Building occupancy analytics based on deep learning through the use of environmental sensor data

Zheyu Zhang

Thesis submitted to the Faculty of the

Virginia Polytechnic Institute and State University

in partial fulfillment of the requirements for the degree of

Master of Science

in

Electrical Engineering

Saifur Rahman, Chair

GuoQiang Yu

Paul K.Ampadu

May 8, 2023

Arlington, Virginia

Keywords: Building Energy Management, Internet of Things (IoT), Deep learning algorithms (LSTM, Attention), Carbon Dioxide (CO_2)

Building occupancy analytics based on deep learning through the use of environmental sensor data

Zheyu Zhang

ABSTRACT

Balancing indoor comfort and energy consumption is crucial to building energy efficiency. Occupancy information is a vital aspect in this process, as it determines the energy demand. Although there are various sensors used to gather occupancy information, environmental sensors stand out due to their low cost and privacy benefits. Machine learning algorithms play a critical role in estimating the relationship between occupancy levels and environmental data. To improve performance, more complex models such as deep learning algorithms are necessary. Long Short-Term Memory (LSTM) is a powerful deep learning algorithm that has been utilized in occupancy estimation. However, recently, an algorithm named Attention has emerged with improved performance. The study proposes a more effective model for occupancy level estimation by incorporating Attention into the existing Long Short-Term Memory algorithm. The results show that the proposed model is more accurate than using a single algorithm and has the potential to be integrated into building energy control systems to conserve even more energy.

Building occupancy analytics based on deep learning through the use of environmental sensor data

Zheyu Zhang

GENERAL AUDIENCE ABSTRACT

The motivation for energy conservation and sustainable development is rapidly increasing, and building energy consumption is a significant part of overall energy use. In order to make buildings more energy efficient, it is necessary to obtain information on the occupancy level of rooms in the building. Environmental sensors are used to measure factors such as humidity and sound to determine occupancy information. However, the relationship between sensor readings and occupancy levels is complex, making it necessary to use machine learning algorithms to establish a connection. As a subfield of machine learning, deep learning is capable of processing complex data. This research aims to utilize advanced deep learning algorithms to estimate building occupancy levels based on environmental sensor data.

Dedicated to my cherished family and inspiring teachers, who have guided me on my journey to explore and understand the world.

Acknowledgments

First and foremost, I would like to express my deepest gratitude to my advisor, Dr. Saifur Rahman, for his guidance and for providing me with the opportunity to delve into research. It is an immense honor to work under the guidance of the president and CEO of IEEE. I am delighted to be his student for the Advanced Alternate Energy System course and to serve as a Graduate Research Assistant (GRA) and Graduate Teaching Assistant (GTA) under his supervision.

Secondly, I would like to extend my heartfelt thanks to Zejia and Jianfeng, who have shared their research experiences with me and have provided assistance in both my research and personal life.

Lastly, I am eternally grateful to my parents and family members for their unwavering support, both financially and emotionally. Their encouragement has been invaluable throughout this journey.

Thank you, everyone. I am truly fortunate to have met and collaborated with all of you. Your support and contributions have been invaluable, and I am deeply grateful for the connections we have made.

Contents

List of Figures

List of Tables

List of Abbreviations

BEMOSS Building Energy Management Open-Source Software

BEMS Building Energy Management System

CNN Convolutional Neural Network

CO_2 Carbon Dioxide

FFNN Feed Forward Neural Network

HAN Hierarchical Attention Networks

HVAC heating, ventilation, and air conditioning

ICT Information and Communications Technology

IoT Internet of Things

LSTM Long Short Term Memory

NN Neural Network

PIR Passive Infrared Sensor

RNN Recurrent Neural Network

Chapter 1

Introduction

Building occupancy analytics serve as the cornerstone for maintaining indoor environments and boosting building energy efficiency, thereby advancing sustainable development. An accurate analysis model enables us to integrate it into building energy control systems, such as heating, ventilation, and air conditioning (HVAC) and lighting systems, which are the primary energy consumers in buildings. By merging robust occupancy analysis with these systems, we can create a more efficient control strategy, resulting in improved energy efficiency and the realization of a smart building.

Occupancy information is a crucial aspect in building management and can be obtained through the use of sensors. Each type of sensor has its unique advantages and disadvantages. Cameras, for instance, can accurately determine occupancy numbers, but they may pose privacy concerns as they require the detection of people's movements. Passive Infrared Sensors (PIRs) are able to detect the presence or absence of occupants, but cannot provide a precise count. Environmental sensors provide a practical solution to the privacy challenge and have the ability to evaluate occupancy levels effectively.Furthermore, their low cost makes them more feasible for future deployment.

In order to establish the relationship between environmental sensor data and occupancy levels, we utilize machine learning techniques known for their proficiency in function approximation. To enhance the performance of our model, we adopt deep learning algorithms like Long Short-Term Memory (LSTM) and Attention and evaluate their effectiveness by comparing them to using LSTM alone. Improved performance in estimating occupancy levels equates to a higher capability in conserving energy, especially when integrated into building control systems.

Chapter 2

Literature Review

There are three fields related to the topic “Occupancy counting based on machine learning by using environmental sensors”: Building Energy, the Internet of Things, and Artificial Intelligence. Figure 2.1 shows the interdisciplinary branch among these three fields. Following we will summarize the work related to this topic in these three fields.

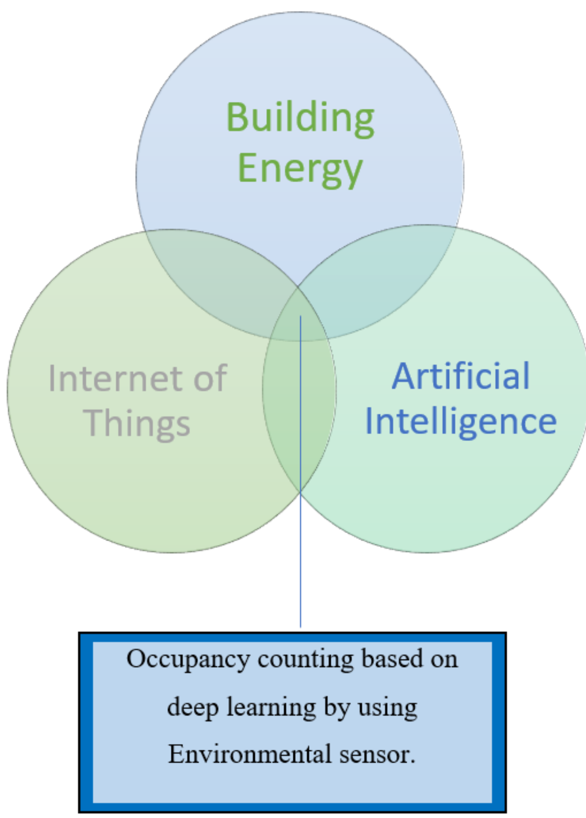

Figure 2.1: Related fields

2.1 Building Energy

2.1.1 Importance of building energy

Building energy is a big consumption in the world. In China, building energy consumption is up to 25% of total social energy consumption [1] and trending upward year by year. Kim et al [2] claim that the global urbanization rate is expected to reach 68% in 30 years from 55% in 2018. Experts in [3] make a prediction that energy consumption can be saved by 34% in existing and new buildings, but the goal should be made by researchers and engineers to explore good energy-saving strategies. Energy utilizations in buildings come from cooling, space heating, water heating, lighting , ICT requirements,plug loads, ventilation,refrigeration,elevator use, etc.[4]. Researchers and engineers are trying to find energy-saving strategies to save as much energy as possible. Al-Ghaili et al [5]state that the factors in the building energy, like energy consumption, energy use, and energy demand, are the problems affecting the building's energy whose objectives are energy savings and occupant's comfort. These factors interact or counteract with each other, for e.g., when we lower energy consumption, and energy demand too much, the energy saving will be good, but the occupants' comfort will be low. As a result, implementing energy-use strategies is crucial for buildings in order to strike a balance between comfort and energy savings.

2.1.2 Energy saving strategies

Many energy-saving strategies exist like [6] trying to find a smart lighting control system to save energy, and [7] focusing on air-conditioning control to save energy. However, we should consider that demand is the basis of energy consumption. Occupancy information constitutes one of the critical building information both in terms of energy consumption and indoor environmental quality [8] since energy can be saved during non-occupied hours when we adopt a proper control strategy. Kim et al [9] also show that there is relationship between occupancy and electricity consumption. Peng et al [10] apply the demand-driven control strategy based on the occupancy data in the office of the commercial buildings in achieving 20.3% energy saving. The occupant's comfort is also a factor we should consider without only focusing on saving energy limitlessly. For e.g., [11] have attempted to minimize energy use without compromising thermal comfort by controlling the HVAC systems.

2.1.3 Occupancy information in building energy

Occupancy information has many properties. Labeodan et al [12] list 6 properties present if the room is occupied, namely the location of the people in the space, the count or number of people, the activities of the people in the space, the identity of the people, and the track or previous location of the people. Sun et al [13] state that only occupancy detection which shows the presence/absence people, counting to indicate how many people are in the space and, identity and tracking to show the activity and location, are the only properties needed to obtain the exact data on occupancy.

Higher-level measurements mean more sensors to collect and more complex algorithms to process the data, which will increase the cost and achievability of building energy system applications. To make occupancy estimation effective for applications [14], occupancy counting will be a good choice with good function and low cost, and compared with identity and location data, it also protects the privacy of people [15]. Occupancy is a hot topic to combine with different areas like machine learning [16], and IoT [17]. The occupancy also has some application areas like control systems [18], Building Energy Management [19], and Edge computing [20]. The above papers show the importance of building energy saving, for effective and low-cost occupancy measurement methods are worth investigating and improving.

2.2 Machine Learning Algorithm

Machine learning is a hot topic in Artificial Intelligence. [21] and [16] made a review of the machine learning algorithms applied to Occupancy estimation like Feed Forward Neural Networks (FFNN), Recurrent Neural Networks (RNN), Convolutional Neural Network (CNN), and Decision-Tree.

2.2.1 Feed Forward Neural Networks (FFNN)

FFNN structure can work as a universal function approximator to approximate any continuous function [22], so it can apply in a wide range of applications. The common FFNN implementation usually adapts two hidden layers to process input data and performs relatively well in the low-scale room environment [23]. When it comes to a larger scale environment, however, the performance will drop, because FFNN lacks temporal interdependence between features [24]. Also, some works like [25] find out the limitation of FFNN on detecting the occupancy period's beginning and end.

2.2.2 Convolutional Neural Network (CNN)

Convolutional NN can discover time series' structures and generates features accordingly if we apply it to the process of time-series data [26], as a type of deep learning, it can learn significant features itself [27], but it needs big data to get a good performance.

2.2.3 Decision Tree

The decision tree algorithm can select the class by deriving the decision node tree where each internal node represents the comparison of a feature value with a learning threshold [28], it has been used to build a framework to combine sensors to do occupancy prediction and gets good result [29].

2.2.4 Recurrent Neural Networks (RNN)

Recurrent NN allows temporal dynamics behavior, it is a common module applied in occupancy estimation named Elman neural network [30]. However, the traditional RNN has major problem named the vanishing gradient. To solve this problem, Long Short Term Memory (LSTM) was developed [31]. In my current research, my module will be based on the LSTM methodology. LSTM is also a kind of deep learning; it requires a big data size and can process the time series data ensuring good performance. Above was a summary of the deep learning methods applied in occupancy measurement.

