Facultad de Informática

Carrera de Ingeniería Informática

Propuesta de técnica de Inteligencia Artificial para la detección de anomalías en la red de datos de la Universidad de Cienfuegos.

Trabajo de diploma para optar por el título de Ingeniería en Informática.

Autor: Janny Hermosa Morell Díaz

Tutor: MSc. Alexis Gómez Domínguez.

Ing. Lissette Montero Herrera.

Ing. Jorge Luis Rivero Pérez.

Cienfuegos

Curso 2009-2010

“Año 52 de la Revolución”

PENSAMIENTO

Para conocer bien las cosas hay que conocer sus pormenores, y como éstos son casi infinitos, nuestro saber es siempre superficial e incompleto.

LA ROCHEFOUCAULD

DEDICATORIA

Dedico este trabajo a mis padres por su incansable labor, por secundarme en mis sueños y darlo todo por verlos hechos realidad, a ellos que son constantes guías de mi vida y a quienes debo todo lo que soy.

AGRADECIMIENTO

A mis tutores por ser incansables.

A Oscarito y Eduardo por brindarme todos sus conocimientos.

A Boris y Raúl por su paciencia y amistad.

A Iris, Lili, Yeidi, Yanelis y todas aquellas personas que de una forma u otra fueron partícipes de este trabajo.

A mi familia.

A la vida.

RESUMEN

El objetivo fundamental de esta investigación está encaminado en la propuesta de técnicas de Inteligencia Artificial para la detección de anomalías en la red de datos de la Universidad de Cienfuegos “Carlos Rafael Rodriguez”. En el presente trabajo se describen dos técnicas inteligentes, Perceptrón Multicapas (MLP) y Clustering, ambas ampliamente aplicadas en la detección de anomalías en diversos ambientes de redes de datos, así como se justifican las herramientas utilizadas durante el desarrollo de la investigación, definiendo los atributos seleccionados para el conjunto de entrenamiento y su estandarización y se muestran los resultados alcanzados en el entrenamiento del MLP, para diferentes modos de evaluación, y los agrupamientos obtenidos mediante la técnica de clustering empleada. Los resultados obtenidos durante el proceso de investigación se traducen a la propuesta de un modelo MLP y Clustering para la correcta clasificación de anomalías en la red de datos de la UCF.

ABSTRACT

The main objective of this investigation is the proposal of techniques of artificial intelligence for the identification of anomalies in the data net of Cienfuegos's University “Carlos Rafael Rodríguez”. In the present investigation two intelligent techniques are described: MLP and Clustering. Both techniques are largely applied in the identification of anomalies with different settings of data nets. The tools used during the process of the investigation are also explained and the attributes selected for the workout and its standardization are defined. The results attained in the workout of the MLP for different types of evaluation are showed as well as the groups obtained through the clustering technique that was used. The results obtained during the investigation process are finally translated into an MLP's and Clustering proposal model for the correct classification of anomalies in the data net of the UCF.

ÍNDICE GENERAL

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

INTRODUCCIÓN

La seguridad de redes es un problema importante en el mundo moderno. El número de computadoras, la conectividad entre ellas y la importancia de su uso para la vida cotidiana del hombre es cada vez mayor y por tanto se hace cada vez más complejo mantener los estándares de seguridad esperados, por lo que las organizaciones y empresas siempre buscan proteger su información y garantizar la confidencialidad, integridad y disponibilidad de sus recursos.

El creciente número de usuarios que se conectan a la red de información, así como el fácil acceso a sitios que muestran cómo explotar las vulnerabilidades de las mismas hacen de cualquier persona un atacante. Por los anteriores motivos es de vital importancia proveer a las redes de datos de mecanismos capaces de detectar intrusiones y manejos maliciosos.

La detección de anomalías surge actualmente como respuesta a una necesidad clave para la identificación de todo tipo de delitos y abusos en el área de las redes de ordenadores. Se pueden disponer muchas medidas de seguridad; pero si éstas se ven superadas de alguna forma, es necesario disponer de una herramienta que lo identifique de manera inmediata. Sería económicamente prohibitivo colocar cortafuegos en todos los puntos críticos de una red y realizar auditorías de red a todas las horas del día. La detección de anomalías proporciona evidencias incriminatorias, con frecuencia el anuncio de la existencia de herramientas de este tipo encargadas de la monitorización y detección de intrusiones en redes, es suficiente para disuadir a los posibles atacantes para que busquen otros objetivos con menos obstáculos de seguridad. La combinación del crecimiento de Internet, las grandes posibilidades financieras que abre el comercio electrónico y la no existencia de sistemas verdaderamente seguros han convertido a estas herramientas en campo clave, estratégico y muy necesario para complementar a la infraestructura de seguridad existente. Por lo que ha devenido en un gran número de investigadores que han dedicado recursos y esfuerzos para ampliar los conocimientos en esta área.

La Inteligencia Artificial (I.A.) es utilizada en este campo, la cual se ha convertido en los últimos años en una de las ramas de las Ciencias de la Computación más difundidas y para las que se ha dedicado un gran esfuerzo, por los beneficios que reporta en la solución de problemas donde es necesaria, principalmente, la manipulación de información simbólica, en aquellos problemas en los que aparece la subjetividad para resolverlos o en los que la

información a manipular es incompleta o borrosa. Se ocupa de la representación, adquisición y procesamiento de conocimientos de forma automatizada, es un campo que por sus investigaciones trata de ser independiente de la informática, y se define como la técnica de software que los programas utilizan para dar solución a algún tipo de problema, pero tratando de asemejar el comportamiento inteligente que se observa en la naturaleza; es decir, trata de resolver problemas y tomar decisiones similares a las que toman los seres humanos al afrontar la vida diaria, realizando programas de computadora que aumenten la capacidad o "inteligencia" de las mismas; el objetivo de sus investigaciones es, aumentar la utilidad de las máquinas y sus procesos.

El análisis de las trazas generadas por los servicios telemáticos y los equipos de la red, es una herramienta muy útil para prevenir y evitar muchos de los problemas que acontecen en las redes de datos, pues permite mantener un control más estricto sobre dichos sistemas. Por esta razón, es usual que los administradores de una red las examinen cuando se están buscando las causas de un error. Sin embargo, ocurre con frecuencia que el cúmulo de datos que se almacena en estos archivos es tan grande que termina por agotar al administrador sin haber encontrado nada útil, situación esta que se agrava cuando se cuenta con un sistema que centraliza toda la información proveniente de los servicios prestados y/o dispositivos de red, tal es el caso del "Sistema para la centralización, análisis y procesamiento de trazas de servicios telemáticos para la Red UCf". El mismo posee una base de datos donde se almacena toda la información proveniente de los servicios que se prestan en la red de ordenadores de la Universidad de Cienfuegos.

Esta herramienta le permite a los administradores realizar un análisis de la información almacenada a través de diferentes funciones que exigen de su participación directa, lo cual implica que los resultados dependen en gran medida de la experiencia acumulada. La herramienta no cuenta con funcionalidades que automaticen la clasificación de eventos anómalos, lo cual resulta una característica deseable en sistemas de este tipo, teniendo en cuenta la diversidad de anomalías conocidas y no conocidas que afectan constantemente la seguridad informática.

Problema a resolver:

Ante esta situación y dada la importancia que tiene mantener la integridad, confidencialidad y disponibilidad de los activos informáticos, queda planteado como problema a resolver: Imposibilidad del "Sistema para la centralización,

análisis y procesamiento de trazas de servicios telemáticos para la Red UCf" de clasificar de manera automática la ocurrencia de eventos anómalos.

Objeto de estudio:

Los sistemas de detección de anomalías en las redes telemáticas.

Campo de acción:

Las técnicas de Inteligencia Artificial Perceptrón Multicapas y Clustering para la clasificación de eventos anómalos en las redes de datos de la UCf.

Objetivo general:

Proponer técnica de Inteligencia Artificial, para la detección automática de anomalías en las trazas almacenadas por el "Sistema para la centralización, análisis y procesamiento de trazas de servicios telemáticos para la Red UCf".

Tareas científicas:

Para el desarrollo de la investigación se utilizarán diferentes métodos y técnicas que nos permitirán enfrentar el problema. Estos métodos y técnicas favorecerán el cumplimiento de las siguientes tareas:

- Revisión bibliográfica de técnicas inteligentes más utilizadas en los sistemas de detección de anomalías.
- Estudio de los principales conceptos asociados a la detección de anomalías en las redes de datos.
- Selección de las tendencias, tecnologías y herramientas para el desarrollo de la investigación.
- Determinación del conjunto de entrenamiento para el aprendizaje, la evaluación y clasificación.
- Aplicación sobre el conjunto de entrenamiento de las técnicas MLP y Clustering.
- Análisis del rendimiento de las técnicas aplicadas en la detección de anomalías en la red de datos de la UCf.

Idea a defender:

Con el uso de técnicas inteligentes se logra clasificar automáticamente anomalías en la red de datos de la Universidad de Cienfuegos.

Principal aporte:

Es el empleo de técnicas de Inteligencia Artificial para el análisis de las trazas en la red de datos de la Universidad de Cienfuegos.

Estructura del trabajo:

El trabajo estará estructurado de la siguiente forma:

Introducción, tres capítulos, conclusiones, recomendaciones, referencias bibliográficas y bibliografía. Los capítulos tendrán la información que a continuación se describe:

Capítulo 1: Descripción de las técnicas inteligentes MLP y Clustering, y su aplicación a la detección de anomalías en las redes de datos.
Capítulo 2: Aplicación del MLP y el Clustering al análisis de trazas.

Capítulo 3: Análisis de los resultados.
Conclusiones.

Recomendaciones.

Referencias Bibliográficas.

Bibliografía.

Capítulo I: Descripción de las técnicas inteligentes MLP y Clustering, y su aplicación a la detección de anomalías en las redes de datos.

En el presente capítulo se explicitan las principales definiciones del término Inteligencia Artificial, rama perteneciente a la ciencia de la Computación que se dedica al empleo de numerosos recursos para el desarrollo de investigaciones en las últimas décadas. Además, se define el aprendizaje como elemento fundamental de cualquier sistema que se considere inteligente y su clasificación: supervisado y no supervisado. Se describen dos técnicas inteligentes, Perceptrón Multicapas (MLP) y Clustering, la primera supervisada y la segunda, no supervisada, ambas ampliamente aplicadas en la detección de anomalías en diversos ambientes de redes de datos.

1.1 Introducción a la Inteligencia Artificial. Definición.

En el año 1937 Alan Turing1 emprendió el estudio formal de la computación, e introdujo una máquina ideal conocida como máquina de Turing (1). Más tarde, en la década del 40, el matemático de origen húngaro John Von Neumann, concibió una computadora basada en lógica digital que opera ejecutando en serie las instrucciones que componen un algoritmo que se codifica en forma de programa almacenado en memoria. Debido a su eficacia, flexibilidad y versatilidad, soportado por el impresionante desarrollo de las tecnologías electrónicas, este último resultó ser el enfoque dominante en las últimas décadas, de modo que el binomio lógica booleana-máquina Von Neumann2 es la base sobre la que se asientan la mayor parte de los actuales computadores digitales.

La computación algorítmica se sustenta en resolver cada problema mediante un algoritmo, que se codifica en forma de programa y se almacena en memoria; el programa es ejecutado en una máquina secuencial. Como desarrollo natural de esta tendencia, algunos pioneros, como Turing o Von

---

Alan Mathison Turing (1912 – 1954). Matemático inglés, fundador de la Ciencia de la Computación.

John Louis Von Neumann (1903 - 1957). Matemático húngaro. Ocupa un lugar privilegiado en la historia de la computación debido a sus múltiples e importantísimos aportes a las computadoras de la primera generación.

Neumann, abrigaban la esperanza de que pudiera incorporarse en una de estas máquinas la capacidad de pensar racionalmente, en la forma de un complejo software. En este sentido, en 1950 Claude Shannon3 y el mismo Turing diseñaron los primeros programas que permitían a un ordenador digital **razonar** y jugar al ajedrez (1). A lo largo de la década de los cincuenta se prosiguió trabajando en este sentido. En el año 1957 A. Newell, H. Simon y J. Shaw (2) presentaron el *Teórico Lógico*, el primer programa capaz de razonar sobre temas arbitrarios. Hacia 1960 John McCarthy (2) profesor del Instituto de Tecnología de Massachusetts acuña el término **Inteligencia Artificial** (IA), para definir los métodos algorítmicos capaces de hacer pensar a los ordenadores. Según Schildt (2) un **programa inteligente** es aquel programa que muestra un comportamiento similar al humano cuando se enfrenta con un problema. No es necesario que el programa resuelva realmente el problema de la misma forma que el hombre. Alan Turing (2) se expresa análogamente al señalar que *"si durante el intercambio entre una computadora y el usuario, este último cree que está intercambiando con otro humano, entonces se dice que el sistema es inteligente"*. Esto se conoce como el test de Turing y fue formulado en 1950. Para Forsyth (2) la I.A. se relaciona con problemas que han escapado de una caracterización matemática. Para Cuena (2), la computación tradicional se basa en el empleo de algoritmos en los cuales el conocimiento y el procedimiento de solución están integrados en un proceso unificado, los sistemas de Inteligencia Artificial utilizan procedimientos estándares que permiten modificar los procesos sin tener que modificar el conocimiento. Realmente no existe aún ninguna definición única y rigurosa que cubra adecuadamente todos los aspectos que el término quiere representar. La investigadora Elaine Rich (3) la define de la forma siguiente "L*a IA es el estudio de cómo lograr que las computadoras hagan cosas que por el momento, las personas hacen mejor. Una computadora encuentra las raíces de una ecuación mucho más rápido y con mayor exactitud que un hombre; sin embargo, el hombre reconoce mucho mejor un conjunto de caracteres*".

A pesar de la diversidad de definiciones existentes, las investigaciones en la IA se realizan con dos propósitos fundamentales, lograr que las computadoras ejecuten tareas que resueltas por humanos se suelen llamar inteligentes y comprender los principios que hacen esta inteligencia posible.

Claude Shannon (1916-2001). Matemático norteamericano. Considerado el padre de la Teoría de la Información.

La IA es una rama de la Ciencia de la Computación dedicada a la creación de hardware y software que imita el pensamiento humano. Su principal objetivo es llevar a la computadora las amplias capacidades del pensamiento humano y, para ello, se convierten a las computadoras en "entes inteligentes" con la creación de software que les permite imitar algunas de las funciones del cerebro humano en aplicaciones particulares. El fin no es reemplazar al hombre, sino proveerlo de una herramienta poderosa para asistirlo en su trabajo.

La IA se ocupa de la representación, adquisición y procesamiento de conocimientos en forma automatizada, de la arquitectura de los programas para estas actividades y de los lenguajes en los que se expresan los programas. La modelación computacional de los procesos cognoscitivos es también un área de interés de la IA, además se incluyen la percepción, la comprensión y síntesis del lenguaje natural, la robótica inteligente, la modelación del razonamiento, la programación automática y otras más, todas ellas de naturaleza no numérica y todavía del dominio de la heurística4.

Las críticas más recurrentes sobre la IA, es que las máquinas no se pueden considerar "inteligentes" hasta que no sean capaces de aprender a hacer cosas nuevas y a adaptarse a nuevas situaciones, en lugar de limitarse a aquellas actividades para las que fueron programadas (3). Ada Augusta una de las primeras filósofas sobre la computación escribió sobre este tema:

"*La máquina analítica no tiene ninguna pretensión de crear nada. Puede hacer lo que sea si se le ha indicado la forma de hacerlo.".*

Por lo que queda claro que no hay nada que prohíba que una computadora se le diga cómo interpretar sus entradas de modo que su rendimiento aumente gradualmente. Simón (1983) propuso que el aprendizaje implica:

La esencia de la palabra heurística es contraria a la de algoritmo en el sentido de que ella es un camino para buscar lo nuevo, mientras el algoritmo es un camino para realizar lo ya muy bien conocido. Así se comprende que el paradigma primario para la resolución de problemas en Inteligencia Artificial sea la búsqueda de la solución orientada por la heurística, para tratar de reducir la explosión combinatoria que genera la búsqueda de todos los caminos posibles que se presenta en la mayoría de los problemas reales.

"… cambios en el sistema que se adaptan en el sentido de que le permiten llevar a cabo la misma tarea o tareas a partir de las mismas condiciones de un modo más eficiente y eficaz cada vez."

El aprendizaje cubre una amplia gama de fenómenos; en uno de los extremos del espectro se encuentra el perfeccionamiento de la habilidad. En general las personas mejoran la ejecución de determinadas tareas con el mero hecho de practicarlas. En el otro extremo del espectro se encuentra la adquisición de conocimiento, la mayoría de los programas de IA confían principalmente en el conocimiento como fuente de recursos. El conocimiento se adquiere principalmente a través de la experiencia.

Los programas de IA requieren de conocimiento y sus técnicas consisten en métodos para explotar este, el cual debe ser representado de manera que:

- Capte generalizaciones: No es una base de datos. No debe ser necesario representar cada situación individual, sino que se agrupen las situaciones que compartan propiedades importantes. Si no tiene esta característica se necesitaría más espacio del disponible y más tiempo del que tenemos para mantenerlo actualizado.
- Pueda ser comprendido por los especialistas que lo proporcionan.
- Deba ser modificable fácilmente.
- Pueda ser usado en muchas situaciones diversas, incluso si no es totalmente preciso o completo.
- Pueda ser usado para extenderse a sí mismo.

La IA incluye la solución de problemas dentro de diversos campos como la robótica, la comprensión y traducción de lenguajes, el reconocimiento y aprendizaje de palabras de máquinas o los variados sistemas computacionales expertos, que son los encargados de reproducir el comportamiento humano en una sección del conocimiento. Sus técnicas son muy aplicables problemas de predicción, clasificación y reconocimiento de patrones.

