IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Recebido em 24 de setembro de 2021, aceito em 22 de dezembro de 2021, data de publicação em 31 de dezembro de 2021, data da versão atual em 20 de janeiro de 2022.

Identificador de objeto digital 10.1109/ACCESS.2021.3140015

Aprendizado profundo generativo para detectar Ataques cibernéticos para o conjunto de dados IoT-23

N. ABDALGAWAD, (Membro Estudante, A. SAJUN, IEEE), Y. KADDOURA, (Membro Estudante, IA ZUALKERNAN Departamento **IEEE), , (Membro, IEEE), E F. ALOUL (Membro Sênior, IEEE)**
de Ciência e Engenharia da Computação, Universidade Americana de Sharjah, Sharjah, Emirados Árabes Unidos

Autor correspondente: N. Abdalgawad (g00068826@alumni.aus.edu)

Este trabalho foi financiado pelo Programa de Acesso Aberto da Universidade Americana de Sharjah.

RESUMO Espera-se que o rápido crescimento da Internet das Coisas (IoT) adicione bilhões de dispositivos IoT conectados à Internet. Esses dispositivos representam uma vasta superfície de ataque para ataques cibernéticos. Por exemplo, esses dispositivos IoT podem ser infectados com botnets para permitir ataques Distributed Denial of Service (DDoS). Os sistemas de detecção de intrusão baseados em assinatura são contramedidas tradicionais para tais ataques. No entanto, esses métodos dependem de especialistas humanos e consomem muito tempo em termos de atualizações e podem não esgotar todos os tipos de ataque, especialmente ataques de dia zero. O aprendizado profundo mostrou alguma promessa na detecção de invasões. Este artigo mostra que é possível usar métodos generativos de aprendizagem profunda como Adversarial Autoencoders (AAE) e Bidirectional Generative Adversarial Networks (BiGAN) para detectar intrusos com base em uma análise dos dados da rede. O conjunto de dados IoT-23 completo publicado recentemente com base na fechadura da porta Somfy, dispositivos Philips Hue e Amazon Echo foi usado para treinar modelos generativos de aprendizado profundo para detectar uma variedade de ataques como DDoS e vários botnets como Mirai, Okiruk e Torii. Mais de 1,8 milhão de fluxos de rede foram usados para treinar os vários modelos. Os modelos generativos resultantes superam as técnicas tradicionais de aprendizado de máquina, como Random Forests. Os modelos baseados em AAE e BiGAN foram capazes de atingir um F1-Score de 0,99. Um BiGAN para detectar ataques desconhecidos também foi treinado para detectar novos ataques de dia zero com um F1-Score de 0,85 a 1.

TERMOS DE INDEXAÇÃO Autoencoders adversários, segurança cibernética, redes adversárias generativas, Internet das Coisas, sistemas de detecção de intrusão.

I. INTRODUÇÃO

Internet das coisas (IoT) é uma das principais tecnologias da atualidade e é considerada uma extensão natural da internet por incorporar comunicações e sensores máquina a máquina. As aplicações de IoT apareceram em uma variedade de domínios, incluindo saúde, fitness, gerenciamento de energia doméstica, automação de sala de aula, cidades inteligentes e muito mais [1]. Um aplicativo IoT típico consiste em três camadas; a camada de percepção, camada de rede e camada de aplicação. A camada de percepção é responsável por detectar e coletar informações sobre o ambiente e enviá-las para a camada de rede.

Por exemplo, câmeras de vigilância são um tipo de sensor que reconhece eventos incomuns como movimento usando sensores.

A camada de rede/transporte é considerada um elo entre a camada de percepção e a nuvem. Esta camada consiste em muitos protocolos de internet e tem que integrar a comunicação tecnologias para troca de informações como Zigbee, 5G, MQTT e Wi-Fi [2]. Por exemplo, uma câmera de vigilância pode usar o roteador doméstico e o Wi-Fi para enviar um evento de detecção de movimento ao servidor principal. A aplicação posteriormente utiliza os dados recebidos da camada de rede para fornecer quaisquer operações ou serviços requeridos pelos usuários [3]. Por exemplo, o serviço de nuvem pode enviar um alarme para um aplicativo móvel usado pelo proprietário de uma casa, indicando que um movimento foi detectado em uma de suas câmeras de vigilância.

A IoT é vulnerável a riscos de segurança em todas as camadas arquitetônicas e tem enfrentado desafios de segurança desde seu surgimento [4]. Por exemplo, Butt *et al.* [5] examinou o tipo de ataques em Smart Health Systems e descobriu que os ataques são Denial of Service Attack (DoS), Fingerprint and Timing based Snooping (FATS), Router Attack, Select Forwarding (SF) Attack, Sensor Attack e Replay Attack . Em geral, a camada de percepção pode sofrer ataques como injeção de código malicioso, espionagem e interferência [3], [6].

Da mesma forma, a camada de rede é suscetível a ataques como

O editor associado coordenando a revisão deste manuscrito e quem o aprovou para publicação foi Yu-Huei Cheng.

spoofing, negação de serviço, man-in-the-middle e informações de roteamento [6]. A privacidade é outra grande preocupação [4]. Os dispositivos IoT requerem sistemas de autenticação fortes, que muitos dispositivos IoT não possuem devido a restrições de recursos como CPU ou limitações de energia/bateria [7]. Por fim, a camada de aplicativos também está aberta a ataques de vírus, worms e ataques de phishing. Andreia *et ai.* [8] classificou tais ataques em quatro categorias; físico, software, rede e criptografia. O ataque físico ocorre quando o agressor está próximo do sistema fisicamente, enquanto o ataque de software é quando o dispositivo contrai um bug que permite o acesso não autorizado ao dispositivo que pode prejudicar o sistema. O ataque de rede ocorre quando a rede IoT é acessada para manipular um dispositivo para causar danos e o ataque de criptografia ocorre quando a criptografia IoT é comprometida.

Botnet é um mecanismo de ataque específico que explora dispositivos IoT. Angrishi *et ai.* [9] descreve um botnet como um grande grupo de dispositivos habilitados para internet que são controlados para fazer solicitações simultâneas a um servidor especificado (ou grupo de servidores) para sobrecarregá-lo e impedi-lo de responder a solicitações legítimas, interrompendo assim essencialmente sua serviço. Este ataque é um ataque distribuído de negação de serviço (DDoS). Dois métodos usados em tal ataque DDoS: reflexão e amplificação. Ambos levam ao esgotamento da largura de banda e dos recursos do alvo. E devido ao aumento da sofisticação desses ataques, eles são muito difíceis de identificar [9]. As botnets IoT não são apenas uma ameaça para os proprietários de dispositivos IoT, mas também para qualquer pessoa na Internet. Como os ataques DDoS precisam de um tráfego de rede significativo para comprometer os serviços, os dispositivos IoT fornecem um host perfeito devido ao grande número de dispositivos IoT disponíveis e em uso hoje, bem como à sua segurança geralmente fraca, tornando-os os frutos mais fáceis [10].

O primeiro botnet conhecido foi o Linux/Hydra, lançado em 2008. Ele tinha recursos de disseminação e capacidade de lançar DDoS. O Psyb0t em 2009 mirou em roteadores e modems, comprometendo quase 100 mil dispositivos. Os métodos de infecção usaram ataques de força bruta usando 6.000 nomes de usuários predefinidos e 13 mil senhas predefinidas [9].
Linux.Darlloz é outro exemplo de um botnet IoT que infectou mais de 31 mil dispositivos e era um worm IoT. Depois de infectar o dispositivo IoT, ele impediria que qualquer usuário acessasse o dispositivo, descartando o tráfego telnet e encerrando o processo telnetd [9]. Spike (Dofloo) é outro botnet que visava PCs baseados em Windows e Linux e estava usando para lançar vários ataques a organizações na Ásia e nos EUA. Teve um pico de 215Gbps, permitindo lançar diversos tipos de ataques por DDoS. O BASHLITE é outro botnet que controlava mais de um milhão de dispositivos IoT e podia lançar ataques a 400 Gbps [9].

Das *et al.* [11] descreve o Mirai, um botnet recente no qual um vírus procura por dispositivos vulneráveis e se liga a eles tornando-os conectados a Servidores de Comando e Controle (servidores C&C). Ao estarem conectados aos servidores C&C, eles ficam vulneráveis a ataques ou podem ser usados para atacar outros dispositivos.

Das *et al.* [11] e Tushir *et al.* [12] afirmaram que os dispositivos IoT conectados a Mirai Botnets são usados principalmente para realizar ataques DDoS contra um dispositivo alvo. Tushir *et ai.* [12] examinaram os efeitos do ataque Mirai em dispositivos IoT e descobriram que o consumo de energia por dispositivos IoT aumenta em cerca de 40% e o armazenamento usado é aumentado pela metade. A extensão do perigo que uma botnet pode representar foi realmente demonstrada em 2016, quando a botnet Mirai foi lançada. Este botnet infectou 4.000 dispositivos por hora e teve cerca de meio milhão de dispositivos infectados ativos em um ataque inovador de 1,1 Tbps.
Os dispositivos IoT infectados estavam espalhados por 164 países. Desde então, houve muitas versões e variações do botnet Mirai, como Persirai, Hajime e BrickerBot [13].
As vítimas de ataques DDoS incluíam sites, provedores de nuvem, indivíduos, faculdades, empresas de telecomunicações, provedores de DNS (Dyn) que ofereciam serviços para vários sites, como Reddit, Amazon, Spotify, Airbnb e outros [9].

Existe claramente a necessidade de reforçar a segurança da IoT para impedir o desenvolvimento de tais botnets, mas também garantir que todas as possíveis vítimas de DDoS estejam bem preparadas para detectar tais ataques, especialmente porque os avanços nessas botnets os tornam muito difíceis de identificar até que seja é tarde demais. De acordo com Statista [14], em 2021, existem quase 8,74 bilhões de dispositivos conectados à IoT em todo o mundo e um white paper da Cisco [15] estima que haverá cerca de 30 bilhões de dispositivos conectados em 2023, em comparação com cerca de 18 em 2018. De acordo com o mesmo artigo, espera ter cerca de 15 milhões de ataques DDoS até 2023, em comparação com 7 milhões em 2018 [15].

