ICCEPM 2022
The 9th International Conference on Construction Engineering and Project Management
Jun. 20-23, 2022, Las Vegas, NV, USA

Blind Drift Calibration using Deep Learning Approach to Conventional Sensors on Structural Model

Jacob Kutchi[1], Kendall Robbins[2], David De Leon[3], Michael Seek[4], Younghan Jung[5]*, Lei Qian[6], Richard Mu[7], Liang Hong[8], Yaohang Li[9]

[1] *Department of Computer Science, Old Dominion University, Norfolk, VA 23529, USA,* E-mail address: jkutc001@odu.edu
[2] *Department of Modeling and Simulation, Old Dominion University, Norfolk, VA 23529, USA,* E-mail address: krobb001@odu.edu
[3] *Department of Engineering Technology, Old Dominion University, Norfolk, VA 23529, USA,* E-mail address: ddele006@odu.edu
[4] *Department of Engineering Technology, Old Dominion University, Norfolk, VA 23529, USA,* E-mail address: mseek@odu.edu
[5] *Department of Construction, Seminole State College, 104 AAA Dr, Heathrow, FL, 32746, USA,* E-mail address: jungy@seminolestate.edu
[6] *Department of Computer Science, Fisk University, Nashville, TN 37208, USA,* E-mail address: lqian@fisk.edu
[7] *TIGER Institute- Advanced Materials, Tennessee State University, Nashville, TN 37209, USA,* E-mail address: rmu@tnstate.edu
[8] *Department of Electrical & Computer Engineering, Tennessee State University, Nashville, TN 37209, USA,* E-mail address: lhong@tnstate.edu
[9] *Department of Computer Science, Old Dominion University, Norfolk, VA 23529, USA,* E-mail address: yaohang@cs.odu.edu

Abstract: The deployment of sensors for Structural Health Monitoring requires a complicated network arrangement, ground truthing, and calibration for validating sensor performance periodically. Any conventional sensor on a structural element is also subjected to static and dynamic vertical loadings in conjunction with other environmental factors, such as brightness, noise, temperature, and humidity. A structural model with strain gauges was built and tested to get realistic sensory information. This paper investigates different deep learning architectures and algorithms, including unsupervised, autoencoder, and supervised methods, to benchmark blind drift calibration methods using deep learning. It involves a fully connected neural network (FCNN), a long short-term memory (LSTM), and a gated recurrent unit (GRU) to address the blind drift calibration problem (i.e., performing calibrations of installed sensors when ground truth is not available). The results show that the supervised methods perform much better than unsupervised methods, such as an autoencoder, when ground truths are available. Furthermore, taking advantage of time-series information, the GRU model generates the most precise predictions to remove the drift overall.

Key words: Deep Learning, Blind Drift Calibration, Sensors, Structure Model

1. INTRODUCTION

Sensor networks are usually composed of a group of small and inexpensive sensors capable of measuring, gathering, processing, and communicating. Sensor networks are designed to monitor the environment and structural health, collect and process the sensing data, and transmit the information to end-users, typically consisting of tens to hundreds or even thousands of sensors. For instance, civil structures generally are large and very complex. An extensive network of sensors is required to assess their condition accurately. Furthermore, most structural health monitoring systems are customized for particular structural elements and, therefore, not easily scalable for entire structures. However, many critical applications of sensor networks have been, including remote sensing, environment monitoring, and surveillance.

Considering a scenario where a set of sensors are deployed to monitor a field of interest in the long term, from these measurements, physical quantities, such as pressure, temperature, and substructures change over time, can be computed. However, certain element defects, such as the fatigue change of the steel shapes, the deformation of plastic components, the over-heating of the battery, etc., can lead to inaccurate measurement readings. Such measurement errors worsen over time, known as the sensor drift problem. In practice, it is extremely time-consuming and requires a lot of manpower to unmount and re-calibrate the deployed sensors against the ground truth. This often even becomes impractical in many situations. Calibrating the sensors without the ground truth data is referred to as the blind calibration problem [1]. The blind drift calibration problem can be generalized for sensor network applications in many systems. Successfully calibrating deployed sensors without ground truth can significantly reduce the maintenance cost and workforce needs.