Teniendo en cuenta que la inteligencia es un término umbral que describe un estado mental en el que se han desarrollado diversas capacidades complejas, para poder simular o desarrollar en las máquinas este estado mental tendría que ser capaces de razonar, conocer, planificar, aprender, comunicarse, percibir y ser capaces de interactuar con su entorno. El objetivo mediato de la IA es el desarrollo de un sistema capaz de exhibir todas estas habilidades de

una forma conjunta y sobre una gran variedad de dominios, excediendo las habilidades de un ser humano.

1.2 Aprendizaje en los Sistemas Inteligentes.

Hasta hace unas pocas décadas atrás, creer que un computador podría aprender era algo inconcebible. La capacidad de aprendizaje que tienen los seres humanos es asombrosa, desde cómo un niño termina hablando o da sus primeros pasos, hasta algo tan básico como la capacidad para reconocer a alguien conocido hace unos años atrás.

El Aprendizaje Automático o Machine Learning, comprende diferentes mecanismos, reglas, enfoques y tecnologías mediante los cuales un computador puede aprender a desarrollar tareas que los seres humanos hacen de forma natural y rápida, como por ejemplo: reconocer imágenes, entender el lenguaje natural y tomar decisiones. Existen dos tipos de aprendizaje: supervisado y no supervisado, para explicar la diferencia básica entre uno y otro se utilizarán como ejemplo los problemas de clasificación: el sistema supervisado utiliza un conjunto de datos (conjunto de entrenamiento) para el proceso de aprendizaje compuesto por las entradas y la salida obtenida (clase a la que pertenece el objeto a clasificar), mientras el no supervisado no requiere conocer la clase a predecir, en este último el propio sistema se autoorganiza para identificar grupos de datos que bajo características comunes se agrupan en una misma clase.

1.2.1 Aprendizaje supervisado.

Un algoritmo de aprendizaje supervisado produce una función que establece una correspondencia entre las entradas y las salidas deseadas del sistema. Un ejemplo de este tipo es el problema de clasificación, donde el sistema de aprendizaje trata de etiquetar (clasificar) una serie de vectores utilizando una entre varias categorías (clases). La base de conocimiento del sistema está formada por ejemplos de etiquetados anteriores.

1.2.2 Aprendizaje no supervisado.

En el aprendizaje no supervisado todo el proceso de modelado se lleva a cabo sobre un conjunto de ejemplos formado tan sólo por entradas al sistema. No se tiene información sobre las clases de esos ejemplos.

1.3 Dos técnicas inteligentes ampliamente aplicadas.

En el presente epígrafe se realiza un estudio de dos técnicas inteligentes ampliamente aplicadas, la primera supervisada y la segunda, no supervisada.

1.3.1 Redes Neuronales Artificiales.

La primera red neuronal artificial conocida, fue desarrollada en 1943 por Warren McCulloch y Walter Pitts (4). En los últimos años, se ha consolidado en un nuevo campo dentro de las Ciencias de la Computación caracterizadas por su inspiración en los sistemas biológicos para resolver problemas relacionados con el mundo real (reconocimiento de formas y toma de decisiones), ofreciendo soluciones robustas y de fácil implementación. Las Redes Neuronales Artificiales (RNAs) actualmente están causando un mayor impacto, debido a su extraordinaria aplicabilidad práctica. Esta tecnología ha captado la atención de los profesionales dedicados a la estadística y al análisis de datos, los cuales comienzan a incorporar las redes neuronales a sus estudios.

Las RNAs o sistemas conexionistas, como también se les conoce, son sistemas de procesamiento de la información cuya estructura y funcionamiento están inspirados en las redes neuronales biológicas. Consisten en un nodo conectado con otros mediante enlaces que corresponden a conexiones axón-sinapsis-dendritas. En la neurona biológica el axón transporta la salida de la neurona hasta las conexiones de otras neuronas, los espacios por donde estos influyen sobre las dendritas se le denomina sinapsis, siendo las dendritas quienes proporcionan las áreas superficiales para facilitar la conexión con otros axones de otras neuronas (Figura 1). A cada enlace está asociado un peso el cual determina la naturaleza e intensidad de la influencia de un nodo sobre otro, es el producto de la salida de la neurona que influye por el peso del enlace que los conecta, por lo que, un peso positivo grande corresponde a una excitación fuerte y un peso negativo pequeño corresponde a una inhibición débil. Cada nodo combina las influencias separadas que recibe en sus enlaces de entrada en una influencia global, mediante una función de activación. Una sola función de activación simplemente pasa la suma de los valores de entrada a través de una función de umbral para determinar la salida del nodo. A continuación se muestran dos imágenes: la primera muestra el modelo de una neurona biológica y la segunda, el de una artificial.

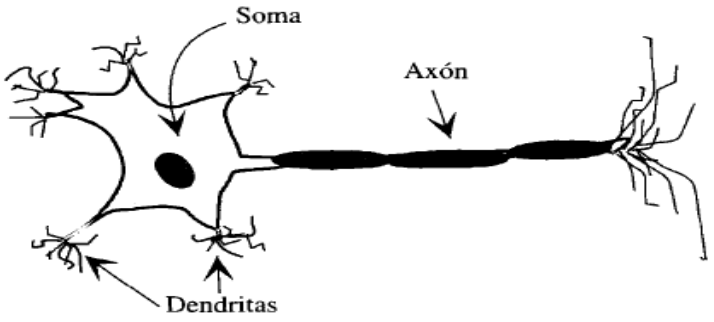

Figura 1: Modelo de una neurona biológica.

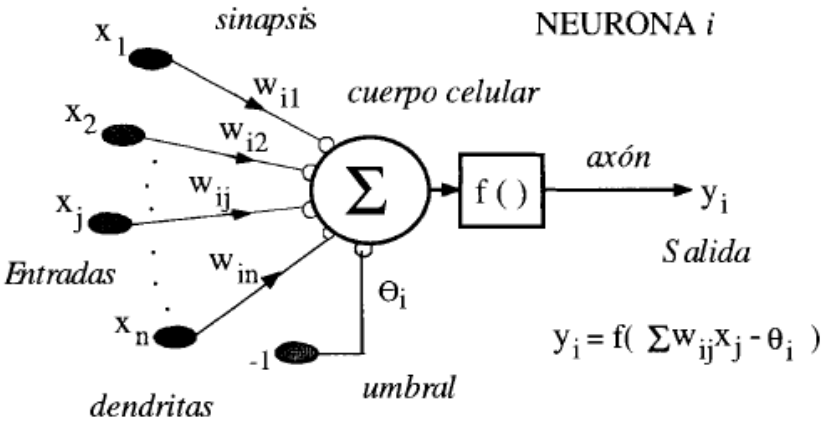

Figura 2: Modelo de una neurona artificial.

Atendiendo a distintos conceptos, pueden establecerse diferentes tipos de topologías neuronales; se define como topología de una red neuronal a la organización o arquitectura del conjunto de neuronas que la forman; esta organización comprende la distribución espacial de las mismas y los enlaces entre ellas. En un sistema de redes neuronales las conexiones sinápticas son direccionales, es decir, la información solamente puede propagarse en un único sentido (desde la neurona presináptica a la postsináptica). En general, las neuronas se suelen agrupar en unidades estructurales que se denominan capas. Las neuronas de una capa pueden agruparse, a su vez, formando grupos neuronales (clusters). Dentro de un grupo, o de una capa si no existe este tipo de agrupación, las neuronas suelen ser del mismo tipo. Finalmente, el conjunto de una o más capas constituye la red neuronal.

Se distinguen tres tipos de capas: de entrada, de salida y ocultas. Una capa de entrada o sensorial está compuesta por neuronas que reciben datos o señales procedentes del entorno (por ejemplo, proporcionados por sensores). Una capa

de salida es aquella cuyas neuronas proporcionan la respuesta de la red neuronal. Una capa oculta es aquella que no tiene una conexión directa con el entorno, es decir, que no se conecta directamente ni a órganos sensores ni a efectores. Este tipo de capa proporciona a la red neuronal grados de libertad adicionales, gracias a los cuales puede encontrar representaciones internas correspondientes a determinados rasgos del entorno, proporcionando una mayor riqueza computacional.

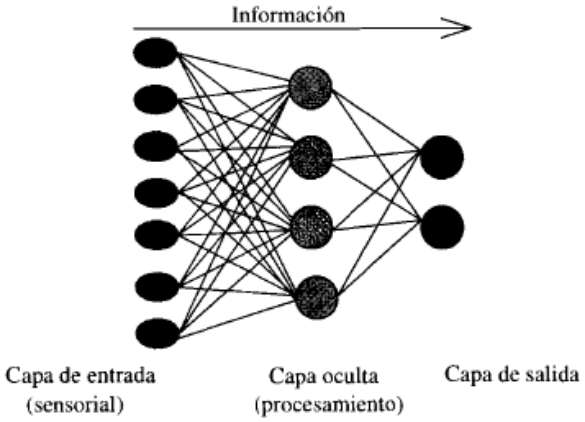

Figura 3**:** Arquitectura unidireccional de tres capas.

Se pueden definir conexiones intra-capa e inter-capa. Las conexiones intra-capa, también denominadas laterales, tienen lugar entre las neuronas pertenecientes a una misma capa, mientras que las conexiones inter-capa se producen entre las neuronas de diferentes capas. Existen además conexiones realimentadas, que tienen un sentido contrario al de entrada-salida. En algunos casos puede existir realimentación incluso de una neurona consigo misma.

Atendiendo a distintos conceptos, pueden establecerse diferentes tipos de arquitecturas neuronales (Figura 4). Así, en relación a su estructura en capas, podemos hablar de redes monocapa y de redes multicapa. Las redes monocapa son aquellas compuestas por una única capa de neuronas. Las redes multicapa (layered networks) son aquellas cuyas neuronas se organizan en varias capas. Asimismo, atendiendo al flujo de datos en la red neuronal, podemos hablar de redes unidireccionales (feedforward) y redes recurrentes (feedback). En las redes unidireccionales**,** la información circula en un único sentido, desde las neuronas de entrada hacia las de salida. En las redes

recurrentes o realimentadas la información puede circular entre las capas en cualquier sentido, incluido el de salida-entrada.

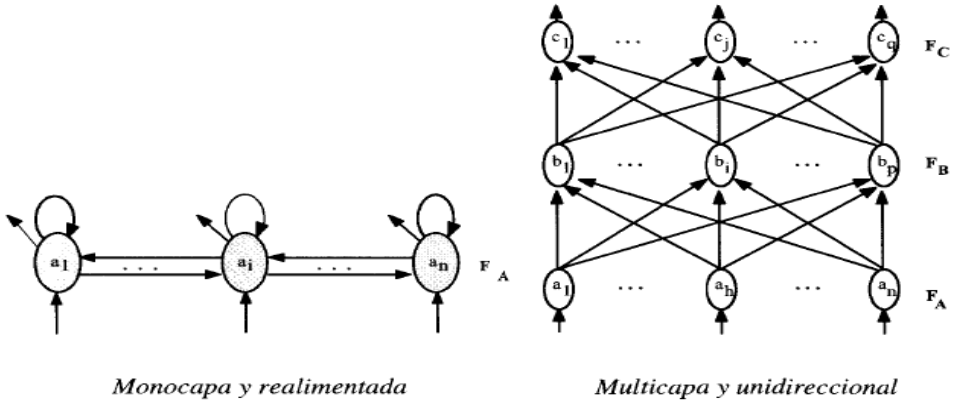

Figura 4: Ejemplos de arquitecturas neuronales.

El proceso de aprendizaje en las RNAs puede ser visto como el problema de actualizar la estructura de red y los pesos de las conexiones para que la red pueda realizar una tarea específica de manera eficiente. La capacidad de aprender de ejemplos y "entender" las relaciones ocultas entre entradas y salidas hace de las redes neuronales una herramienta poderosa para la resolución de problemas, ellas son capaces de extraer los rasgos generales de los ejemplos de entrenamiento. Las RNAs realizan el aprendizaje como resultado de un "entrenamiento", proceso mediante el cual se le presentan a la red un conjunto de datos (generalmente es un subconjunto representativo de todo el conjunto de datos disponibles), y se ajustan los pesos de manera que cada patrón de entrada produzca la salida deseada.

Los dos tipos básicos de aprendizaje son el supervisado y el no supervisado, cuya distinción proviene en origen del campo del reconocimiento de patrones. Ambas modalidades pretenden estimar funciones entrada/salida multivariable o densidades de probabilidad, pero mientras que en el aprendizaje supervisado se proporciona cierta información sobre estas funciones (como la distribución de las clases, etiquetas de los patrones de entrada o salidas asociadas a cada patrón), en el no supervisado no se proporciona información alguna. Las reglas de aprendizaje supervisadas suelen ser computacionalmente más complejas, pero también son más exactos sus resultados.

En el aprendizaje supervisado se presenta a la red un conjunto de patrones, junto con la salida deseada u objetivo, e iterativamente esta ajusta sus pesos hasta que su salida tiende a ser la deseada, utilizando para ello información detallada del error que comete en cada paso. De este modo, la red es capaz de

estimar relaciones entrada/salida sin necesidad de proponer una cierta forma funcional de partida.

En el aprendizaje no supervisado se presentan a la red multitud de patrones sin adjuntar la respuesta que deseamos. La red, por medio de la regla de aprendizaje, estima la función: *densidad de probabilidad*5 px, a partir de lo cual pueden reconocerse regularidades en el conjunto de entradas, extraer rasgos, o agrupar patrones según su similitud (clustering). Un ejemplo típico de modelo que emplea este tipo de aprendizaje es el de los mapas autoorganizados.

El procesamiento en un sistema conexionista no es secuencial sino paralelo, esto es, muchas unidades de procesamiento pueden estar funcionando simultáneamente. En redes neuronales la información de un sistema no está localizada o almacenada en compartimentos discretos, sino que está distribuida a lo largo de los parámetros del sistema. Los parámetros que definen el "conocimiento" que una red neuronal posee en un momento dado son sus conexiones y el estado de activación de sus unidades de procesamiento.

En un modelo conexionista, cada nodo lleva a cabo una computación simple. La fiabilidad de la computación total que el sistema realiza depende de la interacción paralela de un gran número de nodos y, consecuentemente, en la mayoría de los casos, el sistema puede continuar su funcionamiento normal, aunque una pequeña parte del mismo haya resultado dañada. Por otro lado, un modelo conexionista es capaz, en ciertas circunstancias, de reconocer un objeto a pesar de que sólo se le presente como entrada una parte del mismo, o a pesar de que la imagen del objeto esté distorsionada, los mismos no se programan para realizar una determinada tarea sino que son "entrenados" a tal efecto.

Las características de las RNA las hacen bastante apropiadas para aplicaciones en las que no se dispone a priori de un modelo identificable que pueda ser programado, pero se dispone de un conjunto básico de ejemplos de entrada (previamente clasificados o no). Asimismo, son altamente robustas tanto al ruido como a la disfunción de elementos concretos y son fácilmente paralelizables.

Describe la distribución de patrones x pertenecientes al espacio de entrada, a partir de muestras.

Esto incluye problemas de clasificación y reconocimiento de patrones de voz, imágenes y señales. Asimismo se han utilizado para encontrar patrones de fraude económico, hacer predicciones en el mercado financiero, hacer predicciones de tiempo atmosférico.

Esta tecnología puede ser desarrollada tanto en software como en hardware y con ella se pueden construir sistemas capaces de aprender, de adaptarse a condiciones variantes, o inclusive si se dispone de una colección suficientemente grande de datos, predecir el estado futuro de algunos modelos. Las aplicaciones más habituales son las relacionadas con clasificación, estimación funcional y optimización, reconocimiento de patrones, del habla y de caracteres, visión, robótica, control, economía y defensa. Son utilizadas para la predicción, el reconocimiento de patrones, los sistemas de control adaptativo y el análisis de datos siendo una de las técnicas utilizadas por la minería de datos. Constituyen una parte muy importante en el estudio y desarrollo de la IA.

Una de las principales ventajas de las redes neuronales es que, teóricamente, son capaces de aproximar cualquier función continua. Una desventaja importante, sin embargo, es que la solución final depende de las condiciones iniciales de la red.

Existen atendiendo a la revisión bibliográfica realizada cincuenta modelos de RNA conocidos: Perceptrón, Perceptrón Multicapa, ADALINE, MADALINE, Hopfield, Máquina de Boltzman, Mapa de Kohonen, aunque tan sólo aproximadamente una quincena son utilizados con asiduidad en las aplicaciones prácticas.

1.3.1.1 Modelo de Rede Neuronal Artificial. Perceptrón Multicapas.

El modelo de RNA Perceptrón Multicapas tiene sus antecedentes en el Perceptrón Simple.

Perceptrón Simple:

Este modelo neuronal fue introducido por Rosenblatt a finales de los años cincuenta (1). La estructura del Perceptrón se inspira en las primeras etapas de procesamiento de los sistemas sensoriales de los animales (por ejemplo, el de la visión), en los cuales la información va atravesando sucesivas capas de neuronas, que realizan un procesamiento progresivamente de más alto nivel.

El Perceptrón Simple es un modelo unidireccional, compuesto por dos capas de neuronas, una sensorial o de entradas, y otra de salida. Las neuronas de entrada no realizan ningún cómputo, únicamente envían la información a las neuronas de salida. La función de activación de las neuronas de la capa de salida es de tipo escalón.

El algoritmo de aprendizaje del Perceptrón, permite la determinación automática de los pesos sinápticos de los denominados por corrección de errores. Este algoritmo ajusta los pesos en proporción a la diferencia existente entre la salida actual de la red y la salida deseada, con el objetivo de minimizar el error actual de la red.

La importancia histórica del Perceptrón radica en su carácter de dispositivo entrenable, pues el algoritmo de aprendizaje del modelo introducido por Rosenblatt, permite determinar automáticamente los pesos sinápticos que clasifican un conjunto de patrones a partir de un conjunto de ejemplos etiquetados, por lo que puede utilizarse como clasificador. Cada neurona del Perceptrón representa una determinada clase, de modo que dado un vector de entrada, una cierta neurona responde con 0 si no pertenece a la clase que representa, y con un 1 si pertenece. Es fácil ver que una neurona tipo Perceptrón solamente permite discriminar entre dos clases linealmente separables[6], (es decir, cuyas regiones de decisión pueden ser separadas mediante una única condición lineal o hiperplano).

