Impacto do Aprendizado de Máquina Adversário contra Detectores de Anomalias em Séries Temporais *

Felipe Dallmann Tomazeli[1], **Gilberto Fernandes Junior**[1], **e Bruno Bogaz Zarpelão**[1]

[1]Departamento de Computação – Universidade Estadual de Londrina (UEL)
Rod. Celso Garcia Cid, s/n – 86.057-970 – Londrina – PR – Brasil

{felipe.dallmann, gilfernandes, brunozarpelao}@uel.br

Abstract. *Anomaly detection can be employed in time series to automatically identify faults, outages, and misuse in devices, services, and systems. Machine learning algorithms have been successfully applied to detect anomalies in time series of various natures. However, these algorithms are vulnerable to Adversarial Machine Learning attacks, which can result in anomalies not being detected, or normal situations being erroneously detected as anomalies, generating false positives. In light of this reality, this work investigates how attacks based on adversarial examples can impact an anomaly detection model based on a Long Short-Term Memory (LSTM) neural network. Within the scope of this study, two methods of generating adversarial examples are tested: one based on the addition of noise calculated over the standard deviation and another based on the Fast Gradient Sign Method (FGSM) technique. The results showed that the anomaly detection model experiences a decrease in its predictive capability when attacked, but outperforms a classifier based on a Multi-layer Perceptron (MLP) neural network under the same conditions.*

Resumo. *A detecção de anomalias em séries temporais pode ser empregada para identificar automaticamente falhas, interrupções, e uso inadequado em dispositivos, serviços e sistemas. Algoritmos de aprendizado de máquina têm sido aplicados com sucesso para detectar anomalias em séries temporais de diversas naturezas. No entanto, é importante considerar que esses algoritmos são vulneráveis a ataques baseados em Aprendizado de Máquina Adversário, o que pode fazer com que anomalias não sejam detectadas, ou que situações normais sejam erroneamente detectadas como anomalias, gerando falsos positivos. Diante desta realidade, este trabalho investiga como ataques baseados em exemplos adversários podem impactar um modelo de detecção de anomalias baseado em uma rede neural Long Short-Term Memory (LSTM). No escopo deste estudo, são testados dois métodos de geração de exemplos adversários, um baseado na adição de ruído calculado sobre o desvio padrão e outro baseado na técnica Fast Gradient Sign Method (FGSM). Os resultados mostraram que o modelo baseado em detecção de anomalias tem queda de capacidade preditiva quando atacado, mas supera um classificador baseado em uma rede neural Multi-layer Perceptron (MLP) sob as mesmas condições.*

*Os autores Felipe D. Tomazeli e Bruno B. Zarpelão agradecem o apoio financeiro da Fundação Araucária de Apoio ao Desenvolvimento Científico e Tecnológico do Estado do Paraná para a execução deste trabalho por meio do PROIC e do NAPI Norte - CIA Agro, respectivamente.

1. Introdução

As séries temporais são amplamente utilizadas em diversas áreas do conhecimento [Kim and Kumar 2012], incluindo finanças, medicina, engenharia, meteorologia, entre outras. A principal característica das séries temporais é a sua dependência temporal, ou seja, cada observação pode estar relacionada em diferentes níveis a observações anteriores e às subsequentes. Trabalhos recentes mostram que algoritmos de aprendizado de máquina são capazes de modelar precisamente séries temporais, considerando suas particularidades como tendências e sazonalidade. Dessa forma, modelos baseados em algoritmos como LSTM (*Long Short-Term Memory*) podem alcançar bons resultados quando aplicados a previsão de valores futuros, identificação de padrões e detecção de anomalias em séries temporais [Barrera-Animas et al. 2022, Zhang et al. 2024, Khan et al. 2023].

Anomalias em séries temporais podem representar eventos de interesse para aqueles que as consomem ou analisam [Chandola et al. 2009]. Elas podem sinalizar, por exemplo, uma falha em um equipamento ou uma mudança inesperada no comportamento do fenômeno observado. A detecção e tratamento de anomalias colaboram para garantir a confiabilidade dos sistemas. Como mencionado acima, algoritmos de aprendizado de máquina podem ser eficazes na detecção de anomalias em séries temporais. Eles são treinados com dados históricos para aprender padrões normais e, assim, detectar quaisquer valores ou eventos que fogem desses padrões. Esses algoritmos podem se adaptar às mudanças no comportamento da série temporal, melhorando a precisão da detecção de anomalias com o tempo [Choi et al. 2021].

Um problema neste cenário é a vulnerabilidade de algoritmos de aprendizado de máquina a ataques baseados em Aprendizado de Máquina Adversário, que podem afetar a sua eficácia, comprometendo a sua funcionalidade. Dentre esses ataques, temos os ataques de evasão. Eles consistem em inserir pequenas perturbações nos dados originais, produzindo exemplos adversários a fim de enganar o algoritmo de detecção de anomalias. Essas perturbações podem ser imperceptíveis para os humanos, mas são suficientes para causar erros na detecção de anomalias pelo algoritmo de Aprendizado de Máquina. Como resultado, anomalias importantes podem ser ignoradas e eventos normais podem ser erroneamente detectados como anomalias [Carlini and Wagner 2017, Huang et al. 2011].

Diante do exposto, o presente trabalho tem como objetivo analisar os impactos de exemplos adversários contra detectores de anomalias em séries temporais. Nesse sentido, espera-se avaliar a gravidade desses ataques, considerando a magnitude da perturbação aplicada pelo atacante e as diferentes técnicas utilizadas. Para alcançar o objetivo proposto, um modelo de detecção de anomalias baseado em LSTM será submetido aos ataques. Esse modelo é um algoritmo de Aprendizado de Máquina, que tem por objetivo realizar predições de uma série temporal. Com base nessas predições, um limiar será definido a partir do erro quadrático médio entre as predições e os valores reais, permitindo a detecção de anomalias na série temporal. Os ataques serão executados para gerar falsos positivos e falsos negativos. Também será criado um segundo modelo alvo, utilizando um classificador baseado em uma MLP (*Multi-layer Perceptron*), para servir como linha de base e posteriormente ser comparado com o modelo de detecção de anomalias. Os ataques serão criados a partir da técnica FGSM (*Fast Gradient Sign Method*) e um método baseado no desvio padrão dos dados, denominado *naive*.

