# Robustness of Neural Networks used in Electrical Motor Time-Series

**Anonymous Author(s)**
Affiliation
Address
email

## Abstract

Electrical motors are widely used in industrial and emerging applications such as electrical automotive. Industrial 4.0 has led to the usage of neural networks for electrical motor tasks like fault detection, monitoring, and control of electrical motors. The growing increase of neural networks in safety-critical systems requires an in-depth analysis of their robustness and stability. This paper studies the robustness of neural networks used in time-series tasks like system modeling, signal denoising, speed-torque estimation, temperature estimation, and fault detection. The dataset collected for these problems has all types of noise from the operating environment, sensors, and the system itself. This affects the performance of different network architectures during training and inference. We train and analyze under perturbations several different architectures that range from simple linear, convolutional and sequential networks to complex networks like 1D ResNet and Transformers.

## 1 Introduction

Electrical motors are one of the most used heavy devices in industrial and non-industrial places. In heavy industry, they can be found in cranes, tunnel boring machines, and trains, while at home, they can be found inside all electrical appliances like washing machines, fans, and rotational drives. Each of these applications requires different operating reliability from the motor used. These motors also have different types of operating environments and scenarios. A brushless DC motor inside a rotational drive has a consistent operating paradigm. In contrast, an induction motor inside a tunnel boring machine has to face extreme heat, humidity, and dust while having a very inconsistent operation cycle. Given the nature of the operation, it has been vital for the drive control community to use fault detection and observation methods [1, 2]. Recently neural network based methods have been proposed for such tasks [3, 4, 5]. The scale with which neural network based methods have been proposed for electrical motor tasks does not match the very much required robustness and stability analysis of such methods.

Figure 1 shows how naturally occurring perturbation in inputs can cause a neural network to give the wrong output. When such neural networks are cascaded to drive a control system, it can lead to catastrophic failures. In this case, meta-denoiser [6] and speed-torque estimator [7] together take denoised currents and voltages and estimate speed and torque which can then be used to drive the control system. Currents and voltages are measured using sensors that can be affected by extreme heat, water, dust, or material faults. Sensors can also perform poorly due to aging. Motors can age and get affected due to their environment and will behave differently. Given that meta-denoiser and DiagBiRNN have not been trained on a dataset that considers such varied inputs and motor states, the networks are bound to give wrong predictions, which can affect the downstream task of control.

Submitted to Workshop on Robustness in Sequence Modeling at the 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

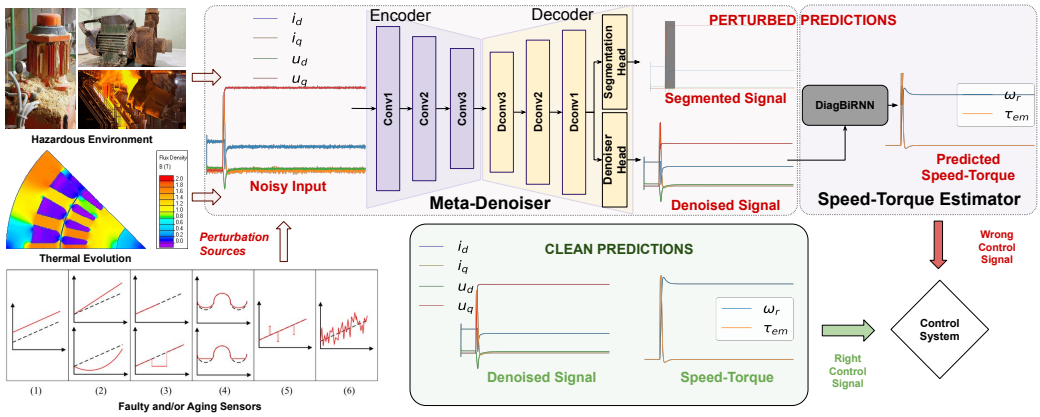


Figure 1: Different sources of perturbations in inputs to neural networks used in electrical motor tasks like denoising and speed-torque estimation.

