A machine learning-based state estimation approach for varying noise distributions

Waleed Hilal, Stephen Gadsden, John Yawney

Waleed Hilal, Stephen A. Gadsden, John Yawney, "A machine learning-based state estimation approach for varying noise distributions," Proc. SPIE 12547, Signal Processing, Sensor/Information Fusion, and Target Recognition XXXII, 1254706 (14 June 2023); doi: 10.1117/12.2663898

SPIE.

Event: SPIE Defense + Commercial Sensing, 2023, Orlando, Florida, United States

A machine learning-based state estimation approach for varying noise distributions

Waleed Hilal*[a], Stephen A. Gadsden[a], John Yawney[ab]
[a]McMaster University, 1280 Main St. W, Hamilton, ON, CA L8S 4L8;
[b]Adastra Corporation, 200 Bay St., Toronto, ON, CA M5J 2J2

ABSTRACT

The field of estimation theory is concerned with providing a system with the ability to extract relevant information about the environment, resulting in more effective interaction with the system's surroundings through more well-informed, robust control actions. However, environments often exhibit high degrees of nonlinearity and other unwanted effects, posing a significant problem to popular techniques like the Kalman filter (KF), which yields an optimal only under specific conditions. One of these conditions is that the system and measurement noises are Gaussian, zero-mean with known covariance, a condition often hard to satisfy in practical applications. This research aims to address this issue by proposing a machine learning-based estimation approach capable of dealing with a wider range of noise types without the need for a known covariance. Harnessing the generative capabilities of machine learning techniques, we will demonstrate that the resultant model will prove to be a robust estimation strategy. Experimental simulations are carried out comparing the proposed approach with other conventional approaches on different varieties of functions corrupted by noises of varying distribution types.

Keywords: Estimation theory, Kalman filter, machine learning, robust estimation, signal filtering, nonlinear systems, non-Gaussian noise

1. BRIEF INTRODUCTION

The Kalman filter (KF) is regarded as one of the greatest discoveries in practical estimation theory, being a linear quadratic estimator, which provides the optimal solution to the linear estimation problem, assuming white measurement and disturbance noise. However, when the assumption of the distribution of the noise being white and Gaussian no longer holds, the KF fails to yield an optimal or reliable estimate.

Machine learning has been shown to be a very effective tool in the research community in the last decade. Specifically, generative models like autoencoders (AE), generative adversarial networks (GANs), transformers and more have demonstrated the ability to learn the underlying distribution of training data and model it so that new samples may be generated from this distribution.

This work proposes a machine learning-based approach to state estimation with the goal of overcoming the limitations associated with the KF in non-Gaussian noise conditions. We propose and design several machine learning algorithms, namely AE networks with varying architectures in the encoder and decoder, which will be trained to accurately filter and estimate the state of a system from measurements corrupted with different kinds of noise distributions. Experimental simulations will be carried out to compare the performance of the proposed machine learning-based models against that of the KF's.

The remainder of this paper is organized as follows: a background on the Kalman filter and estimation problem is detailed in Section 2. Then, the methodology behind the proposed research is outlined in Section 3, followed by a detailed discussion of the results in Section 4, Finally, concluding remarks and suggestions for future work are provided in Section 5.

Signal Processing, Sensor/Information Fusion, and Target Recognition XXXII, edited by Ivan Kadar, Erik P. Blasch, Lynne L. Grewe, Proc. of SPIE Vol. 12547, 1254706
 · 0277-786X · doi: 10.1117/12.2663898

2. BACKGROUND

2.1 Kalman filtering

Linear dynamic systems can be expressed in state-space representation as follows [1], [2]:

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}\boldsymbol{x}_k + \boldsymbol{B}\boldsymbol{u}_k + \boldsymbol{w}_k \tag{1}$$

$$\boldsymbol{z}_k = \boldsymbol{H}\boldsymbol{x}_k + \boldsymbol{v}_k \tag{2}$$

where $\boldsymbol{x}$ represents the system state vector, $\boldsymbol{A}$ is the discretized linear system model matrix of differential equations, $\boldsymbol{B}$ is the input gain matrix, $\boldsymbol{u}$ is the input vector, $\boldsymbol{w}$ is the system noise, $\boldsymbol{z}$ is the measurement vector, $\boldsymbol{H}$ is the linear measurement matrix, $\boldsymbol{v}$ represents the measurement noise, and k represents the current timestep.