Um dos principais métodos de prevenção de tais ataques é a implantação de um forte Sistema de Detecção de Intrusão (IDS) [16] que pode detectar qualquer tipo de intrusão. Atualmente, os IDSs usam dois métodos principais para detectar ataques: baseado em assinatura e baseado em anomalia. Os métodos baseados em assinatura dependem dos ataques conhecidos e suas atualizações são demorada Os ataques baseados em anomalias, por outro lado, são baseados em dados e dependem da máquina entender o comportamento normal e rejeitar todas as conexões de entrada que parecem anormais. Desenvolver um IDS para detecção automática de ataques cibernéticos requer um conjunto de dados apropriado para treinamento. Iot 23 Garcia *et al.* [17] é um desses conjuntos de dados que foi lançado recentemente e aborda especificamente ataques cibernéticos envolvendo dispositivos IoT. Este conjunto de dados foi publicado no início de 202

Este artigo usou o conjunto de dados IoT-23 para explorar o uso de técnicas de aprendizagem profunda generativas que podem detectar e classificar automaticamente os ciberataques de IoT. A principal contribuição deste artigo é que ele usou o conjunto de dados IoT-23 completo para construir um IDS e alcançou resultados de última geração na detecção de anomalias.

II. TRABALHO RELACIONADO

Muito trabalho foi feito na construção de sistemas de detecção de intrusão usando uma variedade de modelos de aprendizado de máquina e aprendizado profundo. Por exemplo, Pang *et al.* [18] revisou os diferentes modelos que foram usados para detecção de anomalias, incluindo Generative Adversarial Networks (GANS). Resende *et ai.* [19] pesquisou diferentes modelos de floresta aleatória

usado em sistemas de detecção de intrusão com uma variedade de conjuntos de dados usando diferentes recursos e classes.

Li *et ai.* [20] propuseram o uso de redes neurais convolucionais (CNN) para classificar os ataques de botnet. Eles dividiram o conjunto de dados em 4 partes separadas (de acordo com as correlações entre os recursos) e treinaram e testaram o mesmo modelo nos dados separadamente para classificação binária (CNN1, CNN2, CNN3, CNN4). Outro modelo chamado CNN0 foi treinado usando os dados completos. Todos os modelos foram treinados no conjunto de dados NSL-KDD [21] e testados no KDDTest+ e KDDTest-21. A maior precisão foi obtida pelo modelo CNN1 em ambos os conjuntos de teste, produzindo 82,62% e 67,22% de precisão, respectivamente. Um conjunto de todos os modelos resultou em precisões de 86,95% e 76,67% em ambos os conjuntos de dados de teste, respectivamente.

Latif *et ai.* [22] apresentou uma rede neural aleatória profunda (DRaNN) e a treinou usando o conjunto de dados UNSW-NB15 [23].
Seu modelo alcançou uma precisão de 99,54%.

Xu *et al.* [24] propuseram um modelo de autoencoder (AE) baseado em Long Short-Term Memory (LSTM) para detecção de intrusões. Eles treinaram o modelo em 5 conjuntos de dados no total: ARP, Fuzzing, Mirai, SSDP Flood e Video Injection da equipe Mirsky [25]. Eles treinaram seu modelo em cada conjunto de dados separadamente com uma classificação binária. Eles então compararam os resultados de seu modelo com dois métodos tradicionais de aprendizado de máquina. Os resultados de seu modelo variaram entre pontuações de F1 de 92,4-96,8 com o Mirai tendo o melhor desempenho em 99,6. Seu modelo geralmente superou o Support Vector Machine (SVM) e K-Nearest Neighbors (KNN), um Autoencoder (AE) e um autocodificador empilhado.

Shahriar *et ai.* [26] propuseram um framework G-IDS que incluía 4 segmentos: módulo de banco de dados, módulo IDS, módulo controlador e módulo sintetizador. O módulo de banco de dados coleta dados reais de detecção de intrusão, bem como dados sintetizados do módulo GAN/sintetizador, cada um com um sinalizador para distinguir as fontes de dados. Os dados sintetizados podem ser pendentes, que não podem ser usados até novo aviso, ou sintéticos, que são dados gerados verificados que permanecem no banco de dados. O módulo controlador inspeciona os dados pendentes e verifica se contribui para o desempenho do IDS e, caso contribua, o sinalizador é alterado para sintético. Seu núcleo era um IDS baseado em ML, que era um modelo ANN multicamadas com 4 camadas ocultas de 50 neurônios cada. Foi treinado duas vezes, uma com os dados pendentes e outra sem para calcular as métricas de desempenho. A GAN foi então usada principalmente para o gerador produzir dados reais para treinar o modelo ANN que foi usado para realizar a classificação multiclasse. A maioria das gravadoras teve pontuações F1 relativamente altas, mas algumas tiveram pontuações mais baixas, como 0,41 e 0,68.

Como a maioria das tarefas de detecção de anomalias, os ataques cibernéticos geralmente resultam em dados desbalanceados. Fan *et ai.* [27] construiu anomalias artificiais baseadas em classes conhecidas para testar seu modelo. Desde o limite entre os dados normais e a anomalia desconhecida é desconhecido e pode estar muito próximo, eles apenas alteraram o valor de um recurso aleatoriamente e mantiveram o restante igual. Eles usaram o conjunto de dados NSL-KDD'99 [21] e seus modelo é um aprendiz de árvore de decisão indutiva, RIPPER. Eles continuaram injetando novos dados em seu conjunto de treinamento e testando com novas anomalias e seus resultados mostraram um aumento nas detecções verdadeiras de 59% para 100%.

RM *e outros.* [28] propuseram um modelo de aprendizado profundo para detectar ataques na Internet das Coisas Médicas (IoMT). Os modelos envolveram uma hibridização da técnica de Análise de Componentes Principais (PCA) e do algoritmo metaheurístico de otimização Gray Wolf. O modelo teve uma precisão 15% superior aos modelos existentes e uma redução no tempo de treinamento em 32%.

As GANs também foram usadas na geração e aumento de dados relacionados a tarefas de detecção de intrusão. Por exemplo, Shahid *et al.* [29] usou uma GAN para construir uma sequência de pacotes.
O conjunto de dados foi construído usando o Google Home Mini criando 42 pacotes em uma sequência com tamanho de vocabulário 535. O modelo consistia em um autoencoder e uma GAN. O codificador construiu o espaço latente da amostra e o enviou para o GAN. A GAN tenta aprender esse espaço latente e gerar amostras com base nele. Em seguida, o decodificador pega o espaço latente gerado e o converte em pacotes reais baseados em texto. Da mesma forma, [30]–[32] usaram GANs para equilibrar os conjuntos de dados e depois treiná-los usando outros modelos. Isso foi feito principalmente gerando mais amostras da classe minoritária para equilibrá-la com a maioria. Por exemplo, Salem *et al.* [30] usaram ciclo-GAN e um Perceptron Multi-Layer para classificação. O F1-score dos métodos tradicionais de balanceamento, como Synthetic Minority Over-Sampling Technique (SMOTE), teve um desempenho ruim quando comparado com seu modelo e o ciclo-GAN teve o melhor desempenho dos três, com um F1-Score de 41,64%.

As GANs também foram usadas para implementar diretamente sistemas de detecção de intrusão. Por exemplo, Huang *et al.* [31] desenvolveram um sistema chamado Sistema de Detecção de Intrusão Adversária Gerativa Desequilibrada (IGAN-IDS). A arquitetura proposta tinha três módulos: extração de características, IGAN e os módulos DNN. A extração de recursos foi um filtro e consistiu em uma incorporação com dimensão de 356 e MLP que consistiu em 2 camadas totalmente conectadas, ambas de dimensões 128 e saída da função sigmoide. Os módulos IGAN eram responsáveis por gerar amostras e tinham, tanto para o discriminador quanto para o gerador, uma taxa de aprendizado de 0,00005 e um tamanho de lote de 128. O discriminador era um MLP 3 totalmente conectado com dimensões 256, 128 e 64. O gerador consistia em 3 totalmente MLP conectadas todas com dimensão 256, 64 kernels de tamanho 16 que constituíam as camadas convolucionais. O IGAN foi totalmente otimizado até que o discriminador convergisse para 0,5. O módulo DNN foi então treinado nas amostras recém-geradas. A DNN tinha 6 camadas: camada totalmente conectada de tamanho 256 com função sigmoide, seguida por 2 camadas convolucionais cada uma de tamanho 64 com função ReLU, seguida por camada dropout com taxa 0,2, seguida por camada totalmente conectada de tamanho 32 com função Leaky ReLU, seguido por uma camada totalmente conectada com o tamanho das classes usando uma função Softmax. O experimento usou 3 conjuntos de dados de intrusões diferentes: NSL-KDD [21], UNSW-NB15 [23], CICIDS2017 com 5, 10 e 6 classes, respectivamente [33].

O sistema alcançou uma precisão de 84,45%, 82,53% e

99,79%, respectivamente, e pontuações F1 de 84,17, 82,86 e 99,79, respectivamente.

Um trabalho semelhante foi relatado por Yilmaz *et al.* [32] onde eles fizeram classificação binária em um conjunto de dados desbalanceado com 5 camadas ocultas para discriminador e gerador com funções de ativação ReLU, Sigmóide e taxas de aprendizado 0,0025 para discriminador e 0,02 para gerador.

Chauhan *et al.* [34] usaram GANs para demonstrar que os métodos de aprendizado profundo não eram suficientes na detecção de novos perfis de ataque. Eles primeiro treinaram uma GAN com base no conjunto de dados CICIDS2017 [33] e usaram o método SHApley Additive exPlanations (SHAP) [35] para extrair recursos do conjunto de dados com base em sua importância e impacto na saída.
A maior taxa de detecção durante o treinamento do modelo foi de 79%. Depois disso, eles usaram duas técnicas para atualizar o perfil do recurso, incluindo aumentar o número de recursos usados e trocar os recursos atuais por outros. Isso resultou em vários ataques adversários que não foram detectados com a taxa de detecção diminuindo para 5,23% e 3,89%, respectivamente, para cada alteração.