In [8], convolutional, feed-forward, and sequential neural networks have been used to model electrical **motor dynamics** by modeling the input-output relationship of its quantities in a data-driven manner. An encoder-decoder based network called DiagBiRNN has also been proposed which is then used in [7] to estimate **speed-torque** from currents and voltages. To apply this network in real-world setting where currents and voltages are noisy, a meta-denoiser network has been proposed in [6] to **denoise** noisy currents and voltages. To model electrical motor dynamics, a multi-scale pyramid and lightweight residual networks have been proposed in [9, 10]. Estimating the temperature of different parts of an electrical motor is important to understand its thermal evolution. Given the complexity of rotating parts, [11] proposed an induction motor temperature dataset. They use convolution and recurrent network-based benchmarks to predict permanent magnet **temperature** using several electrical motor quantities and temperatures of different motor parts, which are easy to record. Several electrical motor fault datasets cover different categories of faults occurring in different types of motors. [12] proposed fault detection and classification dataset to identify the number of **broken bars** from electrical motor quantities and vibrations recorded using accelerometers.

## 2 Robustness Issue in Neural Networks

Test time instability of neural networks is a well-known and active area of research leading to a more explainable and reliable A.I. In [13], the concept of adversarial attacks was first proposed to fool neural networks. Adding a well-crafted subtle perturbation to the input of the neural network produces a misclassification. This scenario is possible even when the model has good clean accuracy. These attacks pose a huge threat to the performance of neural networks. There has been a multitude of works introducing stronger adversarial attacks and their defenses. Goodfellow et al. [14] proposed the Fast Gradient Sign Method (FGSM) to generate $\ell_\infty$ bounded adversarial attacks. This is a white box attack, i.e., it has access to network structure, parameter weights, and all the related training details. The generated inputs are misclassified by adding perturbations and linearizing the cost function in the gradient direction. FGSM is a single-step attack, Madry et al. [15] proposed a multi-step variant of FGSM called Projected Gradient Descent (PGD) attack. Kurakin et al. [16] proposed an optimized FGSM, Iterative Gradient Sign Method (IGSM), which adds perturbations in multiple smaller steps and clips the results after each iteration ensuring that the perturbations are restricted to the neighborhood of the example. Dong et al. [17] added momentum to IGSM attacks. Moosavi et al. [18] proposed Deepfool as a non-targeted attack that tries to find the decision boundary closest to the sample in the input space and then uses the classification boundary to fool the classifier. Moosavi et al. [19] proposed a universal image-agnostic perturbation attack method that fools the classifier by adding a single perturbation to all images in the dataset. Carlini et al. [20] proposed a powerful attack based on L-BFGS. In [21], the authors propose a general framework for the generation of adversarial examples in both classification and regression tasks for applications in the image domain.

Most of these methods belong to the class of white box attackers, i.e., the attacker has access to the information related to the trained neural network model, including the model architecture and its

parameters. A black box attacker is introduced in [22]. Such attackers do not know the model but can interact with it. Ballet et al. proposed a white box attacker specifically for handling tabular [23] for classification tasks. A similar approach using the Jacobian property of the neural network for tabular data is proposed in Gupta et al.[24] for regression tasks. These attacks and defenses constitute the adversarial robustness of neural networks. Robustness of time-series networks due to thermal evolution can be considered as a generalization problem. Such problem have been very well studied in [25, 26].

## 3 Experimental Setup

We use 9 different architectures proposed in [8] for **motor dynamics**, **denoise**, and **speed-torque** tasks. We use networks named FNN, RNN, LSTM, and CNN for baselines. We also use encoder-decoder variants from the paper named Deep, Skip, RNN-Skip, BiRNN-Skip, and DiagBiRNN-Skip to understand robustness behavior with respect to network complexity. Details of these networks can be found in [8]. For the **temperature** task we use DiagBiRNN-Skip and FedFormer [27]. For the **broken bars** task, we use 1D variant of three classification networks namely CRNN [28], ResNet-18 [29], and RegNet-20 [30].