The Kalman filter (KF) works under the assumptions that the system model is relatively well-known, and the initial states are also known, and finally, that the system and measurement noise is normal and Gaussian meaning that it is white with zero mean and known respective covariance matrices [3]. The KF works as a predictor-corrector; the system model is used to obtain an *a priori* or predicted estimate of the states, whereupon measurements combined with the Kalman gain matrix are used to apply a correction term to create an *a posteriori* or updated state estimate [4], [5].

The *a priori* state estimate is first computed using the process model, as can be seen in (3). Then, the *a priori* state covariance matrix is calculated based on the process model and the associated modeling noise covariance matrix $\boldsymbol{Q}_k$, as shown in (4):

$$\hat{\boldsymbol{x}}_{k+1|k} = \boldsymbol{A}\hat{\boldsymbol{x}}_{k+1|k} + \boldsymbol{B}u_k \tag{3}$$

$$\hat{\boldsymbol{P}}_{k+1|k} = \boldsymbol{A}\boldsymbol{P}_{k|k}\boldsymbol{A}^T + \boldsymbol{Q}_k \tag{4}$$

The Kalman gain computation in (5) is based on (4), and is then used to update the state estimate in (6):

$$\boldsymbol{K}_{k+1} = \boldsymbol{P}_{k+1|k}\boldsymbol{H}^T\boldsymbol{S}_{k+1}^{-1} \tag{5}$$

$$\hat{\boldsymbol{x}}_{k+1|k+1} = \hat{\boldsymbol{x}}_{k+1|k} + \boldsymbol{K}_{k+1}\boldsymbol{\nu}_{k+1} \tag{6}$$

where $\boldsymbol{\nu}$ and $\boldsymbol{S}$ are two important terms known as the innovation (or residual), and the innovation covariance, respectively. In the equations below, $\boldsymbol{R}$ is the measurement noise covariance.

$$\boldsymbol{\nu}_{k+1} = \boldsymbol{z}_k - \boldsymbol{H}\boldsymbol{A}\hat{\boldsymbol{x}}_{k+1|k} \tag{7}$$

$$\boldsymbol{S}_{k+1} = \boldsymbol{H}\boldsymbol{P}_{k+1|k}\boldsymbol{H}^T + \boldsymbol{R}_{k+1} \tag{8}$$

The innovation, from (7), represents the difference between the actual measurements and the *a priori* estimate of the measurements. The innovation covariance, as in (8), characterizes the uncertainty in the measurement predictions. These two terms provide an important insight into the estimation process and are often used to assess the filter's overall estimation ability.

The *a posteriori* state error covariance matrix is then calculated in (9), and the process repeats iteratively:

$$\boldsymbol{P}_{k+1|k+1} = (\boldsymbol{I} - \boldsymbol{K}_{k+1}\boldsymbol{H})\boldsymbol{P}_{k+1|k}(\boldsymbol{I} - \boldsymbol{K}_{k+1}\boldsymbol{H})^T + \boldsymbol{K}_{\boldsymbol{k+1}}\boldsymbol{R}_{\boldsymbol{k+1}}\boldsymbol{K}_{\boldsymbol{k+1}}^T \tag{9}$$

where $\boldsymbol{I}$ is the identity matrix. In a successful application of the KF, the state estimates will rapidly converge, providing the optimal statistical estimate based on the given information. The *a posteriori* covariance update in (9) is known as the 'Joseph covariance form' and is often preferred due to its superior numerical characteristics. The Joseph form ensures that the covariance update remains positive-definite, a critical condition in the estimation process to produce meaningful results [3].

