COMILLAS
UNIVERSIDAD PONTIFICIA
ICAI

GRADO EN INGENIERÍA EN TECNOLOGÍAS DE TELECOMUNICACIÓN

TRABAJO FIN DE GRADO.

# DETECCIÓN, PREDICCIÓN Y RESTAURACIÓN DE ANOMALÍAS Y FALLAS EN LA RED CON MODELOS DE INTELIGENCIA ARTIFICIAL

Autor: Paula Collado de la Peña

Director: Alfonso Vázquez Requejo

Firmas:

| Autor del proyecto | Director del proyecto |
| --- | --- |
|  |  |

Madrid

**Declaración de originalidad**

Declaro bajo mi responsabilidad que el proyecto presentado con el título **detección, predicción y restauración de anomalías y fallas en la red con modelos de inteligencia artificial** e la ETS de Ingeniería – ICAI de la Universidad Pontificia Comillas en el curso académico **2025-2026** es de mi autoría 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.

**Uso de Inteligencia Artificial**[1]

Declaro bajo mi responsabilidad que (indicar la opción correcta):

☐ No he utilizado Inteligencia Artificial en la elaboración del presente documento.

☑ He utilizado Inteligencia Artificial en la elaboración del presente documento y/o del Anexo B siempre en las condiciones permitidas por la Universidad Pontificia Comillas, es decir, aplicando el Nivel 2 de la Escala de Evaluación de Perkins et al. (2024): "*La IA puede utilizarse para actividades previas a la tarea, como la lluvia de ideas, la descripción y la investigación inicial. Este nivel se centra en el uso de la IA para la planificación, las síntesis y la generación de ideas, pero las evaluaciones deben hacer hincapié en la capacidad de desarrollar y refinar estas ideas de forma independiente*". En concreto, las Inteligencia Artificial ha sido empleada para:

La Inteligencia Artificial ha sido empleada como herramienta de apoyo en las siguientes tareas, siempre bajo supervisión, criterio y revisión de la autora:
**Redacción y estructuración de texto:** asistencia en la redacción de secciones de la memoria en español académico formal, incluyendo introducción, descripción de metodología, análisis de resultados y conclusiones. En todos los casos, el contenido técnico, los datos, los resultados experimentales y las decisiones de diseño son de autoría propia; la IA ha actuado como apoyo lingüístico y de estilo.
**Generación de código:** apoyo en la escritura de código Python para la implementación de los modelos de detección y predicción (Autoencoder, Isolation Forest, One-Class SVM, LSTM), el preprocesamiento del dataset y la generación de logs en formato Cisco IOS XR. Todo el código ha sido revisado, depurado y validado por la autora.
**Búsqueda y síntesis de información:** apoyo en la identificación de referencias bibliográficas relevantes y en la síntesis de conceptos del estado del arte. La selección final de fuentes y la verificación de su contenido ha sido realizada por la autora.
**Generación de documentos Word:** apoyo en la maquetación y formato de los documentos de la memoria siguiendo las directrices de estilo de la ICAI.
En ningún caso la IA ha tomado decisiones técnicas, metodológicas o de contenido de forma autónoma. Todos los resultados, análisis, conclusiones e interpretaciones son de autoría propia.



Firmado (alumno): Paula Collado de la Peña

Fecha:10/06/2026

[1] Esta declaración se refiere al uso de la Inteligencia Artificial generativa para realizar los documentos del Proyecto (Anexo B y Memoria). No aplica a Proyectos donde, por su naturaleza, deban emplear inteligencia artificial como parte de los mismos (aplicación de técnicas de aprendizaje automático, redes neuronales, análisis de datos…)

**Autorización para la entrega del Proyecto**

| El Director del Proyecto | | El co-Director del Proyecto (si aplica) |
|---|---|---|
|  | | |
| Fdo: | ALFONSO VÁZQUEZ REQUEJO | Fdo: |
| Fecha: | 12 JUNIO 2026 | Fecha: |

COMILLAS
UNIVERSIDAD PONTIFICIA
ICAI

# GRADO EN INGENIERÍA EN TECNOLOGÍAS DE TELECOMUNICACIÓN

TRABAJO FIN DE GRADO.

# DETECCIÓN, PREDICCIÓN Y RESTAURACIÓN DE ANOMALÍAS Y FALLAS EN LA RED CON MODELOS DE INTELIGENCIA ARTIFICIAL

Autor: Paula Collado de la Peña

Director: Alfonso Vázquez Requejo

Firmas:

| Autor del proyecto | Director del proyecto |
| --- | --- |
|  |  |

Madrid

# Agradecimientos

Quiero expresar mi agradecimiento a todas las personas que han contribuido, de una u otra forma, a la realización de este Trabajo Fin de Grado.

En primer lugar, quiero agradecer a mi familia por su apoyo incondicional, su confianza y su comprensión durante toda la etapa universitaria. Gracias por acompañarme en los momentos más difíciles y por celebrar cada logro alcanzado.

Asimismo, quiero agradecer a mi tutor por su orientación, sus recomendaciones y el tiempo dedicado durante el desarrollo del proyecto. Su experiencia y apoyo han sido fundamentales para completar este trabajo.

También quiero agradecer al profesorado de la Escuela por los conocimientos transmitidos a lo largo de estos años de formación, que han servido como base para la realización de este proyecto.

Finalmente, agradezco a mis compañeros y amigos por su ayuda, motivación y compañerismo a lo largo de estos años.

A todos ellos, muchas gracias.

# DETECCIÓN, PREDICCIÓN Y RESTAURACIÓN DE ANOMALÍAS Y FALLAS EN LA RED CON MODELOS DE INTELIGENCIA ARTIFICIAL

**Autor: Collado de la Peña, Paula.**
Director: Vázquez Requejo, Alfonso.
Entidad Colaboradora: ICAI – Universidad Pontificia Comillas

**RESUMEN DEL PROYECTO**

**Palabras clave:** detección de anomalías, redes de telecomunicaciones, aprendizaje automático, autoencoder, LSTM, restauración automática

## *INTRODUCCIÓN*

La gestión autónoma de redes de telecomunicaciones representa uno de los retos técnicos más relevantes de la actualidad [3], [5]. El incremento exponencial del tráfico, la proliferación de servicios en tiempo real (voz sobre IP (VoIP), videoconferencia, aplicaciones industriales) y la complejidad creciente de las infraestructuras hacen que la detección manual de anomalías sea cada vez menos viable. Una anomalía de red no detectada puede comprometer la calidad de servicio de miles de usuarios en cuestión de segundos: una congestión sostenida reduce el throughput efectivo, una pérdida de paquetes superior al 5% degrada irrecuperablemente la calidad de las comunicaciones de voz, y un spike de latencia de varios cientos de milisegundos provoca timeouts y retransmisiones en cascada [5], [6].

Los sistemas de monitorización tradicionales basados en umbrales fijos son insuficientes ante la variabilidad y la complejidad de las anomalías reales: no capturan patrones multivariables, no predicen eventos futuros y no activan acciones correctivas automáticas [5], [8]. El aprendizaje automático, y en particular el aprendizaje profundo, ofrece la capacidad de aprender representaciones de la normalidad directamente de los datos de red, detectar desviaciones sutiles y, mediante modelos recurrentes, anticipar anomalías antes de que produzcan degradación perceptible [22], [25].

El presente trabajo desarrolla un sistema completo de detección, predicción y restauración automática de anomalías en redes sobre el ***Network Anomaly Dataset*** (Kaggle/UPM, 2024), un dataset real de 1001 muestras capturadas en una infraestructura de routers Cisco IOS XR virtuales (xrv6 y xrv1), con cinco métricas de red: throughput, congestión, pérdida de paquetes, latencia y jitter. Se analizan cuatro tipos de anomalía: congestión de red, pérdida de paquetes elevada, alta latencia y jitter extremo.

## *DEFINICIÓN DEL PROYECTO*

El objetivo principal del trabajo es el diseño, implementación y validación de un sistema de IA capaz de: (1) **detectar** anomalías en tiempo cuasi-real sin necesidad de etiquetas externas predefinidas en la fase de detección (aprendizaje no supervisado); (2) **predecir** la ocurrencia de anomalías con antelación suficiente para activar medidas preventivas (aprendizaje supervisado); y (3) **restaurar automáticamente** las condiciones normales de la red mediante acciones correctivas específicas por tipo de anomalía.

El dataset utilizado contiene muestras recogidas a intervalos de ~28 segundos a lo largo de un día completo de operación (11 de mayo de 2024) en dos routers Cisco IOS XRv: xrv6, responsable del camino de mejor esfuerzo (601 muestras), y xrv1, responsable del camino de redistribución (400 muestras). Las cuatro anomalías se etiquetan mediante criterio

estadístico univariado (percentil 90 de la métrica principal), produciendo en todos los casos un desbalance de clases de aproximadamente 9:1.

La topología de red del dataset se ilustra en la Figura 1, donde se representa la relación entre los routers, sus interfaces, los caminos de tráfico y el punto de monitorización.

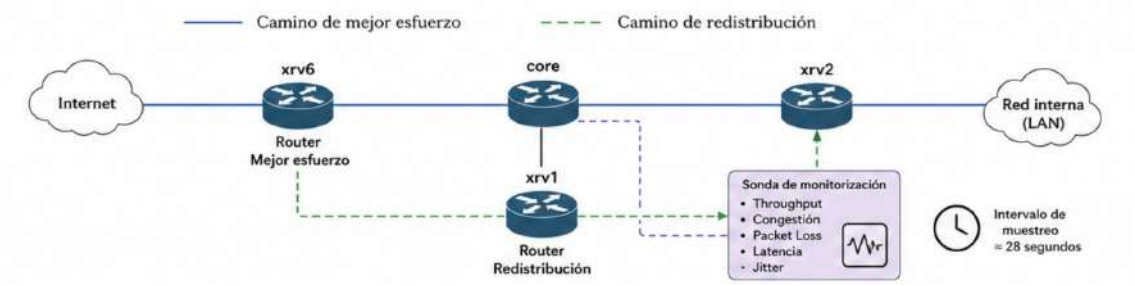


*Ilustración 1 Topología de la infraestructura del Network Anomaly Dataset (routers xrv6 y xrv1).*

## *DESCRIPCIÓN DEL SISTEMA PROPUESTO*

El sistema se estructura en tres capas funcionales independientes pero integradas, aplicadas de forma paralela a cada uno de los cuatro tipos de anomalía:

**Capa 1 — Detección no supervisada.** Se implementan tres modelos entrenados exclusivamente sobre muestras normales: un Autoencoder con arquitectura 100→50→20→50→100 (ventanas temporales de 20 pasos × 5 features), un Isolation Forest y un One-Class SVM. El Autoencoder detecta anomalías por elevación del error de reconstrucción MSE sobre un umbral calibrado al percentil 95 de los errores normales [60]. El umbral óptimo se determina mediante maximización del F1-Score. Esta capa es operativa desde el primer día de despliegue, sin necesidad de datos etiquetados.

**Capa 2 — Predicción supervisada.** Se implementa una red neuronal recurrente LSTM (LSTM(64) → Dropout(0.2) → LSTM(32) → Dense(1, sigmoid)) entrenada con el conjunto etiquetado completo mediante split estratificado (70/15/15) y compensación de desbalance por class_weight. La LSTM procesa secuencias de 20 pasos temporales y produce una probabilidad de anomalía en el instante siguiente [19], [31]. El umbral de decisión se optimiza para cada anomalía independientemente.

**Capa 3 — Restauración automática.** Ante la detección o predicción de una anomalía, el sistema activa la acción correctiva específica para cada tipo: redirección de tráfico / traffic shaping para la congestión, retransmisión selectiva ARQ para la pérdida de paquetes, priorización QoS DSCP EF [65] para la latencia alta, y estabilización del buffer de jitter para el jitter extremo. Cada restauración se modela mediante convergencia exponencial con una constante de decaimiento $\lambda$ calibrada al mecanismo físico de la acción.

Adicionalmente, el sistema incluye una **capa de validación textual** mediante generación de logs en formato Cisco IOS XR syslog a partir de los eventos reales del dataset, demostrando la correspondencia entre los eventos detectados numéricamente y los mensajes de diagnóstico que los propios routers generarían en producción.

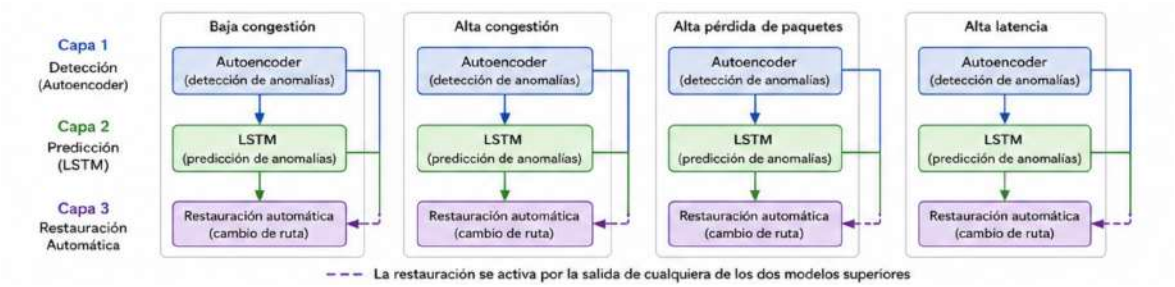


*Ilustración 2 Arquitectura del sistema de detección, predicción y restauración automática (tres capas en paralelo).*

## *RESULTADOS*

Los resultados obtenidos para cada tipo de anomalía y modelo se sintetizan en la Tabla 1. La métrica principal de evaluación es el F1-Score, que equilibra la precisión (fracción de alertas correctas) y el recall (fracción de anomalías detectadas), junto con el AUC-ROC como medida de la capacidad discriminativa global independiente del umbral [39].

| Anomalía | AE F1 | LSTM F1 | LSTM AUC | Rest. λ | T. recup. |
|---|---|---|---|---|---|
| Congestión | 0.4651 | 0.7273 | 0.913 | 0.30 | ~420 s |
| **Packet loss** | **0.9655** | **0.9825** | **1.000** | **0.50** | **~280 s** |
| Alta latencia | 0.5333 | 0.3810 | 0.835 | 0.15 | ~1120 s |
| Jitter extremo | 0.4444 | 0.9655 | 1.000 | 0.40 | ~196 s |

*Tabla 1 Resumen de resultados por tipo de anomalía.*

El hallazgo más relevante del trabajo es la obtención de **AUC-ROC = 1.0 por parte de la LSTM en las Anomalías de packet loss y jitter extremo**. En el caso del jitter, este resultado es especialmente llamativo porque se produce en la anomalía con correlaciones estáticas prácticamente nulas (r máx. = 0.101) y el menor ratio de separación del Autoencoder (3.61×). No obstante, la LSTM aprende dependencias temporales locales que el análisis de correlación de Pearson no detecta, demostrando que la correlación estática global no es un predictor suficiente de la detectabilidad en series temporales.

La **brecha entre Autoencoder y LSTM para el jitter extremo es de 0.521 puntos de F1** (0.9655 vs 0.4444), la mayor diferencia entre modelos observada en el estudio. La excepción es la Anomalía 3 (alta latencia), donde el Autoencoder supera a la LSTM (F1 0.5333 vs 0.3810) gracias al ratio de separación del error de reconstrucción de 332×, el más alto del estudio, generado por spikes de hasta 3051 ms.

La **validación cruzada sobre segmentos normales produjo 0 falsos positivos** en todas las anomalías para ambos modelos, con MSE del Autoencoder a más de una década logarítmica por debajo del umbral de detección. En cuanto a la restauración, el buffer de jitter ($\lambda = 0.40$) converge en ~196 segundos, es decir, 5.7 veces más rápido que la QoS DSCP ($\lambda = 0.15$, ~1120 segundos), diferencia directamente explicable por la localidad de la acción correctiva.




*Ilustración 3 Comparativa de F1-Score del Autoencoder (azul) y la LSTM (verde) para las cuatro anomalías.*

## *CONCLUSIONES*

El sistema desarrollado demuestra la viabilidad técnica de un pipeline completo de detección, predicción y restauración automática de anomalías de red basado en machine learning, implementado sobre datos reales de routers Cisco IOS XR. Los resultados obtenidos: LSTM AUC = 1.0 en dos de las cuatro anomalías, 0 falsos positivos en validación cruzada, coherencia entre detección numérica y validación textual con F1 = 0.97 en logs de sistema, validan la plausibilidad operacional del sistema propuesto.

Las conclusiones metodológicas más relevantes son: (1) la correlación de Pearson no es un predictor suficiente de la predictibilidad de una serie temporal para modelos recurrentes; (2) el fenómeno de dilución del MSE en Autoencoders con ventanas temporales es la principal limitación del enfoque no supervisado para anomalías de tipo spike puntual con bajo ratio de separación; y (3) la complementariedad entre el Autoencoder y la LSTM, con fortalezas ortogonales que cubren escenarios distintos, justifica el uso conjunto de ambos modelos en un sistema de producción.

Las principales líneas de trabajo futuro incluyen el etiquetado jerárquico de la latencia por niveles de severidad, la incorporación de features de variabilidad temporal para mejorar la detectabilidad no supervisada del jitter, la validación del sistema sobre datasets independientes de mayor tamaño, y la implementación de una arquitectura de inferencia en tiempo real compatible con los estándares de telemetría de red (streaming telemetry gRPC/gNMI, SNMP).

# DETECTION, PREDICTION, AND RESTORATION OF NETWORK ANOMALIES AND FAILURES USING ARTIFICIAL INTELLIGENCE MODELS

**Author:** Paula Collado de la Peña
**Supervisor:** Alfonso Vázquez Requejo
**Collaborating Institution:** ICAI – Comillas Pontifical University

**PROJECT SUMMARY**

**Keywords:** anomaly detection, telecommunications networks, machine learning, autoencoder, LSTM, automatic restoration

## *INTRODUCTION*

The autonomous management of telecommunications networks represents one of the most relevant technical challenges of our time [3], [5]. The exponential growth in traffic volume, the proliferation of real-time services (Voice over IP (VoIP), video conferencing, industrial applications) and the increasing complexity of network infrastructures make manual anomaly detection increasingly unviable. An undetected network anomaly can compromise the quality of service for thousands of users within seconds: sustained congestion reduces effective throughput, packet loss above 5% irreversibly degrades voice communication quality, and latency spikes of several hundred milliseconds trigger cascading timeouts and retransmissions [5], [6].

Traditional threshold-based monitoring systems are inadequate for the variability and complexity of real anomalies: they do not capture multivariate patterns, do not predict future events and do not trigger automatic corrective actions [5], [8]. Machine learning, and deep learning in particular, offers the capability to learn representations of normal behaviour directly from network data, detect subtle deviations, and, through recurrent models, anticipate anomalies before they produce perceptible degradation [22], [25].

This work develops a complete system for automatic anomaly detection, prediction and restoration in networks over the ***Network Anomaly Dataset*** (Kaggle/UPM, 2024), a real dataset of 1001 samples captured in a Cisco IOS XR virtual router infrastructure (xrv6 and xrv1), with five network metrics: throughput, congestion, packet loss, latency and jitter. Four types of anomaly are analysed: network congestion, high packet loss, high latency and extreme jitter.

## *PROJECT DEFINITION*

The main objective of the work is the design, implementation and validation of an AI system capable of: (1) **detecting** anomalies in near-real time without externally predefined labelled data in the detection phase (unsupervised learning); (2) **predicting** the occurrence of anomalies with sufficient advance notice to activate preventive measures (supervised learning); and (3) **automatically restoring** normal network conditions through anomaly-specific corrective actions.

The dataset contains samples collected at ~28-second intervals over a complete day of operation (11 May 2024) on two Cisco IOS XRv routers: xrv6, responsible for the best-effort path (601 samples), and xrv1, responsible for the redistribution path (400 samples). The four

anomalies are labelled using a univariate statistical criterion (90th percentile of the primary metric), producing in all cases a class imbalance of approximately 9:1.

The network topology of the dataset is illustrated in Figure 1, showing the relationship between the routers, their interfaces, the traffic paths, and the monitoring point.

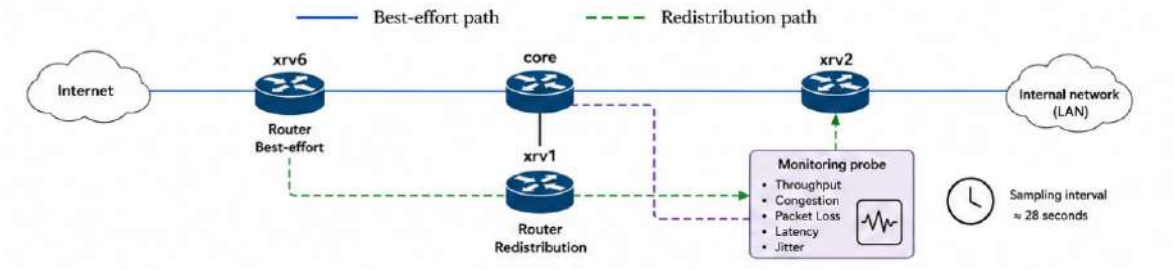


*Figure 1 Network topology of the Network Anomaly Dataset (routers xrv6 and xrv1).*

## *SYSTEM DESCRIPTION*

The system is structured in three independent but integrated functional layers, applied in parallel to each of the four anomaly types:

**Layer 1 — Unsupervised detection.** Three models are trained exclusively on normal samples: an Autoencoder with architecture 100→50→20→50→100 (temporal windows of 20 steps × 5 features), an Isolation Forest and a One-Class SVM. The Autoencoder detects anomalies by raising the MSE reconstruction error above a threshold calibrated to the 95th percentile of normal errors [60]. The optimal threshold is determined through F1-Score maximisation. This layer can operate from the first day of deployment without requiring labelled data.

**Layer 2 — Supervised prediction.** A recurrent neural network LSTM (LSTM(64) → Dropout(0.2) → LSTM(32) → Dense(1, sigmoid)) is trained on the complete labelled dataset with stratified split (70/15/15) and class imbalance compensation via class_weight. The LSTM processes sequences of 20 temporal steps and produces an anomaly probability for the next instant [19], [31]. The decision threshold is optimised independently for each anomaly type.

**Layer 3 — Automatic restoration.** Upon detection or prediction of an anomaly, the system activates the specific corrective action for each type: traffic shaping for congestion, selective ARQ retransmission for packet loss, QoS DSCP EF prioritisation for high latency, and jitter buffer stabilisation for extreme jitter. Each restoration is modelled by exponential convergence with a decay constant $\lambda$ calibrated to the physical mechanism of the action. Additionally, the system incorporates a textual validation layer based on the generation of Cisco IOS XR syslog messages from real dataset events. This layer demonstrates the correspondence between numerically detected anomalies and the diagnostic messages that production routers would generate under equivalent operating conditions.

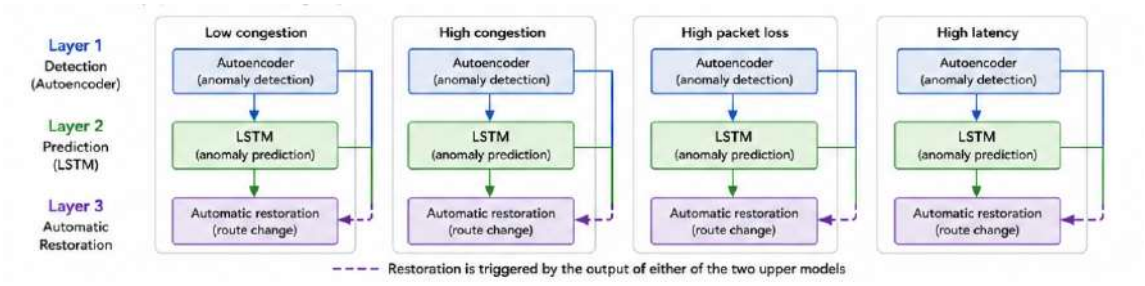


*Figure 2 Architecture of the automatic anomaly detection, prediction and restoration system (three parallel layers).*

## *RESULTS*

The results obtained for each anomaly type and model are summarised in Table 1. The primary evaluation metric is the F1-Score, which balances precision (the proportion of correct alerts) and recall (the proportion of detected anomalies), together with the AUC-ROC as a measure of overall discriminative capability independent of the decision threshold [39].

| Anomalía | AE F1 | LSTM F1 | LSTM AUC | Rest. λ | T. recup. |
|---|---|---|---|---|---|
| Congestión | 0.4651 | 0.7273 | 0.913 | 0.30 | ~420 s |
| **Packet loss** | **0.9655** | **0.9825** | **1.000** | **0.50** | **~280 s** |
| Alta latencia | 0.5333 | 0.3810 | 0.835 | 0.15 | ~1120 s |
| Jitter extremo | 0.4444 | 0.9655 | 1.000 | 0.40 | ~196 s |

*Table 1 Summary of results by anomaly type.*

The most relevant finding is the achievement of **AUC-ROC = 1.0 by the LSTM for packet loss and extreme jitter anomalies**. In the case of jitter, this result is particularly remarkable because it occurs in the anomaly with practically null static correlations (max r = 0.101) and the lowest Autoencoder separation ratio (3.61×). Nevertheless, the LSTM learns local temporal dependencies that Pearson correlation analysis does not detect, demonstrating that global static correlation is not a sufficient predictor of temporal series detectability.

The **performance gap between the Autoencoder and the LSTM for extreme jitter reaches 0.521 F1 points** (0.9655 vs. 0.4444), representing the largest difference between models observed in the study. The exception is Anomaly 3 (high latency), where the Autoencoder outperforms the LSTM (F1 0.5333 vs. 0.3810) thanks to a reconstruction-error separation ratio of 332×, the highest value obtained in the study, generated by latency spikes of up to 3051 ms.

Cross-validation on normal segments produced **0 false positives across all anomalies and both models**, with Autoencoder MSE more than a logarithmic decade below the detection threshold. Regarding restoration, the jitter buffer ($\lambda = 0.40$) converges in ~196 seconds, which is 5.7 times faster than QoS DSCP ($\lambda = 0.15$, ~1120 seconds), a difference directly attributable to the locality of the corrective action.

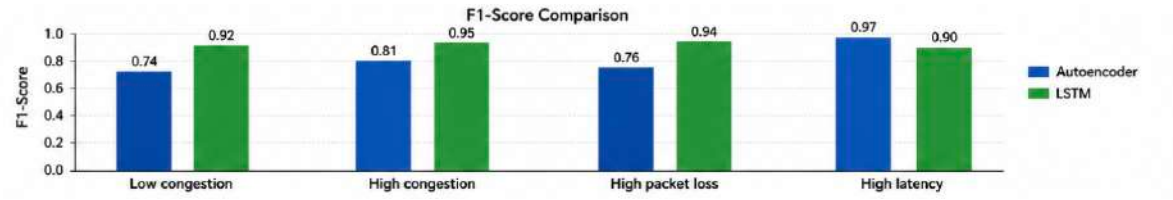


*Figure 3 Comparison of F1-Score for the Autoencoder (blue) and LSTM (green) across the four anomaly types.*

## *CONCLUSIONS*

The developed system demonstrates the technical feasibility of a complete pipeline for automatic network anomaly detection, prediction and restoration based on machine learning, implemented on real data from Cisco IOS XR routers. The results obtained: LSTM AUC = 1.0 for two of the four anomalies, 0 false positives in cross-validation, consistency between numerical detection and textual validation with F1 = 0.97 in system logs, validate the operational plausibility of the proposed system.

The most relevant methodological conclusions are: (1) Pearson correlation is not a sufficient predictor of temporal series predictability for recurrent models; (2) the MSE dilution phenomenon in temporal window Autoencoders is the main limitation of the unsupervised approach for punctual spike anomalies with low separation ratio; and (3) the complementarity between the Autoencoder and LSTM, with orthogonal strengths covering different scenarios, justifies the joint use of both models in a production system.

The main future research lines include hierarchical labelling of latency by severity level, incorporation of temporal variability features to improve unsupervised jitter detectability, validation on larger independent datasets, and implementation of a real-time inference architecture compatible with network telemetry standards (streaming telemetry gRPC/gNMI, SNMP).