Este trabalho está organizado da seguinte forma. A Seção 2 apresenta os trabalhos

relacionados. Na seção 3, é descrita a metodologia empregada para realização dos experimentos que compõem a investigação. A Seção 4 apresenta os resultados obtidos. Por fim, na Seção 5, são apresentadas as conclusões do estudo.

2. Trabalhos Relacionados

Os estudos pioneiros na área de geração de exemplos adversários contra redes neurais se concentraram principalmente em alvos que trabalhavam com a classificação de imagens. Com a disseminação dessas descobertas, os pesquisadores passaram a explorar alvos dedicados a outros tipos de dados, como séries temporais.

Harford et al. [2021] propuseram uma metodologia de geração de exemplos adversários voltada a classificadores de séries temporais multivariadas. Primeiramente, utiliza-se a técnica de destilação para criar um modelo de rede neural compacto que reproduza o modelo alvo. Esse modelo compacto é, então, usado no núcleo de uma GATN (*Gradient Adversarial Transformation Network*), que é treinada para receber como entrada uma série temporal legítima e produzir em sua saída uma versão modificada que engane o alvo. Testes foram realizados contra os classificadores 1-*nearest neighbor dynamic time warping* (1NN-DTW) e *fully convolutional network* (FCN), considerando problemas de classificação com classes binárias e multi-classes. Os resultados mostraram que os ataques foram efetivos, principalmente contra os alvos baseados em FCNs.

Sadeghzadeh et al. [2021] estudaram o impacto de ataques baseados em perturbações universais contra diferentes tipos de classificadores de tráfego de redes. Eles criaram um ataque específico para classificadores de séries temporais denominado AdvBurst, que adiciona, ao final de uma rajada, pacotes com propriedades estatísticas perturbadas para enganar o classificador. Nos testes, o impacto do AdvBurst foi comparado com o RandBurst, que escolhe aleatoriamente propriedades estatísticas dos pacotes para serem perturbadas. Os resultados mostraram que o AdvBurst foi mais efetivo por ampla margem. A perturbação universal também foi a base do ataque criado por Khan et al. [2024], denominado Adapt Edge, que atua contra redes neurais que classificam distúrbios de qualidade de energia elétrica em séries temporais. Testes foram realizados em um conjunto de dados com 17 tipos de distúrbios de qualidade de energia. Os classificadores alvo foram uma rede neural convolucional, uma LSTM e uma ResNet50. Todas elas apresentaram uma queda significativa na acurácia para ataques com diferentes magnitudes.

Outros pesquisadores mantiveram o foco em problemas de classificação, mas trabalharam apenas com situações envolvendo classes binárias. Gallagher et al. [2022] , por exemplo, investigaram o impacto de três tipos de ataques contra uma rede neural convolucional que prevê se um ativo financeiro será vendido ou comprado a partir de sua série histórica de preços. Os dois primeiros tipos ataques foram baseados no FGSM e em uma variação dele. O terceiro tipo é um ataque de envenenamento, no qual dados de treinamento tiveram seus rótulos invertidos para contaminar o modelo da rede neural. Os resultados de testes realizados sobre séries temporais reais mostraram que a capacidade preditiva do modelo alvo foi bastante prejudicada por estes ataques.

Zhou et al. [2022] focam na detecção de anomalias utilizando um classificador binário. A rede neural alvo é modelada para receber leituras de sensores e ações de atuadores, e prever se o estado do sistema será seguro ou inseguro. Para os testes, os autores construíram redes neurais baseadas nos métodos MLP e LSTM, assim como variações

que incluíam conhecimento específico sobre o domínio na forma de *semantic loss functions*. Os resultados mostraram que essas variações foram mais robustas a ataques do tipo FGSM e baseados em ruído Gaussiano. Jia et al. [2021] investigaram a robustez de um detector de anomalias baseado em LSTM e CUSUM contra ataques baseados na análise do gradiente. No detector de anomalias proposto, a LSTM prevê o próximo elemento de uma série temporal e calcula-se o resíduo entre esta previsão e o valor realmente ocorrido. O algoritmo CUSUM é então aplicado para detectar mudanças de comportamento nos resíduos. Resultados mostraram que o ataque foi capaz de degradar o desempenho preditivo do detector de anomalias proposto.

A análise dos trabalhos mostra que grande parte ainda se concentra na classificação de séries temporais. Apenas um trabalho, de Jia et al. [2021], buscou entender o impacto de exemplos adversários em um detector de anomalias que não se baseia em um classificador. O presente trabalho também segue esta direção, mas, diferente de [Jia et al. 2021], não considera o uso do algoritmo CUSUM. Além disso, este trabalho apresenta testes com ataques FGSM e um outro ataque baseado no desvio padrão da série. Ao comparar o impacto em um detector de anomalias baseado em LSTM com um classificador baseado em MLP, foi possível observar como a modelagem da solução pode influenciar em sua robustez, já que ataques baseados em exemplos adversários são normalmente projetados levando em conta problemas de classificação.

3. Materiais e Métodos

O objetivo deste trabalho é analisar o impacto de exemplos adversários em um modelo de detecção de anomalias em séries temporais baseado no algoritmo LSTM. Especificamente, o estudo investiga se um ataque projetado para ser lançado contra classificadores também pode ser eficaz contra modelos de detecção de anomalias. Para isso, além do detector de anomalias baseado em LSTM, implementamos um modelo alvo utilizando uma rede neural do tipo MLP. Este modelo aborda a detecção de anomalias como um problema de classificação, dividindo as observações nas classes normal e anormal. O modelo alvo baseado em MLP servirá como referência (linha de base) para comparar o impacto dos ataques em modelos de detecção de anomalias e classificadores.

Os modelos adversários serão baseados em duas estratégias. Primeiramente, faremos um ataque que denominamos *naive*, no qual o atacante adicionará um ruído às observações obtido a partir do desvio padrão dos dados de treinamento. O segundo ataque será o FGSM, normalmente utilizado para comprometer classificadores, onde o atacante usará um gradiente descendente para maximizar a perturbação adicionada aos dados de entrada. A Figura 1 apresenta a visão geral do experimento proposto.

3.1. Conjunto de dados

O conjunto de dados utilizado neste experimento foi retirado de um *benchmark* público e gratuito [Katser and Kozitsin 2020] de detecção de anomalias em séries temporais. Os dados provêm da leitura de sensores posicionados em um sistema de circulação de água, contendo 8 atributos, cada um representando um dos sensores:

- **Accelerometer1RMS**: mostra a aceleração de vibração.
- **Accelerometer2RMS**: mostra a aceleração de vibração.
- **Current**: mostra a amperagem do motor elétrico.