2.3 IoT sensor

The IoT sensor is an essential part to measure occupancy, there are many kinds of sensors like cameras, Wi-Fi, Passive infrared (PIR) sensors, CO_2 sensors, and electricity meters. There are many research papers focusing on images or videos based on the camera [15], using the images from the camera , using computer vision area to do the people counting in the classroom[32], and using overhead to measure the person tracking and get very high accuracy up to 99%. It seems that the camera is enough to do the occupancy measurements with this so high accuracy and high performance in measuring high-level estimation like tracking and counting [33]. However, because overhead can get more information about people, there will generate more problems. Like privacy problems cameras also can detect people's faces, low-quality images will affect the result by illumination and high cost in deployment with the camera.

2.3.1 Wi-Fi Devices

Wi-Fi devices are also a good way to count people when people connect to the local network with their smart mobile devices [34], while in [35], authors also got good performance on occupancy tracking and detection with high accuracy. But the limitations of Wi-Fi devices are also clear, i.e., people without mobile or decline to connect the Wi-Fi devices will directly decrease the accuracy because Wi-Fi devices will not be able to take them into consideration.

2.3.2 Passive infrared sensors (PIR)

PIR sensors have good performance on occupancy presence [36] and in many real-life applications like our lab room on the roof, there is a PIR sensor to detect my presence by checking my motion, and PIR sensors are a type of low-cost device. Raykov et al [37] tried to use a single PIR sensor to count people but found the result inaccurate, thus demonstrating the limitation of PIR sensors for occupancy counting. PIR sensors only detect a specific area of a room which can be inconvenient in estimating occupancy. For e.g., we sit at a desk next to the entrance and the ceiling PIR sensor located in the center of the room cannot detect my motion. As a result, the lighting turns off frequently and we have to stand up, move towards the center of the room to shake my hand to show my presence to the PIR sensor.

2.3.3 CO_2 sensors

The CO_2 sensor is good at occupancy analytics, [38] shows the correlation between the number of people and CO_2 level. It is the same as a PIR sensor at a low cost and is easy to install [39]. Jiang et al [40] predicted occupancy numbers by using CO_2 concentration and got a very high accuracy, but this result was because absence period contains a huge part of the total time, and their module can detect the absence of the room but when people present in the room the accurate number of people was also a challenge to them to predict. Moreover, the CO_2 rate can be affected by the detector's location and the slow data time delay is also a problem [41].

2.3.4 Electricity meters

Electricity meters are more like PIR sensors on occupancy detection in that they also can only check the occupancy presence. Kleiminger et al [42] derive occupancy data from the electricity meter by monitoring the electric-load curve. Although electricity meter is a low cost device for the experiment, this method will bring privacy risks indirectly to the residential house to display the presence or absence of the house [43]. And this kind of data only display absence or existence in the room, doesn't have the ability to estimate the occupancy level if we put it into the commercial building.

As a result, only using one type of sensor is hard to get a good result, it's beneficial to combine different types of sensors with their advantages to make up for their limitations.

2.4 Research specificity

The research is focusing on Occupancy Analytics, where data from an environmental sensor named Netatmo will be used to collect indoor environmental data in the target building. A deep learning technique named Attention as the data processing method to develop analytics about the building occupancy. By comparing with the hot deep learning algorithm named Long Short Term Memory(LSTM), show the better performance model that can make the occupancy estimation more accurate.

2.4.1 Environmental device

Netatmo collects and records data for five indoor parameters in buildings: temperature, humidity, sound, CO_2, and air pressure, which parameters do not raise any privacy issues because these do not identify any individual's characteristics or behavior while inside the building. [44] shows using non-intrusive sensors can get good performance on the indoor occupancy level. What's more is that the cost of the device is reasonably low and easily deployable for this experiments like this one.

2.4.2 Long Short Term Memory(LSTM)

LSTM is a good deep-learning algorithm that has already been applied in similar experiments showing a very good performance in accuracy than other algorithms. In reference [45], the authors assessed occupancy levels using environmental data, excluding sound data, and demonstrated a detection accuracy of up to 95% and a classification accuracy of up to 76.04%. This paper, however, introduces a tolerance of up to 7 people in order to achieve even higher accuracy.

The work by Ramanujam et al [46] improves upon the LSTM method of Occupancy estimation by adding Convolutional NN and getting the accuracy up to 0.946. However, the data size is too small (only four days of data) as well as the size of the room (only can contain 4 people) as mentioned in [40]. However, the methodology may not be good for counting people when the number of people increases as in my proposed experiment environment. The important works mentioned above do not take HVAC into consideration. HVAC works as a part of the building system and therefore its working status can affect all data that sensors will collect and in turn affect the occupancy estimation. Although [46] has some differences or limitations with respect to my experiment, it is essential to add more powerful mechanism to improve the performance.

2.4.3 Attention Mechanism

Attention mechanism first proposed by [47] in natural language processing area and [48] made a breakthrough work by introducing a new architecture for neural machine translation called Transformer that uses a purely attention-based mechanism to model dependencies between the input and output sequences.

The many-to-one attention is a kind of attention mechanism to improve the performance in the natural language processing area like machine translation, text summarization, and sentiment analysis.

The many-to-one attention mechanism is typically used in conjunction with encoder-decoder architectures for natural language processing tasks such as machine translation, text summarization, and sentiment analysis. The encoder network processes the input sequence and produces a sequence of hidden states, which are used by the attention mechanism to compute the context vector. The decoder network then generates the output sequence based on the context vector and the previous decoder output.

However, the many-to-one mechanism can be applied to other domains beyond natural language processing, where there is a sequence of input vectors and a single output vector. So it's reasonable to apply this mechanism to our model since we have five inputs in time series sequence and one output vector(occupancy level). Following is detailed information about how to the mechanism works.

[49] and many papers show the attention mechanism becomes a standard component in many neural machine translation models, as well as in other natural language processing tasks. The attention mechanism has also been extended and modified in various ways to improve its performance and address specific problems in different domains. So it's reasonable to apply the attention mechanism to train our data to predict the occupancy level.

2.4.4 Contribution

My research focuses on the application of the many-to-one attention mechanism[50] to occupancy level estimation. Occupancy level estimation is an important problem to solve because it can help optimize building energy use by enabling more efficient heating, cooling, and lighting control.

First, we select an appropriate sensor for collecting environmental data by reviewing relevant research papers and websites. This helps us to evaluate various types of sensors, ultimately leading us to choose an environmental sensor. Next, we identify a suitable experimental location where people regularly attend scheduled classes.

Second, we analyze the environmental data obtained from the sensor. We find that the environmental data changes when people are present in the room, and the peak levels of CO_2 vary depending on the number of people occupying the space. As a result, it is feasible to use environmental data to estimate occupancy levels.

Finally, by using environmental data as input, my model with many-to- one attention mechanism can estimate the occupancy level of a building at any given time without getting privacy data of people and in a low cost. The many-to-one attention mechanism is an advanced deep learning algorithm that has the ability to outperform traditional RNN or LSTM models. By incorporating this mechanism, I aim to increase the accuracy of my model and ultimately save more energy in future applications.

Chapter 3

Experiment

3.1 Data Collection

The experiment took place on the sixth floor of the Virginia Tech Arlington campus building, in classroom 6-053, which has a capacity of approximately 25 people. Figure 3.1 illustrates the weather station positioned in the center of the classroom to enhance the accuracy of detection values, as the center offers an equal distance from every seat.

We obtained input data directly from the Netatmo weather website. However, for output data(for verification purposes), such as the number of people, we had to count manually at the start of each class. The Netatmo mobile app features a notification system that alerts us when CO_2 levels exceed a predetermined threshold, indicating the presence of people in the classroom. This enables us to visit the classroom(for occupancy number verification purposes) as soon as we receive the notification, rather than waiting for the entire day, in case someone is there for a brief meeting.

Table 3.1: Input data from Netatmo

Temp	Hum	CO_2	Noise	Pressure
21.5	43	482	53	1020.8
21.6	43	504	52	1020.7
21.6	43	497	52	1020.6
21.7	43	504	56	1020.6
21.7	43	528	59	1020.6
21.7	43	535	57	1020.6

Figure 3.1: Experiment place

3.2 Data Analysis

In the analysis, two values vary dramatically when people join the classroom: CO_2 and sound. These values increase to indicate the presence of people. CO_2 increases as people stay in the room due to respiration, while sound increases due to the activities in the classroom such as classes, meetings, and speeches, generating noise and disrupting the normal sound level.

Although sound is a good indicator of people's presence, to determine the number of people in the classroom, it is not consistent. We chose to use the CO_2 level as the occupancy level indicator. More people generate more CO_2 gas, making it necessary to use a machine learning model to establish the relationship between CO_2 level and occupancy level.

To demonstrate the validity of the research, we chose two different days with varying numbers of people in the classroom for comparison. Firstly, we selected a day on Jan 31, 2023, with CO_2 and sound level data in Figures 3.2 and 3.3. In Figure 3.2, the CO2 level increased from the normal level of 565ppm to a peak value of 754ppm, fluctuating around 710ppm, indicating the presence of people in the classroom. When people left, the CO2 level slowly dropped from 687ppm to the normal level.

Figure 3.2: CO_2 in 1/31/2023

Figure 3.3 shows when people inside room the sound level increases directly up to 70dB and then varied between 75dB to 64dB. Then when people left, the sound level dropped back to 64dB as was the case before. It shows the ability to react if there is people in the room, but since big fluctuations are observed when people are present, it is not have a reliable indicator of room occupancy level.

To perform a comparison, we also selected another day on Feb 9, 2023, with sound and CO2 data shown in Figures 3.4 and 3.5. The sound level trend was similar to that of Jan 31, increasing and decreasing quickly when people entered and left the classroom. However, the CO2 level was different. In Figure 3.2, the CO2 level dropped slowly to the normal level in

Figure 3.3: Sound in 1/31/2023

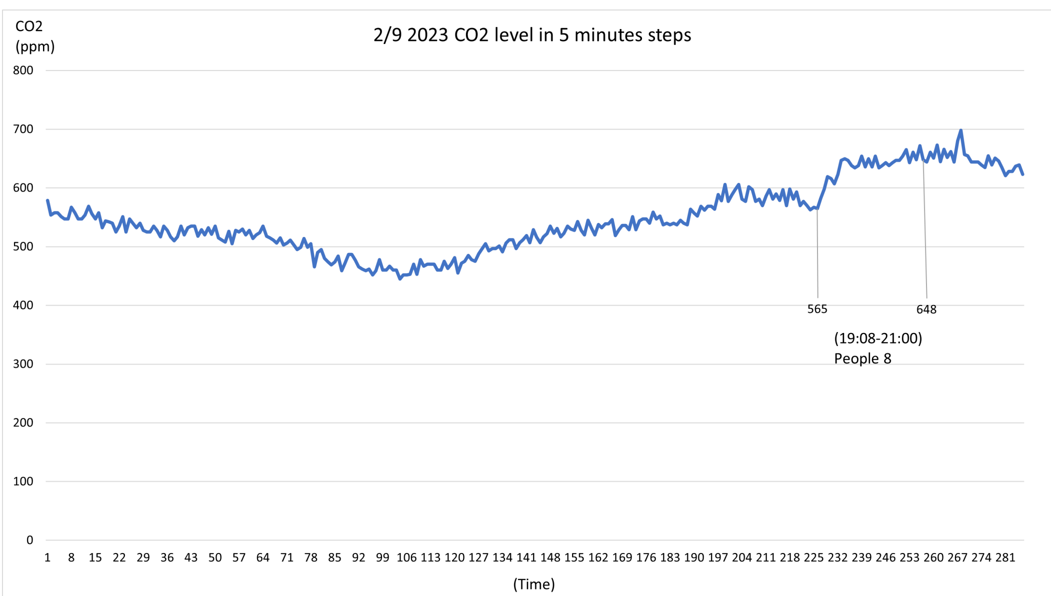

Figure 3.4: CO_2 in 2/9/2023

a few hours after people left, while in Figure 3.4, the CO2 level did not drop over time.