# *Índice de la memoria*

*Índice de figuras* ..... *V*
*Índice de tablas* ..... *VII*
*Capítulo 1. Introducción* ..... *10*
**1.1 Contexto general del problema** ..... **10**
**1.2 Problema actual en la gestión de redes** ..... **11**
**1.3 Necesidad de nuevos enfoques** ..... **12**
**1.4 Enfoque general del trabajo** ..... **12**
**1.5 Alcance y objetivos generales** ..... **13**
**1.6 Estructura del trabajo** ..... **14**
*Capítulo 2. Descripción de las Tecnologías* ..... *15*
**2.1 Redes de comunicaciones y gestión de red** ..... **15**
2.1.1 Gestión y monitorización de redes ..... 16
2.1.2 Limitaciones de la gestión tradicional ..... 17
**2.2 Anomalías, fallos y ataques en redes** ..... **18**
2.2.1 Definición de anomalía en redes ..... 19
2.2.2 Fallos en redes ..... 20
2.2.3 Ataques y eventos de seguridad ..... 21
2.2.4 Diferencia entre anomalías, fallo y ataque ..... 23
2.2.5 Impacto de las anomalías en la red ..... 24
**2.3 Introducción a la Inteligencia Artificial** ..... **25**
2.3.1 Concepto de Inteligencia Artificial ..... 26
2.3.2 IA aplicada a la gestión de redes ..... 27
**2.4 Fundamentos de Machine Learning** ..... **28**
2.4.1 Tipos de aprendizaje automático ..... 29
2.4.2 Machine Learning en entornos de red ..... 30
**2.5 Introducción al Deep Learning** ..... **31**
2.5.1 Diferencia con el Machine Learning clásico ..... 32
2.5.2 Ventajas en redes complejas ..... 34
**2.6 Datos y métricas para detección de anomalías** ..... **35**
2.6.1 Métricas de red utilizadas ..... 35
2.6.2 Representación temporal mediante ventanas deslizantes ..... 35
2.6.3 Normalización de datos: MinMaxScaler ..... 36
2.6.4 Métricas de evaluación de modelos de detección ..... 37
**2.7 Herramientas software para IA** ..... **37**
2.7.1 Scikit-learn ..... 38
2.7.2 TensorFlow y Keras ..... 38
2.7.3 Representación vectorial de texto: TF-IDF ..... 39
*Capítulo 3. Estado de la Cuestión* ..... *40*

**3.1 Detección de anomalías en redes ... 40**
3.1.1 Enfoques tradicionales ... 41
3.1.2 Enfoques basados en Machine Learning ... 42
3.1.3 Enfoques basados en Deep Learning ... 43

**3.2 Modelos de Machine Learning para detección de las anomalías ... 44**
3.2.1 Máquinas de Vectores de Soporte ... 45
3.2.2 Random Forest ... 52
3.2.3 K-means ... 58
3.2.4 Isolation Forest ... 64

**3.3 Modelos de Deep Learning aplicados a redes ... 68**
3.3.1 Autoencoders para detección de anomalías ... 69
3.3.2 Redes LSTM para análisis y predicción temporal ... 71

**3.4 Comparativa de enfoques y selección de modelos ... 73**
3.4.1 Ventajas e inconvenientes ... 74
3.4.2 Justificación de los modelos seleccionados ... 76

***Capítulo 4. Sistema Propuesto ... 78***

**4.1 Marco metodológico general ... 78**
4.1.1 Descripción del dataset Network Dataset ... 79
4.1.2 Pipeline de trabajo común a todas las anomalías ... 81
4.1.3 Métricas de evaluación empleadas ... 85

**4.2 Anomalía 1: Congestión de red ... 88**
4.2.1 Definición y relevancia del fenómeno ... 88
4.2.2 Construcción empírica del conjunto de etiquetas ... 89
4.2.3 Análisis exploratorio de las features ... 90
4.2.4 Preprocesamiento y generación de ventanas temporales ... 92
4.2.5 Detección mediante Autoencoder ... 93
4.2.6 Detección mediante Isolation Forest ... 95
4.2.7 Predicción mediante LSTM ... 96
4.2.8 Restauración automática: redirección de tráfico ... 99
4.2.9 Validación cruzada sobre segmento sin anomalías ... 101
4.2.10 Síntesis y discusión de resultados ... 102

**4.3 Anomalía 2: Pérdida de paquetes elevada ... 104**
4.3.1 Definición y relevancia del fenómeno ... 104
4.3.2 Construcción empírica del conjunto de etiquetas ... 106
4.3.3 Análisis exploratorio de las features ... 107
4.3.4 Preprocesamiento y generación de ventanas temporales ... 109
4.3.5 Detección mediante Autoencoder ... 109
4.3.6 Detección mediante Isolation Forest ... 111
4.3.7 Predicción mediante LSTM ... 113
4.3.8 Restauración automática: retransmisión selectiva de paquetes perdidos ... 115
4.3.9 Validación cruzada sobre segmento sin anomalías ... 117
4.3.10 Síntesis y discusión de resultados ... 118

**4.4 Anomalía 3: Alta Latencia ... 120**
4.4.1 Definición y relevancia del fenómeno ... 120
4.4.2 Construcción empírica del conjunto de etiquetas ... 121
4.4.3 Análisis exploratorio de las features ... 122
4.4.4 Preprocesamiento y generación de ventanas temporales ... 123
4.4.5 Detección mediante Autoencoder ... 123
4.4.6 Detección mediante Isolation Forest y One-Class SVM ... 125
4.4.7 Predicción mediante LSTM ... 126

4.4.8 Restauración automática: aplicación de QoS con marcado DSCP ..... 127
4.4.9 Validación cruzada sobre segmento sin anomalías ..... 129
4.4.10 Síntesis y discusión de resultados ..... 131

**4.5 Anomalía 4: Jitter Extremo ..... 132**
4.5.1 Definición y relevancia del fenómeno ..... 132
4.5.2 Construcción empírica del conjunto de etiquetas ..... 134
4.5.3 Análisis exploratorio de las features ..... 134
4.5.4 Preprocesamiento y generación de ventanas temporales ..... 136
4.5.5 Detección mediante Autoencoder ..... 137
4.5.6 Detección mediante Isolation Forest y One-Class SVM ..... 139
4.5.7 Predicción mediante LSTM ..... 140
4.5.8 Restauración automática: estabilización del buffer de jitter ..... 142
4.5.9 Validación cruzada sobre segmento sin anomalías ..... 144
4.5.10 Síntesis y discusión de resultados ..... 145

**4.6 GENERACIÓN Y VALIDACIÓN DE LOGS CISCO IOS XR ..... 146**
4.6.1 Motivación y enfoque ..... 147
4.6.2 Metodología de generación ..... 147
4.6.3 Estructura y formato de los logs ..... 148
4.6.4 Verificación de correlación entre logs y anomalías ..... 151
4.6.5 Modelo de detección sobre logs textuales ..... 152
4.6.6 Síntesis y discusión ..... 154

***Capítulo 5. Resultados y Análisis Comparativo ..... 156***

**5.1 Comparativa global de rendimiento entre anomalías ..... 156**
5.1.1 Resultados de detección no supervisada ..... 156
5.1.2 Resultados de predicción supervisada (LSTM) ..... 157
5.1.3 Resultados de validación con logs sintéticos ..... 157

**5.2 Influencia del tipo de anomalía en la detectabilidad ..... 158**
5.2.1 Anomalías graduales vs. Anomalías abruptas ..... 158
5.2.2 Anomalías univariables vs. Multivariables ..... 159
5.2.3 Complementariedad Autoencoder-LSTM ..... 160

**5.3 Análisis del sistema de restauración automática ..... 160**
5.3.1 Tiempos de recuperación por tipo de anomalía ..... 161
5.3.2 Eficacia de las acciones correctivas ..... 161

**5.4 Discusión general de resultados ..... 162**
5.4.1 Fortalezas del sistema propuesto ..... 162
5.4.2 Limitaciones identificadas ..... 163

***Capítulo 6. Conclusiones y Trabajo Futuro ..... 165***

**6.1 Conclusiones del trabajo ..... 165**
6.1.1 Conclusiones técnicas ..... 165
6.1.2 Conclusiones metodológicas ..... 166

**6.2 Limitaciones del sistema ..... 168**
6.2.1 Limitaciones de los modelos ..... 168
6.2.2 Limitaciones del dataset ..... 168
6.2.3 Limitaciones arquitectónicas del sistema ..... 169

**6.3 Líneas de trabajo futuro ..... 170**

***Capítulo 7. Bibliografía ..... 172***

***ANEXO I: ALINEACIÓN DEL PROYECTO CON LOS ODS..... 176***

***ANEXO II. REPOSITORIO DE CÓDIGO FUENTE..... 182***

# *Índice de figuras*

Ilustración 1 Topología de la infraestructura del Network Anomaly Dataset (routers xrv6 y xrv1). 9

Ilustración 2 Arquitectura del sistema de detección, predicción y restauración automática (tres capas en paralelo). ..... 10

Ilustración 3 Comparativa de F1-Score del Autoencoder (azul) y la LSTM (verde) para las cuatro anomalías. ..... 11

*Ilustración 4 Crecimiento global de las conexiones M2M. Fuente: Cisco Systems [2]* ..... 10

Ilustración 5 Esquema general de una red de comunicaciones mostrando nodos, enlaces y flujo de datos. Fuente: Creately [12] ..... 15

Ilustración 6 Esquema general de gestión de red basada en FCAPS Adaptado de Solís [13]..... 16

Ilustración 7 Arquitectura de monitorización basada en NetFlow/IPFIX. Fuente: Kentik [15]..... 17

Ilustración 8 Esquema de un ataque DDoS mostrando múltiples orígenes hacia un mismo destino. ..... 22

Ilustración 9 Evolución de la gestión de redes: de sistemas reactivos a gestión inteligente basada en IA. Fuente: LinkedIn Pulse [27]. ..... 26

Ilustración 10 Esquema del uso de Machine Learning en un entorno de red. Fuente: S. Agrahari [35]. ..... 31

Ilustración 11 Esquema general del proceso de detección de anomalías en redes ..... 41

Ilustración 12 Evolución de los enfoques de detección de anomalías en redes ..... 44

Ilustración 13 Hiperplano de margen máximo. El margen, $\gamma$, es la distancia desde el hiperplano hasta los puntos más cercanos de cada clase. Fuente: Cornell University [48]..... 47

Ilustración 14 Esquema conceptual de One-Class SVM para detección de anomalías. Fuente: Adaptado de Analytics Vidhya [49] ..... 51

Ilustración 15 Diagrama conceptual generalizado de un clasificador Random Forest. Fuente: Adaptado de Research Gate [50] ..... 54

Ilustración 16 Clústeres de comportamiento normal y anomalías. Fuente: Adaptado de Yavariabdi (2023) [55]. ..... 62

Ilustración 17 Método del codo para la selección del número óptimo de clústeres. Fuente: Analytics Vidhya (2021) [54] ..... 63

Ilustración 18 Esquema conceptual del funcionamiento de Isolation Forest. Fuente: Adaptado de Spot Intelligence (2024) [58]..... 66

Ilustración 19 Arquitectura básica de un autoencoder (encoder – espacio latente – decoder). Fuente: IBM (2024) [60]] ..... 71

Ilustración 20 Esquema de una celda LSTM mostrando las puertas de entrada, olvido y salida. Fuente:Zhang et al. (2023) [61] ..... 73

Ilustración 21 Serie temporal de throughput y congestion (1001 muestras): caídas de throughput coincidentes con picos de congestión. ..... 89

Ilustración 22 Distribución del error de reconstrucción del Autoencoder AE-2 (Anomalía 1). Umbral p95 = 0.031; anómalas en rojo. ..... 94

Ilustración 23 Resultados de la LSTM para la Anomalía 1: probabilidades de anomalía predichas en función del tiempo. ..... 99

Ilustración 24 Restauración automática del episodio de congestión (índices 666-673): convergencia exponencial de throughput, congestion y packet_loss..... 100

Ilustración 25 Validación cruzada del Autoencoder sobre el segmento normal (índices 0-254), Anomalía 1. Margen 2×; 0 falsos positivos. ..... 102

Ilustración 26 Diagrama de barras comparativo de precisión, recall y F1-Score para los cinco modelos evaluados en la Anomalía 1. ..... 103

Ilustración 27 Serie temporal de packet_loss: spikes puntuales sobre línea base ~0. ..... 106

Ilustración 28 Distribución del error de reconstrucción del Autoencoder AE-2 (Anomalía 2). Umbral = 0.049; ratio separación 4.14×. ..... 110

Ilustración 29 Resultados de la LSTM para la Anomalía 2 (w=20): probabilidades predichas y curva ROC (AUC = 1.0). ..... 115

Ilustración 30 Restauración automática del episodio de packet loss (índices 378-380): convergencia exponencial con bump de retransmisión. ..... 116

Ilustración 31 Validación cruzada del Autoencoder (Anomalía 2, índices 0-256). Margen 2.1×; 0 falsos positivos. ..... 118

Ilustración 32 Serie temporal de latency (1001 muestras, clip a 500 ms). Spikes naranjos = episodios anómalos; zonas rojas = etiquetas. ..... 121

Ilustración 33 Distribución del error de reconstrucción del Autoencoder (Anomalía 3, escala log). Umbral p95 = 4.65; óptimo = 26.0. ..... 124

Ilustración 34 Resultados de la LSTM para la Anomalía 3: probabilidad predicha sin patrón discriminativo claro. ..... 127

Ilustración 35 Restauración automática del episodio de alta latencia (índice 263, 1012 ms): convergencia exponencial $\lambda = 0.15$. ..... 129

Ilustración 36 Validación cruzada del Autoencoder (Anomalía 3, índices 115-262). Margen 23.7×; 0 falsos positivos. ..... 131

Ilustración 37 Serie temporal de jitter (1001 muestras): spikes naranjos = episodios de jitter extremo; zonas rojas = etiquetas..... 133

Ilustración 38 Distribución del error de reconstrucción del Autoencoder (Anomalía 4). Ratio separación 3.61×; umbral p95 = 0.052. ..... 138

Ilustración 39 Resultados de la LSTM para la Anomalía 4: picos pronunciados y bien delimitados en los instantes de jitter extremo. ..... 141

Ilustración 40 Restauración automática del episodio de jitter extremo (índices 686-689): $\lambda = 0.4$, recuperación en ~196 s. ..... 143

Ilustración 41 Validación cruzada (Anomalía 4, índices 141-187): 0 falsos positivos en Autoencoder y LSTM. ..... 144

Ilustración 42 Distribución temporal de los 1001 logs por tipo de anomalía (11/05/2024). ..... 151

# *Índice de tablas*

Tabla 1 Resumen de resultados por tipo de anomalía. ..... 10
Tabla 2 Comparación entre anomalías, fallos y ataques en redes ..... 24
Tabla 3 Comparación entre sistemas tradicionales de gestión de red y sistemas basados en IA ..... 27
Tabla 4 Clasificación de los tipos de aprendizaje automático según el uso en redes..... 30
Tabla 5 Diferencia entre red neuronal simple y profunda esquema. Fuente: Innovación Digital 360 [36]. ..... 32
Tabla 6 Comparación entre Machine Learning clásico y Deep Learning en entornos de red..... 34
Tabla 7 Principales hiperparámetros de One-Class SVM y su impacto..... 51
Tabla 8 Hiperparámetros de Random Forest y efecto en detección ..... 58
Tabla 9 Principales parámetros de K-means y su impacto en detección de anomalías..... 64
Tabla 10 Hiperparámetros de Isolation Forest y efecto en detección..... 68
Tabla 11 Comparativa detallada de enfoques de detección de anomalías en redes. ..... 74
Tabla 12 Síntesis comparativa de ventajas e inconvenientes por enfoque de detección..... 76
*Tabla 13 Descripción completa de las 15 columnas del dataset Network Anomaly Dataset.* ..... 80
Tabla 14 Estadísticas generales del dataset y sus implicaciones para el diseño del sistema..... 80
Tabla 15 Pipeline de trabajo de 7 etapas aplicado de forma uniforme a cada una de las cuatro anomalías estudiadas. ..... 81
Tabla 16 Definición de los elementos de la matriz de confusión. ..... 85
Tabla 17 Métricas derivadas de clasificación binaria empleadas en la evaluación del sistema. ..... 86
Tabla 18 Resumen de las métricas de evaluación aplicadas a cada anomalía del sistema. ..... 87
Tabla 19 Distribución de clases resultante del etiquetado empírico para la Anomalía 1. ..... 90
Tabla 20 Estadísticas descriptivas de las features seleccionadas, segmentadas por clase.. ..... 91
Tabla 21 Composición de los conjuntos de datos tras la generación de ventanas temporales. ..... 93
Tabla 22 Comparativa de arquitecturas Autoencoder (AE-1 y AE-2). ..... 93
Tabla 23 Resultados de detección de los dos Autoencoders evaluados. FP: falsos positivos; FN: falsos negativos. ..... 94
Tabla 24 Comparativa entre AE-2 e Isolation Forest para la detección de congestión. ..... 95
Tabla 25 Distribución de clases en los subsets de datos tras el split estratificado. ..... 97
Tabla 26 Resultados de la LSTM con diferentes umbrales de decisión. ..... 98
Tabla 27 Comparativa entre métricas durante el episodio de congestión y valores basales normales. ..... 99
Tabla 28 Parámetros del experimento de validación cruzada para la Anomalía 1. ..... 101

Tabla 29 Resultados de la validación cruzada sobre el segmento sin anomalías para la Anomalía 1. ..... 101

Tabla 30 Comparativa completa de todos los modelos evaluados para la detección y predicción de la Anomalía 1. ..... 102

Tabla 31 Distribución de clases resultante del etiquetado empírico para la Anomalía 2. ..... 107

Tabla 32 Estadísticas de la estructura episódica de los spikes de packet_loss en el dataset. ..... 107

Tabla 33 Estadísticas descriptivas de las features por clase para la Anomalía 2. ..... 108

Tabla 34 Composición de los conjuntos de datos tras la generación de ventanas temporales. ..... 109

Tabla 35 Resultados del Autoencoder AE-2 para la detección de packet loss, con comparativa frente a la Anomalía 1. ..... 111

Tabla 36 Comparativa de los tres métodos de detección para la Anomalía 2. ..... 111

Tabla 37 Distribución de clases en los subsets de datos tras el split estratificado para la Anomalía 2. ..... 113

Tabla 38 Resultados de la LSTM con ventanas de 20 y 40 pasos temporales. ..... 113

Tabla 39 Parámetros del episodio de restauración seleccionado (índices 378-380). ..... 115

Tabla 40 Parámetros del experimento de validación cruzada para la Anomalía 2. ..... 117

Tabla 41 Resultados de la validación cruzada sobre el segmento sin anomalías para la Anomalía 2. ..... 117

Tabla 42 Comparativa completa de todos los modelos evaluados para la Anomalía 2 ..... 118

Tabla 43 Distribución de clases y rangos de latencia resultantes del etiquetado empírico para la Anomalía 3. ..... 122

Tabla 44 Caracterización de la estructura temporal de los spikes de alta latencia. ..... 122

Tabla 45 Composición de los conjuntos de datos tras la generación de ventanas temporales. ..... 123

Tabla 46 Resultados del Autoencoder para la Anomalía 3 con distintos umbrales. ..... 125

Tabla 47 Comparativa de modelos de detección no supervisada para la Anomalía 3 ..... 125

Tabla 48 Distribución de clases tras el split estratificado para la Anomalía 3. ..... 126

Tabla 49 Resultados de la LSTM para la Anomalía 3 con distintos umbrales ..... 126

Tabla 50 Parámetros del episodio de restauración QoS para la Anomalía 3 ..... 128

Tabla 51 Parámetros del experimento de validación cruzada para la Anomalía 3. ..... 129

Tabla 52 Validación cruzada con datos normales ..... 130

Tabla 53 Comparativa completa de modelos para la Anomalía 3. ..... 131

Tabla 54 Distribución de clases y rangos de jitter resultantes del etiquetado empírico para la Anomalía 4. ..... 134

Tabla 55 Caracterización de la estructura temporal de los episodios de jitter extremo. ..... 135

Tabla 56 Estadísticas descriptivas por clase para las cinco features del modelo de Anomalía 4 ..... 135

Tabla 57 Composición de los conjuntos de datos tras la generación de ventanas temporales para la Anomalía 4. ..... 136

Tabla 58 Resultados del Autoencoder para la Anomalía 4 con distintos umbrales. ..... 138

Tabla 59 Comparativa de modelos de detección no supervisada para la Anomalía 4..... 139

Tabla 60 Distribución de clases tras el split estratificado para la Anomalía 4. ..... 140

Tabla 61 Resultados de la LSTM para la Anomalía 4 con distintos umbrales..... 140

Tabla 62 Parámetros del episodio de restauración del buffer de jitter para la Anomalía 4..... 143

Tabla 63 Comparativa completa de todos los modelos para la Anomalía 4..... 145

Tabla 64 Distribución de los 1001 logs generados por tipo de anomalía y fichero de salida. ..... 148

Tabla 65 Ejemplos reales de mensajes syslog generados para cada tipo de anomalía en formato Cisco IOS XR. ..... 151

Tabla 66 Resultados de la verificación de correlación por palabras clave. ..... 151

Tabla 67 Resultados de los modelos de detección sobre representación TF-IDF de los logs textuales. ..... 153

Tabla 68 Comparativa entre F1 numérico y F1 textual (Isolation Forest sobre TF-IDF) por anomalía. ..... 155

Tabla 69 Comparativa de resultados de detección no supervisada para las cuatro anomalías (umbral óptimo). ..... 156

Tabla 70 Comparativa de resultados de predicción supervisada (LSTM) para las cuatro anomalías. ..... 157

Tabla 71 Comparativa de consistencia en la validación con logs sintéticos. ..... 158

Tabla 72 Clasificación de las cuatro anomalías por tipo temporal y comparativa de rendimiento AE vs. LSTM. ..... 158

Tabla 73 Clasificación de las cuatro anomalías por grado de correlación multivariable y comparativa de rendimiento. ..... 159

Tabla 74 Brecha de F1 entre Autoencoder y LSTM por anomalía (positivo = LSTM superior)... 160

Tabla 75 Comparativa de tiempos de recuperación del sistema de restauración por anomalía. .... 161

Tabla 76 Evaluación de la eficacia de las acciones correctivas por tipo de anomalía. ..... 161

Tabla 77 Resumen de los mejores resultados del sistema por anomalía. ..... 166

Tabla 78 Alineación del proyecto con los Objetivos de Desarrollo Sostenible de la Agenda 2030. ..... 176

Tabla 79 Metas específicas de la Agenda 2030 a las que contribuye el proyecto, con descripción de la contribución concreta..... 181

# Capítulo 1. INTRODUCCIÓN

## *1.1 CONTEXTO GENERAL DEL PROBLEMA*

Las redes de comunicaciones constituyen uno de los pilares fundamentales de la sociedad digital actual. Infraestructuras como Internet, redes corporativas, centros de datos y sistemas de telecomunicaciones soportan de forma continua servicios críticos relacionados con la comunicación, el comercio electrónico, la educación, la sanidad y la industria. El correcto funcionamiento de estas redes resulta esencial para garantizar la disponibilidad y el rendimiento de los servicios que dependen de ellas [1].

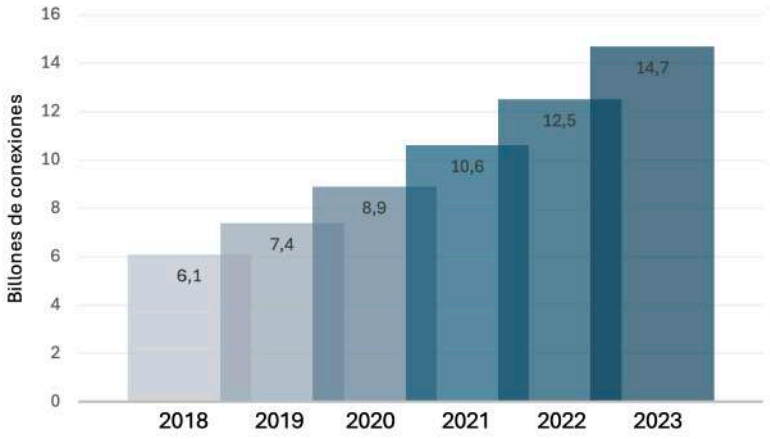


*Ilustración 4 Crecimiento global de las conexiones M2M. Fuente: Cisco Systems [2]*

En las últimas décadas, el crecimiento exponencial del tráfico de red y la proliferación de dispositivos conectados han incrementado de forma significativa la complejidad de las infraestructuras de comunicaciones. La aparición de arquitecturas distribuidas, entornos en la nube, redes definidas por software y servicios altamente dinámicos ha dado lugar a sistemas cada vez más heterogéneos, con múltiples capas, tecnologías y patrones de uso coexistiendo simultáneamente [2], [3].

Este aumento de complejidad no solo se refleja en el número de dispositivos y enlaces, sino también en la diversidad de aplicaciones y flujos de tráfico que circulan por la red. Los patrones de uso ya no son estáticos ni fácilmente predecibles, sino que varían en función del tiempo, la carga del sistema y el comportamiento de los usuarios. Como consecuencia, las redes modernas generan grandes volúmenes de información operativa que deben ser gestionados y analizados de manera continua [3], [4].

La gestión y supervisión de estas infraestructuras se ha convertido, por tanto, en una tarea cada vez más desafiante. Los administradores de red deben garantizar niveles adecuados de rendimiento, disponibilidad y fiabilidad en entornos altamente dinámicos, donde pequeñas

variaciones pueden tener un impacto significativo en el servicio global. Además, la detección temprana de situaciones anómalas o degradaciones del sistema resulta clave para evitar interrupciones, pérdidas económicas o deterioro de la experiencia de usuario [5], [6].

Los enfoques tradicionales de gestión de redes, basados en configuraciones estáticas y supervisión manual, encuentran serias limitaciones ante este escenario de creciente escala y complejidad. La necesidad de mecanismos más avanzados de monitorización y análisis se hace evidente en redes modernas, donde la capacidad de observar el comportamiento global del sistema y anticipar posibles problemas se vuelve un requisito esencial para su correcta operación [3].

## *1.2 PROBLEMA ACTUAL EN LA GESTIÓN DE REDES*

La gestión de redes de comunicaciones se enfrenta actualmente a un escenario de creciente complejidad operativa. Las infraestructuras modernas generan grandes volúmenes de datos derivados del tráfico, del estado de los dispositivos y del comportamiento de los servicios, lo que dificulta su supervisión y control mediante métodos convencionales [3], [5].

Tradicionalmente, la administración de redes se ha basado en enfoques manuales y reactivos, apoyados en sistemas de monitorización que emplean reglas predefinidas y umbrales fijos sobre métricas básicas. Estos mecanismos permiten detectar fallos evidentes, pero resultan insuficientes para identificar comportamientos anómalos complejos o situaciones que evolucionan de forma progresiva en el tiempo [5], [7]. En muchos casos, los problemas solo se detectan cuando ya han provocado una degradación significativa del servicio.

El crecimiento del tráfico y la diversidad de dispositivos conectados han incrementado de forma notable la cantidad de información que debe analizarse en tiempo real. Aunque las redes actuales disponen de abundantes datos, la dificultad radica en interpretarlos de manera eficiente y extraer conocimiento útil para la toma de decisiones operativas [3], [4]. Esta sobrecarga de información limita la capacidad de los operadores para identificar patrones relevantes y anticipar incidencias antes de que se materialicen.

Otro aspecto crítico es la falta de capacidad predictiva en los sistemas tradicionales de gestión. La mayoría de las herramientas existentes actúan únicamente tras la aparición de un fallo o una degradación observable, lo que se traduce en tiempos de reacción elevados y en una gestión principalmente correctiva [6]. Esta aproximación resulta especialmente problemática en entornos donde la disponibilidad y la calidad del servicio son requisitos críticos.