Adversarial autoencoders (AAE) é um autoencoder baseado em um GAN; AAE's pode reduzir a probabilidade de overfitting porque pode influenciar a distribuição aproximada pela camada escondida como mostrado por Makhzani *et al.* [36] e Puuska *et al.* [37]. O codificador tenta gerar amostras com base na distribuição escolhida, enquanto o decodificador tenta recriar os dados originais do espaço latente. O discriminador tenta saber se a amostra que foi gerada pelo codificador, é de fato gerada ou da distribuição escolhida. Por exemplo, Puuska *et al.* [37] usou um autoencoder adversário e foi aplicado a um conjunto de dados de intrusão DARPA e a precisão alcançada foi maior do que a do autoencoder normal, mas com base na detecção de anomalias.

Hara *et ai.* [38] também usaram um AAE para detecção de intrusão. Seu foco foi validar a implementação do aprendizado semi-supervisionado por meio de AAE e DNN, alterando a porcentagem de dados rotulados. Eles usaram o conjunto de dados NSL-KDD [21] e concluíram que quanto maior o rótulo dos dados, maior a precisão. A maior precisão alcançada foi de 83,11%.

A arquitetura BiGAN também foi usada para detecção de intrusão. A diferença entre um GAN regular e um BiGAN é que o BiGAN tem um codificador para mapear os dados de volta ao espaço latente, conforme mostrado em Kaplan *et al.* [39].
O gerador converte algum espaço latente z em dados falsos G(z), e o codificador converte os dados reais x em algum espaço latente representado E(x). Ao contrário dos GANs padrão, o discriminador também aprende entradas concatenadas mapeadas para a mesma dimensão (z, G(z)) e (E(x), x). Donahue *et ai.* [40] primeiro avaliou as capacidades de aprendizado de recursos do BiGAN usando treinamento não supervisionado e, em seguida, transferindo o codificador treinado para uso em tarefas de aprendizado supervisionado. Eles os avaliaram usando imagens no banco de dados ImageNet. Entretanto, sua acurácia máxima de classificação para o ImageNet foi de 56,2%. Por outro lado, Donahue *et al.* [41] tentou estender o BigGAN de última geração para um BiGAN (BigBiGAN)

TABELA 1. Resumo do trabalho relacionado que usou GANs.

Paper	Pros	Cons
Shahriar *et al.* [26]	The work used generative adversarial networks to generate samples for imbalanced datasets.	The work did not use GANs for classification.
Hara *et al.* [38]	The work used AAE to train their models.	Their highest achieved accuracy was 83.11%.
Kaplan *et al.* [39]	The work used BiGAN for anomaly detection using an intrusion detection related dataset.	The KDDCUP99 is proved to have redundancy issues [21].
Puuska *et al.* [37]	The work used AAE and their results showed that AAE had better performance than autoencoders.	The work did not consider classification and their highest accuracy was only 65%.
Chauhan *et al.* [34]	The work used GANs to generate new samples of data that could not be detected.	The work did not use GANs to classify samples.

no mesmo banco de dados. Eles treinaram sua arquitetura usando aprendizado não supervisionado e alcançaram uma precisão de 60,8%, o que melhorou os resultados publicados anteriormente de 55,4%.

Kaplan *e outros.* [39] usou um BiGAN para detecção de anomalias por meio de treinamento nos dados normais do conjunto de dados KDDCUP99 [21] e colocou amostras das classes de ataque no conjunto de teste. O BiGAN com seu algoritmo proposto obteve o melhor desempenho com uma pontuação F1 de 90,8. Alabugin *et al.* [42] usaram uma abordagem semelhante usando BiGAN em dados benignos que eles produziram em seu ambiente de teste.

A Tabela 1 mostra um resumo dos trabalhos mais recentes e relacionados. A tabela mostra prós e contras de cada obra. Os contras não significam necessariamente um problema inerente ao trabalho, mas sim uma lacuna de pesquisa que está sendo abordada neste artigo.

III. METODOLOGIA

A. O CONJUNTO DE DADOS IOT-23

Este artigo usou o conjunto de dados IoT-23 [17]. Esses dados são baseados no tráfego de rede obtido de dispositivos da Internet das Coisas (IoT) com 20 malwares e 3 capturas benignas. Vale ressaltar que as 3 capturas benignas foram realizadas em três dispositivos IoT reais: Somfy door lock, Philips Hue e Amazon Echo. As 20 capturas de malware foram feitas usando um Raspberry Pi.

No conjunto de dados IoT-23, depois que os arquivos .pcap foram gerados, eles foram passados pelo analisador de rede Zeek para gerar arquivos de log. Uma das capacidades do Zeek é produzir arquivos de log de conexão que mostram as propriedades de uma conexão ou um fluxo entre duas entidades [43]. Os arquivos .pcap foram analisados manualmente para identificar as propriedades dos diferentes rótulos. Em seguida, um script python foi executado nos arquivos de log para adicionar rótulos com base na análise. Os tamanhos dos arquivos de malware variavam de alguns kilo Bytes a cerca de 10 Giga Bytes. A unidade de análise é, portanto, um fluxo.

Embora o conjunto de dados IoT-23 tenha vários rótulos, os rótulos têm classes semelhantes. Os rótulos são os diferentes tipos de ataques, mas as classes podem ser uma combinação de diferentes ataques. Por exemplo, um rótulo pode ser C&C ou PartOfAHorizontal

PortScan e ambos têm significados diferentes, enquanto uma classe pode ser C&C-PartOfAHorizontalPortScan, o que significa que ambos os ataques de malware estão presentes para os fluxos dessa classe.

Os rótulos são descritos abaixo [17]:

- Ataque: um tipo de ataque do dispositivo infectado a outro host onde ele tenta tirar proveito de uma vulnerabilidade. • Benigno: nenhuma atividade suspeita ou maliciosa foi encontrada nas conexões.
- C&C: o dispositivo infectado foi conectado a um servidor CC. • DDoS: um ataque distribuído de negação de serviço está sendo executado pelo dispositivo infectado.
- FileDownload: um arquivo está sendo baixado para nosso infectado dispositivo.
- HeartBeat: os pacotes enviados nesta conexão são usados para rastrear o host infectado pelo servidor C&C. • Mirai: as conexões possuem características de um Mirai botnet.
- Okiru: as conexões possuem características de um Okiru botnet.
- PartOfAHorizontalPortScan: As conexões são usadas para fazer uma varredura de porta horizontal para coletar informações para realizar novos ataques. • Torii: as conexões possuem características de um Torii botnet.

A frequência dos fluxos nas classes obtidas a partir dos dados rotulados são mostrados na Tabela 2.

Este conjunto de dados tinha 19 características, conforme mostrado na Tabela 3 ([44], [45]).

Certas características tinham valores ou letras com significados especiais. Essas características foram conn_state e history e a descrição de seus valores pode ser vista na Tabela 4 e Tabela 5 respectivamente [45].

B. SELEÇÃO DE RECURSOS E LIMPEZA DE DADOS

Recursos 'local_orig', 'local_resp' que estavam vazios para todos os arquivos e, portanto, foram descartados. Com base em trabalhos anteriores sobre pré-processamento de conjuntos de dados de detecção de intrusão, recursos como endereços IP e números de porta foram descartados. O recurso 'histórico' era uma sequência de valores que descreviam o histórico da conexão também foi descartado inicialmente. A Fig. 2 mostra o gráfico de correlação das demais feições.

Com base no gráfico de correlação, orig_pkts e orig_ip_bytes se correlacionam, da mesma forma que resp_pkts e resp_ip_bytes. Os recursos correlacionados (orig_ip_bytes e resp_ip_bytes) descartamos. A Fig. 1 mostra boxplots das demais features e como pode ser visto na Fig. 1(a), todos os valores eram zero então esta feature foi descartada. Os boxplots das outras feições mostram variações entre as classes, portanto são distinguíveis. Os recursos finais incluíram proto, serviço, duração, orig_bytes, resp_bytes, conn_state, orig_pkts e resp_pkts.

Classes de minorias extremas com menos de 100 amostras no conjunto de dados foram descartadas; 'C&C-FileDownload', 'Download de arquivo', 'C&C-Torii', 'C&C-HeartBeat-FileDownload',

TABELA 2. Número de fluxos para cada classe no conjunto de dados.

Class	Number of flows
C&C-FileDownload	53
C&C	21995
Benign	30858735
DDoS	19538713
C&C-Torii	30
FileDownload	18
PartOfAHorizontalPortScan	213852924
Attack	9398
C&C-HeartBeat	33673
C&C-HeartBeat-FileDownload	11
C&C-HeartBeat-Attack	834
C&C-PartOfAHorizontalPortScan	888
Okiru	60990708
Okiru-Attack	3
C&C-Mirai	2
PartOfAHorizontalPortScan-Attack	5
Total	**325307990**

TABELA 3. Descrição do recurso do conjunto de dados IoT-23.