2.2 Autoencoders

An autoencoder (AE) is a type of unsupervised deep learning network symmetric in structure with fewer nodes in the middle layers. It has a section that encodes inputs into a lower-dimensional representation and another section that decodes or reconstructs that input again. The goal of training an AE is to learn a reduced encoding of data efficiently and then reconstruct it. As illustrated in Figure 1, the input layer passes the input data to the hidden layer, where the lower-dimensional encoding is learned. Then, the encoding is passed from the hidden layer to the output layer, where it is decoded and reconstructed as much as possible. The number of hidden layers in an AE is arbitrary, with the condition that for each part of the network, e.g., the encoder, each subsequent hidden layer must have fewer neurons than the previous layer. This architecture imposes a bottleneck in the network, restricting the amount of information that can traverse through and in turn forcing a compressed knowledge of the original input [6], [7].

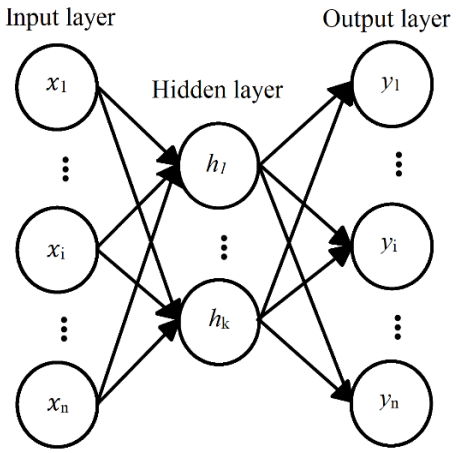

Figure 1. Schematic of an autoencoder network's architecture, where the left side is the encoder and the right side is the decoder.

2.3 Long Short-term Memory Networks

Long Short-Term Memory networks (LSTM) are an extension of recurrent neural networks (RNN), a form of deep neural network primarily used for time-series data proposed by Hochreiter and Schmidhuber in 1997 [8]. Each neuron in an LSTM is a cell with 'memory' that can store information, maintaining its own state, in contrast to RNNs that merely take the current input from their previous hidden state to output a new hidden state. The improved memory capacity of LSTMs is thanks to the introduction of input and output "gates" into the cell, which were shortly followed by the introduction of the forget gate by Gers *et al.* in 2000 [9]. For a thorough review of LSTMs and the different variants, we refer the reader to a recent survey paper by Yu *et al.* [10]. LSTMs currently constitute the state-of-the-art in many real-world applications such as text, writing and speech recognition, as well as natural language processing. They are also well-known for addressing the vanishing gradient problem that is generally associated with RNNs [8].

3. METHODOLOGY

3.1 Dataset Generation

The proposed machine learning-based estimation models were trained using an approach known as domain randomization [11] on several types of functions: exponential, sigmoidal and sinusoidal functions. In order to test and evaluate the performance of the models, testing sets were generated of curves which were drawn from the same function

and noise family type used during training but were never observed or witnessed by the trained models. Each function from the training dataset constituted a ground truth ϕ_g, which was subsequently added with noise ϕ_n, creating the simulated measurement data ϕ:

$$\phi = \phi_g + \phi_n \tag{10}$$

The simulated curves in the datasets involved in this study were generated by the following functions:

$$\phi_g(t;\alpha) = e^{\alpha x} - 1 \tag{11}$$

$$\phi_g(t;\alpha) = \alpha\left(1 + e^{-0.15(t-60)}\right)^{-1} \tag{12}$$

$$\phi_g(t;\alpha,\beta) = \alpha \sin \beta t \tag{13}$$

For equations (11)-(13) above, α and β are scalar constants sampled uniformly in [0.05, 0.085] which is the main mechanism behind the diversity and randomization of the curves in the training and testing set.

As for the simulated noise that is added to the ground truth curves, they consist of Gaussian, bimodal and Cauchy distributions of noise. The Gaussian noise was sampled from a distribution with zero mean and constant known variance. The bimodal noise consisted of a mixture model comprising of two equally weighted Gaussian distributions, with means of -5 and 5, and a common constant covariance. Finally, a standard Cauchy distribution was also used as the third noise distribution to examine the effects of.