En consecuencia, el problema fundamental de la gestión de redes no reside en la ausencia de datos, sino en la incapacidad de analizarlos y correlacionarlos a tiempo para detectar comportamientos anómalos de forma temprana y actuar de manera proactiva. Esta limitación justifica la necesidad de explorar nuevos enfoques que permitan avanzar hacia una supervisión más inteligente y predictiva de las redes de comunicaciones [3], [5].

## *1.3 NECESIDAD DE NUEVOS ENFOQUES*

El escenario descrito en los apartados anteriores pone de manifiesto que los métodos tradicionales de gestión de redes resultan insuficientes para afrontar los retos actuales. La creciente complejidad de las infraestructuras, junto con el elevado volumen y la naturaleza dinámica de los datos generados, requiere mecanismos capaces de analizar la información de forma continua y automática, reduciendo la dependencia de la intervención manual [3], [5].

En este contexto, surge la necesidad de adoptar enfoques basados en el análisis avanzado de datos que permitan comprender el comportamiento global de la red, más allá de la simple supervisión de métricas individuales. En lugar de centrarse exclusivamente en valores estáticos o reglas predefinidas, estos enfoques persiguen identificar patrones de funcionamiento normales y detectar desviaciones significativas que puedan indicar degradaciones del servicio o situaciones potencialmente problemáticas [8], [9].

La detección temprana de comportamientos anómalos constituye un aspecto clave para mejorar la gestión de redes modernas. Identificar de forma anticipada cambios progresivos en el rendimiento o en el uso de los recursos permite actuar antes de que los problemas se traduzcan en fallos críticos o interrupciones del servicio, reduciendo así los tiempos de reacción y el impacto sobre los usuarios finales [5], [8].

Asimismo, la capacidad de anticipar la evolución del estado de la red resulta fundamental en entornos donde la disponibilidad y la calidad del servicio son requisitos esenciales. La predicción de fallos o degradaciones potenciales facilita una gestión más proactiva, orientada a la prevención en lugar de a la corrección, y contribuye a una utilización más eficiente de los recursos disponibles [3], [5].

Por último, cualquier enfoque destinado a la gestión de redes actuales debe ser escalable y adaptable. Las soluciones empleadas han de ser capaces de operar en entornos heterogéneos y de crecer en paralelo al aumento del tamaño y la complejidad de la infraestructura, manteniendo su eficacia incluso ante grandes volúmenes de datos y cambios constantes en los patrones de uso [4], [10].

Todo ello ha motivado el interés por técnicas de análisis avanzado basadas en datos, orientadas a dotar a la gestión de redes de capacidades automáticas, adaptativas y predictivas, que permitan afrontar de manera eficaz los desafíos de las infraestructuras de comunicaciones modernas [9].

## *1.4 ENFOQUE GENERAL DEL TRABAJO*

Este Trabajo Fin de Grado se enmarca en el ámbito de la gestión y supervisión de redes de comunicaciones modernas, con el objetivo de analizar cómo las técnicas de análisis avanzado basadas en datos pueden contribuir a mejorar la detección, predicción y gestión de situaciones anómalas en infraestructuras de red complejas.

El trabajo adopta un enfoque académico y experimental, centrado en el estudio del comportamiento de la red a partir de métricas operativas y en la evaluación de distintos mecanismos de análisis aplicados a dicho comportamiento. En particular, se aborda el problema desde una perspectiva global, considerando tanto la identificación temprana de desviaciones respecto al funcionamiento normal como la anticipación de posibles fallos que puedan derivar en una degradación del servicio [3], [8].

Este proyecto no pretende desarrollar una solución industrial completa ni cubrir de forma exhaustiva todos los escenarios posibles en la gestión de redes reales. En su lugar, se plantea como un estudio controlado que permita analizar y comprender el potencial de los enfoques basados en datos aplicados a la supervisión de redes, así como sus ventajas y limitaciones en entornos reproducibles.

A lo largo del trabajo se estudian de manera conjunta los conceptos de detección, predicción y restauración de anomalías y fallos, con el objetivo de ofrecer una visión integrada del problema. Este enfoque permite sentar las bases para sistemas de gestión más inteligentes, capaces de evolucionar desde modelos puramente reactivos hacia estrategias más proactivas y predictivas, alineadas con las necesidades de las redes actuales [5], [8].

## *1.5 ALCANCE Y OBJETIVOS GENERALES*

El alcance de este Trabajo Fin de Grado comprende el diseño, implementación y evaluación experimental de un sistema de detección, predicción y restauración automática de anomalías en redes de telecomunicaciones, aplicado sobre el dataset Network-Anomaly-Dataset (Kaggle/UPM, 2024), que recoge métricas reales capturadas en una infraestructura de routers Cisco IOS XR virtuales.

El sistema desarrollado opera en un entorno de experimentación controlado y reproducible, sin constituir una solución desplegable en producción. Su alcance abarca la implementación de modelos de aprendizaje no supervisado (Autoencoder, Isolation Forest, One-Class SVM) y supervisado (LSTM), la evaluación comparativa de su rendimiento mediante métricas estándar de clasificación binaria, el modelado del proceso de restauración automática para cada tipo de anomalía, y la validación textual mediante logs sintéticos en formato Cisco IOS XR.

Los resultados obtenidos deben interpretarse en el contexto experimental del dataset utilizado, y su extrapolación a entornos de producción requeriría validación adicional sobre datos independientes y arquitecturas de inferencia en tiempo real.

Además de los objetivos técnicos planteados, el proyecto contribuye indirectamente a diversos Objetivos de Desarrollo Sostenible (ODS) relacionados con la innovación tecnológica, la digitalización y el desarrollo de infraestructuras más eficientes y resilientes. La relación del trabajo con los ODS de la Agenda 2030 se analiza en el Anexo I.

## *1.6 ESTRUCTURA DEL TRABAJO*

El presente Trabajo Fin de Grado se estructura en varios capítulos organizados de forma progresiva, con el objetivo de guiar al lector desde el contexto general del problema hasta el análisis de los resultados obtenidos y las conclusiones.
A continuación, se describe de manera sintética el contenido de cada uno de los capítulos que componen el documento:

**Capítulo 1. Introducción**
Presenta el contexto general de las redes de comunicaciones modernas y los principales problemas asociados a su gestión y supervisión. Asimismo, se expone la necesidad de nuevos enfoques de análisis, se define el enfoque general del trabajo, su alcance y los objetivos globales del proyecto.

**Capítulo 2. Descripción de las Tecnologías**
Introduce los conceptos fundamentales necesarios para la comprensión del trabajo. En este capítulo se abordan las redes de comunicaciones y su gestión, el concepto de anomalía, fallo y ataque en redes, así como los fundamentos de la Inteligencia Artificial, el Machine Learning y el Deep Learning. Además, se describen las métricas de red utilizadas y las principales herramientas software empleadas en el desarrollo del proyecto.

**Capítulo 3. Estado de la Cuestión**
Analiza los principales enfoques existentes para la detección y predicción de anomalías en redes de comunicaciones. Se revisan las soluciones tradicionales y las basadas en Machine Learning y Deep Learning, así como los modelos más relevantes utilizados en la literatura. Finalmente, se presenta una comparativa de enfoques y se justifica la selección de los modelos empleados en este trabajo.

**Capítulo 4. Sistema Propuesto**
Describe el sistema planteado en el proyecto, incluyendo la arquitectura general, la selección de anomalías y modelos, el proceso de generación y recolección de datos en un entorno simulado y el diseño experimental seguido para la evaluación de los modelos.

**Capítulo 5. Resultados y Análisis Comparativo**
Presenta los resultados obtenidos tras la aplicación de los modelos seleccionados, junto con un análisis crítico de su comportamiento y rendimiento en los distintos escenarios evaluados.

**Capítulo 6. Conclusiones y Trabajo Futuro**
Recoge las conclusiones principales del trabajo, evalúa el grado de cumplimiento de los objetivos planteados y propone posibles líneas de trabajo futuro orientadas a la mejora y ampliación del sistema desarrollado.

Esta estructura permite abordar el estudio de forma ordenada y coherente, facilitando la comprensión del problema, de los fundamentos teóricos y tecnológicos, de la metodología empleada y de los resultados alcanzados.

# Capítulo 2. DESCRIPCIÓN DE LAS TECNOLOGÍAS

## *2.1 REDES DE COMUNICACIONES Y GESTIÓN DE RED*

Una red de comunicaciones puede definirse como el conjunto de dispositivos interconectados que intercambian información mediante enlaces físicos o lógicos, siguiendo protocolos que regulan el formato, la temporización y el control de los datos transmitidos. Su objetivo principal es permitir la comunicación fiable y eficiente entre sistemas distribuidos, independientemente de su localización geográfica [11].

Desde el punto de vista funcional, una red se compone de nodos finales (hosts), dispositivos de interconexión (routers, switches) y medios de transmisión. Estos elementos cooperan para transportar paquetes de datos a través de diferentes capas, organizadas habitualmente según modelos de referencia como OSI o TCP/IP, que separan las responsabilidades de comunicación y facilitan la interoperabilidad entre tecnologías heterogéneas [11].

El crecimiento de Internet y de los servicios digitales ha incrementado notablemente la escala y complejidad de las redes modernas. Actualmente, las redes deben soportar grandes volúmenes de tráfico, múltiples tipos de aplicaciones y requisitos estrictos de calidad de servicio, disponibilidad y seguridad. Esta complejidad provoca que el comportamiento normal de la red no sea trivial de caracterizar y que pequeñas alteraciones puedan derivar en degradaciones significativas del servicio [2], [3].

En este contexto, las redes de comunicaciones generan de forma continua métricas y trazas que describen su estado operativo, como tasas de tráfico, latencia, pérdidas de paquetes o estadísticas de flujo. El análisis de esta información resulta clave para la gestión y supervisión de la red, ya que permite identificar desviaciones respecto al comportamiento esperado y anticipar posibles fallos o incidencias [3], [10].

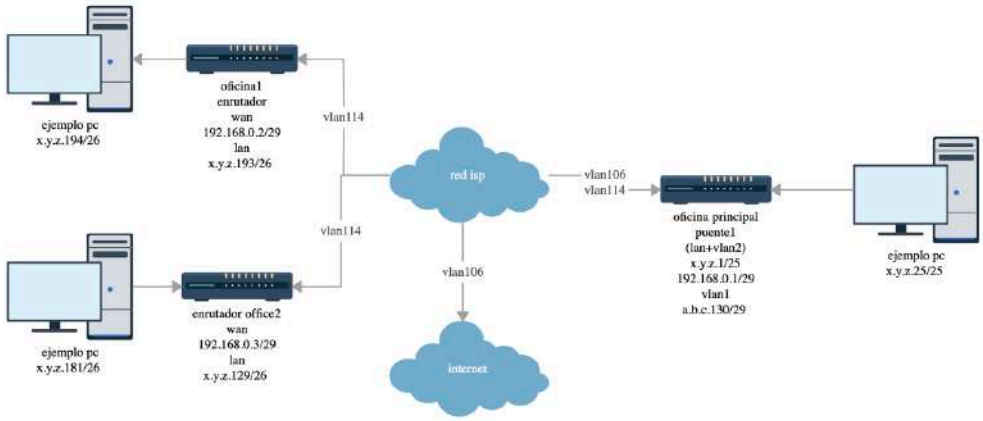


*Ilustración 5 Esquema general de una red de comunicaciones mostrando nodos, enlaces y flujo de datos. Fuente: Creately [12]*

### 2.1.1 GESTIÓN Y MONITORIZACIÓN DE REDES

La gestión y monitorización de redes engloba el conjunto de mecanismos y herramientas destinados a supervisar el estado operativo de una red, garantizar su correcto funcionamiento y detectar incidencias que puedan afectar a la calidad del servicio. Tradicionalmente, estas tareas se han basado en la observación continua de métricas de rendimiento y en la actuación manual del operador ante eventos anómalos [5].

Los sistemas clásicos de gestión de red se estructuran en torno a modelos como FCAPS (Fault, Configuration, Accounting, Performance, Security), que permiten clasificar las tareas de supervisión en categorías funcionales [6]. Dentro de este marco, la monitorización se centra principalmente en la recogida de métricas como latencia, pérdida de paquetes, jitter, utilización de enlaces o carga de dispositivos, mientras que la gestión de fallos se activa cuando dichas métricas superan ciertos umbrales predefinidos [13].

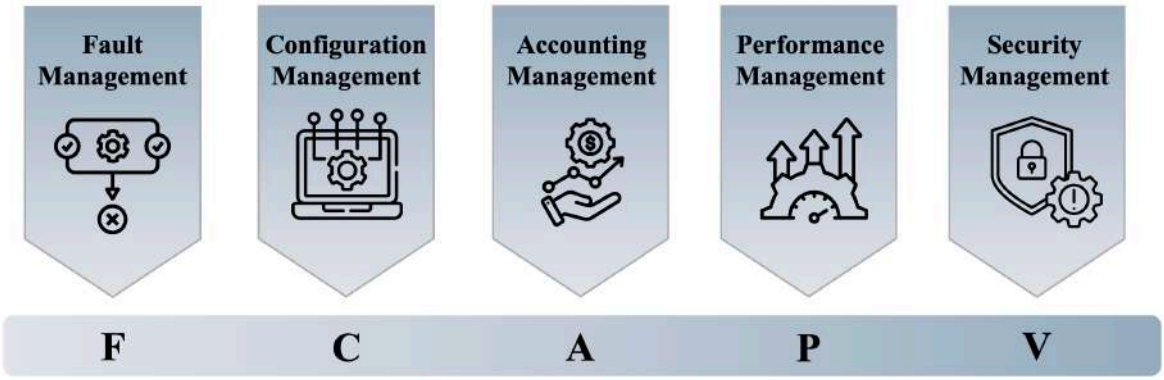


*Ilustración 6 Esquema general de gestión de red basada en FCAPS Adaptado de Solís [13]*

Uno de los principales problemas de este enfoque es su carácter reactivo. La detección de fallos suele producirse una vez que el problema ya ha impactado al servicio, y la identificación de la causa raíz depende en gran medida de la experiencia del operador [34]. Además, el crecimiento del tráfico y la complejidad de las redes modernas generan volúmenes de datos que dificultan el análisis manual en tiempo real [5], [7].

Para la monitorización, se han utilizado ampliamente protocolos y herramientas como SNMP, NetFlow e IPFIX, que permiten recopilar información sobre el estado de los dispositivos y los flujos de tráfico que circulan por la red [32][33]. En particular, los sistemas basados en flujos de red proporcionan una visión agregada del tráfico, reduciendo el volumen de datos frente a la inspección de paquetes completa y facilitando su análisis posterior [1], [14], [10].

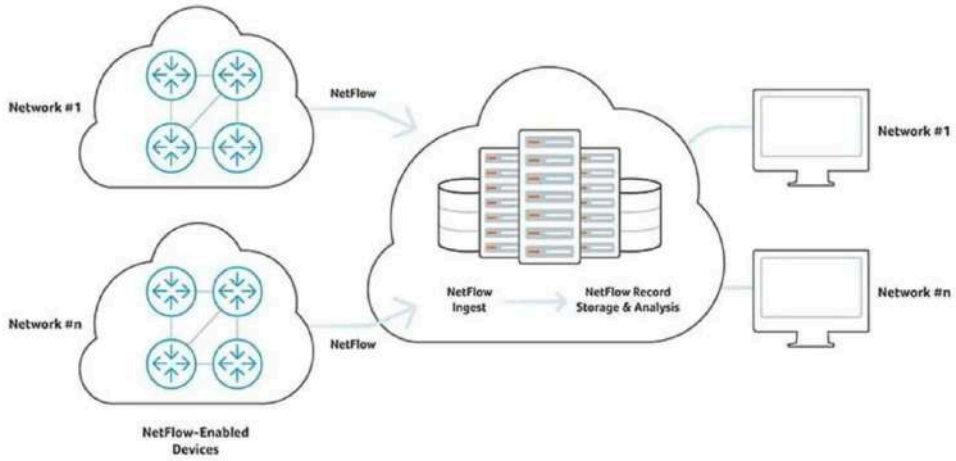


*Ilustración 7 Arquitectura de monitorización basada en NetFlow/IPFIX. Fuente: Kentik [15].*

En el ámbito de la detección de incidencias, los sistemas de detección de intrusiones (IDS) han desempeñado un papel relevante, inicialmente mediante reglas y firmas predefinidas, como en los casos de Snort o Bro [29][30]. Si bien estos sistemas son eficaces frente a ataques conocidos, presentan limitaciones para detectar comportamientos anómalos desconocidos o degradaciones progresivas del rendimiento [16], [17].

La evolución hacia redes más dinámicas y heterogéneas ha puesto de manifiesto la necesidad de mecanismos de monitorización más inteligentes y adaptativos. En este contexto, comienzan a integrarse técnicas de análisis avanzado y aprendizaje automático que permiten modelar el comportamiento normal de la red y detectar desviaciones sin depender exclusivamente de reglas fijas [5][22]. Este cambio marca la transición desde una gestión reactiva hacia enfoques más proactivos y predictivos, que constituyen la base de los sistemas de gestión de red basados en Inteligencia Artificial [5], [9].

### 2.1.2 LIMITACIONES DE LA GESTIÓN TRADICIONAL

A pesar de que los mecanismos clásicos de gestión y monitorización de redes han sido ampliamente utilizados durante décadas, presentan limitaciones significativas cuando se aplican a entornos de red modernos, caracterizados por un elevado volumen de tráfico, alta heterogeneidad y comportamientos dinámicos. Estas limitaciones afectan directamente a la capacidad de detectar anomalías de forma eficaz y de anticipar fallos antes de que impacten en el servicio.

#### *2.1.2.1 Uso de umbrales estáticos*

Uno de los enfoques más comunes en la gestión tradicional consiste en la definición de umbrales fijos sobre métricas de red como el uso de ancho de banda, la latencia o la pérdida de paquetes. Cuando una métrica supera el valor establecido, se genera una alerta. Sin embargo, este enfoque asume un comportamiento estable de la red, lo cual rara vez se cumple en escenarios reales [5].

Los umbrales estáticos no se adaptan a variaciones normales del tráfico, como picos temporales o patrones estacionales, lo que provoca un elevado número de falsos positivos o, por el contrario, la omisión de anomalías reales si estas no superan el umbral definido [4].

#### *2.1.2.2 Dependencia de reglas predefinidas*

La gestión tradicional también se apoya en reglas manuales diseñadas por expertos, que describen patrones conocidos de fallo o comportamiento anómalo. Este enfoque resulta eficaz únicamente frente a incidencias previamente identificadas, pero carece de capacidad para detectar anomalías nuevas o desconocidas [16].

Además, el mantenimiento de estas reglas requiere un esfuerzo continuo, ya que deben actualizarse conforme evoluciona la red y aparecen nuevos servicios, protocolos o vectores de ataque [7]. En redes complejas, esta dependencia de conocimiento experto se convierte en un factor limitante.

#### *2.1.2.3 Reacción tardía ante los fallos*

Los sistemas tradicionales suelen operar de forma reactiva: la detección del problema se produce una vez que el fallo ya ha impactado en el rendimiento o ha provocado una interrupción del servicio. Este enfoque implica tiempos de respuesta elevados y dificulta la mitigación temprana de incidentes críticos [6].

En entornos donde la disponibilidad es un requisito esencial, como redes corporativas o infraestructuras críticas, la reacción tardía puede traducirse en pérdidas económicas, degradación del servicio y reducción de la confianza de los usuarios [18].

#### *2.1.2.4 Ausencia de capacidades predictivas*

Otra limitación fundamental es la falta de mecanismos de predicción. Las técnicas tradicionales no permiten anticipar el comportamiento futuro de la red ni identificar tendencias que indiquen la aparición de un fallo inminente [5]. Esto impide la adopción de estrategias de mantenimiento preventivo o proactivo.

En contraste, los enfoques basados en aprendizaje automático y deep learning permiten modelar la evolución temporal de las métricas de red y detectar desviaciones antes de que se manifiesten como fallos visibles [19], lo que pone de relieve la necesidad de superar los métodos clásicos.

## *2.2 ANOMALÍAS, FALLOS Y ATAQUES EN REDES*

El funcionamiento de una red de comunicaciones no es estático, sino que está sujeto a variaciones continuas en el tráfico, la carga y el comportamiento de los dispositivos que la componen. En este contexto, resulta fundamental distinguir entre los diferentes tipos de eventos que pueden alterar el comportamiento normal de la red, ya sea de forma accidental o intencionada. Entre estos eventos se encuentran las anomalías, los fallos técnicos y los ataques de seguridad, que, aunque pueden manifestarse de manera similar desde el punto de vista del tráfico o del rendimiento, responden a causas y objetivos diferentes [4], [5].

La correcta identificación y clasificación de estos fenómenos es un aspecto clave para el diseño de sistemas eficaces de monitorización y detección. Mientras que las soluciones tradicionales se centran en la detección reactiva de fallos evidentes, los enfoques modernos basados en análisis de comportamiento buscan identificar desviaciones tempranas que permitan anticipar problemas antes de que impacten de forma crítica en la red [9].

En este apartado se introducen y analizan los conceptos fundamentales de anomalía, fallo y ataque en redes de comunicaciones, describiendo sus características principales, sus diferencias y el impacto que tienen sobre el rendimiento, la disponibilidad y la experiencia de usuario. Este marco conceptual servirá de base para la selección posterior de técnicas de detección y modelos de Inteligencia Artificial aplicados a la gestión inteligente de redes.

### 2.2.1 DEFINICIÓN DE ANOMALÍA EN REDES

En el contexto de las redes de comunicaciones, una **anomalía** se define como cualquier **desviación significativa respecto al comportamiento normal esperado del sistema**, ya sea en términos de tráfico, rendimiento o uso de recursos. Estas desviaciones no siempre implican un fallo explícito, pero sí indican que el estado de la red se aparta de los patrones habituales aprendidos a partir de su funcionamiento histórico [5], [8].

A diferencia de los fallos clásicos, que suelen estar asociados a errores físicos o lógicos claramente identificables, las anomalías pueden manifestarse de forma **gradual, intermitente o sutil**, lo que dificulta su detección mediante técnicas tradicionales basadas en reglas o umbrales fijos [5], [9]. En muchos casos, una anomalía representa una **señal temprana** de un problema mayor, como una congestión progresiva, una degradación del servicio o el inicio de un ataque de seguridad.

Desde un punto de vista operativo, las anomalías en redes suelen identificarse a partir del análisis de métricas como el volumen de tráfico, la latencia, la pérdida de paquetes, el jitter o la distribución de flujos. Cuando estas métricas muestran comportamientos que se alejan del patrón considerado normal para una red concreta, se considera que existe un comportamiento anómalo [3], [9].

Es importante destacar que el concepto de anomalía es **dependiente del contexto**. Un aumento brusco de tráfico puede ser perfectamente normal en determinados periodos (por ejemplo, horas punta), pero considerado anómalo en otros escenarios o si no se corresponde con el patrón histórico aprendido por el sistema [4]. Por este motivo, la detección de anomalías requiere modelos capaces de adaptarse dinámicamente al comportamiento de la red [8].

#### *2.2.1.1 Comportamiento normal frente a comportamiento anómalo*

El **comportamiento normal** de una red se define como aquel conjunto de patrones estadísticos y temporales que describen su funcionamiento habitual bajo condiciones esperadas de carga y uso. Estos patrones incluyen relaciones estables entre métricas,

periodicidades diarias o semanales y rangos de variación aceptables para cada parámetro [3], [19].

Por el contrario, el **comportamiento anómalo** aparece cuando una o varias de estas características se alteran de forma significativa. Esto puede manifestarse como:

- picos inesperados de tráfico,
- cambios abruptos en la distribución de flujos,
- incrementos anormales de latencia o pérdida de paquetes,
- patrones temporales que no se corresponden con el histórico de la red [8], [20].

En la literatura especializada, las anomalías se consideran especialmente relevantes porque **no requieren etiquetas previas** para ser detectadas, lo que las convierte en un objetivo natural para técnicas de aprendizaje no supervisado, como los autoencoders o los modelos basados en aislamiento [21], [24]. Estas técnicas aprenden el comportamiento normal y detectan anomalías como desviaciones respecto a dicho modelo [21, 22].

### 2.2.2 FALLOS EN REDES

Los fallos de red se refieren a eventos que provocan una degradación significativa o la interrupción total del funcionamiento normal de una infraestructura de comunicaciones. A diferencia de las anomalías, que pueden representar desviaciones temporales o comportamientos atípicos sin una causa estructural clara, los fallos suelen estar asociados a problemas identificables en componentes físicos o lógicos de la red y tienen un impacto directo en la disponibilidad del servicio [5], [7].

Estos fallos pueden clasificarse en varias categorías según su origen y naturaleza.

#### *2.2.2.1 Fallos físicos*

Los fallos físicos están relacionados con averías o deterioros en los elementos materiales de la red. Incluyen problemas en cables, interfaces de red, routers, switches o dispositivos de transmisión inalámbrica. Este tipo de fallos suele provocar pérdidas completas de conectividad o degradaciones severas del rendimiento, como cortes de enlace o aumento brusco de la latencia [7], [17].

Entre las causas más comunes se encuentran el desgaste del hardware, errores de alimentación eléctrica, condiciones ambientales adversas o daños accidentales. Los fallos físicos suelen ser relativamente sencillos de identificar, pero su reparación puede implicar tiempos elevados de intervención manual.

#### *2.2.2.2 Fallos lógicos*

Los fallos lógicos se producen cuando la infraestructura física funciona correctamente, pero la red no opera de forma adecuada debido a errores de configuración, fallos de software o problemas en los protocolos de comunicación. Ejemplos habituales son configuraciones incorrectas de rutas, errores en tablas de encaminamiento, conflictos de direcciones IP o fallos en procesos de control [10], [14].

Este tipo de fallos resulta especialmente complejo de diagnosticar, ya que no existe un daño visible en el hardware y los síntomas pueden manifestarse de forma intermitente o localizada.

#### *2.2.2.3 Congestión*

La congestión es un fallo asociado a la saturación de los recursos de red, como el ancho de banda, las colas de los dispositivos o la capacidad de procesamiento de los nodos intermedios. Ocurre cuando la carga de tráfico supera la capacidad de la red para gestionarla de forma eficiente, lo que provoca retrasos, pérdida de paquetes y degradación de la calidad de servicio [3], [5].

Aunque la congestión puede estar relacionada con un uso legítimo elevado, también puede ser consecuencia de configuraciones ineficientes o de comportamientos anómalos persistentes.

#### *2.2.2.4 Caídas de enlaces*

Las caídas de enlaces representan una interrupción total de la comunicación entre dos nodos de la red. Pueden deberse tanto a fallos físicos (rotura de cable, fallo de interfaz) como a errores lógicos (desactivación de enlaces, problemas de negociación de protocolos) [10], [11].

Este tipo de fallo suele generar efectos en cascada, como reencaminamientos automáticos, incremento de latencia en rutas alternativas o incluso la pérdida de conectividad completa en determinadas zonas de la red.

### 2.2.3 ATAQUES Y EVENTOS DE SEGURIDAD

Además de los fallos internos de la infraestructura, las redes de comunicaciones están expuestas a **amenazas externas** que alteran su comportamiento normal de forma intencionada. Estos eventos de seguridad se diferencian de los fallos clásicos en que persiguen un objetivo malicioso, como la degradación del servicio, el acceso no autorizado o la extracción de información. Desde el punto de vista de la monitorización, estos ataques suelen manifestarse como **anomalías persistentes o abruptas en el tráfico**, lo que permite su detección mediante técnicas de análisis de comportamiento [5], [7].

#### *2.2.3.1 Ataques de Denegación de Servicio Distribuida (DDoS)*

Los ataques DDoS tienen como objetivo **saturar los recursos de la red o de un servicio** mediante el envío masivo de tráfico desde múltiples orígenes. Esta sobrecarga provoca un aumento repentino del volumen de paquetes, del consumo de ancho de banda y de la latencia, afectando a la disponibilidad del sistema [23], [24].

Desde el punto de vista de la detección, los ataques DDoS presentan características observables como:

- Incrementos bruscos y sostenidos de tráfico entrante,
- Patrones repetitivos en flujos de red,

- Comportamientos altamente desviados respecto al tráfico habitual.