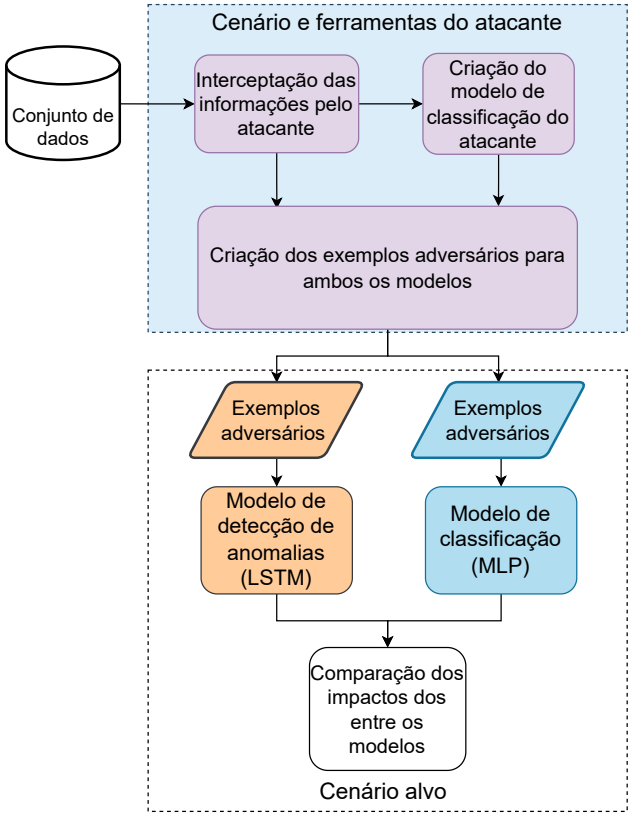

Figura 1. Visão geral dos cenários do atacante e do alvo.

- **Pressure**: representa a pressão no circuito depois da bomba de água.
- **Temperature**: mostra a temperatura do corpo do motor.
- **Thermocouple**: representa a temperatura do fluído no circuito de circulação.
- **Voltage**: Mostra a tensão no motor elétrico.
- **RateRMS**: representa a vazão de circulação do fluído dentro do circuito.

O conjunto de dados é composto por diversos subconjuntos, dos quais foi selecionado aquele que apresenta falhas ligadas ao comportamento exponencial do desequilíbrio do rotor. O subconjunto selecionado contém um total de 751 observações, sendo 557 rotuladas como normais e 194 como anomalias. Os dados foram padronizados utilizando o algoritmo Min-Max, limitando os valores entre 0 e 1 com base nos dados de treinamento.

3.2. Métricas de avaliação

As métricas utilizadas para avaliar a eficácia da detecção de anomalias neste trabalho são derivadas de contagens resumidas em uma matriz de confusão: VP representa a quantidade de observações classificadas corretamente como anomalias, VN representa a quantidade de observações classificadas corretamente como normais, FP representa a quantidade de observações classificadas incorretamente como anomalias e, por fim, FN representa a quantidade de observações classificadas incorretamente como normais.

A partir dessas contagens, calcula-se as métricas a seguir. TFP avalia a proporção de amostras normais que foram erroneamente detectadas como anomalias ($TFP = \frac{FP}{(FP+VN)}$), enquanto TFN avalia a proporção de anomalias não detectadas

($TFN = \frac{FN}{(VP+FN)}$). A precisão avalia a proporção de anomalias corretamente detectadas em relação à quantidade de amostras genuinamente anômalas ($prec = \frac{VP}{(VP+FP)}$). A revocação avalia a proporção de anomalias corretamente detectadas em relação ao total de anomalias existentes ($revoc = \frac{VP}{(VP+FN)}$). A métrica f1-score é uma combinação das métricas precisão e revocação. De maneira geral, mede o grau de assertividade, realizando uma média harmônica entre as duas conforme a equação $f1 = 2 \times (\frac{prec \times revoc}{prec+revoc})$. Por fim, a métrica Matthews Correlation Coefficient (MCC) é considerada mais balanceada e pode ser utilizada nos casos em que as classes tenham tamanhos muito diferentes. O MCC varia de -1 até 1, sendo que 1 significa acurácia máxima, 0 indica que as classificações assumem um comportamento aleatório, e -1 indica que há uma inversão das classes. O MCC é calculado pela equação $MCC = \frac{VP \times VN - FP \times FN}{\sqrt{(VP+FP)(VP+FN)(VN+FP)(VN+FN)}}$.

3.3. Modelo alvo: detecção de anomalias em séries temporais baseada em LSTM

Para construir este modelo alvo, seguiu-se uma abordagem comumente utilizada em detectores de anomalias para séries temporais. Os dados são divididos em três conjuntos: treinamento, definição de limiar e inferência. Para a fase de treinamento, uma porção inicial do conjunto de dados é selecionada de modo que não haja nenhuma anomalia. A LSTM foi modelada de forma a receber 5 observações em sua entrada e prever a sexta observação subsequente. Consequentemente, a série temporal foi dividida em janelas de tempo de tamanho 5. Por exemplo, considerando a série temporal [t_0, t_1, t_2, t_3, t_4, t_5, t_6 ...], a primeira janela de tempo de entrada seria [t_0, t_1, t_2, t_3, t_4], com a saída do modelo sendo a predição [t_5]. A segunda janela seria [t_1, t_2, t_3, t_4, t_5], com a saída sendo a predição para [t_6] e assim por diante. A Figura 2 ilustra esse processo, considerando os 8 atributos contidos no conjunto de dados.

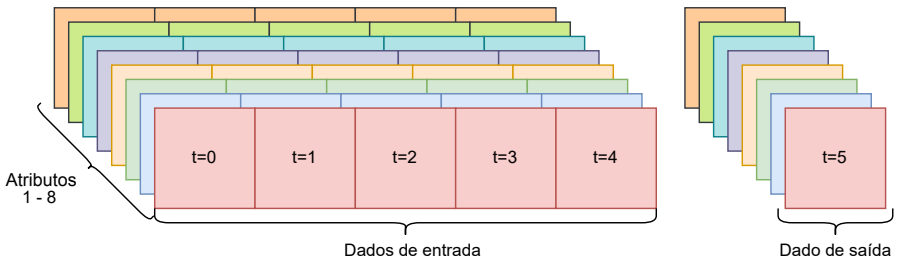

Figura 2. Configuração dos dados de entrada e saída para o modelo LSTM.