Por lo tanto, pese a su gran interés, el perceptrón presenta serias limitaciones, pues solamente puede representar funciones linealmente separables. Así, aunque pueda aprender automáticamente a representar complejas funciones booleanas o resolver con éxito muchos problemas de clasificación, en otras ocasiones fallará estrepitosamente.

Minsky y Papert estudiaron en profundidad el Perceptrón (5), y en 1969 publicaron un exhaustivo trabajo en el que se subrayaba sus limitaciones, lo que resultó decisivo para que muchos de los recursos que se estaban invirtiendo en redes neuronales se desviasen hacia otros campos más prometedores como la Inteligencia Artificial.

Perceptrón Multicapa:

El Perceptrón Simple alcanzó gran popularidad durante la década de los años sesenta, observando sus limitaciones computacionales, se le añadieron capas ocultas llegando a la arquitectura Perceptrón Multicapa (Multilayer Perceptron,

---

Una función se dice **linealmente separable** cuando su espacio de variables de entrada puede ser dividido en regiones de igual salida mediante una única condición lineal (un hiperplano: una línea recta, si trabajamos en dos dimensiones).

MLP), ya que mediante una estructura de dos capas, sin capa oculta (la que corresponde a un Perceptrón Simple), la región de decisión es un hiperplano que separa en dos el espacio de las variables. Haciendo uso de tres capas, con una oculta, se pueden discriminar regiones convexas, sean cerradas o abiertas. Con una estructura de cuatro capas, dos de ellas ocultas, se puede discriminar regiones de forma arbitraria, cuyo único límite viene impuesto por el número de nodos empleados.

Esta estructura se entrena mediante el algoritmo de retropropagación (backpropagation) de errores o variantes, por lo que en muchas ocasiones al conjunto arquitectura más aprendizaje se le denomina *red de retropropagación (BP),* algoritmo que soluciona el problema de entrenar los nodos de las capas ocultas.

La idea general del algoritmo se expresa a continuación.

- La red aprende un conjunto predefinido de pares de entradas y salidas dadas como ejemplo, empleando un ciclo de propagación primero y adaptación después.
- Una vez que se ha aplicado un patrón de entrada como estímulo para la primera capa de unidades de la red, esta se va propagando a través de todas las capas superiores hasta generar una salida.
- La señal de salida se compara entonces con la salida deseada, y se calcula el error para cada unidad de salida. Sin embargo este cálculo del error no se puede hacer directamente en los nodos de la capa intermedia (o de las capas intermedias, si existieran más de una). Lo que se hace es propagar hacia atrás el error. Este proceso de retropropagación del error es lo que se denomina BP.
- Los valores de error se transmiten hacia los nodos de las otras capas, partiendo de la capa de salida. Cada nodo que tuvo algún error, propaga hacia todos los nodos de la capa intermedia que contribuyan directamente a su salida, una fracción del error ocurrido durante el proceso.
- Las unidades de la capa intermedia sólo reciben una fracción de la señal total del error, basándose aproximadamente en la contribución relativa que haya aportado a la unidad de error de la capa de salida. Esto no sólo se hace para un nodo, sino que se realiza la misma operación para todos los nodos de la capa de salida. De esta forma, cada nodo de la(s)

capa(s) intermedia(s) recibirá una fracción de todos los errores que provocó su salida en la propagación anterior.

- Este proceso se repite, capa por capa, hasta que todos los nodos de la red hayan recibido un valor de error que considere su aporte relativo al error total.
- Basándose en el valor de error percibido, se actualizan los pesos de conexión de cada unidad, para hacer que la red converja hacia un estado que permita codificar todas las tramas de entrenamiento.

La importancia de este proceso consiste en que, a medida que se entrena la red, los nodos de las capas intermedias se organizan a sí mismos de tal modo que los distintos nodos aprenden a reconocer distintas características del espacio total de entradas.

Uno de los problemas del algoritmo es que en busca de minimizar la función de error, puede caer en un mínimo local o en algún punto estacionario, con lo cual no se llega a encontrar el mínimo global de la función de error. Sin embargo, no tiene porqué alcanzarse el mínimo global en todas las aplicaciones, sino que puede ser suficiente con un error mínimo preestablecido.

Aplicando *red de retropropagación* a numerosos problemas, se comprobó experimentalmente que éste era capaz de abordar problemas de clasificación de gran envergadura, de una manera eficaz y relativamente simple. Sin embargo, faltaba una demostración teórica que permitiese explicar sus aparentemente enormes capacidades computacionales. Este proceso histórico comienza con McCulloch y Pitts **7** (1), quienes mostraron que mediante su modelo de neurona (esencialmente un dispositivo de umbral) podría representarse cualquier función booleana; mucho más tarde, Denker y otros demostraron que toda función booleana podía ser representada por una red unidireccional multicapa de una sola capa oculta. Lippmann mostró que un Perceptrón con dos capas ocultas bastaba para representar regiones de decisión arbitrariamente complejas. Por otra parte, Lapedes y Farber demostraron que un Perceptrón de dos capas ocultas es suficiente para

---

MacCulloch y Pitts demostraron ya que cualquier función lógica arbitraria puede ser construida con una apropiada combinación de elementos basados en su modelo de neurona, observando que la función NAND podía implementarse con un caso particular de ella.

representar cualquier función arbitraria (no necesariamente booleana). Más tarde, Hecht-Nielsen demostró que una arquitectura de características similares al MLP, con una única capa oculta, resultaba ser un aproximador universal de funciones. Diversos grupos propusieron casi a la par teoremas muy similares que demostraban matemáticamente que un MLP convencional, *de una única capa oculta puede aproximar hasta el nivel deseado cualquier función continua en un intervalo*, por lo tanto, las redes neuronales multicapa unidireccionales son aproximadores universales de funciones.

1.3.2 Clustering.

El Clustering identifica clusters, o regiones densamente pobladas, de acuerdo a alguna medida de distancia o similitud, en un gran conjunto de datos multidimensional. El Clustering se basa en maximizar la similitud de las instancias en cada cluster y minimizar la similitud entre clusters. Es una técnica importante en el rápido crecimiento del campo conocido como la exploración y análisis de datos.

Dentro del análisis de Clustering existen, básicamente, los siguientes tipos de métodos: los jerárquicos, los de partición, los basados en densidad, los métodos basados en cuadrículas, los basados en restricciones y los escalabres. En el caso de los primeros, se intenta ordenar los elementos a distintos niveles de similitud; mientras que los segundos, meramente, asignan cada elemento a un grupo de manera tal de obtener conjuntos homogéneos. Consecuentemente, la configuración final de los grupos debe ser cuidadosamente analizada, para detectar tales situaciones. Esta agrupación se realiza de forma no supervisada, ya que no se conoce de antemano las clases del conjunto de datos de entrenamiento.

Clustering Particional:

Utiliza el algoritmo de partición K-medias: se seleccionan K objetos(datos) del conjunto de entrada los cuales serán los centroides iniciales de los K-grupos, se calculan las distancias de los objetos a cada uno de los centroides y se asignan a aquellos grupos cuya distancia es mínima con respecto al resto de los centroides, se actualizan los centroides como el valor medio de todos los objetos asignados a ese grupo, se re-calculan las distancias, la asignación de los datos y la actualización de los centroides hasta que se satisfaga algún criterio de convergencia.

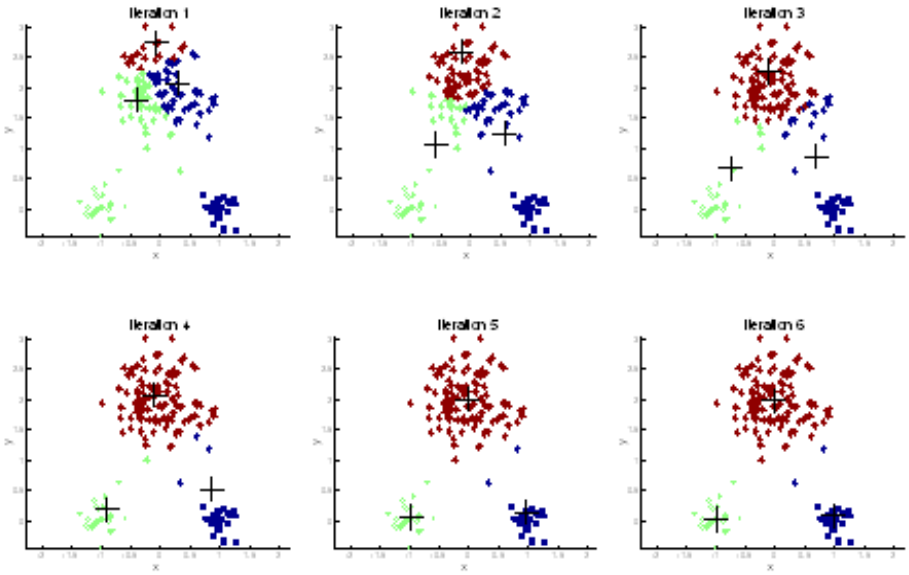

Figura 5: Clustering Particional. K-media

Una de las aplicaciones de las técnicas de cluster es el análisis de datos siendo la Minería de Datos (MD) una de las soluciones que más ayuda a extraer conocimiento a partir de los mismos. Este conocimiento puede obtenerse a partir de la búsqueda de conceptos, ideas o patrones estadísticamente confiables, que no son evidentes a primera vista, desconocidos anteriormente y que pueden derivarse de los datos originales. Dentro de las aplicaciones que se le puede dar a la MD se tienen algunas como: análisis de fidelización de clientes, segmentación de mercados, cross-selling, detección y prevención de fraudes, detección de intrusiones en sistemas computacionales y situaciones en las que se quiera analizar ciertos datos cuyo comportamiento parecen distintos del resto o también conocido como Detección de Anomalías (DA).

Es precisamente la DA la que puede convertirse en una herramienta de enorme utilidad, en conjunto con técnicas de Clustering, posibilita el reconocimiento de grupos de datos cuyo comportamiento es muy diferente al resto y también cuando no se conoce o no se puede etiquetar de manera confiable los datos para su clasificación.

1.4 Aplicaciones de técnicas inteligentes para la clasificación de anomalías en las redes de datos.

Una anomalía se puede definir como la discrepancia de una regla o de un uso. De ese modo, el primer paso de un sistema de detección de anomalías comienza por establecer lo que se considera comportamiento normal de un sistema (usuarios, redes, registros de auditoría y llamadas del sistema de los

procesos.). Una vez definido esto, clasificará como sospechosas o intrusivas aquellas desviaciones que pueda detectar sobre el comportamiento normal (6).

La detección de anomalías depende mucho de la suposición de que los usuarios y las redes se comportan de un modo suficientemente regular, de forma que cualquier desviación significante pueda ser considerada como evidencia de una intrusión (6).

La gran *ventaja* de la detección de anomalías es que el sistema es capaz de aprender el comportamiento normal del objeto de estudio, y a partir de ahí detectar desviaciones del mismo, siendo capaz de detectar tipos de ataques hasta el momento desconocidos. Como *desventaja*, por definición únicamente señala comportamientos inusuales, pero éstos no tienen necesariamente por qué ser ilícitos. Por ello, destaca el problema de su alta tasa de falsos positivos. Otra desventaja de este proceso es la falta de claridad (es un proceso borroso). Un intruso podría actuar lentamente y realizar sus acciones cuidadosamente para modificar el perfil de los usuarios de modo que sus actividades serían aceptadas como legales cuando en realidad deberían lanzar una alarma (falsos negativos) (6).

La gran mayoría de la investigación relacionada con detección de anomalías en redes se ha centrado en la inspección de la información de las cabeceras en los distintos paquetes, tanto en la capa de red (IP), como en las capas de transporte/control (TCP, UDP, ICMP, etc.). La idea de la detección de anomalías a nivel de aplicación es la de extender los modelos de “tráfico normal” analizando también la carga útil de la capa de aplicación (7).

En el presente epígrafe se resumen los resultados de la búsqueda realizada de técnicas inteligentes aplicadas a la detección de eventos anómalos o intrusiones en diversos ambientes de red.

1.4.1 Sistemas existentes vinculados al campo de acción.

El brasileño Cansian y sus colaboradores, desarrollaron el sistema SADI (Sistema Adaptativo de Detección de Intrusos), donde usan una red neuronal para identificar el comportamiento intrusivo de patrones capturados de una red de datos (8).

David Endler utilizó un MLP con dos propósitos, para la detección de uso indebido y para la detección de anomalías a partir de datos de auditoría procedentes del BSM (Basic Security Module) de Solaris (9).

Cannady y Mahaffey de Georgia Technical Research Institute (GTRI), dirigieron una investigación que aplicaba un modelo híbrido de MLP y mapas autoorganizativos para la detección de uso indebido (10). Este mismo equipo realizó otro trabajo en el 2000 utilizando múltiples mapas autoorganizativos para la detección de intrusos (11).

Lippmann y Cunningham realizaron un proyecto que mejoraba el rendimiento de la detección de ataques de tipo U2R (User to Root), el cual se da cuando un atacante que dispone de una cuenta en un sistema informático es capaz de elevar sus privilegios explotando vulnerabilidades en los mismos, un agujero en el sistema operativo o en un programa instalado en el sistema. Se usó una red de tipo MLP para la detección del mismo (12).

Bivens (13) realiza un trabajo parecido al de Cannady (10). Mientras que Cannady propuso un método de detección a nivel de paquete de red, el de Bivens se basa en el método de tiempo de ventana (time-window). Este método les permite reconocer ataques multi-paquetes más largos. También utilizan un modelo híbrido de MLP y mapas autoorganizativos, en este caso para la detección de anomalías.

SIDIRI es una propuesta de un modelo que utiliza sistemas inteligentes, en particular RNAs, como motor de análisis de IDS con el fin de obtener una respuesta inteligente y proactiva8 a las necesidades de seguridad en las redes telemáticas del mundo de hoy, es esencialmente para el protocolo TCP/IP, como parte de la propuesta se realizó un análisis de las ventajas que puede brindar la aplicación de redes neuronales en esta área (14):

- La red aprenderá de manera autónoma a partir de los ejemplos.
- La red neuronal artificial se puede adaptar a nuevos comportamientos, la cual hace al sistema mucho más flexible a variaciones y modificaciones de los métodos de intrusión actualmente conocidos.
- Disminuir la cantidad de falsos positivos y falsos negativos.
- La red infiere ataques que no aprendió.

Casos de estudio

Previo a que el error ocurra.

Se centró la atención en dos sistemas en particular, con los cuales se realizarán comparaciones en los próximos capítulos. En lo adelante se denominarán *casos de estudio.*

Caso de Estudio 1

En el 2001, Leonid Portnoy, Eleazar Eskin y Sal Stolfo del Departamento de Ciencias de la Computación de la Universidad de Columbia, New York, realizaron clustering en instancias de datos que contenían tanto comportamientos normales como ataques. Para ello utilizaron un algoritmo modificado de k-means. Tras realizar la agrupación, utilizaron heurísticas para etiquetar de forma automática cada grupo en normal o ataques (15).

Caso de estudio 2

En el año 2006 Shahbaz Pervez, Iftikhar Ahmad, Adeel Akram y Sami Ullah Swati de la Universidad de Ingeniería y Tecnología de Taxila, Pakistán, utilizaron una Red Neuronal Artificial, MLP de 4 capas, con entrenamiento mediante retropropagación para la detección de intrusos en redes de datos (16).

1.5 Detección de anomalías en la red de datos de la Universidad de Cienfuegos.

La Universidad de Cienfuegos cuenta con un "Sistema para la centralización, análisis y procesamiento de trazas de servicios telemáticos para la Red UCf". La centralización de las trazas es posible debido a la instalación de la herramienta Syslog-ng, y son almacenadas en la base de datos del sistema. Debido a esta centralización el volumen de información es de gran tamaño, por lo cual la actividad de monitoreo diario y detección de eventos anómalos se hace muy tediosa para los administradores de red, no haciéndolo con la frecuencia necesaria. Todas las trazas almacenadas son procesadas por un *Módulo de Gestión de Alarmas* basado en reglas (17). Debido al volumen de información a procesar, así como la cantidad de reglas a clasificar, esta actividad consume mucho tiempo. Además, dado la ocurrencia de una nueva anomalía la misma no podrá ser detectada por dicho módulo, las reglas son definidas por conocimientos previos de lo que es considerado un evento anómalo.

1.6 Selección de las técnicas inteligentes para la detección de anomalías en la Universidad de Cienfuegos.

Después del estudio del estado del arte y de las características de las técnicas de Inteligencia Artificial estudiadas en el proceso de la investigación, así como su aplicación en la clasificación y detección de anomalías, se decidió utilizar MLP y Clustering para la detección de eventos anómalos en la red de datos de la Universidad de Cienfuegos.

Las redes MLP presentan características que hacen ventajoso su uso en el problema de la detección de eventos anómalos, ejemplo de ellos es su gran flexibilidad. El MLP es capaz de analizar los datos provenientes de la red incluso si estos están incompletos o distorsionados. De igual manera es capaz de conducir un análisis sobre datos provenientes de superficies no-lineales.

La inherente velocidad del MLP es otro de sus benéficos, la detección de anomalías para la protección de los recursos de red requiere de rapidez para poder evitar daños que provoquen pérdidas en los sistemas. Sin embargo, lo más importante es su habilidad de aprender las características de los eventos anómalos y luego ser capaces de detectar anomalías nunca antes vistas.

Sus desventajas consisten en la necesidad de proveerla de un conjunto de datos bien seleccionado para su entrenamiento pues la habilidad de detectar eventos anómalos depende de su capacidad de aprender de las características de estos.

Por su parte el modo de funcionamiento de las técnicas de Clustering, posibilitan el reconocimiento de grupos de datos cuyo comportamiento sea muy diferente al resto de los datos y también cuando no se conoce o no se puede etiquetar de manera confiable los datos para su clasificación. Utilizando 2-means, luego de fijados los centroides y formados los clusters, cada instancia nueva pasará a formar parte de uno de los clusters según el grado de similitud.

