Emprego de Redes Neurais Artificiais combinadas com Algoritmos Genéticos na detecção de intrusões em redes de computadores

Heitor Eugênio Gonçalves, Gabriel Andrade Queiroz, Éderson Rosa da Silva

***Resumo*— Neste trabalho, propõe-se uma técnica que combina Redes Neurais Artificiais (RNAs) e Algoritmos Genéticos (AGs) para a detecção de intrusões em redes de computadores. Para a validação da metodologia, foi utilizado o banco de dados CIC-IDS2018, que contém registros detalhados de tráfego de rede durante a realização de diversas intrusões. A aplicação da combinação dessas técnicas resultou em uma acurácia de 99,78% na identificação de intrusões, demonstrando a eficácia e robustez da proposta na detecção de ameaças em ambientes de rede.**

***Palavras-Chave*— Algoritmo Genético, Rede de computadores, Redes Neurais Artificiais.**

***Abstract*— In this work, a technique that combines Artificial Neural Networks (ANNs) and Genetic Algorithms (GAs) for intrusion detection in computer networks is proposed. To validate the methodology, the CIC-IDS2018 dataset was used, which contains detailed records of network traffic during various intrusion attempts. The application of this combination of techniques resulted in an accuracy of 99.78% in identifying intrusions, demonstrating the effectiveness and robustness of the proposal in detecting threats in network environments.**

***Keywords*— Artificial Neural Networks, Computer Network, Genetic Algorithm.**

I. Introdução

A detecção de intrusões é fundamental para preservar a segurança de redes de computadores, diante do aumento constante dos ataques cibernéticos [1]. Com o avanço de tecnologias como a Internet das Coisas (IoT) e redes 5G, cresceu a necessidade de sistemas automáticos e inteligentes de defesa [2]. Nesse cenário, o desenvolvimento de técnicas mais eficientes para identificar ameaças tornou-se um desafio essencial para a segurança da informação.

A crescente complexidade dos ataques a sistemas computacionais exige soluções avançadas para detecção de intrusões. Neste contexto, este trabalho propõe o uso combinado de Redes Neurais Artificiais (RNAs) e Algoritmos Genéticos (AGs) para otimizar o processo de detecção de intrusões em redes.

As RNAs possuem capacidade de classificação e predição, sendo amplamente utilizadas para detectar padrões anômalos em tráfegos de rede. Contudo, a definição de seus hiperparâmetros costuma ser feita empiricamente, o que pode demandar tempo e recursos.

Heitor Eugênio Gonçalves, e-mail: heitor.goncalves@ufu.br; Gabriel Andrade Queiroz, e-mail: gabriel.andrade@ufu.br; Éderson Rosa da Silva, e-mail: ersilva@ufu.br Faculdade de Engenharia Elétrica – Universidade Federal de Uberlândia(UFU). O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001.

Inspirados em mecanismos da evolução natural, os AGs operam por meio de recombinação genética e seleção de indivíduos mais aptos, aplicando esse princípio à otimização de funções. Neste trabalho, são utilizados para ajustar hiperparâmetros e atributos de entrada da RNA de forma automatizada, reduzindo significativamente o tempo de configuração e elevando a eficiência do modelo.

O artigo está estruturado da seguinte forma: a Seção II apresenta os trabalhos relacionados; a Seção III aborda a fundamentação teórica; a Seção IV descreve a metodologia; a Seção V discute os resultados e a Seção VI apresenta as conclusões e sugestões para trabalhos futuros.

II. Trabalhos relacionados

Dentre os conjuntos de dados mais relevantes para detecção de intrusões, destaca-se o CSE-CIC-IDS2018, desenvolvido por Sharafaldin et al. [11]. Esse banco contém tráfego real e simulado, com ataques como força bruta, *Denial of Service* (DoS), *botnet* e *web*, coletado por meio do CICFlowMeter. A infraestrutura simula um ambiente corporativo com 420 máquinas e 30 servidores.

Shieh et al. [12] propuseram uma arquitetura baseada em *Bi-directional Long Short-Term Memory* (BI-LSTM) para detectar ataques DDoS. Embora a abordagem teórica seja promissora, os resultados obtidos foram limitados, com baixa acurácia e falha na identificação da classe de ataque, o que compromete sua aplicabilidade em tempo real.