Clearly, if there is only one sensor, it is impossible to calibrate it if the ground truth is unavailable. Thus, the rationale behind blind drift calibration is to find an alternative calibration reference based on the observed (drifted) measurements from the other sensors. The classical blind calibration methods [2,3,4,5] rely on pre-defined rules for feature extraction and/or application-related assumptions, such as linearity of data space or sparse drift.

Assuming that the N sensors are deployed to an environment with signal sources in a space with S true dimensions, provided $S < N$, the drift-free measurements lie in a low-rank S-dimensional signal subspace. The signal null subspace is a complement of the signal space, which is fully driven by the sensor drifts, [6], which is also low-dimensional. Given the measuring $Y = X + D$, where X is the actual data and D is the drift, then, there exists a projection function $P(\cdot)$ projecting the measurement Y onto the signal null subspace such that

$$P(Y) = P(X + D) = P(D) \tag{1}$$

The importance of the projection function $P(\cdot)$ is that it eliminates the unknown ground truth signals and obtains a projected observation of the drift. Recovering from the projected drift allows us to estimate the drift and then the drift-free measurements. [1] assumed a linear project $P(\cdot)$ and demonstrated success in calibrating drift in wireless sensor networks. However, this approach is limited by the linearity assumption, which can only recover some but not all drifted sensors. The recent advance in deep learning methods [7] allows us to avoid the linearity limitation and obtain recovery with potentially higher accuracy for broader applications

The objective of this work is to investigate deep learning architectures and algorithms for the blind drift calibration problem in more practical and complex experimental setups. In this paper, we design a structure lab testbed to model structural components and systems in bridges or building structures, which is an ideal testbed to mimic the real-life sensing environment with drifts. The structure lab testbed is set up with two sets of sensors in every position – one is timely calibrated, providing the ground truth data while the other is uncalibrated. The actuators applied to the structures are used to generate drift patterns. We implement several deep learning algorithms,

including autoencoder [8], FCNN, LSTM [9], and GRU [10], for blind drift calibration. We then compare the performance of these deep learning blind drift calibration methods on real experimental senor data from the structure lab testbed.

2. STRUCTURE LAB TESTBED

Data collection from the structure lab testbed is orchestrated using Micro-Measurements StrainSmart Data Acquisition Software. The Structure Lab at Old Dominion University is equipped with a Micro-Measurements System 7000 (Figure 1), which is utilized to collect data for use in training the deep learning algorithms. To generate the data for collection, Micro-Measurements CEA series, 120 Ohm, 240UZA strain gauge sensors are attached to the structure to measure strain over time.

Figure 4. Micro-Measurements System 7000 (Micro Measurements, 2021)

Based on Old Dominion University's Structure Laboratory, the testbed design implements three main components. The structure itself, as shown on the right in Figure 2, is modeled after a truss bridge design seen commonly in modern roadway infrastructure. The structure made of steel is composed of eight stacked segments, each measuring 6" by 6" by 12". To simulate structural load, a simple hydraulic load cell is affixed to the base of the structure. The top of the structure is held in place by a steel assembly sturdy enough such that the structure can buckle long before the assembly can budge. Lastly, to measure strain on the structure due to force applied by the load cell, 32 strain gauges are affixed to various points all over the structure. A diagram of gauge placement is shown in Figure 3. Effectively, only 16 strain gauges are measuring strain throughout the structure as gauges are arranged in pairs. One gauge in the pair would be calibrated often to act as the ground truth measurement of the pair. The other gauge remains uncalibrated to allow drift to occur.