For generating adversarial attacks, we use FGSM and DeepFool. The Value of $\epsilon$ is taken at 0.01 and 0.1. In the case of DeepFool number of iterations used is 100. For the **broken bars** task, we report clean accuracy and FGSM and DeepFool attack accuracy. For all the regression tasks, we report clean and FGSM attack values of Mean Absolute Error (MAE), Symmetric Mean Absolute Percentage Error (SMAPE), coefficient of determination ($R^2$) [31], and Root Mean Squared Error (RMSE).

| Network | Attack | MAE | SMAPE(%) | $R^2$ | RMSE |
|---|---|---|---|---|---|
| **Deep** | **Clean** | 0.026 | 5.15 | 0.73 | 0.06 |
| | **FGSM** $\epsilon = 0.01$ | 0.11 | 22.89 | -1.02 | 0.15 |
| | **FGSM** $\epsilon = 0.1$ | **0.15** | **32.11** | **-2.90** | **0.21** |
| **Skip** | **Clean** | **0.02** | **4.62** | **0.77** | **0.05** |
| | **FGSM** $\epsilon = 0.01$ | 0.11 | 24.94 | -1.14 | 0.16 |
| | **FGSM** $\epsilon = 0.1$ | 0.39 | 94.92 | -21.99 | 0.52 |
| **RNN-Skip** | **Clean** | 0.03 | 6.73 | 0.72 | 0.06 |
| | **FGSM** $\epsilon = 0.01$ | **0.09** | **20.36** | **-0.54** | **0.13** |
| | **FGSM** $\epsilon = 0.1$ | 0.29 | 73.5 | -10.87 | 0.37 |
| **BiRNN-Skip** | **Clean** | 0.03 | 6.82 | 0.72 | 0.06 |
| | **FGSM** $\epsilon = 0.01$ | 0.13 | 26.09 | -1.85 | 0.18 |
| | **FGSM** $\epsilon = 0.1$ | 0.69 | 76.03 | -84.77 | 0.99 |
| **DiagBiRNN-Skip** | **Clean** | 0.02 | 4.70 | 0.76 | 0.05 |
| | **FGSM** $\epsilon = 0.01$ | 0.10 | 21.31 | -0.83 | 0.15 |
| | **FGSM** $\epsilon = 0.1$ | 0.32 | 58.40 | -16.23 | 0.44 |

Table 1: Metrics of clean and adversarial predictions from all the encoder-decoder variants trained for **motor dynamics** task.

Tables 1 shows results obtained by training encoder-decoder networks proposed in [8] for **motor dynamics task**. It shows MAE, SMAPE, $R^2$, and RMSE for clean predictions and FGSM predictions at $\epsilon = 0.01$ and $\epsilon = 0.1$. Skip encoder-decoder variant achieves the best clean predictions SMAPE (4.62%) and $R^2$ (0.77). When attacked, RNN-Skip is most robust at $\epsilon = 0.01$ with SMAPE (20.36%), followed by DiagBiRNN-Skip with SMAPE (21.31%). Skip achieves the second worst performance when attacked.

In case of **denoise** task, RNN-Skip obtains the best SMAPE (0.14%) among all the variants. $R^2$ is 0.99 for all the networks. However, when attacked with FGSM at $\epsilon = 0.01$, DiagBiRNN-Skip outperforms every other network with the lowest SMAPE (1.53%). With a more aggressive attack of $\epsilon = 0.1$, DiagBiRNN-Skip still outperforms every other network with SMAPE (13.6%) and $R^2$ (0.72).

Table 2 shows results of the encoder-decoder variants trained for **speed-torque** estimation task. Encoder-decoder variant Skip shows the best SMAPE (0.18%) among all the variants. $R^2$ is 0.99 for all the networks. However, when attacked with FGSM at $\epsilon = 0.01$, DiagBiRNN-Skip outperforms every other network with the lowest SMAPE (2.70%). With a more aggressive attack of $\epsilon = 0.1$, RNN-Skip outperforms every other network with SMAPE (17.96%) and $R^2$ (-0.56). DiagBiRNN-Skip achieves the best $R^2$ of -0.49.