1.7 Conclusiones parciales del capítulo.

La aplicación de técnicas de la Inteligencia Artificial, para la detección de eventos anómalos en redes de datos permite la detección proactiva de ataques, automatiza la búsqueda de nuevos patrones de anomalía y disminuye la complejidad de las tareas de administración de la seguridad en la red. Perceptrón Multicapas y Clustering, son dos técnicas ampliamente aplicadas en el campo de la detección de eventos anómalos, alcanzándose resultados relevantes a nivel internacional. En la Universidad de Cienfuegos, las trazas de los servicios y/o dispositivos de red almacenadas, son analizadas mediante

reglas predefinidas por los expertos. Lo anterior dificulta realizar procesamiento en línea (o detección proactiva de eventos anómalos) por el gran volumen de información, así como la detección de nuevos patrones. La propuesta del presente trabajo es que las trazas sean inicialmente clasificadas en dos grupos, anómalos y no anómalos, utilizando una de las técnicas inteligentes mencionadas. Luego, las trazas clasificadas como anomalías (en menor cantidad) serán procesadas mediante la reglas de expertos.

Capítulo 2: Aplicación del MLP y el Clustering al análisis de trazas.

En este capítulo se justifican las herramientas utilizadas durante el desarrollo de la investigación. Se definen los atributos seleccionados para el conjunto de entrenamiento y su estandarización. Además, se muestran los resultados alcanzados en el entrenamiento del MLP, para diferentes modos de evaluación, y los agrupamientos obtenidos mediante la técnica de clustering empleada.

2.1 Tecnologías y herramientas utilizadas para el desarrollo de la propuesta.

En este epígrafe se describen los elementos considerados para la selección de las tecnologías y herramientas que apoyaron el desarrollo de la investigación.

Python. (18) (19) (20)

Se escogió el lenguaje Python, debido a las grandes funcionalidades que brinda para el trabajo con ficheros de texto y por los módulos y paquetes que tiene incorporado para las operaciones en el campo de las redes. Además, Python es un idioma de programación dinámico que es usado en una gran variedad de dominios de la aplicación. Python se compara a menudo a Tcl, Perl, Ruby, o Java. Algunos de sus rasgos más importantes son:

1. Posee una sintaxis muy clara y legible.
2. Fuertes capacidades de introspección.
3. Orientado a objetos.
4. Sus expresiones son naturales, similares al lenguaje procedural.
5. Completamente modular, con soporte para herencia de paquetes.
6. Fuerte tratamiento a las excepciones.
7. Tipos de datos dinámicos de alto nivel.
8. Extensiones y módulos fácilmente escritos en C, C++ (o Java para Jython, idiomas de .NET para IronPython).

Python posee cientos de bibliotecas de apoyo que lo convierten en la solución para la mayoría de los tipos de problemas que se detecten en la red. Permite escribir el código necesario rápidamente.

Python está disponible para sistemas operativos como: Windows, Linux/Unix, OS/2 y Mac.

PostgreSQL.

El sistema gestor de base de datos seleccionado para el almacenamiento y procesamiento de las trazas, es PostgrSQL. PostgreSQL fue seleccionado con anterioridad en (19) atendiendo a las siguientes características:

- Alta concurrencia: Mediante un sistema denominado MVC (Acceso Concurrente Multiversión), PostgreSQL permite que mientras un proceso escribe en una tabla, otros accedan a la misma tabla sin necesidad de bloqueos.
- Amplia variedad de tipos nativos: PostgreSQL provee nativamente soporte para:
    1. Números de precisión arbitraria.
    2. Texto de largo ilimitado.
    3. Figuras geométricas (con una variedad de funciones asociadas)
    4. Direcciones IP (IPv4 e IPv6).
    5. Bloques de direcciones estilo CIDR.
    6. Direcciones MAC.
    7. Arrays.

Software Weka. (21)

Weka, acrónimo de Waikato Environment for Knowledge Analysis, es un entorno para experimentación de análisis de datos que permite aplicar, analizar y evaluar las técnicas más relevantes de análisis de datos, principalmente las provenientes del aprendizaje automático, sobre cualquier conjunto de datos del usuario. Es un software compuesto de un conjunto de librerías JAVA que ha sido desarrollado en la Universidad de Waikato (Nueva Zelanda) bajo licencia GPL, lo cual ha impulsado que sea una de las suites más utilizadas en el área durante los últimos años.

El paquete WEKA[] contiene una colección de herramientas de visualización y algoritmos para análisis de datos y modelado predictivo, unidos a una interfaz gráfica de usuario para acceder fácilmente a sus funcionalidades. Para ello únicamente se requiere que los datos a analizar se almacenen con un cierto formato, conocido como ARFF (Attribute-Relation File Format). Para éste estudio se ha utilizado su versión 3.4.12.

Los puntos fuertes de Weka son:

1. Está disponible libremente bajo la licencia pública general de GNU.
2. Es muy portable porque está completamente implementado en Java y puede correr en casi cualquier plataforma.
3. Contiene una extensa colección de técnicas para preprocesamiento de datos y modelado.

WEKA soporta varias tareas especialmente, preprocesamiento de datos, clustering, clasificación, regresión, visualización, y selección. Todas las técnicas de WEKA se fundamentan en la asunción de que los datos están disponibles en un fichero plano o una relación, en la que cada registro de datos está descrito por un número fijo de atributos (normalmente numéricos o nominales, aunque también se soportan otros tipos). WEKA también proporciona acceso a bases de datos vía SQL gracias a la conexión JDBC (*Java Database Connectivity*) y puede procesar el resultado devuelto por una consulta hecha a la base de datos.

2.2 Conjunto de datos para el aprendizaje, evaluación y clasificación.

A la hora de entrenar el MLP y los Cluster, se necesita un conjunto de datos suficientemente representativo. Una vez construido los modelos, se debe evaluar su precisión mediante la tasa de aciertos, el número de falsos positivos y el de falsos negativos. Las trazas son gestionadas por la base de datos implementada en el *Sistema para la centralización, análisis y procesamiento de trazas de servicios telemáticos para la Red UCf* (19). La base de datos cuenta con una tabla para cada servicio y/o dispositivo de red, las cuales están conformadas por los campos que componen las cabeceras de las trazas generadas.

En el presente trabajo se analizaron las trazas provenientes de los dispositivos CISCO de las series 3500, 2800, 2600 y 2500. La base de datos posee una tabla generalizadora, la cual recoge las trazas generadas por estos dispositivos, conteniendo los siguientes campos:

- Número Consecutivo.
- Tiempo.
- Cadena.
- Estado.
- Protocolo.

- IP-Origen.
- IP-Destino.
- Cantidad de paquetes.
- Fecha-Hora.
- Puerto-Origen.
- Puerto-Destino.
- Fracción.
- Id-Cisco.
- VLAN-Origen.
- VLAN- Destino.

Para el conjunto de entrenamiento, sólo se tuvo en cuenta los campos siguientes, que según criterios de expertos son los que muestran evidencia en caso de la ocurrencia de algún evento anómalo:

- Estado.
- Protocolo.
- IP-Origen.
- IP-Destino.
- Puerto-Origen.
- Puerto-Destino.
- Cantidad de paquetes.
- Clasificación.

La etiqueta de clasificación es necesaria especificarla para el análisis posterior de los falsos positivos y los falsos negativos (22), y específicamente en el MLP, para su proceso de aprendizaje.

2.2.1 Estandarización de los datos.

Los valores posibles en los que se puede presentar cada campo a analizar son los siguientes:

Atributos	Posibles valores
Estado	denied, permitted
Protocolo	tcp, udp, icmp, igmp
IP-Origen	0.0.0.0, 255.255.255.255, de la 127.0.0.0 a la 127.255.255.255,10.0.0.0/8,192.168.0.0/16, 172.16.0.0/12,224.0.0.0/3,169.254.0.0/16,192.0.2.0/24,Las direcciones IP públicas de la red (en el caso de la UCf son las siguientes: 200.55.186.32/28 y 200.14.50.192/27)
IP-Destino	0.0.0.0, 255.255.255.255, de la 127.0.0.0 a la 127.255.255.255
Puerto-Origen	0 a 1023
Puerto-Destino	1080,2283, 2535, 2745, 3127, 3128, 3410, 5554, 8866, 8800-8900, 9898, 10000, 100080, 12345, 17300, 27374, 65506.
Cantidad de Paquetes	Número entero.

Para la estandarización de los datos se realizaron scripts. Para esta investigación se utilizó la tecnología de programación Python.

Los datos fueron estandarizados según (23), con valores entre 0 y 1:

- Estado: se asignó 1 en caso de permitted y 0, para denied.
- Protocolo: se tomaron para representar las variantes posibles de este campo 4 números: 0 para TCP, 0.333333 para UDP, 0.666666 para ICMP y 1 para IGMP.
- IP-Origen e IP-Destino: todos los números IP fueron divididos entre el mayor número IP que puede presentarse, 255.255.255.255.
- Puerto-Origen y Puerto-Destino: todos los números de puertos fueron divididos entre el mayor número de puerto que puede presentarse 65536.
- Cantidad de paquetes: se asignó 0 para cuando la cantidad de paquetes es menor que 30 y 1 para cantidades mayores.

2.3 Preparación del conjunto de entrenamiento.

Se debe tener en cuenta que la herramienta Weka tiene que manejar grandes conjuntos de datos (21) y que al usar la máquina virtual de Java se dispone de una memoria limitada. Por tal motivo, se seleccionó un conjunto de datos representativo como para poder hacer un análisis que se considere suficientemente bueno, y que a la vez sea lo necesariamente pequeño para que Weka pueda manejarlo con cierta soltura sin llegar a colapsar el equipo.

Se confeccionó un conjunto de entrenamiento con un total de 15 820 instancias, a partir de un muestro realizado por los expertos de las trazas almacenadas en el sistema. Una muestra en estadística consiste en un conjunto de individuos extraídos de una población con el fin de inferir, mediante su estudio, características de toda la población (24). Se dice que una muestra es representativa cuando, por la forma en que ha sido seleccionada, aporta garantías suficientes para realizar inferencias fiables a partir de ella.

Para obtener conclusiones fiables a partir del conjunto de entrenamiento, se tuvo en cuenta su tamaño, así como las instancias que lo componen. De las trazas seleccionadas para el entrenamiento se etiquetaron como anómalas, 2 177 y no a anómalas, 7 315.

2.3.1 Formato del fichero Weka.

Después de seleccionados los datos de entrenamiento, se transformaron a formato Weka. Los datos de entrada sobre los que operarán los algoritmos, deben estar codificados en un formato denominado Attribute-Relation File Format (extensión "arff"). El formato de un fichero arff sigue la estructura siguiente (23):

@relation NOMBRE_RELACION

@attribute at1 tipo

@attribute at2 {valor1, valor2, …}

…

@attribute atN {valor1, valor2, …}

@data

dato11,dato21...,datoN1

…

dato1M,dato2M...,datoNM

Los atributos pueden ser principalmente de dos tipos: numéricos de tipo real o entero (real o integer), y simbólicos (especificando los valores posibles que pueden tomar entre llaves.

En este caso, los atributos de tipo continuo se definieron de tipo integer y los discretos de tipo booleano (toman valores 0 ó 1).

Aplicando el formato ARFF a las instancias del conjunto de entrenamiento se obtuvo el siguiente fichero:

@RELATION Clasificar

@ATTRIBUTE estado INTEGER

@ATTRIBUTE protocolo NUMERIC

@ATTRIBUTE ip_origen NUMERIC

@ATTRIBUTE ip_destino NUMERIC

@ATTRIBUTE puerto_origen NUMERIC

@ATTRIBUTE puerto_destino NUMERIC

@ATTRIBUTE cantidad_paquetes INTEGER

@ATTRIBUTE class {Normal, Anormal}

@DATA

1,0.333333,0.00044753287013,0.000783745611036,0,0.47917175293,0.976150512695,Anormal

1,0.333333,0.000446737838506,0.000783745611036,0,0.245452880859,0.976150512695,Anormal

…

2.3.2 Entrada de datos de entrenamiento al sistema.

Una vez condicionados los datos en formato Weka, se introduce el fichero, para ser procesado por los métodos elegidos, luego de abierto el fichero la aplicación brinda toda la información del mismo: Relación, Instancias, Atributos y cantidad de instancias pertenecientes a cada clase a clasificar (Figura).

Relación: Clasificar.

Instancias: 9492.

Atributos: 8 (estado, protocolo, ip-origen, ip-destino, puerto-origen, puerto-destino, cantidad de paquetes y clase).

Instancias No Anómalas: 7 315.

Instancias Anómalas: 2 177.

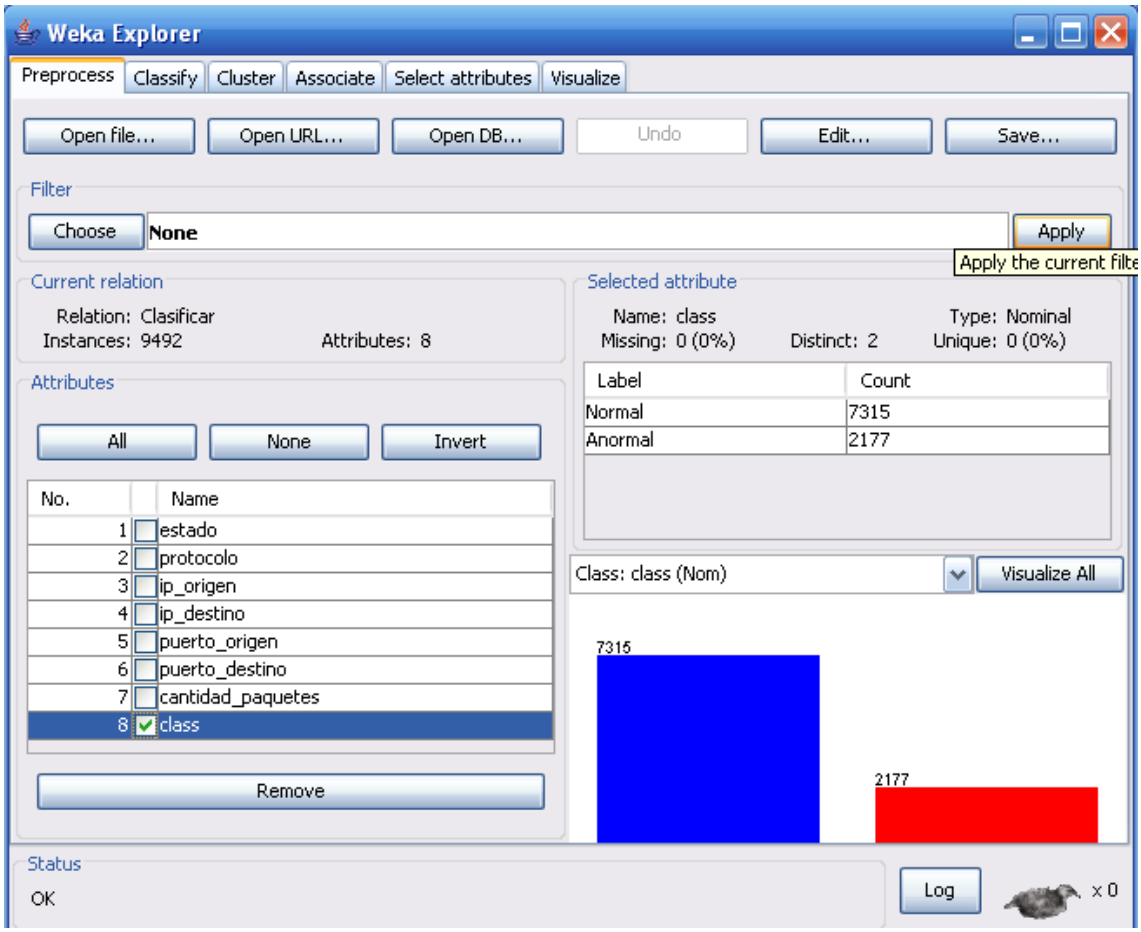

Figura 6 : Ambiente Weka luego de ingresados los datos.

2.4 Resultados del entrenamiento y la evaluación del MLP. Resultados del agrupamiento.

En el presente epígrafe se definen los modos de evaluación que permiten medir la efectividad de las técnicas MLP y Clustering. Se muestran los resultados alcanzados en el entrenamiento del MLP, a partir de la variación de parámetros de entrenamiento, así como, los agrupamientos obtenidos mediante la técnica de clustering empleada.

2.4.1 Modos de evaluación.

Para medir la efectividad de la clasificación es necesario comparar la clase predicha con la clase real de las instancias. Se utilizaron diversos modos para realizar esta evaluación:

Use training set: evaluación del clasificador sobre el mismo conjunto con el que se construye el modelo predictivo para determinar el error, que en este caso es denominado *error de resustitución*.

Supplied test set: evalúa sobre un conjunto independiente. Permite cargar un conjunto nuevo de datos. Sobre cada dato se realizará una predicción de clase para contar los errores.

Cross-Validation: evaluación con validación cruzada. Se dividen las instancias en tantas carpetas como indica el parámetro 'Folds', y en cada evaluación se toman las instancias de cada carpeta como datos de prueba, y el resto como datos de entrenamiento para construir el modelo. Los errores calculados serán el promedio de todas las ejecuciones

Percentage split: se dividen los datos en dos grupos, de acuerdo con el porcentaje indicado (%). El valor indicado es el porcentaje de instancias para construir el modelo, que seguidamente es evaluado sobre el resto de las instancias.

