COMILLAS
UNIVERSIDAD PONTIFICIA

ICAI

GRADO EN INGENIERÍA MATEMÁTICA E INTELIGENCIA ARTIFICIAL

TRABAJO FIN DE GRADO

Uso de técnicas de inspiración cuántica para mejorar la explicabilidad de los algoritmos de aprendizaje automático

Autor: Javier Ahumada Ortiz

Director: Pablo Díez Valle

Co-Director: Sergio Altares López

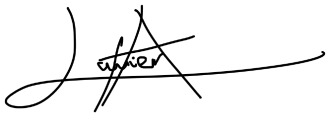
Madrid, Junio 2026

Declaro, bajo mi responsabilidad, que el Proyecto presentado con el título

Uso de técnicas de inspiración cuántica para mejorar la explicabilidad de los algoritmos de aprendizaje automático

en la ETS de Ingeniería - ICAI de la Universidad Pontificia Comillas en el curso académico 2025/2026 es de mi autoría, original e inédito y no ha sido presentado con anterioridad a otros efectos.

El Proyecto no es plagio de otro, ni total ni parcialmente, y la información que ha sido tomada de otros documentos está debidamente referenciada.



Fdo.: Javier Ahumada Ortiz Fecha: 16 / 06 / 2026

Autorizada la entrega del proyecto

EL DIRECTOR DEL PROYECTO

Firmado por DIEZ VALLE PABLO - ***5708** el día 17/06/2026 con un certificado emitido por AC FNMT Usuarios

Fdo.: Pablo Díez Valle Fecha: ..... / ..... / 2026

EL CODIRECTOR DEL PROYECTO

ALTARES LOPEZ SERGIO - 50330419W

Firmado digitalmente por ALTARES LOPEZ SERGIO - 50330419W
Nombre de reconocimiento (DN): c=ES, serialNumber=IDCES-50330419W, givenName=SERGIO, sn=ALTARES LOPEZ, cn=ALTARES LOPEZ SERGIO - 50330419W
Fecha: 2026.06.16 21:08:06 +02'00'

Fdo.: Sergio Altares López Fecha: ..... / ..... / 2026

COMILLAS
UNIVERSIDAD PONTIFICIA
ICAI

GRADO EN INGENIERÍA MATEMÁTICA E INTELIGENCIA ARTIFICIAL

TRABAJO FIN DE GRADO

Uso de técnicas de inspiración cuántica para mejorar la explicabilidad de los algoritmos de aprendizaje automático

Autor: Javier Ahumada Ortiz

Director: Pablo Díez Valle

Co-Director: Sergio Altares López

Madrid, Junio 2026

UNIVERSIDAD PONTIFICIA COMILLAS
Escuela Técnica Superior de Ingeniería (ICAI)
Grado en Ingeniería Matemática e Inteligencia Artificial

USO DE TÉCNICAS DE INSPIRACIÓN CUÁNTICA PARA MEJORAR LA EXPLICABILIDAD DE LOS ALGORITMOS DE APRENDIZAJE AUTOMÁTICO

Autor: Javier Ahumada Ortiz
Director: Pablo Díez Valle
Co-Director: Sergio Altares López

Resumen

Este trabajo desarrolla y valida un sistema de detección de anomalías basado en un modelo generativo *Matrix Product State* (MPS), una red tensorial originada en la física cuántica. El modelo se entrena únicamente con tráfico benigno y asigna a cada observación una puntuación de anomalía, identificando como posibles amenazas aquellas observaciones improbables bajo la distribución aprendida. Validado sobre el conjunto NSL-KDD de detección de intrusiones, alcanza un AUC-ROC de 0,953. Además, el MPS permite obtener explicaciones directamente a partir de su estructura interna e incorpora una métrica específica para evaluar la fiabilidad de dichas explicaciones.

Palabras clave: redes tensoriales; *Matrix Product States*; detección de anomalías; inteligencia artificial explicable; modelos generativos; ciberseguridad; NSL-KDD

Resumen ejecutivo

1 Introducción

A medida que los modelos de aprendizaje automático aumentan su capacidad, también resulta más difícil comprender por qué producen una determinada salida. Esta opacidad es especialmente problemática en ámbitos sensibles como la ciberseguridad, donde no basta con que un modelo acierte, sino que sus decisiones deben poder entenderse, auditarse y justificarse. Esta necesidad ha impulsado la inteligencia artificial explicable (*Explainable AI*, XAI) [1]. La interpretabilidad suele abordarse desde dos enfoques: las técnicas *post-hoc*, que aproximan a posteriori el comportamiento de un modelo ya entrenado pero no garantizan reflejar su razonamiento real, y los modelos interpretables por diseño, cuyas explicaciones se derivan de su propia estructura, pero que tradicionalmente han implicado una pérdida de capacidad expresiva.

En este contexto, las redes tensoriales (*tensor networks*) ofrecen una alternativa prometedora. Concebidas en la física cuántica para representar de forma compacta sistemas de muchos cuerpos [2], capturan dependencias complejas entre variables y, al mismo tiempo, exponen propiedades internas analizables que las hacen especialmente adecuadas para construir modelos intrínsecamente explicables. Entre ellas, el *Matrix Product State* (MPS) puede

emplearse como modelo generativo para aprender la distribución de probabilidad de un conjunto de datos [3], lo que lo convierte en una opción especialmente atractiva para la detección de anomalías [4].

2 Objetivos

El objetivo principal es desarrollar, implementar y evaluar un sistema de detección de anomalías basado en un modelo generativo MPS, atendiendo tanto a su capacidad de detección como a la interpretabilidad de las anomalías identificadas. En concreto, se plantea diseñar un detector semi-supervisado de una sola clase entrenado solo con tráfico benigno, validarlo sobre el conjunto NSL-KDD [5] y analizar su interpretabilidad mediante medidas derivadas de la propia estructura tensorial. La hipótesis de partida es que un MPS puede constituir una alternativa viable para la detección interpretable de anomalías.

3 Descripción del modelo

El sistema parte del enfoque de modelado generativo con MPS propuesto por Han et al. [3] y del esquema de interpretabilidad aplicado a ciberseguridad de Aizpurua et al. [6]. El MPS se utiliza como una *Born Machine*, en el que la probabilidad de una observación se obtiene a partir del cuadrado de una función de onda parametrizada como una cadena unidimensional de tensores (Figura 1). Las variables se discretizan previamente para adaptarlas a la estructura del modelo, y el entrenamiento se realiza mediante un algoritmo de tipo DMRG que recorre la cadena ajustando los tensores para maximizar la verosimilitud del tráfico benigno. Tras el entrenamiento, cada observación se puntúa mediante su log-verosimilitud negativa (NLL), de forma que cuanto menos probable sea, más anómala se considera. La ventaja diferencial del MPS es que su estructura tensorial permite leer directamente cualquier probabilidad de la distribución aprendida (marginales, condicionales y conjuntas) y calcular medidas de entrelazamiento como la entropía de Von Neumann o la información mutua, sin recurrir a mecanismos externos de explicabilidad.

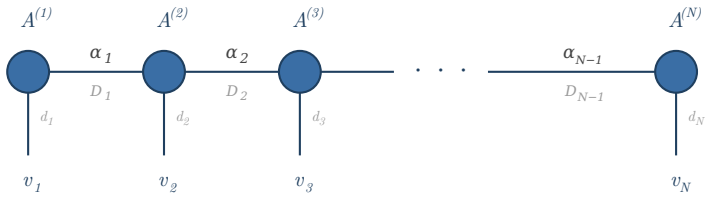


Figura 1: Arquitectura del modelo: un *Matrix Product State* representa la distribución de probabilidad del tráfico normal como una cadena de tensores, uno por característica.

4 Resultados

El modelo se validó sobre el conjunto de test de NSL-KDD (22 544 conexiones, un 56,9 % de ataques). Como detector, alcanza un AUC-ROC de 0,953 y un AUC-PR de 0,950, muy por encima de la línea base aleatoria, y su mejor equilibrio entre precisión y *recall* se obtiene con un F1 de 0,917. El análisis desagregado por familia de ataque revela que el sistema detecta con gran fiabilidad los ataques de denegación de servicio (DoS, AUC-ROC 0,978) y de sondeo (Probe, 0,968), que generan patrones de tráfico claramente anómalos. En cambio, el rendimiento disminuye en R2L (0,879) y U2R (0,889), familias más escasas y difíciles de distinguir porque tienden a camuflarse en sesiones aparentemente legítimas.

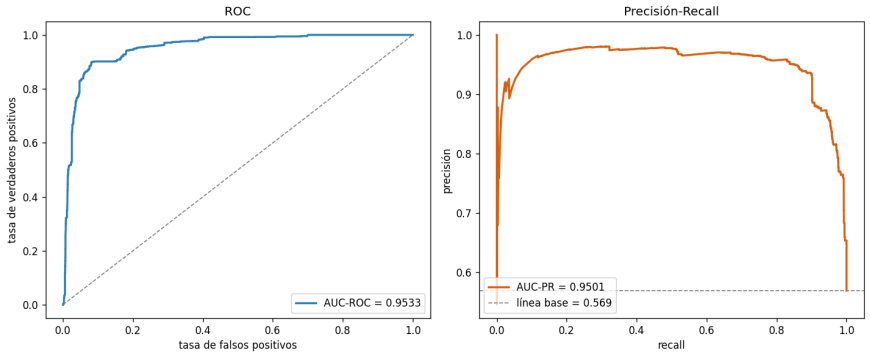


Figura 2: Curvas ROC (izquierda) y precisión-*recall* (derecha) sobre el conjunto de test.

En cuanto a la interpretabilidad, el modelo permite analizar tanto la probabilidad asignada a cada observación como la contribución de sus características, las dependencias aprendidas entre variables y el papel de sus combinaciones. La aportación más destacada es la *cuota de correlación*, una nueva métrica que mide qué parte de una alerta no queda explicada por contribuciones individuales. Una cuota baja indica que el desglose por características es fiable; una cuota alta advierte de que la anomalía reside en la combinación de valores, evitando interpretaciones engañosas.

5 Conclusiones

Los resultados permiten concluir que los modelos MPS no solo son una alternativa viable como detectores generativos de anomalías sobre datos tabulares, sino que además ofrecen una estructura especialmente adecuada para analizar e interpretar sus decisiones sin recurrir a técnicas externas de explicabilidad. La principal aportación de este trabajo es la cuota de correlación, que cualifica la fiabilidad de cada explicación individual. Entre las limitaciones cabe señalar la validación sobre un único conjunto de datos, la dependencia de la discretización previa y la naturaleza marginal de algunas medidas, aspectos que marcan las líneas de trabajo futuro.

6 Referencias

[1] A. Barredo Arrieta et al., «Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI», *Information Fusion*, vol. 58, págs. 82-115, 2020. DOI: 10.1016/j.inffus.2019.12.012.

[2] R. Orús, «A Practical Introduction to Tensor Networks: Matrix Product States and Projected Entangled Pair States», *Annals of Physics*, vol. 349, págs. 117-158, 2014. DOI: 10.1016/j.aop.2014.06.013.

[3] Z.-Y. Han, J. Wang, H. Fan, L. Wang y P. Zhang, «Unsupervised Generative Modeling Using Matrix Product States», *Physical Review X*, vol. 8, n.º 3, pág. 031 012, 2018. DOI: 10.1103/PhysRevX.8.031012.

[4] V. Chandola, A. Banerjee y V. Kumar, «Anomaly Detection: A Survey», *ACM Computing Surveys*, vol. 41, n.º 3, 2009. DOI: 10.1145/1541880.1541882.

[5] M. Tavallaee, E. Bagheri, W. Lu y A. A. Ghorbani, «A Detailed Analysis of the KDD CUP 99 Data Set», en *IEEE Symposium on Computational Intelligence for Security and Defense Applications (CISDA)*, 2009, págs. 1-6. DOI: 10.1109/CISDA.2009.5356528.

[6] B. Aizpurua, S. Palmer y R. Orús, «Tensor Networks for Explainable Machine Learning in Cybersecurity», *Neurocomputing*, vol. 639, pág. 130 211, 2025. DOI: 10.1016/j.neucom.2025.130211.

UNIVERSIDAD PONTIFICIA COMILLAS
Escuela Técnica Superior de Ingeniería (ICAI)
Grado en Ingeniería Matemática e Inteligencia Artificial

QUANTUM-INSPIRED TECHNIQUES FOR IMPROVING THE EXPLAINABILITY OF MACHINE LEARNING ALGORITHMS

Author: Javier Ahumada Ortiz
Director: Pablo Díez Valle
Co-Director: Sergio Altares López

Abstract

This work develops and validates an anomaly detection system based on a generative *Matrix Product State* (MPS) model, a tensor network originally developed in quantum physics. The model is trained exclusively on benign traffic and assigns an anomaly score to each observation, identifying as potential threats those observations that are unlikely under the learned distribution. Validated on the NSL-KDD intrusion detection dataset, it achieves an AUC-ROC of 0.953. In addition, the MPS makes it possible to obtain explanations directly from its internal structure and incorporates a specific metric to assess the reliability of those explanations.

Keywords: tensor networks; *Matrix Product States*; anomaly detection; explainable artificial intelligence; generative models; cybersecurity; NSL-KDD

Executive Summary

1 Introduction

As machine learning models increase in capacity, it also becomes more difficult to understand why they produce a given output. This opacity is particularly problematic in sensitive domains such as cybersecurity, where it is not enough for a model to be accurate: its decisions must also be understandable, auditable, and justifiable. This need has driven the development of explainable artificial intelligence (*Explainable AI*, XAI) [1]. Interpretability is usually approached from two perspectives: *post-hoc* techniques, which approximate the behavior of an already trained model after the fact but do not guarantee that they reflect its actual reasoning, and inherently interpretable models, whose explanations are derived from their own structure, but which have traditionally involved a loss of expressive capacity.

In this context, tensor networks offer a promising alternative. Originally conceived in quantum physics to compactly represent many-body systems [2], they capture complex dependencies between variables while exposing analyzable internal properties that make them especially suitable for building intrinsically explainable models. Among them, the *Matrix Product State* (MPS) can be used as a generative model to learn the probability distribution of a dataset [3], making it a particularly attractive option for anomaly detection [4].

2 Objectives

The main objective is to develop, implement, and evaluate an anomaly detection system based on a generative MPS model, considering both its detection capability and the interpretability of the anomalies identified. Specifically, the work aims to design a semi-supervised one-class detector trained only on benign traffic, validate it on the NSL-KDD dataset [5], and analyze its interpretability through measures derived from the tensor structure itself. The initial hypothesis is that an MPS can constitute a viable alternative for interpretable anomaly detection.

3 Model Description

The system is based on the generative MPS modeling approach proposed by Han et al. [3] and on the interpretability framework applied to cybersecurity by Aizpurua et al. [6]. The MPS is used as a *Born Machine*, in which the probability of an observation is obtained from the square of a wave function parameterized as a one-dimensional chain of tensors (Figure 1). The variables are first discretized to adapt them to the structure of the model, and training is carried out using a DMRG-type algorithm that sweeps through the chain, adjusting the tensors to maximize the likelihood of benign traffic. After training, each observation is scored using its negative log-likelihood (NLL), so that the less likely it is, the more anomalous it is considered. The key advantage of the MPS is that its tensor structure makes it possible to directly read any probability from the learned distribution, including marginal, conditional, and joint probabilities, and to compute entanglement measures such as Von Neumann entropy or mutual information, without resorting to external explainability mechanisms.

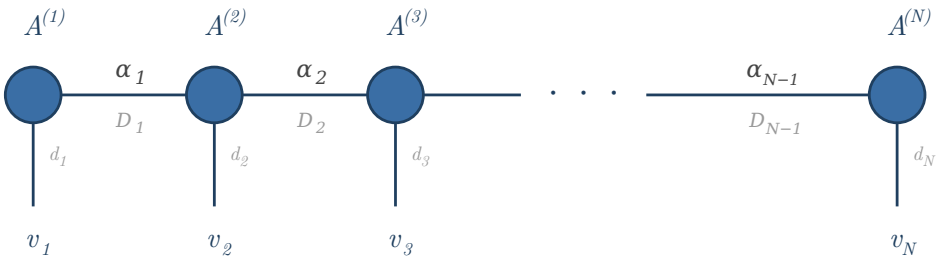


Figure 1: Model architecture: a *Matrix Product State* represents the probability distribution of normal traffic as a chain of tensors, one per feature.

4 Results

The model was validated on the NSL-KDD test set (22 544 connections, 56.9% attacks). As a detector, it achieves an AUC-ROC of 0.953 and an AUC-PR of 0.950, well above the random baseline, and its best balance between precision and *recall* is obtained with an F1 score of 0.917. The attack-family-level analysis shows that the system detects denial-of-service attacks with high reliability (DoS, AUC-ROC 0.978) as well as probing attacks (Probe, 0.968), both of which generate clearly anomalous traffic patterns. By contrast, performance decreases for R2L (0.879) and U2R (0.889), which are rarer and more difficult to distinguish because they tend to be hidden within apparently legitimate sessions.

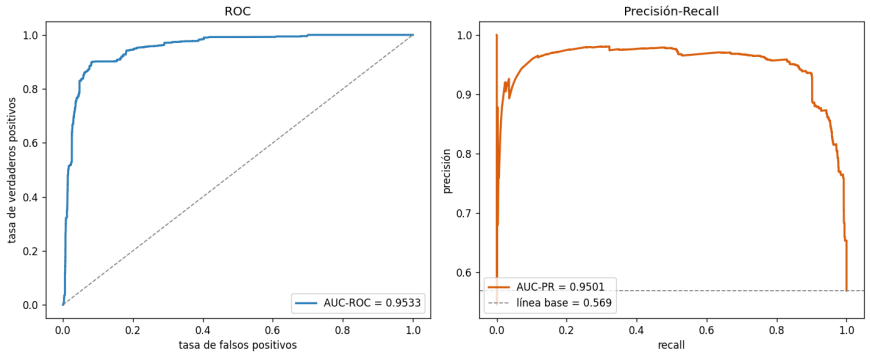


Figure 2: ROC curve (left) and precision-*recall* curve (right) on the test set.

Regarding interpretability, the model makes it possible to analyze both the probability assigned to each observation and the contribution of its features, the dependencies learned between variables, and the role of their combinations. The most notable contribution is the *correlation share*, a new metric that measures what portion of an alert is not explained by individual contributions. A low share indicates that the feature-level breakdown is reliable; a high share warns that the anomaly lies in the combination of values, thereby avoiding misleading interpretations.

5 Conclusions

The results show that MPS models are not only a viable alternative as generative anomaly detectors for tabular data, but also provide a structure that is especially suitable for analyzing and interpreting their decisions without resorting to external explainability techniques. The main contribution of this work is the correlation share, which qualifies the reliability of each individual explanation. The main limitations include validation on a single dataset, dependence on prior discretization, and the marginal nature of some measures, all of which define directions for future work.

6 References

[1] A. Barredo Arrieta et al., «Explainable artificial intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI», *Information Fusion*, vol. 58, pp. 82–115, 2020. DOI: 10.1016/j.inffus.2019.12.012.

[2] R. Orús, «A practical introduction to tensor networks: Matrix product states and projected entangled pair states», *Annals of Physics*, vol. 349, pp. 117–158, 2014. DOI: 10.1016/j.aop.2014.06.013.

[3] Z.-Y. Han, J. Wang, H. Fan, L. Wang, and P. Zhang, «Unsupervised generative modeling using matrix product states», *Physical Review X*, vol. 8, no. 3, p. 031 012, 2018. DOI: 10.1103/PhysRevX.8.031012.

[4] V. Chandola, A. Banerjee, and V. Kumar, «Anomaly detection: A survey», *ACM Computing Surveys*, vol. 41, no. 3, 2009. DOI: 10.1145/1541880.1541882.

[5] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, «A detailed analysis of the KDD CUP 99 data set», in *IEEE Symposium on Computational Intelligence for Security and Defense Applications (CISDA)*, 2009, pp. 1–6. DOI: 10.1109/CISDA.2009.5356528.

[6] B. Aizpurua, S. Palmer, and R. Orús, «Tensor networks for explainable machine learning in cybersecurity», *Neurocomputing*, vol. 639, p. 130 211, 2025. DOI: 10.1016/j.neucom.2025.130211.

Índice

Índice de figuras

Índice de tablas

Capítulo 1 Introducción

1.1. Motivación

En las últimas décadas, la inteligencia artificial y el aprendizaje automático se han convertido en herramientas fundamentales para abordar problemas complejos a partir de datos [1], [2]. En particular, el aprendizaje profundo [3] ha permitido alcanzar resultados excelentes en tareas como la visión por ordenador o el procesamiento de lenguaje natural mediante arquitecturas como redes convolucionales [4], modelos recurrentes [5] o *transformers* [6]. Sin embargo, el aumento de capacidad de estos modelos suele ir acompañado de una mayor complejidad, lo que dificulta comprender por qué producen una determinada salida.

Esta falta de transparencia es especialmente problemática cuando los sistemas de inteligencia artificial se utilizan en contextos donde sus decisiones pueden tener consecuencias relevantes. En estos casos, no basta con que el modelo tenga un buen rendimiento, sino que también es necesario entender los motivos que llevan a una predicción concreta para poder confiar en el sistema. Esta necesidad ha impulsado el desarrollo de la inteligencia artificial explicable (*Explainable AI*, XAI), cuyo objetivo es hacer que el comportamiento de los modelos sea más comprensible para los usuarios [7].

La interpretabilidad se suele abordar desde dos enfoques principales. Por un lado, existen técnicas *post-hoc*, como LIME [8] o SHAP [9], que se aplican a un modelo ya entrenado e intentan aproximar su comportamiento. Estas técnicas tienen la ventaja de poder aplicarse a un modelo ya existente, incluso cuando este no ha sido diseñado para ser interpretable. Sin embargo, estos métodos no siempre garantizan que la explicación obtenida refleje realmente el razonamiento interno del modelo, además de que sus explicaciones pueden ser sensibles a pequeñas variaciones en los datos o al procedimiento utilizado para generar las explicaciones.

Por otro lado, existen modelos cuya interpretabilidad forma parte de su propia estructura. En este caso, las explicaciones se obtienen a partir del propio modelo y no dependen de mecanismos externos de explicabilidad. Algunos ejemplos clásicos son los árboles de decisión [10] o los modelos lineales [2]. Estos modelos permiten analizar de forma más directa qué características influyen en la predicción y cómo lo hacen. Sin embargo, en muchas tareas complejas estos modelos más simples pueden no alcanzar el mismo rendimiento que arquitecturas más expresivas como las redes neuronales profundas.

El reto consiste, por tanto, en desarrollar modelos capaces de representar relaciones complejas entre variables y, al mismo tiempo, ofrecer una interpretabilidad inherente. Es aquí donde las redes tensoriales se presentan como una alternativa prometedora.

Las redes tensoriales (*tensor networks*) surgieron en el ámbito de la física cuántica como una herramienta para representar de forma eficiente el estado de sistemas de muchos cuerpos, cuya descripción exacta requeriría un número muy elevado de parámetros [11]. En los últimos años, su uso se ha extendido al aprendizaje automático, donde han demostrado ser útiles para diseñar modelos compactos capaces de capturar dependencias complejas entre variables. Además, su estructura permite analizar determinadas propiedades internas del modelo, lo que las convierte en candidatas interesantes para el desarrollo de sistemas interpretables [12].

Entre las distintas familias de redes tensoriales, este trabajo se centra en el *Matrix Product State* (MPS), una de las estructuras más sencillas y estudiadas. Un MPS puede utilizarse como modelo generativo para aprender la distribución de probabilidad de un conjunto de datos [13]. Esta propiedad lo hace especialmente interesante para la detección de anomalías [14], puesto que si el modelo se entrena únicamente con datos que representan el comportamiento normal, las observaciones poco probables bajo la distribución aprendida pueden identificarse como anómalas. Además, la propia estructura tensorial permite extraer explicaciones internas sin recurrir a mecanismos externos de explicabilidad [15].

Este trabajo desarrolla y valida un sistema de detección de anomalías basado en un modelo generativo MPS, entrenado de forma semi-supervisada sobre datos de comportamiento normal. El sistema parte del enfoque de modelado generativo con MPS propuesto por Han et al. [13] y del esquema de interpretabilidad aplicado a ciberseguridad de Aizpurua et al. [15]. Aunque el modelo propuesto puede aplicarse a cualquier problema de detección de anomalías sobre datos tabulares, en este trabajo se valida en el contexto de la detección de intrusiones en redes [16] utilizando el conjunto de datos NSL-KDD.

Este trabajo no se limita a reproducir los análisis interpretables del trabajo de referencia [15], sino que los extiende y matiza. En particular, se incorporan análisis de probabilidades conjuntas para identificar combinaciones específicas de valores relevantes, se introduce la entropía de enlace para estudiar cómo se distribuyen las correlaciones a lo largo de la cadena MPS y se propone la cuota de correlación como una medida para evaluar la fiabilidad de las explicaciones marginales de anomalías individuales. Esta última aportación resulta especialmente importante porque advierte cuándo una alerta puede explicarse por valores individualmente improbables y cuándo, por el contrario, la anomalía reside principalmente en la combinación de valores observada.

1.2. Objetivos

El objetivo principal de este Trabajo Fin de Grado es desarrollar, implementar y evaluar un sistema de detección de anomalías basado en un modelo generativo *Matrix Product State* (MPS), prestando especial atención no solo a su capacidad de detección, sino también a la interpretabilidad de las anomalías identificadas. A partir de este objetivo general, se plantean los siguientes objetivos específicos:

- Estudiar el uso de los modelos MPS como modelos generativos aplicados a datos tabulares, analizando su capacidad para aprender la distribución de probabilidad de los datos de entrenamiento.
- Diseñar e implementar un sistema de detección de anomalías semi-supervisado, entrenado únicamente con observaciones normales, de forma que las muestras con baja probabilidad bajo la distribución aprendida puedan ser identificadas como anómalas.

- Validar el sistema propuesto sobre un conjunto de datos de detección de intrusiones en redes, utilizando el conjunto NSL-KDD como caso de estudio representativo de un problema de ciberseguridad.
- Analizar la interpretabilidad inherente del modelo MPS mediante el estudio de medidas asociadas a la propia estructura del modelo como las probabilidades marginales, la entropía de Von Neumann o la información mutua.
- Extender el análisis interpretable del modelo incorporando medidas adicionales, como las probabilidades conjuntas, la entropía de enlace y la cuota de correlación.

En conjunto, estos objetivos buscan comprobar si un modelo generativo basado en MPS puede constituir una alternativa viable para la detección interpretable de anomalías en datos tabulares, combinando capacidad de modelado probabilístico con herramientas internas de explicación.

1.3. Alineación con los ODS

Este Trabajo de Fin de Grado se alinea con varios de los Objetivos de Desarrollo Sostenible (ODS) definidos por la Organización de las Naciones Unidas, especialmente con aquellos relacionados con el desarrollo responsable de la tecnología, la innovación y el uso transparente de sistemas automatizados.

En primer lugar, el trabajo se relaciona de forma directa con el ODS 9: Industria, innovación e infraestructura. La detección de anomalías es una tarea relevante en numerosos entornos industriales, tecnológicos y de ciberseguridad, donde resulta necesario identificar comportamientos inusuales que puedan afectar al funcionamiento de sistemas críticos. En este contexto, el estudio de modelos basados en Matrix Product States (MPS) contribuye al desarrollo de soluciones innovadoras para el análisis de datos, combinando capacidad de detección con una mayor interpretabilidad del modelo.

Además, la explicabilidad resulta especialmente importante para que los sistemas de inteligencia artificial puedan ser utilizados de forma responsable en contextos reales. No basta con detectar una anomalía, sino que también es necesario comprender qué variables han contribuido a dicha detección y cómo se relacionan entre sí.

El trabajo también guarda relación con el ODS 10: Reducción de las desigualdades, al promover el desarrollo de modelos más transparentes y analizables. En sistemas automatizados, la falta de interpretabilidad puede dificultar la identificación de errores, sesgos o comportamientos no deseados. Por ello, el estudio de modelos intrínsecamente interpretables puede contribuir a un uso más fiable, justo y responsable de la inteligencia artificial

En conjunto, este TFG contribuye al avance hacia sistemas de aprendizaje automático más transparentes, auditables y adecuados para su aplicación en problemas reales.

1.4. Estructura del trabajo

El resto de la memoria se organiza de la siguiente forma:

En la Sección 2 se presenta el estado del arte relacionado con el trabajo. En ella se revisan los principales antecedentes sobre Inteligencia Artificial Explicable, métodos de detección de anomalías, modelos generativos y aplicaciones de las redes de tensores al aprendizaje automático, con especial atención a los modelos basados en Matrix Product States (MPS).

En la Sección 3 se describe la metodología seguida para el desarrollo del sistema propuesto. En primer lugar, se formaliza el problema de detección de anomalías abordado y se establecen los requisitos del sistema. A continuación, se presenta el diseño de la solución, incluyendo el modelo generativo MPS, la codificación de características, el algoritmo de entrenamiento y el mecanismo de explicabilidad. Finalmente, se detalla la implementación realizada, el caso de uso considerado, el conjunto de datos NSL-KDD, la arquitectura del modelo, las métricas de evaluación, la estructura del código, el pipeline experimental, las medidas de reproducibilidad y las tecnologías empleadas.

En la Sección 4 se exponen los resultados obtenidos. Esta sección incluye el análisis de la codificación de los datos, el comportamiento del entrenamiento, la evaluación del modelo entrenado y el estudio de su explicabilidad. En particular, se analizan las probabilidades extraídas del modelo, la entropía de Von Neumann, la información mutua, la entropía de enlace, la importancia de las características y la descomposición de anomalías individuales.