	Feature	Description
1	uid	Unique ID
2	id.orig-h	Source IP address
3	id.orig-p	Source port
4	id.resp-h	Destination IP address
5	id.resp-p	Destination port
6	proto	Transaction protocol: icmp, udp, tcp
7	service	dhcp, dns, http, irc, ssh, ssl
8	duration	Total duration of flow
9	orig_bytes	Number of payload bytes the originator sent
10	resp_bytes	Number of payload bytes the responder sent
11	conn_state	Connection state. Possible values are found in Table IV
12	local_orig	T if the connection originated locally and F if it originated remotely
13	local_resp	T if the connection is responded locally and F if it is responded remotely
14	missed_bytes	Number of bytes missed in content gaps, which is representative of packet loss
15	history	State history of connections as a string of letters. The letter is uppercase if it comes from the responder and lowercase if it comes from the originator. Possible letters can be seen in Table V
16	orig_pkts	Number of packets that the originator sent
17	orig_ip_bytes	Number of IP level bytes that the originator sent
18	resp_pkts	Number of packets that the responder sent.
19	resp_ip_bytes	Number of IP level bytes that the responder sent

'PartOfAHorizontalPortScan-Attack', 'Okiru-Attack' e 'C&C-Mirai'. As características 'orig_bytes', 'resp_bytes' e 'duração' continham valores nulos e com base no trabalho anterior, os valores nulos foram substituídos por valor médio das respectivas características. O recurso de 'duração' do tipo de dados foi recodificado de timedelta64 para tempo em segundos. Finalmente, depois de descartar vários recursos, os fluxos restantes continham duplicatas que também foram removidas. Depois de descartar a duplicata

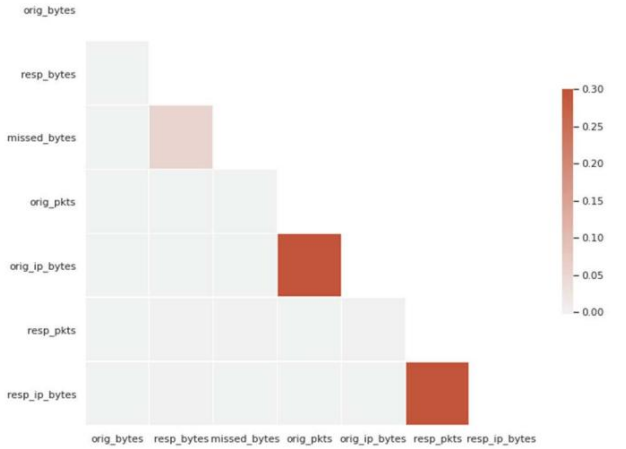

FIGURA 1. Gráfico de correlação das feições.

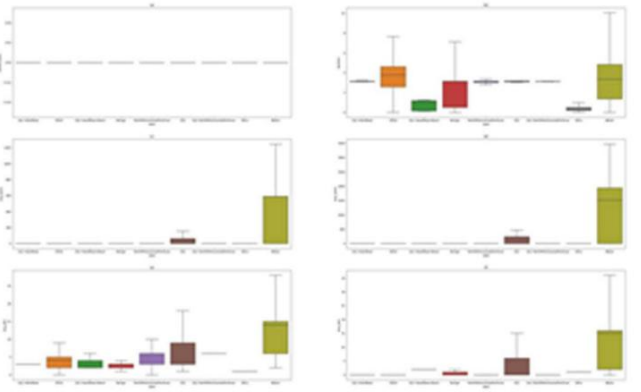

FIGURA 2. Boxplots de diferentes recursos versus rótulos.

fluxos, a Tabela 6 mostra o número final de fluxos restantes em cada classe.

O pré-processamento envolve várias tarefas, como seleção de recursos, codificação, normalização, balanceamento, etc. Faker *et al.* [46] resumiu algumas das técnicas de pré-processamento para dados de rede, incluindo a remoção de informações específicas do soquete, como endereços IP e números de porta, substituição de valores ausentes pelo valor médio do recurso, substituição de valores infinitos pelo valor máximo no recursos, normalizando e codificando os dados.

Recursos categóricos como 'service', 'proto' e 'conn_state' exigiam codificação. Por exemplo, Ieracitano *et al.* [47], Wu *et al.* [48], e Xiao *et al.* [49] usaram codificação one-hot para codificar dados categóricos, enquanto Zhang *et al.* [50] usou codificação variável fictícia. De acordo com o trabalho anterior, o tipo mais comum de codificação era codificação one-hot e, portanto, esses três recursos categóricos eram codificados one-hot. Os dados foram então normalizados entre 0 e 1 usando escala min-max.

Como havia um considerável desequilíbrio de dados mesmo após removendo as classes de extrema minoria, técnicas de balanceamento comuns [51] foram exploradas, nomeadamente Random Over Sampler, Random UnderSampler e Synthetic Minority Oversampling Technique (SMOTE) por Chawla *et al.* [52].

TABELA 4. Descrição dos valores conn_state.

Value	Description
S0	Connection attempt seen, no reply
S1	Connection established, not terminated with no byte count
SF	Normal establishment and termination with byte count
REJ	Connection attempt rejected
S2	Connection established and close attempt by originator seen with no reply from responder
S3	Connection established and close attempt by responder seen with no reply from originator
RSTO	Connection established and originator aborted by sending a RST
RSTR	Responder sent a RST
RSTOS0	Originator sent a SYN followed by a RST and no SYN-ACK seen from the responder
RSTRH	Responder sent a SYN ACK followed by a RST, no SYN seen from the originator
SH	Originator sent a SYN followed by a FIN, no SYN ACK seen from the responder
SHR	Responder sent a SYN ACK followed by a FIN, no SYN seen from the originator
OTH	No SYN seen

TABELA 5. Descrição dos valores históricos.

Value	Description
s	a SYN w/o the ACK bit set
h	a SYN+ACK (“handshake”)
a	a pure ACK
d	packet with payload (“data”)
f	packet with FIN bit set
r	packet with RST bit set
c	packet with a bad checksum (applies to UDP too)
g	a content gap
t	packet with retransmitted payload
w	packet with a zero window advertisement
i	inconsistent packet (e.g. FIN+RST bits set)
q	multi-flag packet (SYN+FIN or SYN+RST bits set)
^	connection direction was flipped by Zeek’s heuristic

TABELA 6. Número de cada classe após descartar duplicatas.

Class	Number
Benign	113,860
DDoS	1,643,225
C&C-HeartBeat-Attack	743
C&C-PartOfAHorizontalPortScan	327
C&C-HeartBeat	10,239
PartOfAHorizontalPortScan	69,198
Okiru	14,942
Attack	9,363
C&C	8,939
Total	**1870836**

Uma combinação de Random UnderSampling e SMOTE foi usada para subamostrar a classe majoritária enquanto sobreamostrava as classes minoritárias.

O regime de balanceamento envolveu o cálculo de 25% da classe majoritária e a amostragem da classe majoritária para esse número. Um limite de 10% desse número de amostra reduzida é selecionado e as classes minoritárias que caíram abaixo desse limite foram amostradas para esse número. Isso ajudou ainda mais a reduzir a taxa de desequilíbrio para 10:1.

C. MODELOS DE DETECÇÃO DE INTRUSÃO

A construção de sistemas de detecção de intrusão usando o conjunto de dados IoT-23 já foi feita antes. Por exemplo, usando apenas as amostras de botnet Mirai do conjunto de dados IoT-23, Hussain *et al.* [53] investigaram a possibilidade de um conjunto de recursos universais que permitiriam que algoritmos de aprendizado de máquina classificassem ataques de botnet Mirai, independentemente do conjunto de dados usado. Eles extraíram recursos dos arquivos .pcap usando o CICFlowmeter e descobriram os 6 principais recursos entre vários conjuntos de dados. Eles criaram conjuntos de treinamento e teste na proporção 80:20 e resultaram em 100% de precisão usando o Random Forest. Da mesma forma, Hegde *et al.* [54] focou na identificação de botnets executando vários classificadores de aprendizado de máquina e aprendizado profundo. Eles usaram o conjunto de dados IoT-23 (apenas para botnets), bem como alguns dados benignos que capturaram de um ambiente de teste. Eles criaram um conjunto de dados pequeno e grande com 95% de dados benignos e 5% maliciosos. Eles alcançaram a maior precisão de 99,9%. Com quatro malwares e três capturas benignas apenas do IoT-23, Dutta *et al.* [44] implementou um modelo de aprendizado profundo para detecção de anomalias usando um método de generalização empilhado, alavancando uma Rede Neural Profunda (DNN) e uma Memória de Longo Prazo (LSTM) com validação cruzada KFold. A maior pontuação de F1 alcançada foi de 0,98 com uma precisão de 0,997. Finalmente, Kumar *et al.* [55] usaram as amostras C&C do conjunto de dados IoT-23 para verificar uma arquitetura.

Este artigo propôs e avaliou três modelos generativos de aprendizado profundo usando o conjunto de dados IoT-23 completo. Cada modelo é descrito a seguir.

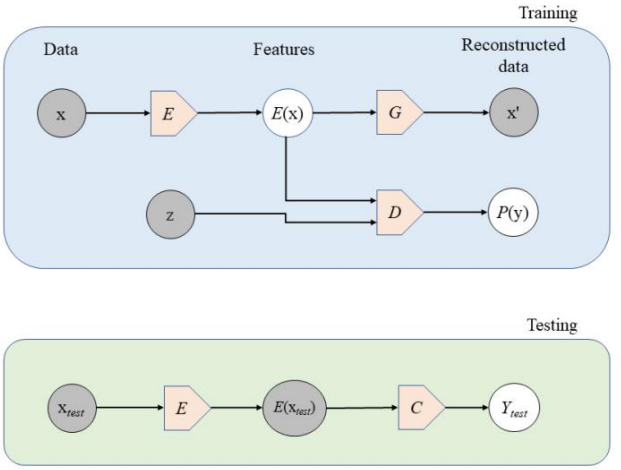

FIGURA 3. AEE e classificador (eg kNN).

D. MODELO DE AUTOENCODER ADVERSARIAL

A Fig. 3 mostra o modelo proposto. Como mostra a figura, um autoencoder adversário [35] pegou uma entrada x de tamanho de recurso 27 e criou uma representação latente E(x) de tamanho 6.
O gerador, G, gera os dados originais, x', a partir dos recursos latentes. O discriminador, D, recebe as feições geradas e um z aleatório e as distingue. A detecção de intrusão é feita codificando os dados de teste para o espaço latente e então treinando um classificador como KNN, por exemplo, para identificar a classe de intrusão (por exemplo, Mirai). Este modelo é similar ao trabalho anterior em [56].