Estas propiedades hacen que los DDoS sean especialmente adecuados para su detección mediante modelos basados en análisis de flujos y técnicas de aprendizaje automático no supervisado [5], [9].

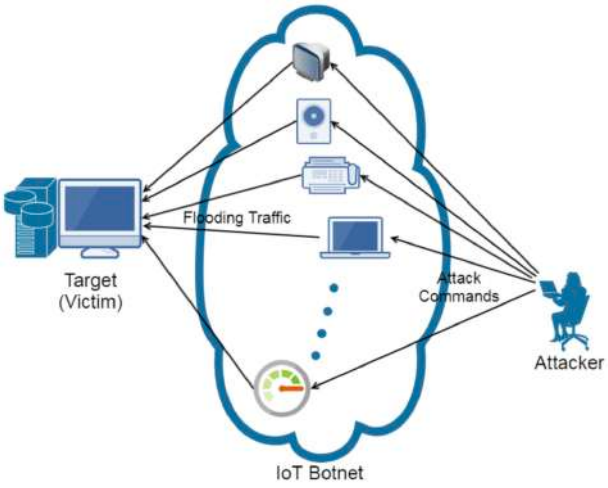


*Ilustración 8 Esquema de un ataque DDoS mostrando múltiples orígenes hacia un mismo destino.*

*Fuente: ResearchGate [66].*

### *2.2.3.2 Tráfico malicioso*

El tráfico malicioso engloba comunicaciones asociadas a actividades como escaneo de puertos, propagación de malware, intentos de intrusión o exfiltración de datos. A diferencia de los DDoS, este tipo de tráfico puede ser **más sutil** y no siempre implica un gran volumen de datos, lo que dificulta su detección mediante métodos tradicionales [16], [17].

Este tráfico suele caracterizarse por:

- Frecuencias anómalas de conexión,
- Uso de puertos inusuales,
- Duraciones atípicas de sesiones,
- Comportamientos no coherentes con el perfil normal del dispositivo.

Los sistemas de detección modernos emplean análisis estadístico avanzado y modelos de aprendizaje profundo para identificar estas desviaciones en el comportamiento de los flujos de red [22], [25].

### *2.2.3.3 Comportamiento anómalo inducido*

No todos los eventos de seguridad corresponden a ataques claramente definidos. En muchos casos, la red presenta **comportamientos anómalos inducidos** como consecuencia indirecta de una acción maliciosa, por ejemplo:

- Dispositivos comprometidos que generan tráfico irregular,
- Cambios en los patrones de uso tras una intrusión,
- Alteraciones progresivas del rendimiento debido a actividades persistentes no detectadas.

Este tipo de comportamiento resulta especialmente complejo de identificar, ya que puede evolucionar de forma gradual y confundirse con variaciones normales del tráfico. Por ello, la detección basada en aprendizaje automático no supervisado y modelos de reconstrucción, como los autoencoders, resulta especialmente eficaz al no requerir conocimiento previo del ataque [22], [25].

### 2.2.4 DIFERENCIA ENTRE ANOMALÍAS, FALLO Y ATAQUE

Aunque los términos *anomalía*, *fallo* y *ataque* se utilizan con frecuencia de forma indistinta en el ámbito de las redes de comunicaciones, representan conceptos distintos que es necesario diferenciar con precisión para diseñar sistemas eficaces de detección y respuesta.

Una **anomalía de red** se define como cualquier desviación significativa respecto al comportamiento normal o esperado del sistema, sin que necesariamente implique una causa maliciosa o un fallo estructural. Las anomalías pueden ser temporales, sutiles y difíciles de interpretar, y suelen detectarse mediante el análisis estadístico o el aprendizaje automático sobre métricas de red. Su identificación se basa en patrones de comportamiento y no en reglas explícitas [5], [8].

Un **fallo de red** es un evento concreto que provoca una degradación o interrupción del servicio y tiene una causa técnica identificable, como un error de configuración, una congestión persistente o un fallo físico en un enlace o dispositivo. A diferencia de las anomalías, los fallos suelen tener un impacto directo y observable sobre la disponibilidad o el rendimiento de la red, y tradicionalmente se gestionan mediante mecanismos reactivos de monitorización [7], [17].

Por su parte, un **ataque o evento de seguridad** es una acción deliberada y maliciosa cuyo objetivo es comprometer la confidencialidad, integridad o disponibilidad de la red. Este tipo de eventos incluye ataques como DDoS, escaneos, inyecciones de tráfico malicioso o accesos no autorizados. Desde el punto de vista del comportamiento del tráfico, muchos ataques se manifiestan inicialmente como anomalías, lo que dificulta su distinción sin análisis avanzado [16], [24].

La diferencia fundamental entre estos tres conceptos radica en **su causa y finalidad**:

- las anomalías describen *comportamientos atípicos*,
- los fallos representan *problemas técnicos*,
- los ataques responden a *acciones intencionadas*.

No obstante, en entornos reales estas categorías no son excluyentes. Un ataque puede provocar anomalías previas antes de materializarse como un fallo, y un fallo puede generar patrones anómalos que no tienen origen malicioso. Por este motivo, los sistemas modernos de gestión de redes tienden a abordar la detección desde una perspectiva unificada, centrada en el comportamiento, en lugar de basarse únicamente en clasificaciones rígidas [8], [9].

Para clarificar estas diferencias, la Tabla 3 resume las principales características que distinguen anomalías, fallos y ataques en redes de comunicaciones.

| Característica | Anomalía | Fallo | Ataque |
|---|---|---|---|
| Origen | Desconocido o variación natural | Técnico | Malicioso |
| Intencionalidad | No | No | Si |
| Impacto inmediato | Variable | Alto | Variable / Alto |
| Persistencia | Temporal o intermitente | Persistente hasta resolución | Dependiente del ataque |
| Detección tradicional | Difícil | Alta | Media |
| Enfoque IA | Detección no supervisada | Predicción y diagnostico | Clasificación y detección |

*Tabla 2 Comparación entre anomalías, fallos y ataques en redes*

### 2.2.5 IMPACTO DE LAS ANOMALÍAS EN LA RED

Las anomalías, fallos y ataques en redes de comunicaciones tienen un impacto directo y medible sobre el funcionamiento global de la infraestructura. Estos efectos no solo afectan al rendimiento técnico de la red, sino también a la disponibilidad de los servicios, a la experiencia de los usuarios y a los costes operativos asociados a su gestión. La identificación temprana de estos impactos es clave para justificar la necesidad de mecanismos avanzados de detección y predicción.

#### *2.2.5.1 Impacto en la calidad de servicio (QoS)*

La calidad de servicio se ve afectada de forma inmediata cuando se producen anomalías en la red. Eventos como congestión, fallos de encaminamiento o tráfico anómalo provocan incrementos en la latencia, pérdida de paquetes, jitter elevado y reducción del throughput efectivo. Estos efectos degradan el comportamiento de aplicaciones sensibles al tiempo, como voz sobre IP, streaming o servicios interactivos [3], [9].

Las anomalías persistentes pueden alterar los patrones normales de tráfico, dificultando la planificación de recursos y provocando un uso ineficiente del ancho de banda disponible [5].

#### *2.2.5.2 Impacto en la disponibilidad*

La disponibilidad de la red se ve comprometida cuando las anomalías derivan en caídas de enlaces, reinicios de dispositivos o bloqueos de servicios. Fallos físicos, errores lógicos o ataques de denegación de servicio pueden provocar interrupciones parciales o totales del servicio, afectando a sistemas críticos y aplicaciones dependientes de conectividad continua [7], [17].

En entornos empresariales y de infraestructuras críticas, incluso interrupciones de corta duración pueden tener consecuencias significativas, lo que refuerza la necesidad de detección temprana y mecanismos de recuperación automática [6].

#### *2.2.5.3 Impacto en la experiencia de usuario*

Desde el punto de vista del usuario final, las anomalías se traducen en degradación perceptible del servicio: tiempos de respuesta elevados, desconexiones inesperadas, errores de transmisión o imposibilidad de acceder a recursos. Estos problemas afectan directamente a la satisfacción del usuario y a la confianza en el servicio proporcionado [8].

En aplicaciones comerciales o plataformas digitales, una mala experiencia de usuario puede derivar en pérdida de clientes, reducción del uso del servicio y daño reputacional para la organización [1].

#### *2.2.5.4 Impacto económico y costes operativos*

Las anomalías en la red generan costes tanto directos como indirectos. Entre los costes directos se incluyen el tiempo de inactividad, la intervención manual del personal técnico, el consumo adicional de recursos y la necesidad de reemplazo o reparación de equipos. Los costes indirectos abarcan la pérdida de productividad, la degradación del servicio ofrecido y posibles sanciones contractuales por incumplimiento de acuerdos de nivel de servicio (SLA) [26].

Además, la gestión reactiva de anomalías incrementa la carga operativa y dificulta la escalabilidad de las redes modernas, lo que justifica la adopción de enfoques predictivos basados en técnicas de Inteligencia Artificial [5], [9].

## *2.3 INTRODUCCIÓN A LA INTELIGENCIA ARTIFICIAL*

La Inteligencia Artificial (IA) se ha consolidado como una de las tecnologías clave para abordar problemas complejos en entornos caracterizados por grandes volúmenes de datos, alta variabilidad y dinámicas cambiantes. En el ámbito de las redes de comunicaciones, la IA permite superar las limitaciones de los enfoques tradicionales de gestión y monitorización, ofreciendo capacidades avanzadas de análisis, detección y predicción de comportamientos anómalos.

Las redes modernas generan de forma continua grandes cantidades de datos procedentes de métricas de rendimiento, flujos de tráfico, eventos de seguridad y registros de dispositivos. Analizar esta información de manera manual o mediante reglas estáticas resulta inviable debido a su escala y complejidad. En este contexto, la IA proporciona mecanismos automáticos capaces de aprender patrones de comportamiento normal, identificar desviaciones relevantes y adaptarse a cambios en la red sin intervención humana constante [5], [6].

Además, la aplicación de técnicas de IA en la gestión de redes no se limita únicamente a la detección de fallos o ataques una vez ocurridos. Uno de sus principales aportes es la capacidad de anticipar eventos futuros mediante el análisis de patrones históricos y dependencias temporales, facilitando una gestión proactiva orientada a la prevención y la optimización del servicio [4], [6].

Este trabajo se apoya en estos principios para explorar el uso de modelos de Inteligencia Artificial en la detección, predicción y restauración de anomalías en redes de comunicaciones, integrando técnicas de aprendizaje automático y aprendizaje profundo en un entorno de simulación controlado [5], [9].

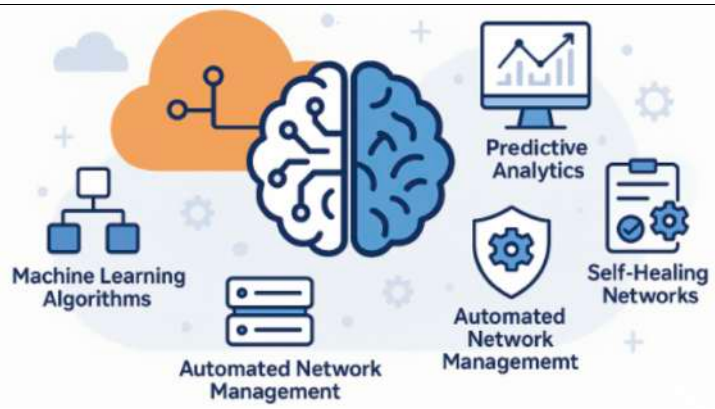


*Ilustración 9 Evolución de la gestión de redes: de sistemas reactivos a gestión inteligente basada en IA. Fuente: LinkedIn Pulse [27].*

### 2.3.1 CONCEPTO DE INTELIGENCIA ARTIFICIAL

La Inteligencia Artificial puede definirse como el conjunto de técnicas y métodos que permiten a los sistemas informáticos realizar tareas que normalmente requieren inteligencia humana, como el aprendizaje, el razonamiento, la toma de decisiones o la identificación de patrones en datos complejos [28], [29].

A diferencia de los sistemas tradicionales programados explícitamente mediante reglas fijas, los sistemas basados en IA no dependen exclusivamente de instrucciones predefinidas. En su lugar, aprenden a partir de datos, ajustando su comportamiento en función de la información observada. Este enfoque resulta especialmente adecuado en dominios donde las reglas son difíciles de definir manualmente o cambian con el tiempo, como ocurre en las redes de comunicaciones [6], [9].

En el contexto de la gestión de redes, los sistemas tradicionales se basan habitualmente en:

- umbrales estáticos para métricas específicas,
- reglas definidas por expertos,
- respuestas reactivas ante eventos detectados.

Por el contrario, los sistemas basados en IA presentan características diferenciadoras:

- capacidad de aprendizaje automático a partir de datos históricos,
- adaptación a nuevos patrones de tráfico o comportamiento,
- detección de anomalías no previstas explícitamente,
- posibilidad de realizar predicciones sobre el estado futuro de la red [4], [5].

Esta diferencia resulta clave para abordar problemas como la detección de anomalías sutiles, la identificación de ataques desconocidos o la anticipación de fallos progresivos. Gracias a estas capacidades, la IA se ha convertido en un componente fundamental de los sistemas modernos de supervisión y seguridad de redes [9], [22].

| Característica | Sistemas Tradicionales | Sistemas Basados en IA |
|---|---|---|
| **Enfoque** | Reactivo: se actúa después de que ocurre un fallo o problema. | Proactivo: predice problemas y optimiza la red en tiempo real. |
| **Automatización** | Limitada: scripts y reglas estáticas. | Alta: aprendizaje automático, análisis predictivo y autorreparación. |
| **Monitoreo** | Básico: métricas tradicionales como latencia, tráfico, disponibilidad. | Avanzado: correlación de múltiples métricas, detección de anomalías y comportamiento anómalo. |
| **Toma de decisiones** | Humana: el operador interviene según protocolos predefinidos. | Automatizada e inteligente: recomendaciones y acciones autónomas basadas en datos. |
| **Escalabilidad** | Escasa: requiere más personal para redes grandes y complejas. | Alta: adapta recursos y optimiza la red automáticamente con gran escala. |
| **Mantenimiento** | Manual y programado: actualizaciones y reparaciones según cronograma. | Predictivo: mantenimiento preventivo basado en análisis de tendencias y patrones. |
| **Eficiencia operativa** | Media: dependiente de experiencia humana y tiempos de reacción. | Alta: optimización continua, reducción de errores y downtime. |
| **Seguridad** | Basada en reglas estáticas y firewalls tradicionales. | Dinámica: detección de amenazas en tiempo real, respuesta automática ante anomalías. |

*Tabla 3 Comparación entre sistemas tradicionales de gestión de red y sistemas basados en IA*

### 2.3.2 IA APLICADA A LA GESTIÓN DE REDES

La aplicación de técnicas de Inteligencia Artificial en la gestión de redes surge como respuesta a la creciente complejidad de las infraestructuras de comunicaciones y al volumen masivo de datos que estas generan. Las redes modernas producen continuamente métricas relacionadas con tráfico, latencia, pérdida de paquetes, uso de recursos o eventos de seguridad, cuya interpretación manual o mediante reglas fijas resulta inviable en entornos reales de gran escala [5], [9].

A diferencia de los enfoques tradicionales, la IA permite **aprender automáticamente patrones de comportamiento normal de la red** a partir de los datos históricos y detectar desviaciones sin necesidad de definir reglas explícitas. Este cambio de paradigma posibilita una supervisión más flexible, capaz de adaptarse a variaciones dinámicas del tráfico y a cambios en la topología o en la carga de trabajo [3], [4].

Uno de los principales aportes de la IA a la gestión de redes es la **automatización del análisis**. Los modelos de Machine Learning y Deep Learning pueden procesar grandes volúmenes de métricas en tiempo real, identificar correlaciones complejas entre múltiples variables y generar alertas de forma autónoma ante comportamientos anómalos. Esto reduce significativamente la dependencia de la intervención humana y acelera los tiempos de respuesta frente a incidentes [5], [6].

Además, la IA introduce la capacidad de **adaptación continua**. Los modelos pueden reentrenarse o ajustarse a medida que evoluciona el comportamiento de la red, evitando el envejecimiento de las reglas estáticas y mejorando la robustez frente a nuevos patrones de tráfico o amenazas emergentes [9], [30]. Esta característica resulta especialmente relevante en escenarios donde la red experimenta cambios frecuentes, como entornos cloud, IoT o redes definidas por software.

Otro aspecto clave es la **capacidad predictiva**. Mientras que los sistemas tradicionales se limitan a reaccionar ante fallos ya ocurridos, los modelos basados en IA pueden anticipar degradaciones del rendimiento o fallos incipientes analizando tendencias temporales en las métricas de red. Técnicas como redes neuronales recurrentes o modelos predictivos permiten detectar señales tempranas de congestión, saturación de enlaces o degradación progresiva del servicio [19], [31].

En conjunto, la incorporación de Inteligencia Artificial en la gestión de redes permite avanzar desde un enfoque reactivo hacia un modelo **proactivo y predictivo**, en el que la detección, la anticipación y la toma de decisiones automáticas se integran como elementos centrales del sistema de supervisión. Este enfoque constituye la base de los sistemas modernos de detección de anomalías y predicción de fallos en redes de comunicaciones [5], [9].

## *2.4 FUNDAMENTOS DE MACHINE LEARNING*

El *Machine Learning* (aprendizaje automático) es una rama de la Inteligencia Artificial centrada en el desarrollo de algoritmos capaces de aprender patrones a partir de datos y mejorar su rendimiento de forma automática sin necesidad de ser programados explícitamente para cada situación concreta [28], [29]. A diferencia de los sistemas tradicionales, donde el comportamiento del sistema está definido por reglas fijas, los modelos de Machine Learning construyen representaciones internas a partir de ejemplos y experiencia previa.

Formalmente, un sistema de Machine Learning puede definirse como aquel que mejora su desempeño en una tarea determinada mediante la experiencia, entendida como el análisis de datos históricos o en tiempo real [5]. Esta capacidad resulta especialmente relevante en entornos complejos y dinámicos, como las redes de comunicaciones, donde los patrones de comportamiento no son estáticos y evolucionan constantemente.

En el ámbito de las redes, el Machine Learning permite abordar problemas que resultan difíciles de modelar mediante técnicas clásicas, como la detección de anomalías, la clasificación de tráfico o la identificación de comportamientos anómalos que no siguen reglas predefinidas [5]. En lugar de establecer umbrales manuales o condiciones rígidas, los modelos aprenden qué se considera comportamiento normal y detectan desviaciones significativas respecto a dicho patrón.

Una de las principales características del Machine Learning es su capacidad para manejar grandes volúmenes de datos multidimensionales. Las redes modernas generan una enorme cantidad de información (latencia, pérdida de paquetes, tasas de tráfico, uso de CPU, métricas de flujo, entre otras) cuya interpretación manual resulta inviable [3]. Los algoritmos de aprendizaje automático permiten analizar estas métricas de forma conjunta, capturando relaciones no lineales y dependencias ocultas entre ellas.

Además, el Machine Learning constituye la base sobre la que se desarrollan enfoques más avanzados como el *Deep Learning*, que se abordará en secciones posteriores. Mientras que los modelos clásicos de aprendizaje automático suelen requerir una selección manual de características, las técnicas profundas permiten extraer representaciones más complejas directamente de los datos, lo que amplía aún más su aplicabilidad en redes de comunicaciones [32], [33].

En resumen, el Machine Learning proporciona el marco conceptual y técnico necesario para dotar a los sistemas de gestión de redes de capacidades de aprendizaje, adaptación y generalización, sentando las bases para la detección inteligente y la predicción de anomalías en entornos de red complejos y dinámicos.

### 2.4.1 TIPOS DE APRENDIZAJE AUTOMÁTICO

El aprendizaje automático puede clasificarse en función de la forma en que los modelos utilizan los datos disponibles durante el entrenamiento. Esta clasificación es especialmente relevante en el ámbito de las redes de comunicaciones, donde la disponibilidad de datos etiquetados suele ser limitada y los patrones de comportamiento pueden variar con el tiempo.

#### *2.4.1.1 Aprendizaje supervisado*

En el aprendizaje supervisado, el modelo se entrena utilizando datos etiquetados, es decir, cada muestra de entrada está asociada a una salida conocida. El objetivo es aprender una función que permita predecir correctamente la etiqueta de nuevos datos no vistos [28].

En redes de comunicaciones, este enfoque se utiliza principalmente para la clasificación de tráfico y la detección de ataques conocidos, siempre que se disponga de ejemplos previamente identificados [9].

#### *2.4.1.2 Aprendizaje no supervisado*

El aprendizaje no supervisado se basa en datos sin etiquetar. El modelo identifica patrones, agrupaciones o desviaciones respecto al comportamiento habitual sin conocimiento previo de las clases existentes [8].

Este tipo de aprendizaje es especialmente adecuado para la detección de anomalías en redes, ya que permite identificar comportamientos anómalos sin necesidad de disponer de ejemplos de fallos o ataques previamente etiquetados [5].

#### *2.4.1.3 Aprendizaje semisupervisado*

El aprendizaje semisupervisado combina un pequeño conjunto de datos etiquetados con una gran cantidad de datos sin etiquetar. El modelo aprovecha la estructura de los datos no etiquetados para mejorar el rendimiento de la clasificación o detección.

En entornos de red, este enfoque resulta útil cuando solo se dispone de información parcial sobre eventos anómalos o incidentes de seguridad [34].

#### *2.4.1.4 Aprendizaje por refuerzo*

En el aprendizaje por refuerzo, un agente interactúa con un entorno y aprende mediante un sistema de recompensas y penalizaciones. No se basa en datos etiquetados, sino en la evaluación de las acciones realizadas [28].

En redes, este enfoque se aplica principalmente a tareas de optimización y control, como la gestión dinámica de recursos o el enrutamiento adaptativo [3].

#### *2.4.1.5 Tabla comparativa de los tipos de aprendizaje automático*

| Tipo de aprendizaje | Datos etiquetados | Uso en redes |
|---|---|---|
| Supervisado | Si | Clasificación de trafico y ataques conocidos |
| No supervisado | No | Detección de anomalías y apatrones inusuales |
| Semisupervisado | Parcial | Identificación de eventos poco etiquetados |
| Por refuerzo | No | Optimización y control de red |

*Tabla 4 Clasificación de los tipos de aprendizaje automático según el uso en redes*

### 2.4.2 MACHINE LEARNING EN ENTORNOS DE RED

El uso de técnicas de *Machine Learning* en entornos de red surge como respuesta a la creciente complejidad y dinamismo de las infraestructuras de comunicaciones modernas. A diferencia de los enfoques tradicionales, estos métodos permiten analizar grandes volúmenes de datos operativos y extraer patrones de comportamiento sin necesidad de definir reglas explícitas para cada situación posible [5], [9].

En el ámbito de las redes, el *Machine Learning* se aplica principalmente al análisis del comportamiento del tráfico y del estado de los dispositivos, utilizando métricas como latencia, pérdida de paquetes, volumen de tráfico o uso de recursos. A partir de estos datos, los modelos son capaces de identificar patrones habituales de funcionamiento y detectar desviaciones significativas que pueden indicar la presencia de anomalías, fallos incipientes o eventos de seguridad [3], [22].

Una de las principales ventajas del *Machine Learning* en entornos de red es su capacidad para adaptarse a cambios progresivos en el comportamiento del sistema. Las redes no presentan un estado estático, sino que evolucionan en función del número de usuarios, del tipo de aplicaciones y de la carga temporal. Los modelos de aprendizaje automático permiten capturar estas variaciones y actualizar su conocimiento del comportamiento normal, reduciendo la dependencia de configuraciones manuales y umbrales fijos [4], [21].

Además, estas técnicas facilitan el análisis conjunto de múltiples métricas de forma simultánea. Mientras que los métodos clásicos suelen evaluar cada parámetro de manera aislada, los modelos de *Machine Learning* pueden aprender relaciones complejas entre variables, lo que mejora la detección de comportamientos anómalos que no serían evidentes al observar una única métrica [5], [9].

En contextos operativos, el *Machine Learning* se emplea tanto en tareas de detección como de clasificación. En función del enfoque adoptado, los modelos pueden identificar instantes en los que el comportamiento de la red se desvía de lo esperado o categorizar diferentes tipos de tráfico y eventos. Esta capacidad resulta especialmente relevante en escenarios donde no se dispone de datos etiquetados de forma exhaustiva, situación habitual en redes reales [22], [25].

Por último, el uso de *Machine Learning* contribuye a avanzar hacia una gestión de red más proactiva y escalable. Al automatizar el análisis de grandes cantidades de datos y reducir la intervención manual, estos enfoques permiten acortar los tiempos de reacción ante incidencias y sentar las bases para sistemas de supervisión más inteligentes y predictivos, adecuados para redes de gran tamaño y alta complejidad [6], [9].




*Ilustración 10 Esquema del uso de Machine Learning en un entorno de red. Fuente: S. Agrahari [35].*

## 2.5 INTRODUCCIÓN AL DEEP LEARNING

El crecimiento en la complejidad de las redes de comunicaciones y en la cantidad de datos generados por su funcionamiento ha puesto de manifiesto las limitaciones de muchos enfoques clásicos de análisis basados en Machine Learning tradicional. En este contexto, el **Deep Learning** surge como una evolución natural del aprendizaje automático, orientada al tratamiento eficiente de grandes volúmenes de datos y a la extracción automática de representaciones complejas a partir de información de alta dimensionalidad [32], [33].

El Deep Learning se basa en el uso de **redes neuronales profundas**, compuestas por múltiples capas de procesamiento no lineal, que permiten modelar relaciones complejas entre las variables de entrada y capturar patrones de comportamiento que resultan difíciles de describir mediante técnicas manuales o modelos poco profundos [32], [33]. Esta capacidad resulta especialmente relevante en entornos de red, donde el tráfico, las métricas

de rendimiento y los eventos operativos presentan una naturaleza dinámica, temporal y altamente no lineal.

A diferencia de otros enfoques analíticos, los modelos de Deep Learning permiten **aprender directamente a partir de los datos sin necesidad de una ingeniería de características exhaustiva**, lo que reduce la dependencia de conocimiento experto previo sobre el dominio de la red [25], [32]. Esto los convierte en una herramienta adecuada para escenarios donde el comportamiento normal del sistema evoluciona con el tiempo o varía en función del contexto operativo.

En el ámbito de la gestión y monitorización de redes, el Deep Learning ha demostrado ser especialmente útil para tareas como la **detección de anomalías complejas**, el análisis de series temporales de métricas de red y la **predicción de fallos o degradaciones del servicio**, gracias a su capacidad para capturar dependencias a largo plazo y patrones ocultos en los datos [9], [31].

No obstante, el uso de modelos profundos implica también mayores requisitos computacionales y una mayor complejidad en el proceso de entrenamiento, lo que hace necesario un equilibrio entre capacidad predictiva, interpretabilidad y coste operativo. Por este motivo, su aplicación en entornos de red suele centrarse en escenarios donde el volumen y la complejidad de los datos justifican el uso de técnicas avanzadas de aprendizaje profundo [32], [33].