Figure 5. A segment of the Structure Lab Structure

For our blind drift calibration testing, long periods of constant sensor measuring are required for sensor drift to start occurring. To account for this, test runs are conducted weekly. Initially, all 32 strain gauges are zeroed and calibrated while the structure had no load applied to it. The data

acquisition is initiated and then a constant load is applied to the structure to introduce a constant strain to be measured. Measurements are recorded at a rate of 1 reading per second from each strain gauge and are recorded for a continuous 7 days. The strain gauges measure strain on the structure which is then converted into voltage units and recorded. After 7 days, sensor measurements are halted, and data is downloaded and parsed for use in training the deep learning algorithms. Then, the ground truth sensor subset is calibrated, and the gauge measurement is initiated again. By not calibrating the other half of the sensor pairings, drift on these sensor readings will continue to grow as each weekly test concludes, producing ideal training data for our use.

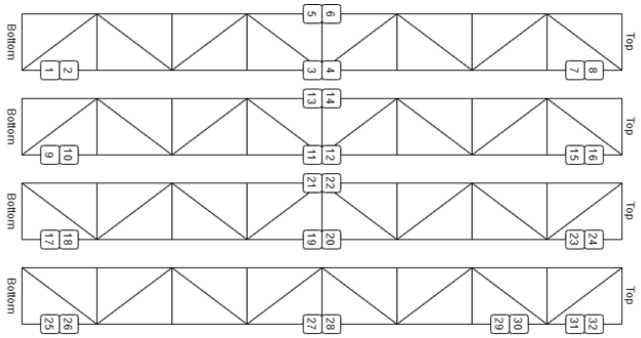

Figure 6. Rotated Strain Gauge Layout on Structure Faces

3. DEEP LEARNING-BASED BLIND DRIFT CALIBRATION

We compare four deep learning architectures in their ability to solve the blind drift calibration problem, an autoencoder, a fully connected neural network (FCNN), a long short-term memory (LSTM), and a gated recurrent unit (GRU).

The input of each model is a 16 x 1 vector of sensor readings at a single timestep, where each entry represents a reading from a gauge sensor, and the desired outputs are the calibrated sensor readings. We split our data into training and testing sets using sensor data of the first 200,000 time steps as the training set and the remaining 128,264 time steps as our testing set. We decide to use a large, long-term test set to test our machine learning abilities to produce calibrated outputs as the drift increased. To preprocess the sensor readings for input to our networks, the training and testing sets are normalized by subtracting the mean and dividing by the standard deviation of all sensor readings in the training set. The autoencoder is trained using unsupervised learning with the uncalibrated sensor readings as input as well as output, where the mean squared error (MSE) is used as the loss function. The FCNN, LSTM, and GRU are trained using supervised learning with the uncalibrated sensor readings as input to the networks and MSE loss is computed between the model's output and the ground truth sensor readings. Each of our networks is trained for 400 epochs using the Adam optimizer [11] to minimize the MSE loss and the best model with the smallest validation error is saved.

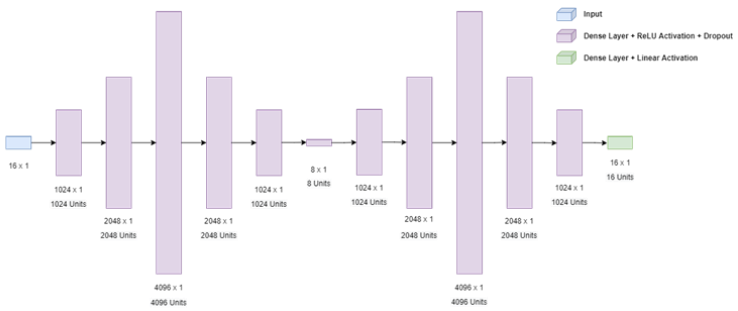

Figure 4. The architecture of the autoencoder neural network for blind drift calibration.

An autoencoder learns a dimensionally reduced representation of input samples using an encoder, then uses a decoder to convert the dimensionally reduced representation back into the original input. The goal of our autoencoder is to learn a latent space to remove drift and recover drift free measurements. The autoencoder architecture we developed is a twelve-layer neural network (Figure 4). The twelve layers are dense, fully connected layers. The first six layers are the encoder portion of the autoencoder. The encoder first increases the dimensions and then decreases the dimensions into the latent space. The sixth layer's output is the final encoding into the latent space. The final six layers are the decoder portion of the autoencoder, converting the latent space back into the output dimensions. As with the encoder, the decoder increases the dimensions and then decreases the dimensions in the subsequent layers into the output dimensions. Each of the dense layers uses a rectified linear unit (ReLU) activation function except for the final dense layer, which uses a linear activation function. Each layer except for the final layer is followed by a dropout layer with a dropout rate of 0.4 to prevent overfitting.