Por último, en la Sección 5 se recogen las principales conclusiones del trabajo, se discuten sus limitaciones y se plantean posibles líneas de trabajo futuro.

Capítulo 2 Estado del arte

Este trabajo se sitúa en la intersección entre la inteligencia artificial explicable y el uso de redes tensoriales en aprendizaje automático, tomando la detección de anomalías como caso de aplicación. En esta sección se revisan en primer lugar los principales enfoques de la interpretabilidad. A continuación, se presentan las redes tensoriales como modelos de aprendizaje automático, atendiendo tanto a su uso en modelado generativo y en detección de anomalías como a las propiedades que las hacen especialmente adecuadas para construir modelos intrínsecamente explicables.

2.1. Inteligencia artificial explicable

A medida que los modelos de aprendizaje automático aumentan su capacidad, también resulta más difícil entender cómo toman sus decisiones. Esta falta de transparencia es especialmente problemática en ámbitos como la ciberseguridad, la sanidad o la justicia, donde no basta con que un modelo ofrezca buenos resultados, sino que también es necesario poder entender, auditar y justificar sus decisiones. Esta necesidad ha impulsado el desarrollo de la inteligencia artificial explicable (*Explainable AI*, XAI) [7].

La interpretabilidad suele abordarse desde dos enfoques principales. El primero comprende las técnicas *post-hoc*, que se aplican a modelos ya entrenados con el objetivo de explicar sus predicciones. Entre las más utilizadas se encuentran LIME [8], que aproxima localmente el comportamiento del modelo mediante un modelo lineal sencillo, y SHAP [9], que reparte la contribución de cada característica utilizando valores de Shapley. Su principal ventaja es que pueden aplicarse a prácticamente cualquier modelo, incluso a arquitecturas que no han sido diseñadas para ser interpretables. Sin embargo, esta generalidad también conlleva limitaciones importantes, puesto que al tratarse de explicaciones externas, no garantizan reflejar el razonamiento real del modelo y pueden ser sensibles a pequeñas variaciones en los datos o en el propio procedimiento de explicación. De hecho, se ha demostrado que algunos métodos de atribución ampliamente utilizados no superan comprobaciones básicas de coherencia, llegando a producir explicaciones similares para modelos entrenados y para modelos con pesos aleatorios [17].

El segundo enfoque consiste en utilizar modelos cuya interpretabilidad forma parte de su propia estructura. En este caso, las explicaciones no se obtienen mediante un método externo, sino directamente a partir del funcionamiento interno del modelo. Ejemplos clásicos de este tipo de modelos son los árboles de decisión [10] y los modelos lineales [2]. Esta postura ha ganado peso en contextos de alto riesgo, donde diversos autores defienden que los modelos deberían ser interpretables por diseño, en lugar de depender de explicaciones *post-hoc* aplicadas a modelos de caja negra [18], [19].

El problema es que los modelos intrínsecamente interpretables más simples no siempre alcanzan el rendimiento de arquitecturas más expresivas, como las redes neuronales profundas, lo que ha consolidado la idea de que existe un compromiso entre capacidad predictiva e interpretabilidad. Una de las motivaciones centrales de este trabajo es avanzar hacia modelos que combinen capacidad expresiva e interpretabilidad intrínseca.

2.2. Redes tensoriales en aprendizaje automático

Las redes tensoriales surgieron en la física cuántica como una herramienta para representar de forma eficiente el estado de sistemas cuánticos de muchos cuerpos. La descripción exacta de estos sistemas requeriría, en general, un número de parámetros que crece exponencialmente con el tamaño del sistema [11]. La eficiencia de las redes tensoriales se apoya en las llamadas leyes de área, que limitan el entrelazamiento de los estados físicamente relevantes y permiten describirlos con un número de parámetros tratable [20]. Entre las distintas familias de redes tensoriales, el *Matrix Product State* (MPS) es una de las estructuras más sencillas y estudiadas, estrechamente relacionada con el algoritmo del grupo de renormalización de la matriz densidad (DMRG) y con la noción de forma canónica [21].

En la última década, estas técnicas se han trasladado también al aprendizaje automático. Un primer hito fue el trabajo de Stoudenmire y Schwab, que mostró que un MPS podía utilizarse como modelo de clasificación supervisada. En su propuesta, el clasificador se parametriza mediante una red tensorial y se optimiza con técnicas inspiradas en el DMRG, obteniendo buenos resultados en tareas con datos de imágenes [22]. Este resultado abrió la puerta a un uso más amplio de las redes tensoriales en aprendizaje automático, tanto en tareas supervisadas como en modelado generativo.

2.3. Redes tensoriales como modelos generativos

Además de utilizarse para clasificación, las redes tensoriales pueden emplearse como modelos generativos capaces de aprender la distribución de probabilidad de un conjunto de datos. Han et al. propusieron utilizar el MPS con este objetivo, dando lugar a las denominadas *Born machines*. En estos modelos, la probabilidad de una configuración se obtiene como el módulo al cuadrado de una amplitud, en analogía con la regla de Born de la mecánica cuántica [13], [23]. Frente a antecedentes clásicos del modelado generativo, como las máquinas de Boltzmann [24], las *Born machines* basadas en MPS presentan dos ventajas importantes: permiten calcular la verosimilitud de forma tratable y facilitan el muestreo directo. Posteriormente, esta línea se ha extendido a estructuras más expresivas, como las redes tensoriales en árbol [25], y se ha estudiado teóricamente la capacidad expresiva de distintas factorizaciones tensoriales para el modelado probabilístico [26].

Esta capacidad generativa conecta de forma natural con la detección de anomalías, cuyo objetivo es identificar observaciones que se desvían de manera significativa del comportamiento esperado [14]. Dado que las anomalías suelen ser escasas, heterogéneas y difíciles de etiquetar, es habitual recurrir a enfoques semi-supervisados o de una sola clase (*one-class*).

En estos enfoques, el modelo se entrena únicamente con observaciones normales y considera anómalas aquellas muestras que tienen baja probabilidad bajo la distribución aprendida [27], [28]. Desde esta perspectiva, un modelo generativo encaja de manera directa en el problema, ya que la probabilidad asignada a cada muestra proporciona una puntuación de anomalía con una interpretación probabilística clara.

La aplicación de redes tensoriales a la detección de anomalías es un área reciente pero en crecimiento. Wang et al. abordaron la detección de anomalías de una clase mediante redes tensoriales y mostraron que estos modelos pueden igualar o superar a algoritmos clásicos y profundos sobre datos tabulares, aprovechando su eficiencia para operar de forma lineal en un espacio de dimensión exponencial [29]. En física de altas energías, también se han empleado modelos MPS entrenados sobre eventos normales para detectar sucesos anómalos en colisiones de protones [30]. En el ámbito de la ciberseguridad, Aizpurua et al. propusieron un detector de anomalías basado en un MPS generativo entrenado de forma semi-supervisada sobre comportamiento normal. Su método se aplicó a un caso real de ciberseguridad con datos discretos y mostró que el MPS podía competir con modelos profundos como los autoencoders y las GAN en capacidad de detección [15].

2.4. Redes tensoriales y explicabilidad

La característica que diferencia a las redes tensoriales de muchos modelos profundos de caja negra es que su propia estructura permite extraer magnitudes internas con interpretación directa, sin necesidad de recurrir a técnicas externas de explicabilidad. A partir de la red entrenada pueden calcularse matrices de densidad reducida y, a partir de ellas, probabilidades marginales y condicionales, entropía de von Neumann asociada a cada característica e información mutua entre pares de variables [12], [31], [32]. Estas magnitudes hacen que las redes tensoriales sean especialmente adecuadas para construir modelos de aprendizaje automático interpretables por diseño.

El trabajo de referencia para esta memoria es el de Aizpurua et al. [15], que explota precisamente esta interpretabilidad estructural en el contexto de la ciberseguridad. Además de detectar anomalías, los autores extraen del MPS probabilidades marginales, matrices de densidad reducida, entropía de von Neumann e información mutua para explicar tanto la distribución aprendida como las anomalías detectadas. Esta combinación de capacidad generativa e interpretabilidad inherente es lo que distingue este enfoque de las alternativas profundas.

Más recientemente, Hohenfeld et al. [33] han extendido el trabajo de Aizpurua et al. en dos direcciones. Por un lado, lo trasladan del dominio discreto a datos de valor real, mediante una codificación basada en polinomios ortogonales. Por otro, además del MPS, introducen las redes tensoriales en árbol como arquitectura para la detección explicable de anomalías. Sus experimentos sobre tres conjuntos de referencia de dominios diversos, como electrocardiogramas, imágenes de satélite y clasificación de correo, muestran un rendimiento competitivo frente a varios modelos de referencia y confirman que el enfoque puede aplicarse más allá del caso original. No obstante, el trabajo mantiene el mismo repertorio de explicaciones que el

trabajo de referencia.

2.5. Cuestiones abiertas y aportaciones de este trabajo

La revisión del estado del arte muestra que el uso de modelos MPS generativos para la detección explicable de anomalías es una línea reciente, pero que ya cuenta con trabajos de referencia. Aizpurua et al. [15] establecieron el marco principal, combinando detección semi-supervisada a partir de comportamiento normal con un conjunto de explicaciones extraídas de la propia estructura del modelo como las probabilidades marginales, matrices de densidad reducida, entropía de von Neumann e información mutua. Más recientemente, Hohenfeld et al. [33] ampliaron este enfoque a datos de valor real y a redes tensoriales en árbol, mostrando que la idea puede aplicarse en dominios diversos. Sin embargo, este último trabajo se aleja de la detección de intrusiones en red y mantiene un repertorio de explicaciones similar al del trabajo de referencia.

A partir de esta revisión pueden identificarse dos aspectos todavía poco explorados. En primer lugar, las explicaciones empleadas hasta ahora se basan principalmente en magnitudes marginales y, en algunos casos, en dependencias entre pares de variables. Esto deja margen para explotar otras fuentes de información, como las probabilidades conjuntas, que permiten identificar combinaciones concretas de valores relevantes, o la distribución de las correlaciones a lo largo de la cadena MPS. En segundo lugar, no se ha abordado de forma explícita la fiabilidad de las propias explicaciones marginales, es decir, hasta qué punto una descomposición por características refleja realmente la puntuación de anomalía asignada por el modelo. Esta cuestión es importante porque una anomalía puede deberse tanto a valores individualmente improbables como a combinaciones poco frecuentes de valores. En este segundo caso, una explicación basada únicamente en marginales puede resultar incompleta o incluso engañosa.

Este trabajo se sitúa precisamente en ese hueco. Por un lado, valida un detector generativo basado en MPS en el contexto de la detección de intrusiones en red, utilizando el conjunto NSL-KDD [34] e incorporando un análisis desagregado por familias de ataque. Por otro lado, no se limita a reproducir las explicaciones del trabajo de referencia, sino que las amplía en tres direcciones. En primer lugar, incorpora probabilidades conjuntas para detectar combinaciones de valores relevantes. En segundo lugar, introduce la entropía de enlace como herramienta para estudiar cómo se distribuyen las correlaciones a lo largo de la cadena. Por último, propone la cuota de correlación como medida para evaluar la fiabilidad de las explicaciones marginales, cuantificando en qué medida la anomalía de una muestra se debe a valores individuales o a correlaciones entre variables.

Capítulo 3 Metodología

3.1. Planteamiento del problema

3.1.1. Fundamentos teóricos

La Inteligencia Artificial se define como la disciplina que estudia los sistemas capaces de percibir su entorno y actuar para alcanzar objetivos [1]. Una de sus ramas principales es el aprendizaje automático, que estudia la construcción de sistemas capaces de inferir patrones y reglas a partir de los datos, en lugar de seguir un comportamiento programado de forma explícita [2]. De este modo, el sistema mejora su desempeño en una tarea a medida que dispone de más experiencia, generalizando a nuevas situaciones a partir de los ejemplos observados.

Los métodos de aprendizaje automático se clasifican tradicionalmente en tres grandes paradigmas según la naturaleza de los datos de los que aprende el sistema [2]. En primer lugar, el aprendizaje supervisado consiste en aprender a partir de datos etiquetados, es decir, combinaciones de la entrada al modelo y la salida deseada para dicha entrada. A partir de estos ejemplos, el sistema aprende a predecir la salida correspondiente a entradas nuevas. Por su parte, el aprendizaje no supervisado consiste en aprender a partir de datos de entrada sin conocer la salida deseada del modelo. En este caso, el objetivo del modelo es descubrir la estructura subyacente de los datos. Por ejemplo, un modelo de aprendizaje no supervisado puede aprender a agrupar instancias similares entre sí, a estimar la distribución de probabilidad de los datos o a reducir la dimensionalidad de los datos eliminando información repetitiva. Finalmente, el aprendizaje por refuerzo consiste en aprender mediante ensayo y error. El sistema aprende en un entorno en el que recibe recompensas o penalizaciones según sus acciones. En este caso, el objetivo del sistema es aprender una política de actuación que maximice la recompensa acumulada a lo largo del tiempo.

Además de estos tres paradigmas principales, existen enfoques intermedios que combinan características de varios de ellos. Entre ellos se encuentra el aprendizaje semi-supervisado [28], situado entre el aprendizaje supervisado y el no supervisado. Por ejemplo, el aprendizaje semi-supervisado puede consistir en aprender a partir de una combinación de datos etiquetados y datos sin etiquetar. Sin embargo, en el contexto de la detección de anomalías [14], cuyo objetivo es identificar patrones que se desvían del comportamiento esperado, el aprendizaje semi-supervisado suele referirse a entrenar al modelo únicamente con datos etiquetados como esperados o benignos. Este enfoque también se conoce como clasificación de una sola clase (*one-class classification*) [27], y consiste en que el modelo aprenda a caracterizar una única clase de referencia y detecte como anómala cualquier observación que se aleje significativamente de ella.

Para caracterizar la clase de referencia son especialmente útiles los modelos generativos. Un modelo generativo es un modelo que aprende la distribución de probabilidad que rige un conjunto de datos [2]. Una de las principales utilidades de estos modelos es generar datos sintéticos a partir de la distribución aprendida. Sin embargo, si el modelo define la distribución de probabilidad de manera explícita, también es capaz de asignar a nuevas observaciones

una medida de verosimilitud que indique qué tan compatibles resultan con la distribución aprendida. Esta propiedad los hace especialmente útiles en tareas como la detección de anomalías, donde si una observación es poco verosímil se puede considerar una anomalía.

3.1.2. Definición formal del problema

El problema abordado en este trabajo es la detección de anomalías. Una anomalía se define como un patrón en los datos que no se ajusta al comportamiento esperado [14]. Por tanto, esta tarea consiste en caracterizar dicho comportamiento esperado para posteriormente identificar las observaciones que se desvíen de él. La detección de anomalías tiene multitud de aplicaciones, como la detección de fraude, la detección de fallos en sistemas críticos o la detección de intrusiones en ciberseguridad.

De acuerdo con Chandola et al. [14], un problema de detección queda definido formalmente por la naturaleza de los datos de entrada, el tipo de anomalía, la disponibilidad de etiquetas y la salida del detector. A continuación, se describen estos cuatro aspectos en el contexto de este trabajo.

En cuanto a los datos de entrada, estos se representan como vectores de N atributos de distinta naturaleza, incluyendo variables binarias, categóricas y continuas. Además, cada observación se evalúa de forma independiente, sin asumir relaciones secuenciales, temporales ni espaciales con las demás. Debido a esta independencia entre instancias, este tipo de datos se conoce como datos puntuales (*point data*).

Asimismo, las anomalías consideradas son anomalías puntuales (*point anomalies*), puesto que cada instancia se clasifica como benigna o anómala de forma independiente, sin necesidad de contexto adicional ni de considerar agrupaciones de varias instancias.

Respecto a la disponibilidad de etiquetas, se adopta un enfoque semi-supervisado. A partir de un conjunto de datos completamente etiquetado, se utilizan únicamente instancias etiquetadas como benignas para entrenar el modelo, con el objetivo de que aprenda a modelar la distribución del comportamiento benigno. Por tanto, las técnicas de entrenamiento son las mismas que en enfoques no supervisados, siendo la única diferencia la eliminación de las instancias anómalas del conjunto de entrenamiento.

Finalmente, el sistema propuesto es de tipo *scoring*. Para cada observación, la salida del modelo es una puntuación basada en la verosimilitud logarítmica negativa (*negative log-likelihood*, NLL). Esta puntuación permite ordenar las instancias según su grado de sospecha, de forma que valores más altos indican que la observación tiene menor probabilidad bajo la distribución aprendida y, por tanto, puede considerarse más anómala. Este enfoque se diferencia de otros sistemas que clasifican directamente las observaciones en una clase benigna o anómala.

3.1.3. Requisitos

Tras delimitar el problema, esta sección formula los requisitos que debe cumplir el sistema. Primero se describen los requisitos funcionales, que describen qué debe hacer el sistema. Por su parte, los requisitos no funcionales describen cómo debe hacerlo.

3.1.3.1. Requisitos funcionales

- **Asignación de una puntuación de anomalía**. El sistema debe ser capaz de asignar a cualquier observación una puntuación numérica que cuantifique su grado de anomalía. Este requisito motiva el uso de un modelo generativo, en el que la verosimilitud logarítmica negativa proporciona una puntuación bien definida y comparable entre observaciones. Este modelo se describe en la Sección 3.2.1.
- **Evaluación del rendimiento de detección**. Se debe poder evaluar cuantitativamente la capacidad del sistema para distinguir entre observaciones normales y anómalas. Este requisito motiva el cálculo de las métricas de rendimiento descritas en la Sección 4.3.
- **Producción de explicaciones**. Debe ser posible obtener una explicación que indique qué variables han contribuido a que una observación reciba una puntuación de anomalía elevada y en qué grado. Este requisito motiva la extracción de las magnitudes interpretables descritas en la Sección 3.2.4.

3.1.3.2. Requisitos no funcionales

- **Interpretabilidad intrínseca**. Las explicaciones del sistema deben derivarse directamente de la estructura interna del modelo, sin recurrir a técnicas post-hoc como SHAP [9] o LIME [8]. Este requisito es el que descarta las redes neuronales convencionales y motiva el uso de redes tensoriales [12], [15].
- **Robustez frente a características heterogéneas**. El sistema debe operar con observaciones cuyas características son de distintos tipos: categóricas, binarias y numéricas. Este requisito motiva la etapa de codificación de características descrita en la Sección 3.2.2.
- **Independencia del dominio**. El sistema debe ser aplicable a distintos conjuntos de datos y escenarios de detección de anomalías sin depender de reglas específicas de un dominio concreto. Este requisito motiva una capa de adaptación que actúa como la conexión entre el problema concreto y el resto de la lógica, como se describe en la Sección 3.3.5.
- **Reproducibilidad y trazabilidad**. Los resultados del sistema deben ser reproducibles y trazables, de forma que las ejecuciones puedan repetirse bajo las mismas condiciones experimentales y sea posible identificar los datos, parámetros, configuraciones y versiones empleados en cada experimento. Este requisito motiva el registro sistemático de la configuración experimental y de los resultados obtenidos, tal como se describe en la Sección 3.3.7.

- **Coste computacional acotado**. El sistema debe mantener un coste computacional compatible con los recursos disponibles. En particular, el entrenamiento y la evaluación del modelo deben poder realizarse en tiempos razonables, y el tamaño del modelo debe mantenerse acotado para limitar el consumo de memoria.

3.2. Diseño de la solución

3.2.1. Modelo generativo MPS

Para modelar la distribución de probabilidad de los datos utilizamos un modelo generativo basado en redes tensoriales (*Tensor Networks*) [11]. En concreto, se sigue el enfoque de modelado no supervisado con *Matrix Product States* (MPS) propuesto por Han et al. [13] y se adapta al problema de detección de anomalías. Por tanto, aunque el método original se plantea como un aprendizaje no supervisado, en este trabajo se emplea un enfoque semi-supervisado de una sola clase, entrenando al modelo únicamente con datos clasificados como benignos. No obstante, el procedimiento de entrenamiento del modelo se mantiene igual al propuesto en el enfoque original.

3.2.1.1. *Born Machine*

Los modelos generativos y la física cuántica comparten una característica relevante, y es que ambos tratan de modelar distribuciones de probabilidad en espacios de alta dimensionalidad. En la mecánica cuántica, el estado de un sistema se describe mediante una función de onda $\Psi(\mathbf{v})$, y la probabilidad de observar una configuración concreta $\mathbf{v} = (v_1, v_2, \ldots, v_N)$ viene dada por la regla de Born [23]:

$$P(\mathbf{v}) = \frac{|\Psi(\mathbf{v})|^2}{Z} \tag{3.1}$$

donde

$$Z = \sum_{\mathbf{v} \in \mathcal{V}} |\Psi(\mathbf{v})|^2 \tag{3.2}$$

es el factor de normalización. También nos podemos referir a este factor como la función de partición, para establecer una analogía con los modelos basados en energía como las máquinas de Boltzmann [24].

Definir la probabilidad a partir del cuadrado de una función presenta dos ventajas: asegura la no negatividad de $P(\mathbf{v})$ sin necesidad de imponer restricciones adicionales sobre los parámetros del modelo, y admite de forma natural una interpretación mecánico-cuántica [13]. Los modelos generativos que explotan esta representación basada en estados cuánticos, en los que la distribución se obtiene aplicando la regla de Born sobre una función de onda parametrizada, reciben el nombre de *Born Machines* [13]. El modelo empleado en este trabajo es una *Born Machine* en la que la función de onda $\Psi(\mathbf{v})$ se parametriza mediante redes tensoriales, en concreto un *Matrix Product State* (MPS).

3.2.1.2. *Matrix Product States* (MPS)

La representación explícita de una función de onda requiere en general un número de componentes que crece exponencialmente con el número de variables. En particular, para un sistema de N variables la función de onda Ψ asigna un coeficiente a cada configuración completa $\mathbf{v} = (v_1, \ldots, v_N)$. Por tanto, si cada variable v_k puede tomar d_k valores posibles, una representación explícita de Ψ requiere almacenar $\prod_{k=1}^{N} d_k$ coeficientes.

Sin embargo, en el ámbito de la mecánica cuántica se han desarrollado técnicas que permiten representar de forma eficiente algunas familias de funciones de onda de alta dimensionalidad. Una de estas técnicas son las redes tensoriales (*Tensor Networks*, TN) [11], que consisten en factorizar vectores, operadores u otros objetos de alta dimensión como una red de tensores de menor orden conectados mediante índices comunes.

En la notación gráfica habitual, cada tensor se representa como un nodo y cada índice del tensor como una línea que sale de dicho nodo. Las líneas que quedan libres corresponden a índices físicos (*physical indices*), es decir, a las variables observables del sistema. Por otro lado, las líneas que conectan los tensores corresponden a índices internos o índices de enlace (*bond indices*). Contraer un índice de enlace significa sumar sobre todos los valores posibles de dicho índice. De este modo, la red completa representa el mismo objeto de alta dimensionalidad que una descripción explícita, pero lo hace a partir de una colección de elementos más pequeños. La Figura 1 ilustra algunos de los tipos de redes tensoriales más comunes.

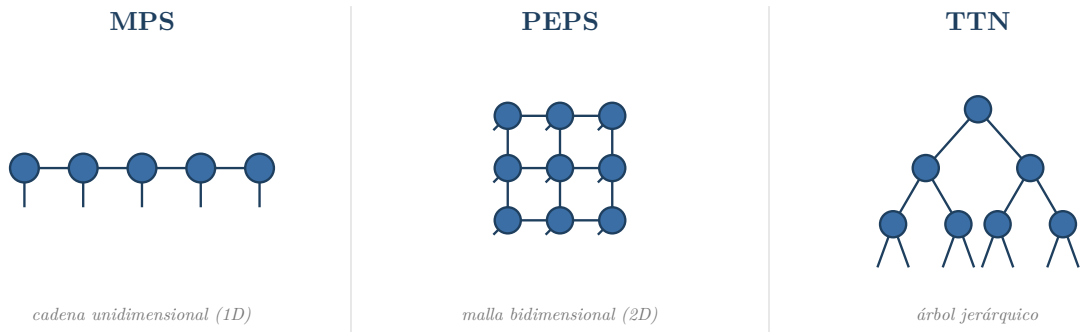


Figura 1: Distintos tipos de redes tensoriales. **MPS** (*Matrix Product State*): cadena unidimensional y la estructura empleada en este trabajo. **PEPS** (*Projected Entangled Pair States*): generalización a una malla bidimensional. **TTN** (*Tree Tensor Network*): red jerárquica en forma de árbol. Las líneas abiertas representan índices físicos y las líneas internas, índices de enlace.

En concreto, en este trabajo parametrizamos la función de onda utilizando un *Matrix Product State* (MPS) [11], una de las formas más simples y utilizadas de red tensorial. Un MPS descompone el tensor de coeficientes de una función de onda en una cadena unidimensional de N tensores, asociando un tensor a cada una de las N variables del sistema. Cada tensor posee

un índice físico v_k de dimensión d_k, que representa los d_k posibles estados de esa variable, junto con uno o dos índices de enlace que lo conectan con sus vecinos en la cadena. En este trabajo se utiliza un MPS con condiciones de contorno abiertas, por lo que el primer y el último tensor no están conectados entre sí y tienen un único índice de enlace.

Matemáticamente, una función de onda parametrizada mediante MPS con condiciones de contorno abiertas es:

$$\Psi(v_1, v_2, \ldots, v_N) = \sum_{\alpha_1,\ldots,\alpha_{N-1}} A^{(1)}_{v_1\alpha_1} A^{(2)}_{\alpha_1 v_2 \alpha_2} \cdots A^{(N)}_{\alpha_{N-1} v_N}. \tag{3.3}$$

donde cada $A^{(k)}$ es un tensor del MPS asociado a la variable v_k. Cada tensor posee un índice físico v_k, cuya dimensión d_k se denomina dimensión física, y uno o dos índices de enlace, α_{k-1} y α_k, cuyas dimensiones D_{k-1} y D_k se denominan dimensiones de enlace. En condiciones de contorno abiertas, los tensores de los extremos solo tienen un índice de enlace, lo que puede interpretarse equivalentemente tomando $D_0 = D_N = 1$. Para una observación concreta $\mathbf{v} = (v_1, \ldots, v_N)$, el valor de cada característica v_k selecciona una matriz del tensor correspondiente, que será un vector en el caso del primer y el último tensor. La contracción de estas matrices y vectores a lo largo de la cadena nos devuelve el escalar $\Psi(\mathbf{v})$.

Para simplificar la notación durante el resto del texto, se utilizará una forma matricial equivalente de la expresión anterior, convención estándar en la literatura [21]. Al evaluar el MPS en una observación concreta, cada valor v_k selecciona una matriz del tensor $A^{(k)}$ (un vector en los extremos), que denotamos $A^{(k)}[v_k]$. De este modo, los índices de enlace α_k quedan implícitos y la amplitud se expresa simplemente como un producto de matrices:

$$\Psi(\mathbf{v}) = A^{(1)}[v_1]\, A^{(2)}[v_2] \cdots A^{(N)}[v_N], \tag{3.4}$$

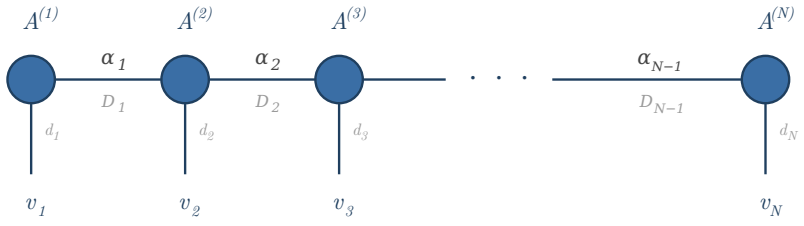


Figura 2: Representación gráfica de un *Matrix Product State* (MPS) con condiciones de contorno abiertas. Cada tensor $A^{[j]}$ se asocia a una variable del sistema y posee un índice físico s_j de dimensión d (líneas verticales) y uno o dos índices de enlace α_k de dimensión χ (líneas horizontales), que se contraen a lo largo de la cadena. Los tensores de los extremos cuentan con un único índice de enlace.

Una de las principales ventajas de la parametrización mediante MPS es su eficiencia para representar la función de onda con un número menor de parámetros. Mientras que una representación explícita de la función de onda requiere almacenar $\prod_{k=1}^{N} d_k$ coeficientes, un MPS con condiciones de contorno abiertas requiere almacenar

$$d_1 D_1 + \sum_{k=2}^{N-1} D_{k-1} d_k D_k + D_{N-1} d_N$$

parámetros. Si denotamos por $d_{\text{máx}}$ la mayor dimensión física y por $D_{\text{máx}}$ la mayor dimensión de enlace, este número queda acotado por $\mathcal{O}(N d_{\text{máx}} D_{\text{máx}}^2)$. Por tanto, siempre que $d_{\text{máx}}$ y $D_{\text{máx}}$ se mantengan acotados o crezcan moderadamente con N, la representación escala de forma polinómica con el número de variables. Sin embargo, esta reducción de parámetros conlleva un compromiso, puesto que las dimensiones de enlace determinan la cantidad de correlaciones que el MPS puede capturar entre distintas partes del sistema [20]. Por tanto, modelos con valores pequeños de D_k producen representaciones más compactas y computacionalmente eficientes, mientras que valores más grandes aumentan la capacidad expresiva de la red a costa de un mayor número de parámetros y un mayor coste de contracción.