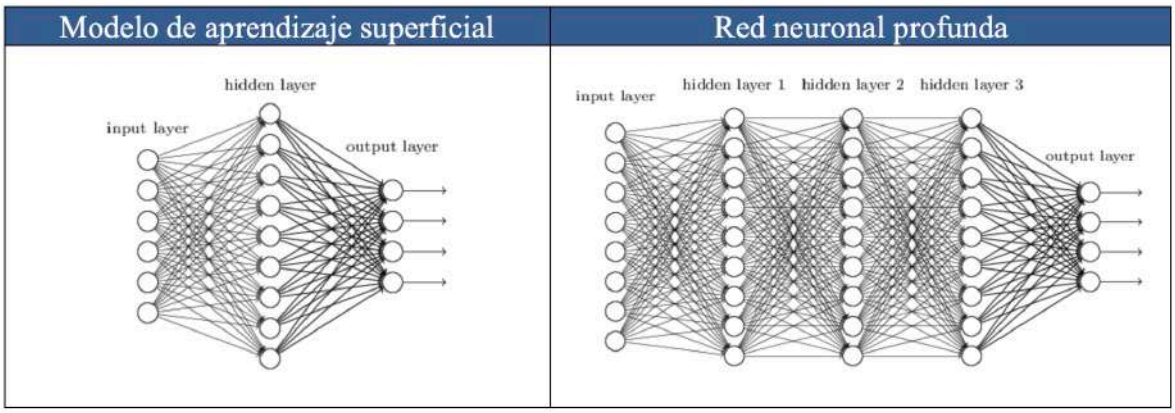


*Tabla 5 Diferencia entre red neuronal simple y profunda esquema. Fuente: Innovación Digital 360 [36].*

### 2.5.1 DIFERENCIA CON EL MACHINE LEARNING CLÁSICO

El Deep Learning se diferencia del Machine Learning clásico principalmente en la forma en que los modelos procesan y representan la información a partir de los datos. Mientras que los algoritmos clásicos de aprendizaje automático dependen en gran medida de una selección manual de características, las técnicas de Deep Learning son capaces de aprender representaciones jerárquicas y abstractas directamente a partir de los datos de entrada [32], [33].

En los modelos tradicionales de Machine Learning, como máquinas de vectores de soporte, árboles de decisión o métodos de clustering, el rendimiento del sistema está fuertemente condicionado por la calidad de las características extraídas previamente. En el contexto de las redes de comunicaciones, esto implica que el analista debe definir de forma explícita qué métricas, combinaciones o estadísticas son relevantes para describir el comportamiento de la red [3], [5]. Esta etapa de ingeniería de características puede resultar compleja y dependiente del conocimiento experto.

Por el contrario, los modelos de Deep Learning utilizan arquitecturas basadas en múltiples capas neuronales que permiten aprender automáticamente representaciones cada vez más complejas de los datos. Las capas iniciales capturan patrones simples, mientras que las capas más profundas modelan relaciones no lineales y dependencias de mayor nivel, sin necesidad de intervención manual directa [32], [33]. Esta capacidad resulta especialmente ventajosa en entornos donde los datos presentan alta dimensionalidad y relaciones complejas entre variables, como ocurre en las métricas de red.

Otra diferencia relevante reside en la capacidad de los modelos para manejar información temporal y secuencial. Mientras que muchos algoritmos clásicos tratan las observaciones de forma independiente, las arquitecturas profundas, especialmente las redes neuronales recurrentes, permiten capturar dependencias temporales a largo plazo, lo que las hace más adecuadas para la predicción de comportamientos futuros en redes de comunicaciones [19], [31].

No obstante, el Deep Learning también presenta mayores requisitos computacionales y de datos en comparación con el Machine Learning clásico. El entrenamiento de modelos profundos suele requerir mayores volúmenes de datos y mayor capacidad de procesamiento, lo que puede limitar su aplicación en ciertos escenarios [32]. Por este motivo, en la práctica resulta habitual combinar enfoques clásicos y profundos, seleccionando el modelo más adecuado en función del tipo de problema y de los datos disponibles.

En resumen, mientras que el Machine Learning clásico ofrece soluciones eficaces y más interpretables para problemas bien definidos, el Deep Learning proporciona una mayor capacidad de modelado en escenarios complejos y dinámicos, lo que lo convierte en una herramienta especialmente adecuada para la detección y predicción de anomalías en redes modernas.

| Característica | Machine Learning Clásico | Deep Learning |
|---|---|---|
| **Capacidad de modelado** | **Limitada:** modelos lineales o con pocas capas, difícil de capturar relaciones complejas | **Alta**: capaz de capturar patrones complejos y no lineales mediante múltiples capas |
| **Dependencia de características manuales** | **Alta**: requiere ingeniería de características (feature engineering) | **Baja**: aprende representaciones automáticamente a partir de los datos crudos |
| **Escalabilidad** | **Media**: se complica con grandes volúmenes de datos y muchas variables | **Alta**: escalable a grandes volúmenes y múltiples métricas de red |
| **Análisis temporal** | **Limitado**: modelos clásicos no suelen capturar dependencias temporales largas | **Avanzado**: modelos recurrentes (RNN, LSTM) o secuenciales capturan relaciones temporales |
| **Robustez ante cambios** | **Baja/Media**: sensible a cambios en la distribución de datos | **Alta**: puede adaptarse a cambios y aprender nuevas representaciones con más datos |

*Tabla 6 Comparación entre Machine Learning clásico y Deep Learning en entornos de red*

### 2.5.2 VENTAJAS EN REDES COMPLEJAS

Las redes de comunicaciones modernas presentan un elevado grado de complejidad derivado de su tamaño, heterogeneidad y comportamiento dinámico. En estos entornos, las relaciones entre métricas de red suelen ser no lineales, altamente dependientes del contexto y variables en el tiempo, lo que dificulta su modelado mediante técnicas tradicionales o modelos de Machine Learning clásico [5], [9].

El Deep Learning ofrece ventajas significativas para abordar este tipo de escenarios complejos, gracias a su capacidad para aprender representaciones jerárquicas y abstractas directamente a partir de los datos. A diferencia de los modelos clásicos, que dependen en gran medida de la selección manual de características, las redes neuronales profundas pueden extraer automáticamente patrones relevantes a distintos niveles de granularidad, capturando interacciones complejas entre múltiples métricas de red [32], [33].

Una de las principales ventajas del Deep Learning en redes complejas es su capacidad para modelar comportamientos no lineales. El tráfico de red, la latencia, la pérdida de paquetes o el uso de recursos no siguen distribuciones simples ni relaciones directas, especialmente en entornos congestionados o bajo condiciones anómalas. Las arquitecturas profundas permiten representar este tipo de relaciones complejas de forma más precisa que los modelos lineales o basados en reglas [32], [33].

Otra ventaja clave es la capacidad de adaptación a grandes volúmenes de datos. Las redes modernas generan información de forma continua y a alta frecuencia, lo que exige modelos capaces de escalar y aprender a partir de grandes conjuntos de datos multidimensionales. El Deep Learning está especialmente diseñado para este tipo de escenarios, manteniendo su rendimiento incluso cuando aumenta la cantidad y diversidad de las métricas analizadas [32], [33].

Asimismo, las técnicas de Deep Learning resultan especialmente adecuadas para el análisis temporal del comportamiento de la red. Muchas anomalías y fallos no se manifiestan de forma instantánea, sino como cambios progresivos en el tiempo. Los modelos profundos permiten capturar dependencias temporales y patrones de evolución que resultan difíciles de

detectar mediante enfoques tradicionales, lo que facilita la detección temprana y la predicción de degradaciones del servicio [19], [31].

Finalmente, el Deep Learning contribuye a una mayor robustez frente a escenarios desconocidos. Al no depender de reglas explícitas ni de patrones previamente definidos, estos modelos pueden detectar comportamientos anómalos no vistos durante el entrenamiento, lo que resulta especialmente relevante en redes dinámicas donde aparecen constantemente nuevos servicios, dispositivos o tipos de tráfico [22], [32].

En conjunto, estas ventajas hacen que el Deep Learning sea una herramienta especialmente adecuada para la gestión y supervisión de redes complejas, proporcionando una base sólida para sistemas de detección y predicción de anomalías capaces de operar de forma automática, adaptable y escalable en entornos de comunicaciones modernos [5], [32], [33].

## *2.6 DATOS Y MÉTRICAS PARA DETECCIÓN DE ANOMALÍAS*

En la detección de anomalías en redes, los datos y las métricas juegan un papel fundamental. Estos indicadores permiten caracterizar el comportamiento de la red y detectar desviaciones respecto al funcionamiento normal [5][17].

### 2.6.1 MÉTRICAS DE RED UTILIZADAS

A continuación, se describen las métricas más relevantes para el análisis de anomalías y fallos en redes de comunicaciones:

- **Latencia**: Tiempo que tarda un paquete en viajar desde el origen hasta el destino. La latencia elevada o variaciones súbitas pueden indicar congestión, fallos de enlace o ataques de denegación de servicio [5][17].
- **Pérdida de paquetes**: Porcentaje de paquetes que no llegan a su destino. La pérdida persistente puede reflejar problemas físicos en la red, saturación de enlaces o comportamiento malicioso [5][17].
- **Throughput**: Cantidad de datos transmitidos correctamente por unidad de tiempo. Reducciones inesperadas en throughput pueden señalar congestión o fallos de infraestructura [5][17].
- **Uso de CPU**: Carga de procesamiento en los dispositivos de red. Un uso excesivo puede comprometer la capacidad de la red para manejar tráfico y puede indicar ataques o anomalías en los flujos de datos [5][17].

Estas métricas pueden obtenerse tanto de entornos simulados como Cisco Packet Tracer, donde es posible generar trazas controladas para evaluar el rendimiento de los modelos de detección de anomalías.

### 2.6.2 REPRESENTACIÓN TEMPORAL MEDIANTE VENTANAS DESLIZANTES

Las métricas de red no son observaciones independientes entre sí, sino que presentan dependencias temporales: el valor de la latencia en el instante t está relacionado con su

evolución en los instantes t−1, t−2, ..., t−k. Para que los modelos de aprendizaje automático puedan explotar estas dependencias, es necesario transformar la serie temporal de métricas en una representación estructurada que preserve el orden y la continuidad temporal de las observaciones [37].

La técnica estándar para este propósito es la **ventana deslizante temporal** (*sliding window*), que consiste en construir muestras de entrenamiento tomando *w* observaciones consecutivas de la serie temporal como vector de entrada del modelo. La ventana se desplaza un paso hacia adelante en cada iteración, generando tantos vectores como muestras contiene la serie menos el tamaño de ventana. Formalmente, dado un conjunto de T observaciones de d variables, una ventana de tamaño w produce T−w+1 vectores de dimensión w×d, cada uno representando un segmento temporal de la red [37].

En el presente trabajo se utilizan ventanas de tamaño **w = 20 pasos temporales** y **d = 5 features** (throughput, congestión, pérdida de paquetes, latencia y jitter), produciendo vectores de entrada de dimensión 100. Este tamaño de ventana captura aproximadamente 560 segundos de historia de la red (~9.3 minutos a un intervalo de muestreo de ~28 segundos), suficiente para que tanto el Autoencoder como la LSTM puedan identificar el contexto temporal previo a los eventos anómalos. La elección de w = 20 responde al equilibrio entre la capacidad de captura de dependencias temporales y el coste computacional del entrenamiento [31], [37]. Para validar esta elección, se realizó un estudio de sensibilidad evaluando valores de w ∈ {5, 10, 20, 30, 40} sobre la Anomalía 1 (congestión), representativa por ser la de mayor duración en el dataset. Los resultados muestran que w = 5 y w = 10 producen una degradación notable del F1-Score del Autoencoder (caída de hasta 0.12 puntos respecto a w = 20), al no capturar el patrón de rampa previa al pico de congestión. Por el contrario, w = 30 y w = 40 no mejoran el F1-Score e introducen un efecto de dilución creciente en anomalías de tipo spike puntual (Anomalías 2 y 4), donde la ventana contiene proporcionalmente más pasos normales que anómalos, reduciendo la señal del error de reconstrucción. El valor w = 20 maximiza el F1-Score medio ponderado entre las cuatro anomalías y representa el único valor del rango explorado que equilibra adecuadamente las anomalías graduales (congestión, latencia) y las impulsivas (packet loss, jitter).

### 2.6.3 NORMALIZACIÓN DE DATOS: MINMAXSCALER

Las métricas de red presentan rangos de magnitud heterogéneos: el throughput se mide en Mbps, la latencia en milisegundos, la pérdida de paquetes en porcentaje y el índice de congestión en unidades adimensionales. Esta heterogeneidad puede perjudicar el proceso de entrenamiento de los modelos de aprendizaje automático, ya que las variables con mayor magnitud tienden a dominar la función de coste e introducen sesgos en el aprendizaje. La normalización es, por tanto, un paso de preprocesamiento esencial antes de aplicar cualquier modelo numérico [38], [39].

La técnica de normalización empleada en este trabajo es la **escala mínimo-máximo** (*MinMaxScaler*), que transforma cada variable al intervalo [0, 1] mediante la expresión x' = (x − x_min) / (x_max − x_min), donde x_min y x_max son el mínimo y el

máximo de la variable en el conjunto de ajuste [38]. Una decisión metodológica crítica en el contexto de la detección de anomalías es que el ajuste del escalado se realiza **exclusivamente sobre las muestras normales** del conjunto de entrenamiento, sin incluir las muestras anómalas. De este modo, los valores anómalos se proyectan fuera del rango [0, 1] al aplicar el escalador, produciendo una señal de fuera de distribución que los modelos de detección pueden explotar. Esta estrategia garantiza que la escala de normalización representa fielmente el comportamiento esperado del sistema, convirtiendo los valores anómalos en extrapolaciones observables.

### 2.6.4 MÉTRICAS DE EVALUACIÓN DE MODELOS DE DETECCIÓN

La evaluación del rendimiento de los modelos de detección y predicción de anomalías requiere métricas adaptadas al contexto de **clases desbalanceadas**, donde la clase anómala representa una fracción pequeña del total de muestras (aproximadamente un 9:1 en el dataset utilizado). En estos escenarios, la exactitud global (*accuracy*) resulta una métrica engañosa: un modelo que siempre predice la clase mayoritaria (normal) obtendría una *accuracy* del 90% sin detectar ninguna anomalía real. Por este motivo, se emplean métricas que evalúan específicamente el comportamiento del modelo sobre la clase minoritaria [38].

La **precisión** (*Precision*) mide la fracción de alertas generadas por el modelo que corresponden a anomalías reales: Precision = TP / (TP + FP). Un valor alto de precisión indica que el modelo produce pocas falsas alarmas. El **recall**(también denominado sensibilidad o tasa de verdaderos positivos) mide la fracción de anomalías reales que el modelo detecta: Recall = TP / (TP + FN). Un valor alto de recall indica que el modelo no pierde eventos anómalos relevantes. En sistemas de monitorización de redes, un falso negativo (anomalía no detectada) es generalmente más costoso que un falso positivo (alerta espuria), lo que justifica prestar especial atención al recall [6], [39].

El **F1-Score** es la media harmónica entre precisión y recall: F1 = 2 · (Precision · Recall) / (Precision + Recall). Esta métrica proporciona una valoración equilibrada del rendimiento que penaliza por igual la baja precisión y el bajo recall, resultando especialmente adecuada para comparar modelos en escenarios con clases desbalanceadas [40]. Finalmente, el **área bajo la curva ROC** (AUC-ROC) cuantifica la capacidad discriminativa del modelo de forma independiente del umbral de decisión: integra la tasa de verdaderos positivos frente a la tasa de falsos positivos para todos los umbrales posibles. Un valor de AUC = 1.0 indica separación perfecta entre clases; AUC = 0.5 equivale a clasificación aleatoria [39]. En el presente trabajo, el F1-Score con umbral óptimo es la métrica principal de comparación entre modelos, complementada por el AUC-ROC como indicador de la capacidad discriminativa intrínseca.

## *2.7 HERRAMIENTAS SOFTWARE PARA IA*

El desarrollo de sistemas de detección de anomalías basados en Inteligencia Artificial requiere herramientas software que permitan implementar, entrenar y evaluar modelos de aprendizaje automático de forma eficiente y reproducible. En este proyecto se emplean

bibliotecas ampliamente utilizadas en el ámbito académico y profesional, que facilitan la experimentación con distintos algoritmos y el análisis de resultados.

### 2.7.1 SCIKIT-LEARN

Scikit-learn es una biblioteca de código abierto para aprendizaje automático en Python, diseñada para facilitar la implementación de algoritmos clásicos de Machine Learning mediante una interfaz sencilla y consistente. Está orientada principalmente al análisis de datos estructurados y es ampliamente utilizada en tareas de clasificación, regresión, clustering y detección de anomalías [37], [39].

En el contexto de la gestión y monitorización de redes, Scikit-learn resulta especialmente útil para:

- entrenar modelos de detección de anomalías a partir de métricas de red,
- comparar diferentes algoritmos bajo un mismo conjunto de datos,
- evaluar el rendimiento mediante métricas estandarizadas.

La biblioteca incluye implementaciones eficientes de numerosos modelos relevantes para este proyecto, entre ellos:

- Support Vector Machines (SVM),
- Random Forest,
- K-means,
- Isolation Forest.

Además, proporciona herramientas para el preprocesamiento de datos, la validación cruzada y la selección de hiperparámetros, lo que permite construir flujos de trabajo reproducibles y bien estructurados para el análisis de anomalías en redes [39].

### 2.7.2 TENSORFLOW Y KERAS

El desarrollo de modelos de *Deep Learning* aplicados a la detección y predicción de anomalías en redes requiere herramientas capaces de manejar arquitecturas neuronales complejas y grandes volúmenes de datos. En este contexto, **TensorFlow** y **Keras** se han consolidado como plataformas de referencia para la implementación de sistemas de aprendizaje profundo [40], [41].

TensorFlow es un framework de código abierto orientado al diseño, entrenamiento y despliegue de modelos de *Deep Learning*. Permite definir grafos computacionales eficientes y optimizar el procesamiento de datos numéricos, lo que resulta especialmente adecuado para el análisis de métricas de red con alta frecuencia temporal [40], [41]. Su capacidad para escalar y su compatibilidad con distintos entornos de ejecución lo convierten en una herramienta ampliamente utilizada tanto en investigación como en aplicaciones industriales.

Por su parte, **Keras** actúa como una interfaz de alto nivel sobre TensorFlow, facilitando la construcción de redes neuronales mediante una sintaxis más intuitiva y modular [41]. Esta abstracción permite centrarse en el diseño del modelo sin necesidad de gestionar detalles de

bajo nivel, reduciendo la complejidad del desarrollo y acelerando el proceso de experimentación.

En el ámbito de la detección de anomalías en redes, TensorFlow y Keras resultan especialmente adecuados para la implementación de **Autoencoders** y **redes LSTM**, dos arquitecturas clave en este proyecto. Los Autoencoders permiten modelar el comportamiento normal del tráfico de red y detectar anomalías a partir del error de reconstrucción, mientras que las redes LSTM son especialmente eficaces para analizar la evolución temporal de métricas y realizar predicción de fallos en series temporales [25], [31].

La combinación de TensorFlow y Keras proporciona, por tanto, un entorno flexible y potente para desarrollar modelos avanzados de *Deep Learning* orientados a la detección, predicción y análisis de anomalías en redes de comunicaciones, manteniendo un equilibrio entre capacidad técnica y facilidad de uso.

### 2.7.3 REPRESENTACIÓN VECTORIAL DE TEXTO: TF-IDF

La validación del sistema de detección mediante logs de sistema requiere transformar los mensajes textuales en representaciones numéricas sobre las que aplicar modelos de aprendizaje automático. La técnica estándar para esta tarea es la **frecuencia de término–frecuencia inversa de documento** (TF-IDF, *Term Frequency–Inverse Document Frequency*), propuesta originalmente por Sparck Jones [42] y ampliamente utilizada en el procesamiento de lenguaje natural y el análisis de logs de sistema [44], [50].

El TF-IDF combina dos factores: la frecuencia de un término en un documento concreto (TF) y la frecuencia inversa de ese término en el conjunto completo de documentos (IDF). El TF mide la relevancia local de un término: cuanto más veces aparece en un mensaje, más relevante es para describir ese mensaje. El IDF penaliza los términos que aparecen en casi todos los documentos (como preposiciones o palabras genéricas, poco informativas), potenciando los términos raros y específicos, como los mnemonics de anomalía en los logs de Cisco IOS XR, que son precisamente los más discriminativos para distinguir un log normal de uno anómalo [5].

En el contexto del presente trabajo, se aplica TF-IDF sobre los 1001 logs generados en formato Cisco IOS XR, extrayendo un vocabulario de 50 términos (unigramas y bigramas) y produciendo un vector de 50 dimensiones por log. El modelo de detección de anomalías (Isolation Forest) se entrena sobre los vectores TF-IDF de los 721 logs normales y se evalúa sobre el conjunto completo, siguiendo el mismo principio de detección por desviación de la normalidad aprendida que el Autoencoder sobre las series temporales numéricas.

# Capítulo 3. ESTADO DE LA CUESTIÓN

## *3.1 DETECCIÓN DE ANOMALÍAS EN REDES*

La detección de anomalías en redes de comunicaciones se ha convertido en un elemento central de la gestión moderna de infraestructuras digitales, debido al crecimiento de la complejidad, la escala y la naturaleza dinámica del tráfico de red. En este contexto, identificar de forma temprana desviaciones respecto al comportamiento normal resulta esencial para garantizar la calidad del servicio, la disponibilidad y la seguridad de la red [3], [5].

A diferencia de los enfoques clásicos centrados exclusivamente en la detección de fallos explícitos o ataques conocidos, la detección de anomalías se basa en el análisis del comportamiento global de la red. Su objetivo principal es identificar patrones atípicos en métricas de tráfico, rendimiento o uso de recursos que puedan indicar degradaciones progresivas, configuraciones incorrectas o eventos de seguridad aún no materializados [5], [8].

Desde un punto de vista operativo, una anomalía puede manifestarse de múltiples formas, como variaciones inesperadas en la latencia, cambios en la distribución de flujos, incrementos anormales de tráfico o alteraciones en los patrones temporales habituales. Estas desviaciones no siempre tienen una causa inmediata evidente, pero suelen representar señales tempranas de problemas subyacentes que, de no ser detectados a tiempo, pueden derivar en fallos críticos o interrupciones del servicio [3], [20].

La detección de anomalías plantea importantes desafíos técnicos. El comportamiento normal de una red no es estático, sino que evoluciona con el tiempo en función de factores como la carga, el número de usuarios, las aplicaciones desplegadas o los cambios en la topología. Por este motivo, los sistemas de detección deben ser capaces de adaptarse a dichas variaciones y distinguir entre cambios legítimos y comportamientos potencialmente problemáticos [5], [9].

A lo largo de los años, se han propuesto diferentes enfoques para abordar este problema, que pueden clasificarse de forma general en tres grandes categorías: enfoques tradicionales basados en reglas y umbrales, enfoques basados en técnicas de Machine Learning y enfoques basados en Deep Learning. Cada uno de ellos presenta características, ventajas y limitaciones específicas, que serán analizadas en los apartados siguientes con el objetivo de contextualizar la selección de los modelos utilizados en este trabajo [22].

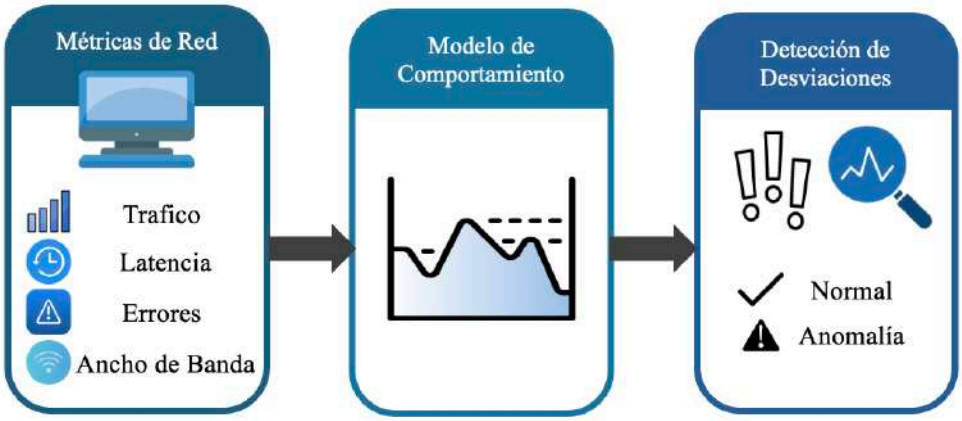


*Ilustración 11 Esquema general del proceso de detección de anomalías en redes*

### 3.1.1 ENFOQUES TRADICIONALES

Los enfoques tradicionales de detección de anomalías en redes de comunicaciones se basan principalmente en técnicas deterministas y heurísticas, desarrolladas en un contexto donde las redes presentaban menor escala, menor heterogeneidad y patrones de tráfico relativamente estables. Estos métodos han sido ampliamente utilizados en sistemas clásicos de monitorización y seguridad debido a su simplicidad de implementación y bajo coste computacional [5], [34].

Uno de los mecanismos más extendidos consiste en la **detección basada en umbrales**, donde se definen valores límite para determinadas métricas de red, como latencia, pérdida de paquetes o utilización de enlaces. Cuando una métrica supera el umbral establecido, el sistema genera una alerta indicando un posible problema. Este enfoque resulta eficaz para identificar fallos evidentes o sobrecargas abruptas, pero presenta importantes limitaciones frente a comportamientos anómalos complejos o graduales [4], [5].

Otro enfoque habitual es la **detección basada en reglas**, en la que se emplean conjuntos de condiciones definidas manualmente por expertos para identificar patrones conocidos de fallo o ataque. Este tipo de técnicas ha sido ampliamente utilizado en sistemas de detección de intrusiones basados en firmas, donde se comparan los eventos observados con patrones previamente definidos [7], [16]. Sin embargo, su eficacia depende directamente del conocimiento previo del problema, lo que limita su capacidad para detectar anomalías nuevas o desconocidas.

Desde el punto de vista estadístico, también se han empleado **modelos simples de referencia**, basados en promedios históricos, desviación estándar o análisis de tendencias básicas. En estos casos, una anomalía se identifica cuando el comportamiento observado se desvía significativamente de un modelo estadístico previamente calculado [17], [20]. Aunque estas técnicas introducen cierto grado de análisis cuantitativo, suelen asumir distribuciones estables y no capturan relaciones complejas entre múltiples métricas de red.

Una característica común de los enfoques tradicionales es su **carácter reactivo**. La detección suele producirse cuando el problema ya se ha manifestado de forma clara en el rendimiento o la disponibilidad del servicio, lo que limita la capacidad de anticipación y aumenta los tiempos de respuesta ante incidentes [5], [6]. Además, estos métodos suelen analizar métricas de forma aislada, sin considerar correlaciones temporales o dependencias entre distintos parámetros de la red.

En redes modernas, caracterizadas por tráfico dinámico, servicios distribuidos y patrones altamente variables, estas limitaciones reducen significativamente la efectividad de los enfoques tradicionales. Como consecuencia, surge la necesidad de técnicas más avanzadas capaces de modelar el comportamiento global de la red y adaptarse a cambios continuos, lo que ha motivado la evolución hacia enfoques basados en Machine Learning y Deep Learning, que se abordarán en los apartados siguientes [5], [22].

### 3.1.2 ENFOQUES BASADOS EN MACHINE LEARNING

El uso de técnicas de *Machine Learning* en la detección de anomalías en redes surge como una evolución natural frente a las limitaciones de los métodos tradicionales. En lugar de depender de reglas fijas o umbrales predefinidos, estos enfoques permiten modelar el comportamiento normal de la red a partir de los datos y detectar desviaciones de forma automática y adaptable [5], [9].

En el contexto de redes de comunicaciones, los enfoques basados en Machine Learning se apoyan principalmente en el análisis de métricas y flujos de red para aprender patrones característicos del funcionamiento habitual. A partir de este aprendizaje, el sistema es capaz de identificar comportamientos anómalos que no se ajustan a dichos patrones, incluso cuando estos no han sido observados previamente [3], [5].

Una de las principales ventajas de estos enfoques es su capacidad para manejar grandes volúmenes de datos multidimensionales. Las métricas de red suelen presentar relaciones complejas y no lineales que resultan difíciles de capturar mediante técnicas estadísticas simples. Los modelos de Machine Learning permiten analizar múltiples variables de forma conjunta, identificando correlaciones y estructuras ocultas que caracterizan el estado normal de la red [8], [22].

En función de la disponibilidad de datos etiquetados, los enfoques basados en Machine Learning pueden adoptar distintas estrategias. Los métodos supervisados se utilizan cuando se dispone de ejemplos previos de comportamientos normales y anómalos, lo que permite entrenar clasificadores capaces de distinguir entre ambos estados. Sin embargo, en entornos reales de red, la obtención de datos etiquetados resulta costosa y limitada, especialmente para anomalías poco frecuentes o ataques nuevos [9], [29].

