore, every pooling layer in the model halves the number of pixels in each row and column, decreasing the overall number of pixels in the frame by a factor of four.

The next layer to be used on the data is a Flatten layer. Before the Flatten layer is applied, the data’s shape is (8, 5, 64), as shown in Table 2. Whereas the 8 and 5 represent the number of pixels wide and tall the image is, the 64 corresponds to the number of filters in the last convolutional layer. Therefore, to flatten the data, the image must be transformed into a one-dimensional array. Each convolutional filter adds another later of the table that must be flattened into the array. Each additional filter used in the last convolutional node lengthens the array by the total number of pixels in the image. Therefore, the image, after being flattened, has a total length of 2560 values, each representing a value in a different filter of the original frame.

The final two layers in the model are Dense layers. A dense layer acquires input from each neuron in the previous layer [16]. Through matrix-vector multiplication, as well as a combination of rotating, scaling, and translating the vector, an output is created. The output of the first Dense layer is a one-dimensional vector with length equal to the first argument of the Dense function. Similar to the other parameters, the argument used is a power of two, allowing faster runtimes from the GPU. In the first Dense layer, a ReLU activation is specified for the output. The second Dense layer utilizes a softmax activation function. The softmax function outputs a vector of values that sum to one [15]. Each value represents the probability of an image belonging to that class. The highest probability is the class the model thinks the image most fits into. The proposed model contains training and validation images with occupancy ranging from zero to four people. Therefore, the final Dense layer has an initial argument of five, one for each basket corresponding to a certain occupancy count.

Once all the layers are completed, the data are in its simplest form, with the model assigning each image to the label closest to its predicted occupancy count. Once all images are sorted for the first epoch, the model trains for an additional 249, updating its prediction algorithm in an attempt to have higher accuracy each epoch.

1. EXPERIMENTAL RESULTS

When the model is fully developed, training and validation can occur. As training continues, the training accuracy and validation accuracy steadily increase, indicating the model is getting better at deciding on the occupancy of a room. The loss function of the data is also tracked as training carries on. The loss function models how well the proposed function by the algorithm models the dataset. If the loss is a small value, the data points lie very close to the function of the model. As the value increases, the data points extend further and further away from the line fitted by the model. When the model first begins training, it can be expected that loss is high when the algorithm is still understanding the data. If the function has many parameters in it, it will model the training data with incredible accuracy. However, when the model tries to validate itself with a different set of data, the algorithm has too many parameters in it to have a proficient accuracy. This phenomenon is referred to as overfitting. A common sign of overfitting is when the loss function begins to increase after training for many epochs. Trying to halt training before the model’s loss function increases allows for the best fit over a variety of different combinations of data.

All in all, the proposed model’s accuracy shows its possible value in the HVAC industry in the future. Almost all proposed people counting systems had accuracy less than 90%. After training for five trials of 250 epochs each, this study’s model reached an average training accuracy of 98.10% and a validation accuracy of 98.31% when trained and validated on a set of 1055 images. The training loss and validation loss of the model steadily decreased to average values of 0.0630 and 0.0871, proving the dataset was not overfit on the training data. Graphs of the accuracy and loss functions for both training and validation of each trial are shown in Fig. 7.

1. CONCLUSION

Using a Raspberry Pi 4, FLIR Lepton® 2.5, and machine learning code in Python, a model was successfully developed and tested for use. Given its high accuracy in both training and validation, the suggested algorithm is prepared for service in a wide range of environments to provide occupancy count data. Its ease of use and straightforward setup allow for it to be used by beginners and experts alike.

The validation accuracy reached of over 99 percent allows the model to be widely produced and utilized, providing accurate counting of individuals in a space. Through its occupancy counting algorithm, heating, air conditioning, and electricity can be more appropriately allocated in commercial buildings around the world. The ability of the model to protect individuals’ privacy, while exceeding a proficient accuracy, will allow it to have a vast impact in the future of energy saving.

1. FUTURE WORK

While the study accomplishes a great deal in the field of occupancy counting, there are several improvements to be made to the model in future experiments.

Developing a method for live detection will allow for even further energy savings. If the model can take a picture and analyze it every minute, the allocation of resources in a room will be even more efficient. Additionally, future models should be able to detect larger numbers of people in an area. In this test, only groups of up to four people were tested. If possible, the algorithm will be able to detect the presence of many more people in a space. Next, it will be useful if the model can determine how an individual feels in relation to the room temperature. Given the body temperatures of all individuals in the room, the model should be able to find a temperature that suits all occupants of an area. Finally, future machine learning will be able to detect the type of clothing of an individual. A person wearing long sleeves and pants might be hot in a space, while a person with short sleeves might be cold. Enabling this prediction technique for a machine learning algorithm will even further equip the model for use in different environments.