3.2 Model Training and Tuning

The first model implemented for this study is an AE network with a basic fully connected multi-layered perception architecture. The network architecture consists of 8 layers, 4 of which are in the encoder section and the remaining 4 in the decoder. In the encoder, the first layer has a total of 128 neurons, followed by 64 in the second layer, then 32 neurons in the third layer and finally, 16 neurons in the last layer of the encoders. This architecture is mirrored by the decoder, which goes from 16 to 32 neurons in the first layers, to 64 in the second layer, 128 in the third layer, and finally the output layer which consists of a single output neuron. The parametric rectified linear unit (PReLU) was used as the activation function in each of the layers, except for the final output layer after the decoder. A regularization method known as dropout was also implemented, with a probability of 1% for any neuron to be dropped. The goal of this is to help the model prevent overfitting the training data. The criterion or objective used to train the model was the mean squared error (MSE) loss alongside an Adam optimizer with a learning rate of $1e^{-4}$. A sampling time of 0.1 seconds was used for all the models in this study.

The second model under study is an AE-LSTM network, with the difference from the first model being the fact that LSTM cells are used internally in the encoder and decoder architecture instead of perceptrons. However, in the encoder, only two LSTM layers are employed with 256 and 64 cells in the first and second layer, respectively. In the decoder, this architecture is mirrored with 64 and 256 cells in the first and second respective layers, followed by a final output layer. Unlike the AE, the Huber loss was the choice for the criterion or objective function as it demonstrates more stable and improved performance than the MSE loss in this setting. Similar to the AE case, an Adam optimizer was used, however with a slightly larger learning rate of $1e^{-3}$. It is important to note that the choice of such hyperparameter values in this case and in the previous case of the AE were chosen after performing an extensive grid search of all the possible hyperparameter values.

3.3 Model Evaluation and Testing

Each of the three different kinds of models (KF, AE, and AE-LSTM) will be implemented 3 times, resulting in a total of 3 tuned models of each architecture or algorithm under study. The first set of models will be trained on a dataset corrupted with Gaussian, white noise, while the second and third set of models will be trained on datasets of bimodal and Cauchy noise, respectively. An extra model was designed and trained for each of the 3 cases, which was an enlarged version of the simple AE architecture described in the previous subsection. In this model, the size of the layers was increased by a factor of 10 to observe whether or not a more complex architecture would affect the performance of the model. Once trained, each model will be evaluated according to the average root mean squared error (RMSE) value of the estimates for the curves in the testing set. The performance results for each model in the different circumstances are discussed in the following section.

4. RESULTS AND DISCUSSION

In this section, the performance of each model in varying noise types is presented. In Table I, the average RMSE of each model over the entire training set is presented in the case of Gaussian noise, whereas Table II and Table III highlight the RMSE of each model in the case of bimodal and Cauchy noise, respectively. In each table, the AE-S model represents the AE with the original architecture described in section 3.2, and AE-L represents the AE model with the increased number of neurons in each layer, as described in section 3.3. As discussed, the models were evaluated on three types of curves, exponential, sigmoidal and sinusoidal curves, and the ratio of each model's RMSE to the lowest RMSE is also presented in the below tables for further comparison purposes.

TABLE I
RMSE OF STUDIED MODELS AND RESPECTIVE RATIO COMPARED TO BEST MODEL FOR GAUSSIAN NOISE

Model	Exp. RMSE	Exp. Ratio	Sig. RMSE	Sig. Ratio	Sin. RMSE	Sin. Ratio
KF	**0.033**	**1.00**	**0.043**	**1.00**	**0.081**	**1.00**
AE-S	0.112	3.39	0.191	4.44	0.230	2.84
AE-L	0.103	3.12	0.156	3.63	0.192	2.37
AE-LSTM	0.089	2.70	0.092	2.14	0.154	1.90

As evident from Table I above, and as can be expected based on theoretical knowledge discussed throughout this paper, the KF yields the best estimate compared to all of the other models. No matter the type of curve, the KF's estimate is the most accurate. The AE-S and AE-L both perform relatively poorly compared to the KF, which has an RMSE value 3 times lower than both AE architectures. Similarly, the AE-LSTM model performs slightly better than the simpler AE-S and AE-L, but is still half or three times less accurate than the KF in performance.

