Predicting the temperature dynamics of scaled model and real-world IoT-enabled smart homes

Jason Ling[†], Sharare Zehtabian[‡], Salih Safa Bacanlı[‡], Ladislau Bölöni[‡] and Damla Turgut[‡]
[†]Department of Computer Science, Pennsylvania State University, Pennsylvania, USA
[‡]Department of Computer Science, University of Central Florida, Orlando, FL, USA
Email: [†]JKL5242@psu.edu, [‡]sharare.zehtabian@knights.ucf.edu, [‡]{sbacanli, lboloni, turgut}@cs.ucf.edu

***Abstract*—Recent advances in IoT sensors and actuators and smart home controllers allow us to collect real-time information about the state of the home and take intelligent actions that maximize the user's goals with respect to comfort, convenience, environmental awareness and cost. While thermal comfort is one of the primary concerns of many users, many homes use a very simple, energy inefficient approach that blankets the home with constant temperature air conditioning. Such systems do not take advantage of more energy efficient and environment friendly natural ways to manage the temperature, such as opening and closing windows, window shades and interior doors. In this paper we develop a deep neural network based model that predicts the temperature in various rooms of the home function of the state of the actuators. We also describe a scaled model of a four room home which allows us to control the doors and windows and collect data using IoT devices. We train and validate our temperature models on both data collected from the scaled model as well as from publicly available datasets from two real-world smart homes.**

I. Introduction

Cyber-physical systems (CPS) integrate computational and physical processes with communication capabilities to interact with the physical world [1]. Although the devices in a smart home are interconnected, they do not work together to adjust comfort parameters, such as a heater or fan, while simultaneously capitalizing on environmental friendly opportunities, such as opening or closing doors and windows. To take advantage of these economical and environmental opportunities, integrating a smart home as a CPS is a desired choice. To keep the comfort level of a smart home ideal, while lowering energy costs, it is important to be able to predict temperature based on previous and current states of a smart home.

Monitoring the outside and inside weather conditions such as temperature and humidity make it possible to predict them based on other current and past measurements. This prediction system can be integrated into the air conditioner for a better energy efficiency by deciding to increase or decrease the temperature beforehand. This also allows the air conditioner to work in a more stable manner, less prone to usage spikes due to unexpected temperature and humidity differences.

One of the challenges of predicting the indoor temperature is the lack of extensive datasets based on which a predictor can be trained. The predictor will clearly need to be specialized to the specific architecture and environment of each home. Even if we collect data from a real-world smart home, this data will be sparse, only considering the specific weather conditions and user action that appear during that specific time period.

In this paper, we describe a technique for predicting the temperature of a smart home using the outside and inside, current and past weather values. To facilitate data collection, we created a scaled-down model home. We collected data by experimenting with remotely controlled doors and windows in various configurations. We collected temperature and humidity data for each room as well as the immediate environment outside the home. In another set of experiments, we also used an academic smart home dataset, to predict the indoor temperature at a specific timestamp from indoor and outdoor historical temperature values. The proposed prediction engines were based on two variants of deep neural networks: fully connected networks and long short term memory (LSTM) networks.

The rest of the paper is organized as follows. We present the related work in Section II. We provide a detailed description of our smart home prototype and methods in Section III. We evaluate our deep learning model on collected and real-world data in Section IV, and finally conclude in Section V.

II. Related Work

Machine learning methods and neural networks have been used to forecast temperature in buildings.

Lin et al. [2] examined the correlation between smart home features and indoor air quality. They collected data in two smart home testbeds and analyzed the impact of the overall smart home behavior and also individual groups of features of the smart home on indoor air quality. To quantify the relationship between these features and the air quality, they used machine-learning methods such as Random Forest, Linear Regression, and Support Vector Regression. The authors found that the temperature features are more frequently selected than other specific activities occurring in the smart homes.

Pubill et al. [3] harvesting energy from indoor artificial light to achieve energy neutral wireless sensor network devices. Lee et al. [4] presented a simulator that creates a 3D virtual smart house and an intelligent agent that can interact with the house by executing different behaviours based on different motivations. Then, they recorded environmental information including temperature via virtual sensors in the house. In their smart house, temperature changes based on interactions between the agent and the house. For example, they increase the temperature when a heater is turned on, a window is closed, or the agent engages in sports or cooking. These long-term simulated data can help the researchers to reduce the testing costs associated with the evaluation of the smart home architecture.