2.4.2 Parámetros a considerar para el entrenamiento del MLP.

Las opciones que pueden ser seleccionadas antes de iniciar el entrenamiento del MLP son las siguientes:

GUI: permite que la aplicación dibuje la red neuronal.

autoBuild: autoconstruye la red neuronal (cantidad de capas ocultas y cantidad de neuronas en cada una de estas).

debug: muestra información adicional en consola.

hiddenLayers: cantidad de neuronas por cada capa oculta que se adiciona a la red.

learningRate: tasa de aprendizaje.

momentum: impulso aplicado a los pesos durante la actualización

nominalToBinaryFilter: procesa las instancias con el filtro. Ayuda a mejorar el rendimiento de la red si hay atributos nominales en los datos.

normalizeAttributes: normaliza los atributos. Ayuda a mejorar el rendimiento de la red. Para esto la clase no tiene que ser numérica, también normaliza atributos nominales (valores entre -1 y 1).

normalizeNumericClass: normaliza la clase si ésta es numérica. Ayuda a mejorar el rendimiento de la red. Normaliza la clase entre -1 y 1. Este es un proceso interno, los valores mostrados son dados en los rangos originales.

randomSeed: inicializa un número aleatorio el cual es usado para ubicar los pesos iniciales en las conexiones entre los nodos, así como para barajar los datos de entrenamiento.

reset: permite a la red neuronal reiniciarse con un bajo ritmo de entrenamiento. Si la red diverge de la respuesta automáticamente se reinicia la red con un bajo ritmo de entrenamiento y comienza el entrenamiento de nuevo. Esta opción es factible sólo si la interface gráfica no está activada.

trainingTime: ritmo de entrenamiento

validationThreshold: umbral

La clase de los datos a clasificar es numérica, la configuración de los parámetros se fijó como se indica a continuación:

- GUI: *true*
- autoBuild: *true*
- debug: *false*
- hiddenLayers: *a*
- nominalToBinaryFilter: *false*
- normalizeAttributes: *true*
- normalizeNumericClass: *trae*
- validationThreshold: *20*
- randomSeed: *0*
- reset: *true*

Los parámetros, learningRate, momentum y trainingTime, se variaron con el fin de comparar los rendimientos del MLP.

2.4.3 Entrenamientos del MLP, variando sus parámetros.

En el presente subepígarfe se muestran los rendimientos del MLP, alcanzados a partir de la variación de sus parámetros. A continuación se identifican las etiquetas utilizadas en las tablas comparativas:

1. learningRate.

2. momentum.
3. trainingTime.
4. validationThreshold.
5. Instancias correctamente clasificadas.
6. % Instancias correctamente clasificadas.
7. Instancias incorrectamente clasificadas.
8. % Instancias incorrectamente clasificadas.
9. Means absolute error.
10. Root mean squared error.
11. Relative absoluted error (%)
12. Root relative squared error (%).

Para el entrenamiento, los parámetros se variaron siguiendo el algoritmo planteado a continuación, el resultado es señalado en las tablas comparativas:

1) se fijó **1** según el valor por defecto de la aplicación, se debe tener en cuenta que este parámetro se debe aumentar a medida que disminuya el error de la red durante la fase de aprendizaje con el fin de acelerar la convergencia, pero sin llegar nunca a valores demasiado grandes.
2) luego, se comenzó a variar **2** desde 0.1 hasta 0.9.
3) después de obtenidas todas las combinaciones de 1) y 2), se fijó **2** según el mejor resultado.
4) se comenzó a variar **1** desde 0.1 hasta 0.9.
5) se seleccionó el mejor rendimiento para cada modo de evaluación.

Si la red deja de aprender antes de llegar a una solución aceptable, se realiza un cambio en el número de neuronas ocultas o en los parámetros de aprendizaje.

1. Modo de Evaluación: Percentage Split 60% (3797 trazas para entrenamiento y 5695 trazas para test)												
Modelo	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
1.1	0.3	0.1	500	20	3637	95.7861	160	4.7861	0.06	0.20	16.47	48.48
1.2	0.3	0.3	500	20	3637	95.7861	160	4.2139	0.06	0.20	16.09	48.40

1.3	0.3	0.5	500	20	3640	95.8652	157	4.1348	0.05	0.20	14.77	47.91
1.4	0.3	0.9	500	20	3646	96.3656	151	3.9768	0.05	0.19	18.15	46.23
1.5	0.5	0.9	500	20	3659	96.3656	138	0.6344	0.04	0.18	11.68	45.30
1.6	0.9	0.9	500	20	3635	95.7335	162	4.2665	0.04	0.20	12.14	47.12
1.7	0.1	0.9	500	20	3653	96.2075	144	3.7925	0.06	0.19	17.36	47.12

Tabla 1: Resultados de los entrenamientos realizados aplicando el modo de evaluación Percentage Split, para definir los parámetros 1 y 2

2. Modo de Evaluación: Cross-validation Fold: 10 (8543 trazas para entrenamiento y 949 trazas para test)												
Modelo	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
2.1	0.3	0.1	500	20	9141	96.8443	347	3.6557	0.06	0.18	16.54	43.51
2.2	0.3	0.3	500	20	9140	96.2916	352	3.7084	0.06	0.18	16.62	43.76
2.3	0.3	0.5	500	20	9141	96.3021	351	3.6979	0.06	0.18	16.73	43.83
2.4	0.3	0.9	500	20	9145	96.3443	347	3.6557	0.06	0.18	15.64	44.48
2.5	0.5	0.9	500	20	9142	96.3127	350	3.6873	0.04	0.19	12.18	44.71
2.6	0.1	0.9	500	20	9140	96.2916	352	3.7084	0.06	0.18	17.68	44.15
2.7	0.1	0.5	500	20	9146	96.3548	346	3.6452	0.06	0.18	16.56	43.74
2.8	0.1	0.3	500	20	9145	96.3443	347	3.6667	0.06	0.18	17.64	43.76
2.9	0.5	0.5	500	20	9144	96.3338	348	3.6662	0.06	0.18	17.64	43.50
2.10	0.5	0.5	500	20	9144	96.3338	348	3.6662	0.06	0.18	17.64	43.50
2.11	0.5	0.1	500	20	9136	96.2495	356	3.7505	0.06	0.18	16.71	44.06

Tabla 2: Resultados de los entrenamientos realizados aplicando el modo de evaluación Cross-validation, para definir los parámetros 1 y 2

3. Modo de Evaluación: Suplied test set. (6328 trazas para test)												
Modelo	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
3.1	0.3	0.1	500	20	6247	98.72	81	1.28	0.04	0.11	15.84	37.26
3.2	0.3	0.3	500	20	6248	98.7358	80	1.2642	0.04	0.11	15.44	37.05
3.3	0.3	0.5	500	20	6248	98.7358	80	1.2642	0.04	0.11	14.34	36.73
3.4	0.3	0.9	500	20	6260	98.9254	68	1.0746	0.01	0.10	4.37	33.84
3.5	0.1	0.9	500	20	6247	98.72	81	1.28	0.03	0.11	13.76	36.93
3.6	0.5	0.9	500	20	6261	98.9254	67	1.0588	0.02	0.10	9.88	34.09
3.7	0.9	0.9	500	20	6264	98.9886	64	1.0114	0.01	0.10	5.97	33.27
3.8	0.1	0.5	500	20	6246	98.7042	82	1.2958	0.01	0.10	5.97	33.27

3.9	0.5	0.5	500	20	6246	98.70.42	82	1.2858	0.03	0.11	12.89	36.97

Tabla 3: Resultados de los entrenamientos realizados aplicando el modo de evaluación Suplied test set, para definir los parámetros 1 y 2

4. Modo de Evaluación: Use training set.												
Modelo	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
4.1	0.3	0.1	500	20	9143	96.3232	349	3.6368	0.06	0.19	18.04	44.67
4.2	0.3	0.3	500	20	9143	96.3232	349	3.6768	0.06	0.19	17.76	44.62
4.3	0.3	0.5	500	20	9141	96.3021	351	3.6979	0.06	0.18	17.09	44.63
4.4	0.3	0.9	500	20	9164	96.5445	351	3.4555	0.03	0.18	10.24	43.92
4.5	0.5	0.9	500	20	9165	96.555	327	3.445	0.04	0.18	13.94	43.51
4.6	0.9	0.9	500	20	9160	96.5023	332	3.4977	0.04	0.18	11.87	43.71
4.7	0.1	0.9	500	20	9141	96.3021	351	3.6979	0.06	0.18	16.23	44.74

Tabla 4: Resultados de los entrenamientos realizados aplicando el modo de evaluación Use training set, para definir los parámetros 1 y 2

Luego de seleccionado el mejor rendimiento para cada modo de evaluación, se realizaron pruebas para definir el parámetro **3** mediante el siguiente algoritmo:

1) Manteniendo los valores de los parámetros **1** y **2** se comenzó a variar **3** desde 100 hasta 1000.
2) En caso de que el rendimiento de la red se mantenga constante para una variación de **3,** se termina el muestreo.
3) Se seleccionó el mejor modelo.

1. Modo de Evaluación: Percentage Split 60% (3797 trazas para entrenamiento y 5695 trazas para test) Modelo 1.5										
Modelo	**3**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**Tiempo Entrenamiento**
1.5.1	100	3654	96.2339	143	3.7671	0.05	0.19	14.21	45.63	35 seg
1.5.2	150	3655	96.2602	142	3.7398	0.04	0.19	12.55	45.91	50 seg
1.5.3	250	3649	96.1022	148	3.8978	0.04	0.19	12.33	46.70	80 seg.
1.5.4	400	3637	95.7861	160	4.2139	0.04	0.20	12.43	48.91	125 seg
1.5.5	500	3659	96.3656	138	3.6344	0.04	0.18	11.48	45.30	155 seg
1.5.6	700	3660	96.3919	137	3.6081	0.04	0.18	12.57	45.23	285 seg
1.5.7	1000	3656	96.2865	141	3.7135	0.04	0.19	12.89	45.72	495 seg

Tabla 5: Resultados de los entrenamientos realizados aplicando el modo de evaluación Percentage Split, para definir el parámetro 3.

2. Modo de Evaluación: Cross-validation Fold: 10 (8543 trazas para entrenamiento y 949 trazas para test) Modelo 2.7										
Modelo	**3**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**Tiempo Entrenamiento**
2.7.1	100	9124	96.1231	368	3.8769	0.06	0.18	17.63	44.45	210 seg.
2.7.2	150	9133	96.2179	359	3.7821	0.06	0.18	17.24	44.19	289 seg.
2.7.3	200	9141	96.3021	351	3.6979	0.06	0.18	17.03	44.00	365 seg.
2.7.4	300	9145	96.3443	347	3.6557	0.05	0.18	16.79	43.84	609 seg.
2.7.5	400	9147	96.3654	345	3.6346	0.05	0.18	16.65	43.78	12 min. 34 seg.
2.7.6	500	9146	96.3548	346	3.6452	0.05	0.18	16.43	43.74	17 min. 18 seg.
2.7.7	700	9147	96.3654	345	3.6346	0.05	0.18	16.48	43.69	23 min. 50 seg.

Tabla 6: Resultados de los entrenamientos realizados aplicando el modo de evaluación Cross-validation, para definir el parámetro 3.

Para este modo de evaluación las pruebas no continuaron desarrollando, para un aumento del parámetro **3** el comportamiento de la red se mantiene estable, sin grandes variaciones que se traduzcan en un aumento del rendimiento del MLP.

3. Modo de Evaluación: Suplied test set. (6328 trazas para test) Modelo 3.7										
Modelo	**3**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**Tiempo Entrenamiento**
3.7.1	100	6240	98.6094	88	1.3906	0.01	0.11	0.84	37.80	20 seg.
3.7.2	150	6254	98.8306	74	1.1694	0.02	0.10	7.52	35.45	28 seg.
3.7.3	250	6259	98.9096	69	1.0904	0.01	0.10	6.99	34.14	49 seg
3.7.4	400	6259	98.9096	69	1.0904	0.01	0.10	4.45	34.21	77 seg
3.7.5	500	6264	98.9886	64	1.0114	0.01	0.10	5.97	33.27	100 seg
3.7.6	700	6263	98.9728	65	1.0272	0.01	0.10	4.34	33.52	139 seg
3.7.7	1000	6247	98.72	81	1.28	0.04	0.11	16.97	37.56	205 seg

Tabla 7: Resultados de los entrenamientos realizados aplicando el modo de evaluación Suplied test set, para definir el parámetro 3.

4. Modo de Evaluación: Use training set. Modelo 4.5										
Modelo	**3**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**Tiempo Entrenamiento**
4.5.1	100	9151	96.4075	341	3.5925	0.04	0.18	11.42	44.72	20 seg.
4.5.2	150	9166	96.5655	326	3.4345	0.04	0.18	11.34	43.93	30 seg.
4.5.3	250	9165	96.555	327	3.445	0.04	0.18	13.26	43.74	50 seg.
4.5.4	400	9165	96.555	327	3.445	0.04	0.18	12.49	43.70	80 seg.
4.5.5	500	9165	96.555	327	3.445	0.04	0.18	13.94	43.51	100 seg.
4.5.6	700	9165	96.555	327	3.445	0.04	0.18	13.87	43.45	140 seg.

Tabla 8: Resultados de los entrenamientos realizados aplicando el modo de evaluación Use training set, para definir el parámetro 3.

El modo de evaluación Use training set presenta el mismo rendimiento de la red para un aumento del parámetro **3**, por lo cual se decidió no seguir la prueba.

2.4.4 Clustering, utilizando 2-means.

Para aplicar la técnica de cluster particional, se tomaron inicialmente 9492 instancias del conjunto de entrenamiento. Se determinaron 2 agrupaciones:

Anómalos y No Anómalos. En la (Figura 9) se muestra el resultado del agrupamiento.

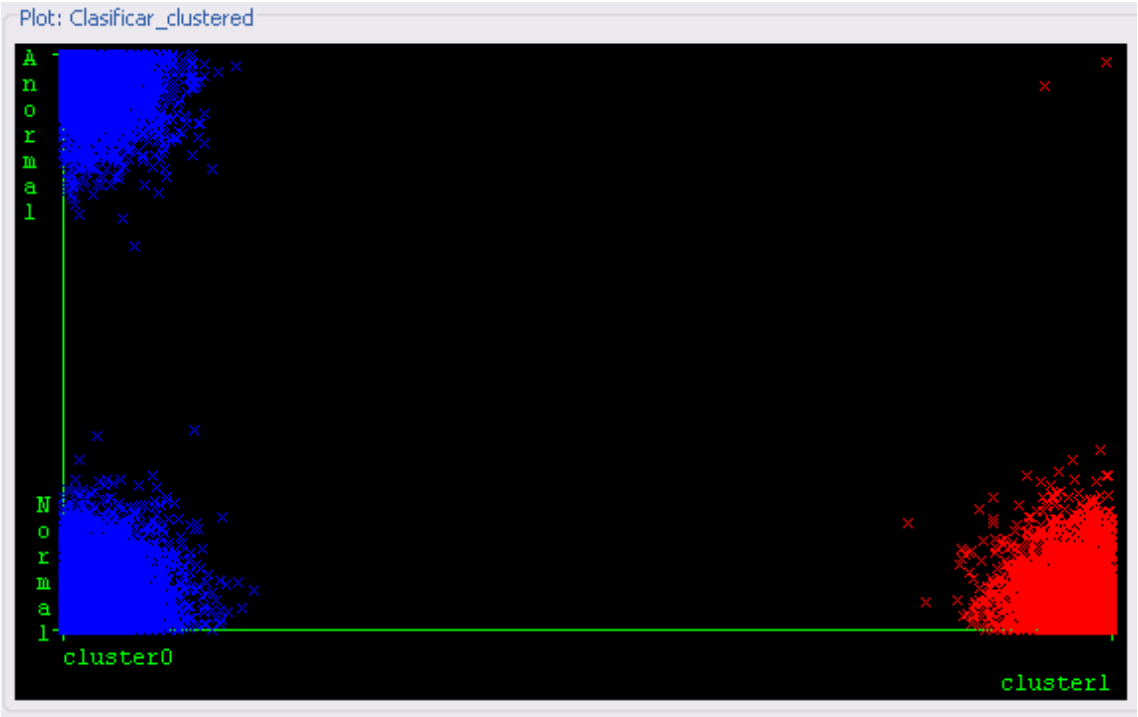

Figura 7: Cluster creados con 9492 instancias de entrenamiento.

Visualmente, se pudo comprobar que el agrupamiento no es bueno. Las instancias azules en la esquina superior izquierda son falsos positivos (falsas alarmas) y las instancias rojas en la esquina inferior derecha son falsos negativos (ataques no detectados).

Se decidió tomar una muestra más para obtener un mejor agrupamiento. El tamaño de la muestra depende de la precisión que se quiera conseguir en la estimación que se realice a partir de ella. Para su determinación se requieren técnicas estadísticas superiores, pero resulta sorprendente cómo, con muestras notablemente pequeñas, se pueden conseguir resultados suficientemente precisos.

Se realizaron pruebas con Weka para determinar con qué número de instancias la técnica de clúster obtenía mejores resultados, las pruebas fueron hechas con conjuntos seleccionados mediante un muestreo no probabilístico intencional (24); donde no todos los sujetos de la población tienen la misma probabilidad de ser elegidos, este tipo de muestreo se caracteriza por un esfuerzo deliberado de obtener muestras “representativas” mediante la

inclusión en la muestra de grupos supuestamente típicos, con el fin de obtener buenas clasificaciones. Se fue reduciendo el número de instancias hasta alcanzar con 150 instancias (100 No Anómalas y 50 Anómalas) un agrupamiento satisfactorio.

