Strategies for guiding an evolutionary wrapper for attack classification.

Javier Maldonado
Departamento de Ingeniería Informática
Universidad Técnica Federico Santa María
Valparaíso, Chile
javier.maldonadoc@usm.cl

María Cristina Riff
Departamento de Ingeniería Informática
Universidad Técnica Federico Santa María
Valparaíso, Chile
maria-cristina.riff@inf.utfsm.cl

***Resumen*—The attack detection is a relevant problem in cybersecurity. The intrusion detection systems analyze network data streams, trying to detect, classify and alert intrusion attempts. This work, shows the configuration flexibility of a wrapper evolutionary algorithm for feature selection, to improve attack detection and classification in the field of cybersecurity. The approach is evaluated using the NSL-KDD dataset and the CART algorithm as the classification method. The results show that the algorithm is capable of achieving different classification objectives, by exchanging the proposed metrics in its components, maintaining the design of the original evolutionary wrapper algorithm.**

***Index Terms*—attack detection, intrusion detection system, evolutionary algorithm, feature selection, decision tree**

I. Introducción

Los sistemas de detección de intrusos (Intrusion Detection Systems (IDS), por sus siglas en inglés), representan uno de los elementos fundamentales en ciberseguridad, para evitar accesos no autorizados a infraestructuras de tecnología, clasificando y alertando intentos de intrusión. Todo esto con el fin de mantener la disponibilidad, integridad y confidencialidad de los sistemas [1]. La gran diversidad y constante evolución de los ataques, hace que la detección sea aún mas difícil [2]. El hecho que los ataques, usualmente, no tienen una distribución uniforme en el tiempo, incrementa aún mas la dificultad de su detección. En este trabajo se usa un algoritmo evolutivo, presentado en [3], diseñado para la selección de características, para ayudar a un clasificador en el contexto de detección de ataques. Manteniendo el diseño original, esta investigación realiza un análisis de cambio de métricas, que guían el algoritmo evolutivo, y su impacto en el comportamiento de un nuevo clasificador. Se evalúa la sensibilidad de los cambios en las métricas del algoritmo utilizando CART [4] como técnica de clasificación y evaluado con el dataset NSL-KDD [5].

II. Algoritmo evolutivo

A continuación se describe cada componente del algoritmo evolutivo, presentado originalmente en [3], el cual utiliza C4.5 como algoritmo de clasificación.

978-1-7281-8328-2/20/$31.00 ©2020 IEEE

II-1. Representación: Se define como un arreglo binario I, de tamaño $1 \times n$, donde n es el número de características. Indica si se selecciona (1) o no (0) una característica en particular para la construcción del árbol de decisión.

II-2. Población inicial: Se genera aleatoriamente. También se incluye un individuo que utiliza todas las características, con el fin de tener una cota de evaluación y todas las características presentes en la población al menos una vez.

II-3. Función de evaluación: Es el criterio de calidad de los individuos, según la métrica configurada.

II-4. Operador Bit-flip: Este es un operador asexual propuesto en [6]. Selecciona un individuo y hace el cambio en los genes según la probabilidad $Pflip$. Está diseñado con el fin de explorar el espacio de búsqueda.

II-5. Operador Swap-M: Este operador asexual, genera un individuo a partir de otro seleccionado de acuerdo a una probabilidad $Pswap$. Está enfocado en mejorar la calidad de clasificación de un individuo que, con la misma cantidad de características, produzca una mejor clasificación. El operador intercambia dos posiciones del individuo i tal que $I_{ip} \neq I_{iq}$ con $p \neq q$, el cual se acepta si el nuevo individuo es mejor que el actual, de acuerdo al valor de una métrica.

II-6. Operador Cross-charact: Utiliza dos individuos para generar otro y se ejecuta según la probabilidad $Pcross$. Su objetivo es heredar características de cada padre de acuerdo a la siguiente función de proporción:

II-6a. Proporción: Dado un individuo I_j con k valores distintos de cero. Se define la función de proporción como

$$RF(I_j) = \frac{Métrica(I_j)}{k} \quad \forall j = 1, ..., Popsize \qquad (1)$$

Donde $Métrica(I_j)$ es el criterio de evaluación respecto a la calidad del individuo. La idea principal de la proporción, es que el algoritmo prefiera árboles con mejor clasificación y menor cantidad de características.