This difference is due to the presence of an HVAC system in our experimental location. The system operates at a high rate during weekdays' working hours and returns to a low-power state at night. It usually starts working at a high rate around 6:00 am and returns to a low rate around 9:00 pm, which can be observed in the sound level figures. When the system is working at a high rate, it generates noise, as seen from the sound level increase from 51dB to 64dB before returning to 51dB. The same condition applies to the CO_2 level, where the system can reduce the CO_2 concentration quickly when operating at a high rate(due to larger volume of air), after people leave. However, when people leave when the airflow volume is low, the CO_2 level remains high. This is why we cannot rely solely on the CO_2

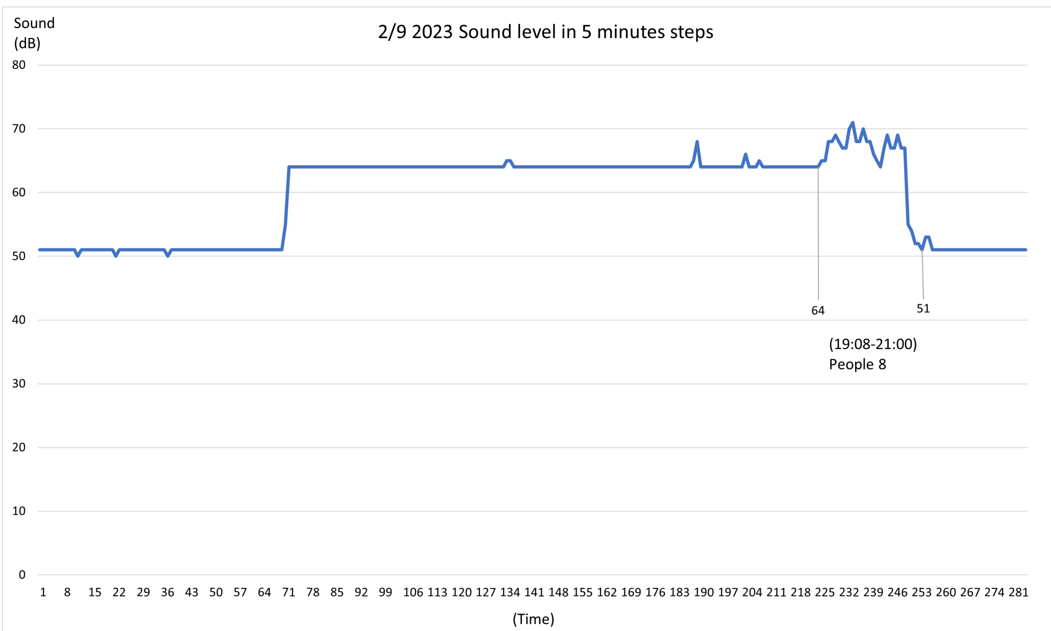

Figure 3.5: sound in 2/9/2023

parameter to estimate the occupancy level. If the CO_2 level remains high during the low air flow rate of the HVAC system, the model may assume that there are people in the room, even though there are no individuals present. This occurs because the air flow is low, and the CO_2 level does not decrease as rapidly as when the HVAC system is operating at a high rate. Therefore, it is essential to consider other parameters, such as sound levels, to better estimate the number of individuals in the room.

My model tries to incorporate several parameters, including sound and CO_2 data, temperature, humidity, and pressure. Although temperature, humidity, and pressure may not exhibit significant variation in the database, they play a crucial role in supporting the sound data to determine the air flow rate of the HVAC system.

Sometimes people will be present in a room during periods of low HVAC system activity, leading to abnormal sound levels. In such cases, the temperature, humidity, and pressure data can help identify the actual rate of HVAC by providing additional information about the room's conditions. Thus, by combining multiple parameters, my model can provide a comprehensive and accurate estimation of the occupancy level.

After determining that it is meaningful to apply environmental data for occupancy level determination, we need to put data in a proper format to allow the deep learning algorithm to train the data and build a relation between the environmental data and occupancy level.

3.3 Data Pre-processing

3.3.1 Normalization

The aim of normalization is to convert all data values into a range between zero and one, in order to facilitate subsequent data processing. This process transforms large numbers in a database into a smaller range, enabling the model to perform computations on smaller numbers instead of large ones. As a result, normalization improves the model's convergence rate, allowing for quicker execution and time savings in the calculation process.

In deep learning calculations, using normalization before initiating gradient descent can speed up the iteration process by reducing the calculation range. Additionally, normalization can help deep learning models avoid gradient explosion, improving overall model stability. Two common normalization methods are Min-Max and Z-score.

Min-Max normalization transforms the values of the data source into the range of 0 to 1. The maximum value in the data is designated as v_{max}, and the minimum value as d_{min}. The data value d is then expressed as a percentage of the difference between d_{max} and d_{min}, as shown in Equation 3.1. This process preserves the relationship between previous data values such that values close to the minimum result in values close to 0, and values close to the maximum result in values close to 1. When d equals d_{max}, the result will be 1, and when d equals d_{min}, the result will be 0.

$$\bar{d} = \frac{d - d_{min}}{d_{max} - d_{min}} \tag{3.1}$$

Min-Max normalization can reserve the relationship in the previous data that value close to d_{min}, the result will be close to 0, the same with the d_{max}, but if there is unusual value which is very big in the data, that will result the difference of the value near the d_{min} will smaller. In addition, if in the future data the range is exceed the current range [min,max], the system will show the mistake and we need to set up new min and max for the data again.

Z-score, also called standard score, is a normalization method that requires the overall data mean value ($\bar{d}$), overall data standard deviation (σ), and individual data (d) before processing. Using equation 3.2, we map the data into a small range. After Z-score processing, the mean value of the overall data is zero, and the standard deviation is one.

$$z - score = \frac{d - \bar{d}}{\sigma} \tag{3.2}$$

Z-score is more adaptable for our model because it can solve situations where the maximum/minimum is unknown. Furthermore, it can process unusual data that exceeds the range of normal data. This is helpful when facing different situations in the future where some data may vary from previous data and need updating. Using the Min-Max method, we may face system error. However, with Z-score, if unusual data exists, the overall standard deviation (σ) will increase and map the unusual value into a small range.

We use the Z-score normalization method to preprocess the input environmental data since the input data range is unpredictable, and it provides better performance. For the output data, we use one-hot encoding to process it.

3.3.2 One-hot encoding

When dealing with the number of individuals prior to one-hot encoding, we should consider the maximum capacity of the room, which is typically around 20 people. However, it is rare for a classroom to have more than 15 students. Although the probability of having more than 15 individuals is very low, it cannot be completely dismissed since there may be circumstances in the future in which there are more than 25 people in the classroom. Thus, we group any number above 15 as 15, which puts them in the same category.

One-hot encoding is a technique that uses binary digits with values of 0 or 1. We set the length of this group to 16 to accommodate categories ranging from 0 to 15. For example, if the number of individuals is 15, after one-hot encoding, only the 16th bit will be set to 1, while all the others will be set to 0.

In the output layer, the data is represented using a single item in one-hot encoding format. To ensure that the sum of all one-hot indices equals 1 and transform the output index range from 0 to 1, we apply the Softmax activation function. The Softmax function, outlined in equation 3.3, is used to obtain output probability instead of a single number. Here, j represents the total number of items from the previous neural network nodes, while i represents the index of the item. The probability of a given element is calculated by dividing the exponent of the element by the sum of the exponents of all the elements.

$$S_i = \frac{e^j}{\sum_i e^j} \tag{3.3}$$

The resulting probability of each index shows the probability of the people number predicted by the model based on the input data. We then transform the index with the highest probability back to the corresponding people number, thus presenting the model's prediction of the number of people in the classroom.

Table 3.2 displays an example of the output data before the transformation into people numbers. We observe that the thirteenth index has the highest probability, indicating that our model predicts 12 individuals in the classroom. This approach ensures that the model provides accurate predictions of the number of people in the classroom, thus improving the effectiveness of our analysis. Upon categorization and implementing one-hot encoding, it's

Table 3.2: One hot encoding

People	0	1	2	3
Possibility	3.15e-02	1.11e-03	1.98e-01	1.21e-01
People	4	5	6	7
Possibility	1.74e-01	9.97e-04	7.63e-04	3.76e-02
People	8	9	10	11
Possibility	1.0577185e-03	4.60e-04	8.24e-04	7.56e-04
People	12	13	14	15
Possibility	4.27e-01	2.04e-03	9.29e-04	4.26e-04

vital to shape the data suitably for input into the attention mechanism. We use Python as our programming language of choice to build the model and tap into deep learning algorithm libraries. Given that the algorithm can compute and produce results independently, our workflow aligns with the stages illustrated in Figure 3.6. These stages encompass data collection, data analytic, and pre-processing to comply with the format demanded by the deep learning algorithm.

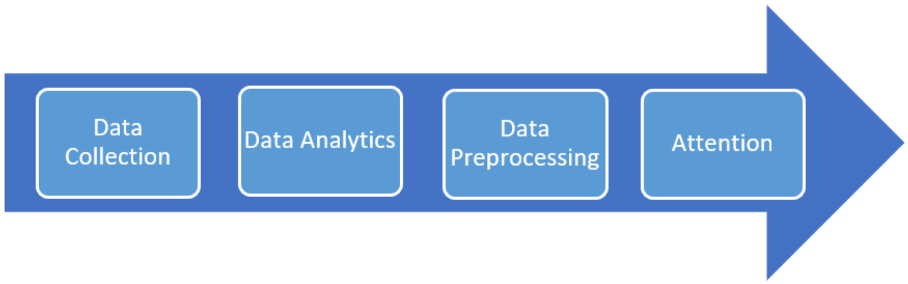

Figure 3.6: Experiment structure

3.3.3 Loss Function

Loss function plays a crucial role in estimating the disparity between the predicted and true values. A lower loss function value implies that the model has better robustness. In our case, we use the one-hot encoding technique along with the softmax activation function to process the output data. Therefore, it is essential to implement the categorical cross-entropy loss function defined in Equation (3.4), where t_i represents the true label, and p_i denotes the probability obtained from the softmax function. As the range of people is between 0 to 15, n is assigned a value of 16.

$$L_{CE} = -\sum_{i=1}^{n} t_i log(p_i) \tag{3.4}$$

Upon analyzing Table 3.3, we can observe that the predicted number of people is 12, whereas the actual count is 7. Consequently, the loss between the two tables is substantial. However, we can minimize the loss function by adjusting the value of index 8 to be closer to 1, while the other indices should be closer to 0.

Table 3.3: True value

People	0	1	2	3
Result	0	0	0	0
People	4	5	6	7
Result	0	0	0	1
People	8	9	10	11
Result	0	0	0	0
People	12	13	14	15
Result	0	0	0	0

3.3.4 Optimizer

The optimizer is an essential tool for determining the minimum loss result in order to achieve better model performance. We employ the Adaptive Momentum (Adam) optimizer to attain the best results. Based on gradient descent, Adam is a stochastic optimization technique that has gained widespread popularity. It merges the strengths of AdaGrad and RMSProp, offering adaptive gradients and the capacity to operate under non-stationary circumstances. Adam stands out as the preferred optimizer for several reasons:

1) Straightforward implementation, effective computation, and minimal memory usage

2) Updates to parameters remain unaffected by gradient scaling transformations

3) Hyper-parameters are easy to comprehend and generally demand minimal or no fine-tuning

4) Appropriate for scenarios involving large-scale data and parameters

5) Relevant for unstable objective functions

6) Well-suited for sparse gradients or those with considerable noise

7) Ability to self-adjust the learning rate

Currently, Adam functions as a high-quality optimizer, adept at managing most scenarios due to its significant benefits. Furthermore, various extensions of Adam exist that enhance its performance, such as AdaMax, Adadelta, and Nadam. To determine the optimal fit for our model, we ought to evaluate each optimizer individually and compare their performances. Ultimately, we should select the one that yields the best results as our final choice.