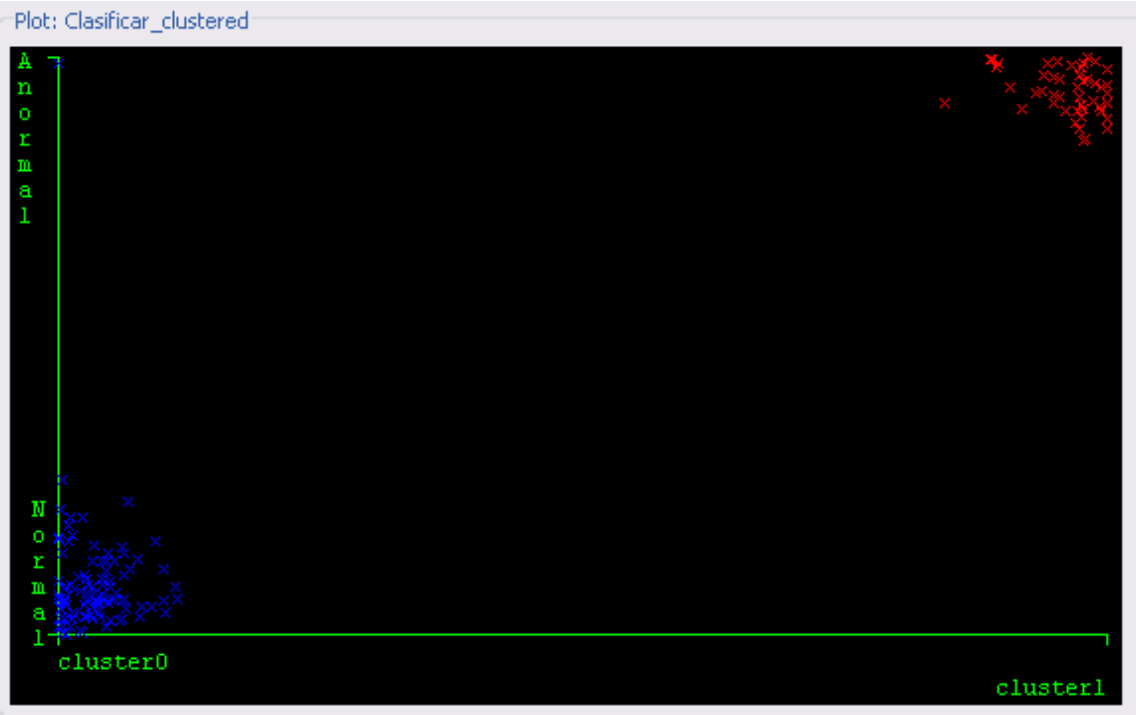

Figura 8: Cluster creados con 150 instancias de entrenamiento.

Los resultados obtenidos fueron los siguientes:

1. Number of iterations: 4
2. Within cluster sum of squared errors: 85.1191449679144
3. Instancias cluster 0 (Normales): 101
4. Instancias cluster 1 (Anormales): 49
5. Falsos positivos: 0
6. Falsos negativo: 1

Los resultados muestran que la clasificación fue buena, sólo se clasificó mal una instancia.

Luego de aplicar el 2-means al conjunto de 150 trazas se fijaron los centroides. Utilizando el modo de evaluación Suplied test set que permite evaluar en un conjunto independiente, se clasificarán 9 492 instancias.

2.5 Conclusiones parciales del capítulo.

En el capítulo se describieron los procedimientos para la selección y estandarización de los datos para el conjunto de entrenamiento, evaluación y clasificación, transformándose en valores continuos entre 0 y 1. Se realizaron entrenamientos del MLP, utilizando Weka, variando los parámetros de entrenamiento y modos de evaluación, observándose en el mejor de los casos rendimientos por encima del 96%. Se realizó la técnica de agrupamiento k-means, para diferentes tamaños de conjuntos, observándose los mejores resultados con 150 instancias.

Capitulo 3: Análisis de los resultados.

En el presente capítulo se describen los resultados del MLP y Cluster alcanzados, luego de realizar un análisis del rendimiento y aplicar selección de atributos, comparando antes y después de la selección. Además, se realiza una comparación de los modelos alcanzados con los casos tipos estudiados.

3.1- Propuestas de modelos para la detección de anomalías en la red de datos de la UCf.

3.1.1- RNA

Después de analizado el funcionamiento y los resultados obtenidos por cada uno de los modos de evaluación aplicados para medir el rendimiento de la RNA, se decidió seleccionar el entrenamiento realizado con **Cross-Validation**. Éste modo de evaluación permite obtener resultados más reales, su funcionamiento consiste en dividir de manera aleatoria el conjunto de instancias dadas para el entrenamiento según el número indicado en **Fold**, luego se selecciona uno de estos conjuntos para la prueba mientras que se entrena con el resto, esta operación se realiza hasta que se prueba en cada uno de los conjuntos que fueron divididos inicialmente. El error es el promedio de todas las pruebas realizadas.

Parámetros de entrenamiento.

Para este entrenamiento se fijó:

- Fold: 10.
- LearningRate: 0.1
- Momentum: 0.5
- TrainingTime: 400
- ValidationThreshold: 20

Arquitectura y algoritmo de entrenamiento.

Se aplicó una red de retropropagación con función de salida sigmoidal. La arquitectura de la red es la siguiente:

- 1 capa de entrada con 7 neuronas.
- 1 capa oculta con 4 neuronas.
- 1 capa de salida con 2 neuronas.

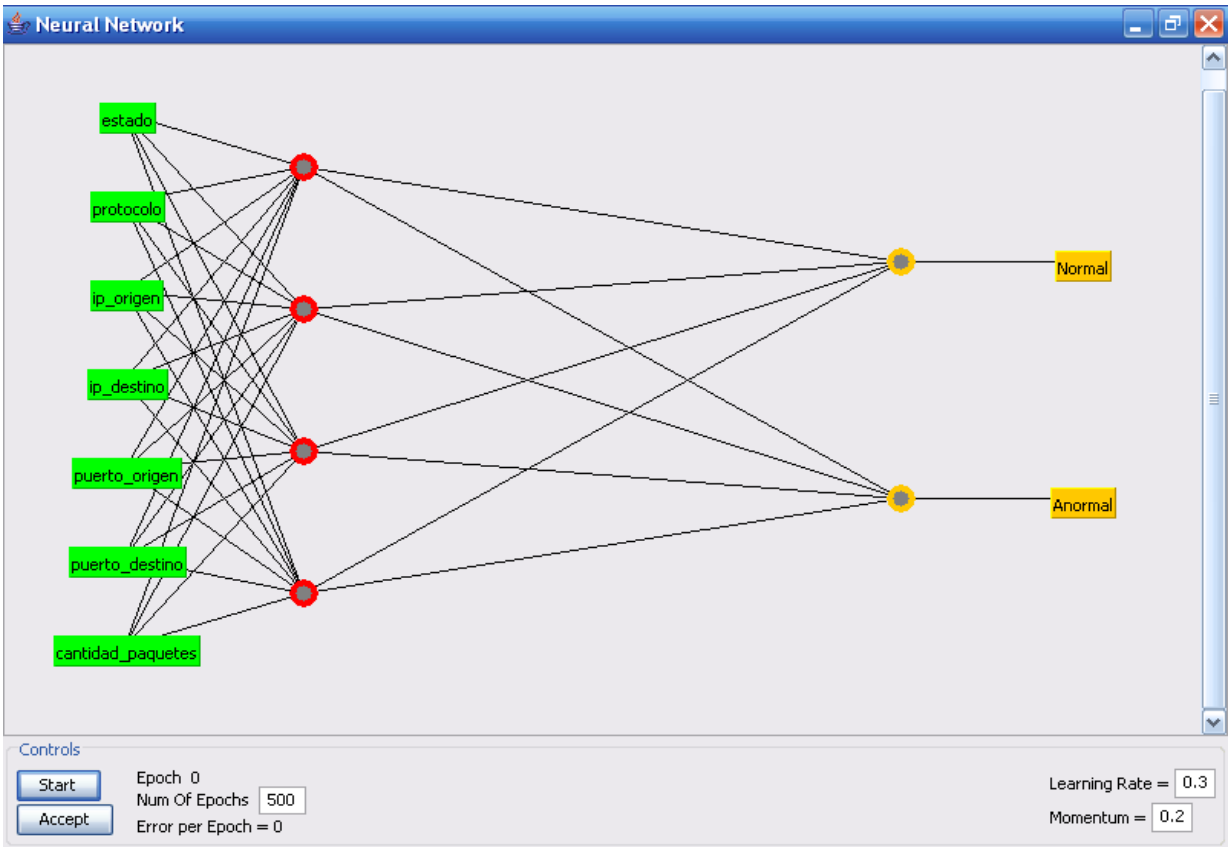

Figura 9: Arquitectura de la RNA aplicada para la detección de anomalías.

Estructura de las capas de la RNA y pesos asociados a las conexiones.

Capa de entrada:

La capa de entrada está compuesta por 7 neuronas, cada una corresponde a un atributo de las trazas pertenecientes a los dispositivos CISCO. Los pesos asociados a las conexiones de esta capa con la capa oculta se muestran a continuación:

Pesos de las conexiones entre la capa de entrada y la capa oculta:

- Nodo 1 de la capa oculta:
  - threshold: 12.231084779131459
  - estado: 12.162546254083082
  - protocolo: -0.22442338029028266
  - ip_origen: 0.14694305945324704
  - ip_destino: -9.117226892025434
  - puerto_origen: 0.04132785843415271
  - puerto_destino: -2.2923110917618623
  - cantidad_paquetes: -19.21785429963397

- Nodo 2 de la capa oculta:
  - threshold: 3.5584469128807803
  - estado: 8.823582842680143
  - protocolo: -0.2869532377658395

- ip_origen: 2.6836922177127884
- ip_destino: -5.332577257507672
- puerto_origen: 0.0345402009268128
- puerto_destino: -1.036338617151403
- cantidad_paquetes: -7.466400222924925

- Nodo 3 de la capa oculta:
  - threshold: 14.82067201792343
  - estado: 14.57890620993206
  - protocolo: 0.00458080806355219
  - ip_origen: 0.42599344328010885
  - ip_destino: -10.86867976272326
  - puerto_origen: -0.023246533163788698
  - puerto_destino: -2.8536208359125848
  - cantidad_paquetes: -23.727004971439545

- Nodo 4 de la capa oculta:
  - threshold: 3.492437544206233
  - estado: 7.597318139081415
  - protocolo: -0.2786103877785631
  - ip_origen: 2.418763948541528

- ip_destino: -4.327308186844815
- puerto_origen: -0.032389400638914545
- puerto_destino: -0.6550579520675672
- cantidad_paquetes: -5.620412544762007

Capa oculta:

La capa oculta está compuesta por 4 neuronas. Los pesos asociados a las conexiones de esta capa con la capa de salida se muestran a continuación:

Pesos de las conexiones entre la capa oculta y la capa de salida:

- Nodo Normal de la capa de salida:
  - threshold: -12.887066758926053
  - Nodo 1: 5.646206548444634
  - Nodo 2: 1.8043303733899303
  - Nodo 3: 7.576431273758215
  - Nodo 4: 1.202245599486451

- Nodo Anormal de la capa de salida:
  - threshold: 12.892356105766588
  - Nodo 1: -5.664382068418689

- Nodo 2: -1.7890021698010474
- Nodo 3: -7.562635942349546
- Nodo 4: -1.2184559722032775

Capa de salida:

La capa de salida está compuesta por dos nodos, cada uno representa una de las clases a clasificar: *Normal* y *Anormal*.

Resultados obtenidos por la RNA:

Los resultados obtenidos por el conjunto de: arquitectura, algoritmo de entrenamiento y parámetros de entrenamiento, son los siguientes:

- Instancias correctamente clasificadas: 9147.
- % Instancias correctamente clasificadas: 96.3654.
- Instancias incorrectamente clasificadas: 345
- % Instancias incorrectamente clasificadas: 3.6346.
- Falsos positivos: 27.
- Falsos negativos: 318.

3.1.2- Clustering

Para la aplicación de la técnica de clustering los centroides fueron fijados por un conjunto de 150 instancias, luego se evaluaron 9492 instancias aplicando el modo de evaluación **Suplied test set,** el cual permite cargar un conjunto nuevo para realizar las pruebas. Luego de cargar el nuevo fichero, las instancias nuevas serán clasificadas según el centroide más cercano.

Los parámetros fijados para el agrupamiento fueron:

- numClusters: 2
- seed: 10

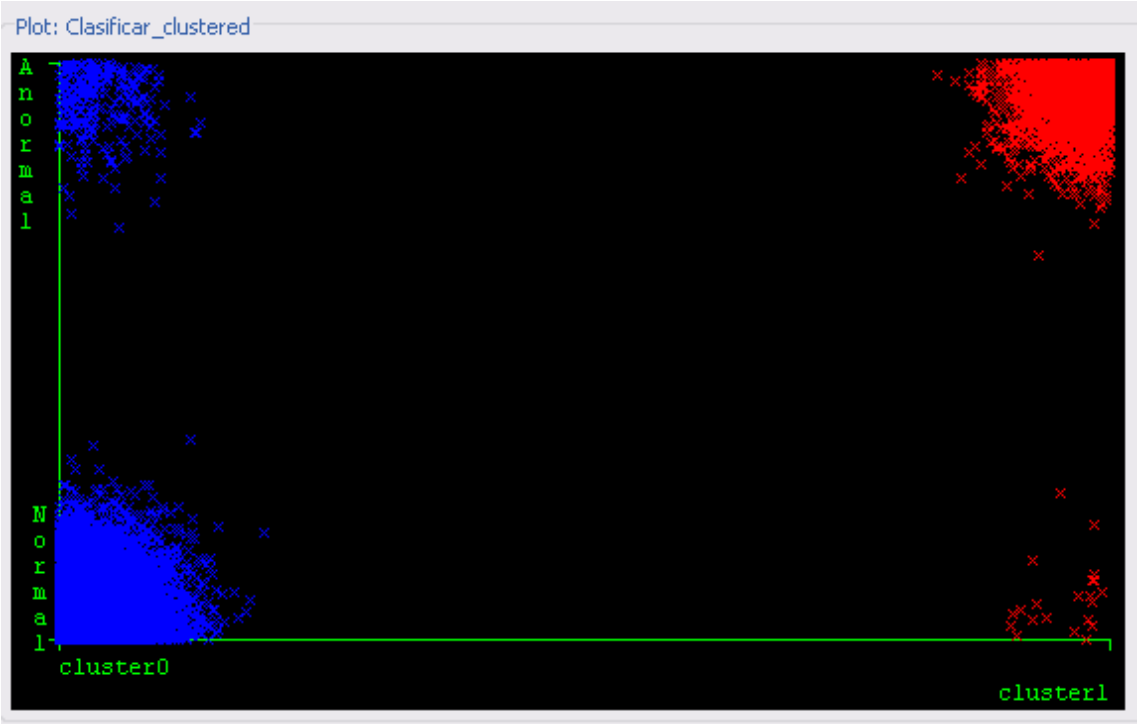

Figura 10: Clusters creados por la 9492 trazas de entrenamiento.

Centroides asociados a los clusters:

- Centroide para la agrupación de las instancias Normales:

Cluster 0

Mean/Mode:	1	0.2046	0.0006	0.0006	0	0.4113	0.1569	Normal
Std Devs:	0	0.1631	0.0003	0.0003	0	0.3755	0.27	N/A

- Centroide para la agrupación de las instancias Anormales:

Cluster 1

Mean/Mode:	1	0.3333	0.0005	0.0008	0	0.4268	0.9103	Anormal
Std Devs:	0	0	0.0002	0	0	0.2817	0.2313	N/A

Resultados obtenidos por la técnica de Cluster:

La técnica de cluster, aplicando el algoritmo de partición K-medias dio como resultado:

- Número de iteraciones: 4
- Suma de los errores cuadráticos intra-culster: 85.1191449679144
- Instancias Cluster 0 (Normales): 7562
- Instancias Cluster 1 (Anormales): 1930
- Instancias correctamente clasificadas: 9245.
- % Instancias correctamente clasificadas: 97.3283.
- Instancias incorrectamente clasificadas: 247.
- % Instancias incorrectamente clasificadas: 2.6717.
- Falsos positivos: 226.
- Falsos negativos: 21.

3.2- Selección de atributos.

Uno de los beneficios más significativos que poseen las RNA es su velocidad, motivo por el cual es de vital importancia explotar esta característica cuando el futuro de su aplicación es detectar anomalías en tiempo real.

La técnica de Clustering ubica en un espacio multidimensional cada instancia a partir de los valores de sus atributos, este espacio tendrá tantos ejes de coordenadas como atributos se tengan, es evidente que una reducción de estos facilitaría la aplicación de esta técnica.

Con el fin de aumentar el rendimiento de las técnicas inteligentes a aplicadas se utilizaron métodos de Selección de Atributos. Estos dan la posibilidad de conocer la influencia de cada uno de los atributos en la clasificación de las instancias como Normales o Anormales.

Para la Selección de Atributos se tuvo en cuanta tres de los métodos brindados por el software Weka:

1. OneAttributeEval.
2. RelieFAttributeEval.
3. SVMAttributeEval.

Resultados obtenidos:

Para la aplicación de los métodos de Selección de Atributos se aplicó el modo de evaluación Cross Validation con Fold igual a 10.

OneAttributeEval:

Los resultados obtenidos por la aplicación de este método son los siguientes:

average	**merit**	**average**	**rank**	**attribute**
96.628	+- 0.056	1	+- 0	7 cantidad_paquetes
93.221	+- 0.147	2	+- 0	3 ip_origen
88.514	+- 0.19	3	+- 0	4 ip_destino
83.622	+- 0.138	4	+- 0	6 puerto_destino
77.065	+- 0.005	5	+- 0	1 estado
77.065	+- 0.005	6	+- 0	5 puerto_origen
77.065	+- 0.005	7	+- 0	2 protocolo

RelieFAttributeEval:

average	**merit**	**average**	**rank**	**attribute**
0.225	+- 0.002	1	+- 0	4 ip_destino
0.142	+- 0.004	2.1	+- 0.3	3 ip_origen
0.133	+- 0.003	2.9	+- 0.3	6 puerto_destino
0.076	+- 0.001	4	+- 0	7 cantidad_paquetes
0.021	+- 0.002	5	+- 0	2 protocolo
0	+- 0	6	+- 0	5 puerto_origen
0	+- 0	7	+- 0	1 estado

SVMAttributeEval.

average	merit	average	rank	attribute
6.8	+- 0.4	1.2	+- 0.4	4 ip_destino
6.2	+- 0.4	1.8	+- 0.4	7 cantidad_paquetes
5	+- 0	3	+- 0	6 puerto_destino
4	+- 0	4	+- 0	3 ip_origen
2.9	+- 0.3	5.1	+- 0.3	1 estado
2.1	+- 0.3	5.9	+- 0.3	2 protocolo
1	+- 0	7	+- 0	5 puerto_origen

Los resultados obtenidos por cada uno de los Métodos de Selección de Atributos aplicados, muestran que los atributos estado, protocolo y puerto-origen son los de menor influencia en la clasificación de una instancia como Normal o Anormal.