II-A. Métricas

A continuación se describen las métricas usadas en esta investigación: average recall (AvRecall), average precision (AvPr) y false positive rate (FPR). En adelante se denota: TP_{i_j} como la cantidad de elementos correctamente clasificados en la clase i del individuo I_j. FN_{i_j} son los eventos incorrectamente

clasificados de la clase i en el individuo I_j. FP_{i_j} son los falsos positivos en las predicciones de la clase i en el individuo I_j, y C es la cantidad de clases de ataques.

II-A1. Average Recall (AvRecall): Se define $Recall_{i_j}$ como,

$$Recall_{i_j} = \frac{TP_{i_j}}{(TP_{i_j} + FN_{i_j})} \quad \forall i = 1, ..., C \tag{2}$$

El $AvRecall$ del individuo I_j se calcula:

$$AvRecall(I_j) = \frac{\sum_{i=1}^{C} Recall_{i_j}}{C} \quad \forall j = 1, ..., Popsize \tag{3}$$

Cuanto mas alto sea el valor de $AvPr$, mejor será la calidad de clasificación.

II-A2. Average Precision (AvPr): Se define $Precision_{i_j}$ como:

$$Precision_{i_j} = \frac{TP_{i_j}}{(TP_{i_j} + FP_{i_j})} \quad \forall i = 1, ..., C \tag{4}$$

El $AvPr$ del individuo I_j se calcula como:

$$AvPr(I_j) = \frac{\sum Precision_{i_j}}{C} \quad \forall i = 1, ..., C \tag{5}$$

Cuanto mas alto sea el valor de $AvPr$, mejor será la predicción.

II-A3. False positive rate (FPR): Corresponde a la tasa de falsos positivos de un individuo I_j (FPR):

$$FPR(I_j) = \frac{FP_{I_j}}{(FP_{I_j} + TN_{I_j})} \quad \forall j = 1, ..., Popsize \tag{6}$$

Donde $FPR(I_j)$ es la tasa de falsos positivos del individuo I_j y FP_{I_j} son los eventos normales clasificados como ataque en de I_j. Y TN_{I_j} es el tráfico normal correctamente clasificado de I_j [7]. Un valor menor de FPR, indica una mejor calidad de clasificación.

III. Diseño experimental

Se utilizó el algoritmo propuesto en [3], usando la misma configuración de parámetros: probabilidad de cruce 0.8, probabilidad de mutación 0.1 y probabilidad de swap 0.7. Para la evaluación de esta propuesta, se seleccionó el dataset NSL-KDD [5], por su amplio uso en el área y CART [4] como algoritmo de clasificación, por su rapidez y facilidad de interpretación. Se realizaron 20 ejecuciones con cada combinación de métricas en cada uno de los operadores, generando 27 configuraciones posibles, con un total de 540 experimentos.

Se realizó un análisis comparativo agrupado por AvRecall, AvPr y FPR, identificados en las configuraciones con 0, 1 y 3 respectivamente. Se intercambiaron las métricas propuestas entre la función de evaluación, el operador swap-m y la función de cross-charact, las cuales se configuraron con las métricas indicadas es dicho orden, mostrando tres dígitos para cada configuración probada.

Para comparar los resultados, se construyeron boxplot y se utiliza la prueba no paramétrica de Friedman, para k muestras pareadas. Se incluye dentro de la comparativa los resultados de CART sin selección de características.

IV. Resultados

A continuación, se muestran las configuraciones probadas del algoritmo. En todas las comparativas se incluyen los resultados obtenidos por CART sin la selección de características. En el Cuadro I se muestran los 10 mejores resultados del test de Friedman para cada uno de las métricas evaluadas.