3.4 LSTM module

3.4.1 Introduction

Long-Short-Term Memory (LSTM) is a unique type of recurrent neural network that surpasses the performance of conventional recurrent neural networks. One of its key advantages is the use of a gate mechanism that controls the flow of information through the network, thereby reducing the vanishing gradient problem that plagues traditional RNNs. Furthermore, LSTMs have the ability to effectively process long sequences of data. An LSTM is composed of multiple LSTM units, as shown in Figure 3.7. A single LSTM unit has three gates: the forget gate (f), the input gate (i), and the output gate (o).

3.4.2 LSTM Cell

The forget gate is responsible for determining how much historical information should be incorporated into the current cell state calculation. The function of the forget gate is illustrated in 3.5. The last time step statement, s_{t-1} in the figure is h, and the current time input, x_t, are used in conjunction with weight matrices (W_f) and a bias vector (b_t), which can be learned through training. The gate controller, which uses the sigmoid function, outputs a

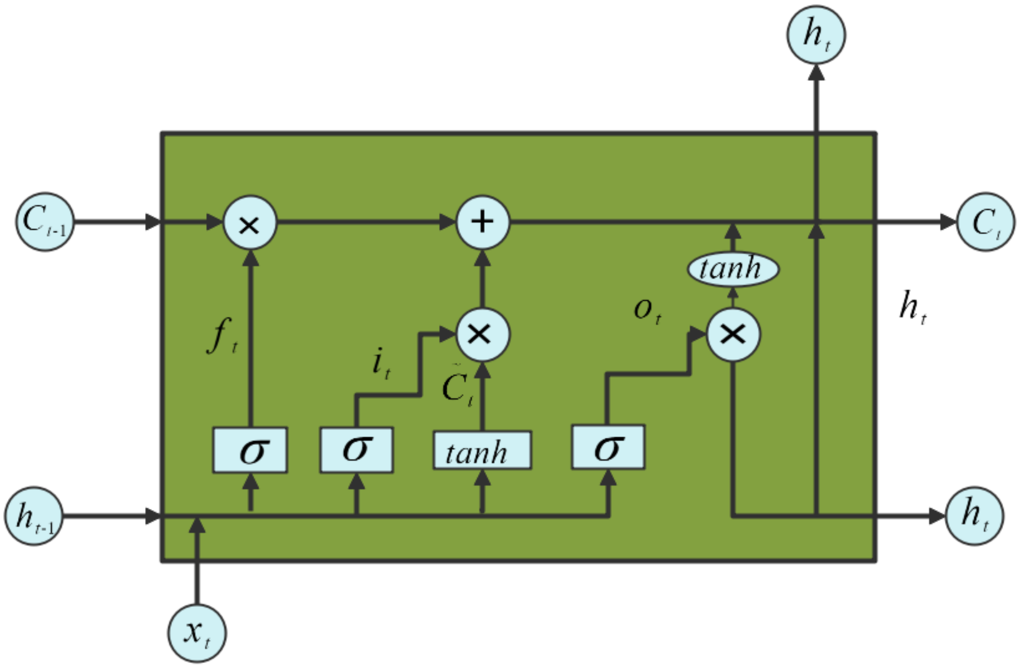

Figure 3.7: LSTM cell

value between 0 and 1 for the information stored in cell state C_{t-1}.

$$f_t = \sigma(W_f \cdot [s_{t-1}, x_t] + b_t) \tag{3.5}$$

The input gate is used for updating the new information into the new cell state. First, the Sigmoid function is used to determine the relevance of the new information, s_{t-1} is the output of last state. Next, the candidate values $\hat{C}_t$ are generated using the hyperbolic tangent (tan) function. These two functions are combined to update the new information into the new cell state.

By using the forget gate and input gate, it is possible to update the new cell state C_t by forgetting some historical information and integrating new information . The cell statement is the most important and unique component of LSTM, as it participates in the calculation of the output for the current state. Subsequently, it can then proceed to the next statement, wherein it can obtain updated information and forget some historical information.

The output gate is derived from both the current input, x_t, and the previous state output, h_{t-1}, and is utilized in calculating the current state output. In order to achieve this, the current cell state, C_t, is also involved. Similar to the forget gate, the output gate employs the Sigmoid function to decide the output information, while the tanh function is utilized in conjunction with the cell state to obtain the current output.

3.5 Attention Mechanism

3.5.1 Introduction of many-to-one mechanism

The many-to-one attention mechanism is a powerful tool that is frequently used in conjunction with encoder-decoder architectures to tackle natural language processing tasks like machine translation, text summarization, and sentiment analysis. In these tasks, the encoder network first processes the input sequence and produces a series of hidden states. These hidden states are then utilized by the attention mechanism to calculate the context vector, which is later employed by the decoder network to generate the output sequence based on the previous decoder output.

Despite being widely used in natural language processing, the many-to-one mechanism can also be applied to other domains where there is a sequence of input vectors and a single output vector. In our case, since we have a time series sequence with five input vectors and one output vector representing the occupancy level, it makes perfect sense to incorporate this mechanism into our model.

3.5.2 Application in the model

Many to one mechanism firstly used in [50] named Hierarchical Attention Networks(HAN), HAN includes word attention layer and sentence attention layer, by getting the hidden state from Bidirectional RNN to train the data. Word in our model is the input features like noise or temperature about the sentence we assume the whole input sequence as a sentence since our data don't like the sentence has the connection with the previous segment.

To analyze the input data at the word level, we begin by feeding it into a Bidirectional LSTM to generate the hidden state output h_{it}. Here, i represents the index of the sentence layer and t represents the index of the word within the current sentence. In our model, we only have one sentence layer. However, not all input data contributes equally to the final result, so the HAN incorporates an attention mechanism to weigh each individual input's importance. This involves computing the attention vector u_{it} (as shown in Equation 3.6) and using the softmax function to determine the importance weight α_{it} (as shown in Equation 3.7). Based on these weights, we can compute the sentence-level attention factor s_i as the sum of the word-level weights (as shown in Equation 3.8).

$$u_{it} = \tanh(W_w h_{it} + b_w) \tag{3.6}$$

$$\alpha_{it} = \frac{\exp(u_{it}^T u_w)}{\sum_t \exp(u_{it}^T u_w)} \tag{3.7}$$

$$s_i = \sum_t \alpha_{it} h_{it} \tag{3.8}$$

When it comes to analyzing input data at the sentence level, the method is similar to that used for individual words. As shown in Figure 3.8, the authors of the paper constructed a many-to-one attention network to connect sentences and words. However, in our model, we

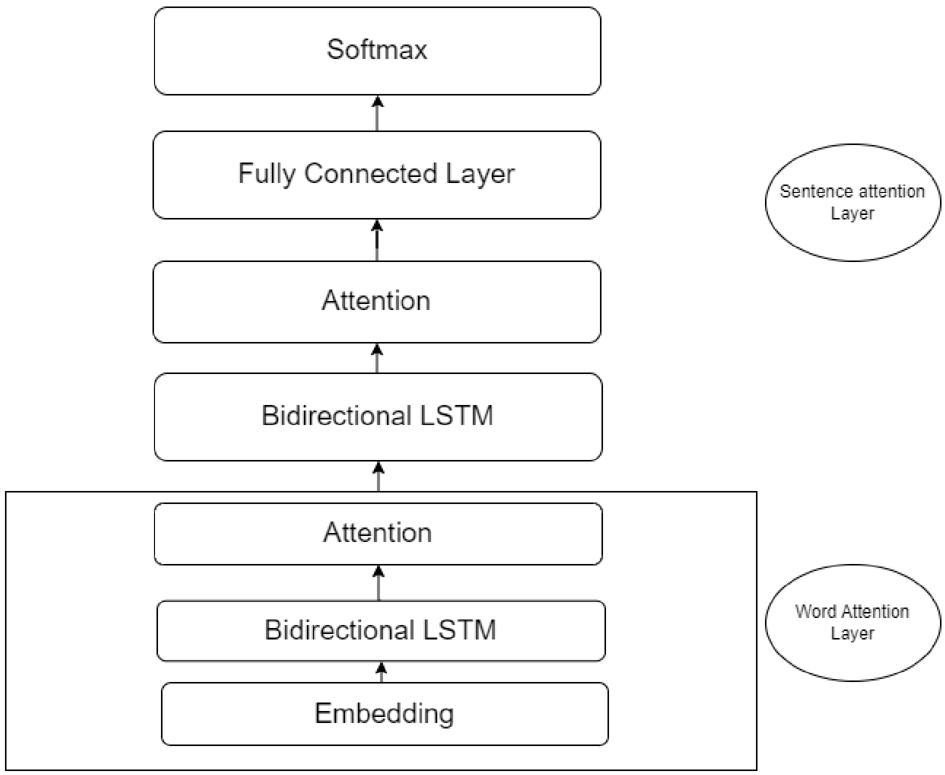

Figure 3.8: Attention structure

don't require connections between sentences. so we only have one sentence layer size and can obtain the result directly. Finally, the authors use a fully connected layer to combine all the units together and integrate them to generate a single output via the softmax function again.

Chapter 4

Evaluation

4.1 F1 score

The F1 score is an effective evaluation method for categorization tasks, as it incorporates both Precision and Recall. For multi-category problems, the micro-F1 method can be employed to assess the performance of a model by comparing its predictions with the ground truth.

In the case of micro-F1, the Recall can be calculated using equation (4.1). Here, TP represents True Positive, which indicates that for category m, both the predicted and actual results are m. FN stands for False Negative, meaning that the actual result is m, but the predicted result is not m. Recall, then, represents the proportion of True Positives when the actual result is m.

$$micro - Recall = \frac{\sum TP_m}{\sum TP_m + \sum FN_m} \tag{4.1}$$

Precision can be calculated using equation (4.2), where FP represents False Positive. This means that the actual result is not m, but the predicted result is m. Therefore, Precision is the proportion of True Positives when the predicted result is m.

$$micro - Precision = \frac{\sum TP_m}{\sum TP_m + \sum FP_m} \tag{4.2}$$

The F1 score in equation (4.3) is based on the harmonic mean of Precision and Recall, striking a balance between these two metrics, as they often exhibit a trade-off. Generally, when the Precision value is high, the Recall value tends to be low, and vice versa. The F1 score calculation aims to balance these two parameters in order to evaluate the overall performance of the model.

$$micro - F1 = 2\frac{Recall_m \times Precision_m}{Recall_m + Precision_m} \tag{4.3}$$

4.2 Performance

This section presents all the result figures, accompanied by F1 scores. The x-axis of each figure indicates five-minute intervals, while the y-axis denotes the number of people, with a maximum of 15. We make this assumption for cases with more than 15 people to prevent situations where the number of people might inadvertently exceed the room's capacity.

In Figure 4.1, we employ the LSTM algorithm without the use of one-hot encoding. The performance illustrated reveals a linear decrease or increase in the number of people, indicating that during certain time periods, the people count is not a whole number. The performance depicted in this figure is unsatisfactory, as the prediction and actual outcome lines do not show significant overlap and F1 score is the lowest of all results 0.7703.

The whole number issue is resolved by incorporating one-hot encoding, as depicted in Figure 4.2. However, the performance remains unsatisfactory, as there is no predicted value in the first wave, even though the actual value is ten, and the F1 score is 0.9382.