Após o treinamento do modelo, o próximo passo é definir o limiar que será empregado para determinar se uma observação é anômala. Os dados utilizados na definição do limiar correspondem a 50 observações subsequentes à fase de treinamento. Para definir o limiar, a série de 50 observações é passada para o modelo treinado, considerando as janelas de 5 observações cada. As previsões realizadas pelo modelo ($\hat{y}_i$) são então comparadas aos elementos reais correspondentes na série (y_i). O Erro Quadrático Médio (EQM) entre as previsões realizadas pelo modelo e os valores reais é calculado de acordo com a fórmula $EQM = \frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2$. Depois de calcular o EQM, é computado o desvio padrão (σ) dos Erros Quadráticos (EQ) das previsões do modelo com os valores reais. O limiar é finalmente calculado pela fórmula $limiar = EQM(\hat{y}_i, y_i) + z \times \sigma(EQ(\hat{y}_i, y_i))$.

A próxima etapa é a fase de inferência. Dados de entrada são passados para o modelo LSTM, que produz uma estimativa. A distância Euclidiana entre a estimativa e

o valor real é comparada com o limiar. Se a distância for maior ou igual ao limiar, é detectada uma anomalia. Caso contrário, a observação é considerada normal. A Figura 3 representa o funcionamento do sistema de detecção de anomalias baseado em LSTM.

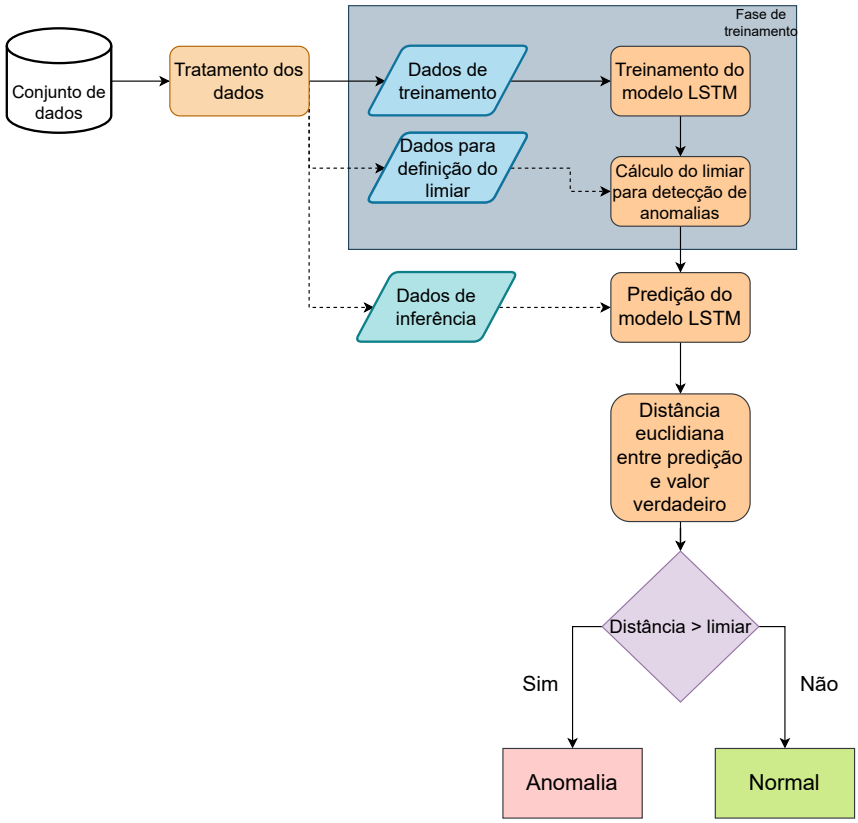

Figura 3. Fluxo de detecção de anomalias do algoritmo baseado em LSTM. As amostras são detectadas pelo sistema como sendo normais ou anômalas.

O modelo de detecção de anomalias é criado contendo duas camadas LSTM empilhadas cada uma com 100 células. Além disso, após as camadas LSTM, há uma camada densa com 16 neurônios e função de ativação ReLu, utilizada para extrair informações úteis das camadas anteriores e reduzir a dimensionalidade dos dados. Por último, há uma camada *Dropout* para reduzir o sobreajuste do modelo e aumentar a generalização, seguido de uma camada densa, que resultará na saída do modelo. O modelo alvo é treinado utilizando o *framework TensorFlow* [Abadi et al. 2015].

3.4. Modelo alvo e modelo adversário baseados em um classificador MLP

Para poder criar exemplos adversários a partir do FGSM, o atacante treina um classificador, denominado modelo adversário, utilizando dados pertencentes ao conjunto de dados do alvo. Este tipo de ataque pressupõe, portanto, que o atacante seja capaz de monitorar ou roubar uma parte desse conjunto de dados. Neste trabalho, o modelo adversário será um classificador baseado em uma MLP. Também haverá um cenário onde o modelo alvo será um classificador baseado em uma MLP, de forma a comparar a sua robustez contra exemplos adversários com o modelo baseado em LSTM.

Na classificação binária, as características de continuidade da série temporal não precisam ser consideradas, já que cada observação é analisada individualmente em relação

ao seu rótulo. Portanto, os dados podem ser preparados de maneira diferente da que foi usada com o modelo baseado em LSTM. 60% dos dados serão utilizados para o treinamento, e, o restante, na fase de inferência. Nesse caso, a presença de amostras normais e anômalas tanto nos dados de treinamento quanto nos dados de inferência é necessária, respeitando a proporção entre elas encontrada no conjunto de dados original. Ainda, considerando que as observações que vão compor os conjuntos de treinamento e inferência são selecionados aleatoriamente, todos os testes envolvendo a MLP foram executados cinco vezes. Para avaliar os experimentos envolvendo essas execuções, é feita uma média das métricas (MCC, F1-score, precisão, revocação, TFP e TFN) calculadas em cada uma delas. O funcionamento desse modelo é representado na Figura 4.