3.2.1.3. Cálculo exacto de Z

Otra ventaja de la parametrización mediante MPS es que, a diferencia de otros modelos generativos, la función de partición Z se puede calcular de manera exacta y eficiente [13]. En los modelos basados en energía como las máquinas de Boltzmann [24] la función de partición suele ser intratable y obliga a recurrir a aproximaciones, ya que la suma sobre todas las configuraciones posibles requiere recorrer $\prod_{k=1}^{N} d_k$ términos, un número que crece exponencialmente con N. En cambio, la estructura del MPS permite evitar esa suma explícita. Dado que $\Psi(\mathbf{v})$ es un producto de matrices (Ecuación 3.3), la función de partición Z se puede obtener contrayendo el MPS con su conjugado.

$$Z = \sum_{\mathbf{v}} |\Psi(\mathbf{v})|^2 = \sum_{\mathbf{v}} \Psi^*(\mathbf{v})\, \Psi(\mathbf{v}) = \langle \Psi | \Psi \rangle. \tag{3.5}$$

Gráficamente, esto equivale a colocar una copia conjugada del MPS sobre la red original y conectar entre sí los índices físicos correspondientes, como se observa en la Figura 3. Al contraer cada índice físico v_k se realiza localmente la suma sobre todos los valores posibles de la variable k. Como estas sumas se realizan tensor a tensor a lo largo de la cadena, no es necesario construir ni recorrer explícitamente el espacio completo de configuraciones.

En la práctica, la contracción se lleva a cabo recorriendo la cadena de izquierda a derecha y acumulando la información en un entorno parcial E, que incorpora un nuevo sitio en cada paso:

$$E^{(k)} = \sum_{v_k} \big(A^{(k)}[v_k]\big)^\dagger \, E^{(k-1)} \, A^{(k)}[v_k], \qquad E^{(0)} = 1, \qquad Z = E^{(N)}. \tag{3.6}$$

La clave de la eficiencia es que en ningún momento aparece un tensor de dimensión exponencial, puesto que el entorno E nunca supera el tamaño $D_k \times D_k$. Por tanto, si las dimensiones físicas y de enlace permanecen acotadas, el coste de calcular Z escala polinómicamente con N, en lugar de hacerlo exponencialmente [11], [21].

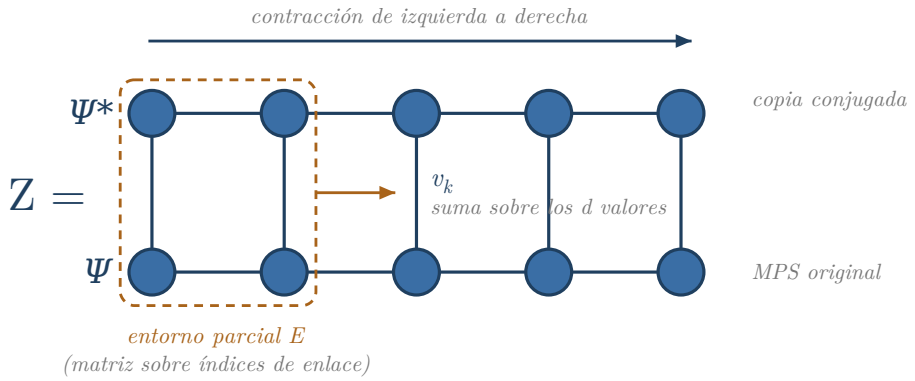


Figura 3: Cálculo de la función de partición $Z = \langle\Psi|\Psi\rangle$ de un MPS. Se coloca una copia conjugada (Ψ^*) sobre el MPS original (Ψ) y se conectan los índices físicos correspondientes (líneas verticales). Contraer cada uno suma localmente sobre los d_k valores de la variable. La contracción se realiza tensor a tensor a lo largo de la cadena, acumulando un entorno parcial E, que es una matriz sobre los índices de enlace, sin construir nunca un tensor de dimensión exponencial. Al no quedar índices abiertos, el resultado es el escalar Z.

3.2.1.4. Forma canónica del MPS

La representación de una función de onda como MPS no es única. Entre dos tensores contiguos puede insertarse una matriz invertible y su inversa, $G\,G^{-1} = I$, absorbiendo cada factor en uno de los tensores vecinos, sin que la función de onda $\Psi(\mathbf{v})$ cambie.

$$A^{(k)}[v_k]\,A^{(k+1)}[v_{k+1}] = A^{(k)}[v_k]\,\underbrace{G\,G^{-1}}_{=I}\,A^{(k+1)}[v_{k+1}] = \tilde{A}^{(k)}[v_k]\,\tilde{A}^{(k+1)}[v_{k+1}], \tag{3.7}$$

Esta libertad permite elegir distintas representaciones del MPS sin modificar el estado físico que representa [21].

Una de estas representaciones es la forma canónica [11], [21]. En esta forma canónica, existe un centro de ortogonalidad y todos los tensores situados a su izquierda y su derecha satisfacen condiciones de ortonormalidad, de manera que

$$\big(A^{(j)}\big)^{\dagger} A^{(j)} = I \quad (j < c), \qquad A^{(j)}\big(A^{(j)}\big)^{\dagger} = I \quad (j > c), \tag{3.8}$$

donde c es la posición del centro de ortogonalidad $A^{(c)}$.

Como consecuencia de la forma canónica, al calcular $Z = \langle\Psi|\Psi\rangle$, los tensores situados a la izquierda y a la derecha del centro de ortogonalidad se contraen con sus conjugados y, por las condiciones de ortonormalidad (Ecuación 3.8), colapsan sucesivamente a la identidad, como puede verse representado en la Figura 4. Por tanto, la contracción de toda la cadena se simplifica hasta reducirse a la norma al cuadrado del centro de ortogonalidad

$$Z = \langle\Psi|\Psi\rangle = \|A^{(c)}\|^2 = \sum_{\alpha_{c-1},\, v_c,\, \alpha_c} \left|A^{(c)}_{\alpha_{c-1}\, v_c\, \alpha_c}\right|^2. \tag{3.9}$$

De este modo, la forma canónica convierte un cálculo global en un cálculo local, lo que reduce en gran medida el coste computacional. Esta propiedad es la que explota el algoritmo de entrenamiento (Sección 3.2.3), que sitúa el centro de ortogonalidad sobre la pareja de sitios que se está optimizando. En la Sección 3.2.3.1 se explica el mecanismo mediante el cual el MPS se transforma y se mantiene en forma canónica.

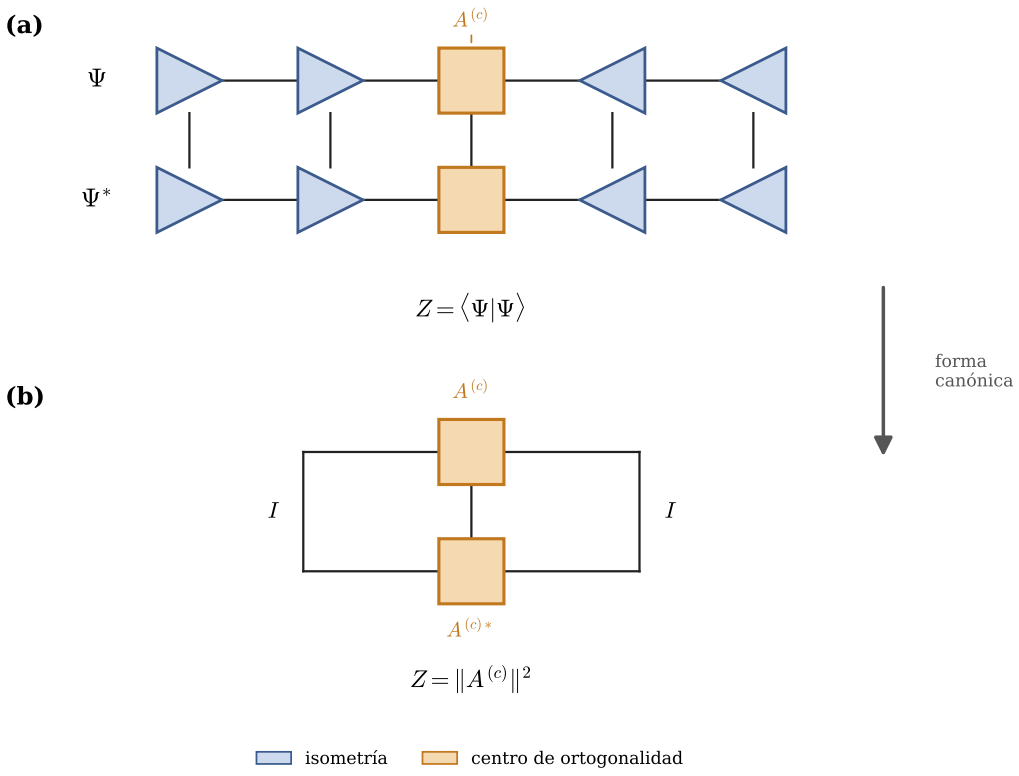


Figura 4: Simplificación del cálculo de $Z = \langle\Psi|\Psi\rangle$ en forma canónica. (a) Contracción completa: el MPS y su copia conjugada con los índices físicos conectados; los tensores a izquierda y derecha del centro de ortogonalidad son isometrías (triángulos orientados hacia el centro). (b) Al contraerse con sus conjugados, las isometrías laterales colapsan a la identidad (I), de modo que la contracción se reduce al bloque central unido a su conjugado por los índices físicos, lo que equivale a la norma al cuadrado del centro de ortogonalidad, $Z = \|A^{(c)}\|^2$.

3.2.1.5. Puntuación de anomalía

El cálculo exacto de la función de partición Z permite obtener de manera exacta la probabilidad de cada observación mediante la regla de Born (Ecuación 3.1):

$$P(\mathbf{v}) = \frac{|\Psi(\mathbf{v})|^2}{Z}, \qquad Z = \sum_{\mathbf{v}} |\Psi(\mathbf{v})|^2.$$

Como el modelo se entrena exclusivamente con observaciones benignas y aprende su distribución, asigna probabilidades altas a las observaciones que se ajustan a esa distribución del comportamiento normal y probabilidades bajas a las que se alejan de ella. Esto permite emplear la propia probabilidad como criterio para la detección de anomalías.

Como puntuación de anomalía, se emplea la verosimilitud logarítmica negativa (*negative log-likelihood*, NLL) [13]:

$$s(\mathbf{v}) = \mathrm{NLL}(\mathbf{v}) = -\log P(\mathbf{v}). \tag{3.10}$$

La NLL invierte la escala de la probabilidad, de manera que cuanto menos probable es una observación bajo el modelo, mayor es su puntuación. Además, trabajar en el dominio logarítmico mejora la estabilidad numérica del cálculo, ya que reduce los problemas numéricos derivados de manejar probabilidades extremadamente pequeñas.

Sustituyendo la probabilidad obtenida con la regla de Born, la puntuación de anomalía puede expresarse como:

$$s(\mathbf{v}) = \mathrm{NLL}(\mathbf{v}) = -\log P(\mathbf{v}) = \log Z - \log |\Psi(\mathbf{v})|^2. \tag{3.11}$$

Esta expresión relaciona directamente la puntuación de anomalía con la función de onda aprendida por el MPS. Para un valor fijo de Z, una observación con una amplitud $|\Psi(\mathbf{v})|^2$ alta tendrá una NLL baja y, por tanto, será considerada compatible con el comportamiento normal. Por el contrario, una observación con una amplitud baja dará lugar a una NLL elevada, lo que indica que el modelo la considera poco probable y, en consecuencia, potencialmente anómala.

3.2.2. Codificación de características

El modelo MPS opera sobre índices físicos discretos. Cada sitio del MPS está asociado a una característica de la observación y recibe, para dicha característica, un índice entero en el rango $[0, d_k)$, que selecciona una de las d_k componentes del tensor $A^{(k)}$.

Sin embargo, las características de las observaciones son heterogéneas, ya que pueden incluir variables categóricas, binarias y numéricas. Por este motivo, es necesario transformar cada una de ellas en un índice entero compatible con la representación del MPS. Esta etapa es la que da respuesta al requisito de robustez frente a características heterogéneas descrito en la Sección 3.1.3.2.

3.2.2.1. Principios de diseño

La codificación se rige por dos principios de diseño que se derivan directamente de los requisitos del sistema.

El primer principio es que el codificador se ajusta exclusivamente sobre las observaciones benignas del conjunto de entrenamiento. Esto significa que todas las decisiones que dependen de los datos, como la construcción de vocabulario de las variables categóricas o la definición de los límites de los intervalos de discretización de las variables numéricas, se toman observando únicamente las observaciones benignas del conjunto de entrenamiento. Este principio responde a la naturaleza semi-supervisada del sistema, puesto que el modelo no debe tener información sobre las anomalías ni el conjunto de test.

El segundo principio es el determinismo. Igual que el resto del sistema, el codificador debe asegurar la reproducibilidad y la trazabilidad exigidas por los requisitos. En este caso, dado un mismo conjunto de entrenamiento, el codificador siempre producirá el mismo esquema de codificación y transformará de manera consistente cualquier nueva observación.

3.2.2.2. Política de asignación de tipo

El codificador no aplica el mismo tratamiento a todas las características, sino que clasifica cada característica en uno de cuatro tipos y le aplica la regla de codificación correspondiente. Las variables que representan valores simbólicos o etiquetas sin una relación ordinal o métrica se asignan al tipo categórico, ya que su información reside en la identidad del valor y no en una distancia numérica entre categorías.

Las variables numéricas se tratan de forma diferente según su variabilidad en el conjunto de entrenamiento benigno. En primer lugar, aquellas que siempre toman el mismo valor en el conjunto de entrenamiento se asignan al tipo constante. En esta categoría también se incluyen aquellas que toman el mismo valor en casi todas las observaciones, definiendo çasi todasçomo una proporción mayor a un umbral definido en la configuración del codificador. En segundo lugar, las variables numéricas que toman un número reducido de valores distintos en el conjunto de entrenamiento se asignan al tipo discreto. En este caso, se considera que el número de valores distintos es reducido cuando no supera un umbral definido en la configuración del codificador. Finalmente, el resto de variables numéricas se asignan al tipo numérico. Estas variables presentan una variabilidad mayor y, por tanto, requieren un proceso específico de discretización antes de poder ser utilizadas como índices físicos del MPS.

3.2.2.3. Codificación de cada tipo

Una vez asignado el tipo de cada característica, la conversión de un valor concreto en un índice entero se realiza según la regla de codificación correspondiente. En las características categóricas se construye un vocabulario a partir de los valores observados en el conjunto de entrenamiento, de modo que cada valor recibe un índice distinto. La dimensión física resultante es igual al número de valores distintos observados más uno, ya que se reserva

un índice adicional para representar valores no vistos durante el ajuste del codificador. Las características discretas siguen la misma lógica, pero su vocabulario está formado por los distintos valores numéricos presentes en el entrenamiento benigno.

Las características constantes se codifican de forma binaria con dimensión física dos. El índice 0 indica que el valor es igual al valor característico del conjunto de entrenamiento, y el índice 1 indica que el valor es distinto. De este modo, el modelo recibe directamente la señal de si la característica mantiene el comportamiento observado o se desvía de él.

Finalmente, las características numéricas con mayor variabilidad se discretizan en intervalos, cuyos límites se obtienen a partir de cuantiles de la distribución de entrenamiento, añadiendo límites infinitos en los intervalos extremos. El uso de cuantiles, en lugar de intervalos de igual anchura, hace que cada intervalo contenga aproximadamente la misma proporción de observaciones benignas. Esto resulta especialmente adecuado para las características con distribuciones de cola larga, en las que una discretización de anchura uniforme dejaría algunos intervalos vacíos o sobrecargados. El índice asignado a cada valor corresponde al intervalo en el que cae dicho valor.

3.2.3. Algoritmo de entrenamiento DMRG

El modelo se entrena por máxima verosimilitud, de forma que los tensores del MPS se ajustan para maximizar la probabilidad que el modelo asigna al conjunto de entrenamiento. Por la monotonía del logaritmo, maximizar la verosimilitud equivale a minimizar la log-verosimilitud negativa (NLL) media sobre dicho conjunto, que se adopta como función de pérdida. Utilizando la definición de la NLL propuesta en la Sección 3.2.1.5 y denotando por B un *minibatch* de observaciones normales, se tiene que la función objetivo $\mathcal{L}$ es:

$$\mathcal{L} = \frac{1}{|B|}\sum_{\mathbf{v}\in B}\mathrm{NLL}(\mathbf{v}) = \frac{1}{|B|}\sum_{\mathbf{v}\in B}\left(\log Z - \log|\Psi(\mathbf{v})|^2\right) = \log Z - \frac{1}{|B|}\sum_{\mathbf{v}\in B}\log|\Psi(\mathbf{v})|^2 \quad (3.12)$$

En lugar de aplicar descenso de gradiente sobre todos los tensores del MPS a la vez, el entrenamiento emplea un algoritmo de barrido inspirado en el *Density Matrix Renormalization Group* [21], adaptado al entrenamiento generativo de MPS siguiendo el enfoque de Han et al. [13]. La idea central es optimizar la cadena por parejas de tensores contiguos, recorriendo la cadena de un extremo a otro de forma repetida. La principal ventaja de este enfoque es que, al separar de nuevo cada pareja de sitios mediante una descomposición en valores singulares, la dimensión de enlace D_k de ese punto de la cadena puede ajustarse de forma adaptativa a la correlación que realmente existe allí, en lugar de fijarse a priori a un valor uniforme para todos los enlaces [13]. La Figura 5 resume el funcionamiento general del algoritmo de entrenamiento DMRG, cuyos elementos principales se describen a continuación.

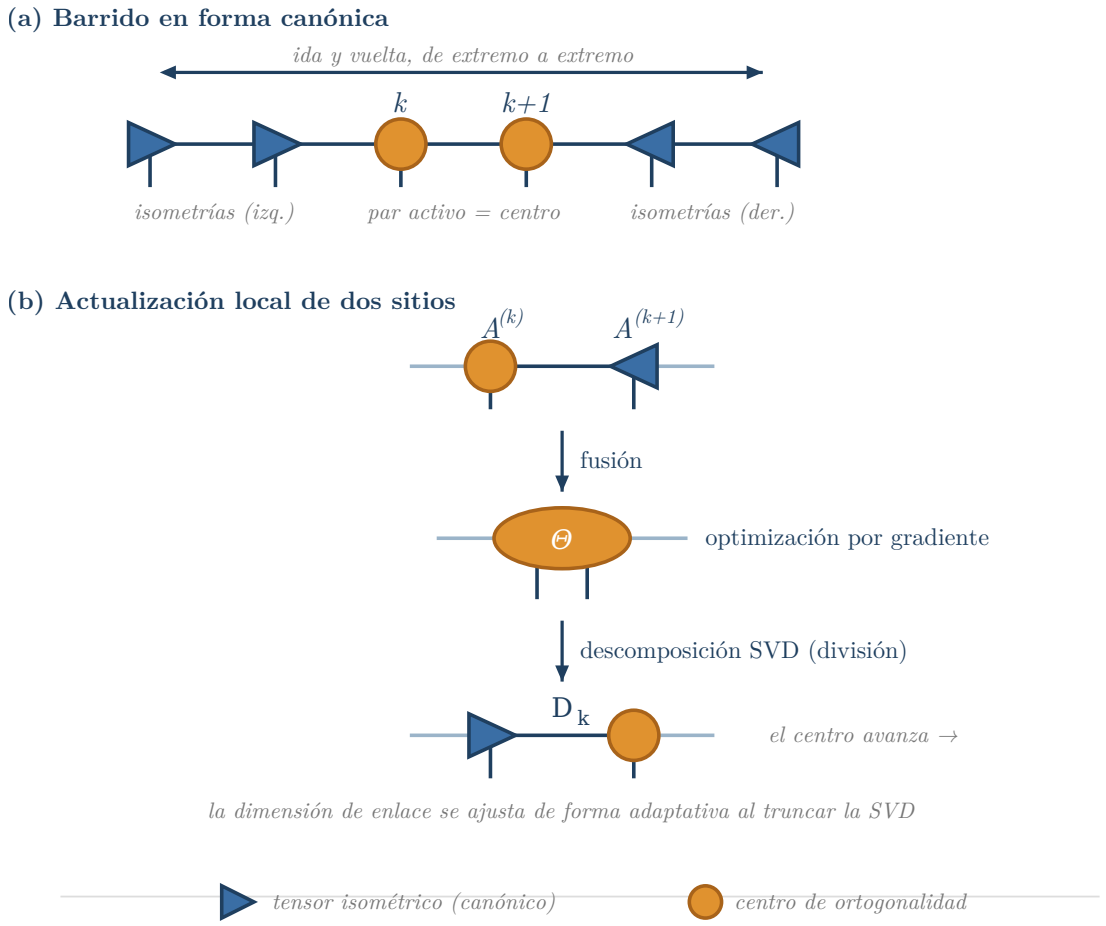


Figura 5: Esquema del algoritmo de barrido tipo DMRG. (a) El MPS se mantiene en forma canónica: los tensores a ambos lados del par activo son isometrías (triángulos orientados hacia el centro) y la optimización recorre la cadena de un extremo a otro de forma repetida. (b) En cada paso local, la pareja de sitios activa se fusiona en un único tensor Θ, se optimiza por descenso de gradiente y se vuelve a separar mediante una descomposición en valores singulares (SVD); el truncamiento de la SVD ajusta de forma adaptativa la dimensión de enlace D_k y el centro de ortogonalidad avanza un sitio.

3.2.3.1. Preparación del estado en forma canónica

Antes de comenzar el entrenamiento, el estado se transforma a su forma canónica mediante un procedimiento estándar basado en descomposiciones QR sucesivas [11], [21]. En esta canonización inicial se emplea la descomposición QR en lugar de la SVD porque en esta etapa solo se necesita ortonormalizar cada tensor, sin truncar ni examinar los valores singulares, y la QR resulta más económica y numéricamente estable [21].

Partiendo del extremo izquierdo de la cadena, cada tensor se reagrupa en forma matricial, agrupando $(\alpha_{k-1}v_k)$ como índice de fila y α_k como índice de columna. El tensor $A^{(k)}$ se trata como una matriz $\hat{A}^{(k)}$ y se factoriza como $\hat{A} = QR$. El factor ortonormal Q se conserva en esa posición, con lo que el tensor queda en forma canónica izquierda, mientras que el factor triangular R se transfiere al tensor contiguo, que lo absorbe.

$$\hat{A}^{(k)} = Q\,R, \qquad A^{(k)} \leftarrow Q, \qquad A^{(k+1)} \leftarrow R\,A^{(k+1)}, \qquad \left(A^{(k)}\right)^{\dagger} A^{(k)} = I. \tag{3.13}$$

Para canonizar de derecha a izquierda, en cambio, se agrupa $(v_k\alpha_k)$ como índice de columna y se factoriza la transpuesta conjugada, $\hat{A}^{\dagger} = QR$, de modo que $\hat{A} = R^{\dagger}Q^{\dagger}$. En este caso, el factor isométrico es $Q^{\dagger}$, que queda como tensor en forma canónica derecha, mientras que $R^{\dagger}$ se absorbe en el sitio contiguo.

$$\hat{A}^{(k)\dagger} = Q\,R, \qquad A^{(k)} \leftarrow Q^{\dagger}, \qquad A^{(k-1)} \leftarrow A^{(k-1)}\,R^{\dagger}, \qquad A^{(k)}\left(A^{(k)}\right)^{\dagger} = I. \tag{3.14}$$

En ambos casos, el factor triangular arrastra la norma hacia el extremo opuesto, que se convierte en el centro de ortogonalidad.

3.2.3.2. Actualización local de dos sitios

El algoritmo actúa sobre un único enlace k, que une los sitios k y $k+1$. En primer lugar, ambos tensores se fusionan en un único tensor Θ de orden cuatro, que conserva los índices de enlace externos y los dos índices físicos. Sobre este tensor fusionado se pueden aplicar uno o varios pasos de descenso de gradiente. Una vez actualizado, Θ se vuelve a separar en dos tensores mediante una descomposición en valores singulares (SVD). Durante esta descomposición se truncan los valores singulares más pequeños, lo que fija la nueva dimensión de enlace D_k [13], [22].

En cada actualización local, el centro de ortogonalidad se sitúa sobre la pareja de sitios $(k, k+1)$ que se está optimizando, fusionada en un único tensor de dos sitios Θ.

$$\Theta[v_k, v_{k+1}] = A^{(k)}[v_k]\,A^{(k+1)}[v_{k+1}] \tag{3.15}$$

Puesto que todos los tensores a su izquierda y derecha cumplen condiciones de ortonormalidad, Θ es el centro de ortogonalidad y la función de partición se reduce localmente a su norma al cuadrado, $Z = \|\Theta\|^2$ (Ecuación 3.9), lo que reducirá el coste computacional del cálculo del gradiente.

Entornos

Para reducir aún más el coste de cada actualización local se emplean entornos [21], [22]. El entorno izquierdo $L_{\mathbf{v}}$ de un enlace es la contracción de todos los sitios situados a su izquierda para una observación $\mathbf{v}$, y el entorno derecho $R_{\mathbf{v}}$ la de todos los sitios situados a su derecha.

$$L_{\mathbf{v}} = A^{(1)}[v_1]\, A^{(2)}[v_2] \cdots A^{(k-1)}[v_{k-1}], \tag{3.16}$$

$$R_{\mathbf{v}} = A^{(k+2)}[v_{k+2}] \cdots A^{(N)}[v_N], \tag{3.17}$$

donde los índices físicos quedan fijados a los valores de la observación $\mathbf{v} = (v_1, v_2, \ldots, v_N)$. Con ellos, la amplitud se factoriza a través del centro como

$$\Psi(\mathbf{v}) = L_{\mathbf{v}}\, \Theta[v_k, v_{k+1}]\, R_{\mathbf{v}}, \tag{3.18}$$

Estas contracciones parciales se almacenan en memoria y se actualizan de forma incremental a medida que el centro de ortogonalidad se desplaza durante el barrido. Por ejemplo, al avanzar el centro hacia la derecha, el entorno izquierdo crece incorporando un único sitio,

$$L^{(j)}_{\mathbf{v}} = L^{(j-1)}_{\mathbf{v}}\, A^{(j)}[v_j], \tag{3.19}$$

y de forma análoga el entorno derecho al avanzar hacia la izquierda. De este modo, pasar de un enlace al siguiente no requiere volver a contraer la cadena completa, sino únicamente actualizar el entorno afectado.

Gradiente de la función de pérdida

El gradiente de la función objetivo respecto al tensor fusionado Θ admite una forma cerrada [13]. Para obtenerlo basta derivar por separado los dos términos de la función de pérdida (Ecuación 3.12)

$$\mathcal{L} = \log Z - \frac{1}{|B|} \sum_{\mathbf{v} \in B} \log |\Psi(\mathbf{v})|^2$$

A diferencia del planteamiento original de Han et al. [13], que restringe la amplitud a valores reales, aquí los tensores del MPS y la amplitud $\Psi(\mathbf{v})$ pueden ser complejos. Como $\mathcal{L}$ es una función real de un parámetro complejo, su derivada se formula mediante el cálculo de Wirtinger, tratando Θ y $\overline{\Theta}$ como variables independientes [35]. La dirección de descenso es entonces la derivada respecto al conjugado, que se reduce al gradiente ordinario en el caso real.

$$G \equiv 2\, \frac{\partial \mathcal{L}}{\partial \overline{\Theta}} \tag{3.20}$$

El primer término del gradiente procede de la derivada de $\log Z$. Gracias a que el MPS está en forma canónica con el centro de ortogonalidad sobre Θ, la función de partición se reduce a $Z = \|\Theta\|^2 = \langle\Theta|\Theta\rangle = \sum_i \Theta_i \overline{\Theta_i}$ (Ecuación 3.9). Por tanto, la derivada de este término respecto a $\overline{\Theta}$ es inmediata.

$$\frac{\partial \log Z}{\partial \overline{\Theta}} = \frac{1}{Z}\frac{\partial Z}{\partial \overline{\Theta}} = \frac{\Theta}{Z} \tag{3.21}$$

Este término actúa como término de normalización, puesto que empuja la amplitud del modelo a la baja de manera global para que la probabilidad total siga sumando uno.

En segundo lugar, el término de datos procede de derivar $\log|\Psi(\mathbf{v})|^2 = \log\left(\Psi(\mathbf{v})\,\overline{\Psi(\mathbf{v})}\right)$. Aplicando la regla de la cadena:

$$\frac{\partial}{\partial \overline{\Theta}} \log\left(\Psi(\mathbf{v})\,\overline{\Psi(\mathbf{v})}\right) = \frac{1}{\Psi(\mathbf{v})\,\overline{\Psi(\mathbf{v})}} \frac{\partial\left(\Psi(\mathbf{v})\,\overline{\Psi(\mathbf{v})}\right)}{\partial \overline{\Theta}} = \frac{1}{\Psi(\mathbf{v})\,\overline{\Psi(\mathbf{v})}} \left(\frac{\partial \Psi(\mathbf{v})}{\partial \overline{\Theta}}\,\overline{\Psi(\mathbf{v})} + \Psi(\mathbf{v})\,\frac{\partial \overline{\Psi(\mathbf{v})}}{\partial \overline{\Theta}}\right) \tag{3.22}$$

La factorización por entornos (Ecuación 3.18) muestra que $\Psi(\mathbf{v}) = L_{\mathbf{v}}\,\Theta[v_k, v_{k+1}]\,R_{\mathbf{v}}$ depende linealmente de Θ y no depende de su conjugado $\overline{\Theta}$. De forma análoga, $\overline{\Psi(\mathbf{v})}$ depende únicamente de $\overline{\Theta}$. Por tanto,

$$\frac{\partial \Psi(\mathbf{v})}{\partial \overline{\Theta}} = 0, \qquad \frac{\partial \overline{\Psi(\mathbf{v})}}{\partial \overline{\Theta}} = \overline{L_{\mathbf{v}} \otimes R_{\mathbf{v}}}. \tag{3.23}$$

Sustituyendo en la Ecuación 3.22:

$$\frac{\partial}{\partial \overline{\Theta}} \log\left(\Psi(\mathbf{v})\,\overline{\Psi(\mathbf{v})}\right) = \frac{1}{\Psi(\mathbf{v})\,\overline{\Psi(\mathbf{v})}} \left(\underbrace{\frac{\partial \Psi(\mathbf{v})}{\partial \overline{\Theta}}}_{=0}\,\overline{\Psi(\mathbf{v})} + \Psi(\mathbf{v})\,\frac{\partial \overline{\Psi(\mathbf{v})}}{\partial \overline{\Theta}}\right) = \frac{\overline{L_{\mathbf{v}} \otimes R_{\mathbf{v}}}}{\overline{\Psi(\mathbf{v})}}. \tag{3.24}$$

Este segundo término es el que incrementa la amplitud en las configuraciones presentes en el *minibatch*, ponderando cada una por un factor inversamente proporcional a la amplitud que el modelo le asigna. De este modo, las observaciones a las que el modelo todavía asigna poca amplitud reciben una corrección mayor.