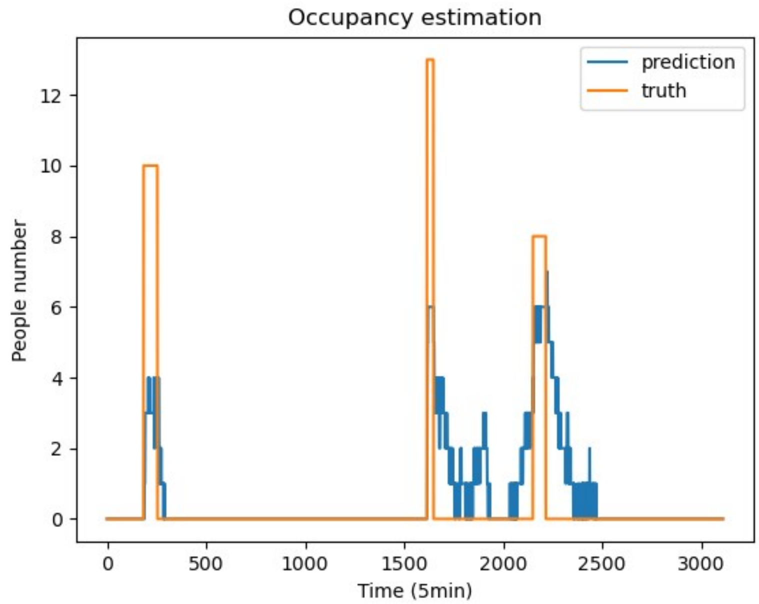

Figure 4.1: LSTM without one-hot encoding

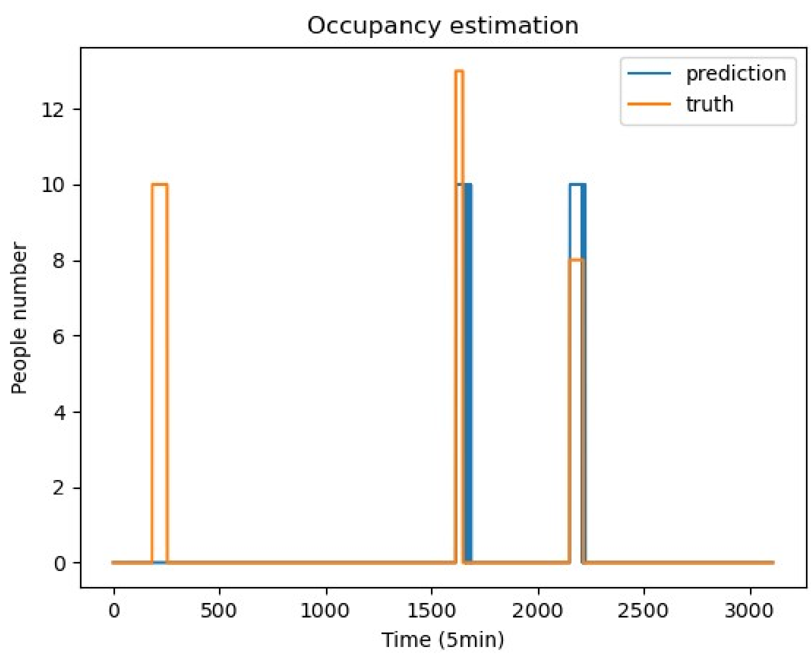

Figure 4.2: LSTM with one-hot encoding

Performance improves when we apply the attention algorithm along with the Adam optimizer, as shown in Figure 4.3. The model predicts the number of people with low tolerance error, sometimes even yielding the exact count. However, the F1 score does not increase significantly (0.9385) due to the presence of noise, causing the predictions to drop to zero and then return when people are present. This makes it difficult to accurately estimate the real-time number of people. For this reason, we begin searching for a better optimizer to

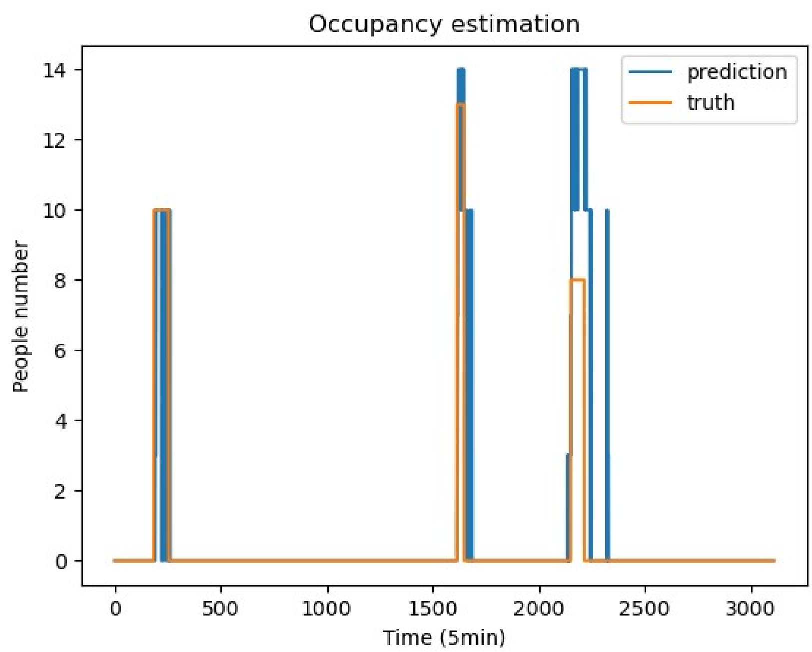

Figure 4.3: Attention-Adam

enhance performance. Adadelta in Figure 4.4, one such optimizer, turns out to be a not good choice due to its time-consuming nature, with epoch times reaching up to 500, and its results disregarding the first wave. However, the F1 score experiences a slight increase to 0.9417, as Adadelta performs well when no people are present. This is unlike LSTM, which sometimes predicts the number of people to be outside the expected range, even when no one is in the room.

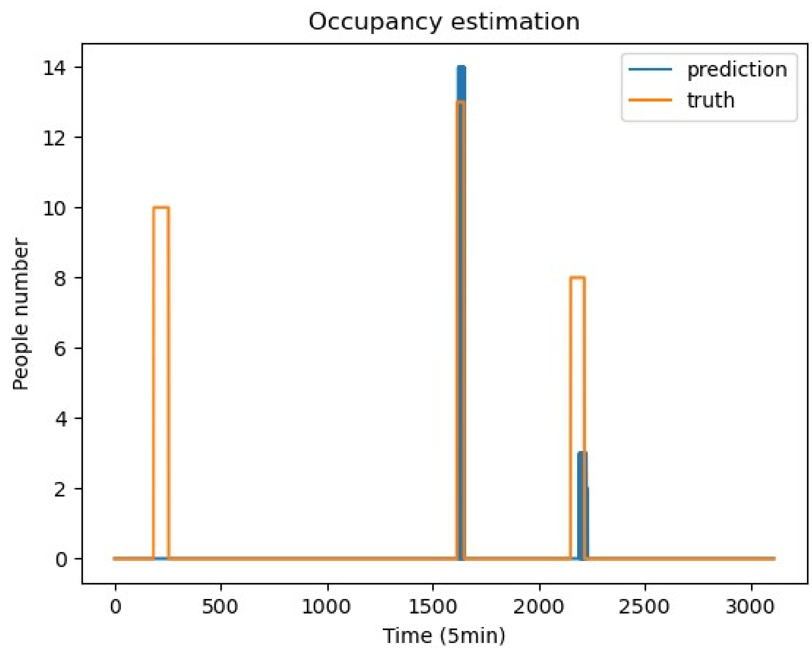

Figure 4.4: Attention-Adadelta

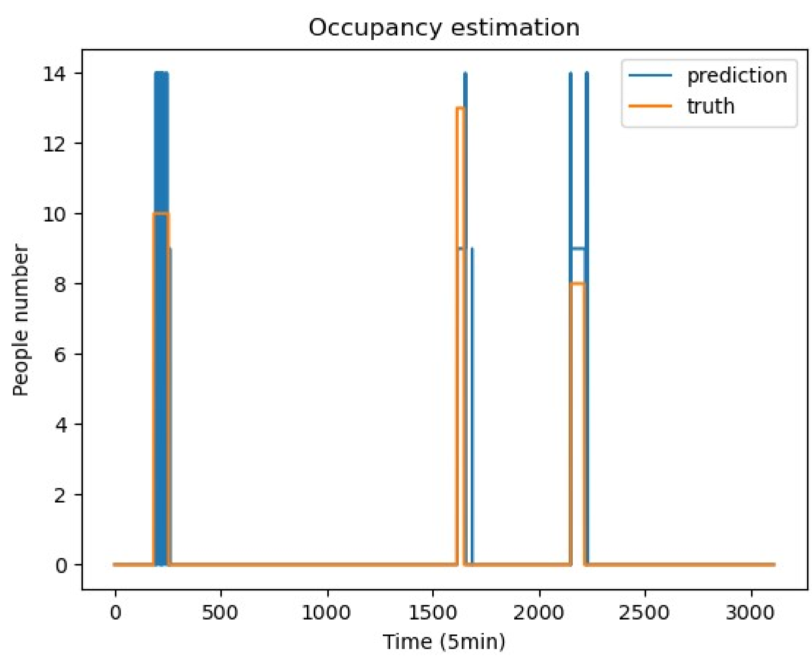

Figure 4.5: Attention-Ftrl

The Follow The Regularized Leader (FTRL) algorithm in Figure 4.5 was developed at Google during the 2010s for click-through rate prediction. It is well-suited for addressing shallow models with large and sparse feature spaces. However, in our model, it takes longer to stabilize, necessitating up to 800 epoch times, and the overall performance is somewhat unsatisfactory, with an F1 score of 0.9356, despite the figure appearing promising.

RMSprop in Figure 4.6, short for Root Mean Squared Propagation, is an extension of the gradient descent optimization algorithm. In RMSprop, the learning rate changes over time instead of being a fixed hyperparameter, making this algorithm suitable for solving small batch models. In our case, it demonstrates decent performance with lower epoch times of 10 and a higher F1 score of 0.9485. This result appears acceptable if we cannot achieve better performance using other optimizers.

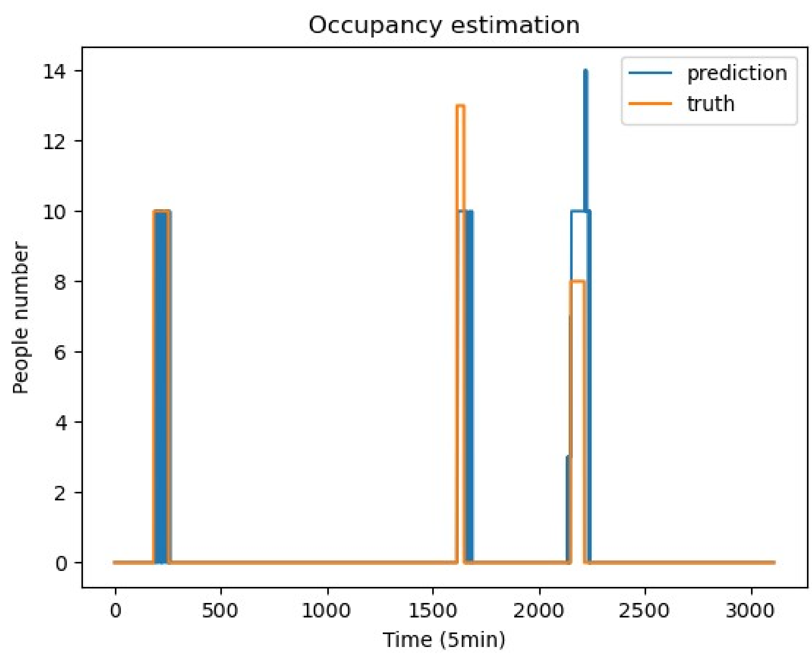

Figure 4.6: Attention-RMSprop

Nesterov-accelerated Adaptive Moment Estimation in Figure 4.7, or Nadam for short, is an extension of the Adam algorithm that incorporates Nesterov momentum to improve performance. However, in our model, the performance is subpar as it misses the first prediction wave, even though the epoch time is relatively short at 10. The resulting F1 score is 0.9456.