A fully connected neural network uses a fully connected architecture between layers where each neuron in one layer is connected to each neuron in the following layer. When trained using supervised learning, an FCNN can learn internal relationships to convert inputs into target outputs. The FCNN we developed uses a similar architecture as the autoencoder but uses a much larger sixth layer with 512 units. Unlike autoencoder, the training of FCNN is supervised using the calibrated ground truth sensor data (Figure 5).

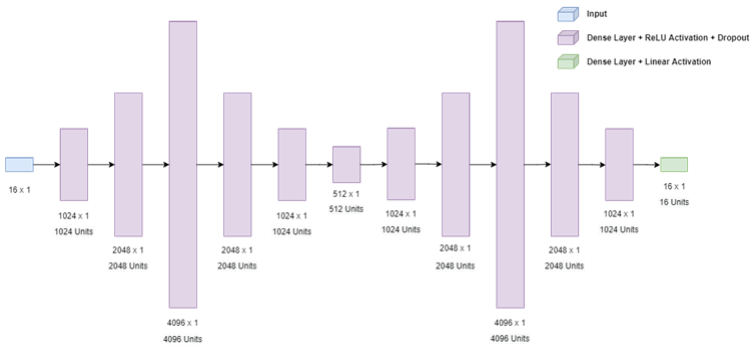

Figure 5. The architecture of a fully connected neural network (FCNN) for blind drift calibration

An LSTM is a form of recurrent neural network which uses feedback connections to pass data sequentially through each cell. Compared to FCNN, which only considers the input of the current time step, LSTM takes advantage time series information to remove drift. Each cell has memory mechanisms that enable the network to remember values over time intervals. The cells operate on data sequentially in time steps with a cell for each time step. Each cell takes the current time step value, the short-term memory state from the previous cell, and the cells' long-term memory state as input. LSTM's cells have three gates an input gate, forget gate, and an output gate. The input gate

decides the information to store in its long-term memory state, the forget gate decides which information to keep or discard, and the output gate uses the current time step, previous cells' short term memory, and current cell's new long-term memory to compute the short-term memory to pass to the sequential cell. In our LSTM, we use bidirectional LSTMs, which operate on the input from the beginning to the end and then from the end to the beginning, concatenating the two outputs. The LSTM architecture we developed is a seven-layer neural network (Figure 6). The seven layers are three stacked bidirectional LSTM layers followed by an average pooling and a flattening layer, then followed by three dense layers. Each LSTM layer returns the full sequences generated by each unit, creating a two-dimensional output. Each of the stacked bidirectional LSTM layers has the same number of units keeping the output dimensions the same as it passes through each layer. The fourth layer is an average pooling and flattening layer to convert the two-dimensional output of the LSTMs into a one-dimensional output. The final three dense layers convert the output of the LSTM into the output dimensions. Each bidirectional LSTM layer uses the standard hyperbolic tangent activation function and sigmoid recurrent activation function. The dense layers use linear activation functions. A dropout layer follows each layer with a dropout rate of 0.4 except for the final layer.

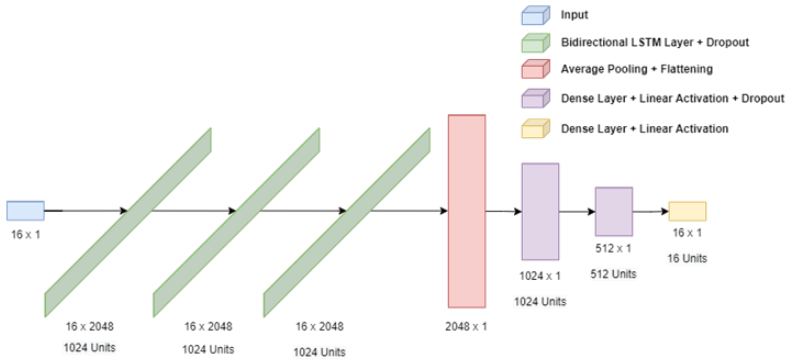

Figure 6. The architecture of LSTM for blind drift calibration.