Cuadro I
Test de Friedman

AvRecall		AvPr		FPR	
Config	Rank	Config	Rank	Config	Rank
011	25.25	103	24.75	301	3.80
001	25.08	100	23.88	311	4.18
031	23.95	111	23.83	331	4.58
000	23.63	110	23.25	330	5.08
030	23.05	113	23.23	303	5.13
003	22.85	131	23.18	333	5.45
033	22.83	101	22.98	310	5.58
010	22.73	130	22.83	300	5.60
013	22.70	133	21.55	313	5.88
130	16.23	030	15.10	CART	14.70

Según el Cuadro I, la mejor configuración respecto a AvRecall, es 011 (AvRecall = 55.92), la 000 aparece en cuarto lugar. Esto evidencia que el algoritmo obtiene mejores resultados con AvRecall en la función de evaluación pero con la colaboración de AvPr en los operadores de swap-m y cross-charact, los cuales ayudan indirectamente al AvRecall, mediante la mejora del precision de las clases de ataques. Los resultados de CART aparecen en el puesto 19. Respecto a la métrica AvPr, la combinación 103 (AvPr = 93.78) es la mejor, involucrando tres métricas: AvPr en la función de evaluación, AvRecall en el swap y FPR en el crossover. CART se ubica en la posición 18 y la configuración 111 se encuentra en tercer lugar, lo que evidencia que la mejora del AvPr requiere que el algoritmo evolutivo tome información de FPR y AvRecall. El mejor resultado de FPR se obtiene con la configuración 301 (FPR = 1.38), lo que combina las métricas AvRecall en el swap-m y AvPr en el cross-charact, estas dos últimas métricas se enfocan en la calidad de la clasificación desde dos puntos de vista: recall y precision, en consecuencia mejoran la calidad del FPR. Por otro lado, el CART sin selección de características aparece en el puesto 10, lo que indica que las configuraciones 0XX y 1XX perjudican la calidad de clasificación del propio clasificador CART. Esto demuestra que el CART es sensible a datos desbalanceados, es decir tiene mejores resultados con las clases mayoritarias.

En la Figura 1 se presentan los resultados respecto a la métrica AvRecall. Se pueden observar 3 grupos diferenciados por la función de evaluación, donde los mejores resultados se obtienen con las configuraciones 0XX. Nótese que las configuraciones 3XX tienen un impacto negativo en el AvRecall, obteniendo peores resultados que el CART sin selección de

características, se muestra una sensibilidad de CART a los datos desbalanceados.

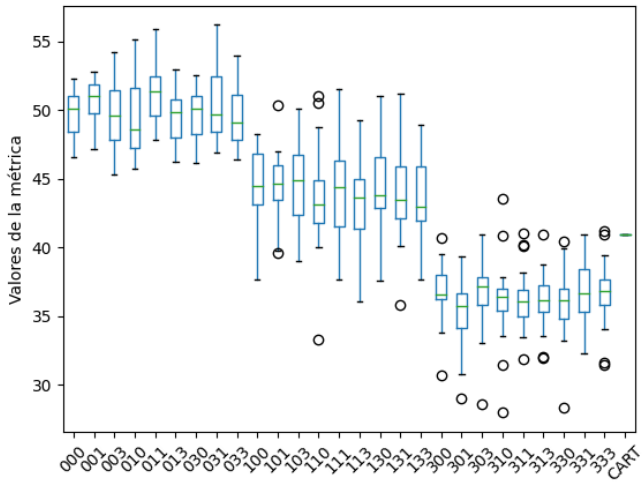

Figura 1. Configuraciones de CART respecto a Av Attack recall

Respecto a la métrica AvPr, se muestran los resultados obtenidos en la Figura 2. Se diferencian 3 grupos respecto a la función de evaluación, donde los mejores resultados se observan con las configuraciones 1XX, produciendo mejores resultados que el CART sin selección de características. Por otro lado, los peores son obtenidos por las configuraciones 3XX, donde se enfoca a mejorar el FPR, desmejorando notablemente la calidad de clasificación de los ataques respecto a la precisión de las clases.