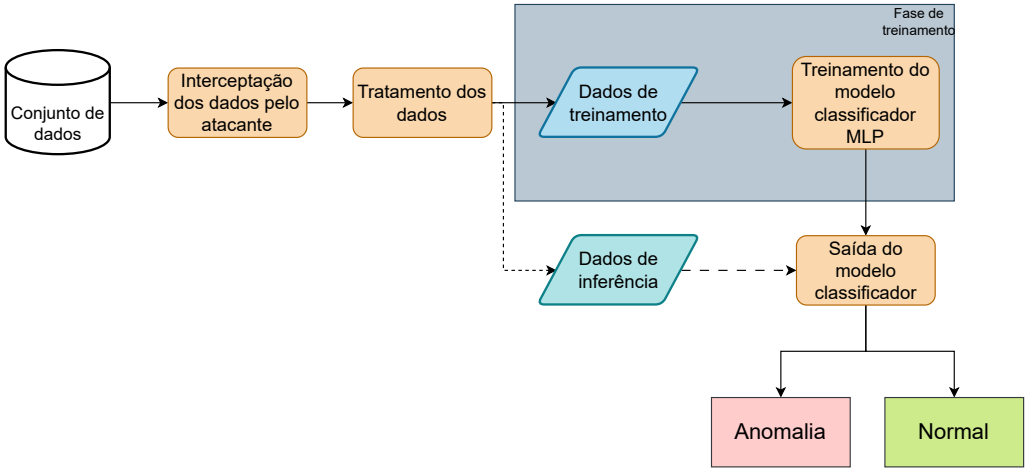

Figura 4. Classificação de observações utilizando uma rede neural MLP. As amostras são classificadas como normais ou anômalas.

A arquitetura da MLP, tanto para o modelo adversário quanto para o modelo alvo, conta com uma camada de entrada com 8 nós, um para cada atributo. Após a camada de entrada, existem duas camadas densas, sendo que a primeira contém 64 neurônios e, a segunda, 32 neurônios. Finalmente, há uma camada de saída que representa a classificação da observação (0 ou 1).

3.5. Exemplos adversários

Exemplos adversários são dados de entrada manipulados intencionalmente com o intuito de levar o modelo alvo a fazer predições equivocadas. Normalmente, esses dados são criados adicionando a menor perturbação possível aos dados originais, de forma que maximize a influência na capacidade de predição do modelo alvo.

Os exemplos adversários foram criados com base em duas técnicas. A primeira técnica, denominada *naive* devido à sua simplicidade, é utilizada como uma linha de base, sendo baseada no desvio padrão dos dados da fase de treinamento do modelo de detecção de anomalias. Para realizar esse ataque é necessário calcular o desvio padrão de cada atributo (*feature*) e adicioná-lo aos atributos da observação que será manipulada. Para criar falsos positivos utilizando o ataque *naive*, o atacante busca manipular a entrada x de forma que se aumente a distância Euclidiana entre a predição do modelo ($f(x)$) e o valor real (y), denotada por $d(f(x), y)$. A entrada manipulada x^* é obtida a partir da equação $x^* = x + k \times \sigma$, onde σ é o desvio padrão dos atributos dos dados de treinamento e k

uma constante que representa a magnitude do ataque. O ataque é bem sucedido quando $d(f(x), y) < limiar$ e $d(f(x^*), y) >= limiar$, ou seja, uma observação que não era anômala passa a ser classificada como tal após a manipulação. Já para produzir falsos negativos, basta inverter a perturbação σ calculada anteriormente: $x^* = x - k \times \sigma$. Neste caso, o ataque é bem sucedido quando temos $d(f(x), y) > limiar$ e $d(f(x^*), y) <= limiar$.

Já a segunda técnica segue o método FGSM [Goodfellow et al. 2014], que se baseia no gradiente da função de perda do modelo adversário treinado pelo atacante. Esse gradiente é utilizado para manipular a entrada legítima na direção que maximize a perda do classificador. Portanto, a perturbação δ será calculada conforme a equação $\delta = \epsilon \cdot sign(\nabla_x J(\theta, x, y))$, onde ϵ é uma constante que representa a magnitude do ataque, $\nabla_x J(\theta, x, y)$ é o gradiente da função de perda, $J(\theta, x, y)$ a função de perda, θ são os parâmetros do modelo, **x** o dado de entrada e **y** a saída esperada para a entrada **x**. Quando é usado contra classificadores, o FGSM prevê que a perturbação δ é somada a todos os atributos da entrada x para produzir x^*. No cenário em que temos a rede neural MLP como modelo alvo, esse procedimento será seguido. Para criar falsos positivos em um modelo de detecção de anomalias utilizando o método FGSM, faz-se necessária uma adaptação nesse procedimento. Para gerar falsos positivos em um modelo de detecção de anomalias, o atacante continuará adicionando a perturbação δ a x para produzir o exemplo adversário x^*. No entanto, quando estiver buscando gerar falsos negativos, o atacante irá calcular x^* subtraindo δ de x.

4. Resultados

Para realizar os experimentos, primeiramente, definiu-se o melhor ajuste para o limiar do modelo de detecção de anomalias através da variação de z entre 15 e 30. A Figura 5 mostra os resultados obtidos, onde é possível notar que os melhores valores de z estão entre 21 e 22. Portanto, o valor $z = 21$ foi escolhido. Nas subseções a seguir, serão apresentados os resultados obtidos para a execução dos ataques *naive* e FGSM contra os modelos baseados em LSTM e MLP.

4.1. Ataque *naive* contra o modelo de detecção de anomalias (LSTM)

Os resultados na Tabela 1 revelam um aumento na taxa de falsos positivos à medida que a magnitude do ataque aumenta. Nesse cenário, houve um aumento máximo de aproximadamente 85% na taxa de falsos positivos, com a magnitude $k = 0,5$. Esse aumento teve um impacto modesto no MCC (redução de 5,36%) e no F1-score (redução de 1,58%).