Serra et al. [5] proposed an energy scheduling method that minimizes energy consumption for a specific time interval according to the user's preferences. Coming up with a good model for prediction can help improve planning for different purposes purposes as well. Jin et al. [6] used prediction to optimize energy consumption alongside user comfort in a smart home. They used a pre-collected dataset for achieving maximum user comfort with minimum energy consumption.

Mateo et al. [7] used machine learning methods to predict temperature of buildings. This study, however, does not apply it to smart home and does not consider door or window actuators. Chen and Irwin [8] proposed a model where they can guess the location of a place by analyzing the weather data. The authors found that anonymous datasets might be analyzed to get the source location of the data. While not directly about temperature prediction in a smart home, this work is notable because of the analysis on weather and solar energy generation of a house dataset.

Teich et al. [9] use neural networks in a smart home environment, there is no use of door and window actuators. There will be more efficiency with lowering energy costs if windows and doors are controllable; therefore, knowing the state of doors and windows is important data for training a neural network.

Serra et al. in [10] introduced a collaborative platform to analyze IoT data which relies on machine learning techniques and centralized and global management of virtual networking and computing. Kim et al. [11] compared different machine learning models for detecting actions of people in a house. Cook et al. [12] proposed a prediction algorithm using a back propagation network to predict the actions of the person in a smart house based on past history.

We have created a smart home prototype that has actuators that control doors and windows, while keeping track of temperature and humidity on the rooms inside the prototype. We also trained a fully connected neural network and long short term memory network with a smart home data set to show that it applies to real world applications.

III. Smart home prototype

Predicting the temperature in different areas of a smart home provides useful information for future planning and optimizing energy consumption. We use deep learning techniques to achieve this goal. We design a smart home prototype with different rooms and sensors to collect data. Furthermore, in order to evaluate the effectiveness of our method we apply the same method on two real-world datasets. In the following, we discuss the details of each step of our method.

A. Scaled-down Smart Home

To study the relationship between internal and external factors on the temperature of a home, we created a scaled down model home instrumented with IoT sensors and actuators. The implemented model, shown in Fig. 1, contains four rooms with the following window and door configurations:

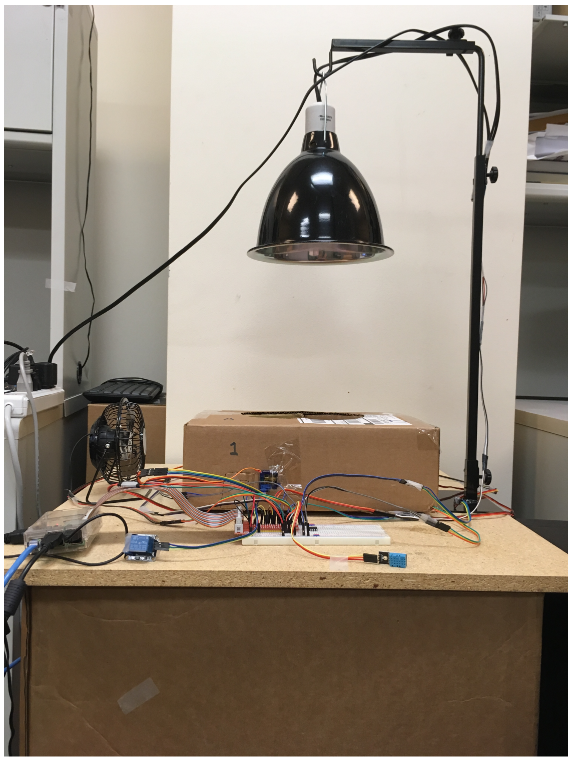


Fig. 1. The scaled-down model home. Two doors and two windows are actuator controlled. The remove controllable heat lamp models the effect of sunlight, while the controllable fan models the effect of the wind.

- Room A: a permanently open window and a controllable door.
- Room B: a controllable door and controllable window.
- Room C: a controllable window and a permanently closed door.
- Room D: a permanently closed door and a permanently closed window.

SG90 Micro Servo motors are used as window and door actuators to adjust the door and window positions. To measure temperature and humidity, the DHT11 sensors are placed inside the rooms and outside of the smart home. In order to simulate the rise and fall of temperature, the SRD-05VDC-SL-C relay is used to control a heating lamp and a fan.