A GRU is another form of the recurrent neural network, much like the LSTM, with only two gates: an update gate and a reset gate. Like the LSTM, the GRU operates on the data in time steps sequentially. The update gate decides the previous information to be passed to the next cell, and the reset gate decides how much information to discard before passing the state to the next cell. This architecture is similar to the LSTM but without the internal long-term memory stored in each cell and the gates performing different operations. Like the LSTM, we use bidirectional GRU layers that operate on the input from the beginning to the end and from the end to the beginning, concatenating the two outputs. The GRU architecture we developed is an eight-layer neural network (Figure 7). The first three layers are stacked with bidirectional GRU's. Each GRU returns the full sequence generated by each unit, creating a two-dimensional output. The number of units of each bidirectional GRU increases at each layer. As with the LSTM, the fourth layer is an average pooling and flattening layer to convert the two-dimensional output of the GRU's into a one-dimensional output. The final four layers are dense layers that convert the output of the GRUs into the output dimensions. The stacked bidirectional GRU's all use the standard hyperbolic tangent activation function and sigmoid for the recurrent activation function. Each dense layer uses a linear activation function. All layers except for the output layer are followed by a dropout layer with a dropout rate of 0.4.

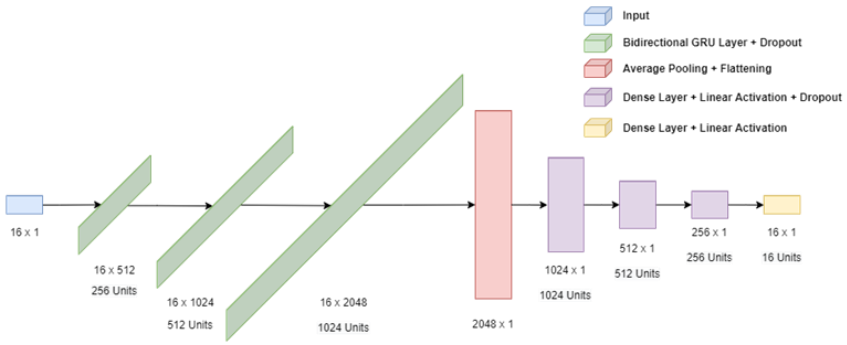

Figure 7. The architecture of GRU for blind drift calibration.

4. RESULTS

Table 1 compares the prediction results of autoencoder, FCNN, LSTM, and GRU on the test set. One can find that compared to unsupervised autoencoder, the supervised learning methods using FCNN, LSTM, or GRU yield significantly more precise predictions. Among the supervised learning methods that take advantage of the available ground truth from the calibrated sensors, the GRU achieves the lowest MSE loss during testing. MSEs of reconstructed errors on the test set are evaluated between the network's output using the uncalibrated sensor as input and the ground truth.

Table 3. Comparison of MSEs in autoencoder, FCNN, LSTM, and GRU

Network	Testing MSE Loss
Autoencoder	5.62E-01
FCNN	7.04E-04
LSTM	6.84E-04
GRU	4.67E-04

Figure 8 shows the 100-time step average of the ground truth sensor reading in the test set while Figure 9 shows the corresponding predictions of autoencoder, FCNN, LSTM, and GRU. One can find that each deep learning model except for the autoencoder learns to recover from the drifted measurements while keeping the general trends of the calibrated sensor. The FCNN, LSTM, and GRU generate similar results, but the GRU achieves the best performance by recovering from the most drift and closely following the trends of the calibrated sensors, thanks to taking advantage of the time series information. In contrast, the autoencoder fails to model the trends of the drift free measurements.