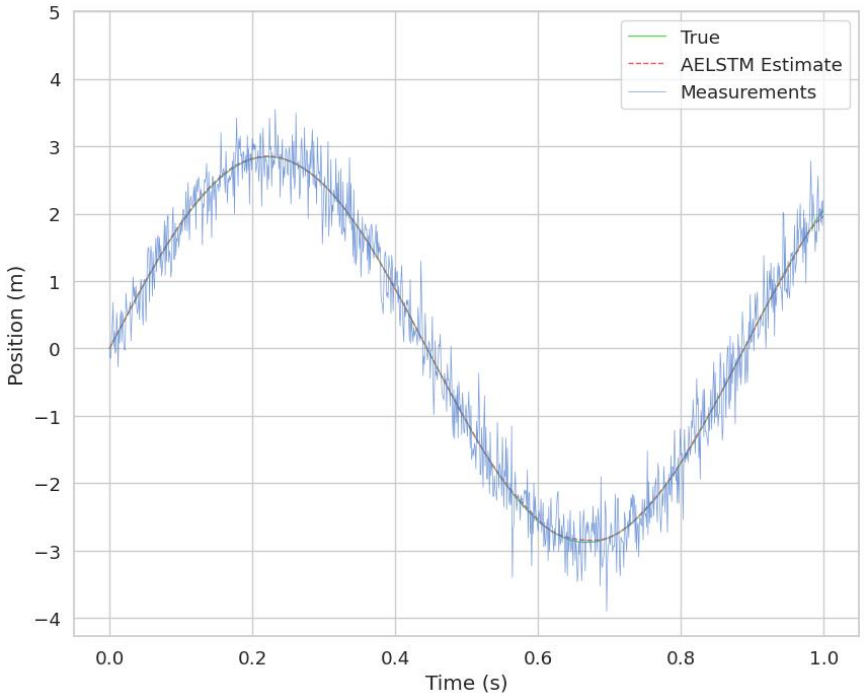

Figure 2. State estimate computed by the AELSTM in the case of Gaussian distributed noise.

TABLE II
RMSE OF STUDIED MODELS AND RESPECTIVE RATIO COMPARED TO BEST MODEL FOR BIMODAL NOISE

Model	Exp. RMSE	Exp. Ratio	Sig. RMSE	Sig. Ratio	Sin. RMSE	Sin. Ratio
KF	1.134	7.61	1.244	7.97	0.712	3.96
AE-S	1.199	8.05	1.135	7.28	1.280	7.11
AE-L	1.021	6.85	0.982	6.29	1.043	5.69
AE-LSTM	**0.149**	**1.00**	**0.156**	**1.00**	**0.180**	**1.00**

In the case of bimodal noise, we begin to witness signs of the KF's degradation in performance. Whilst still yielding a relatively acceptable range of RMSE values, it is evident from Table II that the AE-LSTM is the model which outperforms all the others. This is true in the case of all three types of curves. Interestingly enough, the AE-S renders estimates which are still less accurate than the KF in the case of the exponential and sinusoidal curves, whereas the AE-L is more accurate than both the AE-S and KF. This leads to the notion that it may be possible to achieve more accurate estimates out of an AE model. However, this is associated with its own limitations, as it was encountered throughout this study that the computational expense of the AE-L model exceeded that of the AE-S by almost 3 magnitudes. This renders the AE-L as an impractical solution, despite possibly being able to achieve further improved estimates with more complex architectures. Most interestingly, however, is the fact that the AE-LSTM yields estimates which are significantly more accurate than the other models, by up to 8 times compares to the KF and AE-S, and up to 7 times better than the AE-L.