B. Data Collection

We collected data from our smart home prototype for 38 hours by triggering the actuators and sensors connected to the controllable doors and windows. Every 30 seconds, we randomly select the target door or window to send commands. Similarly, we chose our command (open or close) to the motors on a random basis.

The actuators and sensors are triggered through randomly generated inputs. Consequently, as they are triggered, the state of

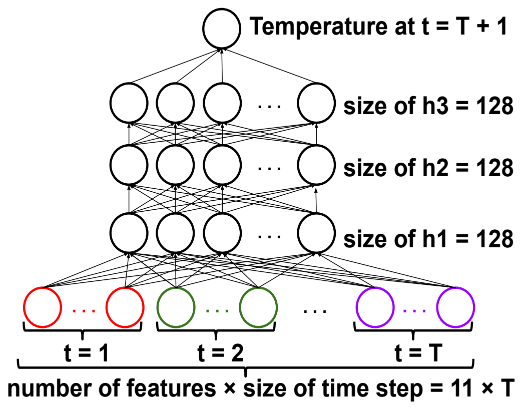


Fig. 2. Architecture of our temperature prediction model with Fully connected neural network

the smart home is stored for further use within a neural network to predict the temperature for the next timestep. We store the position of a door and window through binary representation; 0 and 1 representing closed and open positions respectively. We also store the status of the heating lamp and fan as binary. The temperature and humidity values for each room are stored as a floating point number given by the DHT11 sensor. Through this method, we collected approximately 4,500 timesteps.

C. Real-world Dataset

To evaluate the generalization of our approaches for predicting the temperature in different smart homes, we used the UMassTraceRepository dataset [13] collected in two real homes (Home A and Home B). The data types collected for Home A and Home B and their description are as follows:

- Inside humidity: in-house humidity as percentage
- Outside humidity: outside house humidity as percentage
- Outside temperature: as Fahrenheit
- Wind speed: in miles/hour
- Wind direction degrees: in radians
- Wind gust: sudden increase in wind speed in miles/hour
- Wind gust direction degrees: in radians
- Wind chill: felt temperature related with wind speed
- Heat index: felt temperature related with humidity

D. Fully Connected Neural Network

We trained the real-world dataset with a Fully Connected Neural Network (FC-NN) (See Fig 2). In FC-NN, each neuron in a layer is connected to every neuron in the next layer. The backpropagation algorithm allows us to train FC-NN to learn the patterns in data by optimizing a loss function with respect to its parameters.

E. Long Short Term Memory

Next, we used a recurrent neural network, more specifically, a Long Short-Term Memory (LSTM) network [14] to train the real-world dataset. LSTMs are well known for their ability to learn complex, long term sequential patterns in data and are widely used. They also have the ability to handle different input sizes, although for our purpose we fix the input size to T timesteps.

The architecture of the LSTM configuration we used is shown in Fig. 3. The input features are the state of the doors, the state of the windows, the state of the fan, the state of the heating lamp, and the temperature and humidity of each room at timesteps from $t = 1$ to $t = T$ fed to the network sequentially and the output is the temperature of each room at timestep $t = T + 1$.

F. Implementation Details

We implemented the hardware controllers in Python and utilized multiple libraries to obtain complete functionality of the smart home. We use the RPi.GPIO packages, which is used to access the input/output (IO) pins within the Raspberry Pi. The DHT11 sensors make use of the Adafruit Unified Sensor Library to collect temperature and humidity data. The DHT11 sensors operate within a temperature range of 0°C to 50°C, which is within the range of our experiments. They also provide a ±5 RH humidity accuracy range and a ±2°C temperature accuracy range, which is respectable [15].

The neural network estimators were implemented using the TensorFlow open source software library [16]. We use scikit-learn to shuffle the data for training and set a constant random seed for reproducibility [17]. The accuracy is determined by examining the error of the predicted temperature versus the actual temperature; if the error is less than $\epsilon = \|y - \hat{y}\|$, where y is the actual temperature and $\hat{y}$ is the predicted temperature at timestep $T + 1$. If $\epsilon < 0.2$ in our experiments, then we declare the prediction to be accurate. Due to the sizable amount of data provided, we utilized Google's Colaboratory project for the training of the neural networks on our own data due to its GPU optimization and accelerated computation time, and one GeForce GTX 1080 GPU for training on UMassTraceRepository dataset.