Tabela 1. Métricas considerando o ataque *naive* para geração de falsos positivos no modelo de detecção de anomalias.

k	MCC	F1-score	Precisão	Revocação	TFN	TFP
Sem ataque	0,8629	0,9490	0,9619	0,9365	0,0634	0,0654
0,01	0,8629	0,9490	0,9619	0,9365	0,0634	0,0654
0,05	0,8629	0,9490	0,9619	0,9365	0,0634	0,0654
0,1	0,8629	0,9490	0,9619	0,9365	0,0634	0,0654
0,2	0,8551	0,9465	0,9567	0,9365	0,0634	0,0747
0,3	0,8551	0,9465	0,9567	0,9365	0,0634	0,0747
0,4	0,8397	0,9414	0,9465	0,9365	0,0634	0,0934
0,5	0,8166	0,9340	0,9315	0,9365	0,0634	0,1214

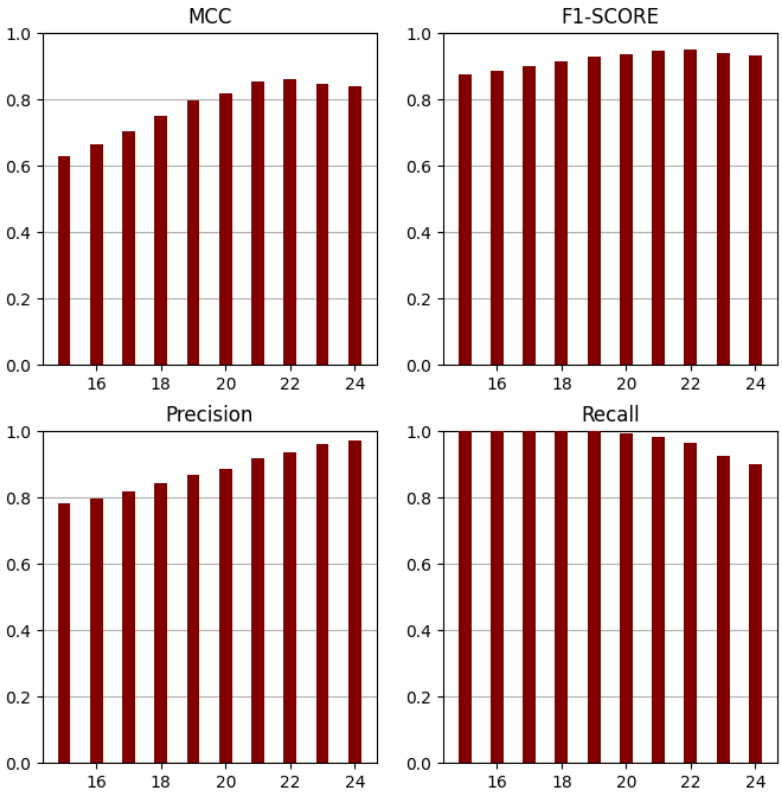

Figura 5. Métricas obtidas para cada variação de z na definição do limiar.

Os resultados dos ataques *naive* para gerar falsos negativos são apresentados na Tabela 2, mostrando um aumento de 66% na taxa de falsos negativos. Como no caso anterior, o impacto nas métricas de detecção de anomalias não é significativo, com uma redução de aproximadamente 6% para o MCC e 2,42% para o F1-Score.

4.2. Ataque FGSM contra o modelo de detecção de anomalias (LSTM)

Nos resultados da Tabela 3, observa-se um aumento de aproximamente 9x na taxa de falsos positivos, que é significativamente maior que no ataque *naive*. As métricas MCC e F1-score, como consequência, tiveram uma redução de 47,64% e 12,44%, respectivamente. Tais resultados demonstram um impacto muito maior em comparação com o ataque *naive*.

Já no segundo cenário, o objetivo é aplicar a perturbação do ataque FGSM em amostras rotuladas como anômalas, ou seja, é esperado o aumento da taxa de falsos negativos. A Tabela 4 representa os resultados obtidos nesse cenário, com um aumento máximo de, aproximadamente, 116% na taxa de falsos negativos, uma redução de 10,25% na métrica MCC e 4,32% de redução para o F1-Score. Com base nos resultados, é possível notar que gerar impacto ao modelo com falsos negativos é mais desafiador do que com falsos positivos. Isso ocorre porque o método de detecção de anomalias leva em consideração um resíduo entre a predição e o valor real. Nesse cenário, gerar falsos negativos significa diminuir esse resíduo para que seja menor que o limiar. No entanto, pode não ser possível alcançar esse objetivo com um ataque como o FGSM, que aplica uma perturbação com a mesma magnitude para todos os atributos de cada observação.

Tabela 2. Métricas considerando o ataque *naive* para geração de falsos negativos no modelo de detecção de anomalias.

k	MCC	F1-score	Precisão	Revocação	TFN	TFP
Sem ataque	0,8629	0,9490	0,9619	0,9365	0,0634	0,0654
0,01	0,8629	0,9490	0,9619	0,9365	0,0634	0,0654
0,05	0,8629	0,9490	0,9619	0,9365	0,0634	0,0654
0,1	0,8495	0,9433	0,9615	0,9259	0,0740	0,0654
0,2	0,8429	0,9405	0,9613	0,9206	0,0793	0,0654
0,3	0,8364	0,9376	0,9611	0,9153	0,0846	0,0654
0,4	0,8235	0,9318	0,9606	0,9047	0,0952	0,0654
0,5	0,8109	0,9260	0,9602	0,8941	0,1058	0,0654

Tabela 3. Métricas considerando o ataque FGSM para geração de falsos positivos no modelo de detecção de anomalias.

ϵ	MCC	F1-score	Precisão	Revocação	TFN	TFP
Sem ataque	0,8629	0,9490	0,9619	0,9365	0,0634	0,0654
0,01	0,8629	0,9490	0,9619	0,9365	0,0634	0,0654
0,05	0,8551	0,9465	0,9567	0,9365	0,0634	0,0747
0,1	0,8166	0,9340	0,9315	0,9365	0,0634	0,1214
0,2	0,7633	0,9170	0,8984	0,9365	0,0634	0,1869
0,3	0,6722	0,8894	0,8468	0,9365	0,0634	0,2990
0,4	0,5646	0,8592	0,7937	0,9365	0,0634	0,4299
0,5	0,4518	0,8309	0,7468	0,9365	0,0634	0,5607

Sabendo que o cálculo dos resíduos leva em consideração esses atributos, seria necessário aplicar perturbações com magnitudes diferentes para cada um deles.

Tabela 4. Métricas considerando o ataque FGSM para geração de falsos negativos no modelo de detecção de anomalias.

ϵ	MCC	F1-score	Precisão	Revocação	TFN	TFP
Sem ataque	0,8629	0,9490	0,9619	0,9365	0,0634	0,0654
0,01	0,8629	0,9490	0,9619	0,9365	0,0634	0,0654
0,05	0,8429	0,9405	0,9613	0,9206	0,0793	0,0654
0,1	0,8047	0,9230	0,96	0,8888	0,1111	0,0654
0,2	0,7863	0,9141	0,9593	0,8730	0,1269	0,0654
0,3	0,7744	0,9080	0,9588	0,8624	0,1375	0,0654
0,4	0,7924	0,9171	0,9595	0,8783	0,1216	0,0654
0,5	0,8561	0,9462	0,9617	0,9312	0,0687	0,0654

Em geral, os resultados apresentados demonstram um aumento na taxa de falsos negativos até um determinado ponto, e, a partir desse ponto, essa taxa começa a diminuir. Isso evidencia que aplicar a mesma magnitude de perturbação para todas as observações

e seus atributos é insuficiente nesse cenário, já que um determinado atributo pode ter uma influência maior no cálculo do resíduo do que os outros.