With these proposed changes, the thermal imaging occupancy detection model will be available for widespread use around the world, solving the world’s current energy use crisis.

REFERENCES

[1] Stauffer, N. (2013) *Reducing wasted energy in commercial buildings*. Available at: https://news.mit.edu/2013/reducing-wasted-energy-in-commercial-buildings (Accessed: 9 August 2022).

[2] Huang, Q., Rodriguez, K., Whetstone, N. and Habel, S., 2019. Rapid Internet of Things (IoT) prototype for accurate people counting towards energy efficient buildings. *J. Inf. Technol. Constr.*, *24*, pp.1-13.

[3] Erickson, V.L., Lin, Y., Kamthe, A., Brahme, R., Surana, A., Cerpa, A.E., Sohn, M.D. and Narayanan, S., 2009, November. Energy efficient building environment control strategies using real-time occupancy measurements. In *Proceedings of the first ACM workshop on embedded sensing systems for energy-efficiency in buildings* (pp. 19-24).

[4] Huang, Q., 2018. Occupancy-driven energy-efficient buildings using audio processing with background sound cancellation. *Buildings*, *8*(6), p.78.

[5] *How Do Thermal Cameras Work?* (2020) Available at: https://www.flir.com/discover/rd-science/how-do-thermal-cameras-work/ (Accessed: 10 August 2022).

[6] Lucas, J. (2018) *What are gamma rays?* Available at: https://www.livescience.com/50215-gamma-rays.html (Accessed: 10 August 2022).

[7] Lucas, J. (2019) *What Are Radio Waves?* Available at: <https://www.livescience.com/50399-radio-waves.html> (Accessed: 10 August 2022).

[8] *FLIR Lepton® 2.5 Full Specifications.* (2018) Available at: flir.com/products/lepton/?creative=599400011978&keyword=flir%20lepton%202.5&matchtype=p&network=g&device=c&utm\_campaign=americas.us.oem.sec.l.aw.rw.oem-restructure.search&gclid=Cj0KCQjwidSWBhDdARIsAIoTVb2huIKee8\_QvXlT0Xcapv28BUCw55E5ks2\_vRqo5ROT6\_m-sdRDG2YaAqRTEALw\_wcB (Accessed 10 August 2022).

[9] Sas, W. (2020) *Wien’s Law Calculator*. Available at: <https://www.omnicalculator.com/physics/wiens-law> (Accessed 10 August 2022).

[10] Butler, S. (2022) *What Is GPIO, and What Can You Use It For?* Available at: <https://www.howtogeek.com/787928/what-is-gpio/> (Accessed 10 August 2022).

[11] *Why do electrical circuits need to be grounded?* (2019) Available at: https://earlybirdelectricians.com/electrical-circuits/ (Accessed 10 August 2022).

[12] *Raspberry Pi 4 Pins – Complete Practical Guide*. (no date) Available at: https://roboticsbackend.com/raspberry-pi-3-pins/#:~:text=used%20for%20communication.-,Raspberry%20Pi%204%20GPIOs,header%20will%20become%20quite%20useful (Accessed 10 August 2022).

[13] Guyon, I. (1997). A scaling law for the validation-set training-set size ratio. *AT&T Bell Laboratories*, *1*(11).

[14] Bierhance, T. (2022) *Do Batch Sizes Actually Need to be Powers of 2?* Available at: https://wandb.ai/datenzauberai/Batch-Size-Testing/reports/Do-Batch-Sizes-Actually-Need-to-be-Powers-of-2---VmlldzoyMDkwNDQx (Accessed: 10 August 2022).

[15] Brownlee, J. (2021) *How to Choose an Activation Function for Deep Learning*. Available at: https://machinelearningmastery.com/choose-an-activation-function-for-deep-learning/ (Accessed: 10 August 2022).

[16] Sharma, P. (2020) *Keras Dense Layer Explained for Beginners*. Available at: https://machinelearningknowledge.ai/keras-dense-layer-explained-for-beginners/#:~:text=The%20dense%20layer%20is%20a,performs%20a%20matrix%2Dvector%20multiplication (Accessed 10 August 2022).