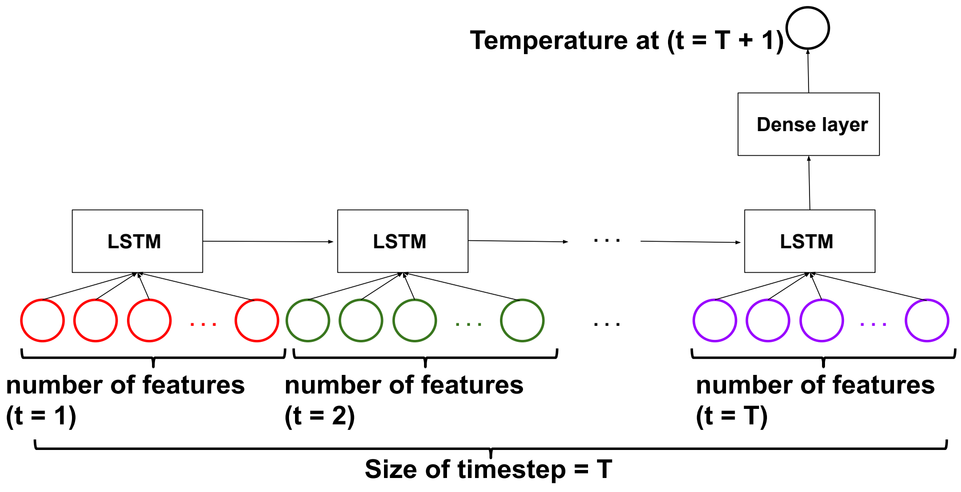
IV. Experiments

We have three sets of experiments to test the generalization of the method and compare LSTM and FC-NN algorithms. Experiment one involves applying LSTM on the dataset collected with our smart home prototype (which is described in section III). In the second and third experiments, we apply FC-NN and LSTM algorithms on UMassTraceRepository datasets respectively. We provide details on each set of experiments in the following subsections.

A. LSTM on collected dataset in smart home prototype

The experiments are performed on dataset collected from our smart home prototype. We fed the training data into the LSTM. We kept track of the temperature of each room and determined each room's accuracy separately.

The network consists of one LSTM layer and we used 80% of the data for training, and 20% of the data for testing. We kept track of the order of the inputs in order to model the problem as a time series problem. The activation function we used is tanh.



Temperature at (t = T + 1)
Dense layer
LSTM
LSTM
LSTM
number of features (t = 1)
number of features (t = 2)
number of features (t = T)
Size of timestep = T

Fig. 3. Architecture of our temperature prediction model with LSTM

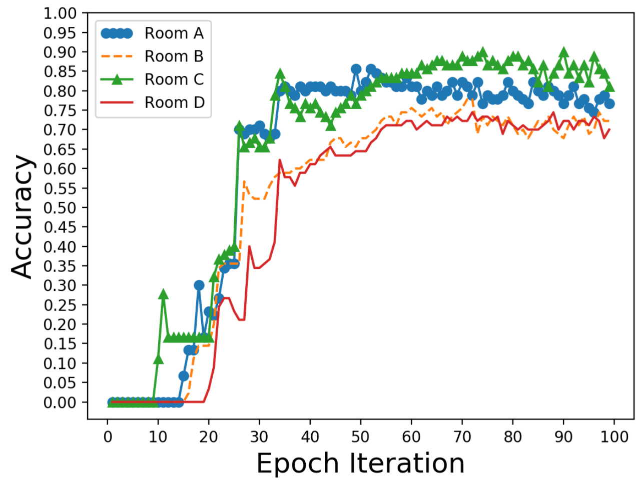


Fig. 4. Prediction accuracy on test dataset collected from smart home prototype

To train our model, we used batch training with each batch being size of 16. The number of timesteps we kept track of at a single time is 10.

As it can be seen in Fig. 4 that the accuracy of each room increases significantly during epochs 5 to 20. The graph shows that the LSTM is a good model for temperature prediction for a smart home environment.