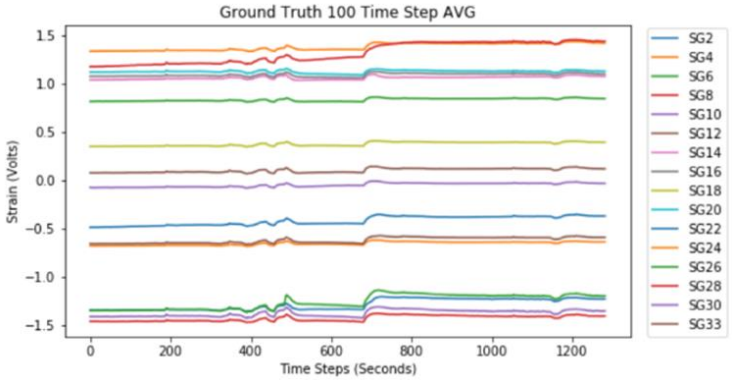

Figure 8. 100-time step average of the ground truth sensor readings in the test set.

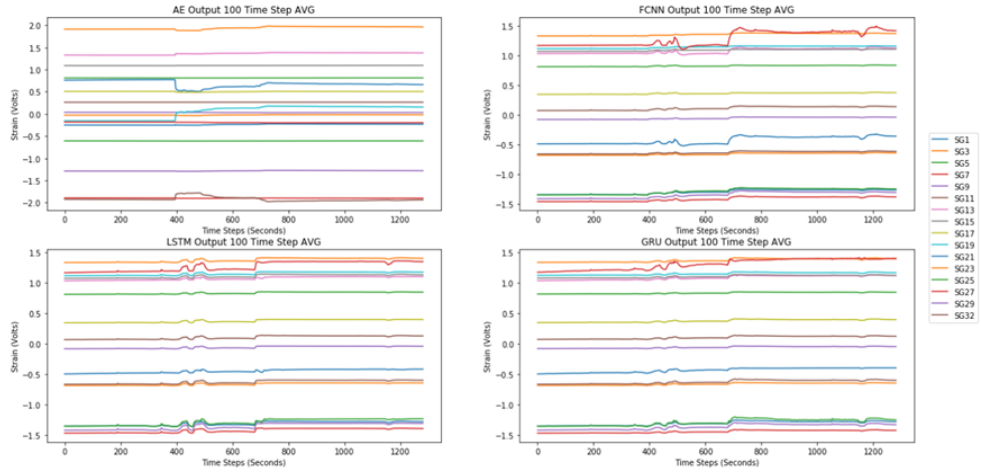

Figure 9. 100-time step average of the predictions of autoencoder, FCNN, LSTM, and GRU on the test set.

We further examine the predictions of the supervised learning models in more detail without 100-time steps averaging. In Figure 10, the uncalibrated sensor SG7 is shown in comparison to each of our model's predictions and the ground truth sensor SG8. One can find that the LSTM predicts the drift higher than the ground truth and overcompensates on the upward trends of the uncalibrated sensor. The supervised learning models all have a reasonable agreement with the ground truth. The FCNN model follows the trends of the calibrated sensor closely but fails to recover from all the drift from time steps 80,000+. The GRU model achieves the most recovery from drift but under compensates for the upward trends of the calibrated sensor. The unsupervised autoencoder outputs an almost straight line without correctly modeling the calibrated sensors' trends.