| Network | Attack | MAE | SMAPE(%) | $R^2$ | RMSE |
|---|---|---|---|---|---|
| **Deep** | **Clean** | 0.00 | 0.3 | 0.99 | 0.00 |
| | **FGSM** $\epsilon = 0.01$ | 0.01 | 2.94 | 0.96 | 0.02 |
| | **FGSM** $\epsilon = 0.1$ | 0.096 | 24.42 | -0.87 | 0.13 |
| **Skip** | **Clean** | **0.00** | **0.18** | **0.99** | **0.00** |
| | **FGSM** $\epsilon = 0.01$ | 0.01 | 2.83 | 0.97 | 0.02 |
| | **FGSM** $\epsilon = 0.1$ | 0.10 | 23.99 | -0.80 | 0.13 |
| **RNN-Skip** | **Clean** | 0.00 | 0.82 | 0.99 | 0.00 |
| | **FGSM** $\epsilon = 0.01$ | 0.01 | 3.13 | 0.96 | 0.02 |
| | **FGSM** $\epsilon = 0.1$ | **0.10** | **17.96** | **-0.56** | **0.12** |
| **BiRNN-Skip** | **Clean** | 0.00 | 0.68 | 0.99 | 0.00 |
| | **FGSM** $\epsilon = 0.01$ | 0.01 | 2.91 | 0.97 | 0.02 |
| | **FGSM** $\epsilon = 0.1$ | 0.10 | 24.84 | -0.67 | 0.12 |
| **DiagBiRNN-Skip** | **Clean** | 0.00 | 0.26 | 0.99 | 0.00 |
| | **FGSM** $\epsilon = 0.01$ | **0.01** | **2.70** | **0.97** | **0.02** |
| | **FGSM** $\epsilon = 0.1$ | 0.10 | 20.11 | -0.49 | 0.12 |

Table 2: Metrics of clean and adversarial predictions from all the networks trained for **speed-torque** estimation task.

| Network | Attack | MAE | SMAPE(%) | $R^2$ | RMSE |
|---|---|---|---|---|---|
| **FedFormer** | **Clean** | 0.03 | 6.47 | 0.96 | 0.04 |
| | **FGSM** | 0.03 | **7.53** | 0.95 | 0.05 |
| **DiagBiRNN-Skip** | **Clean** | **0.00** | **0.82** | **1.00** | **0.01** |
| | **FGSM** | 0.03 | 7.73 | **0.97** | **0.04** |

Table 3: Metrics for clean and adversarial predictions for the two networks trained for permanent magnet **temperature** prediction task.

Table 3 shows results for FedFormer and DiagBiRNN-SKip trained to do permanent magnet **temperature** prediction task. In this case, it can be seen that FedFormer has performed very poorly when compared to DiagBiRNN in terms of clean predictions. However, when attacked with FGSM at $\epsilon = 0.01$, FedFormer manages to be robust and give the same result as DiagBiRNN. This shows that FedFormer is very robust to adversarial examples. It should be noted that the clean metrics of FedFormer can be improved with a better training strategy as we have used default hyperparameters to train the network.

| $\epsilon$ | Method | Clean(%) | FGSM(%) | DeepFool(%) |
|---|---|---|---|---|
| **0.01** | **CRNN** | 80.0 | 79.0 | 79.0 |
| | **ResNet** | 90.0 | 89.0 | 89.0 |
| | **RegNet** | **93.0** | **92.0** | **92.0** |
| **0.1** | **CRNN** | 79.4 | 61.4 | 60.1 |
| | **ResNet** | 88.9 | 80.0 | 79.6 |
| | **RegNet** | **92.3** | **86.8** | **86.6** |

Table 4: Classification results for **broken bars** task.

Table 4 shows results obtained by CRNN, ResNet, and RegNet on **broken bars** task. In this case, RegNet gives the best clean accuracy. When attacked with FGSM and DeepFool at $\epsilon = 0.01$ and $\epsilon = 0.1$, the accuracy of all three networks decreases, but the order remains the same. Although at $\epsilon = 0.1$, RegNet is still more robust compared to the other two netwo

## 4 Conclusion and Future Work

We present a robustness analysis of neural networks used in five different electrical motor tasks. We train a wide array of networks and generate adversarial attacks on them to show their instability. Although our experiments show that networks are somewhat unstable to input perturbations, we still need more sophisticated perturbations to understand the robustness of neural network in electrical motors. In future, it will would be interesting to consider domain knowledge and sequential dynamics of data to generate better attacks and improve the robustness of neural networks for electrical motors. It is also essential to understand the sensitivity of the individual inputs with respect to neural network stability.