B. FC-NN on Home A and Home B

To organize our dataset, we separate the dataset into a training set and testing set. 80% of the entire dataset is used for training while 20% is used for testing. We use batch training, each layer uses Rectified Linear Units (ReLU) as the activation function. As the neural network is trained, we kept track of the cost using the Huber Loss function. For the optimization algorithm, we used the Adam optimizer.

As you can see in Fig. 5, the accuracy converges on training for both houses.

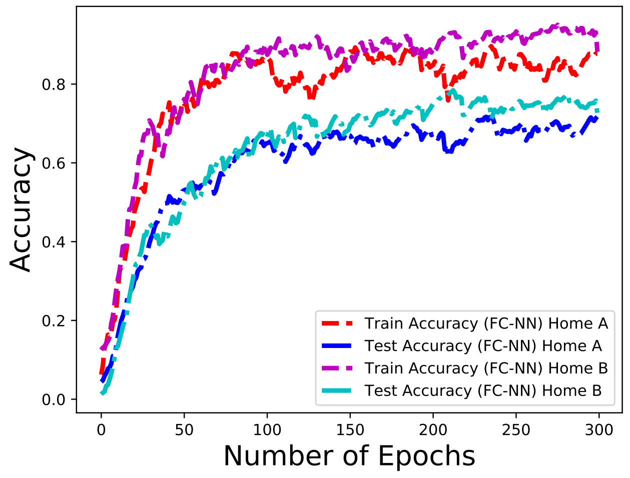


Fig. 5. Accuracy on train and test sets based on FC-NN for Home A and home B (Smoothed)

In training neural networks, hyperparameters play an important role. We compare the accuracy of FC-NN on different hyperparameters in table I for home A and table II for home B.

C. LSTM on Home A and Home B

Similar to FC-NN, we divide the dataset into a training set and testing set. The LSTM accuracy on train and test sets for home A and home B are shown in Fig. 6.

Tables III and IV present the results which are on test data for Home A and Home B. We examined different settings for LSTM network. Based on these experiments, we can fix our hyperparameters and use 64 for size of the LSTM, 10 for the length of input sequence, and 16 for the batch size.

No. of hidden layers	hidden layers	Batch size	Accuracy
1	64	16	0.6214
2	64	16	0.7737
3	64	16	0.7934
5	64	16	0.7748
3	8	16	0.7533
3	128	16	**0.8101**
3	512	16	0.7412
3	64	2	0.6571
3	64	64	0.7656
3	64	256	0.5181

TABLE I
FULLY CONNECTED NN APPROACH RESULTS WITH DIFFERENT SETTINGS ON HOME A TEST DATA (MAXIMUM ACCURACY)

No. of hidden layers	hidden layers	Batch size	Accuracy
1	64	16	0.6327
2	64	16	0.8029
3	64	16	0.8443
5	64	16	0.8298
3	8	16	0.8228
3	128	16	**0.8464**
3	512	16	0.7282
3	64	2	0.7717
3	64	64	0.7448
3	64	256	0.3589

TABLE II
FULLY CONNECTED NN APPROACH RESULTS WITH DIFFERENT SETTINGS ON HOME B TEST DATA (MAXIMUM ACCURACY)

Size of LSTM	Input sequence length	Batch size	Accuracy
8	10	16	0.8165
64	10	16	**0.8661**
512	10	16	0.8087
64	2	16	0.7919
64	4	16	0.7983
64	16	16	0.8409
64	50	16	0.8365
64	10	2	0.8596
64	10	64	0.8634
64	10	256	0.8000

TABLE III
LSTM APPROACH RESULTS WITH DIFFERENT SETTINGS ON HOME A TEST DATA (MAXIMUM ACCURACY)

Size of LSTM	Input sequence length	Batch size	Accuracy
8	10	16	0.8165
64	10	16	**0.8859**
512	10	16	0.7116
64	2	16	0.8145
64	4	16	0.7392
64	16	16	0.7940
64	50	16	0.8750
64	10	2	0.8734
64	10	64	0.8829
64	10	256	0.8609

TABLE IV
LSTM APPROACH RESULTS WITH DIFFERENT SETTINGS ON HOME B TEST DATA (MAXIMUM ACCURACY)