Já Thirimanne et al. [13] propuseram uma rede neural profunda composta por 16 camadas densas, com excelente desempenho em acurácia e F1-score. No entanto, a complexidade da arquitetura impactou negativamente no tempo de inferência, tornando-a menos adequada para aplicações com restrição de tempo.

O estudo de Chiroma et al. [5] revisa o uso de AGs para otimizar RNAs, destacando a aplicação em tarefas como seleção de atributos e definição de topologia. Já Boehm et al. [4] utilizaram AGs na classificação de sinais cerebrais, alcançando uma alta acurácia.

Este trabalho propõe a combinação de AGs e RNAs especificamente para a detecção de intrusões, utilizando o conjunto de dados CSE-CIC-IDS2018 para treinamento e teste. O objetivo é otimizar de maneira eficiente e automatizada o desempenho da rede neural, trazendo uma abordagem inovadora para a segurança em redes.

III. Fundamentação Teórica

A. Redes Neurais Artificiais

As Redes Neurais Artificiais (RNAs) são modelos computacionais inspirados no sistema nervoso humano. Compostas por camadas de neurônios interconectados, elas executam operações matemáticas sobre os dados de entrada, ajustando seus pesos por meio de algoritmos de otimização. Atualmente, são amplamente utilizadas em tarefas como diagnóstico médico, tradução de idiomas e detecção de intrusões.

Neste trabalho, é utilizada a arquitetura de Rede Neural Convolucional (CNN), que extrai características dos dados com filtros e reduz a dimensionalidade por *pooling*. Os vetores resultantes são processados por um Perceptron de Múltiplas Camadas (MLP). Essa combinação é eficaz na detecção de invasões por unir extração de padrões e boa capacidade de classificação.

Conforme ilustrado na Figura 1, a CNN é composta por três etapas principais: convolução (extração de características), *pooling* (redução dimensional com *max pooling*) e *flattening* (vetorização dos dados para entrada na RNA).

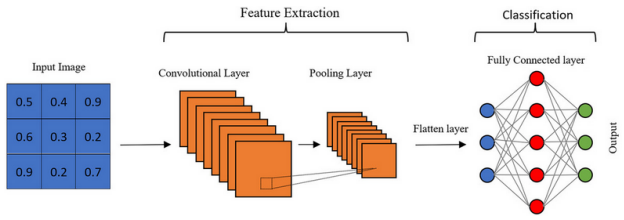


Fig. 1: Arquitetura da Rede Neural Convolucional.
Fonte: [3].

B. Algoritmos Genéticos

Os AGs geram populações de indivíduos por meio de operadores genéticos. O cruzamento e a mutação são aplicados aos indivíduos selecionados, os pais. A probabilidade de cruzamento (pc) de cada indivíduo ser selecionado para ser pai é baseada em sua função de aptidão, conhecida como *fitness* [14]. O operador de cruzamento combina os valores dos pais para gerar os valores dos filhos. O operador empregado neste trabalho é denominado Radcliff, em que, os pais p_1 e p_2 são dados por:

$$p_1 = [p_{m1}, p_{m2}, \ldots, p_{mN}], \tag{1}$$

$$p_2 = [p_{d1}, p_{d2}, \ldots, p_{dN}], \tag{2}$$

com N sendo o número de parâmetros, então os filhos são gerados pelas seguintes equações:

$$filho1 = \beta p_{mn} + (1-\beta)p_{dn}, \tag{3}$$

$$filho2 = (1-\beta)p_{mn} + \beta p_{dn}. \tag{4}$$

em que β é um escalar definido no intervalo [0,1]; neste trabalho, foi fixado em 0,3.

Este trabalho utilizou o método da roleta para a seleção dos indivíduos que realizam o cruzamento em cada geração. Nesta técnica, cada indivíduo da população é representado na roleta proporcionalmente ao seu valor gerado pela função de aptidão. Em outras palavras, indivíduos com maior aptidão têm mais chances de serem escolhidos por ocuparem um espaço maior na roleta.

Ademais, a probabilidade de mutação (pm) indica a chance de indivíduos terem suas características alteradas aleatoriamente. Neste trabalho, empregou-se a técnica de elitismo, em que o melhor indivíduo de cada geração é sempre preservada para a próxima, de modo a não haver retrocessos na evolução.