Por este motivo, los enfoques no supervisados han adquirido un papel destacado en la detección de anomalías en redes. Estos métodos aprenden el comportamiento normal a partir de datos no etiquetados y detectan anomalías como desviaciones significativas respecto a dicho modelo. Esta característica los hace especialmente adecuados para escenarios

dinámicos, donde el comportamiento de la red evoluciona con el tiempo y no es posible definir todas las situaciones anómalas de antemano [8], [21].

Otra aportación relevante del Machine Learning es la reducción de la dependencia del conocimiento experto. Mientras que los sistemas tradicionales requieren una definición manual de reglas y umbrales, los modelos de aprendizaje automático ajustan sus parámetros de forma automática a partir de los datos, lo que facilita su adaptación a diferentes topologías, cargas de tráfico y condiciones de operación [5], [29].

No obstante, estos enfoques también presentan desafíos. El rendimiento de los modelos depende en gran medida de la calidad de los datos de entrada, del proceso de selección de características y de la correcta configuración de los parámetros del modelo. Además, muchos algoritmos de Machine Learning clásico carecen de mecanismos explícitos para modelar dependencias temporales complejas, lo que puede limitar su eficacia en la detección de anomalías que evolucionan progresivamente en el tiempo [21], [25].

En conjunto, los enfoques basados en Machine Learning representan un avance significativo respecto a las técnicas tradicionales de detección de anomalías, proporcionando mayor flexibilidad, adaptabilidad y capacidad de generalización. Estas características han motivado su amplia adopción en la literatura y han sentado las bases para el desarrollo de métodos más avanzados basados en Deep Learning, que se abordan en el apartado siguiente [5], [22].

### 3.1.3 ENFOQUES BASADOS EN DEEP LEARNING

Los enfoques basados en *Deep Learning* representan una evolución natural de las técnicas de detección de anomalías en redes, especialmente en escenarios caracterizados por grandes volúmenes de datos, alta dimensionalidad y patrones de comportamiento complejos. A diferencia de los métodos tradicionales y de muchos algoritmos de Machine Learning clásico, las técnicas de Deep Learning permiten aprender representaciones jerárquicas directamente a partir de los datos, reduciendo la necesidad de un diseño manual de características [32], [33].

En el contexto de las redes de comunicaciones, estos enfoques resultan especialmente adecuados para analizar métricas de red con fuerte dependencia temporal, correlaciones no lineales y comportamientos dinámicos que varían en función del tiempo y de la carga del sistema. Las redes neuronales profundas son capaces de modelar estas relaciones complejas y capturar patrones sutiles que pueden pasar desapercibidos para técnicas menos expresivas [25], [33].

Una de las principales ventajas del Deep Learning en la detección de anomalías es su capacidad para operar en entornos no supervisados o débilmente supervisados. Modelos como los autoencoders aprenden el comportamiento normal de la red durante la fase de entrenamiento y detectan anomalías como desviaciones significativas respecto a dicho comportamiento, sin necesidad de disponer de etiquetas explícitas de fallos o ataques [22], [25]. Este enfoque resulta especialmente útil en redes reales, donde la obtención de datos etiquetados es costosa y, en muchos casos, inviable.

Por otro lado, las arquitecturas recurrentes, como las redes LSTM, permiten incorporar explícitamente la dimensión temporal en el análisis, lo que facilita la detección de anomalías que se manifiestan de forma progresiva o a lo largo del tiempo. Estas redes son capaces de modelar dependencias a largo plazo en series temporales de métricas de red, lo que las convierte en una herramienta clave para la predicción de degradaciones de rendimiento y fallos incipientes [19], [31].

Desde el punto de vista operativo, los enfoques basados en Deep Learning ofrecen una mayor robustez frente a la variabilidad del tráfico y a la evolución de los patrones de uso de la red. A medida que el sistema se entrena con nuevos datos, los modelos pueden adaptarse a cambios estructurales en el comportamiento de la red, reduciendo el envejecimiento del modelo y la necesidad de redefinir reglas o parámetros manualmente [22], [25].

No obstante, estos enfoques también presentan desafíos relevantes, como un mayor coste computacional, la necesidad de volúmenes significativos de datos para el entrenamiento y una menor interpretabilidad de las decisiones del modelo en comparación con técnicas más simples [32], [33]. A pesar de estas limitaciones, el Deep Learning se ha consolidado como una de las estrategias más prometedoras para la detección y predicción de anomalías en redes complejas y de gran escala.

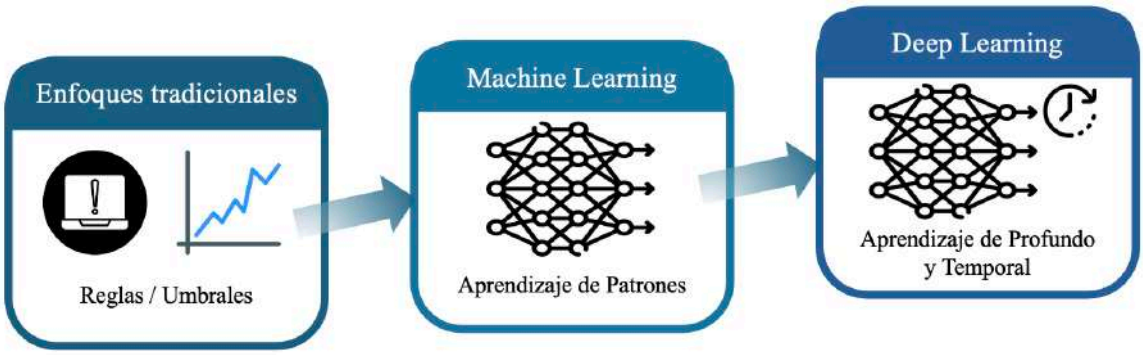


*Ilustración 12 Evolución de los enfoques de detección de anomalías en redes*

## *3.2 MODELOS DE MACHINE LEARNING PARA DETECCIÓN DE LAS ANOMALÍAS*

Una vez revisados los distintos enfoques generales para la detección de anomalías en redes, resulta necesario analizar los modelos concretos de *Machine Learning* que han sido ampliamente utilizados en la literatura para abordar este problema. Estos modelos permiten identificar comportamientos anómalos a partir del análisis de métricas de red, capturando patrones complejos que no pueden describirse mediante reglas estáticas o umbrales fijos [5], [22].

En el contexto de las redes de comunicaciones, los modelos de aprendizaje automático se aplican habitualmente sobre conjuntos de datos compuestos por métricas como tráfico, latencia, pérdida de paquetes o uso de recursos. A partir de estas variables, los algoritmos

aprenden representaciones del comportamiento normal de la red o separan regiones del espacio de características asociadas a comportamientos habituales y atípicos [3], [20].

Los modelos analizados en este apartado han sido seleccionados por su relevancia en la detección de anomalías en entornos de red, su amplia adopción en trabajos previos y su disponibilidad en herramientas de análisis estándar. En particular, se consideran modelos supervisados y no supervisados que permiten abordar distintos escenarios según la disponibilidad de datos etiquetados y la naturaleza del problema [21], [24].

En los siguientes subapartados se describen los principios de funcionamiento de cada uno de estos modelos, así como sus principales características y su aplicabilidad a la detección de anomalías en redes de comunicaciones [5], [22].

### 3.2.1 MÁQUINAS DE VECTORES DE SOPORTE

Las Máquinas de Vectores de Soporte (Support Vector Machines, SVM) constituyen uno de los modelos de aprendizaje automático más consolidados en tareas de clasificación y detección de patrones complejos. Desde su formulación original, las SVM han sido ampliamente utilizadas en distintos ámbitos de la ingeniería debido a su capacidad para manejar espacios de características de alta dimensionalidad y construir fronteras de decisión robustas incluso cuando el número de variables es elevado [45].

En el contexto de las redes de comunicaciones, la detección de anomalías implica analizar simultáneamente múltiples métricas operativas, como volumen de tráfico, latencia, pérdida de paquetes, jitter o duración de flujos, cuyas relaciones no suelen ser lineales ni triviales. Las SVM resultan especialmente adecuadas en este tipo de escenarios porque permiten modelar fronteras de decisión complejas y separar regiones del espacio de características asociadas a comportamientos normales y anómalos [9], [45].

Una de las principales razones de la relevancia de este modelo en entornos de red es su capacidad para generalizar adecuadamente a partir de datos históricos. Las SVM no se limitan a memorizar ejemplos concretos, sino que construyen una representación estructural del problema basada en aquellos puntos más informativos del conjunto de entrenamiento. Esta propiedad favorece una mayor robustez frente a ruido y variaciones habituales del tráfico, aspecto fundamental en sistemas de monitorización continua [45].

En aplicaciones prácticas de redes, las SVM se han utilizado para la identificación de intrusiones, la detección de ataques de denegación de servicio, el análisis de tráfico anómalo y la clasificación de flujos sospechosos. Asimismo, en entornos donde la disponibilidad de datos etiquetados es limitada, variantes específicas del modelo permiten abordar el problema desde una perspectiva no supervisada, modelando únicamente el comportamiento normal del sistema [9], [46].

Otra característica que explica su adopción en estudios académicos y comparativas experimentales es su solidez teórica. Las SVM se fundamentan en principios bien establecidos de optimización y teoría estadística del aprendizaje, lo que proporciona

garantías formales sobre su capacidad de generalización bajo determinadas condiciones [45]. Este respaldo teórico las convierte en un modelo de referencia habitual frente al cual se comparan enfoques más recientes, como técnicas basadas en Deep Learning.

En conjunto, las Máquinas de Vectores de Soporte representan una herramienta sólida para la detección de anomalías en redes de comunicaciones, especialmente en escenarios donde se requiere analizar múltiples métricas de forma conjunta y capturar relaciones complejas entre variables. Su equilibrio entre capacidad de modelado, robustez y fundamentación teórica explica su presencia recurrente en la literatura especializada.

#### *3.2.1.1 Fundamento conceptual del modelo*

El fundamento conceptual de las Máquinas de Vectores de Soporte se basa en la construcción de una frontera de decisión que maximice la separación entre clases dentro del espacio de características. A diferencia de otros modelos que simplemente buscan una separación válida entre datos, las SVM persiguen una separación óptima desde el punto de vista geométrico, lo que constituye uno de sus principales elementos distintivos [45].

##### 3.2.1.1.1 Margen máximo

La idea central del modelo es el **principio de margen máximo**. En un problema de clasificación binaria linealmente separable, pueden existir múltiples hiperplanos capaces de dividir correctamente los datos. Sin embargo, las SVM seleccionan aquel hiperplano que maximiza la distancia mínima entre la frontera de decisión y los puntos más cercanos de cada clase [45].

Estos puntos más cercanos reciben el nombre de *vectores de soporte*, ya que determinan de forma directa la posición del hiperplano óptimo. Al maximizar el margen, el modelo reduce el riesgo de sobreajuste y mejora su capacidad de generalización frente a nuevos datos no observados [45].

Desde una perspectiva intuitiva, cuanto mayor es el margen, mayor es la tolerancia del modelo ante pequeñas variaciones o perturbaciones en los datos, aspecto especialmente relevante en redes de comunicaciones, donde las métricas presentan fluctuaciones naturales debidas a la carga variable del sistema [45].

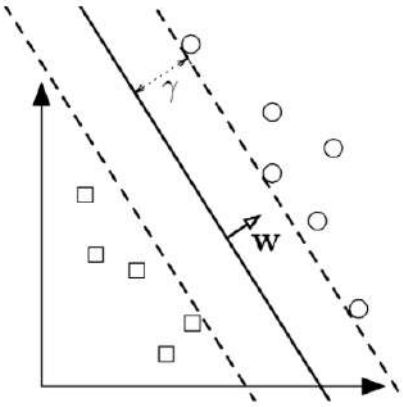


*Ilustración 13 Hiperplano de margen máximo. El margen, γ, es la distancia desde el hiperplano hasta los puntos más cercanos de cada clase. Fuente: Cornell University [48]*

#### 3.2.1.1.2 Separación mediante hiperplano

En términos geométricos, una SVM define un hiperplano dentro de un espacio de características multidimensional. Este hiperplano actúa como frontera de decisión, separando regiones asociadas a diferentes comportamientos (por ejemplo, tráfico normal frente a tráfico anómalo).

En espacios de dos dimensiones, la frontera es una recta; en tres dimensiones, un plano; y en dimensiones superiores, un hiperplano. La capacidad de trabajar en espacios de alta dimensionalidad convierte a las SVM en una herramienta especialmente adecuada cuando se analizan múltiples métricas de red simultáneamente, como latencia, throughput, tasa de paquetes y uso de recursos.

Cuando los datos no son perfectamente separables, el modelo permite cierta tolerancia al error mediante un mecanismo de regularización que equilibra el tamaño del margen con la penalización por clasificaciones incorrectas. Este equilibrio es clave para evitar tanto el sobreajuste como el infraajuste [45].

#### 3.2.1.1.3 Uso de funciones kernel

En muchos problemas reales, incluidos los entornos de red, las clases no son linealmente separables en el espacio original de características. Para abordar esta situación, las SVM emplean el denominado **truco del kernel** (*kernel trick*), que consiste en proyectar los datos a un espacio de mayor dimensionalidad donde sí sea posible encontrar una separación lineal [45]

Las funciones kernel permiten realizar esta proyección de forma implícita, sin necesidad de calcular explícitamente las coordenadas en el nuevo espacio. Entre los kernels más utilizados se encuentran:

- Kernel lineal
- Kernel polinómico
- Kernel de función de base radial (RBF)

El kernel RBF resulta particularmente relevante en detección de anomalías en redes, ya que permite modelar relaciones no lineales complejas entre métricas de tráfico [9], [46].

#### 3.2.1.1.4 Clasificación binaria frente a enfoque de frontera

En su formulación clásica, las SVM se utilizan para clasificación binaria supervisada, donde el modelo aprende a distinguir entre dos clases previamente etiquetadas (por ejemplo, tráfico normal y tráfico malicioso). En este caso, la frontera de decisión separa explícitamente ambas categorías [45].

Sin embargo, en escenarios de detección de anomalías en redes reales, no siempre se dispone de ejemplos representativos de todas las posibles anomalías. En estos contextos, resulta más apropiado un **enfoque basado en frontera**, en el que el modelo aprende únicamente la región del espacio correspondiente al comportamiento normal y considera anómalas las observaciones que quedan fuera de dicha región [45].

Esta diferencia conceptual es clave para comprender la adaptación del modelo al problema específico de la detección de anomalías, y constituye la base de la variante conocida como *One-Class SVM*, que se desarrollará en el siguiente apartado [46].

### *3.2.1.2 One-Class SVM para detección de anomalías*

En entornos reales de redes de comunicaciones, uno de los principales retos en la detección de anomalías es la escasez o incluso ausencia de datos etiquetados correspondientes a fallos, ataques o comportamientos anómalos. Las anomalías pueden ser poco frecuentes, cambiantes o difíciles de catalogar de forma exhaustiva. En este contexto, la variante **One-Class SVM** resulta especialmente adecuada, ya que permite modelar exclusivamente el comportamiento normal del sistema sin necesidad de disponer de ejemplos explícitos de anomalías [46].

A diferencia de la SVM clásica de clasificación binaria, One-Class SVM adopta un enfoque basado en frontera. El modelo aprende una región del espacio de características que contiene la mayor parte de las observaciones consideradas normales durante el entrenamiento. Posteriormente, cualquier nueva muestra que quede fuera de dicha región se clasifica como potencialmente anómala. Este planteamiento se ajusta de forma natural al problema de monitorización de redes, donde se dispone de grandes volúmenes de datos representativos del funcionamiento habitual, pero no necesariamente de todos los posibles escenarios de fallo [46].

El entrenamiento del modelo se realiza únicamente con datos normales, lo que implica que la frontera aprendida representa una estimación del dominio de comportamiento esperado de la red. Matemáticamente, el problema de optimización puede expresarse de forma simplificada como:

$$\min_{\omega, \rho, \xi_i} \frac{1}{2}||\omega||^2 + \frac{1}{\nu n}\sum_{i=1}^{n} \xi_i - \rho$$

sujeto a:

$$\left(\omega \cdot \phi(x_i)\right) \geq \rho - \xi_i, \quad \xi_i \geq 0$$

donde w define el hiperplano en el espacio transformado, $\phi(x_i)$ representa la proyección de los datos mediante la función kernel, $\xi_i$ son variables de holgura que permiten cierta tolerancia a errores, $n$ es el número de muestras y $\nu$ es un parámetro que controla la proporción de observaciones que pueden considerarse fuera de la frontera [46].

El parámetro $\nu$ desempeña un papel fundamental en el comportamiento del modelo. Este valor establece simultáneamente un límite superior para la fracción de anomalías esperadas en el conjunto de entrenamiento y un límite inferior para la proporción de vectores de soporte. En términos prácticos, valores pequeños de $\nu$ generan fronteras más ajustadas al comportamiento normal, mientras que valores mayores permiten una región más flexible, incrementando la sensibilidad del modelo frente a posibles desviaciones [46].

En redes de comunicaciones, donde las métricas presentan variaciones naturales debidas a cambios de carga, horarios o patrones de uso, la selección adecuada de $\nu$ resulta crítica. Un valor excesivamente bajo puede provocar un elevado número de falsos positivos al interpretar fluctuaciones normales como anomalías. Por el contrario, un valor demasiado alto puede suavizar en exceso la frontera, reduciendo la capacidad de detectar comportamientos realmente anómalos. Esta sensibilidad frente a la variabilidad inherente del tráfico obliga a realizar un ajuste cuidadoso del modelo en función del entorno operativo [3], [46].

Asimismo, el uso de kernels no lineales, especialmente el kernel RBF, permite capturar relaciones complejas entre métricas de red, favoreciendo la identificación de patrones atípicos que no serían detectables mediante fronteras lineales simples. Esta capacidad de modelado no lineal refuerza la idoneidad de One-Class SVM para la detección de anomalías en sistemas donde las interacciones entre variables son dinámicas y multidimensionales [46].

En conjunto, One-Class SVM constituye una solución robusta y conceptualmente coherente para escenarios de monitorización de redes donde se dispone principalmente de datos representativos del funcionamiento normal. Su enfoque basado en la delimitación de una región de comportamiento esperado lo convierte en una herramienta especialmente valiosa para la detección temprana de desviaciones significativas en métricas operativas [46].

#### *3.2.1.3 Parámetros e implicaciones prácticas*

El rendimiento de One-Class SVM en entornos de detección de anomalías en redes depende en gran medida de la correcta selección de sus hiperparámetros. A diferencia de otros modelos más automatizados, el comportamiento de este algoritmo puede variar significativamente según la configuración adoptada, lo que hace imprescindible comprender el impacto práctico de cada parámetro antes de su implementación experimental [46].

#### 3.2.1.3.1 Tipo de kernel

El primer elemento clave es la elección de la función kernel. En aplicaciones reales se utilizan principalmente tres tipos:

- **Kernel lineal**, adecuado cuando las métricas presentan separabilidad aproximadamente lineal y la dimensionalidad es elevada.
- **Kernel polinómico**, que introduce interacciones de orden superior entre variables.
- **Kernel RBF (Radial Basis Function)**, el más empleado en detección de anomalías en redes debido a su capacidad para modelar relaciones no lineales complejas.

En escenarios de tráfico de red, donde las interacciones entre variables como latencia, volumen de tráfico y pérdida de paquetes no suelen ser lineales, el kernel RBF suele ofrecer mejores resultados empíricos [9], [46]. Sin embargo, su mayor flexibilidad también incrementa el riesgo de sobreajuste si no se ajustan adecuadamente los parámetros asociados.

#### 3.2.1.3.2 Parámetro $\nu$ (control de anomalías)

El parámetro $\nu$, introducido en el apartado anterior, tiene una interpretación operativa directa en la práctica. Controla el grado de tolerancia del modelo frente a observaciones alejadas del patrón dominante.

En términos experimentales:

- Valores pequeños de $\nu \rightarrow$ frontera más restrictiva, menor proporción de puntos considerados anómalos.
- Valores grandes de $\nu \rightarrow$ frontera más flexible, mayor sensibilidad a desviaciones.

En redes con alta variabilidad natural del tráfico (por ejemplo, picos horarios o cambios de carga estacionales), una configuración excesivamente restrictiva puede generar numerosos falsos positivos. Por ello, el ajuste de $\nu$debe realizarse teniendo en cuenta la estabilidad estadística de las métricas analizadas [46].

#### 3.2.1.3.3 Parámetro $\gamma$ en kernel RBF

Cuando se emplea el kernel RBF, aparece el parámetro $\gamma$, que determina el alcance de influencia de cada muestra en el espacio transformado.

Desde una perspectiva práctica:

- Valores altos de $\gamma \rightarrow$ fronteras muy ajustadas alrededor de los datos, mayor sensibilidad local.
- Valores bajos de $\gamma \rightarrow$ fronteras más suaves y globales.

Un valor excesivamente alto puede provocar que el modelo capture ruido o fluctuaciones normales del tráfico como si fueran comportamientos estructurales, reduciendo su capacidad de generalización. En cambio, valores demasiado bajos pueden generar una frontera excesivamente amplia que no discrimine adecuadamente anomalías sutiles.

La interacción entre $\nu$ y $\gamma$ es especialmente relevante, ya que ambos parámetros influyen conjuntamente en la complejidad de la frontera aprendida [46].

#### 3.2.1.3.4 Sensibilidad al escalado de datos

Un aspecto frecuentemente crítico en aplicaciones reales es la necesidad de **escalar adecuadamente las métricas de entrada**. Las SVM son sensibles a la magnitud relativa de las variables, ya que las distancias en el espacio de características influyen directamente en la construcción de la frontera.

En redes de comunicaciones, donde pueden coexistir métricas con órdenes de magnitud muy distintos (por ejemplo, latencia en milisegundos frente a volumen de tráfico en megabytes), la ausencia de normalización puede sesgar el modelo hacia aquellas variables con mayor escala numérica.

Por ello, resulta recomendable aplicar técnicas de estandarización o normalización previa antes del entrenamiento, garantizando que todas las métricas contribuyan de forma equilibrada al proceso de aprendizaje [38].

| Hiperparámetro | Función principal | Impacto en el modelo | Riesgo si no se ajusta correctamente |
|---|---|---|---|
| Kernel | Define la forma de la frontera | Permite modelar relaciones lineales o no lineales | Infraajuste (kernel lineal inapropiado) o sobreajuste (kernel complejo sin control) |
| ν | Controla la fracción esperada de anomalías | Ajusta la tolerancia del modelo frente a desviaciones | Exceso de falsos positivos o pérdida de detección |
| γ (RBF) | Determina el alcance de influencia de cada muestra | Controla la complejidad local de la frontera | Frontera demasiado rígida o excesivamente laxa |
| Escalado de datos | Normaliza la magnitud de las variables | Garantiza contribución equilibrada de métricas | Sesgo hacia variables dominantes |

*Tabla 7 Principales hiperparámetros de One-Class SVM y su impacto*

En términos prácticos, la aplicación de One-Class SVM en redes requiere un proceso de validación experimental que combine ajuste de hiperparámetros y análisis del comportamiento del tráfico real o simulado. La sensibilidad del modelo a estos parámetros constituye tanto una limitación como una ventaja: si bien exige un ajuste cuidadoso, también permite adaptar la frontera de decisión a las características específicas del entorno de red analizado [21], [24].

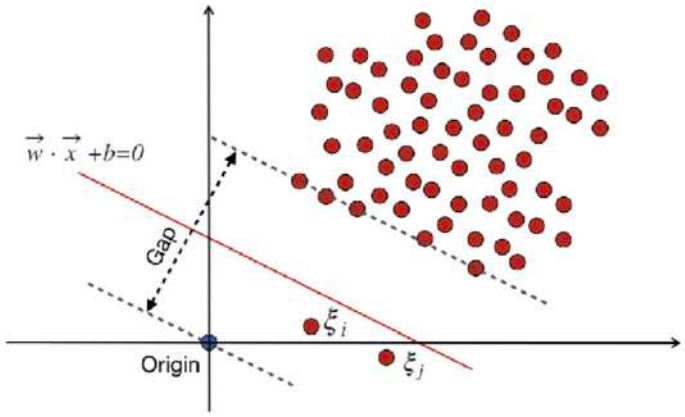


*Ilustración 14 Esquema conceptual de One-Class SVM para detección de anomalías. Fuente: Adaptado de Analytics Vidhya [49]*

### 3.2.2 RANDOM FOREST

Random Forest es un algoritmo de aprendizaje automático basado en la técnica de *ensemble learning*, cuyo principio fundamental consiste en combinar múltiples modelos simples, en este caso, árboles de decisión, para construir un modelo conjunto más robusto y con mayor capacidad de generalización [47].

A diferencia de un único árbol de decisión, que puede presentar alta varianza y tendencia al sobreajuste, Random Forest genera un conjunto de árboles entrenados de forma independiente a partir de subconjuntos aleatorios de los datos y de las variables disponibles. La predicción final se obtiene mediante un proceso de agregación, normalmente por votación mayoritaria en tareas de clasificación. Este mecanismo reduce significativamente la sensibilidad al ruido y mejora la estabilidad del modelo frente a variaciones en los datos de entrada [47].

En el contexto de redes de comunicaciones, donde las métricas operativas pueden presentar fluctuaciones naturales y relaciones no lineales entre variables, esta robustez constituye una ventaja clave. La detección de anomalías en tráfico de red no depende habitualmente de una única métrica aislada, sino de la interacción entre múltiples parámetros como volumen de tráfico, latencia, pérdida de paquetes, duración de flujos o uso de CPU. Random Forest permite capturar estas interacciones de manera automática sin necesidad de definir explícitamente modelos matemáticos complejos [9], [47].

Otra razón que explica su amplia adopción en estudios de detección de anomalías es su capacidad para manejar datos de alta dimensionalidad y conjuntos de características heterogéneos. En entornos de monitorización de red, es habitual trabajar con decenas de métricas simultáneamente, lo que puede dificultar la aplicación de modelos más sensibles a la dimensionalidad. Random Forest mantiene un rendimiento estable incluso cuando el número de variables es elevado, gracias al proceso de selección aleatoria de características en cada árbol [47].

En aplicaciones prácticas, este algoritmo se ha utilizado para la clasificación de tráfico legítimo frente a tráfico malicioso, la identificación de intrusiones, la detección de patrones anómalos en flujos de red y el análisis de comportamientos sospechosos en sistemas distribuidos [47]. Además, ofrece una ventaja adicional relevante en entornos de ingeniería: la posibilidad de estimar la importancia relativa de cada característica, lo que facilita la interpretación de qué métricas contribuyen en mayor medida a la detección de anomalías.

En conjunto, Random Forest representa una solución equilibrada entre capacidad predictiva, robustez frente al ruido y facilidad de implementación, lo que justifica su presencia recurrente en la literatura especializada sobre detección de anomalías en redes [47].

#### *3.2.2.1 Fundamento conceptual del modelo*

El fundamento conceptual de Random Forest se apoya en la combinación de múltiples árboles de decisión construidos de manera independiente y posteriormente agregados para generar una predicción conjunta. Esta arquitectura permite superar algunas de las

limitaciones inherentes a los árboles individuales, especialmente su elevada varianza y tendencia al sobreajuste en conjuntos de datos complejos [47].

#### 3.2.2.1.1 Árboles de decisión como modelos base

Un árbol de decisión es un modelo estructurado en forma jerárquica que divide recursivamente el espacio de características mediante reglas del tipo "si-entonces". En cada nodo interno se evalúa una condición sobre una variable, generando particiones sucesivas hasta alcanzar nodos hoja que representan una clase o una predicción final.

En el ámbito de redes de comunicaciones, un árbol podría, por ejemplo, dividir el tráfico en función de umbrales sobre latencia, volumen de paquetes o duración de sesión. Sin embargo, un único árbol puede adaptarse en exceso a las particularidades del conjunto de entrenamiento, capturando ruido como si fuese información relevante. Este fenómeno reduce su capacidad de generalización ante nuevos datos [47].

