Adversarial Attacks on Time Series Data

Inci Baytaş, Afra Akbaş, and Ceren Tahtasız, Boğaziçi University, Istanbul, Turkey

Email: {first-name.last-name@boun.edu.tr}

I. Introduction

Machine Learning has increasing popularity and presence in our lives. Businesses and individuals mostly use machine learning techniques to forecast trends, classify data, detect abnormalities. These techniques led to the emergence of various areas and applications such as anomaly detection, biometric authentication, and self-driving cars. For these tasks, massive amounts of time-series data are collected, classified, and rendered. Using time-series data in deep learning models is very common, and many researchers try to adopt them in various fields. However, the evolution of these models gives rise to new concerns. People that rely on deep learning models should also consider security and attack tolerance of the models.

One common attack type in deep learning models is adversarial attacks. An adversarial attack perturbates the input data very little that humans can't detect the difference. However, this difference causes the model to output an incorrect answer with high confidence. Several machine learning models, including state-of-art neural networks, are vulnerable to these kinds of attacks. Szegedy et al. [6] gave adversarial examples for the first time in 2014. Their research created a tremendous impact on deep learning approaches. After that, adversarial attacks became a global research topic, but most of the research groups are focused on the scenarios in which input is an image dataset. There's a lack of research for time-series datasets. In this research, we're studying adversarial attacks and simulate them on time-series data. Therefore, we will examine LSTM model robustness which is the most suitable model for time series data, unlike other research papers that deal with CNN for images.

Our motivation is to raise awareness of the security vulnerabilities of machine learning models that use time series data as input. This research will expose blind spots by generating adversarial attacks for classification and regression problems. Perturbated results will be used to make models more robust by training. Results of this research could lead to the design of new machine learning algorithms that are prone to adversarial attacks. Conclusions of our studies may interest machine learning service providers, research groups, and businesses to address security concerns.

II. Related Work

In this section, we propose the work already done. There are several papers written on the adversarial attacks. The first examples of adversarial attacks are in an image recognition task. Their task is building a classification model with CNN architecture, and observing the behavior of the model with an attack. After they have had a well-designed model with high accuracy, they prepared their adversarial data. The data was the same but included a perturbation. They found that with a small perturbation, the results were quite intriguing. [6]

After this research, research areas had expanded and many new research topics emerged. Health care systems pay real attention to this topic since an adversarial attack on patients' data will result in a disaster. It violates safety and causes reliability concerns. In the paper, they tried to examine adversarial attacks on medical images. The images used for cancer diagnosis or some diseases' detection can be easily manipulated by an imperceptible perturbation. Therefore, their medical DNN model is vulnerable to such attacks. To understand these attacks further, they created their adversarial images with state-of-art attack methods. These are Fast Gradient Sign Method (FGSM), Basic Iterative Method (BIM), Projected Gradient Descent method (PGD), and Carlini and Wagner Attack (CW). They applied these attacks and then tried to detect them. [7]

On the other hand, there are few papers about the adversarial attacks on time-series data. Karim et al. [1] worked on time-series classification models and applied their techniques with 42 datasets, which are from the University of California Riverside, onto 1-NN Dynamic Time Wrapping (DTW) and Fully Connected Network (FCN). Their work showed

us all these datasets are vulnerable by these attacks. However, to the best of our knowledge, we know that 1-NN DTW with FCN is not an advanced TSC model today, so these works cannot be considered as sufficient yet. There should be more research to detect and prevent it. Therefore, to add a new perspective, we proposed several attacks in LSTM structures for time-series data so that we take this in a more state-of-art manner.

III. Methods

For time-series data, the most suitable network is Recurrent Neural Network (RNN). It's a specific version of an artificial neural network. RNN's use their internal state (memory) to process sequences of inputs, while in other neural networks, all inputs are independent of each other. A general problem with RNN is called Vanishing Gradient Problem. Gradients carry information that updates RNN parameters. In long data sequences, while moving to initial layers of the network, the gradient becomes smaller in each step. When the gradient is extremely small, parameter updates become insignificant. The model cannot learn the parameters. Even a big difference in the value of parameters does not affect the output as much as it should do.

LSTM networks, a special kind of RNN, solves this problem. As a modified version of RNN, LSTMs have additional units such as memory cells that can store information for a long time. It has 3 gates (forget, input, and output) to control the cell states. This architecture allows for solving long term dependencies by memorizing previous data easily. In our project, we're attacking an LSTM to use a state-of-the-art model.