IV. Metodologia

A. Banco de dados de teste

O banco de dados utilizado para treinar e testar as RNAs desenvolvidas neste trabalho é o CIC-IDS2018 [11]. Ele contém tráfego normal e de ataques, capturado pelo CICFlowMeter, que gera 80 atributos em formato CSV ou PCAP, dos quais 78 foram usados. Foram selecionados os ataques DoS Slowloris, GoldenEye, SlowHTTPTest e Hulk por representarem diferentes estratégias de negação de serviço, o que permite ao modelo aprender a identificar variações complexas e ampliar sua capacidade de detecção.

A quantidade de tráfego foi reduzida em 98% com o objetivo de diminuir o tempo de treinamento das RNAs e acelerar a evolução da população dos AGs. Essa redução preservou a proporção entre as classes de tráfego normal e de ataque, sendo que o número de registros de tráfego de ataque foi de 13.086 e de tráfego normal foi de 28.696. Além disso, 70% dos dados foram destinados ao treinamento, enquanto os 30% restantes foram reservados para o teste. Estudos prévios, como os de Nguyen et al. [10] e Lopez-Martin et al. [9], indicam que conjuntos de dados reduzidos e balanceados, quando bem representativos, podem manter o desempenho preditivo de modelos de aprendizado de máquina, ao mesmo tempo em que reduzem significativamente o custo computacional. O banco de dados usado pode ser visualizado em Dataset.

B. Algoritmo Genético aplicado a Rede Neural Artificial

Neste trabalho foram construídos dois AGs, um para definição dos atributos de entrada da RNA e outro para a escolha dos hiperparâmetros. Ambos os AGs foram construídos usando populações de 20 indivíduos e 50 gerações, de modo que os indivíduos mais aptos são aqueles que proporcionaram maior acurácia durante seu treinamento. A probabilidade de cruzamento e de mutação em ambos AG é de 90% e 30%, respectivamente. O treinamento de cada indivíduo é feito em 100 épocas, porém, estabeleceu-se que quando o valor da função de custo não evoluir em pelo menos 10^{-4} em 10 épocas seguidas, o treinamento é interrompido.

A CNN empregada neste trabalho contém a seguinte estrutura:

- Camada de convolução com 32 mapas de características, com matrizes de entrada 4x5 (4 linhas e 5 colunas) e função de ativação definida por um gene do tipo *ativacao*.
- *Max Pooling* de tamanho 2x2.
- *Flatten*.

- 3 camadas densas com funções de ativação e número de nós de cada camada com definições feitas, respectivamente, por genes do tipo *ativacao* e *numero_de_nos*.

Primeiramente, utiliza-se o AG para selecionar os 20 melhores atributos de entrada, e em seguida, o AG é empregado para encontrar os hiperparâmetros ideais da CNN. Como a captura de tráfego forneceu 78 atributos, o AG responsável pela definição dos atributos de entrada contém indivíduos com 20 genes, sendo que cada gene pode assumir um valor de 1 a 78.

No caso dos AGs utilizados para a definição dos hiperparâmetros, a CNN utiliza os genes relativos ao otimizador e à função de perda durante seu treinamento. Portanto, é necessário que, ao considerar também os genes da estrutura da rede, o AG contenha indivíduos compostos por 9 genes: 4 do tipo *ativacao*, 3 do tipo *numero_de_nos*, 1 do tipo *otimizador* e 1 do tipo *loss*. Dessa forma, os indivíduos possuem genes divididos em 4 tipos:

1) *numero_de_nos*
   a) Indica a quantidade de nós em cada camada.
   b) Contém 8 alelos, isto é, 8 possíveis quantidades de nós, variando no intervalo de 9 a 16.
2) *ativacao*
   a) Indica qual a função de ativação usada.
   b) Possui 8 alelos, cada um correspondendo a uma das seguintes funções: linear, ReLU, SELU, ELU, softmax, softplus, Mish e Swish.
3) *otimizador*
   a) Indica o otimizador empregado no treinamento.
   b) É composto por 3 alelos, os quais representam os seguintes otimizadores: SGD, Adam e Adamax.
4) *loss*
   a) Indica a função de custo empregada no treinamento.
   b) Contém 3 alelos, cada um associado a uma das funções de custo: binary crossentropy, Poisson e KLDivergence.