Reuniendo ambas contribuciones según los signos de la función de pérdida, promediando el término de datos sobre el minibatch y adoptando la dirección de descenso $G = 2\,\partial\mathcal{L}/\partial\overline{\Theta}$ se obtiene la forma cerrada:

$$G = \frac{2\,\Theta}{Z} - \frac{2}{|B|}\sum_{\mathbf{v}\in B} \frac{\overline{L_{\mathbf{v}} \otimes R_{\mathbf{v}}}}{\overline{\Psi(\mathbf{v})}}. \tag{3.25}$$

El equilibrio entre ambos términos es lo que conduce al modelo a concentrar amplitud, y por tanto probabilidad, en las regiones compatibles con el tráfico normal, y a retirarla del resto. Este gradiente se aplica durante n_{desc} pasos sucesivos sobre Θ antes de separar de nuevo la pareja de sitios.

Separación por SVD

Una vez optimizado el tensor de dos sitios Θ, se divide de nuevo en dos tensores mediante una descomposición en valores singulares [21], [22]. Para ello se reagrupan sus índices de modo que Θ se trata como una matriz, con $(\alpha_{k-1}v_k)$ como índice de fila y $(v_{k+1}\alpha_{k+1})$ como índice de columna, y se factoriza mediante una descomposición en valores singulares:

$$\Theta = U\,S\,V^{\dagger}, \tag{3.26}$$

La SVD garantiza que U y V son isometrías, es decir, $U^{\dagger}U = I$ y $V^{\dagger}V = I$, mientras que S contiene los valores singulares. Esta propiedad es la que permite recuperar la forma canónica sin ningún paso adicional [11], [21].

Si el barrido avanza hacia la derecha, se define

$$A^{(k)} \leftarrow U, \qquad A^{(k+1)} \leftarrow S\,V^{\dagger}, \tag{3.27}$$

Como $A^{(k)}$ es precisamente el factor isométrico U, hereda su ortonormalidad y queda en forma canónica izquierda de forma que $\left(A^{(k)}\right)^{\dagger}A^{(k)} = I$, mientras que los valores singulares se absorben en $A^{(k+1)}$, que se convierte en el nuevo centro de ortogonalidad. De este modo, el centro queda desplazado un sitio hacia la derecha.

Por el contrario, si el barrido avanza hacia la izquierda, se absorben los valores singulares en el tensor izquierdo:

$$A^{(k)} \leftarrow U\,S, \qquad A^{(k+1)} \leftarrow V^{\dagger}, \tag{3.28}$$

En este caso, $A^{(k+1)}$ queda en forma canónica derecha, $A^{(k+1)}\left(A^{(k+1)}\right)^{\dagger} = I$, y el centro de ortogonalidad queda desplazado hacia la izquierda.

La SVD no sólo vuelve a separar el tensor, sino que también nos proporciona información sobre el grado de correlación entre las dos mitades de la cadena separadas por el enlace k. Para verlo, basta reescribir (3.26) como una suma de productos externos:

$$\Theta = \sum_{i=1}^{\chi} s_i\,\mathbf{u}_i\,\mathbf{v}_i^{\dagger}, \tag{3.29}$$

donde $\mathbf{u}_i$ son las columnas de U, $\mathbf{v}_i^{\dagger}$ las filas de $V^{\dagger}$ y los s_i son los valores singulares (las entradas de S), ordenados de mayor a menor. Cada término de la suma empareja un patrón del bloque izquierdo $(\mathbf{u}_i)$ con uno del bloque derecho $(\mathbf{v}_i^{\dagger})$, y el valor singular s_i mide el peso de esa pareja, de forma que cuanto mayor es s_i, más contribuye esa combinación a reconstruir Θ. Por tanto, un único s_i grande seguido de muchos muy pequeños indica que basta una pareja para describir el tensor sin mucho error, es decir, que las dos mitades apenas están correlacionadas y que casi toda la información se concentra en unos pocos términos. El error cometido al conservar únicamente los primeros D' valores singulares se mide con el peso descartado (*discarded weight*) [21]:

$$\varepsilon_{D'} \;=\; \frac{\sum_{i>D'} s_i^2}{\sum_i s_i^2} \;=\; 1 - \frac{\sum_{i\le D'} s_i^2}{\|\Theta\|_F^2}, \tag{3.30}$$

Por otro lado, que muchos valores singulares tengan magnitud parecida indica una correlación fuerte, porque hacen falta muchas parejas para reconstruir Θ. Esta forma de entender la SVD se conoce como descomposición de Schmidt en mecánica cuántica [11], [21], y es la que justifica el truncamiento de los valores singulares más pequeños, así como las columnas de U y las filas de $V^\dagger$ asociadas, después de cada descomposición. Al hacerlo, no solo se aproxima el tensor, sino que también se fija la nueva dimensión de enlace, que queda determinada por el número de valores singulares mantenidos tras el truncamiento.

En la práctica, el rango conservado en cada enlace se fija combinando dos criterios. En primer lugar, se descartan aquellos valores singulares con $s_i/s_1 < \delta$ [13], [22], donde δ es un umbral prefijado. De esta manera se eliminan los términos numéricamente despreciables frente al valor singular más dominante. Este criterio evita arrastrar direcciones que no aportan información y que sólo introducirían ruido numérico. Además, se impone una cota a la nueva dimensión de enlace D_k [13], [22], de forma que aunque queden más valores por encima del umbral, nunca se retienen más de $D_{\text{máx}}$. Este segundo criterio acota explícitamente el coste computacional y el uso de memoria.

La cota sobre las dimensiones de enlace no es estática, sino que comienza en un valor reducido D_{init} y crece de forma multiplicativa a lo largo del entrenamiento hasta alcanzar como máximo $D_{\text{máx}}$. El crecimiento no es automático en cada barrido, sino que está condicionado a que durante varios barridos consecutivos algún enlace haya alcanzado el valor de la cota actual, lo que indica que la SVD habría conservado más rango si se le hubiera permitido, o que el peso descartado en el truncamiento haya superado un umbral prefijado, lo que indica que el truncamiento está eliminando información no despreciable. De esta forma, la dimensión de enlace se ajusta a la complejidad real de los datos, evitando dimensiones innecesariamente grandes [13].

Además de acotar el coste computacional y de memoria, este mecanismo actúa como una forma de regularización, puesto que la capacidad del modelo solo aumenta cuando existe evidencia sostenida de que es necesaria, lo que reduce el riesgo de que el modelo crezca para ajustar ruido.

Pseudocódigo de la actualización local de dos sitios

Reuniendo la fusión de los dos tensores, el descenso de gradiente y la separación por SVD, el algoritmo completo de la actualización local de dos sitios queda recogido en el siguiente pseudocódigo. El apartado (b) de la Figura 5 ofrece además una representación visual del proceso.

Algoritmo 1 Actualización local de dos sitios

Require: Enlace k; dirección del barrido; tasa de aprendizaje η; entornos L, R; minibatch B; cota $D_{\text{máx}}$; umbral SVD δ

Fusionar los sitios k y $k+1$ en un único tensor Θ
for n_{desc} pasos **do**
    Calcular el gradiente G en forma cerrada (Ec. 3.25)
    if G no es finito **then**
        Descartar el paso y continuar
    else
        $\Theta \leftarrow \Theta - \eta\, G$
    end if
end for
Reagrupar Θ como matriz y descomponer $\Theta = USV^{\dagger}$ (SVD)
Truncar los valores singulares con $s_i/s_1 < \delta$, sin superar $D_{\text{máx}}$
if el barrido avanza a la derecha **then**
    $A^{(k)} \leftarrow U$; $A^{(k+1)} \leftarrow S\,V^{\dagger}$ ▷ centro a la derecha
else
    $A^{(k)} \leftarrow U\,S$; $A^{(k+1)} \leftarrow V^{\dagger}$ ▷ centro a la izquierda
end if
Calcular el peso descartado en el truncamiento (Ec. 3.30)
Actualizar de forma incremental el entorno que crece

3.2.3.3. Barrido y bucle de entrenamiento

Un barrido consiste en aplicar la actualización local a todos los enlaces de la cadena de forma ordenada. Un barrido a la derecha recorre los enlaces desde $k = 0$ hasta $k = N - 2$, actualizando en cada paso los tensores k y $k+1$, y desplazando el centro de ortogonalidad hacia la derecha. Un barrido a la izquierda los recorre en sentido inverso. Por cada *minibatch* se realiza un barrido a la derecha seguido de uno a la izquierda, de manera que todos los enlaces se actualizan dos veces. Una época de entrenamiento procesa de este modo todos los *minibatches*. Al final de cada época el estado se renormaliza y se recanonicaliza, y se mide la NLL sobre el conjunto de entrenamiento y sobre el de validación si está disponible para monitorizar el progreso.

El entrenamiento incorpora varios mecanismos con el propósito de que se mantenga estable. En primer lugar, la tasa de aprendizaje se reduce de forma adaptativa cuando la NLL deja de mejorar durante un número de épocas igual a la paciencia, multiplicándose por un factor de reducción ρ menor que uno. Si la tasa de aprendizaje cae por debajo de un valor mínimo, el entrenamiento se detiene. Además, se contempla una parada temprana si la NLL no mejora durante un número prolongado de épocas. Finalmente, si todos los gradientes de varias épocas consecutivas resultan no finitos, situación en la que el entrenamiento no puede progresar, se aborta el entrenamiento.

A lo largo del proceso se conserva una copia del mejor modelo observado según la métrica de seguimiento. Esta copia se restaura al terminar, de modo que el modelo final no es el de la última época sino el mejor de todos.

Integrando la canonización inicial del modelo, los barridos sobre todos los enlaces, el crecimiento adaptativo de la dimensión de enlace y los mecanismos de estabilidad del entrenamiento, el algoritmo completo del bucle de entrenamiento queda de la siguiente manera. El apartado (a) de la Figura 5 ofrece además una representación visual del proceso.

Algoritmo 2 Bucle de entrenamiento DMRG

Require: Datos de entrenamiento; cota inicial D_{init}; cota máxima $D_{\text{máx}}$; tasa de aprendizaje inicial η_0; número de loops n_{loops}

Normalizar y canonicalizar el MPS
$D \leftarrow D_{\text{init}}$; $\eta \leftarrow \eta_0$; mejor $\leftarrow \infty$
for loop $= 1$ a n_{loops} **do**
    for all minibatch B **do**
        Construir los entornos L, R
        Barrido a la derecha: Algoritmo 1 en $k = 0, \ldots, N-2$
        Reconstruir los entornos L, R
        Barrido a la izquierda: Algoritmo 1 en $k = N-2, \ldots, 0$
    end for
    Renormalizar y recanonicalizar el MPS
    Medir la NLL
    if la NLL mejora **then**
        Guardar copia del modelo; reiniciar la paciencia
    else
        Incrementar la paciencia; si supera el límite, $\eta \leftarrow \eta \cdot \rho$
    end if
    if la cota D ha sido limitante durante varios loops seguidos **then**
        $D \leftarrow \min\big(D_{\text{máx}}, \lceil D \cdot f \rceil\big)$
    end if
end for
Restaurar la mejor copia del modelo guardada

3.2.4. Mecanismo de explicabilidad

Los modelos de aprendizaje profundo habituales en detección de anomalías, como los autoencoders o las redes generativas adversarias, suelen operar mediante millones de parámetros distribuidos en capas no lineales. Por ello, para interpretar sus decisiones se necesitan herramientas externas y aproximadas como LIME [8] o SHAP [9].

En cambio, en el MPS la explicación de una decisión puede obtenerse directamente a partir de los tensores entrenados. Su estructura factorizada permite calcular de forma exacta

y eficiente distintas cantidades interpretables mediante contracciones parciales del propio modelo, sin introducir un mecanismo explicativo adicional.

El conjunto de análisis que se describe a continuación sigue el propuesto por Aizpurua, Palmer y Orús [15] para la interpretabilidad de MPS en ciberseguridad, y lo amplía y corrige en varios puntos que se irán señalando.

3.2.4.1. La matriz de densidad reducida

El modelo no define directamente una distribución de probabilidad, sino una función de onda $\Psi(\mathbf{v})$ sobre todas las configuraciones posibles $\mathbf{v} = (v_1, v_2, \ldots, v_N)$ de las N características, y la probabilidad de una configuración concreta se obtiene mediante la regla de Born (Ecuación 3.1), $P(\mathbf{v}) = |\Psi(\mathbf{v})|^2/Z$.

El estado completo del modelo se codifica mediante el operador de densidad

$$\rho = \frac{|\Psi\rangle\langle\Psi|}{Z}, \qquad Z = \langle\Psi|\Psi\rangle \tag{3.31}$$

Se trata de una matriz de tamaño $(d_1 \cdot d_2 \cdot \cdots \cdot d_N) \times (d_1 \cdot d_2 \cdot \cdots \cdot d_N)$ cuya diagonal son las probabilidades $P(\mathbf{v})$ de todas las configuraciones completas. Sin embargo, su dimensión crece exponencialmente con el número de características, por lo que construirla explícitamente resulta inviable.

La matriz de densidad reducida (RDM) de una o varias características es una matriz más pequeña que concentra toda la información relativa a esas características. La matriz de densidad reducida de una característica k se obtiene a partir del operador de densidad tomando la traza parcial sobre todas las demás características:

$$\rho_k = \mathrm{Tr}_{\neq k}\left(\frac{|\Psi\rangle\langle\Psi|}{Z}\right) \tag{3.32}$$

Esta operación es el análogo cuántico de marginalizar la distribución conjunta sobre el resto de variables. El resultado es una matriz pequeña, de tamaño $d_k \times d_k$, que concentra toda la información relativa a la característica k. De forma análoga, la matriz de densidad reducida de dos características, ρ_{ij}, se obtiene marginalizando sobre todas salvo esas dos.

El cálculo directo de las trazas parciales exigiría sumar sobre el espacio exponencial de configuraciones del resto de características, lo que sería inviable. Sin embargo, gracias a la estructura de *Matrix Product State* del modelo, el cálculo de la traza parcial se convierte en una secuencia de contracciones de coste polinómico [11]. El esquema de contracción para obtener la RDM de uno y de dos sitios se ilustra en la Figura 6. Sin la forma de *Matrix Product State*, la matriz de densidad reducida sería un objeto exacto pero incalculable. Esta eficiencia es lo que convierte la explicabilidad del MPS en una herramienta aplicable.

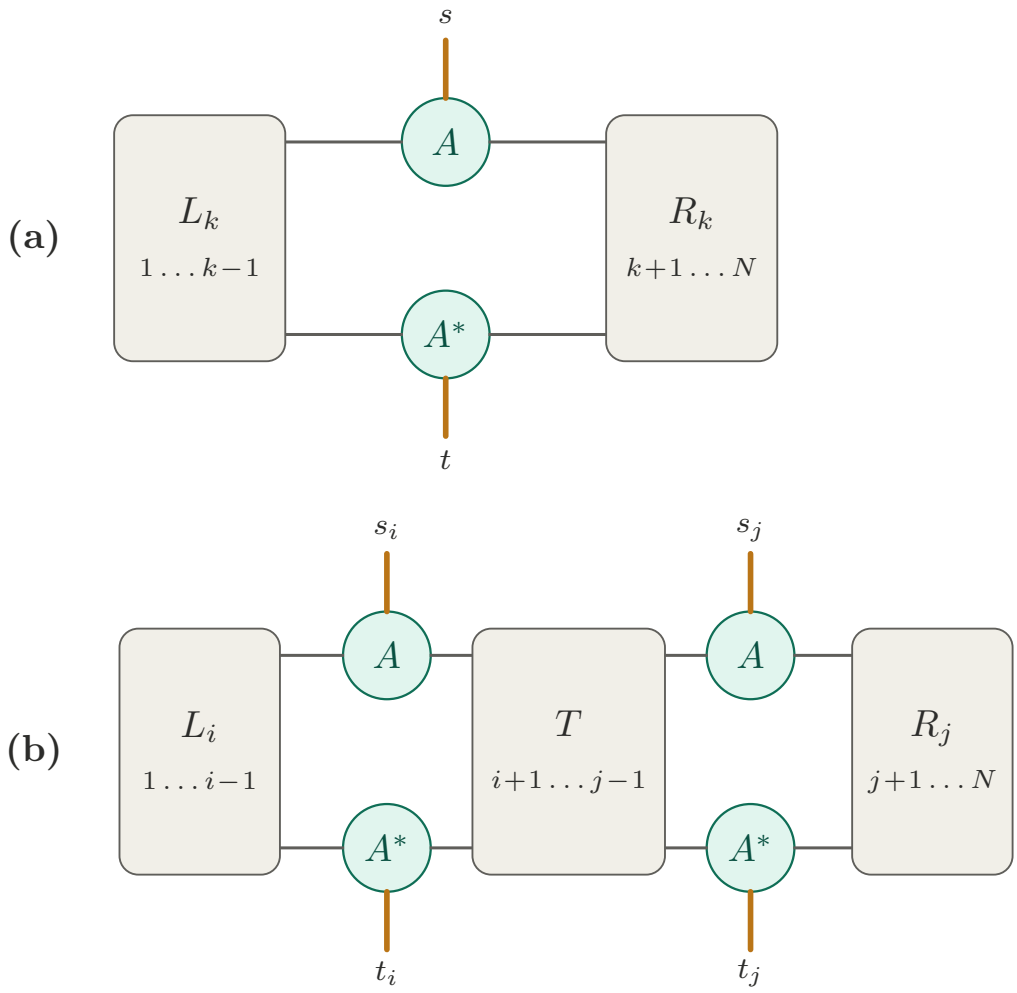


Figura 6: Esquema de contracción de las matrices de densidad reducidas a partir del MPS. Fila superior: tensores del ket $A^{(k)}$; fila inferior: sus conjugados $A^{*(k)}$ del bra. Las líneas horizontales son los enlaces virtuales y las verticales en color los índices físicos que quedan *abiertos*. **(a)** RDM de un sitio: los entornos L_k y R_k resumen los sitios trazados a izquierda y derecha, y el sitio k deja abiertos sus índices físicos s (ket) y t (bra), que forman la matriz ρ_k. **(b)** RDM de dos sitios: además de L_i y R_j aparece el entorno intermedio T, la matriz de transferencia que resulta de trazar los sitios $i+1, \ldots, j-1$ y que conecta los enlaces (ket y bra) del sitio i con los del sitio j.

La matriz de densidad reducida es un elemento fundamental en la explicabilidad del modelo, ya que de ella se derivan tanto las distribuciones de probabilidad como las medidas de entrelazamiento. Por tanto, las distintas herramientas de explicabilidad descritas en los siguientes apartados no son métodos independientes, sino que se tratan de lecturas diferentes de las matrices de densidad reducida. Por ejemplo, los elementos diagonales de la RDM de

una característica ρ_k son las probabilidades marginales que el modelo asigna a cada valor de la característica, mientras que su espectro de autovalores cuantifica el grado de entrelazamiento entre la característica y el resto del sistema.

3.2.4.2. Extracción directa de probabilidades

La primera capacidad interpretativa de las matrices de densidad reducida consiste en leer directamente las distribuciones de probabilidad que el modelo ha aprendido para cada característica. Los elementos diagonales de la matriz de densidad reducida de una característica son las probabilidades marginales que el modelo asigna a cada uno de sus valores posibles. Como el modelo se entrena exclusivamente con observaciones benignas, esta distribución representa el comportamiento normal de cada característica. A diferencia de los modelos de aprendizaje profundo, donde obtener estas distribuciones requiere de muestreos u otros métodos aproximados, el MPS permite acceder a estas probabilidades de forma directa a partir de su estructura tensorial.

Al comparar las probabilidades marginales derivadas del modelo con la frecuencia empírica observada en los datos, puede observarse que ambas distribuciones no coinciden exactamente. Esta diferencia no se debe a un defecto de ajuste del modelo, sino que es una consecuencia de que la marginal del modelo está condicionada por la estructura de correlaciones que el MPS captura, mientras que la frecuencia empírica trata cada característica de forma aislada. Por tanto, identificar las características en las que el modelo se aparta más de la frecuencia empírica puede ayudar a detectar aquellas características cuyo comportamiento depende más del resto del sistema, y que por tanto no deberían analizarse de forma aislada. Como se verá más adelante, esta discrepancia está relacionada con la entropía de Von Neumann de la característica, de forma que a mayor entrelazamiento, mayor separación entre la marginal del modelo y la frecuencia empírica.

3.2.4.3. Probabilidades condicionales

Fijando el índice físico de una característica j a un valor determinado y obteniendo la RDM de otra característica i, se observa que la RDM resultante es distinta de la RDM marginal obtenida en el apartado anterior. Esto ocurre porque los elementos de la diagonal de la nueva RDM ya no son las probabilidades marginales de la característica i, sino sus probabilidades condicionadas al valor al que hemos fijado la característica j.

Este análisis permite estudiar cómo se modifica la distribución esperada de una característica cuando otra adopta un valor concreto. Por tanto, las probabilidades condicionales proporcionan una herramienta útil para analizar dependencias específicas entre pares de características y para interpretar situaciones particulares.

3.2.4.4. Probabilidades conjuntas

A diferencia de la RDM de una característica, que proporciona las probabilidades marginales de una característica, la RDM de dos características nos proporciona las probabilidades conjuntas de ese par. En este caso, los elementos de la diagonal de la matriz indican la probabilidad de que ambas características tomen simultáneamente cada combinación posible de valores.

Comparar esta distribución conjunta con el producto de las marginales correspondientes, es decir, con la distribución que se obtendría si ambas características fueran independientes, permite identificar qué combinaciones concretas de valores ocurren con mayor o menor frecuencia de la esperada si ambas características fueran independientes. Si el cociente

$$\frac{p(v_i, v_j)}{p(v_i)\, p(v_j)}$$

es mayor que uno, indica que ambos valores aparecen en conjunto más que por separado, mientras que un valor menor que uno indica que su aparición conjunta es menos probable pese a que cada valor sea común por separado.

Este nuevo análisis, añadido a los del trabajo de referencia [15], resulta especialmente útil porque no se limita a indicar si dos características están correlacionadas, sino que permite identificar qué combinaciones específicas de valores explican esta correlación. De este modo, las probabilidades conjuntas proporcionan una interpretación aún más detallada. Mientras que medidas como la información mutua dicen cuánto dependen dos características entre sí, la probabilidad conjunta dice cómo.

3.2.4.5. Entropía de Von Neumann

En la mecánica cuántica, la entropía de Von Neumann cuantifica la incertidumbre asociada al estado reducido de un subsistema. Es la generalización al ámbito cuántico de la entropía de Shannon, la medida básica de incertidumbre de la teoría de la información [32]. Dada una distribución de probabilidad $\{p_i\}$ sobre un conjunto de resultados posibles, su entropía de Shannon se define como

$$H = -\sum_i p_i \log p_i \tag{3.33}$$

Esta cantidad vale cero cuando un único resultado es seguro, de forma que no hay ninguna incertidumbre, y alcanza su valor máximo cuando todos los resultados son igualmente probables. De esta forma, la entropía de Shannon mide qué tan dispersa o impredecible es una distribución.

La entropía de Von Neumann traslada esta idea a una matriz de densidad. Para un subsistema A se define como

$$S(\rho_A) = -\mathrm{Tr}(\rho_A \log \rho_A), \tag{3.34}$$

siendo ρ_A la matriz de densidad reducida del subsistema A. Es equivalente a la entropía de Shannon aplicada a los autovalores de ρ_A [31], que forman una distribución de probabilidad al ser no negativos y sumar uno.

En este trabajo, se analiza la entropía de Von Neumann de la RDM asociada a un único sitio del MPS. Esta cantidad mide el grado de entrelazamiento de la característica asociada a ese sitio con el resto del sistema. Por tanto, una entropía próxima a cero indica que para el modelo la característica es prácticamente independiente de las demás, mientras que una entropía elevada indica que el comportamiento de la característica está altamente correlacionado con el resto. Es esta misma magnitud la que explica la discrepancia entre la marginal del modelo y la frecuencia empírica señalada en el apartado de extracción de probabilidades.

3.2.4.6. Información mutua

La información mutua entre dos características cuantifica la dependencia estadística entre ambas. A diferencia de un coeficiente de correlación lineal, esta medida permite capturar dependencias no lineales [32]. La información mutua entre dos variables X e Y se define como

$$I(X;Y) = \sum_{y\in Y}\sum_{x\in X} p(x,y)\log\left(\frac{p(x,y)}{p(x)\,p(y)}\right) \tag{3.35}$$

donde $p(x)$ y $p(y)$ son las probabilidades marginales y $p(x,y)$ la probabilidad conjunta. Esta cantidad es nula si y solo si ambas variables son independientes.

En el marco de matrices de densidad reducida, puede utilizarse una formulación análoga basada en la entropía de Von Neumann. Para dos subsistemas A y B, la información mutua se expresa como

$$I(A;B) = S(\rho_A) + S(\rho_B) - S(\rho_{AB}), \tag{3.36}$$

donde ρ_A y ρ_B son las RDM individuales, ρ_{AB} es la RDM conjunta, y $S(\cdot)$ denota la entropía de Von Neumann.

En este trabajo, esta medida se emplea para identificar pares de características entre los que el MPS ha capturado una dependencia significativa. La información mutua resume en un escalar la intensidad de dicha dependencia, pero no indica por sí sola qué combinaciones concretas de valores son responsables de ella. Por este motivo, se utiliza como criterio para seleccionar los pares de características más relevantes, para posteriormente analizarlas con mayor detalle mediante sus probabilidades conjuntas o condicionadas.

3.2.4.7. Entropía de enlace

A las medidas anteriores, este trabajo añade la entropía de enlace. A diferencia de todas las anteriores, esta no es una propiedad de unas características concretas, sino del orden en que todas las características se disponen a lo largo de la cadena. Mientras que la entropía de sitio separa una única característica del resto, la entropía de enlace divide la cadena en dos bloques y mide cuánta correlación cruza el corte. Un valor alto indica que las características

a ambos lados de ese corte están fuertemente acopladas, mientras que un valor cercano a cero indica que la cadena se comporta como si en ese punto estuviera separada en dos bloques prácticamente independientes.

Para calcularla se recurre a la descomposición en valores singulares asociada al enlace. Al separar la cadena por el corte k en dos bloques A y B, el estado admite una descomposición de Schmidt [31]

$$|\Psi\rangle = \sum_i \sigma_i \, |u_i\rangle_A |v_i\rangle_B. \tag{3.37}$$

cuyos coeficientes son los valores singulares $\sigma_1, \sigma_2, \ldots$ de ese enlace. A partir de ellos se construye una distribución de probabilidad a partir de sus cuadrados normalizados

$$p_i = \frac{\sigma_i^2}{\sum_j \sigma_j^2} \tag{3.38}$$

Estos valores son no negativos y suman uno, y coinciden con los autovalores de la matriz de densidad reducida de cualquiera de los dos bloques [11], [21]. La entropía de enlace es por tanto la entropía de Shannon de esa distribución,

$$S_k = -\sum_i p_i \log p_i, \tag{3.39}$$

que es exactamente la entropía de Von Neumann de la bipartición definida por el corte. Mientras que la entropía de sitio definida en el apartado anterior aisla una única característica del resto del sistema, la entropía de enlace mide la cantidad de información compartida entre los dos bloques separados por el enlace.

Esta medida conecta directamente con la dimensión de enlace del modelo, pues, como establece el trabajo de Han et al. [13], la entropía de enlace acota inferiormente la dimensión de enlace necesaria en ese corte D_k,

$$S_k \leq \log D_k, \tag{3.40}$$

Como consecuencia, la cantidad de entrelazamiento que puede atravesar cada corte está limitada por su dimensión de enlace. Comparar la entropía de cada enlace con su cota teórica indica qué enlaces operan cerca de su capacidad y se beneficiarían de una mayor dimensión, y cuáles están infrautilizados.

3.2.4.8. Identificación de anomalías

Todas las medidas anteriores caracterizan al modelo de forma global. Sin embargo, ante una alerta, un analista no solo necesita saber cómo se comporta el modelo en promedio, sino por qué cada evento individual ha sido marcado como una anomalía. El MPS también proporciona esta explicación de forma nativa, sin necesidad de herramientas externas como los análisis de sensibilidad o la maximización de activaciones que necesitarían un autoencoder o una red generativa para aproximar el mismo razonamiento.

La puntuación de anomalía de un evento $\mathbf{v}$ es su *negative log-likelihood* (NLL) bajo el modelo,

$$\mathrm{NLL}(\mathbf{v}) = -\log P(\mathbf{v}), \tag{3.41}$$

de modo que cuanto menos probable es un evento para el modelo entrenado con datos benignos, mayor es su puntuación.