Part 1) Univariate Linear Regression

As the first step of our project, we chose a time-series dataset that suits to our research. For simplicity, we looked for a dataset that is univariate. Working with univariate datasets is easier than multivariate because results are easier to plot and interpret. For high accuracy, we looked for a dataset that has enough (3000+) data points. After some research through popular databases, we found hourly energy consumption data from Kaggle that contains 14 years of data. It has more than 120000 hours of data on American energy consumption. It was one of the most extensive datasets that we found in time-series data. When we piece together it into daily energy consumption, it has more than 5000 points. Since our aim is to build an RNN model with time-series data, we converted the data into a time-series form.

After we decided on our dataset, we loaded our data and prepared it for the model. We normalized the data and split it into train and test sets. We decided to use %80 of the data for training and %20 for testing. We also reshaped our x-values and y-values to convert it into a time-series format. For each day, x-values has the first 23 hourly data, and y-values has the last hour.

Using the dataset, we built our RNN model and implemented an adversarial attack. We used one LSTM layer, one Dropout layer, and one Dense layer. LSTM layer is for our time-series data. To prevent the overfitting during the training, we added a Dropout layer after LSTM. For the regression part, we have a Dense layer at the end with one unit. We chose the optimizer Adam and the loss function MSE (mean squared error). Then we trained our model and tested it. The R2 score shows us the accuracy score of our prediction. Our model predicts the 24th values with nearly 0.90-accuracy.

The next move is generating an adversarial attack on our model. We did some research for examples of it, and we found different approaches. We tried to apply Fast Gradient Sign Attack (FGSM) - one of the most commonly used techniques - in two ways. In the first part, we applied FGSM in the NumPy form. On the other hand, we used TensorFlow features in the second.

However, observing the impact of the attacks is not easy in regression models. Therefore, we decided to move forward to our research with a classification model.

Part 2) Univariate Linear Classification

We used EEG Eye State Dataset from the UCI Machine Learning Repository. It has 14980 data points. Each data point consists of 14-time steps of the sensor data. The result of this monitoring is a binary value which suggests if the eye is open (0) or closed (1). After data preparation, we used %70 of data for training and %30 for testing. We used one LSTM

layer, one Dropout layer, and one Dense layer. To reach a high level of accuracy, we had used a modified Adam optimizer with a higher learning rate with a high epoch number. When we tested our model, it predicts the eye-state with accuracy 0.82. Then we applied Fast Gradient Sign Attack (FGSM) using TensorFlow features. We generated adversarial samples with epsilon 0.2. Although the trend of data values did not change too much, the accuracy dropped to 0.64.

Part 3) Multivariate Classification

We decided to use the MNIST dataset for the multivariate classification task since it gives us the opportunity of visualizing attacks. The dataset contains 70,000 hand-written-digit images, and each image is 28*28 matrix. We interpreted each line as a feature and each column as a time step. Then, we split the data as 60,000 for training and 10,000 for testing. Our sequential model has one LSTM layer, one Dropout layer, and one Dense layer with the activation sigmoid. For the training, we used a modified version of Adam optimizer with a high learning rate again, but this time we decrease the number of epochs. The accuracy score of the original data is 0.98. To generate adversarial attacks, we used the FGSM and Basic Iterative Method. In FGSM, we chose the epsilon as 0.2, and the accuracy dropped to 0.18. On the other hand, in the Basic Iterative method, we applied attack in 5 iterations with epsilon 0.1 and alpha 0.05. The accuracy dropped more than FGSM (0.12), even though the epsilon value is smaller.