4.3. Ataque *naive* contra o modelo de classificação (MLP)

Conforme a Tabela 5, os resultados obtidos mostram que o modelo de classificação baseado em MLP, sem ataques, apresenta um desempenho melhor em comparação ao modelo de detecção de anomalias. No entanto, quando submetido ao ataque *naive*, o modelo de classificação teve um aumento máximo de 605,08% na taxa de falsos positivos, impacto maior do que no modelo de detecção de anomalias, que apresentou um aumento de apenas 85% na taxa de falsos positivos. Isso sugere que, embora o modelo de classificação tenha um desempenho melhor no geral, ele é mais vulnerável a exemplos adversários.

Tabela 5. Métricas considerando o ataque *naive* para geração de falsos positivos no modelo de classificação.

k	MCC	F1-score	Precisão	Revocação	TFN	TFP
Sem ataque	0,9533	0,9649	0,9821	0,9482	0,0517	0,0059
0,01	0,9533	0,9649	0,9821	0,9482	0,0517	0,0059
0,05	0,9533	0,9649	0,9821	0,9482	0,0517	0,0059
0,1	0,9533	0,9649	0,9821	0,9482	0,0517	0,0059
0,2	0,9304	0,9482	0,9482	0,9482	0,0517	0,0178
0,3	0,9304	0,9482	0,9482	0,9482	0,0517	0,0178
0,4	0,8875	0,9166	0,8870	0,9482	0,0517	0,0416
0,5	0,8875	0,9166	0,8870	0,9482	0,0517	0,0416

No segundo caso, a Tabela 6 apresenta os resultados do ataque *naive* visando gerar falsos negativos para o modelo de classificação. O modelo de classificação apresentou uma aumento muito maior na taxa de falsos negativos, de 1100%, quando comparado com o outro modelo, que obteve um aumento de 66%.

Tabela 6. Métricas considerando o ataque *naive* para geração de falsos negativos no modelo de classificação.

k	MCC	F1-score	Precisão	Revocação	TFN	TFP
Sem ataque	0,9533	0,9649	0,9821	0,9482	0,0517	0,0059
0,01	0,9416	0,9557	0,9818	0,9310	0,0689	0,0059
0,05	0,9181	0,9369	0,9811	0,8965	0,1034	0,0059
0,1	0,9063	0,9272	0,9807	0,8793	0,1206	0,0059
0,2	0,8708	0,8971	0,9795	0,8275	0,1724	0,0059
0,3	0,7870	0,8199	0,9761	0,7068	0,2931	0,0059
0,4	0,6878	0,7173	0,9705	0,5689	0,4310	0,0059
0,5	0,5393	0,5432	0,9565	0,3793	0,6206	0,0059

Em resumo, os resultados demonstram que o modelo de classificação baseado em MLP foi superior ao modelo LSTM em condições normais, sem a presença de ataques. No entanto, quando submetido a ataques usando a técnica *naive*, sofreu um impacto mais

significativo do que o modelo de detecção de anomalias. Destaca-se que essa técnica é uma abordagem simplificada que não leva em consideração os parâmetros do classificador, o que sugere que um ataque mais sofisticado pode causar um impacto ainda mais expressivo.

4.4. Ataque FGSM contra o modelo de classificação (MLP)

Com base nos resultados apresentados na Tabela 7, pode-se notar que o modelo de classificação foi significativamente mais afetado pelo ataque FGSM que o modelo de detecção de anomalias. O ataque FGSM é mais sofisticado, portanto, espera-se que seja mais efetivo do que o ataque *naive*.

Tabela 7. Métricas considerando o ataque FGSM para geração de falsos positivos no modelo de classificação.

ϵ	MCC	F1-score	Precisão	Revocação	TFN	TFP
Sem ataque	0,9533	0,9649	0,9821	0,9482	0,0517	0,0059
0,01	0,9533	0,9649	0,9821	0,9482	0,0517	0,0059
0,05	0,8978	0,9243	0,9016	0,9482	0,0517	0,0357
0,1	0,8773	0,9090	0,8730	0,9482	0,0517	0,0476
0,2	0,8386	0,8799	0,8208	0,9482	0,0517	0,0714
0,3	0,5425	0,6547	0,5	0,9482	0,0517	0,3273
0,4	0,3018	0,5	0,3395	0,9482	0,0517	0,6369
0,5	0,0035	0,4044	0,2570	0,9482	0,0517	0,9464

Ao comparar os dois modelos, pode-se observar que o modelo de classificação (MLP) foi mais afetado pelo ataque FGSM, apresentando um aumento exorbitante de aproximadamente 159x na taxa de falsos positivos. Em contrapartida, o modelo de detecção de anomalias (LSTM) teve um aumento de 9x na taxa de falsos positivos sob o mesmo ataque. Esses resultados sugerem que o modelo MLP é mais vulnerável aos ataques do que o modelo LSTM.

Já para os testes em que buscamos aumentar a taxa de falsos negativos, os resultados podem ser observados na Tabela 8. Neste cenário, o modelo de classificação apresentou um aumento máximo de aproximadamente 18x na taxa de falsos negativos, enquanto o modelo de detecção de anomalias teve um aumento de apenas 2x. Isso se deve ao fato de que o processo de geração de exemplos adversários para o modelo de detecção de anomalias é baseado na distância Euclidiana entre a predição e o valor real, tornando mais difícil gerar um exemplo adversário que diminua essa distância.

4.5. Discussão

Os resultados indicaram que o modelo de detecção de anomalias baseado em LSTM mostrou-se significativamente mais resistente aos ataques do que o classificador baseado em MLP. Independentemente do tipo de ataque, seja ele *naive* ou FGSM, e do objetivo, seja aumentar falsos positivos ou negativos, o impacto nas métricas de desempenho do classificador MLP foi sempre muito mais pronunciado. Por exemplo, o ataque *naive* elevou os falsos positivos da LSTM em 85%, enquanto a MLP apresentou um aumento

Tabela 8. Métricas considerando o ataque FGSM para geração de falsos negativos no modelo de classificação.