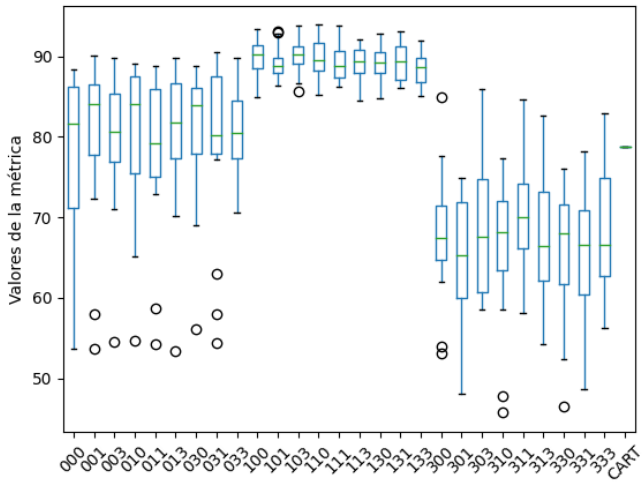

Figura 2. Configuraciones de CART respecto a Av Precision

En la Figura 3 se muestran los boxplot asociados a la métrica FPR. Se puede observar que el grupo de configuraciones 3XX obtiene los mejores resultados. Se evidencia que el algoritmo se enfoca en mejorar el FPR cuando se configura en su función y obtiene mejores resultados que el CART sin selección de características.

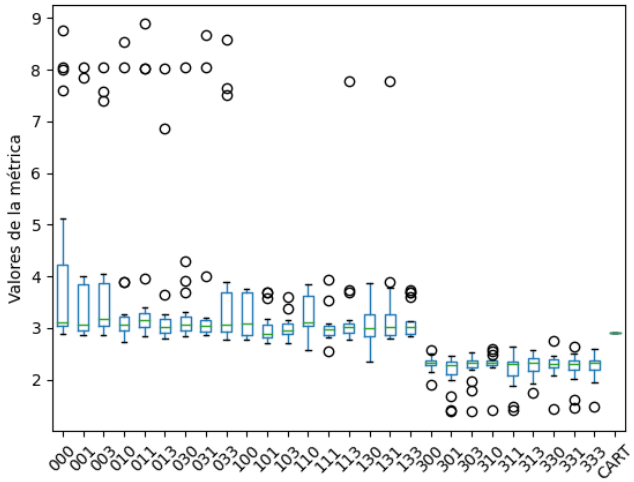

Figura 3. Configuraciones de CART respecto a FPR

V. Trabajos relacionados

En la literatura se pueden encontrar varios trabajos aplicados a la selección de características usando un clasificador para evaluar un subconjunto de características. Algunos algoritmos evolutivos [8]–[10]. Otros tomando en cuenta algoritmos de enjambres: Artificial Bee Colony (ABC) [11], Particle Swarm Optimization (PSO) [12] y Firefly algorithm (FFA) [13]. En cuanto a los algoritmos de clasificación se pueden encontrar diversas técnicas como Support Vector Machines (SVM) [14], [15] y Neural Network (NN) [12], [16], así como también numerosos trabajos basados en árboles de decisión, tales como ID3 [17], C4.5 [3], [10], [13] y el ensamblado Random Forest (RF) [10], [18], [19].

VI. Conclusión

En este trabajo, se ha probado la flexibilidad de configuración de un algoritmo evolutivo para la selección de características, aplicado en el área de detección de intrusos. Se utilizó el mismo algoritmo evolutivo y configuración de parámetros presentada en [3] con el algoritmo de clasificación CART y se aplicaron diversas métricas en sus operadores, esto con el fin de analizar el impacto en la calidad de clasificación. Con el algoritmo de clasificación y el intercambio de las métricas usadas entre sus distintos operadores y función de evaluación, se pudo evidenciar la flexibilidad de éste diseño, el cual puede cambiar su comportamiento sin cambios estructurales dentro del algoritmo. Este diseño proporciona facilidad de adaptación a los objetivos y necesidades de análisis requeridos, por ejemplo, por un analista de seguridad, el cuál le permite enfocarse en algún comportamiento que desee estudiar.

REFERENCIAS

[1] F. H. Botes, L. Leenen, and R. De La Harpe, "Ant colony induced decision trees for intrusion detection," in *European Conference on Information Warfare and Security, ECCWS*, no. June. Elsevier Scopus, 2017, pp. 53–62.

[2] S. W. Lin, K. C. Ying, C. Y. Lee, and Z. J. Lee, "An intelligent algorithm with feature selection and decision rules applied to anomaly intrusion detection," *Applied Soft Computing Journal*, vol. 12, no. 10, pp. 3285–3290, 2012.