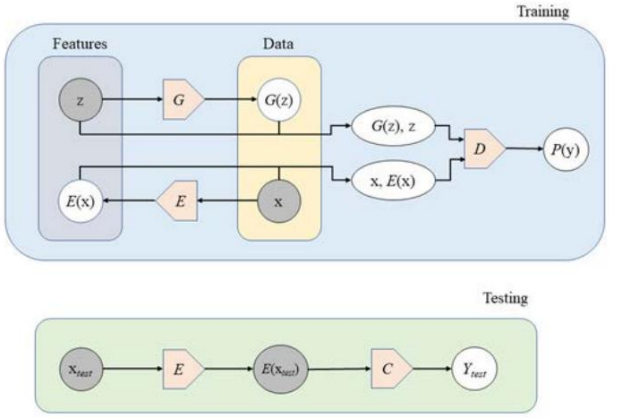

FIGURA 4. BiGAN e classificador (por exemplo, kNN).

Os 27 recursos originais foram usados e reduzidos a uma dimensão latente de tamanho 6 usando o autoencoder. O AAE foi treinado por 1000 épocas com tamanho de lote de 10. Otimizador de Adam com uma taxa de aprendizado de 10^(ÿ4). A equação de perda usada para o autoencoder foi o erro quadrático médio conforme mostrado na equação (1). A função de perda do discriminador e do gerador foi a entropia cruzada binária que é mostrada na equação (2). Tabela 7, Tabela 8 e Tabela 9 mostram as características das camadas do codificador, decodificador e discriminador, respectivamente.

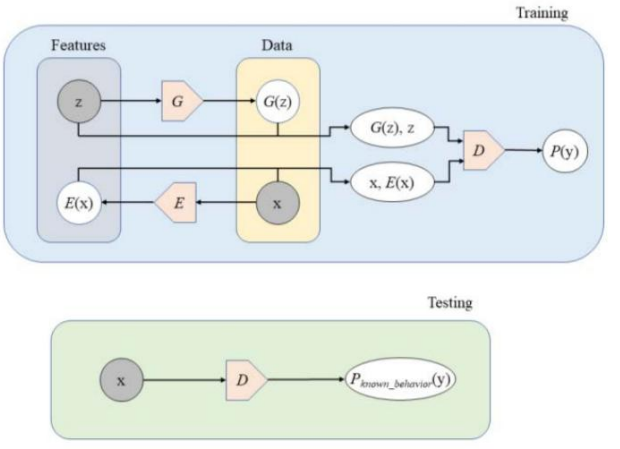

FIGURA 5. Arquitetura BiGAN conhecida/desconhecida.

$$MSE = \frac{1}{n}\sum_{i=1}^{\text{não}} (Ai \text{ ÿ } Pi)^{2} \quad (1)$$

onde *Ai = real, Pi = previsto, n = ponto de dados*

$$\min_{G.}\ \max_{D.}\ \text{Exÿpdata } [logD(x)] + Ezÿp(z)\ [log(1 \text{ ÿ } D(G(z)))] \quad (2)$$

E. ADVERSARIAL GENERATIVO BIDIRECIONAL REDES (BIGAN)

Como mostra a Fig. 4, um BiGAN também foi usado para criar uma representação latente dos dados de entrada. O codificador E pegou todos os 27 recursos e produziu sua representação latente de tamanho

TABELA 7. Características das camadas do codificador AAE.

Layer Number	Neurons	Input Dimension	Activation Function
1	128	41	Relu
2	128	-	Relu
3	16	-	-

TABELA 8. Características das camadas do decodificador AAE.

Layer Number	Neurons	Input Dimension	Activation Function
1	128	16	Relu
2	128	-	Relu
3	41	-	Sigmoid

TABELA 9. Características das camadas do discriminador AAE.

Layer Number	Neurons	Input Dimension	Activation Function
1	128	16	Relu
2	128	-	Relu
3	1	-	Sigmoid

TABELA 10. Características das camadas do codificador BiGAN.

Layer Number	Type	Neurons	Input Dimension	Activation Function
1	Dense	512	27	LeakyRelu
2	BatchNormalization	-	-	-
3	Dense	512	-	LeakyRelu
4	BatchNormalization	-	-	-
5	Dense	8	-	-

8 e o gerador G produziu dados de rede gerados a partir do ruído z do mesmo tamanho que a representação latente. Ambos os conjuntos (dados de entrada, E (dados de entrada)) e (G (ruído z), ruído z)) foram alimentados ao discriminador, D. O BiGAN foi treinado usando 1.000 épocas com um tamanho de lote de 32. Otimizador Adam com um foi utilizada uma taxa de aprendizado de 0,0002. A função objetivo do BiGAN é mostrada na equação (3). Tabela 10, Tabela 11 e Tabela 12 descrevem as camadas do codificador, gerador e discriminador.

$$\min_{G,E}\max_{D} V(D, E, G) \tag{3}$$

onde

$$V(D, E, G) := E_{x\ddot{y}px} E_{z\ddot{y}pE}\underbrace{(|x)[\log D(x,z)]}_{\log D(x,E(x))} + E_{z\ddot{y}pz}\underbrace{Ex\ddot{y}pG(|z)[\log(1\ddot{y}D(G(z),z))]}_{\log(1\ddot{y}D(G(z),z))}$$

F. ADVERSARIAL GENERATIVO BIDIRECIONAL REDES (BIGAN) PARA DETECTAR DESCONHECIMENTOS

Um BiGAN que pode detectar anomalias desconhecidas, bem como seu teste, é mostrado na Fig. 5. A arquitetura de treinamento é

TABELA 11. Características das camadas dos geradores BiGAN.

Layer Number	Type	Neurons	Input Dimension	Activation Function
1	Dense	512	27	LeakyRelu
2	BatchNormalization	-	-	-
3	Dense	512	-	LeakyRelu
4	BatchNormalization	-	-	-
5	Dense	8	-	tanh

TABELA 12. Características das camadas do discriminador BiGAN.

Layer Number	Type	Neurons	Input Dimension	Activation Function
1	Dense	1024	27	LeakyRelu
2	Dropout	-	-	-
3	Dense	1024	-	LeakyRelu
4	Dropout	-	-	-
5	Dense	1024	-	LeakyRelu
6	Dropout	-	-	-
7	Dense	1	-	Sigmoid

TABELA 13. Gerador de teste conhecido/desconhecido.

Layer Number	Type	Neurons	Input Dimension	Activation Function
1	Dense	512	27	LeakyRelu
2	BatchNormalization	-	-	-
3	Dense	512	-	LeakyRelu
4	BatchNormalization	-	-	-
5	Dense	8	-	sigmoid

semelhante ao GAN anterior usado. A arquitetura de teste difere aqui onde estamos usando o discriminador treinado, D, para distinguir se a entrada é um comportamento conhecido ou desconhecido. O BiGAN foi treinado em todos os dados, incluindo benignos e anomalias. Para testar se o BiGAN poderia detectar anomalias desconhecidas, anomalias desconhecidas sintéticas foram criadas selecionando aleatoriamente um subconjunto de recursos (1 ou 2, por exemplo) e, em seguida, alterando aleatoriamente o valor dos recursos. O BiGAN foi treinado por 40.000 épocas com um tamanho de lote de 32 e usou o otimizador Adam com uma taxa de aprendizado de 0,003 e um espaço latente de 8. As arquiteturas usadas para o codificador e as camadas do discriminador são mostradas anteriormente na Tabela 10 e na Tabela 12 As camadas do gerador são descritas na Tabela 13.

4. AVALIAÇÃO A

estratégia de amostragem estratificada de 10 vezes foi usada para avaliação. Essa estratégia gera conjuntos de treinamento e teste em cada divisão. Quando o conjunto de treinamento é gerado, ele é normalizado e balanceado conforme mencionado anteriormente. Neste ponto, diferentes classificadores e GANs foram treinados nos dados de treinamento.

Quatro métricas foram usadas para avaliar os modelos: accu corrida, recall, precisão e F1-score. Dado Verdadeiros Positivos (TP), Verdadeiros Negativos (NP), Falsos Positivos (FP) e Falsos Negativos (FN), as equações de precisão, recall, precisão e pontuação F1 são escritas em (4), (5), (6) e (7) respectivamente.

TABELA 14. Precisão, recall e pontuação de F1 de KNN.

Class	Precision	Std	Recall	Std	F1-score	Std
Benign	0.89	0.0045	0.78	0.0043	0.83	0.0037
DDoS	1.00	0.0000	1.00	0.0000	1.00	0.0000
C&C-HeartBeat-Attack	0.48	0.0408	0.96	0.0246	0.64	0.0360
C&C-PartOfAHorizontalPortScan	0.02	0.0032	0.42	0.0623	0.04	0.0059
C&C-HeartBeat	0.21	0.0095	0.36	0.0118	0.26	0.0103
PartOfAHorizontalPortScan	0.94	0.0024	0.76	0.0050	0.84	0.0032
Okiru	0.43	0.0090	0.69	0.0132	0.53	0.0101
Attack	0.76	0.0118	0.92	0.0079	0.83	0.0079
C&C	0.39	0.0079	0.52	0.0099	0.44	0.0075

TABELA 15. Precisão de RF, rechamada e pontuação F1.

Class	Precision	Std	Recall	Std	F1-score	Std
Benign	0.84	0.0037	0.73	0.0049	0.78	0.0038
DDoS	1.00	0.0000	1.00	0.0001	1.00	0.0000
C&C-HeartBeat-Attack	0.54	0.0516	0.83	0.0359	0.65	0.0405
C&C-PartOfAHorizontalPortScan	0.01	0.0026	0.30	0.0892	0.02	0.0050
C&C-HeartBeat	0.10	0.0058	0.18	0.0092	0.13	0.0070
PartOfAHorizontalPortScan	0.86	0.0038	0.67	0.0079	0.75	0.0055
Okiru	0.37	0.0070	0.50	0.0118	0.43	0.0075
Attack	0.83	0.0114	0.88	0.0089	0.86	0.0070
C&C	0.29	0.0112	0.47	0.0122	0.36	0.0116

TABELA 16. Precisão AAE + KNN, recall e F1-score.