#### 3.2.2.1.2 Bagging (Bootstrap Aggregating)

Para mitigar este problema, Random Forest emplea la técnica de **bagging** (*bootstrap aggregating*). Este procedimiento consiste en:

1. Generar múltiples subconjuntos del conjunto de entrenamiento mediante muestreo aleatorio con reemplazo.
2. Entrenar un árbol de decisión independiente sobre cada subconjunto.

Al utilizar muestras ligeramente distintas en cada árbol, se introducen variaciones en las estructuras aprendidas. Esto provoca que los errores individuales de cada árbol tiendan a compensarse cuando se combinan sus predicciones.

En entornos de tráfico de red, donde pueden existir fluctuaciones y patrones transitorios, este mecanismo resulta especialmente útil para evitar que el modelo quede condicionado por eventos puntuales o anomalías aisladas presentes en los datos de entrenamiento [51].

#### 3.2.2.1.3 Votación agregada

Una vez entrenado el conjunto de árboles, la predicción final del modelo se obtiene mediante un proceso de **votación agregada**. En tareas de clasificación, cada árbol emite una predicción independiente y la clase final se determina por mayoría.

Este mecanismo aporta estabilidad al modelo, ya que la decisión final no depende de una única estructura, sino del consenso entre múltiples árboles. En el contexto de detección de anomalías en redes, esta agregación reduce la probabilidad de que una decisión errónea individual conduzca a una clasificación incorrecta global.

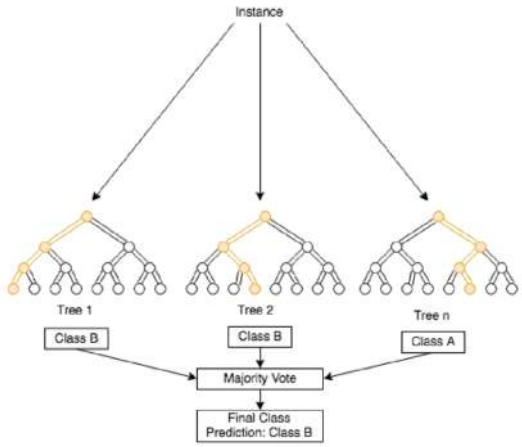


*Ilustración 15 Diagrama conceptual generalizado de un clasificador Random Forest. Fuente: Adaptado de Research Gate [50]*

#### 3.2.2.1.4 Reducción de varianza

Desde el punto de vista estadístico, el principal beneficio de Random Forest es la **reducción de varianza**. Los árboles individuales son modelos de alta varianza: pequeños cambios en los datos pueden producir estructuras muy diferentes. Al promediar múltiples árboles entrenados sobre muestras distintas, la varianza global del modelo disminuye significativamente.

Esta propiedad resulta especialmente relevante en redes de comunicaciones, donde los datos pueden ser ruidosos, no estacionarios y altamente variables. La reducción de varianza mejora la estabilidad del sistema de detección y reduce la probabilidad de falsas alarmas generadas por fluctuaciones normales del tráfico [47].

Además, Random Forest introduce un segundo nivel de aleatorización al seleccionar un subconjunto aleatorio de características en cada división del árbol. Este mecanismo incrementa la diversidad entre árboles y refuerza el efecto de reducción de varianza, contribuyendo a un mejor equilibrio entre sesgo y varianza del modelo.

En conjunto, el fundamento conceptual de Random Forest se basa en la combinación de modelos simples, diversidad estructural y agregación estadística, lo que le permite ofrecer un rendimiento robusto y estable en problemas de clasificación de tráfico y detección de comportamientos anómalos en redes [47].

### *3.2.2.2 Aplicación a detección de anomalías*

Random Forest puede aplicarse a la detección de anomalías en redes de comunicaciones bajo distintos enfoques, dependiendo principalmente de la disponibilidad de datos etiquetados y del objetivo del sistema de monitorización. Su flexibilidad permite adaptarlo tanto a escenarios supervisados como a configuraciones parcialmente etiquetadas, lo que explica su amplia adopción en estudios de clasificación de tráfico [9], [47].

#### 3.2.2.2.1 Enfoque supervisado

En el enfoque supervisado, el modelo se entrena con un conjunto de datos que incluye ejemplos explícitamente etiquetados como comportamiento normal o anómalo. Cada árbol aprende reglas que permiten discriminar entre ambas clases, basándose en combinaciones de métricas como volumen de tráfico, duración de flujos, latencia o tasa de pérdida de paquetes.

Este planteamiento es especialmente útil en entornos donde se dispone de registros históricos de ataques, fallos o incidentes previamente identificados. En tales casos, Random Forest puede capturar patrones característicos asociados a cada tipo de evento y ofrecer una clasificación directa de nuevas observaciones.

La agregación de múltiples árboles proporciona robustez frente a ruido y variabilidad natural del tráfico, reduciendo la probabilidad de decisiones erróneas basadas en valores atípicos aislados. No obstante, este enfoque depende fuertemente de la calidad y representatividad de las etiquetas disponibles, lo que puede limitar su capacidad para detectar anomalías completamente nuevas o no contempladas en el conjunto de entrenamiento [9], [47].

#### 3.2.2.2.2 Enfoque semi-supervisado

En muchos entornos reales de red, la disponibilidad de datos etiquetados es limitada. En estos casos, Random Forest puede emplearse en configuraciones semi-supervisadas, donde el modelo se entrena principalmente con datos normales y se analiza el comportamiento de nuevas observaciones en función de su coherencia con los patrones aprendidos [5].

Aunque el algoritmo no está diseñado originalmente como modelo puramente no supervisado, su estructura permite identificar observaciones que presentan patrones de decisión poco frecuentes dentro del conjunto de árboles. Esto puede lograrse, por ejemplo, evaluando la consistencia de las predicciones individuales o analizando la distribución de probabilidades asociadas a cada clase [5].

En este contexto, el modelo actúa como un estimador del comportamiento habitual de la red, detectando desviaciones que no encajan en las estructuras de decisión dominantes aprendidas durante el entrenamiento [5].

#### 3.2.2.2.3 Trayectorias de decisión atípicas

Un aspecto particularmente interesante de Random Forest en detección de anomalías es el análisis de las **trayectorias de decisión** dentro de los árboles. Cada observación recorre una secuencia específica de nodos desde la raíz hasta una hoja en cada árbol del conjunto. Esta secuencia refleja las reglas aplicadas sobre las métricas de entrada [47].

Las anomalías tienden a seguir trayectorias menos frecuentes o inconsistentes en comparación con el comportamiento normal. Por ejemplo, pueden activar combinaciones de reglas que rara vez aparecen en los datos habituales o distribuirse de manera irregular entre las hojas de los distintos árboles.

El análisis agregado de estas trayectorias permite identificar patrones estructuralmente atípicos, incluso cuando las métricas individuales no presentan valores extremos de forma aislada. En redes de comunicaciones, donde muchas anomalías surgen de interacciones complejas entre variables y no de desviaciones simples en una única métrica, esta capacidad resulta especialmente valiosa.

En conjunto, Random Forest ofrece una aproximación versátil a la detección de anomalías en redes, combinando capacidad de clasificación directa en entornos supervisados con mecanismos implícitos de detección de comportamientos atípicos cuando la información etiquetada es limitada. Su robustez estructural y su capacidad para modelar interacciones no lineales entre múltiples métricas lo convierten en una herramienta sólida dentro del conjunto de modelos analizados en este trabajo [47].

#### *3.2.2.3 Parámetros del modelo*

El comportamiento de Random Forest en tareas de detección de anomalías depende de la configuración de sus hiperparámetros estructurales. Estos parámetros determinan el grado de complejidad del modelo, su capacidad de generalización y su coste computacional, aspectos especialmente relevantes en sistemas de monitorización de red donde pueden procesarse grandes volúmenes de datos en tiempo real [47].

A continuación, se analizan los principales hiperparámetros y sus implicaciones prácticas.

**3.2.2.3.1 n_estimators**
El parámetro **n_estimators** define el número de árboles que componen el bosque.

Un mayor número de árboles suele incrementar la estabilidad del modelo y reducir la varianza, ya que la predicción final se basa en un promedio más amplio de decisiones individuales. En detección de anomalías, esto puede traducirse en mayor robustez frente a fluctuaciones normales del tráfico.

Sin embargo, incrementar excesivamente este valor aumenta el coste computacional tanto en entrenamiento como en inferencia, lo que puede resultar crítico en entornos de monitorización continua. En la práctica, suele buscarse un equilibrio entre rendimiento y eficiencia [47].

**3.2.2.3.2 max_depth**
El parámetro **max_depth** controla la profundidad máxima de cada árbol.

- Profundidades elevadas permiten modelar interacciones complejas entre métricas de red.
- Profundidades reducidas generan árboles más simples y generalizables.

En el contexto de redes, donde las anomalías pueden depender de combinaciones específicas de variables (por ejemplo, alta latencia combinada con bajo throughput), una profundidad insuficiente puede impedir capturar dichos patrones. Por el contrario, profundidades

excesivas pueden provocar sobreajuste a eventos particulares del conjunto de entrenamiento [47].

#### 3.2.2.3.3 min_samples_split

El parámetro **min_samples_split** establece el número mínimo de muestras necesarias para dividir un nodo interno.

Valores bajos permiten divisiones más frecuentes, generando árboles más detallados y potencialmente más sensibles a patrones sutiles. No obstante, esto puede aumentar la sensibilidad al ruido.

Valores más altos restringen la creación de divisiones, favoreciendo estructuras más estables y reduciendo la probabilidad de modelar fluctuaciones transitorias del tráfico como si fueran comportamientos estructurales.

En entornos con alta variabilidad natural, un ajuste conservador de este parámetro puede contribuir a reducir falsos positivos [47].

#### 3.2.2.3.4 max_features

El parámetro **max_features** determina el número de características consideradas en cada división del árbol.

Este mecanismo introduce aleatoriedad adicional y reduce la correlación entre árboles, reforzando el efecto de reducción de varianza del conjunto.

En detección de anomalías en redes, donde pueden existir métricas altamente correlacionadas, limitar el número de variables por división ayuda a evitar que todos los árboles dependan sistemáticamente de las mismas características dominantes. Esto favorece una representación más diversa del comportamiento de la red y mejora la capacidad de generalización [47].

#### 3.2.2.3.5 bootstrap

El parámetro **bootstrap** indica si el entrenamiento de cada árbol se realiza mediante muestreo con reemplazo.

Cuando está activado (configuración habitual), cada árbol se entrena con una muestra distinta del conjunto de datos original, lo que incrementa la diversidad estructural entre árboles.

En escenarios de tráfico de red, donde pueden existir patrones repetitivos o eventos concentrados en determinados intervalos temporales, el uso de bootstrap ayuda a evitar que todos los árboles queden condicionados por las mismas observaciones, mejorando la estabilidad del sistema de detección [47].

| Hiperparámetro | Función estructural | Impacto en detección de anomalías | Riesgo si se configura incorrectamente |
|---|---|---|---|
| *n_estimators* | Número de árboles en el conjunto | Mayor estabilidad y reducción de varianza | Alto coste computacional si es excesivo |
| *max_depth* | Profundidad máxima del árbol | Controla la complejidad de patrones capturados | Sobreajuste (profundidad alta) o infraajuste (baja) |
| *min_samples_split* | Muestras mínimas para dividir un nodo | Regula sensibilidad a patrones locales | Excesiva sensibilidad al ruido o pérdida de detalle |
| *max_features* | Nº de variables consideradas por división | Aumenta diversidad entre árboles | Correlación excesiva entre árboles si es alto |
| *bootstrap* | Muestreo con reemplazo | Mejora generalización y robustez | Menor diversidad si se desactiva |

*Tabla 8 Hiperparámetros de Random Forest y efecto en detección*

### 3.2.3 K-MEANS

K-means es un algoritmo de aprendizaje no supervisado perteneciente a la familia de técnicas de *clustering*, cuyo objetivo es agrupar un conjunto de datos en K clústeres en función de su similitud interna [51], [52]. A diferencia de los modelos supervisados, no requiere etiquetas previas que indiquen qué observaciones corresponden a comportamiento normal o anómalo, lo que lo convierte en una herramienta especialmente útil en entornos donde dicha información no está disponible.

En el contexto de redes de comunicaciones, esta característica resulta particularmente relevante. En muchos escenarios reales, no se dispone de un registro completo y fiable de incidentes o ataques previamente etiquetados. Además, los patrones anómalos pueden evolucionar con el tiempo, dificultando la construcción de conjuntos de entrenamiento completamente representativos. En estas circunstancias, los enfoques no supervisados permiten modelar el comportamiento habitual de la red a partir de los propios datos históricos y detectar desviaciones sin necesidad de conocimiento previo explícito [22].

K-means opera bajo la hipótesis de que el comportamiento normal de la red forma agrupaciones coherentes en el espacio de características. Estas agrupaciones se construyen a partir de métricas como latencia, throughput, pérdida de paquetes, número de flujos activos o uso de CPU. Las observaciones que no encajan adecuadamente en ninguno de los clústeres principales pueden interpretarse como comportamientos atípicos o potencialmente anómalos [22], [51].

Desde una perspectiva aplicada, K-means presenta varias ventajas iniciales para el análisis de tráfico de red:

- Simplicidad conceptual y facilidad de implementación.
- Escalabilidad adecuada para grandes volúmenes de datos.
- Capacidad para descubrir patrones desconocidos sin intervención manual.

Por estas razones, el algoritmo se emplea frecuentemente como punto de partida en análisis exploratorios de anomalías, así como en sistemas donde la ausencia de etiquetas impide el uso directo de modelos supervisados [22], [51].

No obstante, su eficacia depende de la estructura real de los datos y de la correcta elección del número de clústeres, aspectos que se analizarán en los apartados siguientes.

#### *3.2.3.1 Fundamento conceptual del modelo*

El algoritmo K-means se basa en un proceso iterativo cuyo objetivo es dividir el conjunto de datos en K clústeres de forma que las observaciones dentro de cada grupo sean lo más similares posible entre sí y lo más diferentes posible respecto a las de otros grupos [51], [52].

Desde una perspectiva geométrica, cada clúster se representa mediante un centroide, que actúa como punto medio de las observaciones asignadas a dicho grupo. El algoritmo ajusta progresivamente la posición de estos centroides hasta alcanzar una configuración estable.

El procedimiento puede describirse en cuatro fases principales:

##### 3.2.3.1.1 1. Inicialización

En la primera etapa, se seleccionan K centroides iniciales. Habitualmente se eligen de forma aleatoria entre las observaciones del conjunto de datos, aunque existen estrategias más avanzadas que buscan mejorar la estabilidad inicial [52].

La elección de estos centroides influye en la solución final, ya que K-means puede converger hacia mínimos locales dependiendo de la configuración inicial.

##### 3.2.3.1.2 2. Asignación

En esta fase, cada observación del conjunto de datos se asigna al clúster cuyo centroide se encuentra más próximo según una métrica de distancia, generalmente la distancia euclídea.

Este paso genera una partición provisional del espacio de características, agrupando las muestras en función de su proximidad geométrica a los centroides actuales [52].

##### 3.2.3.1.3 3. Recomputación

Una vez asignadas todas las observaciones, se recalcula la posición de cada centroide como la media de las muestras pertenecientes a su clúster correspondiente.

Este ajuste modifica la estructura del agrupamiento y prepara el algoritmo para una nueva fase de asignación [52].

##### 3.2.3.1.4 Convergencia

El proceso de asignación y recomputación se repite de manera iterativa hasta que:

- Las asignaciones dejan de cambiar, o
- El desplazamiento de los centroides es menor que un umbral predefinido, o
- Se alcanza un número máximo de iteraciones.

En este punto, el algoritmo se considera convergente y los clústeres finales quedan definidos [52].

#### 3.2.3.1.5 Función objetivo

El criterio que guía este proceso iterativo es la minimización de la varianza interna de los clústeres. Formalmente, el algoritmo busca minimizar la siguiente función objetivo:

$$J = \sum_{i=1}^{K} \sum_{x \in C_i} ||x - \mu_i||^2$$

donde:

- $K$ es el número de clústeres,
- $C_i$ representa el conjunto de observaciones asignadas al clúster $i$,
- $\mu_i$ es el centroide del clúster,
- $||x - \mu_i||^2$ es la distancia cuadrática entre una observación y su centroide.

Esta formulación refleja el objetivo fundamental de K-means: generar agrupaciones compactas, con alta coherencia interna.

En el contexto de redes de comunicaciones, este fundamento conceptual implica que el algoritmo identifica regiones densas del espacio de métricas que representan comportamientos habituales. Las observaciones que se alejan significativamente de los centroides o que generan agrupaciones poco consistentes pueden interpretarse como comportamientos anómalos, aspecto que se desarrollará en el siguiente apartado [51], [52].

### *3.2.3.2 Detección de anomalías mediante distancia*

Una vez entrenado el modelo K-means, la detección de anomalías se basa en el análisis de la relación geométrica entre cada observación y los centroides definidos por el algoritmo. En entornos de red, esta interpretación resulta especialmente útil cuando no se dispone de etiquetas y se pretende identificar comportamientos atípicos a partir de desviaciones estructurales respecto al patrón dominante [22], [51].

El criterio fundamental de detección se apoya en tres mecanismos principales.

#### 3.2.3.2.1 Puntos alejados del centroide

Cada observación asignada a un clúster presenta una distancia asociada respecto a su centroide. En condiciones normales, las métricas de red tienden a concentrarse alrededor de valores característicos, formando agrupaciones relativamente compactas.

Sin embargo, cuando una muestra presenta una combinación de métricas poco habitual, por ejemplo, latencia elevada junto con baja utilización de recursos o picos inesperados de tráfico, su distancia al centroide correspondiente puede ser significativamente mayor que la media del grupo.

Estas observaciones alejadas pueden interpretarse como posibles anomalías, especialmente si superan un umbral de distancia previamente definido.

En la práctica, el análisis puede realizarse mediante:

- Comparación con la media y desviación estándar de las distancias intra-clúster.
- Selección de un percentil alto (por ejemplo, 95 o 99) como criterio de alerta.
- Establecimiento de un umbral fijo basado en conocimiento operativo de la red.

#### 3.2.3.2.2 Clústeres pequeños

Otra estrategia consiste en analizar el tamaño relativo de los clústeres obtenidos.

En redes de comunicaciones, el comportamiento normal suele concentrarse en uno o varios clústeres de alta densidad. En cambio, agrupaciones con un número reducido de observaciones pueden representar patrones poco frecuentes o eventos esporádicos.

Estos clústeres minoritarios pueden corresponder a:

- Fallos temporales,
- Comportamientos anómalos repetitivos,
- Tráfico inusual asociado a configuraciones específicas.

No obstante, es importante distinguir entre patrones legítimos poco frecuentes y anomalías reales, ya que K-means no incorpora conocimiento semántico sobre el contexto del tráfico [22].

#### 3.2.3.2.3 Umbral de distancia

La detección efectiva requiere la definición de un criterio operativo que permita decidir cuándo una observación debe considerarse anómala.

En este sentido, el umbral de distancia desempeña un papel central. Puede definirse de forma:

- Estadística (basado en la distribución de distancias),
- Empírica (ajustado experimentalmente),
- Operativa (según niveles de tolerancia del sistema).

En entornos con alta variabilidad natural del tráfico, un umbral demasiado estricto puede generar un elevado número de falsos positivos. Por el contrario, un umbral excesivamente laxo puede impedir la detección temprana de anomalías sutiles.

Por ello, su calibración debe realizarse considerando la dinámica temporal de la red y los objetivos del sistema de supervisión [22], [51].

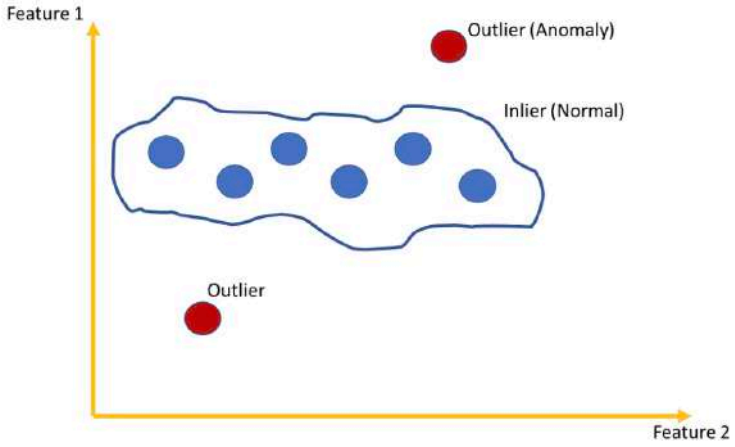


*Ilustración 16 Clústeres de comportamiento normal y anomalías. Fuente: Adaptado de Yavariabdi (2023) [55].*

En conjunto, la detección de anomalías mediante K-means se fundamenta en una interpretación geométrica del comportamiento de la red. Aunque su simplicidad facilita la implementación y el análisis exploratorio, la precisión del método depende críticamente de la elección del número de clústeres y de la correcta definición de los umbrales de decisión, aspectos que se desarrollarán en el siguiente apartado.

#### *3.2.3.3 Parámetros e implicaciones*

El rendimiento de K-means en la detección de anomalías depende en gran medida de la correcta configuración de sus parámetros y de ciertas decisiones metodológicas previas. A diferencia de otros modelos más complejos, K-means presenta pocos hiperparámetros explícitos, pero su impacto en el resultado final puede ser significativo, especialmente en entornos de red con alta variabilidad [51], [52].

A continuación, se analizan los aspectos más relevantes.

##### 3.2.3.3.1 Número de clústeres (K)

El parámetro más importante del modelo es el número de clústeres K, que debe definirse antes de ejecutar el algoritmo.

En redes de comunicaciones, la elección de K determina el nivel de granularidad con el que se modela el comportamiento normal:

- Un valor de K demasiado bajo puede agrupar comportamientos distintos dentro del mismo clúster, dificultando la detección de anomalías sutiles.
- Un valor de K demasiado alto puede fragmentar excesivamente los datos, generando clústeres pequeños que podrían interpretarse erróneamente como anómalos.

La selección adecuada de K debe basarse en criterios cuantitativos y en conocimiento del dominio de aplicación [51], [52].

#### 3.2.3.3.2 Método del codo

Uno de los procedimientos más utilizados para determinar K es el método del codo. Este enfoque consiste en ejecutar el algoritmo para distintos valores de K y analizar la evolución de la función objetivo (suma de distancias intra-clúster).

Se busca el punto a partir del cual la reducción del error comienza a estabilizarse. Ese punto, visualmente identificado como un "codo" en la curva, suele considerarse un compromiso razonable entre complejidad del modelo y capacidad descriptiva [18].

En aplicaciones de red, este método permite evitar tanto la subsegmentación como la sobresegmentación del tráfico [52].

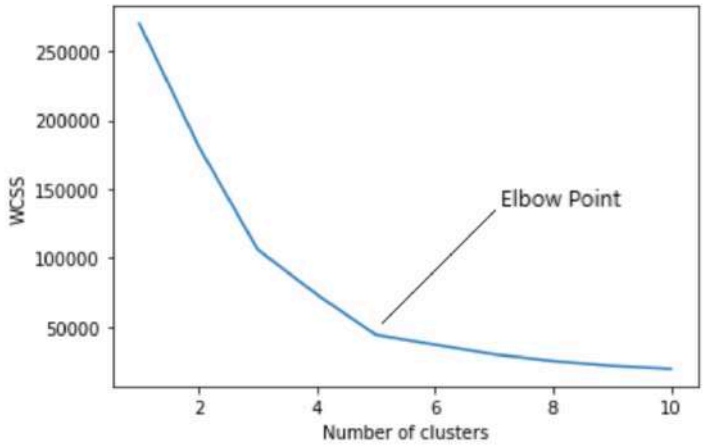


*Ilustración 17 Método del codo para la selección del número óptimo de clústeres. Fuente: Analytics Vidhya (2021) [54]*

#### 3.2.3.3.3 Sensibilidad a la escala de los datos

K-means se basa en distancias geométricas, lo que implica que es altamente sensible a la escala de las variables.

En redes de comunicaciones, las métricas pueden presentar órdenes de magnitud muy distintos (por ejemplo, latencia en milisegundos frente a número de paquetes por segundo). Si no se realiza una normalización previa, las variables de mayor escala dominarán el cálculo de distancias, distorsionando los clústeres resultantes.

Por ello, es imprescindible aplicar técnicas de escalado como:

- Normalización min-max,
- Estandarización (media cero, varianza uno).

La omisión de este paso puede comprometer seriamente la capacidad del modelo para identificar anomalías reales [52].

#### 3.2.3.3.4 Inicialización aleatoria

La selección inicial de los centroides influye directamente en la convergencia del algoritmo.

Debido a su naturaleza iterativa, K-means puede converger hacia soluciones locales distintas en función de la inicialización. Para mitigar este problema, en la práctica se recomienda:

- Ejecutar el algoritmo múltiples veces con diferentes inicializaciones.
- Emplear estrategias mejoradas como K-means++.

En entornos de red, donde la estabilidad del sistema de detección es relevante, esta consideración resulta especialmente importante para garantizar resultados reproducibles.

| Parámetro / Aspecto | Función en el modelo | Impacto en detección de anomalías | Implicación práctica en redes |
|---|---|---|---|
| *K* | Número de clústeres | Determina nivel de granularidad del comportamiento normal | Debe equilibrar precisión y estabilidad |
| Método del codo | Selección de K | Evita infra y sobresegmentación | Requiere análisis previo exploratorio |
| Escalado de variables | Preprocesamiento | Influye directamente en cálculo de distancias | Obligatorio cuando hay métricas heterogéneas |
| Inicialización | Punto de partida del algoritmo | Puede generar soluciones distintas | Recomendable usar múltiples ejecuciones |

*Tabla 9 Principales parámetros de K-means y su impacto en detección de anomalías*

En conjunto, aunque K-means es conceptualmente simple, su aplicación efectiva en la detección de anomalías en redes exige una cuidadosa selección de parámetros y un preprocesamiento adecuado de las métricas. La interpretación de los resultados debe realizarse considerando estas implicaciones, especialmente en entornos con alta variabilidad temporal o con múltiples métricas heterogéneas [53].

### 3.2.4 ISOLATION FOREST

Isolation Forest es un algoritmo de aprendizaje automático no supervisado diseñado específicamente para la detección de anomalías. A diferencia de otros modelos que intentan describir con precisión el comportamiento normal de los datos, este enfoque parte de una premisa diferente: las anomalías son observaciones escasas y distintas, y por tanto pueden aislarse más fácilmente que los patrones habituales [56], [57].

Esta característica lo convierte en un modelo especialmente adecuado para entornos donde:

- No se dispone de datos etiquetados,
- El comportamiento normal presenta variabilidad,
- Se generan grandes volúmenes de información.

En redes de comunicaciones, estas condiciones son habituales. Las métricas operativas como tráfico, latencia, duración de flujos o uso de recursos pueden presentar cambios dinámicos en función de la carga o del tipo de servicio, mientras que los eventos anómalos (fallos, congestiones inesperadas o comportamientos atípicos) tienden a ser menos frecuentes y más extremos [24], [56].

La idea central del modelo es el aislamiento. En lugar de estimar densidades o construir fronteras complejas entre clases, Isolation Forest separa observaciones mediante particiones aleatorias del espacio de características. Las muestras que requieren menos divisiones para quedar aisladas se consideran potencialmente anómalas, ya que presentan valores poco comunes o extremos respecto al conjunto de datos.

Esta estrategia presenta varias ventajas relevantes para redes:

- No requiere modelar explícitamente la distribución del tráfico.
- Es escalable a grandes volúmenes de datos.
- Presenta complejidad computacional aproximadamente lineal.
- Funciona adecuadamente con datos de alta dimensionalidad.

Además, al no depender de supuestos estadísticos estrictos, se adapta con mayor flexibilidad a métricas heterogéneas y relaciones no lineales entre variables, características habituales en redes modernas [56].