V. Resultados

O gráfico da Figura 2 mostra a maior acurácia entre os indivíduos de cada geração do AG utilizado para determinar os atributos de entrada. Nota-se que a maior acurácia alcançada foi de 99,05%. A Figura 3 apresenta a matriz de confusão, que indica que a RNA conseguiu classificar corretamente a maioria das amostras em suas respectivas categorias.

Os atributos de entrada do indivíduo com maior acurácia são: *Fwd Pkts/s*, *Bwd PSH Flags*, *Bwd Header Len*, *ACK Flag Cnt*, *Fwd Header Len*, *Fwd IAT Mean*, *ECE Flag Cnt*, *Fwd IAT Max*, *Fwd Seg Size Min*, *Bwd IAT Max*, *Fwd PSH Flags*, *Idle Mean*, *Fwd Seg Size Avg*, *URG Flag Cnt*, *Bwd Pkt Len Std*, *Fwd Pkt Len Min*, *Flow IAT Std*, *RST Flag Cnt*, *Down/Up Ratio*, *Flow Byts/s*.

A Figura 4 apresenta o gráfico da maior acurácia entre os indivíduos de cada geração do AG aplicado à escolha dos hiperparâmetros da CNN. A Figura 5 mostra a matriz de confusão da CNN com maior desempenho, na qual se observa a ocorrência de poucos falsos positivos e negativos, evidenciando boa capacidade de generalização e alta precisão. O AG identificou configurações que proporcionaram à CNN com acurácia de 99,78%.

Fig. 2: Gráfico da maior acurácia entre os indivíduos em cada geração para o AG de atributos de entrada.

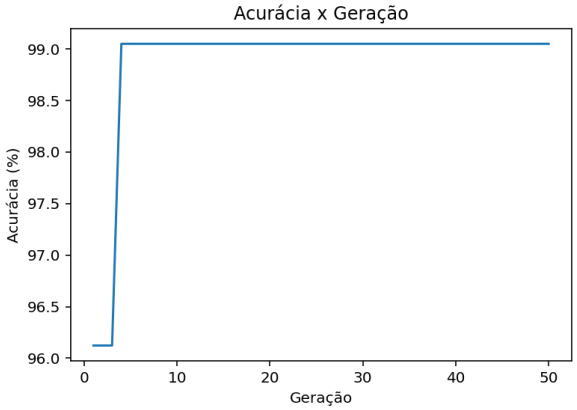


Fig. 3: Matriz de confusão para o AG de atributos de entrada.

	0	1
0	8478	104
1	15	3937

Fig. 4: Gráfico da maior acurácia entre os indivíduos em cada geração para o AG de hiperparâmetros.

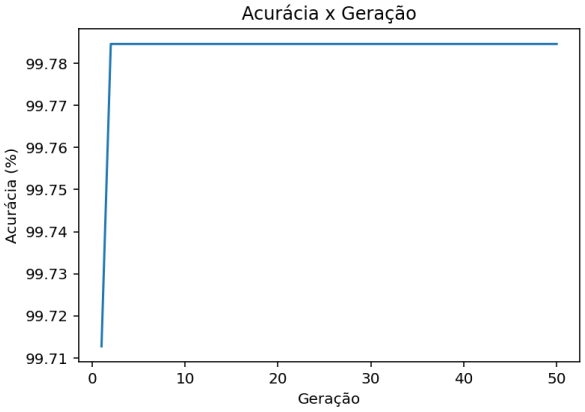


Fig. 5: Matriz de confusão do AG de hiperparâmetros.

	0	1
0	8562	20
1	7	3945

Os hiperparâmetros do indivíduo com maior acurácia do AG são:

- i) Camada de convolução: ativação = *mish*
- ii) Primeira camada densa: *número_de_nós* = 12, ativação = *mish*

- iii) Segunda camada densa: *número_de_nós* = 10, ativação = *selu*
- iv) Terceira camada densa: *número_de_nós* = 15, ativação = *elu*
- v) Função de perda: *binary crossentropy*
- vi) Otimizador: *adam* (*learning rate* = 0,001)