Class	Precision	Std	Recall	Std	F1-score	Std
Benign	0.99	0.0100	0.99	0.0122	0.99	0.0100
DDoS	1.00	0.0000	1.00	0.0000	1.00	0.0000
C&C-HeartBeat-Attack	0.99	0.0164	1.00	0.0092	0.99	0.0100
C&C-PartOfAHorizontalPortScan	0.94	0.0739	0.93	0.0907	0.93	0.0751
C&C-HeartBeat	0.93	0.0843	0.93	0.0772	0.93	0.0789
PartOfAHorizontalPortScan	0.99	0.0092	0.99	0.0092	0.99	0.0092
PartOfAHorizontalPortScan	0.99	0.0092	0.99	0.0092	0.99	0.0092
Okiru	0.98	0.0264	0.98	0.0313	0.98	0.0290
Attack	1.00	0.0000	1.00	0.0030	1.00	0.0000
C&C	0.96	0.0439	0.96	0.0372	0.96	0.0408

TABELA 17. Precisão BiGAN + KNN, recall e F1-score.

Class	Precision	Std	Recall	Std	F1-score	Std
Benign	0.99	0.0122	0.99	0.0104	0.99	0.0100
DDoS	1.00	0.0000	1.00	0.0000	1.00	0.0000
C&C-HeartBeat-Attack	1.00	0.0090	0.99	0.0166	0.99	0.0100
C&C-PartOfAHorizontalPortScan	0.94	0.0764	0.92	0.0837	0.93	0.0748
C&C-HeartBeat	0.93	0.0789	0.93	0.0808	0.93	0.0815
PartOfAHorizontalPortScan	0.99	0.0092	0.99	0.0092	0.99	0.0092
Okiru	0.98	0.0284	0.98	0.0295	0.98	0.0290
Attack	1.00	0.0000	1.00	0.0030	1.00	0.0000
C&C	0.96	0.0395	0.95	0.0454	0.96	0.0408

A pontuação F1 é usada aqui porque métricas alternativas como

A precisão é potencialmente enganosa para conjuntos de dados desequilibrados que são comuns em tarefas de detecção de anomalias. Como a pontuação F1 é uma média geométrica de recuperação e precisão, essa métrica fornece resultados mais significativos para conjuntos de dados desequilibrados.

$$Precisão = \frac{(TP + NT)}{todos} \quad (4)$$

$$lembre\text{-}se = \frac{PT}{(TP + FN)} \quad (5)$$

$$Precisão = \frac{PT}{(TP + PF)} \quad (6)$$

$$F1 = \frac{2\times Recall \times Precisão}{(Recall + Precisão)} \quad (7)$$

K Nearest Neighbor (KNN) e Random Forest (RF) foram usados para comparação de linha de base. A Tabela 14 e a Tabela 15 mostram os resultados da precisão, recall e F1-score do KNN e do RF, respectivamente.

A Tabela 14 e a Tabela 15 mostram que RF e KNN não tiveram um bom desempenho e resultaram em pontuações F1 tão baixas quanto 0,02 em alguns casos.

Os resultados para usar Adversarial Autoencoder e BiGAN são mostrados na Tabela 16 e na Tabela 17, respectivamente.

A Tabela 19 mostra que todas as métricas, incluindo acurácia, precisão, recall e F1-Score, foram diferentes ao longo do

TABELA 18. Comparação dos vários modelos com base na macro média F1.

Model	Average Macro F1 Scores	Std
KNN	0.6019	0.0050
RF	0.5518	0.0055
AAE + KNN	0.9743	0.0182
BiGAN + KNN	0.9741	0.0186

TABELA 19. Comparação estatística de várias métricas dos modelos (teste de Kruskal-Wallis; N = 10).

Metric	Test Statistics	p-value
Accuracy	32.986	0
F1-Score	32.934	0
Precision	32.943	0
Recall	32.933	0

TABELA 20. Métricas de desempenho para o BiGAN detectar incógnitas.

# features changed/ metric	1	2	4	8	16	27
Average Precision	1	1	1	1	1	1
Average Recall	0.81	0.95	1	1	1	1
Average Accuracy	0.89	0.97	1	1	1	1
Average F1-scores	0.85	0.94	1	1	1	1

vários modelos. Os testes par a par de Mann-Whitney ($p < 0,05$) mostraram que AAE + KNN e BiGAN + KNN não foram significativamente diferentes para todas as métricas, embora ambos fossem diferentes de RF e KNN para todas as métricas.

A Tabela 18 resume os resultados. Como pode ser visto, tanto o AAE quanto o BiGAN tiveram F1-Score significativamente melhor do que KNN ou Random Forest.

A Tabela 19 mostra que todas as métricas, incluindo exatidão, precisão, recall e F1-Score, foram diferentes nos vários modelos. Os testes par a par de Mann-Whitney ($p < 0,05$) mostraram que AAE + KNN e BiGAN + KNN não foram significativamente diferentes para todas as métricas, embora ambos fossem diferentes de RF e KNN para todas as métricas.

Por fim, os modelos BiGAN e AAE também foram avaliados quanto à qualidade dos dados gerados. A primeira avaliação foi o Leave One Out (LOO) do KNN usando a Distância Euclidiana proposta por Guan *et al.* [57]. Nesta avaliação, os dados reais rotulados como 1 e os dados gerados pela GAN rotulados como 0 foram passados para um KNN. Para o AAE, os dados sintéticos foram gerados usando o decodificador em entradas aleatórias do espaço latente. O KNN (com distância = 1) foi então treinado em todos os dados, exceto em uma instância que foi usada para teste.

Tanto para AAE quanto para BiGAN, a precisão de quase 100% significa que o decodificador e o gerador não estavam gerando dados da mesma distribuição. No entanto, o mapeamento para o espaço latente assim gerado foi suficiente para uma melhor classificação do que o espaço original.

A segunda avaliação foi o GAN-train e GAN-train conforme proposto por Shmelkov *et al.* [58]. Nesta avaliação, o classificador, como KNN, foi utilizado duas vezes. O classificador foi treinado primeiro em dados reais de trens e avaliado em GAN/AAE dados gerados. O KNN também foi treinado em dados gerados e testado em dados de teste reais. Tanto no AAE quanto no BiGAN, a precisão foi quase zero, suportando a conjectura de que onde o AAE e o BiGAN foram capazes de representar os dados, mas não puderam gerar dados provenientes da mesma distribuição.

A Tabela 20 mostra que o BiGAN para detectar anomalias foi muito eficaz com F1-Scores de 1, pois o número de mutações para recursos foi aumentado. Mesmo para uma única mutação, o modelo teve um F1-Score decente de 0,85.

V. DISCUSSÃO

Conforme observado na seção de resultados, as precisões e as pontuações F1 de AAE + KNN e BiGAN + KNN foram quase as mesmas. Ambos os modelos foram capazes de reconhecer DDoS e Attack. As classes C&C-PartOfAHorizontalPortScan e C&C-HeartBeat obtiveram os menores escores de F1 (0,93). Isso pode ter surgido por dois motivos. A primeira razão é que ambas as classes incluíam dispositivos conectados ao servidor C&C e a segunda razão é que tanto PartOfHorizontalPortScan quanto HeartBeat significavam que os dispositivos estavam sendo rastreados, mas de maneiras diferentes.

Como os arquivos .pcap dos dados coletados estão disponíveis, analisadores de rede adicionais podem ser usados para recuperar recursos adicionais. Também há mais trabalho a ser feito para aumentar o número de instâncias das classes minoritárias.

Outro aspecto interessante que pode ser feito para a detecção de anomalias é encontrar as características compartilhadas entre diferentes ataques e possivelmente detectar novos ataques que compartilham essas mesmas características.

SERRA. CONCLUSÃO

O rápido aumento na implantação de dispositivos IoT os tornou um participante perigoso e desavisado em ataques cibernéticos.

Este artigo mostrou que, para um conjunto limitado de ataques e dispositivos IoT, é possível usar métodos generativos de aprendizado profundo como AAE e BiGAN para classificar ataques com uma precisão muito alta. Embora existam vários conjuntos de dados sobre detecção de intrusão, é melhor usar um conjunto de dados gerado a partir de dispositivos IoT. Portanto, neste artigo, usamos um conjunto de dados recente chamado IoT-23. Implementamos modelos de linha de base, Adversarial Autoencoders e GANs bidirecionais.

Nossos resultados mostram que os modelos baseados em GAN são mais eficazes na identificação e classificação de ataques. Também tentamos randomizar o conjunto de teste de forma que possamos injetar novas informações e o modelo foi capaz de considerá-lo uma anomalia.

AGRADECIMENTOS Este

artigo representa a opinião dos autores e não representa a posição ou opiniões da American University of Sharjah.

REFERÊNCIAS

[1] B. Alqahtani e B. AlNajrani, "Um estudo dos protocolos e comunicação da Internet das Coisas", em *Proc. 2ª Int. Conf. Comp. Inf. Sci. (ICCIS),* outubro de 2020, p. 1–6, doi: 10.1109/ICCIS49240.2020.9257652.

[2] W. Yang, "Pesquisa sobre problemas de segurança de rede e contramedidas baseadas na tecnologia da Internet das Coisas," *J. Phys., Conf. ser.,* vol. 1744, nº. 4, fev. 2021, art. 042010, doi: 10.1088/ 1742-6596/1744/4/042010.

[3] S. Fenanir, F. Semchedine, S. Harous e A. Baadache, "Uma detecção de intrusão baseada em codificador automático profundo semi-supervisionado para IoT," *Ingénierie Syst. informação,* vol. 25, não. 5, pág. 569–577, novembro de 2020, doi: 10.18280/ isi.250503.