Es en este apartado donde este trabajo introduce su corrección más importante respecto al de referencia (Aizpurua et al. [15]). En él, calculan la verosimilitud de una observación como el producto de las probabilidades marginales del valor que toma cada característica en esa observación:

$$P(\mathbf{v}) = \prod_{i=1}^{N} P_i(v_i), \tag{3.42}$$

y por tanto calculan la NLL como

$$\mathrm{NLL}(\mathbf{v}) = -\sum_{i} \log P_i(v_i). \tag{3.43}$$

Sin embargo, esta factorización solo es exacta cuando las características son independientes, es decir, cuando el modelo no captura ninguna correlación. Esto es especialmente incorrecto en un MPS, ya que una de las principales ventajas del modelo es precisamente su capacidad para capturar las correlaciones entre características. Por tanto, utilizar únicamente el producto de marginales como puntuación de anomalía descarta justamente la información que hace que un MPS se distinga de un modelo ingenuo basado en independencia.

A pesar de esta limitación, el desglose por característica sigue siendo útil en muchos casos. Para una alerta concreta, se define la contribución de cada característica como su verosimilitud negativa marginal,

$$a_i(\mathbf{v}) = -\log P_i(v_i), \tag{3.44}$$

que mide cuán improbable es de forma aislada el valor que toma. El problema es que, como el modelo captura correlaciones, la suma de estas contribuciones no coincide con la puntuación de anomalía real. A esta discrepancia la llamamos residuo de correlación

$$r(\mathbf{v}) = \mathrm{NLL}(\mathbf{v}) - \sum_{i=1}^{N} a_i(\mathbf{v}). \tag{3.45}$$

Bajo la suposición de independencia del trabajo de referencia, este residuo sería siempre cero, así que cualquier valor distinto de cero es información que solo aporta un modelo que captura correlaciones.

Para corregir esta limitación del trabajo de referencia y, al mismo tiempo, aprovechar la utilidad de su métrica, este trabajo introduce la cuota de correlación:

$$c(\mathbf{v}) = \frac{|r(\mathbf{v})|}{\mathrm{NLL}(\mathbf{v})}. \tag{3.46}$$

Esta magnitud indica al analista hasta qué punto el desglose marginal puede interpretarse como una explicación fiable de la alerta. Cuando la cuota es baja, la suma de las contribuciones marginales reproduce casi toda la puntuación de anomalía, por lo que el desglose es fiel y el analista puede leer en él el motivo de la alerta. En cambio, cuando la cuota es alta, gran parte de la anomalía no se debe a valores individualmente improbables, sino a una combinación inusual de valores. En este caso, el desglose marginal resulta insuficiente y conviene complementar el análisis con medidas basadas en dependencias como probabilidades conjuntas, probabilidades condicionadas o información mutua.

3.2.4.9. Importancia de las características

En muchos casos, el objetivo de un detector de anomalías no es detectar cualquier observación que se desvíe del comportamiento normal, sino encontrar aquellas que suponen realmente una amenaza. Por ejemplo, en un sistema de detección de fraude el objetivo es identificar operaciones fraudulentas, y en un sistema de ciberseguridad el objetivo es detectar intrusiones. Sin embargo, el modelo no aprende esa noción de amenaza, sino una representación del comportamiento normal.

Utilizar un detector de anomalías como detector de amenazas se apoya en la hipótesis de que las amenazas tienden a apartarse del comportamiento normal. Sin embargo, lo contrario no es siempre cierto. Un valor de una variable puede ser tan improbable en las amenazas como lo es en las observaciones benignas, por lo que no toda contribución a la puntuación de anomalía resulta necesariamente útil para distinguir amenazas reales.

Por tanto, la importancia de una característica no coincide necesariamente con su contribución a la detección de anomalías. Para cuantificar esta importancia, se compara el comportamiento del modelo sobre las observaciones benignas y las amenazas por separado. Para cada característica i se lee la probabilidad $P_i(v_i)$ que el modelo asigna al valor efectivamente observado en cada evento y se promedia por separado sobre los eventos benignos y los eventos de ataque del conjunto de evaluación, obteniendo $\overline{P}_i^{\text{benigno}}$ y $\overline{P}_i^{\text{ataque}}$. A partir de estas dos cantidades se define la importancia de la característica como la diferencia entre ambas medias:

$$\Delta_i = \overline{P}_i^{\text{benigno}} - \overline{P}_i^{\text{ataque}}. \tag{3.47}$$

Un valor de Δ_i próximo a cero indica que los valores que son típicos en las observaciones benignas también lo son en las amenazas. Por tanto, esa característica no ayuda por sí sola a distinguir amenazas. Por el contrario, un Δ_i alto indica que las amenazas presentan por lo general valores que son poco probables según el modelo. Por tanto, si esa característica tiene una contribución elevada a la detección de una anomalía, es más probable que esa anomalía corresponda a una amenaza real y no a un evento benigno atípico. Finalmente, un valor negativo de Δ_i indica que los valores observados en las amenazas resultan incluso más probables bajo el modelo que los observados en las muestras benignas, por lo que la característica tampoco ayuda por sí sola a detectar amenazas.

Es importante tener en cuenta dos limitaciones de esta medida. En primer lugar, esta medida se apoya en las probabilidades marginales de cada característica, por lo que no tiene en cuenta las correlaciones entre ellas. Una característica puede tener un Δ_i pequeño y, sin embargo, contribuir de forma decisiva a la detección de amenazas a través de sus combinaciones específicas con otras características. En segundo lugar, esta medida se obtiene de manera supervisada, puesto que utiliza las etiquetas del conjunto de evaluación para separar los eventos benignos y de ataque. Por tanto, a diferencia de todas las medidas anteriores, no es una propiedad intrínseca del modelo en sí, sino un análisis realizado a posteriori para relacionar la explicación del MPS con la noción externa de amenaza.

3.3. Implementación

3.3.1. Caso de uso

Dentro de la detección de anomalías existen muchos casos de uso distintos. El modelo generativo basado en *Matrix Product States* descrito en la Sección 3.2 es aplicable a cualquier problema de detección de anomalías que admita una codificación discreta adecuada.

Para validar el sistema, el caso de uso elegido en este trabajo es la detección de intrusiones en redes de comunicaciones. Un sistema de detección de intrusiones de red (*Network Intrusion Detection System*, NIDS) [16] monitoriza el tráfico que circula por una red con el fin de identificar aquella actividad que compromete la confidencialidad, la integridad o la disponibilidad de los recursos. Tradicionalmente, estos sistemas se dividen en dos familias según el modo en que deciden qué tráfico es sospechoso:

- **Detección basada en firmas** (*signature-based* o *misuse detection*): mantiene una base de datos de patrones de ataque conocidos y marca como intrusión todo el tráfico que coincide con alguno de ellos. Ofrece una precisión muy alta y pocas falsas alarmas sobre los ataques catalogados, pero es incapaz de detectar ataques que no figuren previamente en la base de firmas y exige una actualización continua de la misma.
- **Detección basada en anomalías** (*anomaly-based*): construye un modelo del comportamiento normal de la red y marca como sospechosa cualquier desviación apreciable respecto a él. A cambio de una mayor tasa de falsas alarmas, su principal ventaja es la capacidad de detectar ataques no observados con anterioridad.

El modelo generativo basado en *Matrix Product States* encaja en esta segunda familia de detectores de intrusiones de red. Además, la detección de intrusiones es un dominio en el que la interpretabilidad del modelo es especialmente importante. El volumen de alertas que gestiona un centro de operaciones de seguridad puede ser muy elevado, y una puntuación opaca que señale una conexión como sospechosa sin indicar el motivo aporta poca información al analista encargado de priorizar, validar o descartar cada alerta. Un detector que, además de la puntuación de anomalía, indique qué características han hecho que una conexión resulte anómala facilita la labor de los analistas y hace que el sistema sea más útil en la práctica.

Para validar el sistema sobre este caso de uso se emplea el conjunto de datos NSL-KDD, un referente consolidado en la evaluación de sistemas de detección de intrusiones que se describe en detalle en la Sección 3.3.2.

3.3.2. Conjunto de datos utilizado: NSL-KDD

NSL-KDD es una versión depurada del conjunto KDD Cup 1999 [36], que durante años fue el estándar para evaluar sistemas de detección de intrusiones. Sin embargo, el conjunto padecía diversos problemas, que fueron estudiados por Tavallaee et al. [34], dando lugar a una versión corregida llamada NSL-KDD.

El principal defecto de KDD'99 es el elevado número de observaciones repetidas. Esto provoca que durante el entrenamiento el modelo tienda a ajustarse en mayor medida a las observaciones más frecuentes. Además, durante la evaluación provoca que los resultados sean artificialmente altos incluso cuando el modelo no generaliza bien. NSL-KDD corrige este defecto eliminando los registros duplicados tanto en el conjunto de entrenamiento como en el de test. Como consecuencia, el conjunto de datos es además más pequeño, lo que reduce el coste de tanto el entrenamiento como la validación.

Además, NSL-KDD añade un campo de dificultad a cada observación, que mide cuántos de entre 21 clasificadores de referencia consiguen clasificarla correctamente. Debido a que la mayoría de las observaciones resultaron muy fáciles de evaluar en KDD'99, para NSL decidieron muestrear las observaciones de forma inversamente proporcional a su dificultad, evitando que el conjunto quede dominado por casos triviales. Por ejemplo, las observaciones con nivel de dificultad entre 0 y 5, que representan los casos más difíciles, tenían una presencia muy reducida en el conjunto original, por lo que se conserva una mayor proporción de ellos en NSL-KDD. Además de evitar que el conjunto esté dominado por casos triviales, esto también hace el nuevo conjunto más reducido y manejable computacionalmente.

Sin embargo, este conjunto de datos no está libre de problemas. El KDD'99 original, y por tanto NSL-KDD, deriva de capturas de tráfico de finales de los años noventa, por lo que no refleja los patrones de tráfico ni las técnicas de ataque actuales. Además, algunas de las características fueron generadas de manera sintética para asegurar la privacidad, y no se comprobó de manera analítica ni experimental que fueran realistas.

Por tanto, NSL-KDD no es un conjunto de datos pensado para entrenar un sistema que vaya a entrar en producción, sino que está construido con la idea de validar la viabilidad de un sistema propuesto. Siendo este el objetivo de este trabajo, NSL-KDD se ha escogido por distintas razones. En primer lugar, es un conjunto de referencia consolidado que facilita la comparación con literatura previa. En segundo lugar, su tamaño moderado lo hace tratable con recursos computacionales acotados. Finalmente, su estructura tabular, con características categóricas, binarias y numéricas, permite evaluar la capacidad del codificador propuesto para manejar observaciones heterogéneas.

3.3.2.1. Composición del conjunto de datos

Cada registro del conjunto de datos representa una conexión de red y está descrito por 41 características, además de una etiqueta que indica si la conexión corresponde a tráfico normal o a un ataque. La característica `num_outbound_cmds` toma siempre el valor 0 en NSL-KDD, por lo que esta se descarta, quedando 40 características modeladas.

El conjunto de entrenamiento no contiene todos los tipos de ataque presentes en el conjunto de test, puesto que el objetivo es que el sistema sea capaz de detectar ataques zero-day. Por tanto, mientras que en el conjunto de entrenamiento solo hay 22 tipos de ataque distintos, en el conjunto de test hay 37. En total, el conjunto reúne 39 tipos de ataque distintos. Los ataques pertenecen a una de cuatro familias distintas:

- Ataque de denegación de servicio (DoS): el atacante intenta hacer que algún recurso de cómputo, memoria o red quede saturado, impidiendo o degradando la atención de solicitudes legítimas.
- Ataque de sondeo (Probe): el atacante intenta recopilar información sobre la red o sus equipos.
- Ataque de remoto a local (R2L): un atacante que puede enviar paquetes a una máquina a través de una red, pero que no tiene una cuenta en esa máquina, explota alguna vulnerabilidad para obtener acceso local como usuario de dicha máquina.
- Ataque de usuario a root (U2R): un atacante que ya dispone de una cuenta de usuario normal explota alguna vulnerabilidad para obtener privilegios de *root* en el sistema.

En este trabajo, la asignación de cada tipo de ataque a su familia sigue la categorización oficial de NSL-KDD [34], que se muestra en la Tabla 1. Bajo dicha categorización, el conjunto de test queda repartido en 9711 conexiones normales y 12833 de ataque, distribuidas en 7458 ataques DoS, 2421 de sondeo, 2754 R2L y 200 U2R, lo que evidencia el fuerte desbalanceo del conjunto hacia el tráfico normal y los ataques DoS.

Tabla 1: Asignación de cada tipo de ataque a su familia según la categorización oficial de NSL-KDD [34] y número de conexiones de cada familia en el conjunto de test. El conjunto reúne 39 tipos de ataque distintos repartidos en cuatro familias.

Familia	**N.º tipos**	**Conexiones**	**Tipos de ataque**
DoS	10	7458	`back`, `land`, `neptune`, `pod`, `smurf`, `teardrop`, `apache2`, `udpstorm`, `processtable`, `mailbomb`
Probe	6	2421	`satan`, `ipsweep`, `nmap`, `portsweep`, `mscan`, `saint`
R2L	15	2754	`guess_passwd`, `ftp_write`, `imap`, `phf`, `multihop`, `warezmaster`, `warezclient`, `spy`, `xlock`, `xsnoop`, `snmpguess`, `snmpgetattack`, `sendmail`, `named`, `worm`
U2R	8	200	`buffer_overflow`, `loadmodule`, `rootkit`, `perl`, `sqlattack`, `xterm`, `ps`, `httptunnel`
Total	**39**	**12833**	

Las características de los registros se pueden dividir en tres grupos:

- Características básicas: son todos los atributos que pueden ser extraídos de una conexión TCP/IP.
- Características de tráfico: son todos los atributos que se calculan observando las conexiones ocurridas dentro de un periodo determinado. Se dividen a su vez en dos grupos:
  - Características de mismo host: examinan las conexiones en los últimos 2 segundos que tienen el mismo host de destino que la conexión actual.
  - Características de mismo servicio: examinan las conexiones en los últimos 2 segundos que tienen el mismo servicio que la conexión actual.

  Estas características se recalculan también en ventanas de 100 conexiones, puesto que algunos ataques de sondeo utilizan intervalos de tiempo de mucho más de 2 segundos.
- Características de contenido: Son características capaces de buscar comportamientos sospechosos en la porción de datos de la conexión.

  Estas características están orientadas principalmente a detectar ataques de tipo R2L y U2R, que a diferencia de los ataques DoS y de sondeo no presentan patrones frecuentes de intrusión. Esto se debe a que los ataques DoS y de sondeo implican muchas conexiones hacia uno o varios hosts en un período de tiempo muy corto. Sin embargo,

los ataques R2L y U2R están incrustados en las porciones de datos de los paquetes y normalmente implican solo una conexión.

Tabla 2: Clasificación oficial de las características de NSL-KDD [34]. Las características de tráfico temporales se calculan sobre una ventana de 2 s y las basadas en host repiten ese cálculo sobre una ventana de 100 conexiones.

Tipo	**Características**
Básicas	`duration`, `protocol_type`, `service`, `flag`, `src_bytes`, `dst_bytes`, `land`, `wrong_fragment`, `urgent`
De contenido	`hot`, `num_failed_logins`, `logged_in`, `num_compromised`, `root_shell`, `su_attempted`, `num_root`, `num_file_creations`, `num_shells`, `num_access_files`, `num_outbound_cmds`*, `is_host_login`, `is_guest_login`
De tráfico, mismo host (2 s)	`count`, `serror_rate`, `rerror_rate`, `same_srv_rate`, `diff_srv_rate`
De tráfico, mismo servicio (2 s)	`srv_count`, `srv_serror_rate`, `srv_rerror_rate`, `srv_diff_host_rate`
De tráfico, basadas en host (100 con.)	`dst_host_count`, `dst_host_srv_count`, `dst_host_same_srv_rate`, `dst_host_diff_srv_rate`, `dst_host_same_src_port_rate`, `dst_host_srv_diff_host_rate`, `dst_host_serror_rate`, `dst_host_srv_serror_rate`, `dst_host_rerror_rate`, `dst_host_srv_rerror_rate`

* `num_outbound_cmds` es siempre 0 en NSL-KDD, por lo que se descarta, quedando 40 características modeladas.

3.3.2.2. Partición empleada en este trabajo

El conjunto de datos original NSL-KDD se encuentra ya dividido en un conjunto de entrenamiento y uno de test. El conjunto de entrenamiento *KDDTrain+* contiene 125 973 observaciones, de las que 67 343 son observaciones normales y 58 630, aproximadamente el 47 %, son ataques. En cambio, el conjunto de test es más pequeño, con 22 544 observaciones, y contiene una mayor proporción de ataques, aproximadamente un 57 % (12 833). En este trabajo utilizaremos tres subconjuntos derivados de estos dos.

Del conjunto de entrenamiento, mantenemos únicamente los datos etiquetados como normales, dado que el modelo se ajusta de forma semisupervisada. Por tanto, nos quedamos

únicamente con 67 343 observaciones. De ellas, una pequeña proporción se reserva como conjunto de validación, también compuesto solo por tráfico normal. Este conjunto sirve para vigilar el sobreajuste durante el entrenamiento y para posteriormente establecer el umbral de decisión. En esta implementación concreta se utiliza para validación un 15 %, quedando 57 242 observaciones para entrenamiento y 10 101 para validación.

El conjunto de test mantiene tanto las conexiones normales como las de ataque, y se utilizará para el cálculo de todas las métricas de rendimiento.

3.3.3. Arquitectura del modelo MPS

El modelo utilizado en este trabajo es una *Born Machine* parametrizada mediante *Matrix Product States*, descrita en la Sección 3.2.1. Esta sección se limita a describir la arquitectura concreta del modelo utilizado en los experimentos.

3.3.3.1. Número de sitios

El modelo consta de 40 sitios, uno por cada característica del conjunto de datos, tras descartar `num_outbound_cmds` como se describe en la Sección 3.3.2. La dimensión física de cada sitio queda determinada por el esquema de codificación generado por el codificador a partir del conjunto de entrenamiento, según las reglas descritas en Sección 3.2.2.

3.3.3.2. Dimensiones de enlace

La dimensión de enlace D_k de cada uno de los 39 enlaces internos de la cadena es el parámetro que regula la capacidad expresiva del modelo en ese punto. En este trabajo, las dimensiones de enlace no se fijan a priori a un valor uniforme, sino que se determina de forma adaptativa durante el entrenamiento tal como se explica en la Sección 3.2.3.

No obstante, se impone una cota superior común a todas las dimensiones de enlace, que limita el tamaño máximo que cualquier enlace puede alcanzar a lo largo del entrenamiento. Esta cota cumple dos funciones. Por un lado, acota el coste computacional y de memoria del modelo. Por otro lado, actúa como mecanismo de regularización, ya que impide que el modelo capture correlaciones excesivamente complejas que podrían reflejar ruido en los datos en lugar de estructura relevante del tráfico normal.

3.3.3.3. Orden de sitios

El orden de las características en el modelo es una elección que afecta a su capacidad expresiva. El modelo captura con mayor eficacia las correlaciones entre características próximas en la cadena, mientras que las correlaciones a larga distancia requieren transmitir información a través de muchos enlaces intermedios y, por tanto, dimensiones de enlace más elevadas. Este es un fenómeno bien conocido en el contexto original del DMRG, donde se denomina

problema de ordenamiento y ha sido estudiado en profundidad [37]. El criterio más extendido para abordarlo consiste en agrupar en la cadena los sitios fuertemente correlacionados, empleando como medida de correlación la información mutua entre pares de sitios.

Sin embargo, en este trabajo se adopta el orden original del conjunto de datos, es decir, el orden en el que las características aparecen listadas en la especificación de NSL-KDD. Esta elección tiene dos ventajas: es más reproducible, y agrupa de forma natural bloques de características afines. Por ejemplo, las características de tasa de error de servicio aparecen contiguas, lo que reduce la distancia entre características probablemente correlacionadas.

3.3.4. Pruebas y métricas de evaluación para la validación del sistema

Esta sección establece las pruebas y métricas de evaluación con las que se validará el sistema sobre el conjunto de test.

3.3.4.1. Métricas dependientes del umbral

Antes de explicar las métricas conviene definir distintos conceptos:

- Verdaderos positivos (TP): observaciones de ataque correctamente identificadas como anómalas.
- Falsos positivos (FP): observaciones normales que han sido incorrectamente clasificadas como anómalas.
- Verdaderos negativos (TN): observaciones normales que han sido correctamente identificadas como normales.
- Falsos negativos (FN): observaciones de ataque que han sido incorrectamente clasificadas como normales.

A partir de estos valores obtenemos distintas métricas:

- Precisión: mide la proporción de las observaciones clasificadas como anómalas que efectivamente corresponden a ataques.

$$\text{Precisión} = \frac{TP}{TP + FP}$$

- Recall: mide la proporción de ataques que el sistema ha conseguido detectar.

$$\text{Recall} = \frac{TP}{TP + FN}$$

- F1-score: es la media armónica entre la precisión y el recall. Resume ambas en una única cifra y es útil para comparar modelos en los que la precisión es más alta en uno pero el recall más alto en el otro.

$$F1 = 2 \cdot \frac{\text{Precisión} \cdot \text{Recall}}{\text{Precisión} + \text{Recall}}$$

- Tasa de falsos positivos (FPR): mide la proporción de conexiones normales incorrectamente clasificadas como ataques.

$$FPR = \frac{FP}{FP + TN}$$

  Esta métrica es importante desde el punto de vista operativo, puesto que cada falso positivo se traduce en una alerta que un analista humano debe revisar. En un entorno con un alto volumen de tráfico, un FPR elevado puede producir un número de alertas difícil de gestionar.

Todas estas métricas dependen del umbral que se aplique sobre la puntuación de anomalía para decidir qué observaciones se consideran anómalas. En este trabajo se ha optado por no comprometer el sistema con un único umbral, sino por caracterizar su comportamiento a lo largo de distintos umbrales. Esta decisión responde a dos motivos. En primer lugar, permite realizar un análisis más completo y transparente, puesto que fijar un único umbral, posiblemente favorable, ocultaría el resto del compromiso entre detección y falsas alarmas. En segundo lugar, el umbral adecuado para un detector de intrusiones depende del contexto de despliegue, por lo que es una decisión que corresponde al operador y no al diseñador del modelo. El umbral elegido dependerá de factores como el volumen de tráfico, el coste de cada falsa alarma o la tolerancia a ataques no detectados.

3.3.4.2. Métricas independientes del umbral

Estas métricas evalúan la calidad del modelo antes de fijar un umbral de decisión, valorando únicamente su capacidad para asignar puntuaciones de anomalía más altas a los ataques que a las conexiones normales. Esto las hace muy interesantes pues permiten evaluar el modelo generativo de manera independiente al umbral que posteriormente elija cada operador.

La métrica más extendida es el área bajo la curva ROC (AUC-ROC). La curva ROC representa la tasa de verdaderos positivos (TPR) frente a la tasa de falsos positivos (FPR) a medida que el umbral barre todo el rango posible. Su área se interpreta como la probabilidad de que el modelo asigne una puntuación de anomalía mayor a una conexión de ataque elegida al azar que a una conexión normal elegida al azar. Sus valores oscilan entre el 0,5, que indica que el modelo es equivalente a elegir al azar, y el 1, que corresponde a una separación perfecta entre observaciones normales y ataques.

$$\text{AUC-ROC} = \int_0^1 \text{TPR}(\text{FPR})\, d\text{FPR} \tag{3.48}$$

La limitación de esta métrica es que si hay una clase que esté poco representada, el valor de AUC-ROC puede mantenerse alto incluso cuando el modelo produce una precisión baja en términos absolutos [38]. El motivo está en el denominador del FPR, $FPR = FP/(FP+TN)$, puesto que si los verdaderos negativos (TN) son muy numerosos, hacen falta muchos falsos positivos para que el FPR aumente de forma apreciable, de modo que la curva ROC apenas se degrada aunque el modelo genere un volumen de falsas alarmas considerable. En esa situación, el AUC-ROC puede mantenerse alto incluso cuando la precisión (*precision*), $TP/(TP+FP)$, es baja, es decir, aunque la mayoría de las observaciones marcadas como anómalas no sean realmente ataques.

En el caso de NSL-KDD, los datos normales y los ataques están equilibrados en el conjunto de test, con una proporción ligeramente mayor de ataques, por lo que ese defecto no afectará al análisis global. Sin embargo, al desagregar los resultados por familia de ataque (Sección 3.3.4.3) este efecto sí puede dificultar el análisis de rendimiento del modelo frente a familias muy poco representadas como R2L y U2R.

Por este motivo, medimos también el área bajo la curva de precisión-recall (AUC-PR) [38]. La curva de precisión-recall sustituye el FPR por la precisión, que reacciona directamente a cada falso positivo y no depende del número de verdaderos negativos. Esto la convierte en una métrica más informativa cuando la clase de interés está poco representada.

$$\text{AUC-PR} = \int_0^1 \text{Precision}(\text{Recall})\, d\text{Recall} \tag{3.49}$$

3.3.4.3. Análisis desagregados

Con el objetivo de encontrar las diferencias de rendimiento frente a los distintos tipos de ataque, las métricas anteriores también se analizarán de forma desagregada. En NSL-KDD [34] se agrupan los ataques en cuatro familias: denegación de servicio (DoS), ataque de sondeo (Probe), ataque de remoto a local (R2L) y ataque de usuario a root (U2R). Las características de los ataques pertenecientes a cada familia son muy distintas. Por ejemplo, mientras que los ataques de denegación de servicio generan patrones de volumen muy anómalos, los ataques R2L suelen camuflarse entre sesiones aparentemente legítimas. Analizar las métricas de forma desagregada por familias nos permite estudiar el rendimiento del sistema frente a cada una de ellas.

3.3.5. Estructura del código

El código fuente del sistema se organiza en tres capas con una separación estricta de responsabilidades. En primer lugar, un núcleo de cómputo independiente del dominio que contiene toda la lógica del modelo generativo basado en *Matrix Product States*. En segundo

lugar, una capa de adaptación específica del conjunto de datos. Finalmente, un conjunto de scripts de orquestación que encadenan las etapas del flujo de trabajo.

El núcleo está formado por tres módulos. El primero, `mps.py`, contiene la clase que representa el *Matrix Product State* y concentra el grueso de la lógica del modelo. Este módulo gestiona, entre otros aspectos, la representación de la cadena de tensores, la contracción para evaluar la amplitud de una configuración, el cálculo de la puntuación de anomalía y las operaciones de canonización y normalización. El segundo, `dmrg_trainer.py`, implementa el algoritmo de entrenamiento de dos sitios descrito en la Sección 3.2.3. Este módulo no importa la clase del modelo, sino que opera sobre cualquier objeto que se le proporcione, de modo que el entrenador queda completamente desacoplado de la representación concreta del modelo. El tercero, `mps_explainability.py`, contiene la clase de explicabilidad, que depende del módulo del modelo y se encarga de las matrices de densidad reducida y de extraer toda la información derivada de ellas. Estos tres módulos no contienen ninguna referencia al conjunto de datos NSL-KDD ni a la detección de intrusiones de red, por lo que son reutilizables sobre cualquier problema de detección de anomalías que admita una codificación discreta.

La capa de adaptación consiste en un único módulo `encoder_nsl_kdd.py`, que encapsula todo lo específico del conjunto de datos: la discretización de las características, la construcción del esquema de codificación y la generación de los tensores de enteros que consume el modelo. Al estar aislada en un solo módulo sin dependencias del núcleo, esta capa actúa como la conexión entre el problema concreto y el resto de la lógica. Por tanto, adaptar el sistema a otro conjunto de datos se reduciría a sustituir este codificador.

Por último, los scripts de orquestación ejecutan las distintas etapas del flujo de trabajo. El script `train_mps_nsl_kdd.py` se encarga del entrenamiento del modelo, `evaluate_mps_nsl_kdd.py` de la evaluación, y `explain_mps_nsl_kdd.py` de la generación de los diagnósticos de explicabilidad. Cada uno combina las piezas del núcleo y de la capa de adaptación necesarias para producir los artefactos correspondientes.

3.3.6. Pipeline

El pipeline se divide en cuatro etapas: la codificación, el entrenamiento, la evaluación y la explicabilidad. En primer lugar se ejecuta la codificación, seguida del entrenamiento. Una vez entrenado el modelo, la evaluación y la explicabilidad se pueden ejecutar en cualquier orden. Las etapas no se comunican en memoria, sino a través de artefactos persistidos en disco. Cada script lee los ficheros producidos por los anteriores y escribe los suyos, lo que permite ejecutar cada etapa de forma independiente, inspeccionar los resultados intermedios y repetir solo la parte del flujo que sea necesaria sin rehacer todo el proceso.

La primera etapa es la codificación. A partir de los ficheros de entrenamiento y test originales del conjunto NSL-KDD, el codificador discretiza las características según el sistema descrito en la Sección 3.2.2 y produce los tensores de enteros de cada partición junto con sus metadatos y el esquema de codificación. Este esquema, guardado explícitamente, garantiza que todas las etapas posteriores interpreten cada característica con exactamente la misma discretización.

La segunda etapa es el entrenamiento. El script de entrenamiento carga los tensores de la partición de entrenamiento y conserva únicamente las instancias normales. Una fracción de ellas se reserva como conjunto de validación para el seguimiento de la convergencia y la parada temprana, mientras que el resto se utiliza para ajustar el modelo a la distribución del tráfico benigno. El resultado de esta etapa es el modelo entrenado, junto con el historial de entrenamiento y un registro detallado de la ejecución.