## References


[1] C. C. Chan and H. Wang, "An effective method for rotor resistance identification for high-performance induction motor vector control," *IEEE TIE*, vol. 37, no. 6, pp. 477–482, 1990.

[2] R. Marino, S. Peresada, and P. Tomei, "Global adaptive output feedback control of induction motors with uncertain rotor resistance," *TACON*, vol. 44, no. 5, pp. 967–983, 1999.

[3] A. A. Silva, A. M. Bazzi, and S. Gupta, "Fault diagnosis in electric drives using machine learning approaches," in *IEMDC*, pp. 722–726, 2013.

[4] R. Zhang, Z. Peng, L. Wu, B. Yao, and Y. Guan, "Fault diagnosis from raw sensor data using deep neural networks considering temporal coherence," *Sensors*, vol. 17, no. 3, p. 549, 2017.

[5] S. Verma, N. Henwood, M. Castella, J.-C. Pesquet, and A. K. Jebai, "Can GANs recover faults in electrical motor sensors?," in *ICLR Workshop*, 2022.

[6] S. Verma, N. Henwood, M. Castella, J.-C. Pesquet, *et al.*, "Neural speed-torque estimator for induction motors in the presence of measurement noise," *IEEE TIE*, 2022.

[7] S. Verma, N. Henwood, M. Castella, J.-C. Pesquet, *et al.*, "Neural networks based speed-torque estimators for induction motors and performance metrics," in *IECON*, pp. 495–500, 2020.

[8] S. Verma, N. Henwood, M. Castella, F. Malrait, and J.-C. Pesquet, "Modeling electrical motor dynamics using encoder-decoder with recurrent skip connection," in *AAAI*, pp. 1387–1394, 2020.

[9] K.-C. Huang, H.-H. Yang, and W.-T. Chen, "Multi-scale aggregation with self-attention network for modeling electrical motor dynamics," in *IROS*, pp. 7097–7103, 2021.

[10] H.-H. Yang, K.-C. Huang, W.-T. Chen, and S.-Y. Kuo, "Lrg-net: Lightweight residual grid network for modeling electrical induction motor dynamics," in *EUSIPCO*, pp. 1536–1540, 2021.

[11] W. Kirchgässner, O. Wallscheid, and J. Böcker, "Estimating electric motor temperatures with deep residual machine learning," *IEEE TPEL*, vol. 36, no. 7, pp. 7480–7488, 2021.

[12] N. A. R. Maciejewski, A. E. Treml, and R. A. Flauzino, "A systematic review of fault detection and diagnosis methods for induction motors," in *ICEE*, pp. 86–90, 2020.

[13] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," *arXiv:1312.6199*, 2013.

[14] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *arXiv:1412.6572*, 2014.

[15] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," *arXiv:1706.06083*, 2017.

[16] A. Kurakin, I. Goodfellow, S. Bengio, *et al.*, "Adversarial examples in the physical world," 2016.

[17] Y. Dong, F. Liao, T. Pang, H. Su, J. Zhu, X. Hu, and J. Li, "Boosting adversarial attacks with momentum," in *CVPR*, pp. 9185–9193, 2018.

[18] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "Deepfool: a simple and accurate method to fool deep neural networks," in *CVPR*, pp. 2574–2582, 2016.

[19] S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard, "Universal adversarial perturbations," in *CVPR*, pp. 1765–1773, 2017.

[20] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *S&P*, pp. 39–57, 2017.

[21] E. R. Balda, A. Behboodi, and R. Mathar, "Perturbation analysis of learning algorithms: A unifying perspective on generation of adversarial examples," *arXiv:1812.07385*, 2018.

[22] J. Su, D. V. Vargas, and K. Sakurai, "One pixel attack for fooling deep neural networks," *IEEE TEC*, vol. 23, no. 5, pp. 828–841, 2019.

[23] V. Ballet, X. Renard, J. Aigrain, T. Laugel, P. Frossard, and M. Detyniecki, "Imperceptible adversarial attacks on tabular data," *arXiv:1911.03274*, 2019.

[24] K. Gupta, J.-C. Pesquet, B. Pesquet-Popescu, F. Kaakai, and F. Malliaros, "An adversarial attacker for neural networks in regression problems," in *IJCAI AI Safety Workshop*, 2021.