ϵ	MCC	F1-score	Precisão	Revocação	TFN	TFP
Sem ataque	0,9533	0,9649	0,9821	0,9482	0,0517	0,0059
0,01	0,9416	0,9557	0,9818	0,9310	0,0689	0,0059
0,05	0,6750	0,7032	0,9696	0,5517	0,4482	0,0059
0,1	0,5247	0,525	0,9545	0,3620	0,6379	0,0059
0,2	0,2180	0,1562	0,8333	0,0862	0,9137	0,0059
0,3	0,1516	0,0967	0,75	0,0517	0,9482	0,0059
0,4	0,1516	0,0967	0,75	0,0517	0,9482	0,0059
0,5	0,1516	0,0967	0,75	0,0517	0,9482	0,0059

próximo a 600%. Outro ponto relevante dos resultados é a maior dificuldade em gerar falsos negativos em comparação aos falsos positivos. Produzir falsos negativos geralmente requer modificar a entrada para que ela se ajuste ao limiar que separa amostras normais e anormais, reduzindo a distância da entrada para as amostras normais. Já a geração de falsos positivos costuma ser mais simples, pois envolve alterar a entrada de modo que ela se distancie de uma amostra considerada normal, algo frequentemente alcançado com a adição de uma perturbação significativa. Em outras palavras, qualquer alteração na amostra que ultrapasse o limiar entre amostras normais e anormais é suficiente para gerar um falso positivo, o que é mais fácil do que encontrar uma perturbação precisa o bastante para fazer a amostra parecer normal. Finalmente, o ataque FGSM demonstrou sua maior sofisticação ao causar impactos muito mais significativos do que o ataque *naive*, que possui uma concepção muito mais simples.

5. Conclusão

Este trabalho buscou analisar o impacto de exemplos adversários contra detectores de anomalias, utilizando dados de um sistema ciberfísico de circulação de água com leituras de 8 sensores a cada segundo. Foram implementados dois modelos alvo: um de detecção de anomalias, que compara resíduos entre dados reais e previstos por uma rede LSTM para identificar anomalias, e um modelo de classificação baseado em uma MLP, criado para comparar os impactos dos ataques em ambos os modelos. Entre os ataques, houve um mais simples, denominado *naive*, que adicionava uma perturbação baseada no desvio padrão dos dados de treinamento para criar os exemplos adversários. O outro ataque, mais sofisticado, foi baseado no método FGSM e explora características de classificadores supervisionados. Os ataques FGSM destacaram-se por aumentar drasticamente as taxas de falsos positivos e negativos, com o modelo de classificação sendo mais suscetível que o de detecção de anomalias. O modelo de detecção, embora com resultados inferiores quando não havia ataques, mostrou-se mais robusto contra exemplos adversários.

Para o futuro, espera-se desenvolver técnicas de defesa contra exemplos adversários. A exploração de ataques específicos para modelos de regressão em séries temporais também é importante para entender vulnerabilidades e desenvolver contramedidas, fortalecendo a segurança e confiabilidade dos modelos de aprendizado de máquina nesse contexto.

Referências

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

Barrera-Animas, A. Y., Oyedele, L. O., Bilal, M., Akinosho, T. D., Delgado, J. M. D., and Akanbi, L. A. (2022). Rainfall prediction: A comparative analysis of modern machine learning algorithms for time-series forecasting. *Machine Learning with Applications*, 7:100204.

Carlini, N. and Wagner, D. (2017). Adversarial examples are not easily detected: Bypassing ten detection methods. In *Proceedings of the 10th ACM workshop on artificial intelligence and security*.

Chandola, V., Banerjee, A., and Kumar, V. (2009). Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3).

Choi, K., Yi, J., Park, C., and Yoon, S. (2021). Deep learning for anomaly detection in time-series data: Review, analysis, and guidelines. *IEEE Access*, 9:120043–120065.

Gallagher, M., Pitropakis, N., Chrysoulas, C., Papadopoulos, P., Mylonas, A., and Katsikas, S. (2022). Investigating machine learning attacks on financial time series models. *Computers & Security*, 123:102933.

Goodfellow, I. J., Shlens, J., and Szegedy, C. (2014). Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.

Harford, S., Karim, F., and Darabi, H. (2021). Generating adversarial samples on multivariate time series using variational autoencoders. *IEEE/CAA Journal of Automatica Sinica*, 8(9):1523–1538.

Huang, L., Joseph, A. D., Nelson, B., Rubinstein, B. I., and Tygar, J. D. (2011). Adversarial machine learning. In *Proceedings of the 4th ACM workshop on Security and artificial intelligence*.

Jia, Y., Wang, J., Poskitt, C. M., Chattopadhyay, S., Sun, J., and Chen, Y. (2021). Adversarial attacks and mitigation for anomaly detectors of cyber-physical systems. *International Journal of Critical Infrastructure Protection*, 34:100452.

Katser, I. D. and Kozitsin, V. O. (2020). Skoltech anomaly benchmark (skab). `https://www.kaggle.com/dsv/1693952`.

Khan, A. F., Kamalakannan, K., and Ahmed, N. S. S. (2023). Integrating machine learning and stochastic pattern analysis for the forecasting of time-series data. *SN Computer Science*, 4(5):484.

Khan, S. U., Mynuddin, M., and Nabil, M. (2024). Adaptedge: Targeted universal adversarial attacks on time series data in smart grids. *IEEE Transactions on Smart Grid*, pages 1–1.

Kim, K.-D. and Kumar, P. R. (2012). Cyber–physical systems: A perspective at the centennial. *Proceedings of the IEEE*, 100(Special Centennial Issue).

Sadeghzadeh, A. M., Shiravi, S., and Jalili, R. (2021). Adversarial network traffic: Towards evaluating the robustness of deep-learning-based network traffic classification. *IEEE Transactions on Network and Service Management*, 18(2):1962–1976.

Zhang, K., Wen, Q., Zhang, C., Cai, R., Jin, M., Liu, Y., Zhang, J. Y., Liang, Y., Pang, G., Song, D., and Pan, S. (2024). Self-supervised learning for time series analysis: Taxonomy, progress, and prospects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–20.

Zhou, X., Kouzel, M., and Alemzadeh, H. (2022). Robustness testing of data and knowledge driven anomaly detection in cyber-physical systems. In *2022 52nd Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W)*, pages 44–51.