En la etapa de evaluación, se utiliza el modelo entrenado para calcular la puntuación de anomalía de todo el conjunto de test, que incluye tanto conexiones normales como de ataque. Para ajustar los umbrales se reconstruye la misma partición de validación que se usó en el entrenamiento empleando la misma semilla, para evitar que el umbral se ajuste a los datos del test o del entrenamiento. A partir de estas puntuaciones se calcula el conjunto de métricas de detección descritas en la Sección 3.3.4 y se produce un informe de métricas legible por máquina además de distintas figuras.

En la etapa de explicabilidad se calculan todos los diagnósticos descritos en la Sección 3.2.4 a partir del modelo entrenado y del conjunto de test. Como resultado de esta etapa se generan informes legibles por máquina y distintas figuras interpretables para un analista, que permiten estudiar tanto el comportamiento global del modelo como las razones que explican determinadas alertas.

3.3.7. Reproducibilidad

El flujo de trabajo se ha diseñado de forma que, a partir de los ficheros originales de NSL-KDD, una ejecución desde cero reproduzca de manera determinista tanto el modelo entrenado como las métricas de evaluación y los diagnósticos de explicabilidad que se presentan en este trabajo. Esto se consigue gracias al control de todas las fuentes de aleatoriedad y al desacoplamiento de las etapas a través de ficheros persistentes en disco.

Las operaciones aleatorias del flujo de trabajo se controlan mediante una única semilla, fijada en 123. Esta semilla inicializa el generador de PyTorch utilizado en la inicialización aleatoria de los tensores del *Matrix Product State*. Además, se reutiliza la misma semilla para inicializar el generador que se utiliza para obtener la partición entre entrenamiento y validación. Durante la evaluación, esta misma partición de validación se reconstruye con la semilla y la fracción definidas en el entrenamiento, de modo que el umbral de decisión se calibra únicamente sobre tráfico normal no usado para ajustar el modelo. Así se evita utilizar datos de test o de entrenamiento para fijar el umbral. Además, todos los cálculos se realizan en doble precisión, `float64`, para reducir la acumulación de errores numéricos.

Además, cada etapa del pipeline escribe en disco todos los artefactos que necesitan las etapas posteriores. Esto permite reanudar el flujo desde cualquier punto, inspeccionar los resultados intermedios y regenerar las métricas o figuras sin repetir necesariamente todo el proceso. Es especialmente importante el esquema de codificación (`encoding_schema.json`), que registra de forma explícita la discretización aprendida para cada característica, incluyendo

sus dimensiones físicas, vocabularios y cortes de cuantiles. Al reutilizar el mismo esquema en entrenamiento, evaluación y explicabilidad, se garantiza que una misma observación se traduzca siempre a los mismos índices físicos del MPS. Además, se almacenan los tensores codificados de cada partición, sus metadatos, el modelo entrenado (`mps_trained.pt`), el historial de entrenamiento y los ficheros de métricas y diagnósticos en formato legible por máquina.

3.3.7.1. Configuración del experimento

Para que el experimento sea completamente reproducible se documentan en la Tabla 3 todos los hiperparámetros empleados. Estos abarcan la capacidad del modelo y el control adaptativo de las dimensiones de enlace descrito en la Sección 3.3.3.2, los parámetros del algoritmo de entrenamiento DMRG de la Sección 3.2.3 y la política de calibración de umbrales de la evaluación. Las versiones concretas del software y el equipo sobre el que se ejecutaron los experimentos se detallan en la Sección 3.3.8.

Tabla 3: Hiperparámetros empleados en el entrenamiento y la evaluación del modelo. El conjunto completo, junto con el código que los consume, está disponible en el repositorio público del proyecto [39].

Bloque	**Parámetro**	**Valor**
Capacidad del modelo	Dimensión de enlace inicial	2
	Cota inicial de la dimensión de enlace	8
	Cota máxima de la dimensión de enlace	128
	Factor de crecimiento del enlace	1,5
	Umbral de peso descartado	10^{-3}
	Iteraciones de confirmación de crecimiento	5
	Corte de la descomposición SVD	10^{-8}
	Precisión numérica	`float64`
Entrenamiento DMRG	Número de iteraciones (*loops*)	150
	Pasos de descenso por enlace	2
	Minilotes por iteración	todos (pase completo)
	Tasa de aprendizaje inicial	8×10^{-4}
	Factor de reducción de la tasa	0,5
	Tasa de aprendizaje mínima	5×10^{-5}
	Paciencia (reducción de la tasa)	5
	Umbral de mejora	10^{-3}
	Paciencia (parada temprana)	15
	Tamaño de lote	1024

Continúa en la siguiente página

Tabla 3: Hiperparámetros del experimento, continuación.

Bloque	Parámetro	Valor
	Métrica de parada	NLL de validación
Datos y evaluación	Fracción de validación	0,15
	Percentiles de umbral reportados	94,5, 99
	Barrido de percentiles	90–99,5 (paso 0,5)
Aleatoriedad	Semilla global	123

3.3.7.2. Disponibilidad del código

La totalidad del código fuente del sistema, junto con los scripts de orquestación que reproducen cada etapa del pipeline y los artefactos generados, está disponible de forma pública en el repositorio del proyecto: `https://github.com/javierahumada4/Tensor-Networks-for-XAI` [39]. Cada script recibe como argumento el directorio de datos y reproduce su etapa de forma independiente, de manera que ejecutar en orden la codificación, el entrenamiento, la evaluación y la explicabilidad regenera todos los resultados presentados en este trabajo.

3.3.8. Tecnologías y recursos empleados

El sistema se ha desarrollado en Python 3.11, empleando PyTorch 2.9 como única librería de cálculo tensorial. Una decisión de diseño central de este trabajo es que tanto la representación del *Matrix Product State* como el algoritmo de entrenamiento DMRG y la totalidad de los mecanismos de explicabilidad se han implementado desde cero sobre tensores de PyTorch, sin recurrir a bibliotecas especializadas de redes tensoriales. Aunque esta elección incrementa el tiempo de desarrollo, otorga control completo sobre cada operación tensorial, lo que resulta esencial para implementar con precisión el esquema de explicabilidad propuesto. El resto de librerías de terceros se limita a tareas auxiliares. NumPy y pandas se utilizan para la manipulación de datos tabulares, scikit-learn para el cálculo de las métricas de evaluación y matplotlib para la generación de figuras.

El entrenamiento y la evaluación se realizaron sobre un ordenador portátil con procesador Intel Core Ultra 7 255HX de 20 núcleos. Se utilizó un entorno WSL2 con Ubuntu 24.04 sobre Windows, al que se le asignaron 15GiB de memoria. Aunque el equipo dispone de una GPU NVIDIA GeForce RTX 5070 Ti con CUDA, el entrenamiento se ejecutó sobre CPU. Esta elección no supone una limitación para el método, puesto que el algoritmo DMRG de dos sitios se realiza mediante un barrido secuencial de la cadena, actualizando un enlace cada vez y resolviendo una descomposición en valores singulares por enlace. Este patrón de cómputo está dominado por operaciones de álgebra lineal de pequeño tamaño ejecutadas de forma encadenada, por lo que no se beneficia de la paralelización masiva de la GPU en la misma medida que el entrenamiento de una red neuronal densa.

Capítulo 4 Resultados

Este capítulo presenta los resultados obtenidos al ejecutar el flujo completo del sistema sobre NSL-KDD y validarlo sobre el subconjunto de test descrito en la Sección 3.3.2.2. Los resultados siguen el orden del pipeline descrito en la Sección 3.3.6. En primer lugar, se describe el resultado de la codificación, después los resultados del entrenamiento, a continuación la evaluación de la capacidad de detección utilizando las métricas definidas en la Sección 3.3.4, y por último los diagnósticos de explicabilidad descritos en la Sección 3.2.4.

4.1. Resultado de la codificación

La etapa de codificación descrita en la Sección 3.2.2 transforma las 40 características heterogéneas del conjunto de datos en características discretas aptas para ser utilizadas como índices físicos del MPS. La Tabla 4 muestra el esquema resultante de ajustar el codificador sobre el conjunto de entrenamiento. Para cada característica se indican el tipo asignado y su dimensión física, junto con la información que define la codificación según el caso: el vocabulario en las características categóricas, los valores posibles en las discretas, los límites de los intervalos en las numéricas y el valor de referencia en las características que son constantes sobre el tráfico benigno.

El resultado más llamativo es que 17 de las 40 características son constantes sobre el tráfico benigno de entrenamiento. En estos casos, el codificador las reduce a un sitio binario que solo distingue si el valor coincide con el valor característico del tráfico normal o se aparta de él. Esto indica que una parte sustancial de las características del conjunto de datos apenas varía en condiciones normales, por lo que su utilidad para la detección reside precisamente en identificar desviaciones respecto a ese valor de referencia.

Además, de las 19 características numéricas que se discretizan por cuantiles, la mayoría queda dividida en solo dos intervalos. Esto sugiere que sus distribuciones benignas están fuertemente concentradas y que un único corte ya captura la mayor parte de la variación observada en el tráfico normal.

Las dos características con mayor dimensión física son las categóricas `service` (d=27) y `flag` (d=11), ya que en una variable categórica la dimensión del sitio viene dada por el tamaño de su vocabulario. El resto de características se mantiene en dimensiones bajas, entre 2 y 4, lo que contribuye a mantener acotado el coste computacional del modelo.

Tabla 4: Esquema de codificación producido por el codificador, con los 40 sitios en el orden de la cadena del MPS. La dimensión física d_k es el número de valores discretos que puede tomar el sitio. La última columna recoge la información específica de cada tipo de característica: los cortes de discretización en las numéricas, con los extremos $\pm\infty$ implícitos; el vocabulario de valores en las categóricas y discretas, donde `UNKNOWN/OTHER` corresponde al índice reservado para valores no observados durante el ajuste; o el valor normal de referencia en las características constantes sobre el tráfico benigno.

Sitio	Característica	Tipo	d_k	Límites de intervalo / vocabulario / valor normal
0	`duration`	Numérico	2	Cortes: 0
1	`protocol_type`	Categórico	4	`icmp, tcp, udp, UNKNOWN`
2	`service`	Categórico	27	`IRC, X11, auth, domain, domain_u, eco_i, ecr_i, finger, ftp, ftp_data, http, imap4, ntp_u, other, pop_3, private, red_i, shell, smtp, ssh, telnet, tftp_u, tim_i, time, urh_i, urp_i, UNKNOWN`
3	`flag`	Categórico	11	`OTH, REJ, RSTO, RSTR, S0, S1, S2, S3, SF, SH, UNKNOWN`
4	`src_bytes`	Numérico	4	Cortes: 129, 233, 324
5	`dst_bytes`	Numérico	4	Cortes: 105, 379, 2056
6	`land`	Constante	2	Valor normal: 0
7	`wrong_fragment`	Constante	2	Valor normal: 0
8	`urgent`	Constante	2	Valor normal: 0
9	`hot`	Constante	2	Valor normal: 0
10	`num_failed_logins`	Constante	2	Valor normal: 0
11	`logged_in`	Discreto	3	`0, 1, OTHER`
12	`num_compromised`	Constante	2	Valor normal: 0
13	`root_shell`	Constante	2	Valor normal: 0
14	`su_attempted`	Constante	2	Valor normal: 0
15	`num_root`	Constante	2	Valor normal: 0
16	`num_file_creations`	Constante	2	Valor normal: 0
17	`num_shells`	Constante	2	Valor normal: 0
18	`num_access_files`	Constante	2	Valor normal: 0
19	`is_host_login`	Constante	2	Valor normal: 0
20	`is_guest_login`	Constante	2	Valor normal: 0
21	`count`	Numérico	4	Cortes: 1, 4, 14
22	`srv_count`	Numérico	4	Cortes: 2, 5, 18
23	`serror_rate`	Constante	2	Valor normal: 0
24	`srv_serror_rate`	Constante	2	Valor normal: 0
25	`rerror_rate`	Constante	2	Valor normal: 0
26	`srv_rerror_rate`	Numérico	2	Cortes: 0
27	`same_srv_rate`	Numérico	2	Cortes: 1
28	`diff_srv_rate`	Numérico	2	Cortes: 0
29	`srv_diff_host_rate`	Numérico	2	Cortes: 0,11

(continúa en la página siguiente)

(continuación de la página anterior)

Sitio	Característica	Tipo	d_k	Límites de intervalo / vocabulario / valor normal
30	`dst_host_count`	Numérico	3	Cortes: 40, 156
31	`dst_host_srv_count`	Numérico	2	Cortes: 121
32	`dst_host_same_srv_rate`	Numérico	2	Cortes: 0,75
33	`dst_host_diff_srv_rate`	Numérico	2	Cortes: 0,02
34	`dst_host_same_src_port_rate`	Numérico	3	Cortes: 0,01, 0,08
35	`dst_host_srv_diff_host_rate`	Numérico	2	Cortes: 0,03
36	`dst_host_serror_rate`	Numérico	2	Cortes: 0
37	`dst_host_srv_serror_rate`	Numérico	2	Cortes: 0
38	`dst_host_rerror_rate`	Numérico	2	Cortes: 0
39	`dst_host_srv_rerror_rate`	Numérico	2	Cortes: 0

4.2. Resultado del entrenamiento

El modelo se entrena por máxima verosimilitud sobre las 57 242 conexiones normales de entrenamiento mediante el algoritmo de barrido DMRG de dos sitios descrito en la Sección 3.2.3. La Figura 7 muestra la evolución de la NLL media de entrenamiento y de validación a lo largo de los barridos, junto con la cota de dimensión de enlace, que se incrementa de forma progresiva durante el entrenamiento.

Ambas curvas descienden de forma sostenida desde una NLL media inicial superior a 10,0. La NLL de validación alcanza su mínimo, 8,30, en el barrido 28, momento en el que la NLL de entrenamiento es de 8,08. A partir de ese punto, la NLL de entrenamiento sigue bajando y la cota de las dimensiones de enlace sigue subiendo. Sin embargo, la NLL de validación deja de mejorar e incluso empeora ligeramente. Este comportamiento indica un posible sobreajuste, en el que la capacidad adicional proporcionada por las dimensiones de enlace más grandes se emplea en ajustarse a los datos de entrenamiento sin mejorar la generalización. La ejecución completa se prolongó hasta el barrido 43 para confirmar que la NLL de validación no volvía a mejorar.

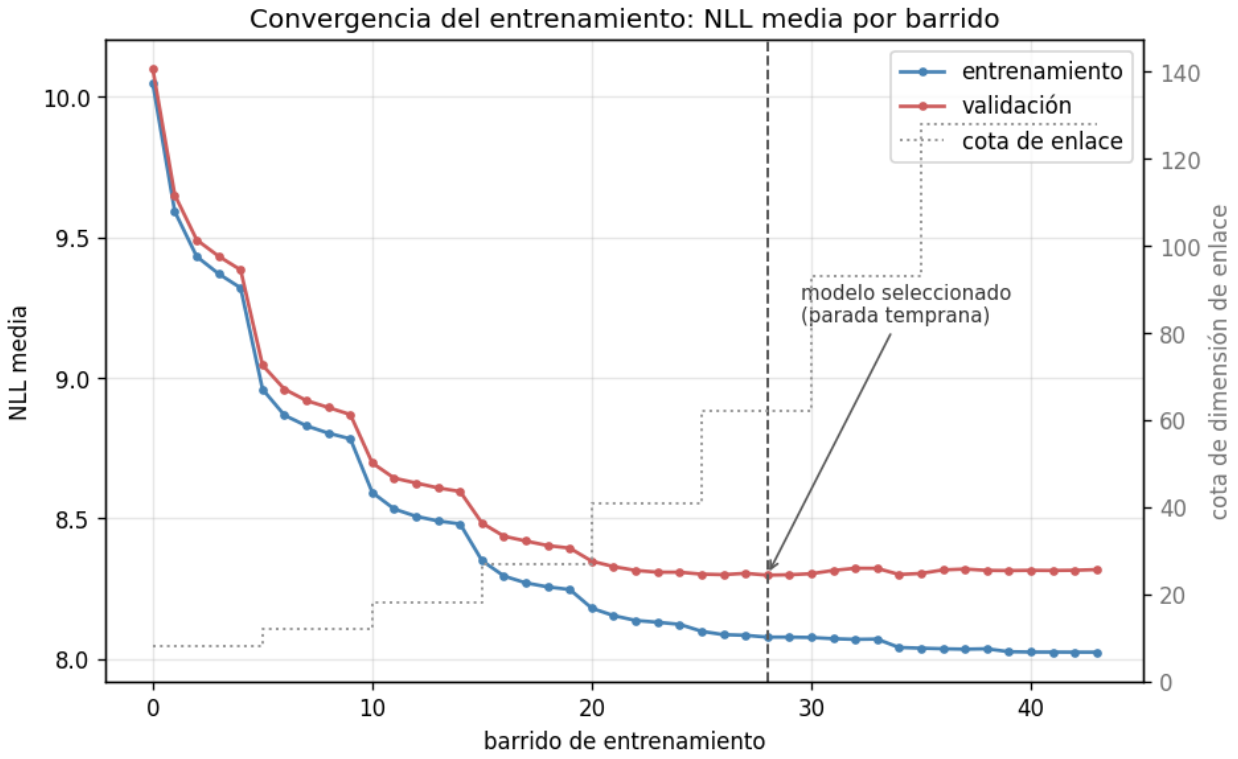


Figura 7: Evolución de la NLL media de entrenamiento y de validación por barrido. La línea de puntos indica la cota de dimensión de enlace, que crece de forma escalonada (eje derecho). La línea vertical discontinua marca el modelo seleccionado por parada temprana, en el mínimo de la NLL de validación.

Finalmente, el modelo seleccionado por el criterio de parada temprana es el del barrido 28, donde se alcanza el mínimo de la NLL de validación. Además, como en ese punto la cota de enlace todavía no había alcanzado su valor final, este modelo tiene un tamaño mucho menor que los modelos obtenidos en barridos posteriores. En concreto, sus dimensiones de enlace oscilan entre 2 y 62, mientras que otros modelos posteriores llegan a dimensiones de enlace de hasta 128. Esto supone una ventaja añadida en coste de cómputo y memoria. El perfil completo de dimensiones de enlace del modelo seleccionado es

$$(D_0, \ldots, D_{38}) = (2,\ 5,\ 62,\ 62,\ 62,\ 39,\ 20,\ 20,\ 40,\ 62,\ 62,\ 62,\ 62,$$
$$62,\ 62,\ 62,\ 62,\ 62,\ 62,\ 62,\ 62,\ 62,\ 62,\ 62,\ 62,\ 54,$$
$$27,\ 27,\ 54,\ 62,\ 62,\ 62,\ 62,\ 62,\ 32,\ 16,\ 8,\ 4,\ 2).$$

El entrenamiento completo duró aproximadamente 62 minutos con el equipo descrito en la Sección 3.3.8, mientras que el modelo seleccionado en el barrido 28 se alcanzó en torno a los 12 minutos.

4.3. Evaluación del modelo entrenado

En esta sección se evalúa el rendimiento del modelo entrenado en su capacidad como detector de intrusiones de red. Para ello, se asigna una puntuación de anomalía a cada observación del conjunto de test, que contiene 22 544 conexiones, de las que 9 711 son normales y 12 833 son ataques (aproximadamente un 56,9 %). Sobre estas puntuaciones se calculan las métricas mostradas en esta sección.

4.3.1. Separación global de las puntuaciones

La Figura 8 muestra la distribución de la puntuación de anomalía para las conexiones normales y de ataque por separado. Se observa que el modelo concentra las conexiones normales en valores bajos de NLL, mientras que la distribución de los ataques es mucho más dispersa y alcanza valores más altos. Sin embargo, ambas distribuciones se solapan parcialmente en la región intermedia, lo que anticipa que ningún umbral permite separar perfectamente las dos clases.

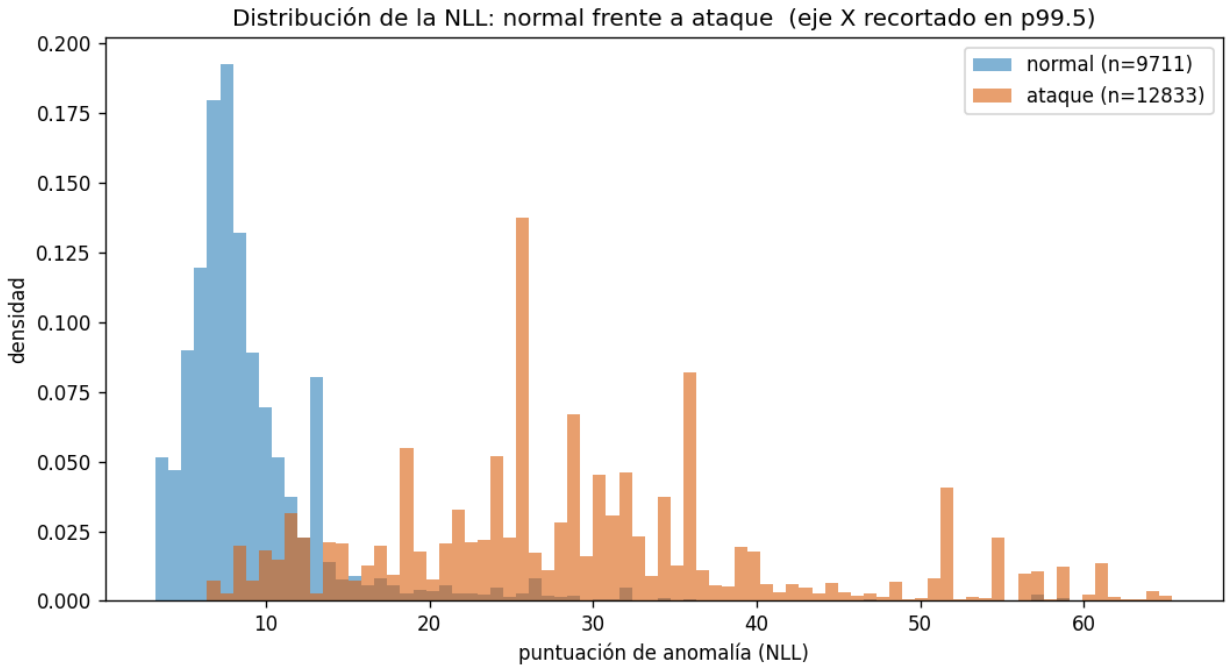


Figura 8: Distribución de la puntuación de anomalía (NLL) para las conexiones normales y de ataque del conjunto de test. El eje horizontal se ha recortado en el percentil 99,5 para facilitar la visualización.

Para cuantificar la separación real entre la NLL asignada a las conexiones benignas y a los ataques se emplean las métricas independientes del umbral explicadas en la Sección 3.3.4.2. La Figura 9 muestra las curvas ROC y PR.

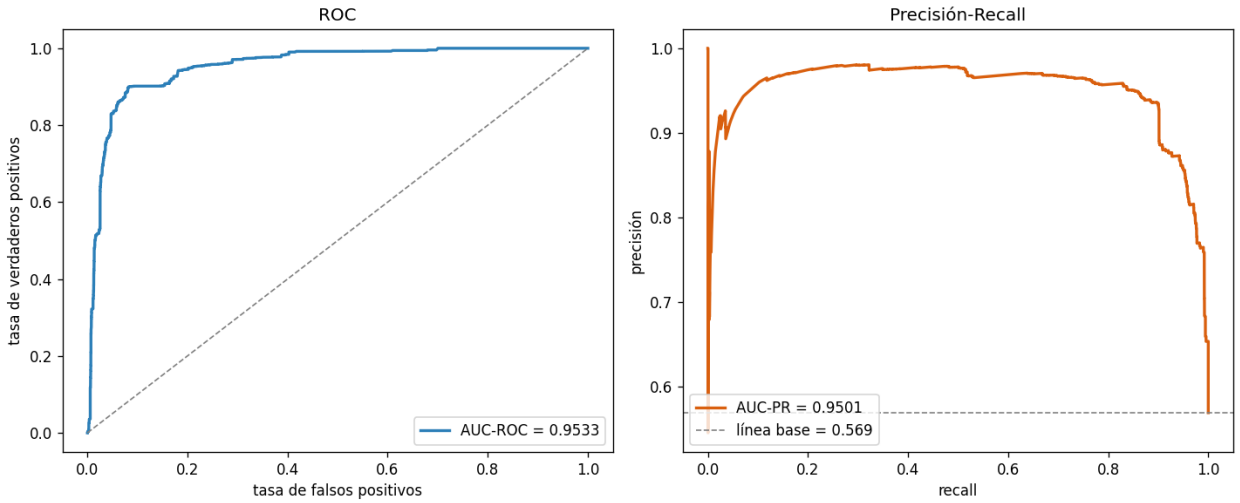


Figura 9: Curvas ROC (izquierda) y precisión-recall (derecha) sobre el conjunto de test completo. La curva de precisión-recall se mantiene por encima de 0,9 de precisión en casi todo el rango de recall, degradándose únicamente al exigir recalls muy próximos a uno.

En primer lugar, se observa que el modelo alcanza un AUC-ROC de 0,953 sobre el conjunto de test completo. Esto significa que, tomando al azar una conexión de ataque y una conexión normal, el modelo asigna una puntuación mayor a la de ataque en el 95,3 % de los casos, mucho mayor al 50 % esperado de un clasificador aleatorio.

Por otro lado, el modelo alcanza un AUC-PR de 0,950. Un AUC-PR elevado indica que el modelo mantiene una precisión alta a medida que aumenta el recall, es decir, que detecta una gran proporción de ataques sin que las conexiones marcadas como ataque dejen de ser mayoritariamente ataques reales. Puesto que la proporción de ataques en el conjunto de test es del 56,9 %, un clasificador aleatorio tendría una precisión esperada de 0,569. El amplio margen entre el AUC-PR obtenido de 0,950 y la línea base de 0,569 indica que la capacidad del modelo para separar las dos clases no se debe simplemente a la proporción de clases en el conjunto de datos.

4.3.2. Comportamiento a lo largo del umbral

En esta sección se caracteriza el comportamiento del modelo a lo largo de distintos umbrales. Cada umbral se fija a partir de un percentil de la puntuación de anomalía asignada por el modelo a las observaciones del conjunto de validación. Calibrar sobre el conjunto de validación permite fijar los umbrales utilizando datos que el modelo no ha visto en el entrenamiento y que son independientes del conjunto de test sobre el que se va a evaluar. Cada

percentil elegido determina la tasa de falsos positivos que cabría esperar si los datos de test se comportaran como los de validación.

La Figura 10 representa la precisión, el recall y el F1 sobre el conjunto de test al barrer el percentil del umbral entre el percentil 90 y el 99,5. En esta figura se observa el compromiso esperado entre precisión y recall, de forma que umbrales bajos detectan una mayor proporción de ataques a costa de generar más falsas alarmas, mientras que los umbrales más altos reducen las falsas alarmas pero dejan escapar más ataques. El F1 máximo, que mide el equilibrio entre precisión y recall, se obtiene en torno al percentil 94,5, con un valor de 0,917.

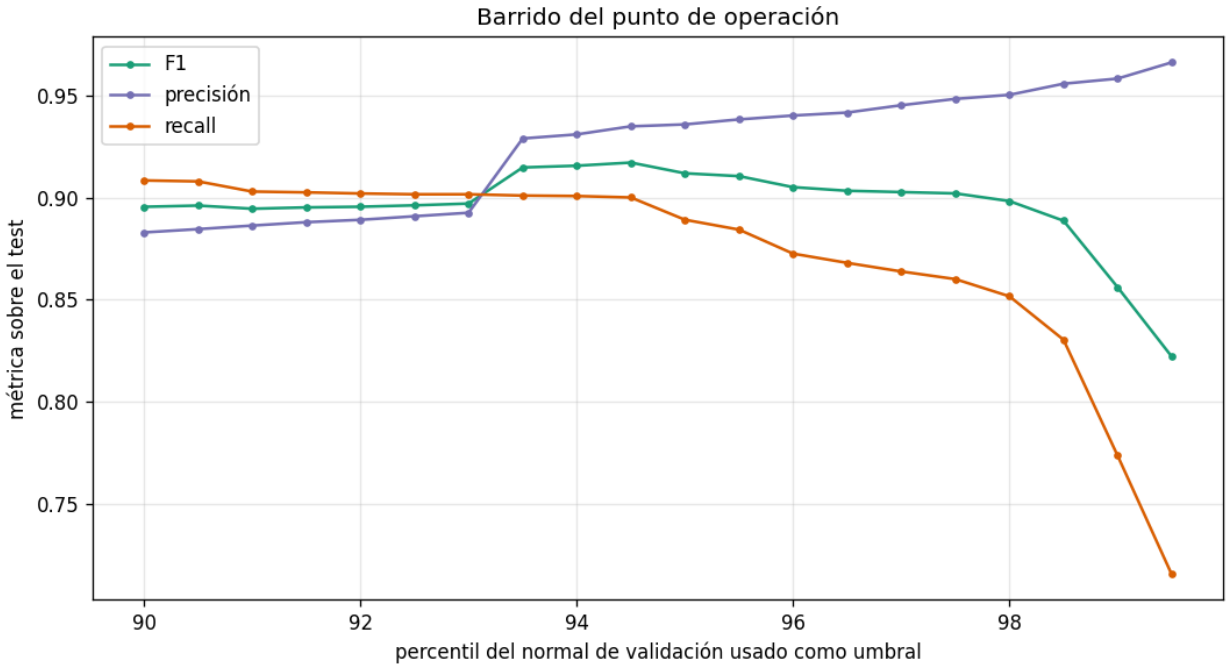


Figura 10: Precisión, recall y F1 sobre el conjunto de test en función del percentil del tráfico normal de validación empleado como umbral. A medida que el umbral se eleva, la precisión aumenta de forma monótona y el recall disminuye, reflejando el compromiso característico entre detección y falsas alarmas.