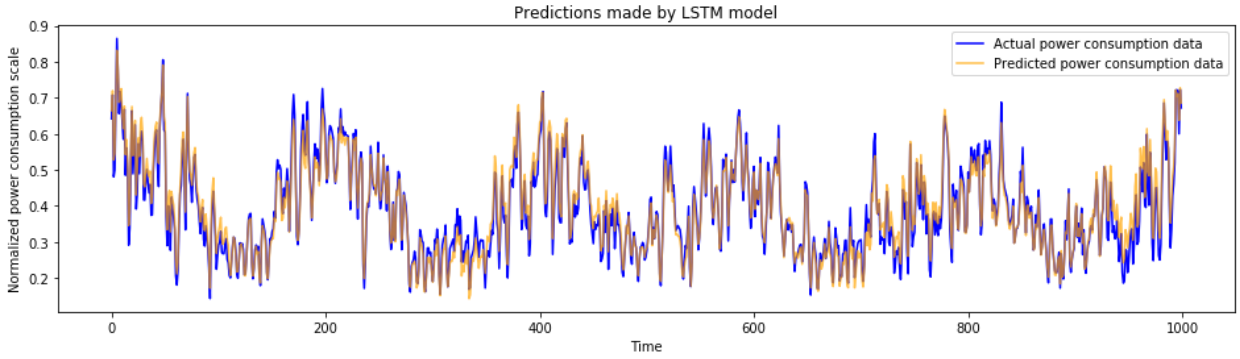
IV. Experiments

Part 1) Univariate Linear Regression

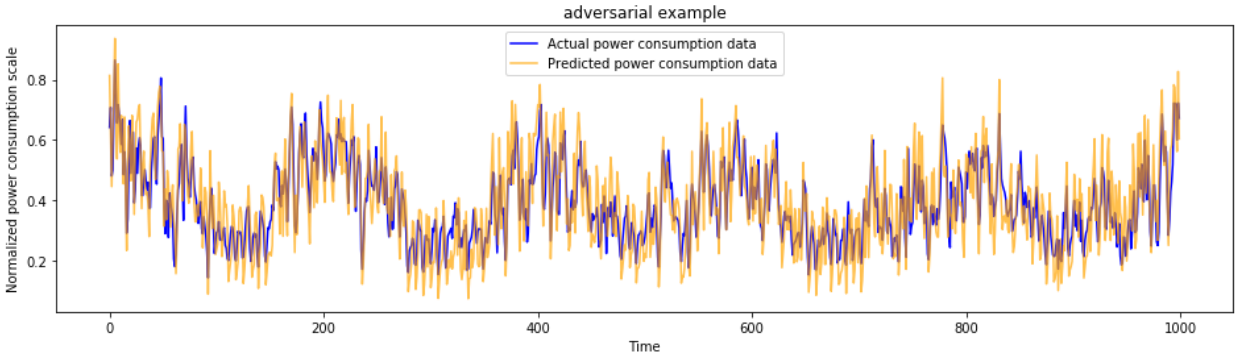
The first result that we gain is with the actual data. It gives us a 0.94 accuracy score.

You can find the model and experiments here: https://www.kaggle.com/afraakbas/kernel24ddf1b96c

Here is the graph of the prediction that we make and the real values:



Example of prediction and real values:
prediction => [0.65382934 0.7030181 0.51656264 0.5309163 0.695841130.8274534
0.76988006 0.67851347 0.69814914 0.71051574]
real values => [0.64223656 0.70820405 0.48088619 0.50018617 0.70212238 0.86490009 0.7600844 0.65619958 0.71683009
0.6853047]



FGSM with NumPy form:

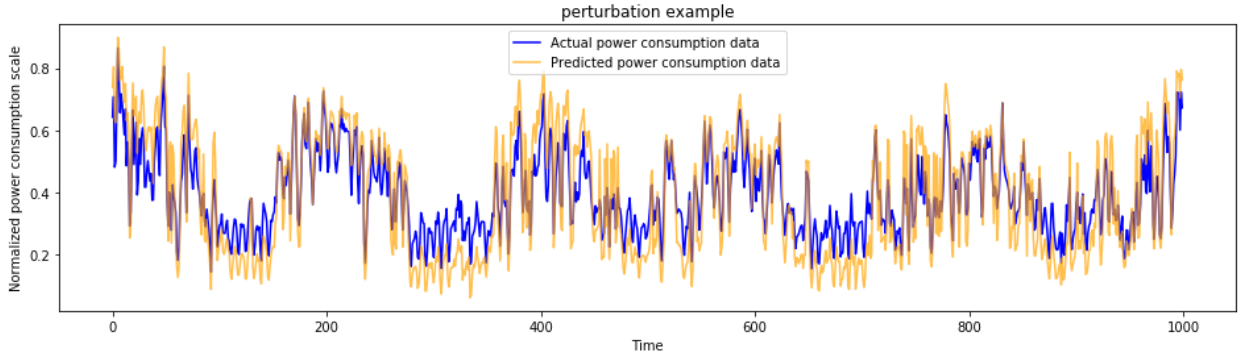
- If we choose the epsilon=0.1, our accuracy drops to 0.516222475407115.