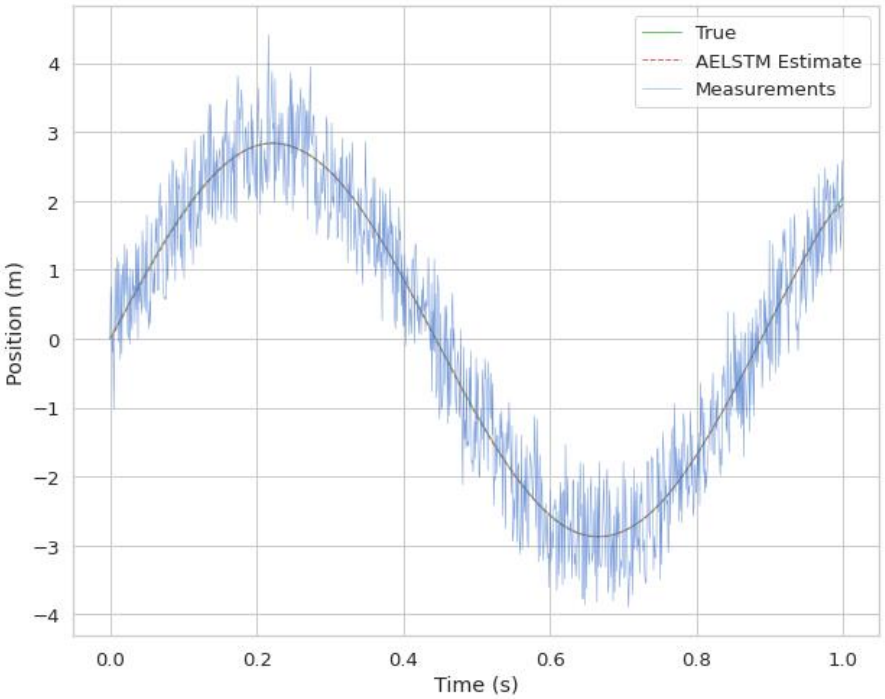

Figure 3. State estimate computed by the AELSTM in the case of bimodal distributed noise.

These findings are further corroborated in the case of Cauchy noise, as can be seen from the RMSE results presented in Table III. From this table, we confirm that the AE-LSTM offers a practical solution to the non-Gaussian noise limitations encountered by the KF, performing up to 3 times better than the KF. An interesting observation can be made in that the effect of Cauchy noise on the RMSE results of all models is significantly less than that of the bimodal noise distribution. Further investigation into ways to remedy the extreme degradation in estimate accuracy in the case of these noises is an interesting avenue for future research.

TABLE III
RMSE OF STUDIED MODELS AND RESPECTIVE RATIO COMPARED TO BEST MODEL FOR CAUCHY NOISE

Model	Exp. RMSE	Exp. Ratio	Sig. RMSE	Sig. Ratio	Sin. RMSE	Sin. Ratio
KF	1.182	2.85	1.198	3.72	1.288	2.01
AE-S	1.239	2.99	1.034	3.21	1.399	2.18
AE-L	0.821	1.98	0.837	2.60	1.203	1.88
AE-LSTM	**0.415**	**1.00**	**0.322**	**1.00**	**0.641**	**1.00**