[25] N. Ng, K. Cho, and M. Ghassemi, "Ssmba: Self-supervised manifold based data augmentation for improving out-of-domain robustness," *ArXiv*, vol. abs/2009.10195, 2020.

[26] N. Ng, K. Cho, N. Hulkund, and M. Ghassemi, "Predicting out-of-domain generalization with local manifold smoothness," *ArXiv*, vol. abs/2207.02093, 2022.

[27] T. Zhou, Z. Ma, Q. Wen, X. Wang, L. Sun, and R. Jin, "Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting," *arXiv:2201.12740*, 2022.

[28] B. Shi, X. Bai, and C. Yao, "An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition," *TPAMI*, vol. 39, no. 11, pp. 2298–2304, 2017.

[29] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CVPR*, pp. 770–778, 2016.

[30] I. Radosavovic, R. P. Kosaraju, R. Girshick, K. He, and P. Dollár, "Designing network design spaces," in *CVPR*, pp. 10425–10433, 2020.

[31] A. C. Cameron and F. A. Windmeijer, "An R-squared measure of goodness of fit for some common nonlinear regression models," *Journal of Econometrics*, vol. 77, no. 2, pp. 329 – 342, 1997.

[32] F. Jadot, F. Malrait, J. Moreno-Valenzuela, and R. Sepulchre, "Adaptive regulation of vector-controlled induction motors," *IEEE TCST*, vol. 17, no. 3, pp. 646–657, 2009.

# A Appendix

## A Datasets

We use dataset proposed in [6] for **motor dynamics**, **denoise**, and **speed-torque** tasks. The dataset consists of following quantities currents ($i_d$, $i_q$), voltages ($u_d$, $u_q$), noisy currents ($\hat{i_d}$, $\hat{i_q}$), noisy voltages ($\hat{u_d}$, $\hat{u_q}$), rotor speed ($\omega_r$), and mechanical torque ($\tau_{em}$). The dataset consists of simulations using the control law proposed in [32]. The dataset contains 100 hours of simulation data which cover a wide range of operating trajectories. All three tasks are regression problems where some motor quantities are estimated from other quantities. In case of **motor dynamics** inputs are voltages ($u_d$, $u_q$) and rotor speed ($\omega_r$). The quantities that have to be predicted are currents ($i_d$, $i_q$) and mechanical torque ($\tau_{em}$). The **denoise** problem deals with denoising noisy currents ($\hat{i_d}$, $\hat{i_q}$), noisy voltages ($\hat{u_d}$, $\hat{u_q}$). For the **speed-torque** estimation we predict rotor speed ($\omega_r$), and mechanical torque ($\tau_{em}$) from currents ($i_d$, $i_q$) and voltages ($u_d$, $u_q$).

**Temperature** dataset [11] has been used for data-driven thermal modeling to remove or reduce the cost of placing thermal sensors deep inside moving parts of motors. It consists of different experiments where the temperature of the stator and rotor were measured in real operating conditions. Currents ($i_d$, $i_q$), voltages ($u_d$, $u_q$), speed ($\omega_r$), and torque ($\tau_{em}$) are the electrical motor quantities. Permanent magnet ($\vartheta_{PM}$), stator yoke ($\vartheta_{SY}$), stator tooth ($\vartheta_{ST}$), stator winding ($\vartheta_{SW}$), ambient temperature outside of stator ($\vartheta_a$), and coolant temperature ($\vartheta_c$) are the recorded temperatures. The objective is to predict permanent magnet ($\vartheta_{PM}$) temperature from all other quantities, making this a regression task.

**Broken bars** dataset [12] is a multiclass classification dataset. The data set contains currents, voltages, and torque as electrical signals. Accelerometers placed in 5 different motor parts are used to collect vibrations/mechanical signals. In total, 400 experiments, each lasting 20 seconds, are performed. The objective is to predict how many broken bars are in the motor from its electrical and mechanical signals.