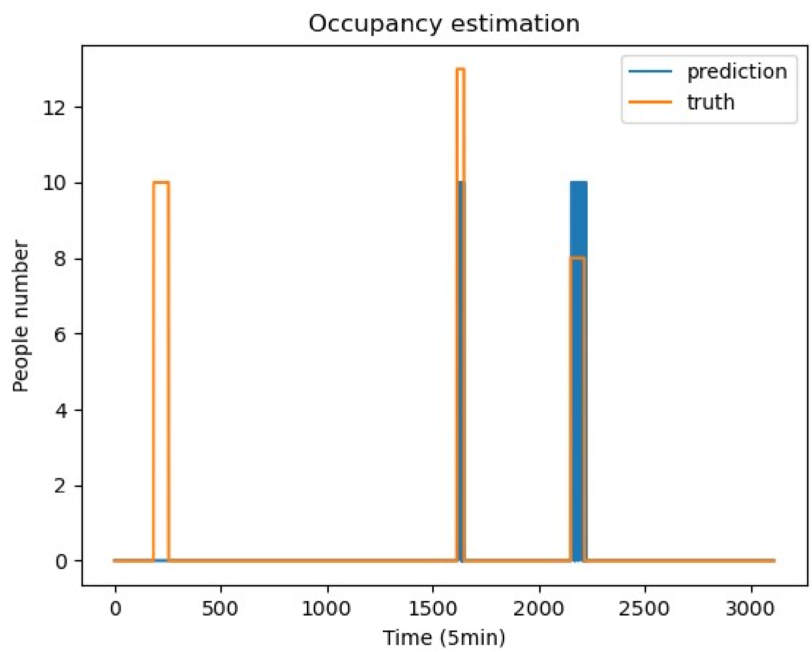

Figure 4.7: Attention-Nadam

The Adaptive Gradient Algorithm in Figure 4.8, known as Adagrad, is a form of stochastic optimization that can adjust the learning rate automatically, eliminating the need for manual modifications. While our model can precisely represent all wave patterns, it does not excel at predicting the presence or absence of people. In some instances, the model anticipates people being present when they are not. Consequently, the F1 score stands at 0.9257, requiring as many as 250 epochs to reach this performance level.

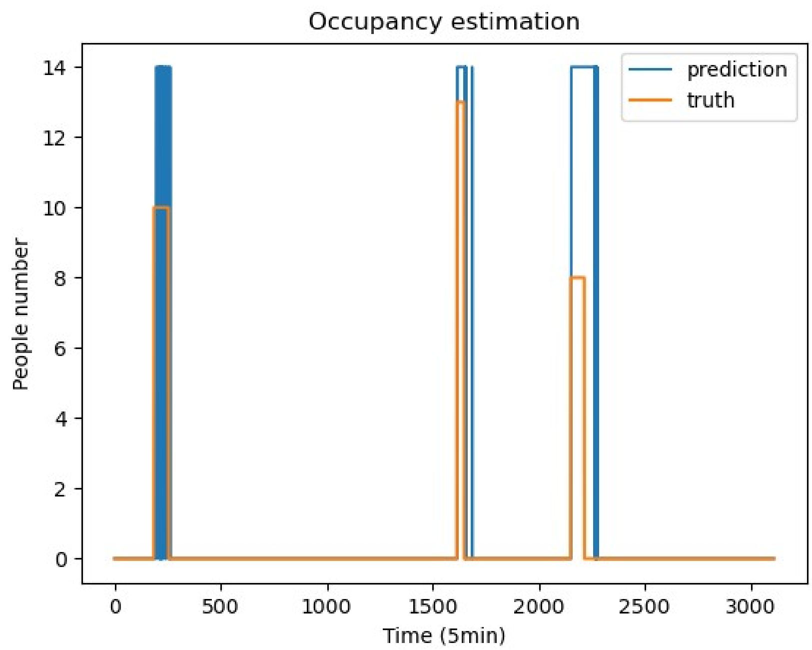

Figure 4.8: Attention-Adagrad

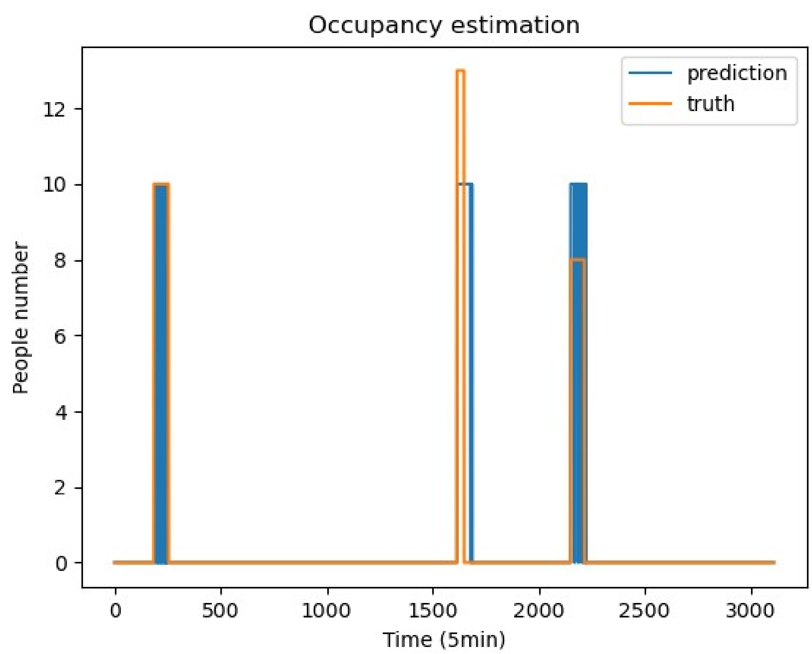

Figure 4.9: Attention-SGD

Stochastic gradient descent(SGD) in Figure 4.9 is an iterative method similar to gradient descent optimization. It is an important method in machine learning with long history about 70 years. In our model, it shows relative good performance with F1 score 0.9437 and epoch times 50. The result is similar to RMSprop method is acceptable if we can not find better method.

As depicted in Figure 4.10, Adamax is an extension of the Adam version of gradient descent and can generalize the approach to the maximum. In some cases, it may yield better results. In our model, Adamax outperforms all other optimizers under the dropout value is 0.3 achieving an F1 score of 0.9579 after 50 epochs.

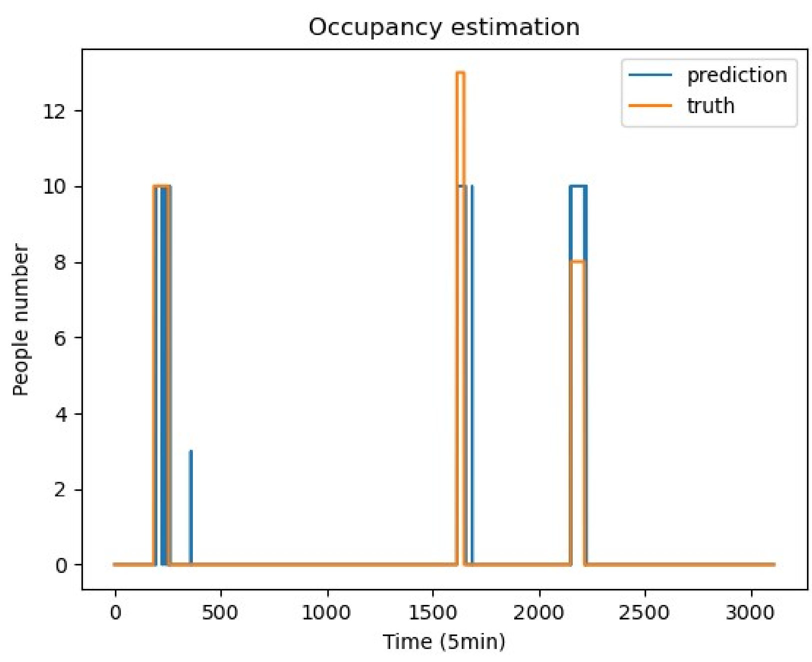

Figure 4.10: Attention-AdaMax dropout=0.3

To attain the best performance, I modified some parameters, such as dropout, to stabilize the model and ensure acceptable results. Given that attention is a complex deep-learning technique, our training data might not be sufficient, potentially leading to over-fitting. To mitigate over-fitting, we employed the dropout method and adjusted the corresponding parameter in our model. Dropout is a regularization technique used to train neural networks with various architectures simultaneously. During training, the model randomly drops out or disregards some layer outputs, forcing network layers to compensate for errors made by preceding layers.

In Figure 4.11, we keep using AdaMax optimizer and decreasing the dropout to 0.2. With the same epoch times the result is not good as dropout=0.3

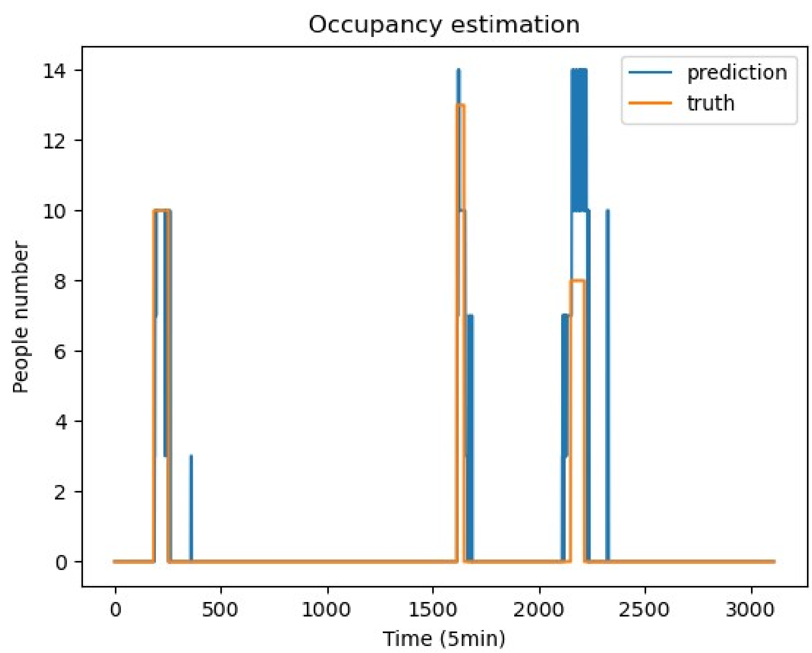

Figure 4.11: Attention-AdaMax dropout=0.2

In Figure 4.12, we increased the dropout value to 0.4 to examine if performance could be improved. Upon conducting the initial test, the result was better than that of the 0.3 case, with the F1 score rising to 0.9585 – the highest value we have ever achieved. If the model remains stable, we will consider this result as our final outcome.