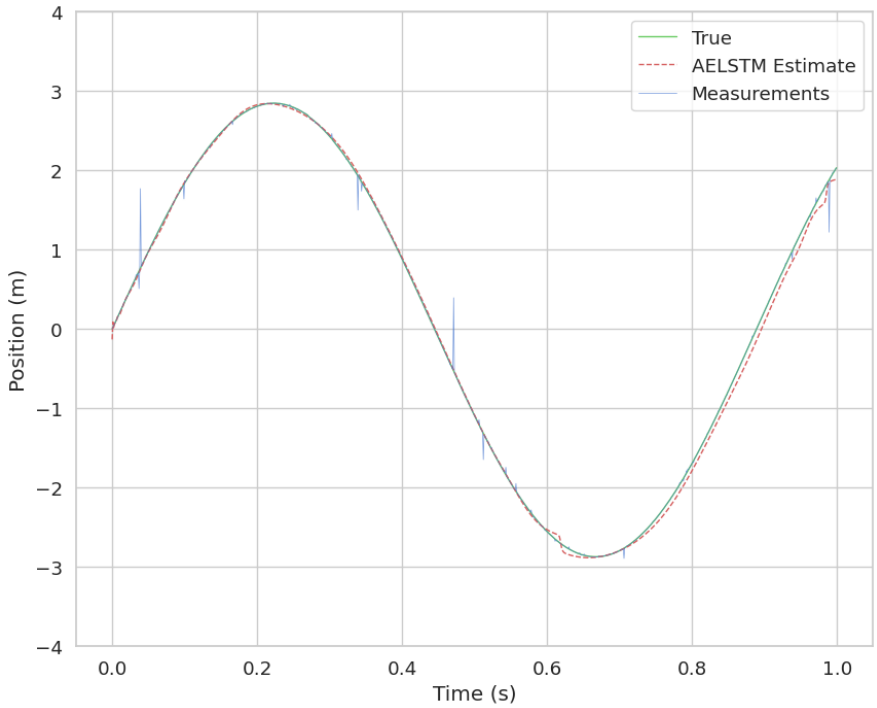

Figure 4. State estimate computed by the AELSTM in the case of Cauchy distributed noise.

5. CONCLUSION

This paper aims to address the limitations of traditional estimation theory filters like the KF when faced with varying noise distributions that are non-Gaussian. Machine learning models, specifically generative models, have been proposed as a means of capturing the true underlying distribution of the data in order to output a reliable estimate when faced with non-Gaussian noise. Through experimental simulations involving several types of curves, it was demonstrated by this research that the proposed AELSTM model can address the issues related with non-Gaussian noise distributions. While simpler models like the AE can perform well with complex architectures, this poses other issues in terms of computational power requirements. This study will serve as a foundation for further investigation into combining and balancing the optimality of the KF with the robustness of the machine learning algorithms examined in this paper.

6. REFERENCES

[1] Y. Bar-Shalom, X.-R. Li, and T. Kirubarajan, *Estimation with Applications to Tracking and Navigation*. 2001. doi: 10.1002/0471221279.

[2] J. Goodman, W. Hilal, S. A. Gadsden, and C. D. Eggleton, “Adaptive SVSF-KF estimation strategies based on the normalized innovation square metric and IMM strategy,” *Results in Engineering*, vol. 16, 2022, doi: 10.1016/j.rineng.2022.100785.

[3] M. S. Grewal and A. P. Andrews, *Kalman Filtering: Theory and Practice with MATLAB®: Fourth Edition*, vol. 9781118851210. 2014. doi: 10.1002/9781118984987.

[4] W. Hilal, S. A. Gadsden, S. A. Wilkerson, and M. A. Al-Shabi, “A square-root formulation of the sliding innovation filter for target tracking,” 2022. doi: 10.1117/12.2618965.

[5] W. Hilal, S. A. Gadsden, S. A. Wilkerson, and M. A. Al-Shabi, “Combined particle and smooth innovation filtering for nonlinear estimation,” 2022. doi: 10.1117/12.2618973.
[6] J. Chen, X. Feng, L. Jiang, and Q. Zhu, “State of charge estimation of lithium-ion battery using denoising autoencoder and gated recurrent unit recurrent neural network,” *Energy*, vol. 227, 2021, doi: 10.1016/j.energy.2021.120451.
[7] W. Hilal, S. A. Gadsden, and J. Yawney, “Financial Fraud: A Review of Anomaly Detection Techniques and Recent Advances,” *Expert Systems with Applications*, vol. 193. Elsevier Ltd, May 01, 2022. doi: 10.1016/j.eswa.2021.116429.
[8] S. Hochreiter and J. Schmidhuber, “Long Short-Term Memory,” *Neural Comput*, vol. 9, no. 8, 1997, doi: 10.1162/neco.1997.9.8.1735.
[9] F. A. Gers, J. Schmidhuber, and F. Cummins, “Learning to forget: Continual prediction with LSTM,” *Neural Comput*, vol. 12, no. 10, 2000, doi: 10.1162/089976600300015015.
[10] Y. Yu, X. Si, C. Hu, and J. Zhang, “A review of recurrent neural networks: Lstm cells and network architectures,” *Neural Computation*, vol. 31, no. 7. 2019. doi: 10.1162/neco_a_01199.
[11] M. L. Weiss, R. C. Paffenroth, and J. R. Uzarski, “The Autoencoder-Kalman Filter: Theory and Practice,” in *Conference Record - Asilomar Conference on Signals, Systems and Computers*, 2019. doi: 10.1109/IEEECONF44664.2019.9048687.