[4] C. Vorakulpipat, E. Rattanalerdnusorn, P. Thaenkaew e HD Hai, "Desafios, tendências e preocupações recentes relacionados à segurança da IoT: um estudo evolutivo", em Proc . *20ª Int. Conf. Adv. comum. Tech nol. (ICACT),* fevereiro de 2018, p. 405–410, doi: 10.23919/ICACT.2018. 8323774.

[5] SA Butt, JL Diaz-Martinez, T. Jamal, A. Ali, E. De-La-Hoz-Franco e M. Shoaib, "Ameaças de segurança de saúde inteligentes da IoT", em Proc . *19ª Int. Conf. Comput. Sci.Appl. (ICCSA),* julho de 2019, p. 26–31, doi: 10.1109/ICCSA.2019.000-8.

[6] M. Burhan, RA Rehman, B. Khan e B.-S. Kim, "Elementos de IoT, arquiteturas em camadas e questões de segurança: uma pesquisa abrangente," *Sensors,* vol. 18, não. 9, Set. 2018, Art. n. 9, doi: 10.3390/s18092796.

[7] A. Hameed e A. Alomary, "Questões de segurança em IoT: uma pesquisa", em *Proc. Int. Conf. Innov. Intel. Inform., Comput., Technol. (3ICT),* set. 2019, p. 1–5, doi: 10.1109/3ICT.2019.8910320.

[8] I. Andrea, C. Chrysostomou e G. Hadjichristofi, "Internet das coisas: vulnerabilidades e desafios de segurança", em *Proc. IEEE Simp. computador comum. (ISCC),* julho de 2015, p. 180–187, doi: 10.1109/ISCC.2015.7405513.

[9] K. Angrishi, "Transformando a Internet das Coisas (IoT) na Internet dos vulneráveis ities (IoV): IoT botnets," 2017, *arXiv:1702.03681.*

[10] R. Ahmad e I. Alsmadi, "Abordagens de aprendizado de máquina para segurança de IoT: uma revisão sistemática da literatura", *Internet Things,* vol. 14 de junho de 2021, Art. n. 100365, doi: 10.1016/j.iot.2021.100365.

[11] S. Das, PP Amritha e K. Praveen, "Detecção e prevenção de ataque mirai", em *Proc. Computador macio. Signal Process.,* Cingapura, 2021, p. 79–88.

[12] B. Tushir, H. Sehgal, R. Nair, B. Dezfouli e Y. Liu, "O impacto dos ataques DoS em dispositivos IoT com restrição de recursos: um estudo sobre o ataque Mirai," 2021, *arXiv :2104.09041.*

[13] C. Kolias, G. Kambourakis, A. Stavrou e J. Voas, "DDoS in the IoT: Mirai and other Botnets," *Computer,* vol. 50, não. 7, pág. 80–84, 2017, doi: 10.1109/MC.2017.201.

[14] *Dispositivos conectados à IoT em todo o mundo 2019-2030.* Acesso: 17 de junho de 2021. [On-line]. Disponível: https://www.statista.com/statistics/1183457/iot-connected-devices-worldwide/

[15] *Relatório Anual da Internet da Cisco — White Paper do Relatório Anual da Internet da Cisco (2018-2023).* Acesso: 31 de maio de 2021. [Online]. Disponível: https://www.cisco.com/c/en/us/solutions/collateral/executive perspectives/ annual-internet-report/white-paper-c11-741490.html [16] S. Hajiheidari, K. Wakil , M. Badri e NJ Navimipour, "Sistemas de detecção de intrusão na Internet das coisas: uma investigação abrangente," *Comput. netw.,* vol. 160, pág. 165–191, set. 2019, doi: 10.1016/j.comnet.2019.05.014.

[17] S. Garcia, A. Parmisano e MJ Erquiaga, "IoT-23: Um conjunto de dados rotulado com tráfego de rede IoT malicioso e benigno (versão 1.0.0)," Zenodo, *vol* . 20, pág. 15, jan. 2020, doi: 10.5281/zenodo.4743746.

[18] G. Pang, C. Shen, L. Cao e AVD Hengel, "Aprendizado profundo para detecção de anomalias: uma revisão", *ACM Comput. Surv.,* vol. 54, nº. 2, pág. 1–38, março de 2021, doi: 10.1145/3439950.

[19] PAA Resende e AC Drummond, "Uma pesquisa de métodos baseados em floresta aleatória para sistemas de detecção de intrusão," *ACM Comput. Surv.,* vol. 51, nº. 3, pág. 48, 2018, doi: 10.1145/3178582.

[20] Y. Li, Y. Xu, Z. Liu, H. Hou, Y. Zheng, Y. Xin, Y. Zhao e L. Cui, " Detecção robusta para invasão de rede de IoT industrial baseada em multi CNN fusão," *Medição,* vol. 154, março de 2020, art. 107450, doi: 10.1016/j.measurement.2019.107450.

[21] M. Tavallaee, E. Bagheri, W. Lu e AA Ghorbani, "Uma análise detalhada do conjunto de dados KDD CUP 99," em *Proc. IEEE Simp. computador Intel. Seguro. Pedido de defesa,* julho de 2009, p. 1–6, doi: 10.1109/CISDA.2009. 5356528.

[22] S. Latif, Z. Idrees, Z. Zou e J. Ahmad, "DRaNN: um modelo de rede neural aleatória profunda para detecção de intrusão em IoT industrial", em Proc . *Int. Conf. Reino Unido-China Emerg. Tecnol. (UCET),* ago. 2020, pág. 1–4, doi: 10.1109/ UCET51115.2020.9205361.

[23] N. Moustafa e J. Slay, "UNSW-NB15: Um conjunto de dados abrangente para sistemas de detecção de intrusão de rede (conjunto de dados de rede UNSW-NB15)," em Proc . *Mil. comum. Sistema de Informação Conf. (MilCIS),* Nov. 2015, p. 1–6, doi: 10.1109/ MilCIS.2015.7348942.

[24] Y. Xu, Y. Tang e Q. Yang, "Aprendizado profundo para detecção de intrusão IoT baseado em LSTMs-AE," em *Proc. 2ª Int. Conf. Artif. Intel. Adv. Manuf.,* Nova York, NY, EUA, outubro de 2020, p. 64–68, doi: 10.1145/3421766.3421891.

[25] Y. Mirsky, T. Doitshman, Y. Elovici e A. Shabtai, "Kitsune: Um conjunto de autoencoders para detecção de intrusão de rede online", em *Proc. Netw. Distribui. sist. Seguro. Symp.,* 2018, p. 1–15.

[26] MH Shahriar, NI Haque, MA Rahman e M. Alonso, Jr., "G-IDS: Sistema de detecção de intrusão assistido por redes adversárias generativas", 2020, arXiv:2006.00676 .

[27] W. Fan, M. Miller, SJ Stolfo, W. Lee e PK Chan, "Usando anomalias artificiais para detectar invasões de rede desconhecidas e conhecidas" em *Proc. IEEE Int. Conf. Data Mining,* novembro de 2001, pp. 123–130, doi: 10.1109/ICDM.2001.989509.

[28] SP RM, PKR Maddikunta, M. Parimala, S. Koppu, TR Gadekallu, CL Chowdhary e M. Alazab, "Uma engenharia de recursos eficaz para DNN usando híbrido PCA-GWO para detecção de intrusão na arquitetura IoMT," *computador com.,* vol. 160, pág. 139–149, julho de 2020, doi: 10.1016/j.comcom.2020.05.048.

[29] MR Shahid, G. Blanc, H. Jmila, Z. Zhang e H. Debar, "Aprendizado profundo generativo para geração de tráfego de rede da Internet das coisas", em 2020 IEEE 25th Pacific Rim Int. Symp . *Computação confiável. (PRDC),* dezembro de 2020, p. 70–79, doi: 10.1109/PRDC50213.2020.00018.

[30] M. Salem, S. Taheri e JS Yuan, "Geração de anomalias usando redes adversárias generativas na detecção de intrusão baseada em host", em *Proc. 9ª IEEE Ann. Computação ubíqua. Elétron. Móvel Comun. Conf.,* nov. 2018, pág. 683–687, doi: 10.1109/UEMCON.2018.8796769.

[31] S. Huang e K. Lei, "IGAN-IDS: Uma rede adversária generativa desequilibrada para o sistema de detecção de intrusão em trabalhos de rede ad-hoc," Ad Hoc Netw., *vol* . 105, agosto de 2020, Art. n. 102177, doi: 10.1016/j.adhoc.2020.102177.

[32] I. Yilmaz, R. Masum e A. Siraj, "Abordando o problema de dados desequilibrados com rede adversária generativa para detecção de intrusão", em *Proc. IEEE 21st Int. Conf. Inf. Reuse Integr. Data Sci. (IRI),* agosto de 2020, p. 25–30, doi: 10.1109/ IRI49571.2020.00012.

[33] I. Sharafaldin, AH Lashkari e AA Ghorbani, "Para gerar um novo conjunto de dados de detecção de intrusão e caracterização do tráfego de intrusão", em *Proc. 4ª Int. Conf. Inf. Syst. Seguro. Privacidade,* 2018, p. 108–116, doi: 10.5220/0006639801080116.

[34] R. Chauhan e S. Shah Heydari, "Ataque DDoS contraditório polimórfico em IDS usando GAN", em *Proc. Int. Simp. Netw., Comput. comum. (ISNCC),* outubro de 2020, p. 1–6, doi: 10.1109/ISNCC49221.2020.9297264.

[35] SM Lundberg, GG Erion e S.-I. Lee, "Atribuição consistente de recursos individualizados para conjuntos de árvores", 2018, *arXiv:1802.03888.*

[36] A. Makhzani, J. Shlens, N. Jaitly, I. Goodfellow e B. Frey, "Adversarial autoencoders," 2015, *arXiv:1511.05644.*

[37] S. Puuska, T. Kokkonen, J. Alatalo e E. Heilimo, "Detecção de intrusão de trabalho de rede baseada em anomalias usando wavelets e autoencoders adversários", em Proc . *inov. Seguro. Soluções Inf. Technol. Commun.,* Cham, Suíça, 2019, p. 234–246, doi: 10.1007/978-3-030-12942-2_18.