However, when we ran the same model with identical parameters again in 4.13, the perfor-

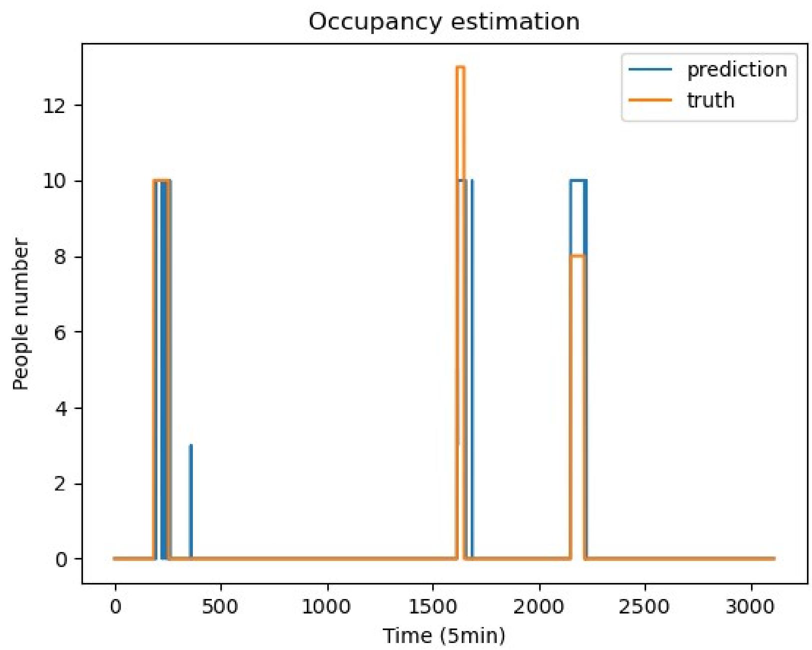

Figure 4.12: Attention-AdaMax dropout=0.4

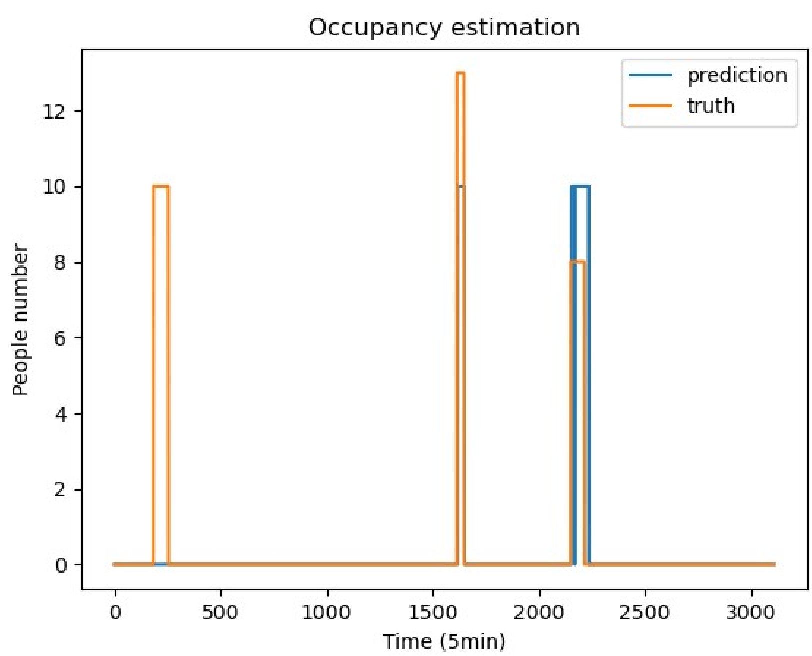

Figure 4.13: Attention-AdaMax dropout=0.4 unstable

mance changed, as the model entirely missed the first wave and the F1 score dropped to 0.9385. We conducted several additional tests, and the results were inconsistent – sometimes the performance was excellent, while other times it was entirely unacceptable. This outcome indicates that the model, under these parameters, is unstable and unreliable. Therefore, we cannot use the model in this state to estimate occupancy levels.

Table 4.1 summarizes all results using different algorithms and parameters. It is evident that the many-to-one attention algorithm outperforms the LSTM algorithm under our experimental conditions. Furthermore, by adjusting various parameters, we discovered that the attention algorithm with the AdaMax optimizer and a dropout value of 0.3 offers the best performance in terms of both stability and F1 score.

In terms of applying these results,if we prioritize reduced computation time, we can use the RMSProp optimizer to obtain relatively acceptable results. For the most accurate occupancy level estimation, we should employ the attention algorithm with the AdaMax optimizer.

Table 4.1: Performance Summary

Algorithm	Optimizer	dropout	Epoch	F1
LSTM(no-onehot)	Adam	0.3	70	0.7703
LSTM	Adam	0.3	70	0.9382
Attention	Adam	0.3	8	0.9385
Attention	Adadelta	0.3	500	0.9414
Attention	Ftrl	0.3	800	0.9356
Attention	RMSProp	0.3	10	0.9485
Attention	Nadam	0.3	10	0.9456
Attention	Adagrad	0.3	250	0.9257
Attention	SGD	0.3	50	0.9437
Attention	AdaMax	0.2	20	0.9337
Attention	AdaMax	0.4	50	0.9585
Attention	AdaMax	0.4	50	0.9385
Attention	AdaMax	0.3	50	0.9579

In summary, we have presented the results obtained for estimating occupancy levels. We have demonstrated the improvement offered by the attention algorithm and identified the best performance achievable under this algorithm.

Chapter 5

Conclusion and Limitation

In conclusion, occupancy information is an invaluable asset for optimizing energy usage in buildings, taking into account the diverse activities of the occupants. To accurately estimate occupancy levels, we harnessed multi-variable environmental sensors and deployed the Netatmo Weather Station in a classroom to gather time-sequenced data for our experiment. We subsequently adapted a deep learning algorithm, the Attention Mechanism, which is characterized by its intricate structure. This approach empowered us to delve into the relationship between environmental data and occupancy levels. By adjusting and comparing different settings, our model demonstrated considerable performance in estimating occupancy levels. Our findings confirm that the Attention Mechanism outperforms the LSTM algorithm in accurately estimating occupancy levels when processing environmental data.

As the experiment progressed, I discovered a limitation of my research data: the data is affected by seasonal changes. The experiment took place from November to February, during the winter months in Arlington. During this time, the HVAC system operates at high power to heat the room. However, as the weather warms up in March and April, the HVAC system does not need to use as much power to maintain a comfortable temperature, resulting in reduced noise from the machinery compared to the winter months. From Figure 5.1, it is evident that there is no significant difference in sound levels between person number 4 and person number 8. Consequently, sound levels may not be efficient in estimating the number of individuals in a room but can effectively gauge their presence.

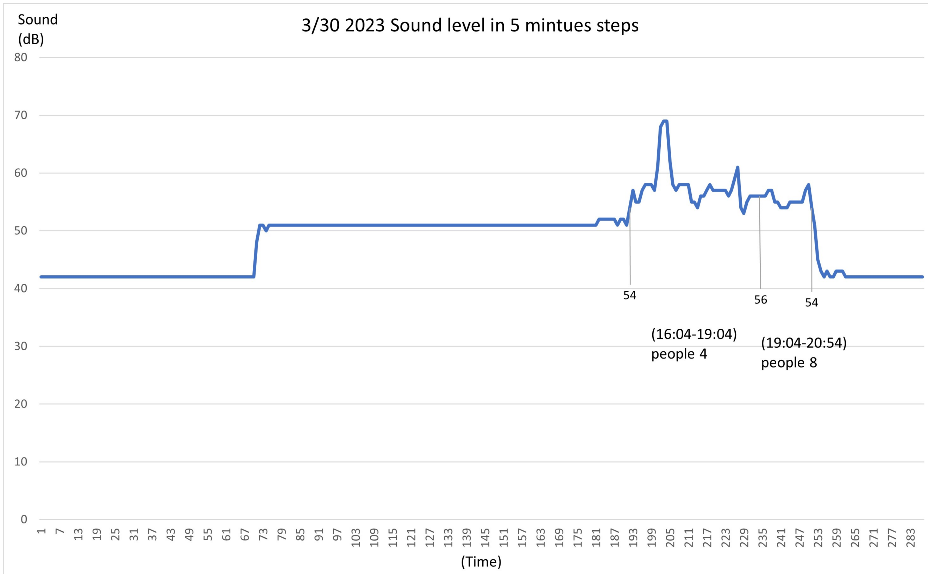

Figure 5.1: Sound in 3/30/2023

However, Table 5.1 reveals that the sound levels in spring, when people are in the room, are comparable to those in winter when the room is empty. This suggests that using the winter schedule experimental data as training data and the spring data as test data could lead to inconsistent test results. The model may not detect people exist because one of environmental data proves the no people in the memory.

Table 5.1: Sound level variation

Date	people	Sound(dB)
1/31/2023	yes	68-74
1/31/2023	no	51-64
3/30/2023	yes	54-60
3/30/2023	no	41-52

In Figure 5.2, we incorporate spring data as an extension of the test data. The model can accurately estimate the first three waves, which represent winter data. However, for the last two waves, the model incorrectly predicts that there are no people in the room, while in reality, the opposite is true.

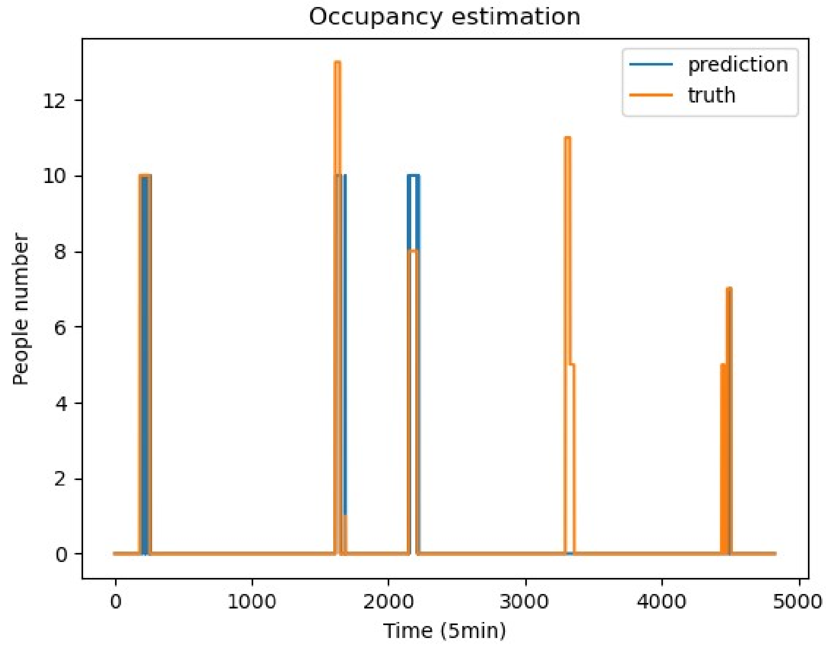

Figure 5.2: limitation in estimation for spring data

This outcome supports our previous assumptions and highlights the seasonal limitations of our experiment. Above all, employing the Attention mechanism in conjunction with the Adamax optimizer can yield optimal performance, even when faced with seasonal constraints.

Chapter 6

Future work

6.1 Model improvement

To enhance the model's performance and address its limitations, we can consider the following three strategies:

1) Collect more data: Gathering additional data from various occupancy levels can help improve performance. By collecting data for more than one year, we may overcome the seasonal limitations.

2) Add more features: Incorporating relevant features, such as outdoor temperature to represent seasons, could potentially mitigate the seasonal limitations.

3) Employ better algorithms: As new and improved algorithms become available in the future, one can apply them to our model to achieve better performance.

Moreover, as we delve deeper into the experiment, we may uncover further methods for refining and optimizing our model. Ongoing research and development can lead to an improved understanding of the problem and the identification of more effective solutions.

6.2 Application

Occupancy data is crucial for various applications, such as controlling lighting and HVAC systems based on room occupancy to conserve energy. My work will concentrate on enhancing our model's performance and exploring its practical applications.

Once our model achieves satisfactory performance, its applications will become increasingly appealing for future endeavors. We plan to integrate our model into a Building Energy Management System (BEMS). One potential platform is the Building Energy Management Open-Source Software (BEMOSS) [51], which has the capability to apply our model to lighting and HVAC control. For instance, by combining our model with BEMOSS, we aim to achieve automatic control of lighting and HVAC systems in classrooms. The model can detect occupancy levels in the classroom, and BEMOSS can adjust lighting brightness or HVAC output to meet the room's requirements. When the room is occupied by more people, we can increase lighting levels and configure the HVAC system to maintain a comfortable environment for students during their studies. Conversely, when the room is unoccupied or has few occupants, there is no need to keep all lights on at full power.