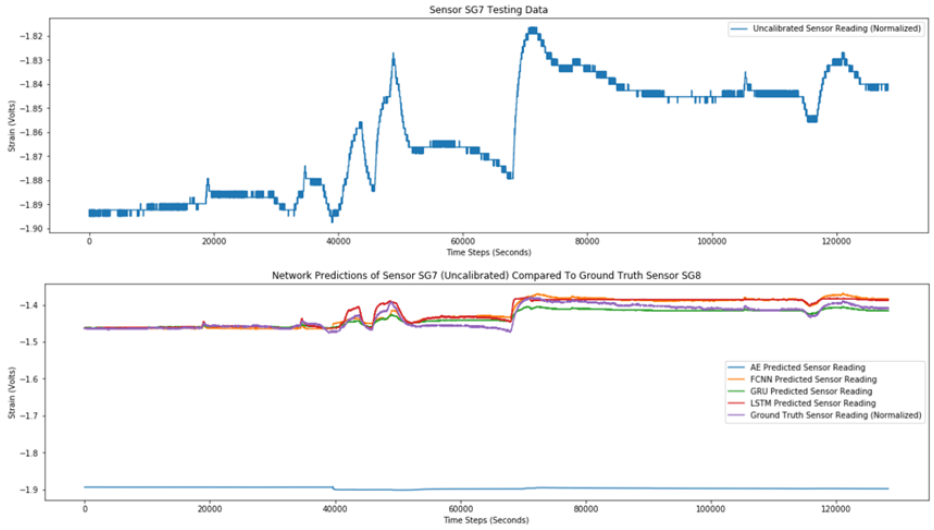

Figure 10. The comparison of SG7 and SG8 Sensors

The top figure shows the testing data of uncalibrated sensor SG7's readings. The bottom figure shows the network predictions using uncalibrated sensor SG7 in comparison to calibrate sensor SG8.

5. CONCLUSION

We build a structural model testbed to benchmark various deep learning blind drift calibration methods. Overall, the supervised learning architectures achieve good results in recovering from drifted measurements while closely following the trends of the calibrated sensor. The GRU model achieves the overall best results on our test set. The recurrent neural network models promise to remove drift and model the trends using their recurrent and memory mechanisms compared to the autoencoder and FCNN. The GRU and LSTM have robust memory mechanisms for time series, which can allow them to maintain states which can produce precise predictions close to drift free measurements.

Our future research will be on more powerful machine learning methods, such as ensemble-based approaches to integrate multiple deep learning models, to achieve better blind drift calibration enhancement. We will also further compare deep learning models with classical blind drift calibration methods on the structure lab testbed.

ACKNOWLEDGMENTS

This work is supported by ONR RSLP 2020-21 project. We thank Joanne Pilcher for helpful discussions.

REFERENCES

[1] L. Balzano, R. Nowak, "Blind calibration of sensor networks," Proceedings of ACM/IEEE International Conference on Information Processing in Sensor Networks, 2007.

[2] R. Tan, G. Xing, Z. Yuan, X. Liu, J. Yao, "System-level calibration for data fusion in wireless sensor networks," ACM Transactions on Sensor Network, 9(3), 2013.

[3] C. Xiang, P. Yang, C. Tian, H. Cai, Y. Liu, "Calibrate without calibrating: An iterative approach in participatory sensing network," IEEE Trans. Parallel Distrib. Syst., 26(2): 351–361, 2015.

[4] V. Bychkovskiy, S. Megerian, D. Estrin, "A collaborative approach to in-place sensor calibration," Proceedings of ACM/IEEE International Conference on Information Processing in Sensor Networks, 2013.

[5] C. Dorffer, M. Puigt, G. Delmaire, and G. Roussel, "Blind mobile sensor calibration using an informed nonnegative matrix factorization with a relaxed rendezvous model," Proceedings of International Conference on Acoustics, Speech, and Signal Processing, 2016.

[6] Y. Wang, A. Yang, Z. Li, X. Chen, P. Wang, H. Yang, "Blind drift calibration of sensor networks using sparse Bayesian learning," IEEE Sensors J., 16(16): 6249–6260, 2016.

[7] Y. Wang, A. Yang, X. Chen, P. Wang, Y. Wang, H. Yang, "A Deep Learning Approach for Blind Drift Calibration of Sensor Networks," IEEE Sensors Journal, 17(13): 4158–4171, 2017.

[8] G. E. Hinton, R. R. Salakhutdinov, "Reducing the Dimensionality of Data with Neural Networks", Science, **313**: 504-507, 2006.

[9] S. Hochreiter, J. Schmidhuber, “Long Short-Term Memory,” Neural Computation, **9**(8): 1735-1780, 1997.

[10] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, Y. Bengio, "Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation," arXiv:1406.1078, 2014.

[11] D. P., Kingma, J. Ba. "Adam: A method for stochastic optimization." *arXiv:1412.6980*, 2014.