[38] K. Hara e K. Shiomoto, "Sistema de detecção de intrusão usando aprendizado semi-supervisionado com codificador automático adversário," em *Proc. Rede IEEE/ IFIP. op. Gerenciar. Symp.,* abr. 2020, p. 1–8, doi: 10.1109/ NOMS47738.2020.9110343.

[39] MO Kaplan e SE Alptekin, "Uma abordagem baseada em BiGAN aprimorada para detecção de anomalias", *Proc. computador Sci.,* vol. 176, pág. 185–194, 2020, doi: 10.1016/j.procs.2020.08.020.

[40] J. Donahue, P. Krähenbähl e T. Darrell, "Adversarial feature learning," 2016, *arXiv:1605.09782.*

[41] J. Donahue e K. Simonyan, "Aprendizagem de representação adversarial em larga escala", 2019, *arXiv:1907.02544.*

[42] SK Alabugin e AN Sokolov, "Aplicação de redes adversárias generativas para detecção de anomalias em sistemas de controle industrial", em *Proc. Global Smart Ind. Conf. (GloSIC),* novembro de 2020, p. 199–203, doi: 10.1109/GloSIC50886.2020.9267878.

[43] *Introdução ao script — Livro de Zeek (Git/ Master).* Acesso: 19 de maio de 2021. [Online]. Disponível: https://docs.zeek.org/en/master/scripting/intro.html#writing-scripts-connection-record

[44] V. Dutta, M. Chora, M. Pawlicki e R. Kozik, "Um conjunto de aprendizado profundo para anomalias de rede e detecção de ataques cibernéticos," *Sensors,* vol. 20, não. 16, Jan. 2020, Art. n. 16, doi: 10.3390/s20164583.

[45] *Base/ Protocols/ Conn/ Main.Zeek—Livro de Zeek (V4.0.1).* Acesso: 19 de maio de 2021. [Online]. Disponível: https://docs.zeek.org/en/lts/scripts/base/protocols/conn/main.zeek.html [46] O. Faker e E. Dogdu, " Detecção de intrusão usando big data e aprendizagem profunda técnicas," no *Proc. ACM Southeast Conf.,* Nova York, NY, EUA, abril de 2019, p. 86–93, doi: 10.1145/3299815.3314439.

[47] C. Ieracitano, A. Adeel, M. Gogate, K. Dashtipor, FC Morabito, H. Larijani, A. Raza e A. Hussain, "Sistema de aprendizado profundo otimizado orientado por análise estatística para detecção de intrusão" 2018, *arXiv:1808.05633.*

[48] K. Wu, Z. Chen e W. Li, "Um novo modelo de detecção de intrusão para uma rede massiva usando redes neurais convolucionais," *IEEE Access,* vol. 6, pág. 50850–50859, 2018, doi: 10.1109/ACCESS.2018.2868993.

[49] Y. Xiao, C. Xing, T. Zhang e Z. Zhao, "Um modelo de detecção de intrusão baseado em redução de recursos e redes neurais convolucionais", *IEEE Access,* vol. 7, pág. 42210–42219, 2019, doi: 10.1109/ACCESS.2019.2904620.

[50] L. Zhang, M. Li, X. Wang e Y. Huang, "Uma detecção de intrusão de rede aprimorada baseada em rede neural profunda", *IOP Conf. Ser. Mater. Sci. Eng.,* vol. 563, agosto de 2019, art. 052019, doi: 10.1088/1757-899X /563/5/052019.

[51] JM Johnson e TM Khoshgoftaar, "Pesquisa sobre aprendizado profundo com desequilíbrio de classes", *J. Big Data,* vol. 6, pág. 27 de dezembro de 2019, doi: 10.1186/s40537-019-0192-5 .

[52] NV Chawla, KW Bowyer, LO Hall e WP Kegelmeyer, "SMOTE: técnica de sobreamostragem minoritária sintética", *J. Artif. Intel. Res.,* vol. 16, pág. 321–357, junho de 2002, doi: 10.1613/jair.953.

[53] F. Hussain, SG Abbas, UU Fayyaz, GA Shah, A. Toqeer e A. Ali, "Towards a universal features set for IoT botnet attack detection," 2020, arXiv:2012.00463 .

[54] M. Hegde, G. Kepnang, M. Al Mazroei, JS Chavis e L. Watkins, "Identificação da atividade de botnet no tráfego de rede iot usando aprendizado de máquina" em Proc . *Int. Conf. Intl. Data Sci. Technol. Appl. (IDSTA),* outubro de 2020, p. 21–27, doi: 10.1109/IDSTA50958.2020.9264143.

[55] A. Kumar, M. Shridhar, S. Swaminathan e T. Joon Lim, "Detecção precoce baseada em aprendizado de máquina de botnets IoT usando tráfego de borda de rede", 2020, arXiv:2010.11453 .

[56] F. Aloul, I. Zualkernan, N. Abdalgawad, L. Hussain e D. Sakhnini, "Detecção de intrusão de rede na borda IoT usando codificadores automáticos adversários", em Proc . *Int. Conf. Inf. Technol. (ICIT),* julho de 2021, p. 120–125, doi: 10.1109/ICIT52682.2021.9491694.

[57] S. Guan e M. Loew, "Avaliação do desempenho da rede adversária generativa com base na análise direta das imagens geradas", em *Proc. Aplicativo IEEE imagem Padrão reconhecido. Workshop (AIPR),* outubro de 2019, p. 1–5, doi: 10.1109/AIPR47015.2019.9174595.

[58] K. Shmelkov, C. Schmid e K. Alahari, "How good is my GAN?" 2018, *arXiv:1807.09499.*

N. ABDALGAWAD (membro estudante, IEEE) recebeu o B.Sc. graduou-se *(cum laude)* em engenharia da computação pela American University of Sharjah, Emirados Árabes Unidos, em 2020, onde atualmente cursa o M.Sc. graduação em engenharia da computação. Ela também está trabalhando como professora de pós-graduação e assistente de pesquisa na American University of Sharjah. Seus interesses de pesquisa incluem aprendizado de máquina, segurança cibernética, matrizes de portas programáveis em campo e computação em nuvem. Ela é membro da Upsilon Pi Epsilon Honor Society e IEEE-HKN. Ela recebeu a bolsa Abdulla Al Ghurair Foundation for STEM Education. Ela ganhou o prêmio de Melhor Assistente de Ensino de Pós-Graduação do ano de 2020 no Departamento de Ciência da Computação e Engenharia da American University of Sharjah.

A. SAJUN recebeu o diploma de bacharel em engenharia da computação em 2020. Atualmente, ele está cursando o mestrado em engenharia da computação na American University of Sharjah. Ele também está trabalhando como assistente de pesquisa no campo de aprendizado profundo, Internet das Coisas e sistemas inteligentes de energia solar. Seus interesses de pesquisa incluem aprendizado profundo, Internet das Coisas, monitoramento automatizado da vida selvagem e energia inteligente.

Y. KADDOURA (membro estudante, IEEE) recebeu o B.Sc. graduada (Hons.) em engenharia da computação e menor em ciência da computação pela American University of Sharjah (AUS), em 2019, onde atualmente cursa o M.Sc. graduação em engenharia da computação. Ela também está trabalhando como assistente de pesquisa em diagnóstico de falhas em máquinas de estado finito. Seus interesses de pesquisa incluem computação em nuvem, aprendizado de máquina e segurança cibernética. Ela é um membro do Upsilon Pi Epsilon Honor Society e IEEE-HKN Honor Society.

IA ZUALKERNAN (membro, IEEE) recebeu o BS (Hons.) e Ph.D. graduou-se em ciência da computação pela University of Minnesota, Minneapolis, em 1983 e 1991, respectivamente. Ele foi professor assistente no Departamento de Engenharia Elétrica e de Computação da Universidade Estadual da Pensilvânia, de 1992 a 1995. Ele foi engenheiro de projeto principal da AMCS Inc., Chanhassen, Minnesota, de 1995 a 1998.

Ele era o CEO da Askari Information Systems, de 1998 a 2000, e Chief Technology Officer da Knowledge Platform, Inc., Cingapura, de 2000 a 2003. Em 2003, ingressou na American University of Sharjah, nos Emirados Árabes Unidos, onde atualmente é professor de informática ciência e engenharia. Ele é o autor ou co-autor de mais de 200 artigos revisados por pares e recebeu o prêmio Chester Sall da IEEE Consumer Electronics Society 2020.

Seus interesses de pesquisa incluem sistemas de consumo, aplicativos de internet baseados em sensores, AIoT para dispositivos de consumo e Internet das Coisas (IoT).

F. ALOUL (Membro Sênior, IEEE) recebeu o título de BS *(summa cum laude)* em engenharia elétrica da Lawrence Technological University, Michigan, EUA, e o MS e Ph.D. formado em ciência da computação e engenharia pela University of Michigan, Ann Arbor, EUA. Atualmente é Professor e Chefe do Departamento de Ciência da Computação/Engenharia e Diretor do HP Institute, American University of Sharjah (AUS), Emirados Árabes Unidos. Possui mais de 130 publicações em periódicos e conferências internacionais, além de uma patente nos Estados Unidos. Seus interesses de pesquisa atuais incluem segurança cibernética, aplicativos móveis e otimização de design.

Ele recebeu vários prêmios, incluindo o Prêmio de Diversidade de Engenharia da Airbus do Global Engineering Deans Council (GEDC), o Prêmio Sheikh Khalifa de Ensino Superior, o Prêmio AUS de Excelência em Ensino, o Prêmio Abdul Hameed Shoman para Jovens Pesquisadores Árabes e o Prêmio Sheikh Prêmio de Rashid para Outstanding Achievement Scientific. Ele é palestrante convidado regular e palestrante em várias conferências internacionais relacionadas à segurança cibernética, tecnologia, inovação e educação. Ele também é um Certified Information Systems Security Professional (CISSP).