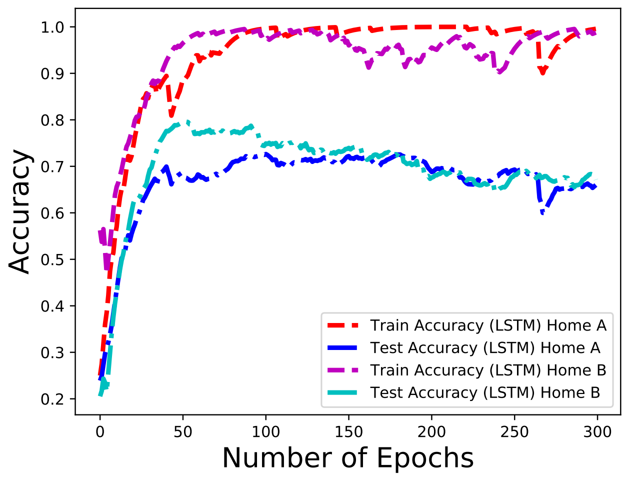


Fig. 6. Accuracy on train and test sets based on LSTM for Home A and Home B (Smoothed)

D. LSTM versus FC-NN

We compare LSTM and FC-NN for the temperature predictions. While both methods can achieve good performance, FC-NNs are expensive in terms of memory (number of parameters) and computation (connections). In addition, generally networks which have large number of parameters, have slower training time and higher chances of overfitting.

Our results show that LSTM is a better fit as it reaches to desired performance sooner (See Fig. 7). They are also getting better results in terms of accuracy. We believe that the ability to handle sequential data while using fewer parameters is the reason for this result which in turn makes LSTM a better choice for our purpose.

Fig. 7 shows how the accuracy on train set for home A and home B differ from each other.

Similar to the LSTM based prediction results on collected data from scaled-down Smart Home which is shown in Fig. 4, the graphs in Fig. 7 show that an LSTM is a good model for temperature prediction. We can see that LSTM converges after a few iteration and the training accuracy is 1.0, while FC-NN is not as fast as LSTM in terms of convergence.

V. CONCLUSION

In this paper, we described a technique for predicting the temperature in an IoT augmented smart home based on historical sensor readings of indoor and outdoor data, as well as the configuration of open doors and windows. We have evaluated the approach on both a scaled-down model home as well as a publicly available smart home dataset. We found that both fully connected and LSTM neural networks are applicable, but that latter performs more accurately and reaches the desired performance sooner. Future work includes extending this work to implement smart home temperature controllers that can maintain the desired temperature range with lower energy usage and environmental impact.

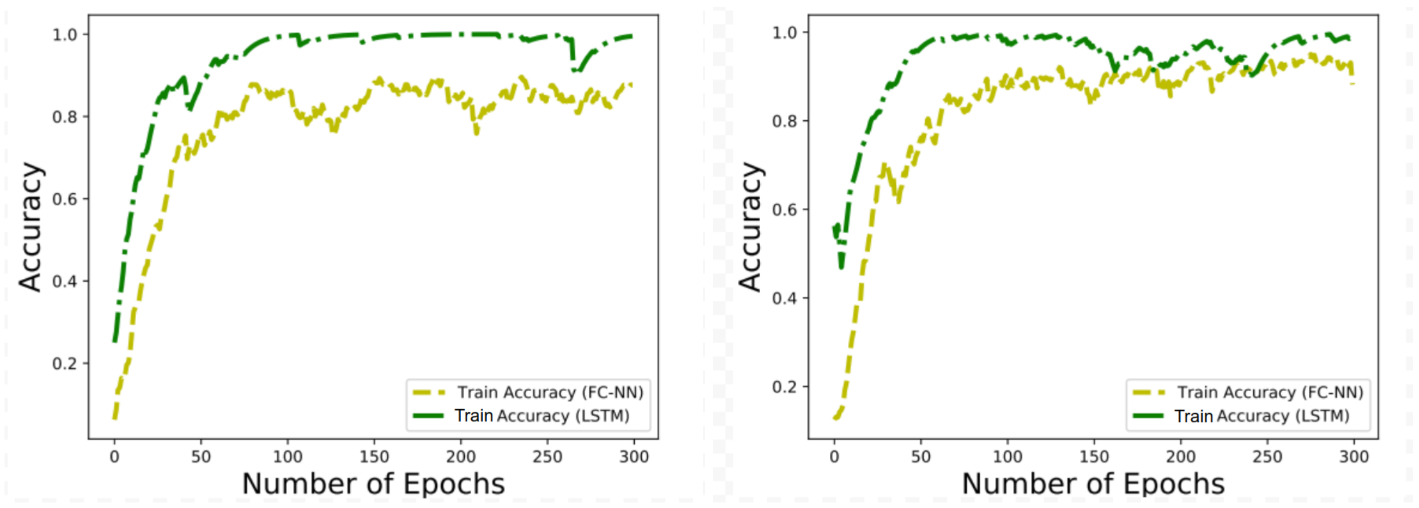


Fig. 7. Accuracy on train sets based on FC-NN and LSTM for Home A (left) and Home B (right)