Para las pruebas posteriores el conjunto de entrenamiento será transformado. Las trazas pertenecientes a los dispositivos de red CISCO almacenadas en la base de datos, fueron reducidas quedando para su análisis los siguientes campos:

- Ip-Origen.
- Ip-Destino.
- Puerto-Destino.
- Cantidad de Paquetes.
- Clasificación.

3.3- Resultados obtenidos por la aplicación de las técnicas inteligentes después de aplicar la selección de atributos.

3.3.1: RNA

Se experimentó con una RNA de diferente arquitectura (Figura 11), reduciéndose el número de neuronas de la capa de entrada en consecuencia con los resultados mencionados anteriormente, de igual manera se redujo el número de neuronas de la capa oculta.

Para los experimentos realizados se mantuvieron las características anteriores como son: algoritmo de entrenamiento, función de salida, parámetros de entrenamiento y Modo de Evaluación Cross-validation con Fold igual a 10, con la finalidad de poder comparar los resultados obtenidos por ambas RNAs.

Arquitectura de la RNA del experimento 2:

- 1 capa de entrada con 4 neuronas (se eliminaron las entradas de los atributos estado, protocolo, puerto-origen)
- 1 capa oculta con 3 neuronas.
- 1 capa de salida con 2 neuronas.

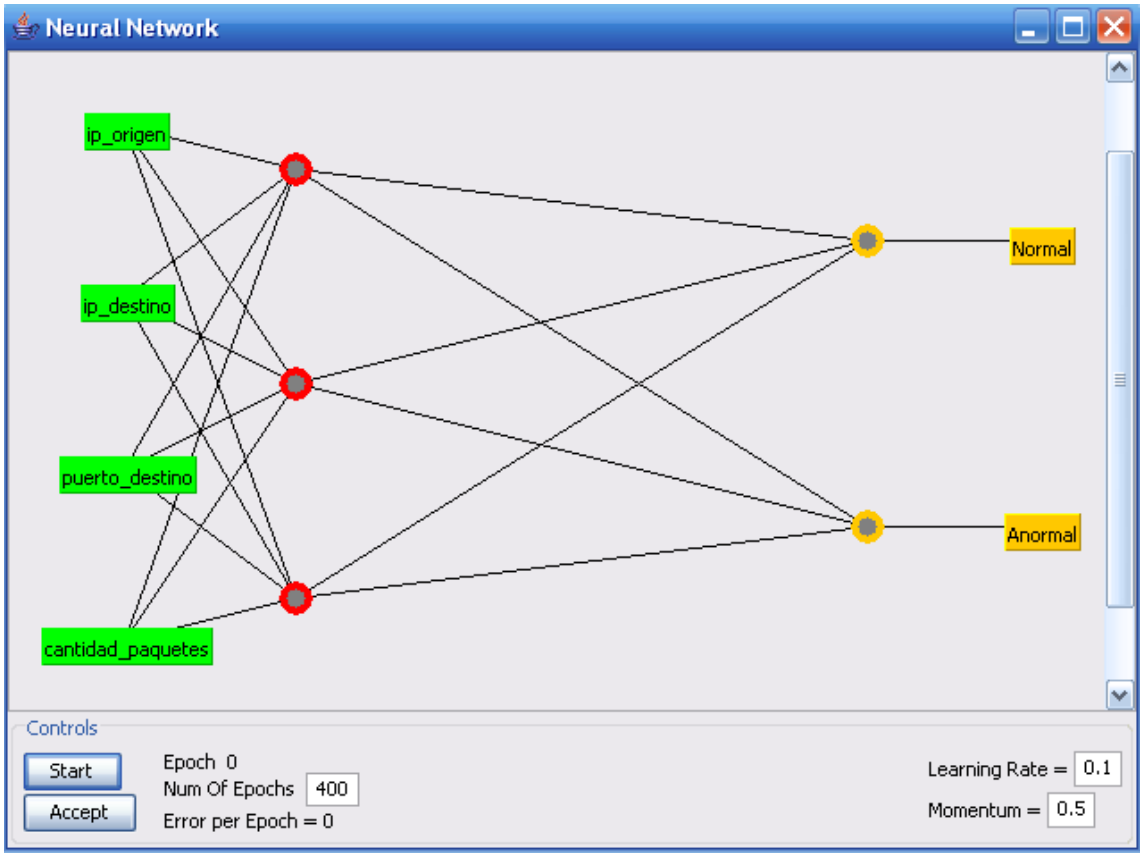

Figura 11: Arquitectura de la RNA del segundo modelo propuesto.

Resultados obtenidos por la RNA del experimento 2:

- Instancias correctamente clasificadas: 9137.
- % Instancias correctamente clasificadas: 96.26.
- Instancias incorrectamente clasificadas: 355
- % Instancias incorrectamente clasificadas: 3.74.
- Falsos positivos: 42.
- Falsos negativos: 313.

3.3.2- Clustering.

Para la aplicación de la técnica de clustering a las trazas luego de aplicado a estas la Selección de Atributos se realizó la misma operación que en las pruebas anteriores, primeramente se fijaron los centroides con un conjunto de

150 instancias, quedando estas correctamente agrupadas para un 100% de buena clasificación (Figura).

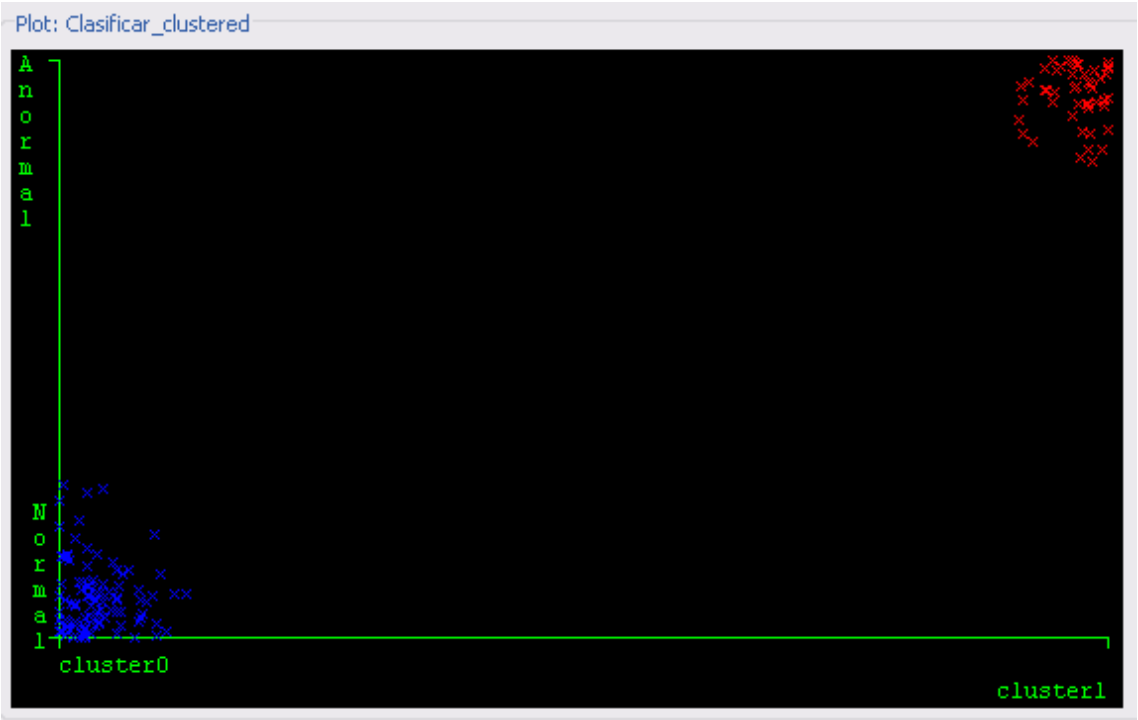

Figura 12: Clusters obtenidos por el conjunto de 150 instancias.

Luego, mediante el modo de evaluación Suplied test set se cargó un conjunto nuevo de 9492 instancias, para el cual se obtuvo los siguientes resultados:

Resultados obtenidos por la técnica de Cluster:

1. Number of iterations: 3
2. Within cluster sum of squared errors: 42.62465127934575.
3. Instancias cluster 0 (Normales): 7388.
4. Instancias cluster 1 (Anormales): 2104.
5. Instancias correctamente clasificadas: 9419.
6. % Instancias correctamente clasificadas: 99.231.
7. Instancias incorrectamente clasificadas: 73.
8. % Instancias incorrectamente clasificadas: 0.7690.

9. Falsos positivos: 73
10. Falsos negativos: 0

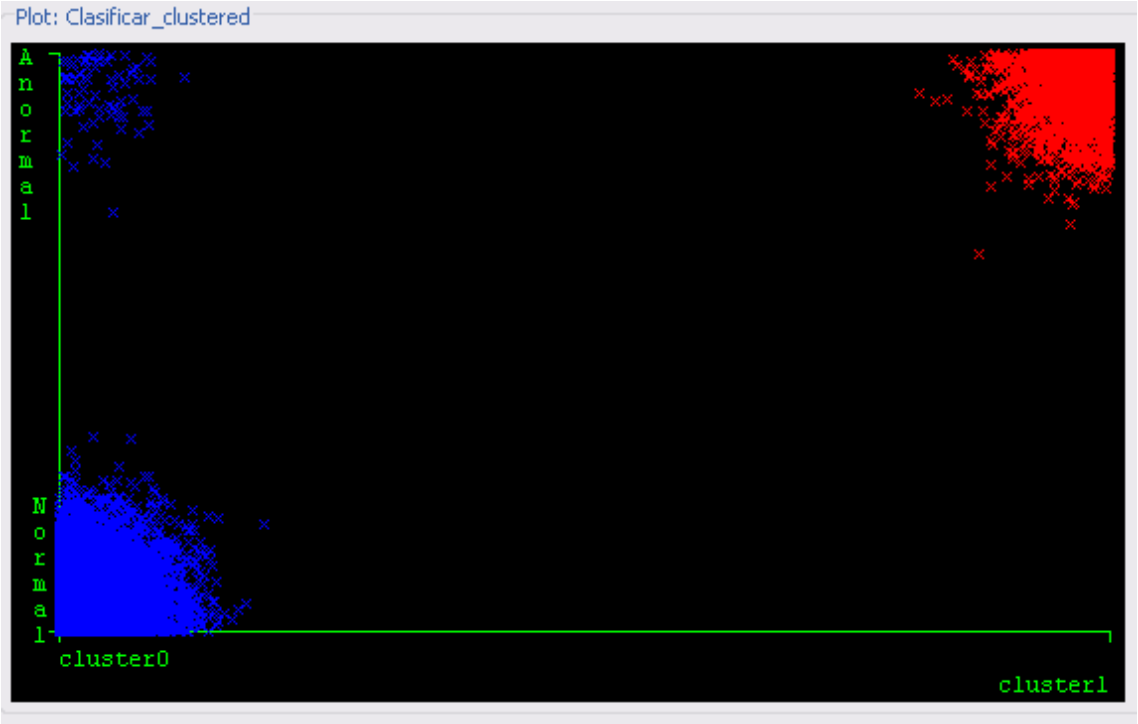

Figura 13: Clusters obtenidos por el conjunto de 9492 instancias

3.4-Comparación de los resultados obtenidos antes y después de la selección de atributos.

A continuación se realizará una comparación de los resultados obtenidos por la aplicación de las técnicas inteligentes a ambos grupos de datos, con la finalidad de determinar si una reducción del número de atributos a tener en cuenta para la clasificación de las instancias proporciona un mejor rendimiento de ambas técnicas.

3.4.1-Comparación de los resultados obtenidos por los modelos de RNAs.

En este sub-epígrafe se realizará un análisis de los resultados obtenidos por ambas RNAs, RNA propuesta como primer modelo (sub-epígrafe 3.2.1) y la RNA propuesta en el segundo modelo (sub-epígrafe 3.4.1).

A continuación se brinda una tabla con el resumen de los resultados obtenidos por cada uno de los modelos propuestos:

Parámetros	RNA (primer modelo)	RNA (segundo modelo)
Número de capas.	**3**	**3**
Número de capas ocultas.	**1**	**1**
Número de neuronas en la capa de entrada.	**7**	**4**
Número de neuronas en la capa oculta.	**4**	**3**
Número de neuronas en la capa de salida.	**2**	**2**
Instancias correctamente clasificadas.	**9147**	**9137**
% Instancias correctamente clasificadas.	**96.3654**	**96.26**
Instancias incorrectamente clasificadas.	**345**	**355**
% Instancias incorrectamente clasificadas.	**3.6346**	**3.74**
Falsos positivos.	**27**	**42**
Falsos negativos.	**318**	**313**

Tabla 9: Comparación de los modelos de RNAs propuestos.

Los resultados mostrados en la tabla anterior evidencian que con la eliminación de los atributos estado, protocolo y puerto-origen, la RNA no mejora notablemente su rendimiento, ni existen diferencias significativas entre la cantidad de falsos positivos y de falsos negativos.

3.4.2- Clustering.

En este sub-epígrafe se realizará una comparación de los resultados obtenidos por las técnicas de clustering según el primer y segundo modelos de cluster obtenidos (sub-epígrafes 3.2.2 y 3.4.2 respectivamente).

A continuación se brinda una tabla con el resumen de los resultados obtenidos:

Parámetros	Clustering (primer modelo)	Clustering (segundo modelo)
Número de iteraciones	**4**	**3**
Instancias correctamente clasificadas.	**9245**	**9419**
% Instancias correctamente clasificadas.	**97.3283**	**99.231**
Instancias incorrectamente clasificadas.	**247**	**73**
% Instancias incorrectamente clasificadas.	**2.6717**	**0.7690**
Falsos positivos.	**226**	**73**
Falsos negativos.	**21**	**0**

Tabla 10: Comparación de los modelos de clustering propuestos.

Los resultados mostrados en la tabla anterior evidencian que con la eliminación de los atributos estado, protocolo y puerto-origen de las instancias a clasificar, la técnica de clustering mejora su rendimiento notablemente y disminuye la cantidad de falsos negativos, reduciendo la cifra a cero, y la cantidad de falsos positivos.

Centroides asociados a los clusters:

- Centroide para la agrupación de las instancias Normales:

Cluster 0					
Mean/Mode:	0.0006	0.0006	0	0.1585	Normal
Std Devs:	0.0003	0.0003	0	0.2709	N/A

- Centroide para la agrupación de las instancias Anormales:

Cluster 0					
Mean/Mode:	0.0005	0.0008	0	0.8921	Normal
Std Devs:	0.0002	0	0	0.2626	N/A

3.5- Comparación con los casos tipo.

En este epígrafe se realizará un análisis comparativo entre los resultados obtenidos en la presente investigación y los casos tipos analizados en el estudio de los sistemas existentes.

Parámetros	Caso tipo.	Modelo propuesto.
Número de capas.	**4**	**3**
Algoritmo de entrenamiento.	**Retro-propagación**	**Retro-propagación**
Instancias para el entrenamiento	**8462**	**9492**
% de precisión.	**96**	**96,3654**
Root means saquared error.	**0.058**	**0.18**

Tabla 11: Comparación del modelo propuesto y el caso tipo. RNA.

Parámetros	Caso tipo.	Modelo propuesto.
Número clúster.	**C**	**2**
Métrica.	**Distancia Euclidiana**	**Distancia Euclidiana**
Instancias para el agrupamiento	**4000000**	**9492**
Modo de evaluación	**Cross-Validation, Fold = 10**	**Suplied test set**
% de precisión.	**56.25**	**99.231**
% de Falsos Positivos	**0.3**	**0**

Tabla 12 : Comparación del modelo propuesto y el caso tipo. Clustering.

Comparando la red neuronal del subepígrafe 3.2.1 y la propuesta en el caso tipo, se percibe que se alcanzan resultados similares.

No es factible comparar el caso tipo de cluster con la propuesta del presente trabajo. En el caso tipo no se fijan los números de cluster, un cluster puede llegar a ser un sólo tipo de ataque, por tanto n cluster representarían las anomalías. Utilizan el modo de evaluación Cross-validation, de ahí el porcentaje de clasificación que alcanza. Además, el conjunto de entrenamiento asciende a 4 000 000 de instancias.

3.6- Conclusiones parciales del capítulo.

Se describieron los modelos de MLP y Cluster alcanzados en el entrenamiento. El rendimiento de ambas técnicas inicialmente se comportó por encima del 96%, lo que en el caso del MLP es un resultado similar a los alcanzados internacionalmente. Aplicando algoritmos de selección de atributos, no se mejoran significativamente los resultados del MLP atendiendo a que la cantidad de falsos negativos se mantiene con valores similares. En el caso de Cluster, la reducción de atributos eleva el rendimiento a un 99,23% y reduce los falsos negativos a 0 y los falsos positivos a 73.

CONCLUSIONES

La investigación en su proceso dialéctico propicia una comprensión holística del objeto de estudio, que en nuestro caso ha permitido corroborar el empleo de las técnicas inteligentes MLP y Clustering para la detección de anomalías en redes de datos.

Se seleccionó un conjunto de entrenamiento, a partir de un muestreo no probabilístico intencional desarrollado por los expertos, para el aprendizaje, la evaluación y la clasificación.

A pesar de que con la aplicación de algoritmos de selección de atributos, se elevan notablemente los resultados de la técnica de Clustering, no es posible en esta etapa de la investigación excluir definitivamente los atributos de menor peso en la clasificación, siendo necesario para ello emplear una muestra mayor y más diversa de eventos anómalos.

Se propone clasificar las trazas según la menor distancia euclidiana a los centroides de los cluster obtenidos luego de aplicar los métodos de selección de atributos, en lugar de emplear MLP atendiendo a que presenta un elevado número de falsos negativos.