## B Metrics

$$MAE(y, \hat{y}) = \frac{1}{T} \sum_{t=1}^{T} |y_t - \hat{y}_t| \tag{1}$$

$$SMAPE(y, \hat{y}) = \frac{100}{T} \sum_{t=1}^{T} \frac{|\hat{y}_t - y_t|}{|\hat{y}_t| + |y_t|} \tag{2}$$

$$R^2(y, \hat{y}) = 1 - \frac{\sum_{t=1}^{T}(\hat{y}_t - \bar{y})^2}{\sum_{t=1}^{T}(y_t - \bar{y})^2} \tag{3}$$

$$RMSE(y, \hat{y}) = \sqrt{\frac{1}{T}\sum_{t=1}^{T}(y_t - \hat{y}_t)^2} \tag{4}$$

where $y_t$ is the ground truth, $\hat{y}_t$ is the predicted output of the model at time $t$, and $T$ is the total experiment duration. $\bar{y}$ denotes the mean of ground truth $y$.

## C Results

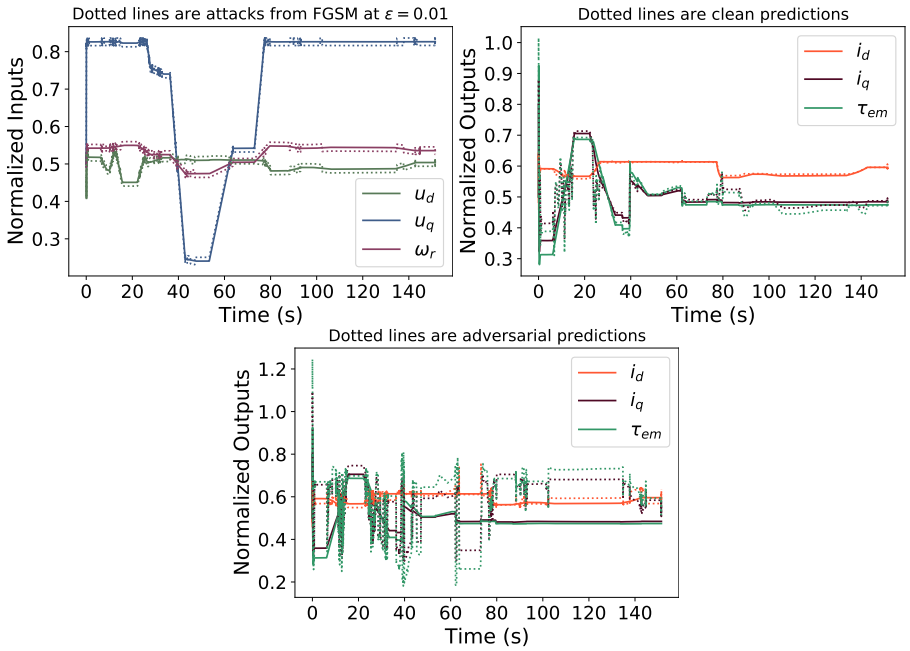


Figure 2: A sample from validation set of **motor dynamics** task showing clean input, clean output, DiagBiRNN-Skip clean prediction, FGSM generated adversarial example and adversarial prediction.

Figure 2 shows an example trajectory from the validation set of the **motor dynamics** dataset. In the top sub-figure normalized values of $u_d$, $u_q$, and $\omega_r$ are showed as normal lines. FGSM is used with $\epsilon = 0.01$ to attack the DiagBiRNN-Skip network to generate an adversarial example shown using the dotted lines in the same sub-figure. It can be seen that the adversarial example has an offset when compared to the clean input, but this offset can be positive and negative. There is also intermittent noise like big offsets, for example, one in the 90s. The middle sub-figure shows the ground truth of output signals $i_d$, $i_q$, and $\tau_{em}$ using normal lines and DiagBiRNN-Skip predictions using dotted lines. It can be seen that the network has poor predictions during the first ramp between 5s and 20s. The bottom figure shows ground truth as normal lines and dotted lines are the predictions of DiagBiRNN when the generated adversarial example is given as the input. It can be seen that the adversarial predictions are very noisy compared to clean predictions when the perturbations in the input are minimal. This shows that the attacks on the networks are strong and establishes that robustness study of such networks is important.

Tables 5 shows results obtained by simple networks proposed in [8] for **motor dynamics task**. It shows MAE, SMAPE, $R^2$, and RMSE for clean predictions and FGSM predictions at $\epsilon = 0.01$ and $\epsilon = 0.1$. CNN variant achieves the best clean predictions MAE (0.02), SMAPE (4.86%) and $R^2$ (0.76). When attacked, RNN is most robust at both epsilon values.