Pontua-se que, nos experimentos realizados com o AG de atributos de entrada, observou-se que o indivíduo com a maior acurácia foi identificado na 4ª geração, com a acurácia permanecendo estável nas gerações subsequentes, conforme ilustrado na Figura 2. De maneira análoga, no AG de hiperparâmetros, a estabilização ocorreu na 2ª geração, quando o indivíduo com a maior acurácia foi encontrado e a acurácia não apresentou mais aumentos, conforme demonstrado na Figura 4. Esses resultados indicam que o AG foi eficaz na identificação de soluções ótimas de forma relativamente rápida, com o modelo de hiperparâmetros alcançando a convergência ainda mais precocemente. Tais observações sugerem que a busca pelos melhores conjuntos de atributos e hiperparâmetros foi eficiente, permitindo uma exploração satisfatória do espaço de soluções e a identificação de ótimos locais de forma eficaz.

Com o objetivo de avaliar a eficácia do modelo proposto em relação a abordagens consolidadas na literatura, realizou-se uma comparação direta com dois modelos amplamente utilizados: o modelo BI-LSTM apresentado por Shieh et al. [12] e o modelo DNN descrito por Thirimanne et al. [13]. Para garantir a equidade na comparação, todos os modelos foram treinados utilizando a mesma base de dados e configurações experimentais: 100 épocas e *batch_size* fixado em 50. Além disso, os atributos de entrada para todas as redes corresponderam aos selecionados pela AG como os mais relevantes. Essa padronização assegura uma análise justa e controlada do desempenho entre as diferentes arquiteturas propostas.

A Tabela I apresenta os principais resultados obtidos por cada modelo, incluindo a acurácia, o F1-score ponderado e o tempo de previsão sobre o conjunto de teste. Esses indicadores permitem uma análise quantitativa do desempenho dos modelos, tanto em termos de eficácia quanto de eficiência computacional.

TABELA I: Comparação entre o modelo proposto e trabalhos relacionados

Modelo	**Acurácia**	**F1-score (ponderado)**	**Tempo (s)**
Shieh et al.	68,47%	56,00%	1,7534
Thirimanne et al.	99,74%	100,00%	1,3583
Proposto	**99,78%**	**100,00%**	**0,6627**

As matrizes de confusão apresentadas nas Figuras 6, 7 e 5 permitem uma análise comparativa entre os três modelos avaliados. O modelo de Shieh et al. (Figura 6) demonstrou incapacidade de detectar a classe de ataques, classificando corretamente apenas o tráfego benigno. O modelo de Thirimanne et al. (Figura 7) obteve excelente desempenho, com poucos falsos positivos e negativos. Já o modelo proposto (Figura 5) apresentou a melhor performance, com o menor número de erros e superior capacidade de generalização.

A fim de compreender as particularidades estruturais de cada abordagem e justificar os desempenhos observados,

Fig. 6: Matriz de confusão do modelo BI-LSTM proposto por Shieh et al. [12].

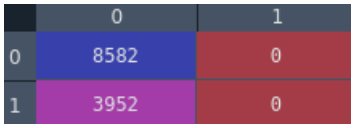
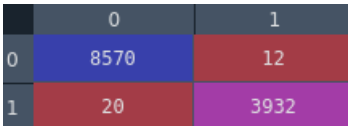


Fig. 7: Matriz de confusão do modelo DNN proposto por Thirimanne et al. [13].



apresenta-se a seguir um detalhamento das arquiteturas implementadas. Essa análise permite observar diferenças fundamentais entre os modelos, como o número de camadas, funções de ativação utilizadas, estratégias de regularização, tipo de otimizador e complexidade geral da arquitetura — fatores que influenciam diretamente tanto a capacidade de generalização quanto o tempo de processamento.

Modelo de Shieh et al. [12]

- i) Primeira camada BI-LSTM: 64 unidades, ativação padrão (tanh), *return_sequences=True*
- ii) Dropout: taxa = 0,3
- iii) Segunda camada BI-LSTM: 32 unidades, ativação padrão (tanh)
- iv) Dropout: taxa = 0,3
- v) Camada de saída: 1 nó, ativação = sigmoid
- vi) Otimizador: SGD (learning rate = 0,00859, momentum = 0,89, decay = 10^{-3}, clipnorm = 0,9)
- vii) Loss: binary crossentropy

Modelo de Thirimanne et al. [13]