RECOMENDACIONES

En el transcurso de toda investigación científica y como resultado lógico de la solución de problemas científicos, existen determinadas variables de investigación que por ser fundamentales para el logro de los objetivos, tienen un seguimiento a partir del problema que genera sus relaciones; sin embargo, en muchas ocasiones la propia profundización en sus características delatan otras nuevas para las cuales no se tienen solución de inmediato y suelen poseer la importancia necesaria para constituirse en nuevos problemas científicos, lo que sin dudas es un resultado importante de cualquier trabajo de investigación.

Aun cuando se hayan obtenido los resultados esperados de una investigación se necesitan ciertos esfuerzos para poder implementarlos, en este sentido nos permitiremos proponer algunas recomendaciones que pueden contribuir a perfeccionar los resultados obtenidos en nuestro estudio y pueden abrir nuevas líneas de investigación:

Seleccionar una muestra mayor y más diversa de eventos anómalos a fin de obtener resultados más fiables en el empleo de los algoritmos de selección de atributos, logrando así disminuir la cantidad de falsos positivos y falsos negativos.

Implementar el algoritmo de clasificación propuesto como una funcionalidad del “Sistema para la centralización, análisis y procesamiento de trazas de servicios telemáticos para la Red UCf” y evaluar sus resultados.

Continuar con el desarrollo de la investigación a fin de aplicar técnicas inteligentes híbridas que permitan el análisis de secuencias de trazas en el tiempo que indican la sucesión de un evento anómalo.

REFERENCIAS BIBLIOGRÁFICAS

1. **Molina, Bonifacio Martín del Brío y Alfredo Sanz.** *Redes Neuronales y Sistemas Difusos.* Segunda Edición. Zaragoza, España : ALFAOMEGA, 2001.

2. **Dr. Bello, Rafael.** *Monografía de Redes Neuronales Artificiales. .* Centro de Estudios Informáticos, Universidad Central "Marta Abreu" de Las Villas.

3. **Elaine Rich, Kevin Knight.** *Inteligencia Artificial, Segunda Edición.* [ed.] Juan Stumpf. Segunda Edición. s.l. : McGRAW-HILL/INTERAMERICANA DE ESPAÑA, S.A, 1994.

4. **McCulloch, W. S., Pitts, W.** *A logical calculus of the ideas immanent in nervous activity.Bulletin of Mathematical.* 1943.

5. *Conferencia de la CUJAE.* Habana : s.n., 2004.

6. **Ortega, Urko Zurutuza.** *Estado del Arte. Sistema de Detección de Intrusos.* Departamento de Informática, Escuela Politécnica. Mondragón : s.n., 2004.

7. **Kruegel, Christopher, Toth, Thomas y Kirda, Engin.** Service Specific Anomaly Detection for Network Intrusion Detection. *In Proceedings of the Symposium on Applied Computing (SAC), ACM Press.* España : s.n., 2002.

8. *Neural Networks Applied in Intrusion Detection Systems. In Proceedings of the IEEE International Joint Conference on Neural Networks.* **Bonifacio Jr., J. M., Cansian, A. M., Carvalho, A. C. P. L. F., and Moreira.** 1998.

9. *Intrusion detection: Applying machine learning to solaris audit data .* **Endler, D.** [ed.] IEEE Computer Society Press. Scottsdale, AZ. IEEE Computer Society. 1998. In Proceedings of the 1998 Annual Computer Security Applications Conference (ACSAC'98).

10. *Neural Networks for Misuse Detection: Initial Results.* **Cannady, J.** 1998. Proceedings of the Recent Advances in Intrusion Detection '98 Conference. págs. 31-47.

11. *Multiple Self-Organizing Maps fo rIntrusion Detection.* **Rhodes, B. C., Mahaffey, J. A., and Cannady, J. D.** 2000. In Proceedings of the 23rd National Information Systems Security Conference.

12. *Improving Intrusion Detection Performance using Keyword Selection and Neural Networks.* **Lippmann, R. P. and Cunningham, R. K.** 1999. Web proceedings of the 2nd International Workshop on Recent Advances in Intrusion Detection (RAID'99).

13. **A. Bivens, C. Palagiri, R. Smith, B. Szymanski and M. Embrechts.** *Network-based Intrusion Detection using Neural Networks. Intelligent Engineering Systems through Artificial Neural Networks.* New York, NY, : ASME Press, 2002. Vol. Vol 12.

14. **Arroyave, Juan David, Herrera, Jonathan y Vásquez, Esteban.** *Propuesta de modelo para un Sistema Inteligente de Detección de Intrusos en Redes Informáticas.* Escuela de Ingeniería de Antioquia. 2007. VII Jornada Nacional de Seguridad Informática ACIS.

15. **Leonid, Portno y Stolfo, Eleazar Eskin y Sal.** *Intrusion Detection Whit Unlabeled Data Usisng Clustering.* Departamento de Ciencias de la Computación, Universidad de Columbia. New York : s.n., 2001.

16. **Cannady, James.** *Artificial Neural Networks for Misuse Detection.* School of Computer and Information Sciences, Nova Southeastern University. Fort Lauderdale, FL : s.n.

17. **Quintana, Iris Castillo.** *Módulo de gestión de Alrmas.* 2010.

18. Python. [En línea] Marzo de 2010. http://www.python.org..

19. **Rivero Pérez, Jorge Luis.** *"Sistema para la centralización, análisis y procesamiento de trazas de servicios telemáticos para la Red UCf".* Informática, Universidad de Cienfuegos. 2009. Documento de Tesis.

20. Ventajas de Python. [En línea] Marzo de 2010. http://www.lawebdelprogramador.com/news/mostrar_new.php?id=79&texto=Pytho.

21. Weka (Aprendizaje Automático). [En línea] 2007.

22. Falso Positivo. [En línea] Abril de 2010. http://www.perantivirus.com/sosvirus/pregunta/falsopos.htm..

23. **Aldaba-Rubira, Emiliano.** *Introducción al reconocimiento de patrones mediante Redes Neuronales.* Barcelona : UPC-Campus Terrassa-DEE-EUETIT Colom. 1 08222.

24. *Conceptos Generales de Muestreo. Importancia y usos del muestreo.*

BIBLIOGRAFÍA

A. Bivens, C. Palagiri, R. Smith, B. Szymanski and M. Embrechts. *Network-based Intrusion Detection using Neural Networks. Intelligent Engineering Systems through Artificial Neural Networks.* New York, NY, : ASME Press, 2002. Vol. Vol 12.

Actualidad de la Tecnologçia de Detección de Intrusos en las Redes. **MSc. Baluja, García, Walter.** Habana : s.n., CITMATEL 2003. VIII Evento Internacional de Redes y Telecomunicaciones.

Albeiro, Calderón, Jhon, Moreno, Cadavid, Julian y Ovalle, Demetrio Arturo. *Red Neuronal para la clasificación de fallas en Líneas de Transamición a partir de Registros de Osciloperturbografía.* Madellín : s.n., 2008. ISSN 0012-7353.

Aldaba-Rubira, Emiliano. *Introducción al reconocimiento de patrones mediante Redes Neuronales.* Barcelona : UPC-Campus Terrassa-DEE-EUETIT Colom. 1 08222.

Alegre, López, Anita del C. *Simulación de Redes Neuronales Artificiales. Una Aplicación Didactica.* Facultad de Ciencias Exactas, Naturales y Agrimensura, Universidad Nacional del Nordeste. Corrientes : s.n., 2003.

Aplicación de minería de Datos para la Detección de Anomalías. Un Caso de Estudio. **Cuevas, Ania Cravero Leal y Sanuel Sepúlveda.** Temuco : WORKSHOP INTERNATIONAL, 2009, Vol. EIG2009.

Aplicación de Redes neuronales para la de Intrusos en Redes y Sistemas de Informaciçon. **Rivera, Pere, Carlos Alfonso, Montoya, Britto, Jaime Andrez y Echeverry, Isaza, Gustava Adolfo.** ISSN 0122-1701, 2005.

Aplicación de redes neuronales Para la Detección de Intrusos en Redes y Sistemas de Información. **Carlos Alfonso Pérez Rivera, Jaime Andres Britto Montoya y Gustabo Adolfo Isaza Echeverry.** ISSN 0122-1701, Pereira : s.n., 2005, Vol. No 27.

Arboleda Torres, Andres Felipe y Bedón Cortázar, Charles Edwuard. *Sistema de Detección de Intrusos Utilizando Inteligencia Artficial.* Depertamento de Sistemas, Universidad de Cuaca, Facultada de Ingenierçia Electrçonica y Comunicaciones. Popayán : s.n., 2004.

Arrans, Matias,Antonio L., Cruz, Alberto y Sanz, Bobi, Miguel A. *AMADIS: Arquitectura Multi-Agente para la Detección de Anomalías y Diagnóstico Inteligente de sistemas industriales.* Universisdad de pntificia Comillas. Madrid : s.n.

Ayuso, Manuel Antolín y Barcenilla Mancha, Miguel Ángel. *Minería de Datos: Intrusiones de Red.*

Barrera, García-Orea, Alejandro. *Presente y Futuro de los IDS.* Departamento de Inteligencia Artificial, Universidad Polit´ecnica de Madrid.

Bello, Rafael. *Inteligencia Artificial.* [Power Point] 2005.

Brito, Sarasa, Raycos, Rosete, Suárez, Alejandro y Acosta, Sánchez, Rolando. *Resultados de la Aplicación de Algoritmos de Minería de datos a la Gestión Docente.* Facultad de Ingeniería Informática, Instituto Superior Politécnico José Antonio Echverría(CUJAE). Ciudad de la Habana : s.n.

Cannady, James. *Artificial Neural Networks for Misuse Detection.* School of Computer and Information Sciences, Nova Southeastern University. Fort Lauderdale, FL : s.n.

Carnelli, Eías. *Aplicación de redes Neuronales Al Procesamiento de Alarmas en Sistemas S.C.A.D.A.* Habana : s.n., 2007.

Carnelli, Elías. *Aplicación de Redes Neuronales al Procesamiento de Alarmas en Sistemas S.C.A.D.A.* 2007.

Conceptos Generales de Muestreo. Importancia y usos del muestreo.

Conferencia de la CUJAE. Habana : s.n., 2004.

D. Michie, D.J. Spiegelhalter, C.C. Taylor, [ed.]. *M Machine Learning, Neural and Statistical.* 1994.

Dr. Bello, Rafael. *Monografía de Redes Neuronales Artificiales. .* Centro de Estudios Informáticos, Universidad Central "Marta Abreu" de Las Villas.

Elaine Rich, Kevin Knight. *Inteligencia Artificial, Segunda Edición.* [ed.] Juan Stumpf. Segunda Edición. s.l. : McGRAW-HILL/INTERAMERICANA DE ESPAÑA, S.A, 1994.

Evaluación de la tecnología de detección y respuesta a intrusiones e intentos de ataque en redes y telecomunicaciones. **Prof. Dr. Bertolín, Areitio, Javier.** Deusto : s.n., 2000.

Evaluación de las Tecnologías de detección y respuestas a intruciones e intentos de ataques en redes y telecomunicaciones. **Bertolín, Dr. Javier Areito.** Deusto : s.n., 2000.

Falso Positivo. [En línea] Abril de 2010. http://www.perantivirus.com/sosvirus/pregunta/falsopos.htm..

Fernández, Ing. Iren Lorenzo Fonseca y Dr. Rogelio Lau. *Detección de Intrusoso en la red utilizando redes neuronales, Resultados Preliminares.* Habana : s.n.

García, Pérez, Lynette y Ing. Martínez, Delgado, Edith. *BASE CONCEPTUAL DE UN SISTEMA INTELIGENTE DE APOYO A LAS DECISIONES MULTICRITERIO.* Instituto Superior Politécnico José Antonio Echeverría (CUJAE). Ciudad de la Habana : s.n.

Gonzales, Peña, Daniel. *Resumen de Tesis: Modelo para la integración de conocimiento biológico explícito en técnicas de clasificación aplicadas a datos procedentes de microarrays de ADN.* Departamento de Informática., Escuela Superior de Ingeniería Informática. Universidad de Vigo. Virgo : s.n., 2010. Informe de Tesis.

González, Castaño,Idis, y otros. *Una Metodología para la Detección de Anomalías en una Red a partir las Trazas de Ejecución de los servicios.* Universidad de la Habana. Ciudad de la Habana : s.n.

Improving Intrusion Detection Performance using Keyword Selection and Neural Networks. **Lippmann, R. P. and Cunningham, R. K.** 1999. Web proceedings of the 2nd International Workshop on Recent Advances in Intrusion Detection (RAID'99).

Lorenzo, Fonseca, Iren y Dr. Lau, Fernández, Rogelio. *Detección de Intrusos en la Red utilizando Redes Neuronales: Resultados Preliminares.* Instituto Superior Politécnico José Antonio Echeverría (CUJAE). Ciudad de la Habana : s.n.

Intrusion detection: Applying machine learning to solaris audit data . **Endler, D.** [ed.] IEEE Computer Society Press. Scottsdale, AZ. IEEE Computer Society. 1998. In Proceedings of the 1998 Annual Computer Security Applications Conference (ACSAC'98).

Jain, Anil K. y C., Dubes Richard. *Algorithms of Clustering Data.* Michigan : Editorial/production supervision, 1988. ISBN 0-13-022278-x.

Jain, Lakhmi C. y Martin, N.M. *Fusion of Neural Networks, Fuzzy Systems and Genetic Algorithms: Industrial Applications.* s.l. : CRC Press, CRC Press LLC., 1998. ISBN: 0849398045.

Juan David Arroyave, Jonathan Herrera y Esteban Vázques. *Propuesta de un modelo para un Sistema de Detección de Intrusos en Redes Informáticas (SIDIRI).* Antioquia : s.n., 2007.

Leonid, Portno y Stolfo, Eleazar Eskin y Sal. *Intrusion Detection Whit Unlabeled Data Usisng Clustering.* Departamento de Ciencias de la Computación, Universidad de Columbia. New York : s.n., 2001.

Machine Learning, Neural and Statistical, Classification. s.l. : D. Michie, D.J. Spiegelhalter, C.C. Taylor., 1994.

Martínez, Gras, Rodolfo, Mateo, Pérez, Miguel Ángel y Albert, Gardiola,María del Carmen. *El uso de técnicas de investigación en línea: desde el análisis de logs hasta la encuesta electrónica.*

McCulloch, W. S., Pitts, W. *A logical calculus of the ideas immanent in nervous activity.Bulletin of Mathematical.* 1943.

Minería de Datos: Conceptos y Tendencias. **Riquelme, José C., Ruíz, Roberto y Gilbert, Karina.** 029, Valencia : s.n., 2006, Revista Iberoamericana de Inteligencia Artificial, Vol. 10, págs. 11-18. Asociación Española para la Inteligencia Artificial. ISSN-1988-3064.

Mitchell, Tom. *Machine Learning.* s.l. : McGraw-Hill Science/Engineering/Math, 1997. ISBN: 0070428077..

Molina, Bonifacio Martín del Brío y Alfredo Sanz. *Redes Neuronales y Sistemas Difusos.* Segunda Edición. Zaragoza, España : ALFAOMEGA, 2001.

Multiple Self-Organizing Maps fo rIntrusion Detection. **Rhodes, B. C., Mahaffey, J. A., and Cannady, J. D.** 2000. In Proceedings of the 23rd National Information Systems Security Conference.

N. Altalef, "Syslog Centralizado con detección de eventos.". [En línea] Marzo de 2010. http://www.redklee.com.ar..

Neural Networks Applied in Intrusion Detection Systems. In Proceedings of the IEEE International Joint Conference on Neural Networks. **Bonifacio Jr., J. M., Cansian, A. M., Carvalho, A. C. P. L. F., and Moreira.** 1998.

Neural Networks for Misuse Detection: Initial Results. **Cannady, J.** 1998. Proceedings of the Recent Advances in Intrusion Detection '98 Conference. págs. 31-47.

Pervez, Shahbaz, y otros. *A Comparative Analysis of Artificial Neural Network Technologies in Intrusion Detection Systems.* University of Engineering and Technology. Taxila, Pakistan : s.n., 2006.

Python. [En línea] Marzo de 2010. http://www.python.org..

Quintana, Iris Castillo. *Módulo de gestión de Alrmas.* 2010.

Santamaria, Ruiz, Wilfredy. *Técnicas de Minería de Datos Aplicada a la Detección de Fraude: Estado de Arte.* Universidad Nacional de Columbia. Tesis de Maestría.

Scarfone, Karen y Mell, Peter. *Guide to Intrusion Detection and Prevention Systems (IDPS).* Department of Commerce, National Institute of Standard and Technology. Gaithersburg : s.n., 2007. Recommendations of the National Institute of Standards and Technology. MD 20899-8930.

Tolmos, Rodríguez-Piñero, Piedad. *Redes Neuronales Artificiales para el Análisis de Componentes Principales. La Red de Oja.*

Valente, Emilio. *LOG ANALYZER for Dummies.* 2007.

Ventajas de Python. [En línea] Marzo de 2010. http://www.lawebdelprogramador.com/news/mostrar_new.php?id=79&texto=Pytho.

Viacava, Carnesolta,Yoanna, Brito, Sarasa, Raycos y Acosta, Sánchez,Rolando. *Minería de Datos para le obtención de Patrones en el sistema Docente del Instituto Superior Politécnico José Antonio Echeverría.* Ciuda de la Habana : s.n.

Weka (Aprendizaje Automático). [En línea] 2007.

Winston, Patrick H. *Inteligencia Artificial.* 1994. Capitulo 2.

Winston, PatrickHenry. *Inteligencia Artificial.* Massachusetts : ADDISON-WESLEY IBEROAMÉRICA, 1992. 51876.

Zurutuza, Urko y Uribeetxeberria, Roberto. *Revisión del estado actual de la investigación en el uso de data mining para la Detección de Intruciones.* Departamento de Informática, Escuela Superior Politécnica de Mondragón. Mondragón Unibertsitatea : s.n.