actual values =>[[[0.51632121][0.47648008][0.62380539][0.6175996
][0.61611021][0.62777709][0.65049026][0.70659054][0.77845352][0.80848951][0.8020355
][0.76840015][0.72229118][0.68741467][0.65185553][0.63156262][0.61890282][0.5934591
][0.59383145][0.61381407][0.6602954][0.68052625][0.67010053]]]

adversarial data values => [[[0.61632121][0.57648008][0.52380539][0.5175996
][0.51611021][0.52777709][0.55049026][0.60659054][0.67845352][0.70848951][0.7020355
][0.66840015][0.62229117][0.58741467][0.55185553][0.53156261][0.51890282][0.4934591
][0.49383145][0.51381407][0.7602954][0.78052625][0.77010054]]]

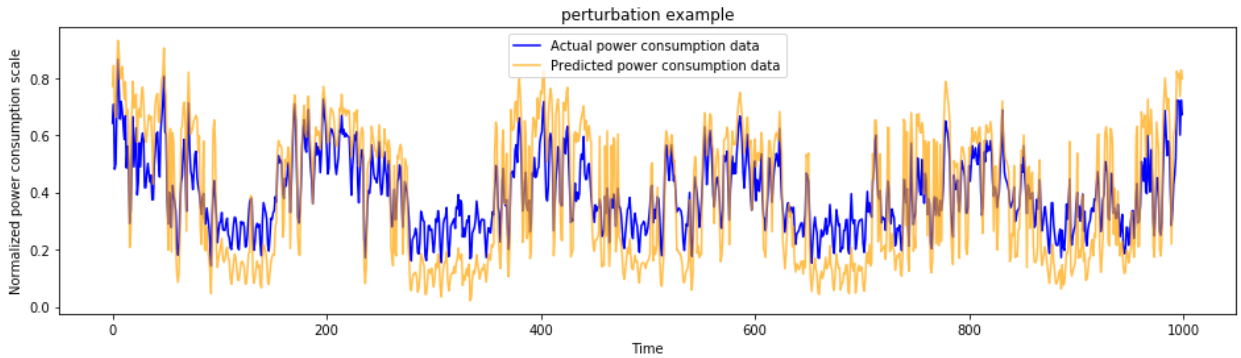
FGSM with TensorFlow:

- If we choose the epsilon=0.1, our accuracy drops to 0.49598275240824385.



actual values => [[[0.51632121][0.47648008][0.62380539][0.6175996
][0.61611021][0.62777709][0.65049026][0.70659054][0.77845352][0.80848951][0.8020355
][0.76840015][0.72229118][0.68741467][0.65185553][0.63156262][0.61890282][0.5934591][0.59383145][0.61381407][0.6602954
][0.68052625][0.67010053]]]

perturbed data values => [[[0.6163212][0.5764801][0.7238054][0.71759963][0.7161102][0.7277771][0.7504903
][0.80659056][0.87845355][0.9084895][0.90203553][0.86840016][0.8222912][0.78741467][0.75185555][0.7315626][0.7189028
][0.69345915][0.6938315][0.7138141][0.7602954][0.7805263][0.77010053]]]

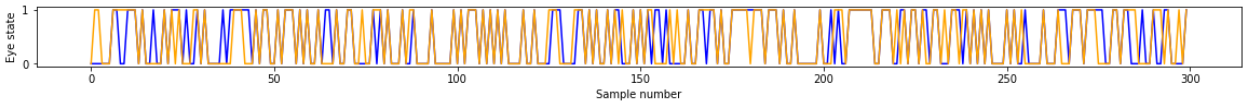


- If we choose the epsilon=0.15, our accuracy drops -0.025727959232535014.

actual values => [[[0.51632121][0.47648008][0.62380539][0.6175996
][0.61611021][0.62777709][0.65049026][0.70659054][0.77845352][0.80848951][0.8020355
][0.76840015][0.72229118][0.68741467][0.65185553][0.63156262][0.61890282][0.5934591][0.59383145][0.61381407][0.6602954
][0.68052625][0.67010053]]]
perturbed data values => [[[0.66632116][0.6264801][0.7738054][0.7675996][0.7661102][0.7777771][0.80049026][0.8565905
][0.92845356][0.95848954][0.95203555][0.91840017][0.8722912][0.8374146][0.80185556][0.78156257][0.7689028][0.7434591
][0.7438315][0.7638141][0.81029534][0.83052623][0.82010055]]]