- i) Camada densa: número_de_nós = 64, ativação = relu
- ii) Camada densa: número_de_nós = 160, ativação = relu
- iii) Camada densa: número_de_nós = 352, ativação = relu
- iv) Camada densa: número_de_nós = 320, ativação = relu
- v) Camada densa: número_de_nós = 448, ativação = relu
- vi) Camada densa: número_de_nós = 384, ativação = relu
- vii) Camada densa: número_de_nós = 192, ativação = relu
- viii) Camada densa: número_de_nós = 224, ativação = relu
- ix) Oito camadas adicionais com 32 nós cada, ativação = relu
- x) Camada de saída: 1 nó, ativação = sigmoid
- xi) Otimizador: SGD (*learning rate* = 0,001, momentum = 0)
- xii) Loss: binary crossentropy

A análise comparativa entre o modelo proposto nesta pesquisa e os trabalhos relacionados evidencia a superioridade da solução desenvolvida para os ataques e base de dados considerados. Foram utilizadas as mesmas condições de treinamento

em todos os experimentos, com 100 épocas e *batch_size* fixo de 50, assegurando uma base equitativa para comparação. O modelo proposto alcançou a maior acurácia, de 99,78%, bem como o maior F1-score ponderado, de 100,00%, desempenho este equivalente ao do modelo de Thirimanne et al. Além disso, obteve o menor tempo de previsão, registrando apenas 0,6627 segundos. Esses resultados demonstram não apenas a elevada capacidade de predição do modelo, mas também sua superioridade em termos de eficiência computacional, fator essencial em sistemas de detecção de intrusão em tempo real.

O modelo de Shieh et al. [12], embora baseado em uma arquitetura bidirecional com uso de regularização L2 e camadas de *dropout*, apresentou desempenho consideravelmente inferior. A acurácia observada foi de apenas 68,47%, com F1-score ponderado de 56,00%. A matriz de confusão evidenciou uma falha completa na identificação da classe positiva, já que o modelo não foi capaz de classificar corretamente nenhum dos exemplos de ataque, resultando em um valor de *recall* igual a zero. Além disso, foi o modelo que apresentou o maior tempo de inferência, com duração de 1,7534 segundos, comprometendo sua viabilidade em aplicações que demandam respostas rápidas.

O modelo proposto por Thirimanne et al. [13], por outro lado, apresentou resultados expressivos, com acurácia de 99,74% e F1-score ponderado de 100,00%. Entretanto, o tempo de previsão registrado foi de 1,3583 segundos, significativamente superior ao do modelo desenvolvido nesta pesquisa. Esse maior tempo de inferência pode ser atribuído à elevada complexidade da arquitetura utilizada, composta por 16 camadas densas. Embora essa profundidade contribua para a excelente capacidade de classificação, o que, por sua vez, acarreta um custo computacional significativo, o que pode representar um entrave à sua adoção em sistemas que exigem alto desempenho em tempo real.

Já o modelo proposto nesta pesquisa combina de forma eficiente uma camada convolucional com funções de ativação modernas (*mish* e *elu*) e uma arquitetura enxuta de três camadas densas, além do uso do otimizador Adam. Como resultado, o modelo obteve desempenho superior ou equivalente em termos de acurácia e F1-score, e foi significativamente mais rápido, sendo ideal para aplicações sensíveis ao tempo.

Dessa forma, os dados demonstram que o modelo desenvolvido nesta pesquisa é uma solução altamente eficaz e superior para detecção de ataques DDoS, superando os trabalhos relacionados tanto na qualidade das predições quanto na velocidade de execução.

Adicionalmente, uma das principais vantagens do uso de AGs é sua eficiência em espaços de alta dimensionalidade. Neste trabalho, existem $\binom{78}{20} \approx 1{,}08 \times 10^{19}$ combinações possíveis de atributos de entrada. Para os hiperparâmetros da RNA (4 ativações com 8 opções, 3 camadas com 8 tamanhos, 3 otimizadores e 3 funções de perda), temos:

$$(8^4)(8^3)(3)(3) = 4096 \times 512 \times 9 = 18{,}874{,}368 \qquad (5)$$

O total estimado de combinações é:

$$\approx 1{,}08 \times 10^{19} \times 1{,}88 \times 10^{7} \approx 2{,}03 \times 10^{26} \qquad (6)$$

Enquanto a busca exaustiva seria inviável, o AG avaliou cerca de 1000 combinações de atributos e 1000 de hiperparâmetros, totalizando 2000 configurações, menos de 10^{-23} do espaço total, evidenciando a eficiência da abordagem com baixo custo computacional.