Bibliography

[1] J. Chen and Y. Liu, "Research on energy-saving optimization model based on building energy consumption data," in *2022 International Conference on Machine Learning and Knowledge Engineering (MLKE)*, pp. 173–176, 2022.

[2] M. Kim, N.-s. Kim, Y. Song, and C. S. Pyo, "Residual seq2seq model for building energy management," in *2019 International Conference on Information and Communication Technology Convergence (ICTC)*, pp. 1126–1128, 2019.

[3] E. Efficiency, Z. Dadi, I. Bashmakov, W. Chandler, U. Farinelli, M. Halpeth, N. Jollands, T. Kaiser, J. Laitner, M. Levine, F. Moisan, R. Moss, H.-C. Park, A. Platonova-Oquab, R. Schaeffer, J. Sathaye, J. Siegel, D. Ürge Vorsatz, and E. Worrell, *Realizing the Potential of Energy Efficiency: Targets, Policies, and Measures for G8 Countries.* 01 2007.

[4] F. M. Bhutta, "Application of smart energy technologies in building sector — future prospects," in *2017 International Conference on Energy Conservation and Efficiency (ICECE)*, pp. 7–10, 2017.

[5] A. M. Al-Ghaili, H. Kasim, N. M. Al-Hada, B. N. Jørgensen, M. Othman, and J. Wang, "Energy management systems and strategies in buildings sector: A scoping review," *IEEE Access*, vol. 9, pp. 63790–63813, 2021.

[6] S. Park, S. Seo, B. Lee, J. Byun, and S. Park, "An energy efficient smart led lighting system for building energy management," in *The 18th IEEE International Symposium on Consumer Electronics (ISCE 2014)*, pp. 1–2, 2014.

[7] A. Yanping and Q. Ping, "Air-conditioning energy-saving control strategy at subway station based on mas evolutionary algorithm," in *2015 8th International Conference on Intelligent Computation Technology and Automation (ICICTA)*, pp. 122–125, 2015.

[8] J. Yang, M. Santamouris, and S. E. Lee, "Review of occupancy sensing systems and occupancy modeling methodologies for the application in institutional buildings," *Energy and Buildings*, vol. 121, pp. 344–349, 2016.

[9] Y.-S. Kim and J. Srebric, "Impact of occupancy rates on the building electricity consumption in commercial buildings," *Energy and Buildings*, vol. 138, pp. 591–600, 2017.

[10] Y. Peng, A. Rysanek, Z. Nagy, and A. Schlüter, "Occupancy learning-based demand-driven cooling control for office spaces," *Building and Environment*, vol. 122, pp. 145–160, 2017.

[11] W. Jung and F. Jazizadeh, "Human-in-the-loop hvac operations: A quantitative review on occupancy, comfort, and energy-efficiency dimensions," *Applied Energy*, vol. 239, pp. 1471–1508, 2019.

[12] T. Labeodan, W. Zeiler, G. Boxem, and Y. Zhao, "Occupancy measurement in commercial office buildings for demand-driven control applications—a survey and detection system evaluation," *Energy and Buildings*, vol. 93, pp. 303–314, 2015.

[13] K. Sun, Q. Zhao, and J. Zou, "A review of building occupancy measurement systems," *Energy and Buildings*, vol. 216, p. 109965, 2020.

[14] B. S. Çiftler, S. Dikmese, . Güvenç, K. Akkaya, and A. Kadri, "Occupancy counting with burst and intermittent signals in smart buildings," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 724–735, 2018.

[15] Z. Chen, Y. Wang, and H. Liu, "Unobtrusive sensor-based occupancy facing direction detection and tracking using advanced machine learning algorithms," *IEEE Sensors Journal*, vol. 18, no. 15, pp. 6360–6368, 2018.

[16] O. Dalhoumi, M. Amayri, and N. Bouguila, "A review of neural networks for buildings occupancy measurement," in *2022 IEEE International Conference on Industry 4.0, Artificial Intelligence, and Communications Technology (IAICT)*, pp. 29–35, 2022.

[17] K. Rastogi and D. Lohani, "Iot-based occupancy estimation models for indoor non-residential environments," in *2019 IEEE 16th India Council International Conference (INDICON)*, pp. 1–4, 2019.

[18] C. de Bakker, T. van de Voort, J. van Duijhoven, and A. Rosemann, "Assessing the energy use of occupancy-based lighting control strategies in open-plan offices," in *2017 IEEE 14th International Conference on Networking, Sensing and Control (ICNSC)*, pp. 476–481, 2017.

[19] W. Shen and G. Newsham, "Implicit occupancy detection for energy conservation in commercial buildings: A review," in *2016 IEEE 20th International Conference on Computer Supported Cooperative Work in Design (CSCWD)*, pp. 625–631, 2016.

[20] S. Zemouri, D. Magoni, A. Zemouri, Y. Gkoufas, K. Katrinis, and J. Murphy, "An edge computing approach to explore indoor environmental sensor data for occupancy measurement in office spaces," in *2018 IEEE International Smart Cities Conference (ISC2)*, pp. 1–8, 2018.

[21] M. T. Mulia, S. H. Supangkat, and N. Hariyanto, "A review on building occupancy estimation methods," in *2017 International Conference on ICT For Smart Society (ICISS)*, pp. 1–7, 2017.

[22] K. Hornik, "Approximation capabilities of multilayer feedforward networks," *Neural Networks*, vol. 4, pp. 251–257, 1991.

[23] T. Ekwevugbe, N. Brown, V. Pakka, and D. Fan, "Improved occupancy monitoring in non-domestic buildings," *Sustainable Cities and Society*, vol. 30, pp. 97–107, 2017.

[24] K. Lam, M. Hoeynck, B. Dong, B. Andrews, Y.-S. Chiou, R. Zhang, D. Benitez, and J.-H. Choi, "Occupancy detection through an extensive environmental sensor network in an open-plan office building," *IBPSA 2009 - International Building Performance Simulation Association 2009*, 01 2009.

[25] H. Han, K.-J. Jang, C. Han, and J. Lee, "Occupancy estimation based on co2 concentration using dynamic neural network model," 09 2013.

[26] B. Zhao, H. Lu, S. Chen, J. Liu, and D. Wu, "Convolutional neural networks for time series classification," *Journal of Systems Engineering and Electronics*, vol. 28, no. 1, pp. 162–169, 2017.

[27] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[28] M. Amayri, A. Arora, S. Ploix, S. Bandhyopadyay, Q.-D. Ngo, and V. R. Badarla, "Estimating occupancy in heterogeneous sensor environment," *Energy and Buildings*, vol. 129, pp. 46–58, 2016.

[29] A. Dey, X. Ling, A. Syed, Y. Zheng, B. Landowski, D. Anderson, K. Stuart, and M. E. Tolentino, "Namatad: Inferring occupancy from building sensors using machine learning," in *2016 IEEE 3rd World Forum on Internet of Things (WF-IoT)*, pp. 478–483, 2016.

[30] J. L. Elman, "Finding structure in time," *Cognitive Science*, vol. 14, no. 2, pp. 179–211, 1990.

[31] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, pp. 1735–1780, 11 1997.

[32] F. Conti, A. Pullini, and L. Benini, "Brain-inspired classroom occupancy monitoring on a low-power mobile platform," in *2014 IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 624–629, 2014.

[33] B. S. Totada and S. D. Cabrera, "Detection of people from time-of-flight depth images using a cell-tracking methodology," in *2018 IEEE International Symposium on Signal Processing and Information Technology (ISSPIT)*, pp. 084–089, 2018.

[34] X. Lu, H. Wen, H. Zou, H. Jiang, L. Xie, and N. Trigoni, "Robust occupancy inference with commodity wifi," in *2016 IEEE 12th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, pp. 1–8, 2016.

[35] H. Zou, H. Jiang, J. Yang, L. Xie, and C. Spanos, "Non-intrusive occupancy sensing in commercial buildings," *Energy and Buildings*, vol. 154, pp. 633–643, 2017.

[36] S. S. Shetty, H. D. Chinh, M. Gupta, and S. K. Panda, "User presence estimation in multi-occupancy rooms using plug-load meters and pir sensors," in *GLOBECOM 2017 - 2017 IEEE Global Communications Conference*, pp. 1–6, 2017.

[37] Y. P. Raykov, E. Ozer, G. Dasika, A. Boukouvalas, and M. A. Little, "Predicting room occupancy with a single passive infrared (pir) sensor through behavior extraction," in *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, (New York, NY, USA), p. 1016–1027, Association for Computing Machinery, 2016.

[38] Y.-C. Li, N.-H. Hsieh, and S.-C. Chen, "Assessing the seasonality of occupancy number-associated co(2) level in a taiwan hospital.," pp. 16422–16432, 2019.

[39] H. Rahman and H. Han, "Bayesian estimation of occupancy distribution in a multi-room office building based on co2 concentrations," *Building Simulation*, vol. 11, no. 3, pp. 1996–8744, 2018.

[40] Y. Wei, S. Wang, L. Jin, Y. Xu, and T. Ding, "Indoor occupancy estimation from carbon dioxide concentration using parameter estimation algorithms," *Building Services Engineering Research and Technology*, vol. 43, no. 4, pp. 419–438, 2022.

[41] S. Wolf, D. Caľı, J. Krogstie, and H. Madsen, "Carbon dioxide-based occupancy estimation using stochastic differential equations," *Applied Energy*, vol. 236, pp. 32–41, 2019.

[42] W. Kleiminger, C. Beckel, and S. Santini, "Household occupancy monitoring using electricity meters," in *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, UbiComp '15, (New York, NY, USA), p. 975–986, Association for Computing Machinery, 2015.

[43] D. Chen, D. Irwin, P. Shenoy, and J. Albrecht, "Combined heat and privacy: Preventing occupancy detection from smart meters," in *2014 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, pp. 208–215, 2014.

[44] A. P. Singh, V. Jain, S. Chaudhari, F. A. Kraemer, S. Werner, and V. Garg, "Machine learning-based occupancy estimation using multivariate sensor nodes," in *2018 IEEE Globecom Workshops (GC Wkshps)*, pp. 1–6, 2018.

[45] Z. Chen, C. Jiang, M. K. Masood, Y. C. Soh, M. Wu, and X. Li, *Deep Learning for*

Building Occupancy Estimation Using Environmental Sensors, pp. 335–357. Cham: Springer International Publishing, 2020.

[46] E. Ramanujam, A. Sharma, J. J. Hussian, and T. Perumal, “Improving indoor occupancy estimation using a hybrid cnn-lstm approach,” in *2022 International Conference on Intelligent Controller and Computing for Smart Power (ICICCSP)*, pp. 1–6, 2022.

[47] D. Bahdanau, K. Cho, and Y. Bengio, “Neural machine translation by jointly learning to align and translate,” *CoRR*, vol. abs/1409.0473, 2014.

[48] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, “Attention is all you need,” in *Advances in Neural Information Processing Systems* (I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds.), vol. 30, Curran Associates, Inc., 2017.

[49] J. Devlin, M. Chang, K. Lee, and K. Toutanova, “BERT: pre-training of deep bidirectional transformers for language understanding,” in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)* (J. Burstein, C. Doran, and T. Solorio, eds.), pp. 4171–4186, Association for Computational Linguistics, 2019.

[50] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy, “Hierarchical attention networks for document classification,” in *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, (San Diego, California), pp. 1480–1489, Association for Computational Linguistics, June 2016.

[51] W. Khamphanchai, A. Saha, K. Rathinavel, M. Kuzlu, M. Pipattanasomporn, S. Rahman, B. Akyol, and J. Haack, “Conceptual architecture of building energy management

open source software (bemoss)," in *IEEE PES Innovative Smart Grid Technologies, Europe*, pp. 1–6, 2014.