Part 2) Univariate Linear Classification

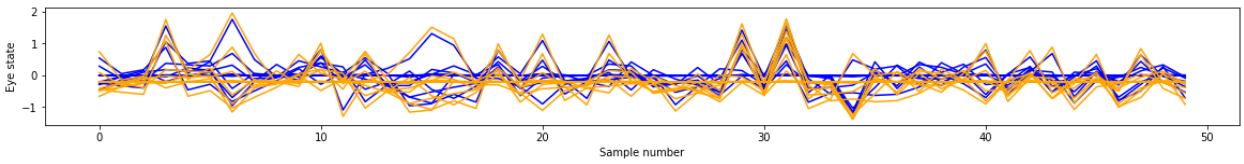
The trained model had %97 accuracy but the accuracy of the model on the test data is %82. We believe this is due to overfitting and we will work on improving this rate. Results before the attack:

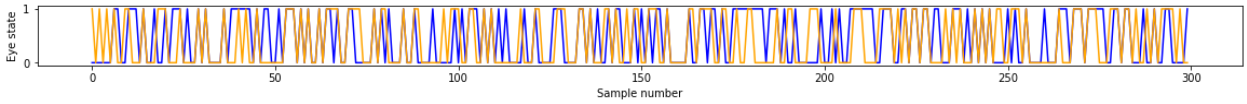


You can find the model and experiments here: https://www.kaggle.com/cerentahtasiz/eye-data-prediction

After normalizing and perturbating the input, you can compare the original input and perturbated input below.

As a result, the test results' accuracy had dropped to %65. After attack:



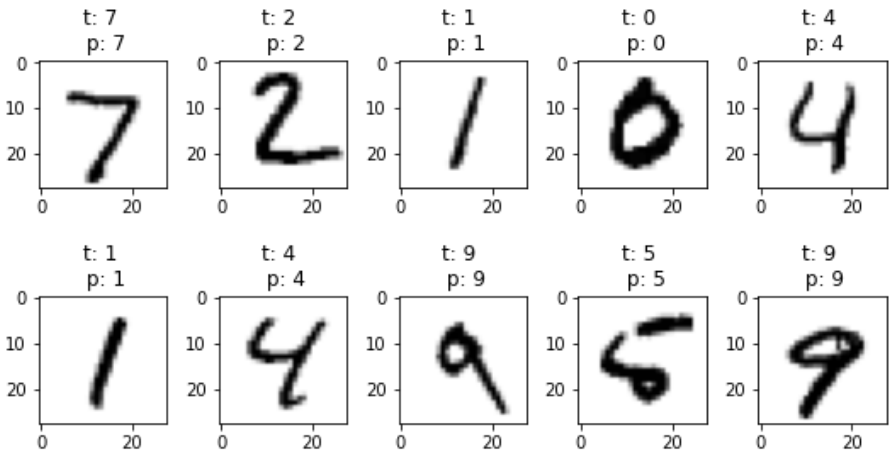


Part 3) Multivariate Classification

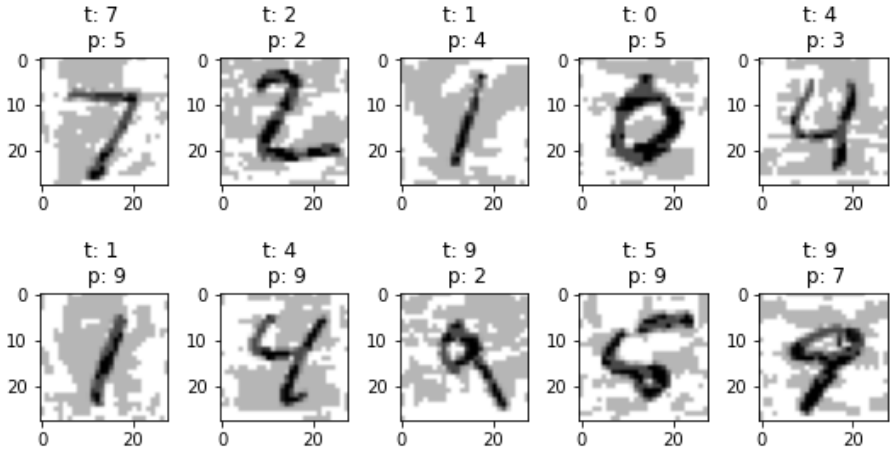
The accuracy score of the model is 0.98, with the original data.