VI. Conclusão

Este trabalho teve como objetivo explorar o uso de AG para otimizar os hiperparâmetros e os atributos de entrada de CNNs na detecção de intrusões. Através dessa abordagem, obteve-se uma acurácia de 99,78% na detecção de ataques do tipo DoS. Os resultados demonstram a eficácia dos AGs em conjunto com as CNNs para aprimorar o desempenho da detecção de intrusões, ressaltando a relevância dessa combinação no contexto da segurança cibernética.

Como perspectivas para trabalhos futuros, sugere-se a investigação de outros modelos de RNAs além das CNNs, para verificar a aplicabilidade e desempenho de diferentes arquiteturas. Também seria interessante expandir o conjunto de ataques considerados, incorporando uma maior diversidade de tipos de intrusões. Além disso, uma abordagem inovadora seria incluir ataques direcionados a redes 5G, ampliando assim o escopo do banco de dados para contemplar essa nova geração de redes e suas vulnerabilidades específicas.

Referências

[1] Teresa F. Lunt, "A survey of intrusion detection techniques," *Computers & Security*, vol. 12, no. 4, pp. 405–418, 1993.

[2] Stefan Axelsson, "Intrusion detection systems: A survey and taxonomy," *Technical Report*, Department of Computer Engineering, Chalmers University of Technology, 2000.

[3] Alsaleh, A. and Perkgoz, C. (2023). A space and time efficient convolutional neural network for age group estimation from facial images. *PeerJ Computer Science*, 9:e1395.

[4] Boehm, O., Hardoon, D. R., and Manevitz, L. M. (2011). Classifying cognitive states of brain activity via one-class neural networks with feature selection by genetic algorithms. *International Journal of Machine Learning and Cybernetics*, 2:125–134.

[5] Chiroma, H., Abdulkareem, S., Abubakar, A., and Herawan, T. (2017). Neural networks optimization through genetic algorithm searches: a review. *Applied Mathematics & Information Sciences*, 11(6):1543–1564.

[6] Drewek-Ossowicka, A., Pietrołaj, M., and Rumiński, J. (2021). A survey of neural networks usage for intrusion detection systems. *Journal of Ambient Intelligence and Humanized Computing*, 12:497–514.

[7] Gonçalves, H. E. (2023). Comparação da eficiência de modelos de redes neurais artificiais na detecção de intrusões em redes de computadores. *Trabalho de Conclusão de Curso, Universidade Federal de Uberlândia*.

[8] Kramer, O. (2017). *Genetic Algorithms*. Springer.

[9] Lopez-Martin, M., Carro, B., Sanchez-Esguevillas, A., and Lloret, J. (2017). Network traffic classifier with convolutional and recurrent neural networks for internet of things. *IEEE Access*, 5:18042–18050.

[10] Nguyen, T. T. A., Hoang, D. T., Nguyen, D. T., Niyato, D., Long, V., and Hossain, E. (2020). Evaluation of deep learning models for intrusion detection in cyber-physical systems. *IEEE Systems Journal*, 14(2):2916–2925.

[11] Sharafaldin, I., Lashkari, A. H., and Ghorbani, A. A. (2018). Toward generating a new intrusion detection dataset and intrusion traffic characterization. In *Proceedings of the 4th International Conference on Information Systems Security and Privacy (ICISSP)*, pages 108–116.

[12] Shieh, C.-S., Lin, W.-W., Nguyen, T.-T., Chen, C.-H., Horng, M.-F., and Miu, D. (2021). Detection of unknown DDoS attacks with deep learning and Gaussian mixture model. *Applied Sciences*, 11(11):5213.

[13] Thirimanne, S. P., Jayawardana, L., Yasakethu, L., Liyanaarachchi, P., and Hewage, C. (2022). Deep neural network based real-time intrusion detection system. *SN Computer Science*, 3(2):145.

[14] Sivanandam, S. N. and Deepa, S. N. (2008). *Introduction to Genetic Algorithms*. Springer.