Acknowledgement

The support for this work was provided by the National Science Foundation REU program under Award No. 1560302. Any opinions, findings, and conclusions and recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References

[1] T. Sanislav and L. Miclea, "Cyber-physical systems - concept, challenges and research areas," *Control Engineering and Applied Informatics*, vol. 14, pp. 28–33, 01 2012.

[2] B. Lin, Y. Huangfu, N. Lima, B. Jobson, M. Kirk, P. OKeeffe, S. Pressley, V. Walden, B. Lamb, and D. Cook, "Analyzing the relationship between human behavior and indoor air quality," *Journal of Sensor and Actuator Networks*, vol. 6, no. 3, p. 13, 2017.

[3] D. Pubill, J. Serra, and C. Verikoukis, "Harvesting artificial light indoors to power perpetually a wireless sensor network node," in *2018 IEEE 23rd International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD)*, 2018, pp. 1–6.

[4] W. Lee, S. Cho, P. Chu, H. Vu, S. Helal, W. Song, Y.-S. Jeong, and K. Cho, "Automatic agent generation for iot-based smart house simulator," *Neurocomputing*, vol. 209, pp. 14–24, 2016.

[5] J. Serra, D. Pubill, A. Antonopoulos, and C. Verikoukis, "Smart HVAC control in IoT: Energy consumption minimization with user comfort constraints," *The Scientific World Journal*, vol. 2014, 2014.

[6] W. Jin, I. Ullah, S. Ahmad, and D. Kim, "Occupant comfort management based on energy optimization using an environment prediction model in smart homes," *Sustainability*, vol. 11, no. 4, p. 997, 2019.

[7] F. Mateo, J. J. Carrasco, A. Sellami, M. Milln-Giraldo, M. Domnguez, and E. Soria-Olivas, "Machine learning methods to forecast temperature in buildings," *Expert Systems with Applications*, vol. 40, no. 4, pp. 1061 – 1068, 2013.

[8] D. Chen and D. Irwin, "Weatherman: Exposing weather-based privacy threats in big energy data," in *Proc. of 2017 IEEE International Conference on Big Data*, 12 2017, pp. 1079–1086.

[9] T. Teich, F. Roessler, D. Kretz, and S. Franke, "Design of a prototype neural network for smart homes and energy efficiency," *Procedia Engineering*, vol. 69, pp. 603 – 608, 2014, international Symposium on Intelligent Manufacturing and Automation.

[10] J. Serra, L. Sanabria-Russo, D. Pubill, and C. Verikoukis, "Scalable and flexible IoT data analytics: when machine learning meets SDN and virtualization," in *2018 IEEE 23rd International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD)*, 2018, pp. 1–6.

[11] E. Kim, S. Helal, and D. Cook, "Human activity recognition and pattern discovery." *IEEE Pervasive Computing*, vol. 9, no. 1, pp. 48 – 53, 2010.

[12] D. J. Cook, M. Youngblood, E. O. Heierman, K. Gopalratnam, S. Rao, A. Litvin, and F. Khawaja, "Mavhome: an agent-based smart home," in *Proceedings of the First IEEE International Conference on Pervasive Computing and Communications, 2003. (PerCom 2003).*, March 2003, pp. 521–524.

[13] S. Barker, A. Mishra, D. Irwin, E. Cecchet, P. Shenoy, and J. Albrecht, "Smart*: An open data set and tools for enabling research in sustainable homes," in *ACM Proceedings of the 2012 Workshop on Data Mining Applications in Sustainability (SustKDD)*, 2012.

[14] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[15] W. W. Gay, *Experimenting with Raspberry Pi*. Apress, 2014, ch. DHT11 Sensor, pp. 1–13.

[16] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/

[17] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.