You can find the model and experiments here: https://www.kaggle.com/afraakbas/mnsit

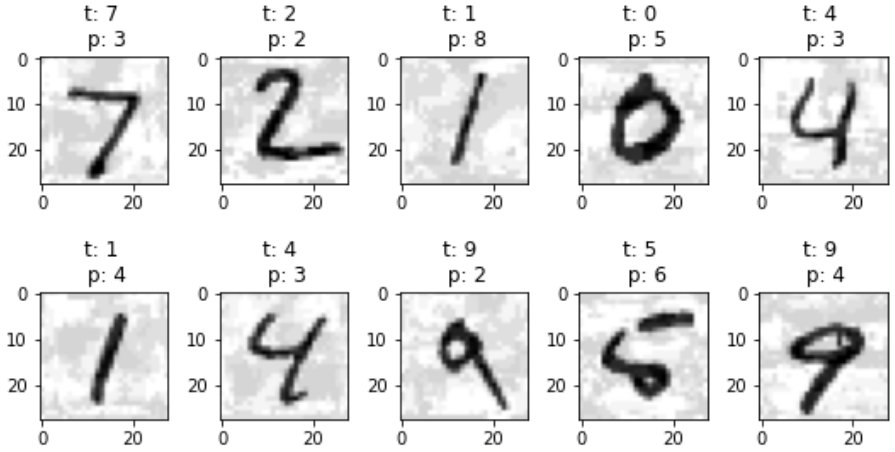
Without an attack:



- When we applied the FGSM method with epsilon = 0.2, the accuracy had dropped 0.1868.



- With the Basic Iterative method, we applied a perturbation in five iterations with epsilon = 0.1 and alpha=0.05, the accuracy results in 0.14.



*You can find whole projects in our Github repository:https://github.com/cerentahtasiz/adversarial-attacks.

V. Conclusion

In this paper, we tried to examine the adversarial attacks on time-series data. We had simulated two commonly used adversarial-attack methods, Fast Gradient Sign Attacks (FGSM) and Basic Iterative Method, on LSTMs for both regression and classification models using various datasets. We noted that a small perturbation on the input dataset results in sharp drops on the accuracy rates of the models, which means that even a state of art ML model, LSTM is vulnerable to adversarial attacks for time series data. To make robust models, machine learning scientists must also consider adversarial attacks. In our studies, we also saw that iteratively simulating attacks results in lower accuracy rates than at one time. For further studies, this should also take into account.

VI. Future Works

Adversarial attacks could be simulated on multivariate datasets for linear regression tasks. Since observing adversarial attacks on regression models is harder than observing them on classification models, audio datasets could be used for regression tasks. In this way, generated attacks can become more evident as image data. Our experiments could be repeated on more datasets to demonstrate the vulnerabilities. The results of this study could be used to build models that are more prone to adversarial attacks and new machine learning algorithm designs.

VII. References

[1] Adversarial Attacks on Time Series by Fazle Karim, Graduate Student Member, IEEE Somshubra Majumdar2, and Houshang Darabi1, Senior Member, IEEE

[2] Adversarial Attacks on Deep Neural Networks for Time Series Classification by Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar and Pierre-Alain Muller IRIMAS, Universite´ Haute-Alsace, Mulhouse, France

[3] Explaining and Harnessing Adversarial Examples by Ian J. Goodfellow, Jonathon Shlens & Christian Szegedy, Google Inc., Mountain View, CA

[4] Crafting Adversarial Input Sequences for Recurrent Neural Networks by Nicolas Papernot and Patrick McDaniel, The Pennsylvania State University, University Park, PA and Ananthram Swami and Richard Harang, United States Army Research Laboratory Adelphi, MD

[5] Adversarial Examples in Modern Machine Learning: A Review by Rey Reza Wiyatno Anqi Xu Ousmane Dia Archy de Berker, Element AI, Montréal, Canada

[6] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, “Intriguing properties of neural networks,” ArXiv, 2014.

[7] X. Ma, Y. Niu, L. Gu, Y. Wang, Y. Zhao, J. Bailey, F. Lu, “Understanding Adversarial Attacks on Deep Learning-Based Medical Image Analysis Systems,”, ArXiv, 2014.

Tutorial: https://machinelearningmastery.com/time-series-forecasting-long-short-term-memory-network-python/
Datasets: https://www.kaggle.com/robikscube/hourly-energy-consumption#AEP_hourly.csv
http://archive.ics.uci.edu/ml/datasets/EEG+Eye+State
Our github reporsitory: https://github.com/cerentahtasiz/adversarial-attacks