[3] J. Maldonado, M.-C. Riff, and E. Montero, "Improving Attack Detection of C4.5 using an Evolutionary Algorithm," in *2019 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, jun 2019, pp. 2229–2235.

[4] L. Breiman, J. Friedman, R. Olshen, and C. Stone, "Classification and regression trees. statistics/probability series," 1984.

[5] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," in *IEEE Symposium on Computational Intelligence for Security and Defense Applications, CISDA 2009*, no. Cisda. IEEE, 2009, pp. 1–6.

[6] L. Davis, *Handbook of genetic algorithms*. CUMINCAD, 1991.

[7] M. R. Gauthama Raman, N. Somu, K. Kirthivasan, R. Liscano, and V. S. Shankar Sriram, "An efficient intrusion detection system based on hypergraph - Genetic algorithm for parameter optimization and feature selection in support vector machine," *Knowledge-Based Systems*, vol. 134, pp. 1–12, oct 2017.

[8] H. Gharaee, M. Fekri, and H. Hosseinvand, "Intrusion Detection System Using SVM as Classifier and GA for Optimizing Feature Vectors," *International Journal of Information & Communication Technology Research (IJICTR)*, vol. 10, no. 1, pp. 26–35, 2018.

[9] F. H. Almasoudy, W. L. Al-yaseen, and A. K. Idrees, "ScienceDirect ScienceDirect Differential Evolution Wrapper Feature Selection for Intrusion Differential Evolution Wrapper Feature Selection for Intrusion Detection System Detection System," *Procedia Computer Science*, vol. 167, no. 2019, pp. 1230–1239, 2020.

[10] C. Khammassi and S. Krichen, "A NSGA2-LR wrapper approach for feature selection in network intrusion detection," *Computer Networks*, vol. 172, no. November 2019, p. 107183, 2020.

[11] M. Mazini, B. Shirazi, and I. Mahdavi, "Anomaly network-based intrusion detection system using a reliable hybrid artificial bee colony and AdaBoost algorithms," *Journal of King Saud University - Computer and Information Sciences*, vol. 31, no. 4, pp. 541–553, 2018.

[12] H. Li, W. Guo, G. Wu, and Y. Li, "A RF-PSO based hybrid feature selection model in intrusion detection system," *Proceedings - 2018 IEEE 3rd International Conference on Data Science in Cyberspace, DSC 2018*, pp. 795–802, 2018.

[13] B. Selvakumar and K. Muneeswaran, "Firefly algorithm based feature selection for network intrusion detection," *Computers and Security*, vol. 81, pp. 148–155, 2019.

[14] H. Polat and O. Polat, "Detecting DDoS Attacks in Software-Defined Networks Through Feature Selection Methods and Machine Learning Models," *Sustainability*, vol. 3, no. 12, p. 16, 2020.

[15] R. Vijayanand and D. Devaraj, "A Novel Feature Selection Method Using Whale Optimization Algorithm and Genetic Operators for Intrusion Detection System in Wireless Mesh Network," *IEEE Access*, vol. 8, pp. 56 847–56 854, 2020.

[16] S. Latha, "HPFSM - A High Pertinent Feature Selection Mechanism for Intrusion Detection System," *International Journal of Pure and Applied Mathematics*, vol. 118, no. 9, pp. 77–83, 2018.

[17] S. Mohammadi, H. Mirvaziri, M. Ghazizadeh-Ahsaee, and H. Karimipour, "Cyber intrusion detection by combined feature selection algorithm," *Journal of Information Security and Applications*, vol. 44, pp. 80–88, 2019.

[18] S. Soheily-Khah, P. F. Marteau, and N. Bechet, "Intrusion detection in network systems through hybrid supervised and unsupervised machine learning process: A case study on the iscx dataset," in *Proceedings - 2018 1st International Conference on Data Intelligence and Security, ICDIS 2018*, 2018, pp. 219–226.

[19] J. Maldonado and M.-C. Riff, "Evaluating different metric configurations of an evolutionary wrapper for attack detection," in *31st International Conference on Tools with Artificial Intelligence ICTAI*, no. 1, 2019, pp. 1–5.