En la Figura 11 se muestran las matrices de confusión para el umbral del percentil 94,5, que es el que maximiza el F1, y el umbral del percentil 99, que es un punto de operación más conservador y adecuado para despliegues en los que se quiera reducir el número de falsas alertas. En el punto equilibrado (p94.5), el sistema detecta el 90,0 % de los ataques con una precisión del 93,5 %. En el punto más conservador (p99), la precisión sube al 95,8 %, generando menos falsas alertas a costa de reducir la proporción de ataques detectados al 77,4 %. La elección entre ambos, o cualquier otro umbral, depende del coste relativo que el operador asigne a una falsa alarma frente a un ataque no detectado.

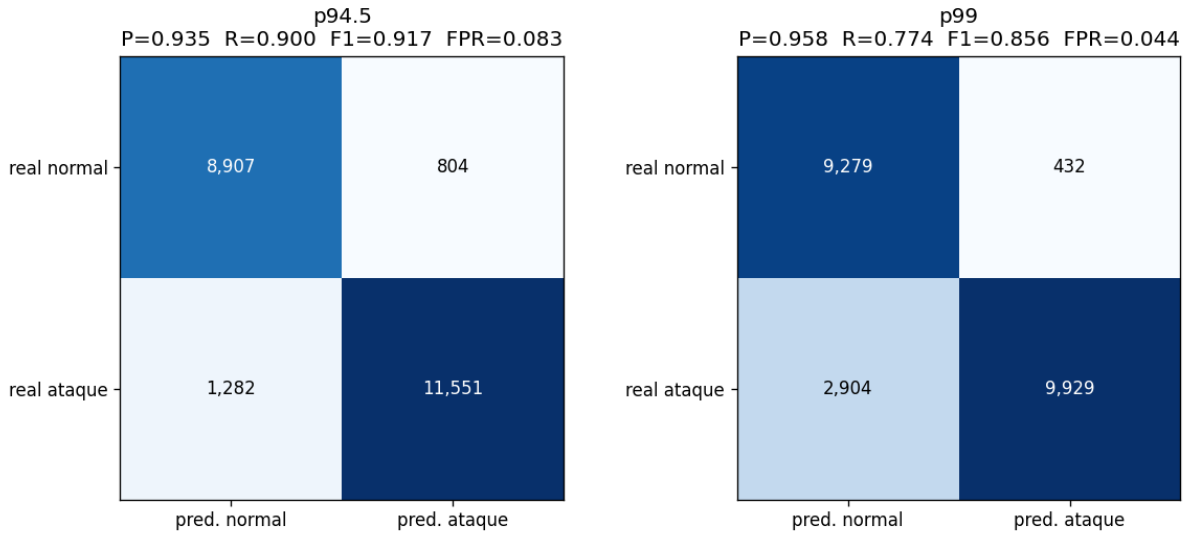


Figura 11: Matrices de confusión sobre el conjunto de test en los dos puntos de operación. A la izquierda, el percentil 94,5; a la derecha, el percentil 99.

4.3.3. Análisis desagregado por familia de ataque

También es importante evaluar el rendimiento del modelo frente a las distintas familias de ataque. Como ya se vio en la Sección 3.3.2.1, el conjunto de datos NSL-KDD presenta cuatro familias de ataque distintas: denegación de servicio (DoS), sondeo (Probe), remoto a local (R2L) y usuario a root (U2R). El conjunto de test tiene 9 711 conexiones normales y 7 458 observaciones de ataques DoS, pero tan solo 2 421 de Probe, 2 754 de R2L y únicamente 200 de U2R. Por tanto, un buen rendimiento global del modelo no garantiza que detecte bien todos los tipos de ataque.

La Tabla 5 muestra las métricas desagregadas por familia, calculando en cada caso la separación entre dicha familia de ataque y el tráfico normal.

Familia	n	AUC-ROC	AUC-PR	Línea base	× azar	Recall (p94.5)	Recall (p99)
DoS	7 458	**0,978**	**0,947**	0,434	2,2	**0,991**	0,855
Probe	2 421	0,968	0,851	0,200	4,3	0,957	**0,862**
R2L	2 754	0,879	0,660	0,221	3,0	0,635	0,508
U2R	200	0,889	0,107	0,020	**5,3**	0,475	0,330

Tabla 5: Métricas de detección desagregadas por familia de ataque, calculadas frente al tráfico normal. La columna *línea base* es la prevalencia de la familia dentro del subconjunto familia-vs-normal, es decir, el AUC-PR que alcanzaría un clasificador aleatorio; la columna × azar es el cociente entre el AUC-PR obtenido y dicha línea base. El recall se mide en los puntos de operación de los percentiles 94,5 y 99.

Los ataques de la familia DoS se detectan con gran fiabilidad, con un AUC-ROC de 0,978 y recalls del 99,1 % en el punto de operación equilibrado y del 85,5 % en el conservador. Este resultado es coherente con la naturaleza de los ataques de denegación de servicio. Puesto que su objetivo es saturar un servicio o recurso, suelen generar un volumen anómalo de conexiones, paquetes o peticiones en un corto periodo de tiempo, además de aumentar las tasas de error, los intentos fallidos de conexión o la repetición de patrones similares. Esto los convierte en ataques fáciles de detectar para un modelo entrenado como detector de anomalías, lo que explica su alta tasa de detección.

Los ataques de sondeo (Probe) también se detectan con gran fiabilidad, con AUC-ROC de 0,968 y recalls del 95,7 % en el punto de operación equilibrado y del 86,2 % en el conservador. Estos ataques generan patrones sistemáticos y poco habituales al sondear el sistema, por ejemplo mediante múltiples conexiones a distintos puertos o direcciones en poco tiempo. Por ello también se diferencian bien del tráfico normal y resultan fáciles de identificar para el modelo.

En cambio, los ataques de las familias R2L y U2R son mucho más difíciles de detectar, porque no suelen generar patrones de tráfico tan evidentes como los ataques DoS o Probe. R2L cae a un AUC-ROC de 0,879 mientras que U2R presenta un AUC-ROC de 0,889. Además, su detección es mucho más sensible al umbral: el recall de R2L pasa del 63,5 % en el punto equilibrado al 50,8 % en el conservador, y el de U2R del 47,5 % al 33,0 %. Estos valores reflejan la limitación intrínseca de un detector de anomalías ante ataques que se camuflan dentro de sesiones aparentemente legítimas.

A pesar de estas limitaciones, el rendimiento global del modelo sigue siendo alto porque el conjunto de test está dominado por las familias que el modelo detecta mejor. En concreto, DoS y Probe representan conjuntamente alrededor del 77 % de las conexiones de ataque, mientras que R2L y U2R suponen aproximadamente el 21,5 % y el 1,6 %, respectivamente. Son estos análisis desagregados los que permiten detectar el rendimiento más bajo sobre las familias de ataque infrarrepresentadas.

No obstante, las bajas métricas en R2L y U2R no implican que el modelo no extraiga ninguna señal útil sobre ellas. Utilizando la AUC-PR, podemos comparar el rendimiento del modelo frente al que alcanzaría un clasificador aleatorio. De esta forma, el AUC-PR de 0,107 en U2R, que en términos absolutos parece muy limitado, equivale en realidad a 5,3 veces el valor que alcanzaría un clasificador aleatorio sobre ese mismo subconjunto. Por su parte, el AUC-PR de R2L supera su línea base por un factor de 3. Estos multiplicadores muestran que el modelo también captura señal por encima del azar en las familias más escasas, aunque su rendimiento absoluto siga siendo modesto.

4.4. Explicabilidad del modelo entrenado

Esta sección muestra los diagnósticos de explicabilidad descritos en la Sección 3.2.4. A diferencia de las métricas anteriores, la mayoría de estas cantidades son propiedades intrínsecas del modelo, derivadas directamente de la contracción de sus tensores. Primero se analizan las lecturas directas de las matrices de densidad reducida, después las medidas de entrelazamiento y, por último, los análisis supervisados de importancia y la descomposición de anomalías individuales.

4.4.1. Extracción de probabilidades y entropía de Von Neumann

Como se presentó en la Sección 3.2.4.2, la primera lectura de la matriz de densidad reducida son las probabilidades marginales que el modelo asigna a cada valor de cada característica. A diferencia de un modelo de caja negra, del que solo se obtiene la puntuación final, en el MPS se puede leer cualquier probabilidad de la distribución aprendida, incluyendo marginales, condicionales y conjuntas.

La Figura 12 muestra la distribución aprendida para tres características. Estas marginales reflejan lo que el modelo considera tráfico normal. Por ejemplo, ha aprendido que el servicio se concentra en unos pocos valores, como `http`, y que la característica `flag` está dominada por el estado `SF`. Al comparar cada distribución con su frecuencia empírica se observa que ambas no son idénticas. Esta diferencia no es un defecto de ajuste, sino el resultado de que el MPS ha capturado correlaciones entre características, algo que un simple recuento de frecuencias no refleja. El caso más ilustrativo es el de `dst_bytes`, donde se observa que, mientras que empíricamente casi toda la masa se concentra en el intervalo inferior, la marginal del modelo se reparte de forma mucho más uniforme entre los cuatro intervalos.

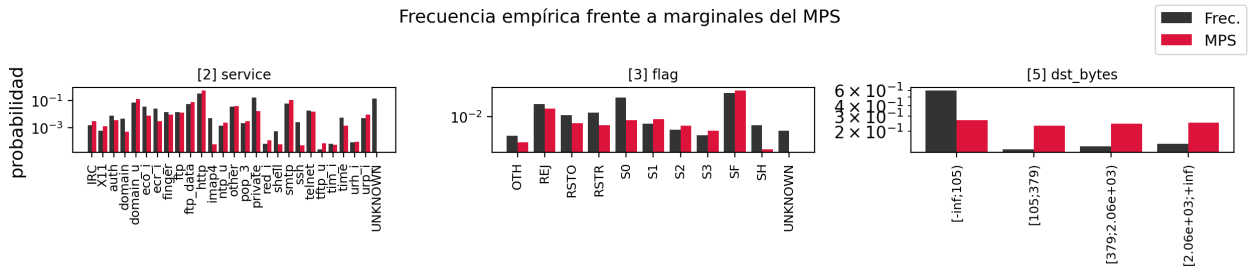


Figura 12: Comparación, para las tres características de mayor disparidad, entre la frecuencia empírica de cada valor en el tráfico benigno (barras oscuras) y la probabilidad marginal que el modelo extrae de la matriz de densidad reducida (barras rojas), en escala logarítmica. El eje horizontal muestra los valores de cada característica: el vocabulario en `service` y `flag` y los límites de los intervalos en `dst_bytes`.

Esta discrepancia puede interpretarse con la entropía de Von Neumann de cada característica, que mide su grado de entrelazamiento con el resto del sistema. La Figura 13 muestra dicha entropía para las 40 características. Los valores más altos corresponden a `service` (1,56), `count` (1,33), `srv_count` (1,28), `dst_bytes` (1,20) y `src_bytes` (1,16), que son por tanto las características que el modelo considera más correlacionadas con las demás y que tendrán, por tanto, mayor separación entre la marginal del modelo y la frecuencia empírica. Por el contrario, hay siete características que presentan una entropía prácticamente nula, lo que indica que el modelo las trata prácticamente como independientes del resto del sistema.

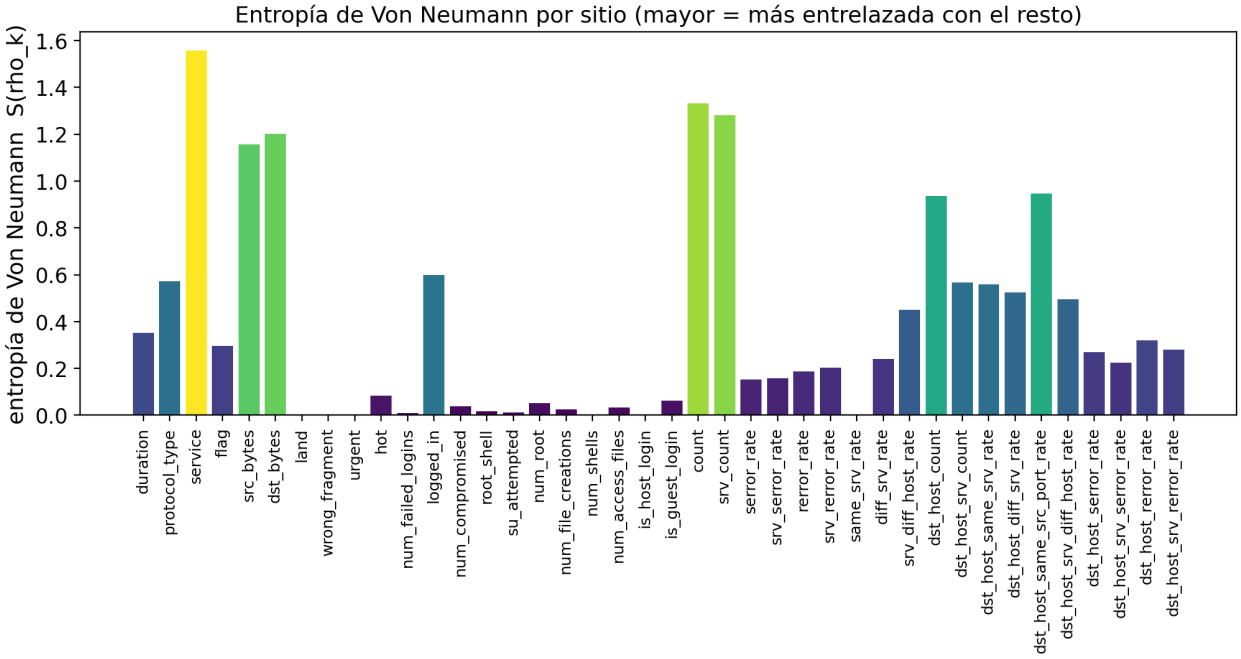


Figura 13: Entropía de Von Neumann de la matriz de densidad reducida de cada característica. Un valor alto indica que la característica está fuertemente entrelazada con el resto del sistema, y un valor próximo a cero, que el modelo la trata como casi independiente.

4.4.2. Información mutua y dependencias entre características

La información mutua entre pares de características cuantifica cuánta información comparten, lo que permite capturar dependencias estadísticas que una correlación lineal no podría detectar. En el MPS esta cantidad se obtiene directamente sin necesidad de estimadores externos, a partir de las entropías de las matrices de densidad reducidas mediante la relación $I(A;B) = S(\rho_A) + S(\rho_B) - S(\rho_{AB})$. La Figura 14 muestra la matriz completa para las 40 características.

La matriz es mayoritariamente oscura, lo que indica que la mayoría de pares de características apenas comparten información. El par con mayor información mutua es, con diferencia, `count` y `srv_count` ($I = 1{,}16$), seguido de `dst_bytes` y `src_bytes` ($I = 0{,}80$),

`dst_bytes` y `service` ($I = 0{,}71$), y `service` y `src_bytes` ($I = 0{,}69$). Estos pares coinciden con las características que ya destacaban por su alta entropía de Von Neumann.

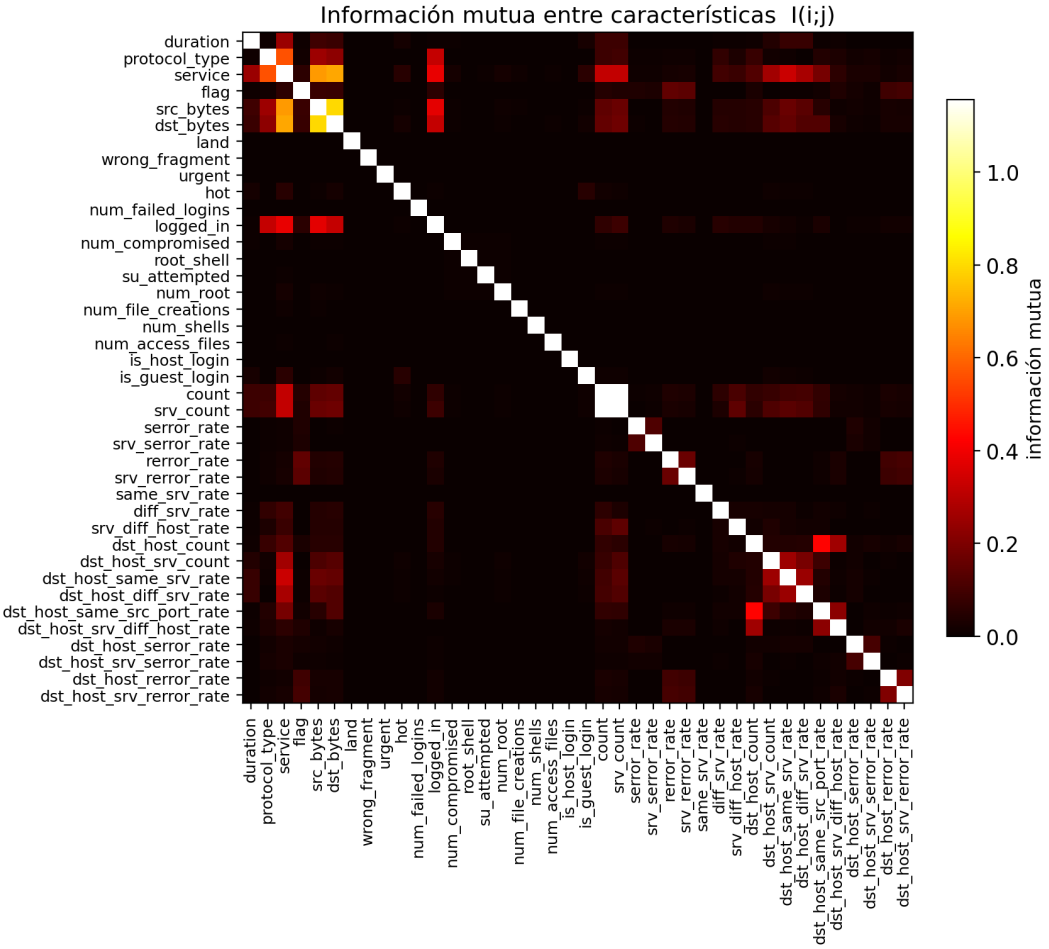


Figura 14: Información mutua entre pares de características derivada del modelo. Las celdas más claras señalan los pares con una dependencia más fuerte. Destaca el bloque formado por `count` y `srv_count` y el formado por `service`, `src_bytes` y `dst_bytes`.

Más allá de su valor interpretativo, esta medida tiene además una utilidad práctica, ya que permite identificar grupos de características redundantes o fuertemente dependientes. Además, existe la posibilidad de reordenar la cadena del MPS situando las características más correlacionadas en posiciones contiguas, lo que aprovecha de forma más eficiente la capacidad de las dimensiones de enlace.

Aunque la información mutua resume en un único escalar la intensidad de la dependencia entre dos características, no indica qué combinaciones concretas de valores son responsables de dicha dependencia. Para analizar este aspecto se estudian las probabilidades conjuntas mediante el *lift*, definido como el cociente entre la probabilidad conjunta de dos valores y el

producto de sus probabilidades marginales.

$$\text{lift}(a, b) = \frac{P(a, b)}{P(a)P(b)} \tag{4.1}$$

Un *lift* mayor que uno indica que una combinación de valores aparece más a menudo de lo que cabría esperar si las características fueran independientes, mientras que un *lift* menor que uno indica que aparece con menos frecuencia de la esperada.

Para ilustrar este análisis se utiliza como ejemplo el par de características con mayor información mutua, `count` y `srv_count`. La Figura 15 muestra el resultado. Se observa que las combinaciones en las que ambas características toman valores bajos, o ambas valores altos, aparecen con mucha más frecuencia de la que cabría esperar si fueran independientes. Por el contrario, las combinaciones cruzadas, en las que una es alta y la otra baja, aparecen con frecuencias muy inferiores a las esperadas. El modelo ha aprendido, por tanto, que estas dos características tienden a crecer y decrecer juntas, una relación que un análisis basado únicamente en marginales no podría revelar.

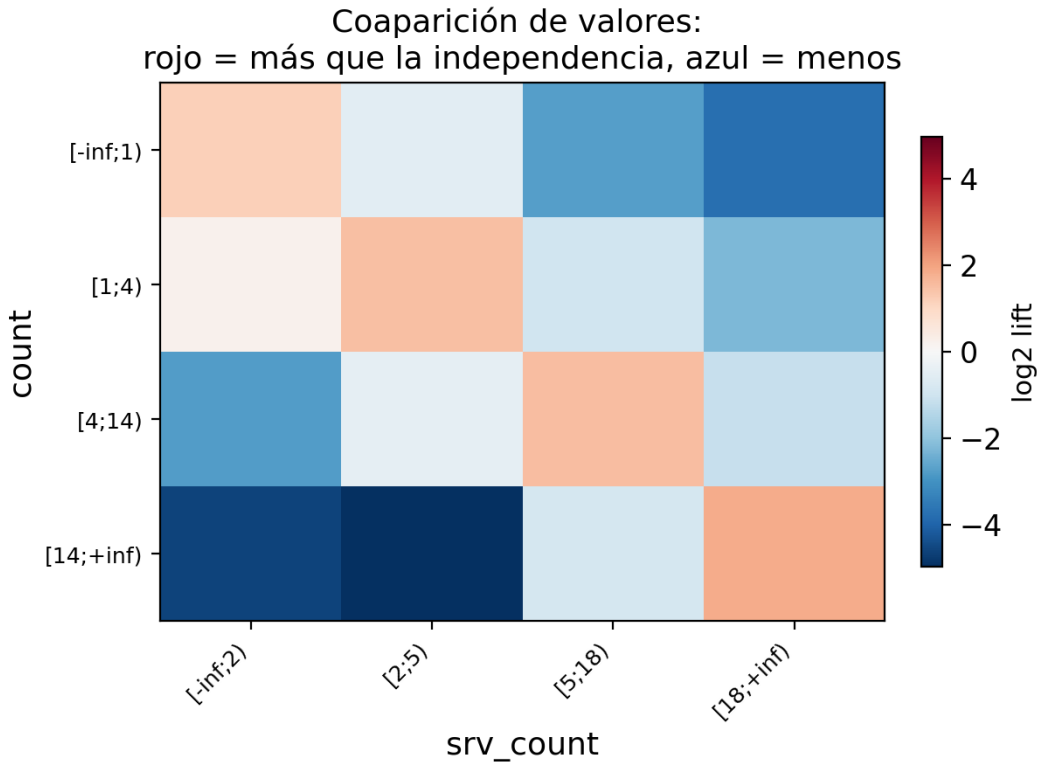


Figura 15: Cociente entre la probabilidad conjunta y el producto de marginales para el par `count`–`srv_count`. Los valores mayores que uno (diagonal) indican combinaciones que aparecen más de lo esperado bajo independencia, y los menores que uno, combinaciones infrarrepresentadas.

También se puede realizar un análisis según cómo los distintos valores de una característica j condicionan la distribución de una característica i. Como ejemplo, en la Figura 16 se muestra cómo cambia la distribución de `count` al fijar el valor de `srv_count` a su intervalo más bajo. Se observa cómo la masa de probabilidad de `count` se concentra en su intervalo inferior, que pasa de una probabilidad marginal del 31 % a una probabilidad condicionada del 71 %. Además, se observa cómo la probabilidad de los intervalos superiores disminuye de forma progresiva a medida que aumenta el valor de `count`, lo que refleja que ambas características covarían.

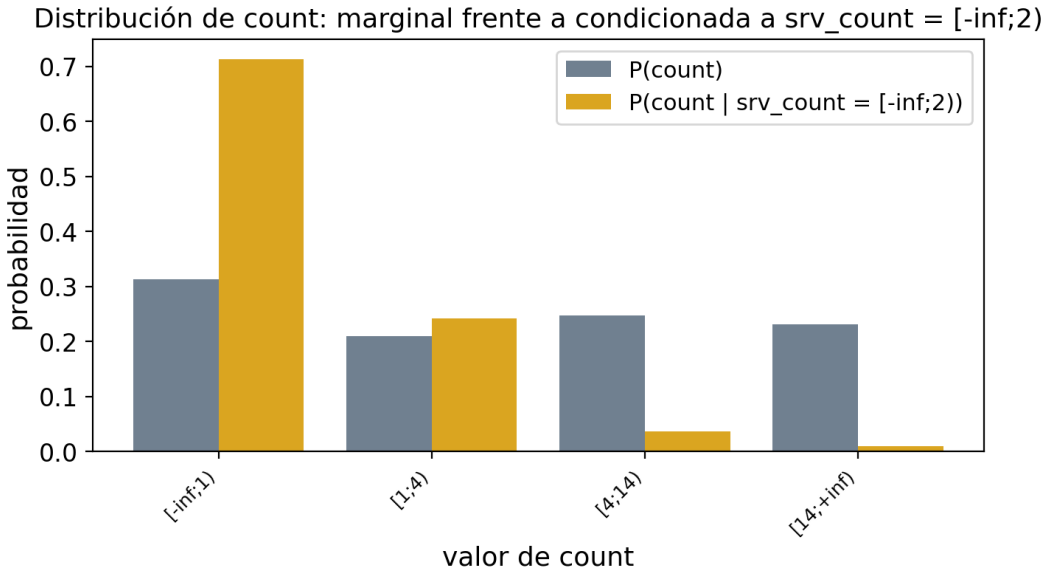


Figura 16: Distribución de `count` sin condicionar (marginal) y condicionada a un valor fijo de `srv_count`. El condicionamiento concentra la masa de probabilidad en el intervalo más bajo de `count`.

4.4.3. Entropía de enlace

A diferencia de las medidas anteriores, la entropía de enlace no es una propiedad de una característica concreta, sino de cada corte que separa la cadena en cada uno de los 39 enlaces. La Figura 17 representa, para cada enlace, su entropía junto con la cota teórica $\log D_k$ que impone su dimensión de enlace. La entropía de enlace más alta se alcanza en el corte situado entre `count` y `srv_count` ($S = 2{,}65$), seguido por los cortes situados en el bloque de características de host de destino, como `dst_host_count` y `dst_host_srv_count` ($S = 2{,}34$). Estos enlaces son los que transportan la mayor cantidad de correlación a lo largo de la cadena, lo que es coherente con la presencia a ambos lados del corte de algunas de las características más correlacionadas del modelo.

Por otro lado, se observa que la entropía de enlace nunca se acerca a cero, su valor mínimo es 0,28 en el corte entre las dos últimas características y su valor medio a lo largo de la cadena es de 1,78. Esto indica que todos los enlaces transportan una cantidad apreciable de correlación, por lo que el modelo está aprovechando su estructura tensorial. En cuanto a la saturación, definida como el cociente entre la entropía de cada enlace y su cota, los valores máximos rondan el 0,66, lo que indica que ningún enlace agota por completo su capacidad. Por tanto, la dimensión de enlace asignada durante el entrenamiento es suficiente para representar las correlaciones presentes en el tráfico normal, sin que ningún corte quede limitado por su capacidad.

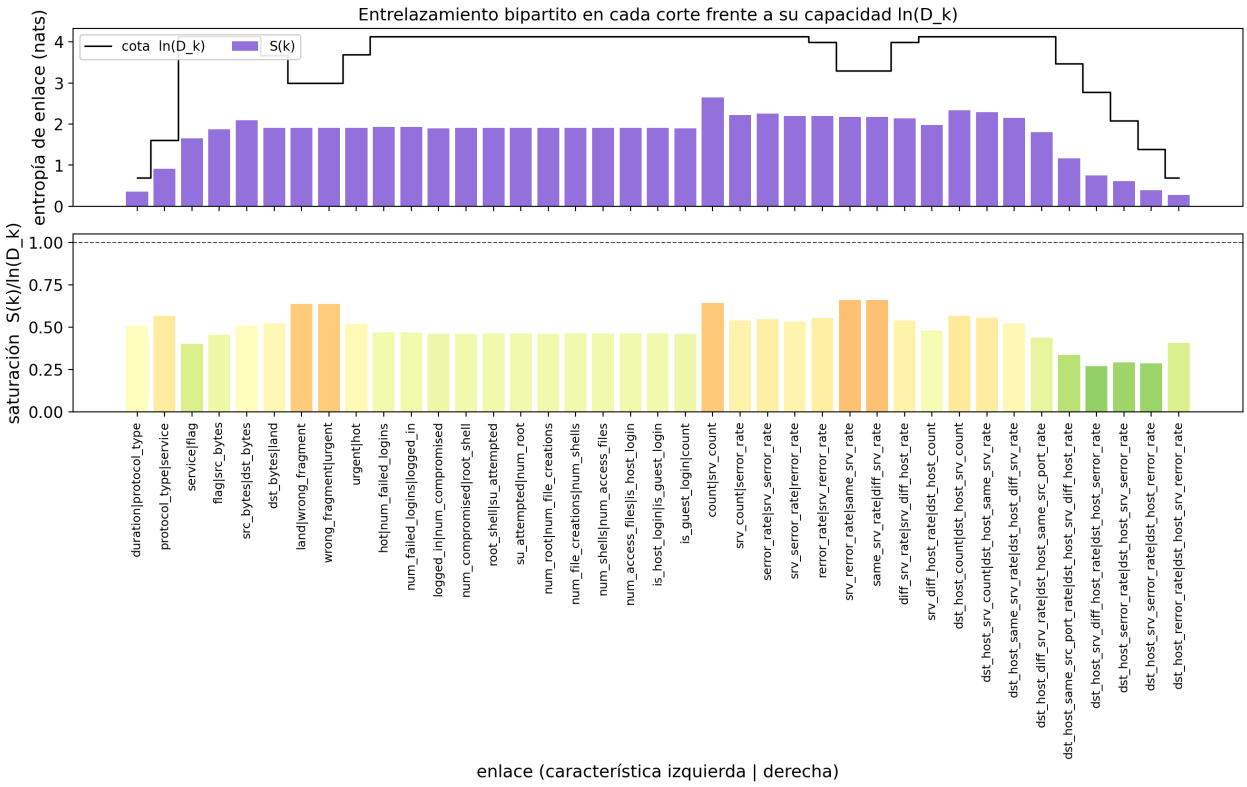


Figura 17: Entropía de cada uno de los 39 enlaces de la cadena frente a su cota teórica $\log D_k$. La saturación, definida como el cociente entre ambas, indica qué enlaces operan cerca de su capacidad.