Table 6 shows results of the encoder-decoder variants trained for **denoise** task. RNN-Skip obtains the best SMAPE (0.14%) among all the variants. $R^2$ is 0.99 for all the networks. However, when

| Network | Attack | MAE | SMAPE(%) | $R^2$ | RMSE |
|---|---|---|---|---|---|
| **FNN** | **Clean** | 0.03 | 5.97 | 0.72 | 0.06 |
| | **FGSM** $\epsilon = 0.01$ | 0.12 | 24.06 | -1.57 | 0.17 |
| | **FGSM** $\epsilon = 0.1$ | 0.82 | 66.11 | -130.99 | 1.23 |
| **CNN** | **Clean** | **0.02** | **4.86** | **0.76** | **0.05** |
| | **FGSM** $\epsilon = 0.01$ | 0.11 | 22.02 | -1.05 | 0.15 |
| | **FGSM**$\epsilon = 0.1$ | 0.56 | 94.63 | -45.14 | 0.73 |
| **RNN** | **Clean** | 0.03 | 6.33 | 0.73 | 0.06 |
| | **FGSM** $\epsilon = 0.01$ | **0.11** | **21.91** | **-0.77** | **0.14** |
| | **FGSM** $\epsilon = 0.1$ | **0.22** | **45.53** | **-6.99** | **0.30** |
| **LSTM** | **Clean** | 0.03 | 5.05 | 0.75 | 0.05 |
| | **FGSM** $\epsilon = 0.01$ | 0.12 | 24.73 | -1.90 | 0.18 |
| | **FGSM** $\epsilon = 0.1$ | 0.39 | 65.19 | -32.92 | 0.63 |

Table 5: Metrics of clean and adversarial predictions from all the simple networks trained for **motor dynamics** task.

| Network | Attack | MAE | SMAPE(%) | $R^2$ | RMSE |
|---|---|---|---|---|---|
| **Deep** | **Clean** | 0.00 | 0.22 | 0.99 | 0.00 |
| | **FGSM** $\epsilon = 0.01$ | 0.01 | 1.63 | 0.99 | 0.01 |
| | **FGSM** $\epsilon = 0.1$ | 0.06 | 13.95 | 0.58 | 0.09 |
| **Skip** | **Clean** | 0.00 | 0.18 | 0.99 | 0.00 |
| | **FGSM** $\epsilon = 0.01$ | 0.01 | 1.59 | 1.00 | 0.01 |
| | **FGSM** $\epsilon = 0.1$ | 0.07 | 15.25 | 0.67 | 0.08 |
| **RNN-Skip** | **Clean** | **0.00** | **0.14** | **0.99** | **0.00** |
| | **FGSM** $\epsilon = 0.01$ | 0.01 | 1.76 | 1.00 | 0.01 |
| | **FGSM** $\epsilon = 0.1$ | 0.08 | 16.16 | 0.66 | 0.08 |
| **BiRNN-Skip** | **Clean** | 0.00 | 0.15 | 0.99 | 0.00 |
| | **FGSM** $\epsilon = 0.01$ | 0.01 | 1.59 | 1.00 | 0.01 |
| | **FGSM** $\epsilon = 0.1$ | 0.06 | 14.04 | 0.71 | 0.08 |
| **DiagBiRNN-Skip** | **Clean** | 0.00 | 0.18 | 0.99 | 0.00 |
| | **FGSM** $\epsilon = 0.01$ | **0.01** | **1.53** | **1.00** | **0.01** |
| | **FGSM** $\epsilon = 0.1$ | **0.07** | **13.6** | **0.72** | **0.08** |

Table 6: Metrics of clean and adversarial predictions from all the networks trained for motor **denoise** task.

attacked with FGSM at $\epsilon = 0.01$, DiagBiRNN-Skip outperforms every other network with the lowest SMAPE (1.53%). With a more aggressive attack of $\epsilon = 0.1$, DiagBiRNN-Skip still outperforms every other network with SMAPE (13.6%) and $R^2$ (0.72).