4.4.4. Importancia de las características

A diferencia de todas las medidas anteriores, la importancia de las características es un análisis supervisado a posteriori, ya que utiliza las etiquetas del conjunto de test para separar las observaciones benignas de las amenazas. Para cada característica se mide la diferencia Δ_i entre la probabilidad media que el modelo asigna a los valores observados en el tráfico benigno y la que asigna a los valores observados en los ataques. Un valor alto de Δ_i indica que, en

esa característica, los ataques presentan valores menos probables que los del tráfico benigno. Por tanto, dicha característica contribuye a que las anomalías detectadas se correspondan con ataques reales y no simplemente con valores que el modelo consideraría improbables por igual en ambas clases. Un valor próximo a cero o negativo, en cambio, indica que la característica, considerada de forma aislada, no separa ataques de tráfico normal.

La Figura 18 ordena las características por esta magnitud. Las características más discriminativas entre el tráfico normal y los ataques son `flag` ($\Delta = 0{,}53$) y `dst_host_rerror_rate` (0,46). Conviene observar que algunas características presentan un Δ_i cercano a cero o incluso negativo, como `srv_diff_host_rate` o `dst_host_count`. Esto indica que los valores que toman en los ataques son tan probables, o incluso más, que los del tráfico benigno. Estas características no contribuyen por sí solas a distinguir amenazas, aunque sí puedan hacerlo a través de sus correlaciones con otras. Por tanto, esta medida solo evalúa la importancia de cada característica de forma aislada, una limitación que se deriva del carácter marginal de esta medida y que ya se señaló en la metodología.

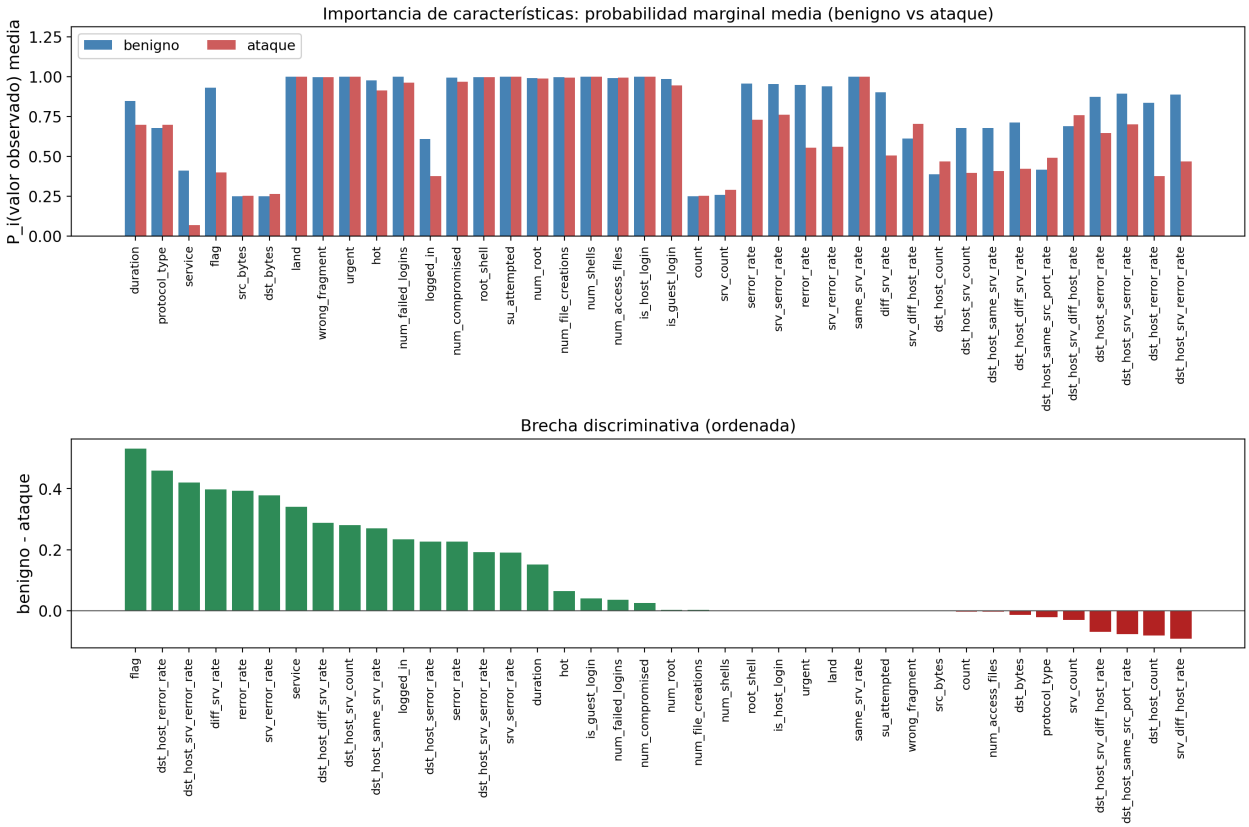


Figura 18: Importancia de cada característica medida como la diferencia Δ_i entre la probabilidad media que el modelo asigna al valor observado en el tráfico benigno y en los ataques. Un valor alto indica que las amenazas presentan valores poco probables según el modelo en esa característica.

El mismo análisis se puede desglosar por familia de ataque, midiendo la diferencia de la probabilidad media asignada a cada familia con la del tráfico normal. La Figura 19 muestra que las características relevantes difieren entre familias. Por ejemplo, en DoS, son especialmente importantes `flag` ($\Delta = 0{,}69$) y las variables asociadas a errores, como `dst_host_rerror_rate` ($\Delta = 0{,}51$). Esto indica que, para esta familia, el modelo distingue los ataques principalmente por patrones anómalos en el estado de las conexiones y en las tasas de error, más que por una única característica de volumen. Por otro lado, en R2L cobran mayor importancia `service` ($\Delta = 0{,}39$) y `duration` ($\Delta = 0{,}39$). Esto concuerda con que estos ataques se distingan más por el tipo de servicio y la duración de la sesión que por anomalías de volumen.

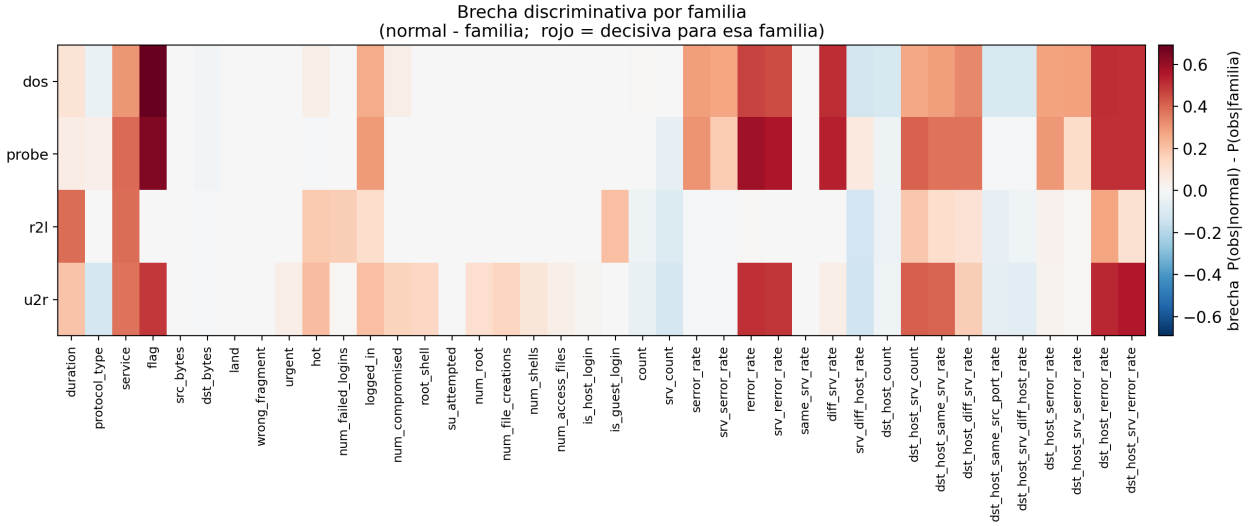


Figura 19: Importancia de las características desglosada por familia de ataque. Las características más discriminativas difieren de una familia a otra, lo que refleja la distinta naturaleza de cada tipo de ataque.

4.4.5. Descomposición de anomalías individuales

La capacidad explicativa más relevante desde el punto de vista operativo es la descomposición de la puntuación de una conexión en las contribuciones de cada característica. Ante una alerta, este desglose indica al analista qué valores han contribuido en mayor medida a que la conexión se considere anómala. Como ya se comentó en la Sección 3.2.4.8, esta métrica está limitada porque no captura la información que aportan las correlaciones del modelo. Por este motivo, en este trabajo se ha introducido una métrica adicional que indica al analista la fiabilidad de cada explicación. Esta métrica es la cuota de correlación, que mide qué fracción de la NLL real de la observación no queda explicada por la suma de las contribuciones marginales.

$$c = \frac{|\, r \,|}{\mathrm{NLL}(\mathbf{x})} = \frac{\left|\, \mathrm{NLL}(\mathbf{x}) - \sum_i \big(-\log P_i(v_i)\big) \,\right|}{\mathrm{NLL}(\mathbf{x})} \tag{4.2}$$

Una cuota baja indica que la suma de las contribuciones por característica reproduce casi por completo la puntuación asignada por el modelo, por lo que el desglose marginal es fiel a la decisión real del MPS y puede interpretarse directamente. Por el contrario, una cuota alta indica que una parte importante de la anomalía no reside en valores anómalos individuales, sino en la combinación de valores observada. En este caso, el desglose por características es incompleto y el analista deberá analizar la estructura de correlaciones.

La Figura 20 muestra la descomposición de distintas observaciones a las que el modelo ha asignado una alta puntuación de anomalía. En primer lugar, tenemos tres observaciones con una cuota de correlación casi nula (entre el 0,1 % y el 0,3 %). Esto significa que las contribuciones marginales reproducen prácticamente toda la puntuación de anomalía real, por lo que el desglose por características puede considerarse fiable y un analista puede leer directamente qué variables explican la alerta. Por ejemplo, en la primera observación, correspondiente a un ataque de sondeo, observamos que lo que más ha contribuido a la puntuación de anomalía son valores poco habituales en `service`, `flag` y en las tasas de error de la conexión. Esta explicación es coherente con el comportamiento esperado en un sondeo de servicios, donde la anomalía suele manifestarse en el tipo de servicio solicitado y en patrones anómalos de respuesta o error.

Por otro lado, la figura también incluye tres observaciones con una cuota de correlación elevada, en torno al 35 %. En estos casos, la explicación marginal deja de ser representativa de la decisión del modelo. Tomando como ejemplo la última observación, se observa que la suma de las contribuciones marginales sobreestima la puntuación real. Mientras que la NLL asignada por el modelo es de solo 48,9, las contribuciones por característica suman 66,0. La diferencia entre ambas cantidades produce un residuo de correlación de $-17{,}1$ y una cuota del 35,1 %, lo que advierte al analista de que no debe fiarse plenamente del desglose por características. Al analizar la observación en mayor detalle, se observa que la característica `service` aporta por sí sola cerca de 36 unidades de NLL, lo que indica que el valor observado es extremadamente improbable de forma aislada. Sin embargo, el MPS no evalúa cada característica de forma aislada, sino en relación con el resto, y en este caso las demás variables proporcionan un contexto en el que el valor de `service` es más probable. En este caso, fiarse únicamente de las contribuciones marginales llevaría a una explicación engañosa, y la cuota de correlación elevada sirve de advertencia de ello.

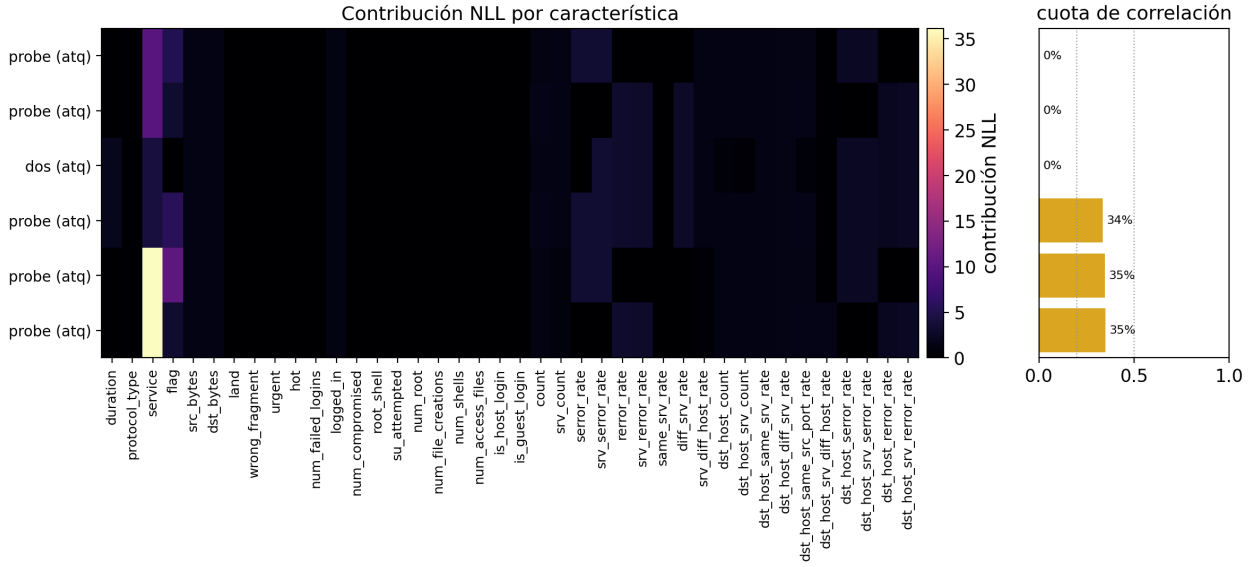


Figura 20: Descomposición de la puntuación de anomalía en las contribuciones de cada característica, para las conexiones de menor (arriba) y mayor (abajo) cuota de correlación de entre las más anómalas. El panel derecho muestra dicha cuota: en verde cuando las contribuciones por característica explican la puntuación (desglose fiable) y en rojo cuando la mayor parte reside en las correlaciones (desglose engañoso).

Capítulo 5 Conclusiones y Trabajo Futuro

5.1. Conclusiones

En este Trabajo de Fin de Grado se ha desarrollado, implementado y evaluado un sistema de detección de anomalías basado en un modelo generativo *Matrix Product State* (MPS), aplicado al problema de detección de intrusiones de red mediante el conjunto de datos NSL-KDD. El enfoque propuesto se basa en entrenar el modelo únicamente con observaciones normales, de forma que las conexiones poco probables bajo la distribución aprendida puedan ser identificadas como anómalas.

Los resultados obtenidos muestran que los MPS son una alternativa viable para modelar distribuciones de probabilidad de datos tabulares discretizados. El modelo entrenado fue capaz de asignar puntuaciones de anomalía significativamente mayores a las conexiones de ataque que a las conexiones normales, alcanzando un AUC-ROC de 0,953 y un AUC-PR de 0,950 sobre el conjunto de test. Además, el barrido de umbrales permitió identificar distintos puntos de operación en función del compromiso deseado entre precisión y *recall*. En particular, el mejor equilibrio se obtuvo en torno al percentil 94,5, con un valor de F1 de 0,917.

El análisis desagregado por familias de ataque permitió comprobar que el sistema no presenta el mismo rendimiento ante distintos tipos de amenaza. Mientras que las familias DoS y Probe fueron detectadas con gran fiabilidad, el modelo presentó un rendimiento más limitado a la hora de detectar ataques R2L y U2R. Este resultado es coherente con la naturaleza de estas familias, puesto que los ataques DoS y Probe generan patrones de tráfico claramente más anómalos, mientras que los ataques R2L y U2R se asemejan más al comportamiento normal y aparecen con menor frecuencia en el conjunto de datos.

Desde el punto de vista de la interpretabilidad, el trabajo ha mostrado que la estructura del MPS permite extraer distintas medidas internas del modelo sin recurrir a técnicas externas de explicabilidad. En particular, se han analizado probabilidades marginales, probabilidades condicionales, probabilidades conjuntas, entropía de Von Neumann, información mutua, entropía de enlace e importancia de características. Estas medidas permiten estudiar tanto qué valores son poco probables de forma individual como qué dependencias entre variables han sido aprendidas por el modelo.

Una de las principales aportaciones del trabajo ha sido la incorporación de la cuota de correlación como medida complementaria para interpretar anomalías individuales. Esta medida permite distinguir entre anomalías que son explicables a partir de las marginales de sus valores y anomalías cuya puntuación depende en mayor medida de las correlaciones entre variables. De esta forma, se evita interpretar incorrectamente una alerta cuando la descomposición por características no reproduce fielmente la puntuación real asignada por el MPS.

En conjunto, los resultados permiten concluir que los modelos MPS no solo pueden utilizarse como detectores generativos de anomalías, sino que también ofrecen una estructura adecuada para analizar e interpretar las decisiones del modelo.

5.2. Limitaciones

A pesar de los resultados obtenidos, el trabajo presenta varias limitaciones. En primer lugar, la validación experimental se ha realizado sobre un único conjunto de datos, NSL-KDD. Aunque este conjunto permite estudiar de forma clara el problema de detección de intrusiones, sería necesario evaluar el enfoque en otros conjuntos de datos más recientes y en dominios distintos donde las características de los datos y los tipos de anomalía pueden ser diferentes.

En segundo lugar, el modelo requiere una etapa previa de discretización de las variables. Esta codificación permite adaptar datos tabulares heterogéneos a la estructura del MPS, pero también puede provocar pérdida de información en variables continuas. Por tanto, la calidad de la discretización influye directamente tanto en el rendimiento del detector como en la interpretabilidad de las probabilidades extraídas.

Otra limitación importante está relacionada con la naturaleza marginal de algunas de las métricas de interpretabilidad utilizadas. Medidas como las probabilidades marginales o ciertas estimaciones de importancia de características analizan el comportamiento de cada variable de forma individual, por lo que no siempre capturan adecuadamente las dependencias entre variables que el modelo MPS sí puede estar utilizando para asignar una determinada puntuación de anomalía. En consecuencia, una explicación basada únicamente en este tipo de métricas puede resultar incompleta, especialmente en aquellos casos en los que la anomalía se debe principalmente a combinaciones poco frecuentes de valores y no a valores individuales anómalos.

Por último, aunque se han analizado distintas medidas internas de explicabilidad, no se ha realizado una evaluación exhaustiva de la estabilidad, fidelidad y utilidad práctica de las explicaciones generadas. En particular, no se ha estudiado de forma sistemática cómo varían las explicaciones ante pequeñas perturbaciones de entrada o del modelo, ni hasta qué punto son coherentes entre muestras similares o útiles para un analista humano en un escenario real.

5.3. Trabajo futuro

Como línea de trabajo futuro, sería interesante evaluar el sistema propuesto sobre problemas de detección de anomalías en otros dominios como fraude, mantenimiento predictivo o monitorización industrial. Esto permitiría analizar si las conclusiones obtenidas se mantienen en escenarios con características, distribuciones y tipos de anomalía diferentes.

En relación con el preprocesado de datos, podrían estudiarse estrategias más avanzadas de codificación de características que conserven mejor la información de variables continuas. También podría analizarse el impacto de estas decisiones sobre la interpretabilidad de las probabilidades marginales, condicionales y conjuntas.

Otro aspecto a considerar sería la investigación de métodos para optimizar el orden de las características en la cadena MPS. Un orden más adecuado podría facilitar la captura de correlaciones relevantes con menores dimensiones de enlace, reduciendo el coste computacional y mejorando la interpretabilidad de medidas como la entropía de enlace.

Desde el punto de vista de la explicabilidad, se podría profundizar en el análisis de anomalías dominadas por correlaciones, desarrollando métodos que permitan representar de forma más clara qué dependencias entre variables contribuyen a una determinada alerta.

Por último, sería conveniente evaluar la estabilidad, fidelidad y utilidad práctica de las explicaciones generadas por cada enfoque. Para ello, podrían diseñarse experimentos que midan la sensibilidad de las explicaciones ante pequeñas perturbaciones de entrada o del modelo, así como su coherencia entre muestras similares.

Capítulo 6 Bibliografía

[1] S. Russell y P. Norvig, *Artificial Intelligence: A Modern Approach*, 4th. Pearson, 2021.

[2] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2006.

[3] I. Goodfellow, Y. Bengio y A. Courville, *Deep Learning*. MIT Press, 2016.

[4] Y. LeCun, L. Bottou, Y. Bengio y P. Haffner, «Gradient-Based Learning Applied to Document Recognition», *Proceedings of the IEEE*, vol. 86, n.º 11, págs. 2278-2324, 1998. DOI: 10.1109/5.726791.

[5] S. Hochreiter y J. Schmidhuber, «Long Short-Term Memory», *Neural Computation*, vol. 9, n.º 8, págs. 1735-1780, 1997. DOI: 10.1162/neco.1997.9.8.1735.

[6] A. Vaswani et al., «Attention Is All You Need», en *Advances in Neural Information Processing Systems*, vol. 30, 2017, págs. 5998-6008.

[7] A. Barredo Arrieta et al., «Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI», *Information Fusion*, vol. 58, págs. 82-115, 2020. DOI: 10.1016/j.inffus.2019.12.012.

[8] M. T. Ribeiro, S. Singh y C. Guestrin, «"Why Should I Trust You?": Explaining the Predictions of Any Classifier», en *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, págs. 1135-1144. DOI: 10.1145/2939672.2939778.

[9] S. M. Lundberg y S.-I. Lee, «A Unified Approach to Interpreting Model Predictions», en *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[10] J. R. Quinlan, «Induction of Decision Trees», *Machine Learning*, vol. 1, n.º 1, págs. 81-106, 1986. DOI: 10.1007/BF00116251.

[11] R. Orús, «A Practical Introduction to Tensor Networks: Matrix Product States and Projected Entangled Pair States», *Annals of Physics*, vol. 349, págs. 117-158, 2014. DOI: 10.1016/j.aop.2014.06.013.

[12] S.-J. Ran y G. Su, «Tensor Networks for Interpretable and Efficient Quantum-Inspired Machine Learning», *Intelligent Computing*, vol. 2, pág. 0061, 2023. DOI: 10.34133/icomputing.0061.

[13] Z.-Y. Han, J. Wang, H. Fan, L. Wang y P. Zhang, «Unsupervised Generative Modeling Using Matrix Product States», *Physical Review X*, vol. 8, n.º 3, pág. 031 012, 2018. DOI: 10.1103/PhysRevX.8.031012. arXiv: 1709.01662 [cond-mat.stat-mech].

[14] V. Chandola, A. Banerjee y V. Kumar, «Anomaly Detection: A Survey», *ACM Computing Surveys*, vol. 41, n.º 3, 2009. DOI: 10.1145/1541880.1541882.

[15] B. Aizpurua, S. Palmer y R. Orús, «Tensor Networks for Explainable Machine Learning in Cybersecurity», *Neurocomputing*, vol. 639, pág. 130 211, 2025. DOI: 10.1016/j.neucom.2025.130211.

[16] P. García-Teodoro, J. Díaz-Verdejo, G. Maciá-Fernández y E. Vázquez, «Anomaly-based network intrusion detection: Techniques, systems and challenges», *Computers & Security*, vol. 28, n.º 1–2, págs. 18-28, 2009. DOI: 10.1016/j.cose.2008.08.003.

[17] J. Adebayo, J. Gilmer, M. Muelly, I. Goodfellow, M. Hardt y B. Kim, «Sanity Checks for Saliency Maps», en *Advances in Neural Information Processing Systems*, vol. 31, 2018. arXiv: 1810.03292.

[18] C. Rudin, «Stop Explaining Black Box Machine Learning Models for High Stakes Decisions and Use Interpretable Models Instead», *Nature Machine Intelligence*, vol. 1, n.º 5, págs. 206-215, 2019. DOI: 10.1038/s42256-019-0048-x.

[19] C. Rudin, C. Chen, Z. Chen, H. Huang, L. Semenova y C. Zhong, «Interpretable Machine Learning: Fundamental Principles and 10 Grand Challenges», *Statistics Surveys*, vol. 16, págs. 1-85, 2022. DOI: 10.1214/21-SS133.

[20] J. Eisert, M. Cramer y M. B. Plenio, «Colloquium: Area laws for the entanglement entropy», *Reviews of Modern Physics*, vol. 82, n.º 1, págs. 277-306, 2010. DOI: 10.1103/RevModPhys.82.277.

[21] U. Schollwöck, «The Density-Matrix Renormalization Group in the Age of Matrix Product States», *Annals of Physics*, vol. 326, n.º 1, págs. 96-192, 2011. DOI: 10.1016/j.aop.2010.09.012.

[22] E. M. Stoudenmire y D. J. Schwab, «Supervised Learning with Tensor Networks», en *Advances in Neural Information Processing Systems 29 (NIPS 2016)*, 2016, págs. 4799-4807.

[23] M. Born, «Zur Quantenmechanik der Stoßvorgänge», *Zeitschrift für Physik*, vol. 37, n.º 12, págs. 863-867, 1926. DOI: 10.1007/BF01397477.

[24] D. H. Ackley, G. E. Hinton y T. J. Sejnowski, «A Learning Algorithm for Boltzmann Machines», *Cognitive Science*, vol. 9, n.º 1, págs. 147-169, 1985. DOI: 10.1207/s15516709cog0901_7.

[25] S. Cheng, L. Wang, T. Xiang y P. Zhang, «Tree Tensor Networks for Generative Modeling», *Physical Review B*, vol. 99, n.º 15, pág. 155 131, 2019. DOI: 10.1103/PhysRevB.99.155131.

[26] I. Glasser, R. Sweke, N. Pancotti, J. Eisert y J. I. Cirac, «Expressive Power of Tensor-Network Factorizations for Probabilistic Modeling», en *Advances in Neural Information Processing Systems*, vol. 32, 2019, págs. 1496-1508.

[27] S. S. Khan y M. G. Madden, «One-class classification: taxonomy of study and review of techniques», *The Knowledge Engineering Review*, vol. 29, n.º 3, págs. 345-374, 2014. DOI: 10.1017/S026988891300043X.

[28] O. Chapelle, B. Schölkopf y A. Zien, eds., *Semi-Supervised Learning*. Cambridge, MA: MIT Press, 2006.

[29] J. Wang, C. Roberts, G. Vidal y S. Leichenauer, «Anomaly Detection with Tensor Networks», *arXiv preprint arXiv:2006.02516*, 2020. arXiv: 2006.02516 [cs.LG].

[30] E. Puljak, M. Pierini y A. Garcia-Saez, «Tensor Network for Anomaly Detection in the Latent Space of Proton Collision Events at the LHC», *Machine Learning: Science and Technology*, vol. 6, n.º 4, pág. 045 001, 2025. DOI: 10.1088/2632-2153/ae0243. arXiv: 2506.00102 [hep-ph]. dirección: https://doi.org/10.1088/2632-2153/ae0243.

[31] M. A. Nielsen e I. L. Chuang, *Quantum Computation and Quantum Information*, 10th Anniversary Edition. Cambridge University Press, 2010.

[32] T. M. Cover y J. A. Thomas, *Elements of Information Theory*, 2nd. Wiley-Interscience, 2006.

[33] H. Hohenfeld, M. Beuerle y E. Mounzer, «Explaining Anomalies with Tensor Networks», en *2025 IEEE International Conference on Quantum Artificial Intelligence (QAI)*, 2025, págs. 165-170. DOI: 10.1109/QAI63978.2025.00033. arXiv: 2505.03911 [cs.LG]. dirección: https://doi.org/10.1109/QAI63978.2025.00033.

[34] M. Tavallaee, E. Bagheri, W. Lu y A. A. Ghorbani, «A Detailed Analysis of the KDD CUP 99 Data Set», en *IEEE Symposium on Computational Intelligence for Security and Defense Applications (CISDA)*, 2009, págs. 1-6. DOI: 10.1109/CISDA.2009.5356528.

[35] K. Kreutz-Delgado, «The Complex Gradient Operator and the CR-Calculus», University of California, San Diego, inf. téc., 2009, arXiv:0906.4835. arXiv: 0906.4835.

[36] S. Stolfo, W. Fan, W. Lee, A. Prodromidis y P. Chan, *KDD Cup 1999 Data*, UCI Machine Learning Repository, 1999. DOI: 10.24432/C51C7N. dirección: https://doi.org/10.24432/C51C7N.

[37] M. Ali, *On the Ordering of Sites in the Density Matrix Renormalization Group Using Quantum Mutual Information*, arXiv preprint arXiv:2103.01111, 2021. DOI: 10.48550/arXiv.2103.01111. arXiv: 2103.01111 [quant-ph]. dirección: https://arxiv.org/abs/2103.01111.

[38] T. Saito y M. Rehmsmeier, «The Precision-Recall Plot Is More Informative than the ROC Plot When Evaluating Binary Classifiers on Imbalanced Datasets», *PLOS ONE*, vol. 10, n.º 3, e0118432, 2015. DOI: 10.1371/journal.pone.0118432.

[39] J. Ahumada Ortiz, *Tensor Networks for XAI: detección explicable de anomalías mediante* Matrix Product States, Repositorio de código, Último acceso: junio de 2026, 2026. dirección: https://github.com/javierahumada4/Tensor-Networks-for-XAI.