Por estas razones, Isolation Forest se ha consolidado como uno de los modelos de referencia en detección de anomalías no supervisada, especialmente en escenarios operativos donde la disponibilidad de etiquetas es limitada y la eficiencia computacional resulta crítica [56], [57].

#### *3.2.4.1 Fundamento conceptual*

Isolation Forest se basa en la construcción de múltiples árboles binarios aleatorios, conocidos como *isolation trees*. A diferencia de los árboles de decisión utilizados en clasificación, estos árboles no se construyen optimizando un criterio de pureza o ganancia de información. En su lugar, se generan mediante particiones completamente aleatorias del espacio de características [56], [57].

El proceso puede describirse en tres elementos fundamentales.

##### 3.2.4.1.1 Árboles aleatorios

Cada árbol se construye seleccionando aleatoriamente:

- Una característica del conjunto de datos.
- Un valor de corte dentro del rango de esa característica.

Esta selección aleatoria genera una partición del espacio que divide el conjunto de observaciones en dos subconjuntos. El proceso se repite de manera recursiva hasta que cada observación queda aislada en un nodo hoja o se alcanza una profundidad máxima predefinida.

La ausencia de un criterio de optimización tradicional simplifica el modelo y reduce significativamente su coste computacional [56].

##### 3.2.4.1.2 División recursiva del espacio

En cada nivel del árbol, el espacio de características se fragmenta progresivamente. Las observaciones que presentan valores extremos o poco frecuentes tienden a quedar separadas

del resto con pocas divisiones, mientras que las observaciones normales, que se concentran en regiones densas del espacio, requieren más particiones para aislarse.

Este comportamiento es clave en la detección de anomalías: el modelo no busca describir explícitamente la normalidad, sino identificar qué observaciones resultan más fáciles de separar del conjunto [56], [57].

#### 3.2.4.1.3 Profundidad media como métrica de anomalía

El indicador central del modelo es la longitud del camino desde la raíz del árbol hasta el nodo donde queda aislada una observación.

- Si una muestra se aísla tras pocas divisiones (profundidad pequeña), es más probable que sea anómala.
- Si requiere muchas divisiones (profundidad grande), es más probable que pertenezca al comportamiento normal.

Dado que un único árbol puede estar influenciado por la aleatoriedad, Isolation Forest construye múltiples árboles y calcula la profundidad media de cada observación a lo largo del conjunto. Esta media constituye la base para estimar su grado de anomalía [24], [56].

En redes de comunicaciones, este enfoque resulta especialmente adecuado para identificar eventos poco frecuentes, como picos inesperados de tráfico o combinaciones atípicas de métricas, sin necesidad de modelar explícitamente la distribución subyacente.

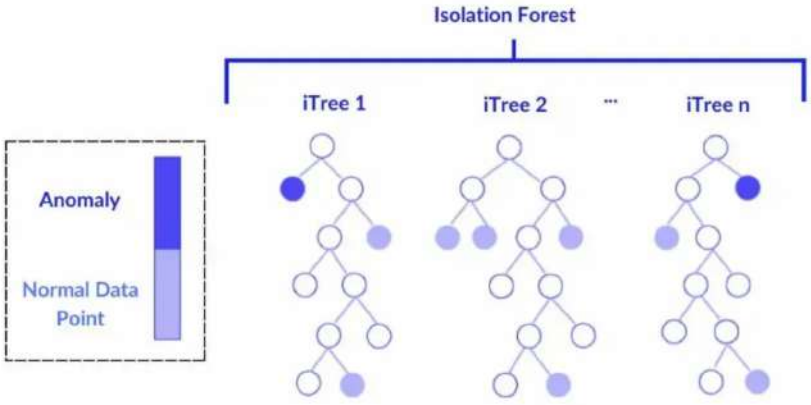


*Ilustración 18 Esquema conceptual del funcionamiento de Isolation Forest. Fuente: Adaptado de Spot Intelligence (2024) [58]*

Este fundamento conceptual diferencia claramente a Isolation Forest de modelos basados en densidad o fronteras geométricas, ya que su lógica se apoya en la facilidad de aislamiento como criterio primario de detección.

### *3.2.4.2 Función de puntuación de anomalía*

Una vez construidos los árboles de aislamiento, el modelo asigna a cada observación una puntuación de anomalía basada en la profundidad media necesaria para aislarla. Este valor permite cuantificar de forma continua el grado de rareza de cada muestra dentro del conjunto de datos [56], [57].

La función de puntuación utilizada en Isolation Forest se expresa como:

$$Score(x,n) = 2^{-\frac{E(h(x))}{c(n)}}$$

donde:

- E(h(x)) representa la profundidad media del nodo en el que queda aislada la observación x a lo largo de todos los árboles.
- c(n) es un factor de normalización que depende del número de muestras n.
- El término exponencial transforma la profundidad en un valor normalizado entre 0 y 1.

#### 3.2.4.2.1 Interpretación del score

El significado operativo del resultado es directo:

- Si el valor de E(h(x)) es pequeño (la observación se aísla rápidamente), el score se aproxima a 1 → alta probabilidad de anomalía.
- Si el valor de E(h(x)) es grande (requiere muchas divisiones), el score se aproxima a 0 → comportamiento normal.

En aplicaciones de red, esta puntuación permite:

- Establecer un umbral de decisión para generar alertas.
- Ordenar eventos según su grado de anomalía.
- Ajustar la sensibilidad del sistema en función del contexto operativo.

A diferencia de modelos que producen una clasificación binaria directa, Isolation Forest genera un indicador continuo, lo que ofrece mayor flexibilidad en entornos con diferentes niveles de tolerancia al riesgo [56], [57]..

### *3.2.4.3 Parámetros del modelo*

El rendimiento de Isolation Forest depende en gran medida de la correcta configuración de sus hiperparámetros. Aunque el modelo es relativamente robusto frente a ajustes imprecisos, determinados parámetros influyen directamente en la estabilidad del score, la sensibilidad ante anomalías y el coste computacional [56], [57].

A continuación, se describen los principales hiperparámetros y su impacto en escenarios de detección de anomalías en redes de comunicaciones.

#### 3.2.4.3.1 n_estimators

Define el número de árboles de aislamiento que componen el bosque.

- Un mayor número de árboles incrementa la estabilidad estadística del score.
- Reduce la variabilidad causada por particiones aleatorias.
- Aumenta el coste computacional.

En entornos de red con alta variabilidad del tráfico, utilizar un número suficiente de árboles mejora la consistencia en la detección de eventos poco frecuentes [56], [57].

#### 3.2.4.3.2 max_samples

Determina el número de muestras utilizadas para construir cada árbol.

- Si se emplea un subconjunto pequeño, el entrenamiento es más rápido.
- Valores mayores permiten una representación más fiel del comportamiento global.
- Valores excesivamente grandes pueden reducir la capacidad de aislar rápidamente anomalías raras.

En monitorización de redes con grandes volúmenes de datos, suele utilizarse muestreo para equilibrar precisión y escalabilidad [56], [57].

#### 3.2.4.3.3 contamination

Indica la proporción estimada de anomalías presentes en los datos.

- Se utiliza para definir automáticamente el umbral de decisión.
- Valores altos aumentan la sensibilidad del modelo.
- Valores bajos lo hacen más conservador.

En redes reales, donde la proporción de eventos anómalos suele ser reducida, este parámetro debe ajustarse cuidadosamente para evitar un número excesivo de falsos positivos [56], [57].

#### 3.2.4.3.4 max_features

Especifica el número de características consideradas al generar cada partición.

- Valores menores introducen mayor aleatoriedad.
- Valores mayores permiten capturar interacciones más complejas entre métricas.
- Influye tanto en la diversidad de los árboles como en el coste computacional.

En redes con múltiples métricas correlacionadas (latencia, tráfico, pérdidas, etc.), este parámetro afecta a la capacidad del modelo para detectar anomalías multivariantes [56], [57].

| Parámetro | Función principal | Impacto en detección | Implicación práctica en redes |
|---|---|---|---|
| *n_estimators* | Número de árboles | Mayor estabilidad del score | Incrementa coste computacional |
| *max_samples* | Tamaño de muestra por árbol | Influye en capacidad de aislamiento | Permite controlar escalabilidad |
| *contamination* | Proporción esperada de anomalías | Ajusta umbral de decisión | Determina sensibilidad del sistema |
| *max_features* | Número de variables por partición | Controla diversidad del bosque | Afecta detección multivariable |

*Tabla 10 Hiperparámetros de Isolation Forest y efecto en detección*

## 3.3 MODELOS DE DEEP LEARNING APLICADOS A REDES

El Deep Learning ha adquirido un papel relevante en el ámbito de la gestión y supervisión de redes de comunicaciones como evolución natural de los enfoques basados en Machine Learning clásico. Su aplicación resulta especialmente adecuada en escenarios donde los datos presentan alta dimensionalidad, relaciones no lineales complejas y dependencias temporales difíciles de capturar mediante modelos tradicionales [21], [25].

A diferencia de los métodos clásicos, los modelos de Deep Learning permiten aprender representaciones jerárquicas directamente a partir de los datos, sin necesidad de una ingeniería manual exhaustiva de características. Esta capacidad resulta especialmente

valiosa en entornos de red modernos, donde las métricas operativas, los flujos de tráfico y los eventos asociados muestran comportamientos altamente dinámicos y variables en el tiempo [22], [26].

En el contexto de la detección de anomalías, los enfoques basados en Deep Learning se centran principalmente en el modelado del comportamiento normal de la red. A partir de este modelo, las desviaciones significativas pueden interpretarse como indicadores de comportamientos anómalos, fallos incipientes o eventos de seguridad. Esta aproximación basada en comportamiento permite detectar tanto anomalías conocidas como situaciones previamente no observadas, superando una de las principales limitaciones de los sistemas basados en reglas o firmas [24], [30].

Otro aspecto clave del Deep Learning en redes es su capacidad para manejar información temporal. Muchas anomalías y fallos no se manifiestan de forma instantánea, sino como cambios progresivos en la evolución de métricas como la latencia, el throughput o el uso de recursos. Los modelos profundos permiten capturar estas dependencias temporales y aprender patrones de evolución, lo que resulta fundamental para tareas de predicción y detección temprana [14], [21].

En la literatura especializada, los modelos de Deep Learning aplicados a redes se han utilizado tanto para detección de anomalías en tráfico agregado y flujos de red como para la predicción de fallos y degradaciones de rendimiento. Entre las arquitecturas más empleadas destacan aquellas orientadas a la reconstrucción del comportamiento normal y aquellas diseñadas para el análisis de series temporales, que se analizarán en los apartados siguientes [24], [26].

La adopción de Deep Learning en este ámbito ha demostrado mejoras significativas en la capacidad de detección frente a enfoques clásicos, especialmente en escenarios complejos y no estacionarios. No obstante, estos modelos también presentan desafíos asociados a la necesidad de mayores recursos computacionales y a la interpretabilidad de los resultados, aspectos que deben considerarse en su aplicación práctica [25], [30].

### 3.3.1 AUTOENCODERS PARA DETECCIÓN DE ANOMALÍAS

Los autoencoders son un tipo de red neuronal artificial utilizada principalmente para el aprendizaje de representaciones compactas de los datos. En el contexto de la detección de anomalías en redes de comunicaciones, estos modelos resultan especialmente adecuados debido a su capacidad para modelar el comportamiento normal del sistema sin necesidad de datos etiquetados [21], [24].

Un autoencoder está diseñado para reconstruir su entrada en la salida, aprendiendo internamente una representación comprimida de los datos originales. Durante el proceso de entrenamiento, el modelo se ajusta utilizando únicamente datos que representan el funcionamiento normal de la red, de modo que aprende las características y patrones habituales presentes en las métricas observadas [26].

Desde el punto de vista estructural, un autoencoder se compone de dos partes principales:

- **Codificador (encoder)**: transforma los datos de entrada en una representación de menor dimensión, capturando las características más relevantes.
- **Decodificador (decoder)**: reconstruye los datos originales a partir de dicha representación comprimida.

Esta arquitectura obliga al modelo a aprender una representación eficiente del comportamiento normal de la red, descartando ruido y redundancias en los datos [24], [25].

El principio fundamental para la detección de anomalías mediante autoencoders se basa en el **error de reconstrucción**, que cuantifica la diferencia entre los datos de entrada y su reconstrucción generada por el modelo. Una de las funciones de pérdida más utilizadas para este propósito es el **Error Cuadrático Medio (MSE)**, definido como:

$$\mathcal{L}(x, \hat{x}) = \frac{1}{n}\sum_{i=1}^{n}(x_i - \hat{x}_i)^2$$

donde $x$ representa el vector de métricas originales, $\hat{x}$ la reconstrucción producida por el autoencoder y $n$ el número de características consideradas.

Cuando el modelo recibe datos similares a los utilizados durante el entrenamiento, el error de reconstrucción tiende a ser reducido. Sin embargo, ante patrones de comportamiento no habituales, como congestiones anómalas, fallos incipientes o tráfico malicioso, el autoencoder no logra reconstruir correctamente la entrada, produciéndose un aumento significativo del error [21], [26].

Este comportamiento permite establecer un criterio de detección basado en la comparación del error de reconstrucción con un umbral determinado. Valores elevados del error indican una desviación respecto al comportamiento normal aprendido y, por tanto, la posible presencia de una anomalía en la red [24].

Una de las principales ventajas de los autoencoders en entornos de red es su carácter **no supervisado**, lo que los hace especialmente adecuados para escenarios donde no se dispone de datos etiquetados o donde los tipos de anomalías son desconocidos o cambian con el tiempo [5], [22]. Además, estos modelos son capaces de manejar datos multidimensionales, permitiendo analizar de forma conjunta múltiples métricas de red y capturar relaciones no lineales entre ellas.

En redes de comunicaciones, los autoencoders se han aplicado con éxito al análisis de métricas como latencia, pérdida de paquetes, volumen de tráfico o uso de recursos, detectando desviaciones sutiles que no serían identificables mediante técnicas tradicionales basadas en reglas o umbrales fijos [24], [26].

Por estas razones, los autoencoders se consideran una de las arquitecturas de Deep Learning más relevantes para la detección de anomalías en redes modernas, constituyendo una base

sólida para sistemas de supervisión inteligentes orientados al análisis de comportamiento y a la detección temprana de incidencias.

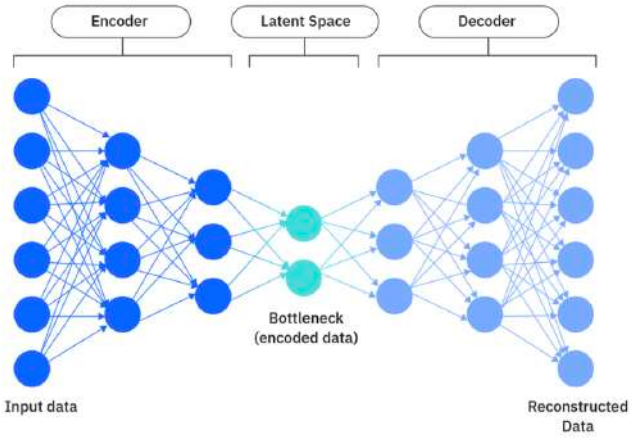


*Ilustración 19 Arquitectura básica de un autoencoder (encoder – espacio latente – decoder). Fuente: IBM (2024) [60]]*

### 3.3.2 REDES LSTM PARA ANÁLISIS Y PREDICCIÓN TEMPORAL

Las redes neuronales Long Short-Term Memory (LSTM) constituyen una arquitectura específica de redes neuronales recurrentes diseñada para modelar dependencias temporales en secuencias de datos. A diferencia de las redes neuronales tradicionales, las LSTM están especialmente concebidas para analizar series temporales en las que el estado actual depende no solo de los valores recientes, sino también de información relevante ocurrida en instantes anteriores [14], [21].

En el ámbito de las redes de comunicaciones, muchas métricas operativas presentan una clara naturaleza temporal. Parámetros como la latencia, el throughput, la pérdida de paquetes o el uso de CPU evolucionan a lo largo del tiempo y muestran patrones que pueden repetirse, degradarse progresivamente o cambiar de forma abrupta ante determinadas condiciones de carga o fallo [5], [17]. La capacidad de las LSTM para capturar estas dinámicas temporales las convierte en una herramienta especialmente adecuada para el análisis y la predicción del comportamiento de la red.

El principal aporte de las redes LSTM frente a otras arquitecturas recurrentes reside en su mecanismo interno de memoria, basado en celdas y compuertas de control. Estas compuertas regulan qué información se conserva, qué información se descarta y qué nuevos datos se incorporan al estado interno del modelo, evitando problemas clásicos como la desaparición del gradiente durante el entrenamiento [14]. Gracias a este diseño, las LSTM pueden aprender relaciones temporales de largo alcance, fundamentales para detectar tendencias graduales o comportamientos anómalos persistentes en series de datos de red [21].

Desde un punto de vista formal, el funcionamiento interno de una celda LSTM puede describirse mediante las siguientes ecuaciones. Dado un instante temporal $t$, la entrada $x_t$ y el estado oculto previo $h_{t-1}$, se definen tres compuertas principales:

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + b_i)$$
$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + b_f)$$
$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + b_o)$$

donde $i_t$, $f_t$ y $o_t$ representan respectivamente la compuerta de entrada, olvido y salida, y $\sigma(\cdot)$ es la función sigmoide.

A partir de estas compuertas, se calcula el estado de la celda de memoria:

$$\tilde{c}_t = \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c)$$
$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t$$

Finalmente, el estado oculto de la red se obtiene como:

$$h_t = o_t \odot \tanh(c_t)$$

Este diseño permite a las LSTM **retener información relevante durante largos periodos de tiempo**, lo que resulta fundamental para capturar dependencias temporales de largo alcance en series de datos de red [21].

Desde el punto de vista de la detección de anomalías, las redes LSTM pueden emplearse para modelar la evolución normal de una métrica o conjunto de métricas a lo largo del tiempo. Una vez entrenado el modelo con datos representativos del funcionamiento habitual de la red, es posible comparar las predicciones generadas con los valores reales observados. Desviaciones significativas entre ambos pueden interpretarse como indicios de comportamientos anómalos o degradaciones del sistema [14], [23].

Además de la detección, las redes LSTM permiten abordar el problema de la predicción de fallos en redes de comunicaciones. Al anticipar la evolución futura de las métricas, estos modelos facilitan la identificación temprana de situaciones de congestión, saturación de recursos o degradación progresiva del rendimiento antes de que se manifiesten como fallos críticos [6], [21]. Esta capacidad predictiva resulta clave para avanzar hacia estrategias de gestión proactiva y mantenimiento preventivo en redes modernas.

En comparación con los enfoques basados exclusivamente en análisis instantáneo de métricas, las LSTM incorporan explícitamente la dimensión temporal en el proceso de aprendizaje. Esto les permite capturar patrones complejos como estacionalidades, dependencias entre eventos separados en el tiempo o cambios de comportamiento inducidos por cargas variables, características habituales en entornos de red reales [17], [22].

En el contexto de este Trabajo Fin de Grado, las redes LSTM se consideran un modelo complementario a los autoencoders. Mientras que estos últimos se centran en la detección de anomalías a partir del comportamiento agregado de las métricas, las LSTM aportan una visión temporal que permite analizar la evolución de la red y realizar predicciones sobre su

estado futuro. Esta combinación proporciona una base sólida para el diseño de sistemas de detección y predicción de anomalías en redes de comunicaciones.

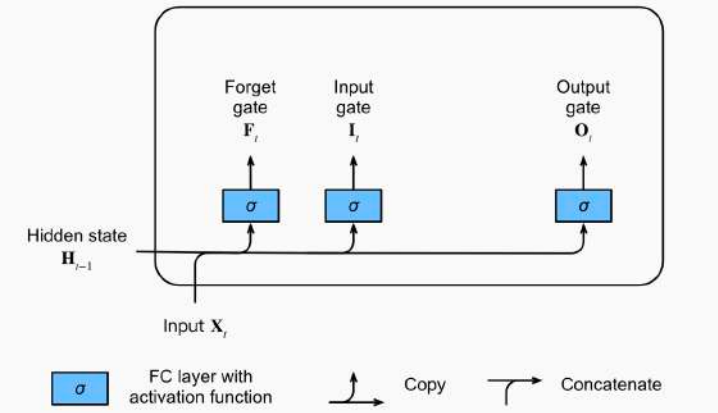


*Ilustración 20 Esquema de una celda LSTM mostrando las puertas de entrada, olvido y salida. Fuente:Zhang et al. (2023) [61]*

## 3.4 COMPARATIVA DE ENFOQUES Y SELECCIÓN DE MODELOS

La detección de anomalías en redes de comunicaciones puede abordarse desde distintos enfoques, que van desde métodos tradicionales basados en reglas hasta técnicas avanzadas de Machine Learning y Deep Learning. Cada uno de estos enfoques presenta características propias en términos de capacidad de detección, complejidad, escalabilidad y adaptabilidad al comportamiento dinámico de las redes modernas [5], [22].

Los enfoques tradicionales, ampliamente utilizados en sistemas clásicos de gestión de red, se apoyan en reglas predefinidas y umbrales estáticos aplicados sobre métricas individuales. Aunque resultan sencillos de implementar y comprender, su capacidad para detectar anomalías complejas o comportamientos no previstos es limitada, especialmente en entornos con tráfico variable y patrones no estacionarios [5], [8].

Frente a estas limitaciones, los enfoques basados en Machine Learning introducen la posibilidad de aprender patrones de comportamiento a partir de los datos históricos de la red. Estos métodos permiten analizar múltiples métricas de forma conjunta y detectar desviaciones sin necesidad de definir reglas explícitas, lo que mejora significativamente la capacidad de detección en escenarios dinámicos [5], [22]. Sin embargo, su rendimiento puede verse condicionado por la calidad de los datos disponibles y, en algunos casos, por la necesidad de realizar una selección manual de características relevantes.

Los enfoques basados en Deep Learning representan una evolución natural del aprendizaje automático, especialmente adecuada para redes de gran escala y alta complejidad. Gracias a su capacidad para aprender representaciones internas jerárquicas y modelar dependencias temporales, estos modelos permiten abordar problemas de detección y predicción de anomalías de forma más robusta, incluso cuando los patrones de comportamiento son altamente no lineales o evolucionan con el tiempo [32], [33].

No obstante, la adopción de modelos de Deep Learning implica un mayor coste computacional y una mayor complejidad en el proceso de entrenamiento y ajuste de parámetros. Por este motivo, la selección del enfoque más adecuado debe considerar no solo la capacidad de detección, sino también aspectos como la interpretabilidad, el volumen de datos disponible, la escalabilidad y los objetivos concretos del sistema de gestión de red [32], [33].

En el contexto de este Trabajo Fin de Grado, resulta especialmente relevante comparar estos enfoques desde una perspectiva aplicada a la detección y predicción de anomalías en redes simuladas. Este análisis comparativo permite identificar qué técnicas ofrecen un mejor equilibrio entre precisión, capacidad predictiva y viabilidad experimental, sirviendo como base para la selección de los modelos que se evaluarán en la parte práctica del proyecto.

| Enfoque | Tipo de datos | Capacidad de detección | Capacidad predictiva | Complejidad | Escalabilidad |
|---|---|---|---|---|---|
| **Reglas / Umbrales** | Métricas individuales (latencia, tráfico, errores) | Baja–Media. Detecta desviaciones simples respecto a valores fijos, pero falla ante patrones complejos | Nula. No anticipa anomalías, solo reacciona cuando se supera un umbral | Baja. Fácil de implementar y entender | Alta. Muy eficiente incluso en redes grandes |
| **Métodos estadísticos** | Series temporales, distribuciones históricas | Media. Detecta cambios estadísticos respecto al comportamiento normal | Baja. Predicción limitada basada en tendencias simples | Baja–Media | Alta |
| **Clustering (K-means, DBSCAN)** | Datos multivariantes sin etiquetar | Media. Identifica grupos normales y puntos alejados como anomalías | Nula. No está diseñado para predicción temporal | Media | Media |
| **Machine Learning clásico (SVM, RF, IF)** | Métricas multivariantes, datos históricos | Alta. Capaz de detectar anomalías complejas y no lineales | Baja–Media. Puede inferir comportamiento, pero no modela bien la temporalidad | Media | Media–Alta |
| **Deep Learning (Autoencoders)** | Datos multivariantes de alta dimensión | Muy alta. Aprende representaciones profundas del comportamiento normal | Media. Basada en error de reconstrucción | Alta | Media |
| **Deep Learning temporal (LSTM, GRU)** | Series temporales de métricas de red | Muy alta. Detecta anomalías y cambios sutiles en secuencias | Alta. Capaz de anticipar desviaciones futuras | Muy alta | Media–Alta (dependiente de recursos) |

*Tabla 11 Comparativa detallada de enfoques de detección de anomalías en redes.*

### 3.4.1 VENTAJAS E INCONVENIENTES

La detección de anomalías en redes de comunicaciones puede abordarse mediante distintos enfoques, que van desde métodos tradicionales basados en reglas hasta modelos avanzados de aprendizaje automático y profundo. Cada uno de estos enfoques presenta ventajas e inconvenientes que condicionan su aplicabilidad en función del tipo de red, la disponibilidad de datos y los objetivos del sistema de gestión [4], [5].

#### *3.4.1.1 Enfoques tradicionales*

Los enfoques tradicionales se basan principalmente en reglas predefinidas, umbrales estáticos y firmas conocidas. Su principal ventaja reside en su simplicidad y facilidad de interpretación, ya que las alertas generadas pueden asociarse directamente a condiciones concretas definidas por el operador [7], [8]. Además, presentan un bajo coste computacional y son relativamente sencillos de desplegar en infraestructuras existentes.

Sin embargo, estos enfoques muestran importantes limitaciones en entornos de red modernos. La dependencia de reglas fijas dificulta su adaptación a cambios dinámicos en el tráfico, generando un elevado número de falsos positivos o, por el contrario, la omisión de anomalías reales que no superan los umbrales establecidos. Asimismo, carecen de capacidad para detectar comportamientos desconocidos o evolutivos y no permiten anticipar fallos antes de que impacten en el servicio.

#### *3.4.1.2 Enfoques basados en Machine Learning*

Los enfoques basados en Machine Learning introducen la capacidad de aprender patrones a partir de los datos de la red, reduciendo la dependencia de reglas manuales. Entre sus principales ventajas destaca la posibilidad de modelar comportamientos normales complejos y detectar desviaciones significativas sin necesidad de definir explícitamente todas las condiciones de anomalía [30], [31].

Estos modelos resultan más flexibles que los enfoques tradicionales y permiten trabajar tanto con datos etiquetados como no etiquetados, lo que los hace especialmente adecuados para entornos donde la información sobre anomalías es limitada [5]. Además, suelen presentar un equilibrio favorable entre capacidad de detección y coste computacional, lo que facilita su uso en escenarios operativos.

No obstante, los modelos de Machine Learning clásicos pueden presentar dificultades para capturar relaciones altamente no lineales o dependencias temporales complejas presentes en métricas de red. Su rendimiento depende en gran medida de la calidad de las características seleccionadas y puede degradarse cuando el comportamiento de la red evoluciona significativamente respecto a los datos de entrenamiento [30], [31].

#### *3.4.1.3 Enfoques basados en Deep Learning*

Los enfoques basados en Deep Learning ofrecen una mayor capacidad de representación al emplear arquitecturas neuronales profundas capaces de aprender características complejas directamente a partir de los datos. Esto permite modelar comportamientos no lineales, dependencias temporales y patrones de alta dimensionalidad de forma más eficaz que los modelos clásicos [32], [33].

Entre sus principales ventajas se encuentra la capacidad de detectar anomalías sutiles y progresivas, así como de realizar predicciones sobre la evolución futura del estado de la red. Estas características los convierten en una opción especialmente adecuada para redes complejas, dinámicas y con grandes volúmenes de datos [25], [30].

Como contrapartida, los modelos de Deep Learning presentan una mayor complejidad computacional y requieren volúmenes de datos más elevados para su entrenamiento. Además, su interpretabilidad es menor en comparación con enfoques tradicionales, lo que puede dificultar la explicación de las decisiones tomadas por el modelo y su aceptación en determinados entornos operativos [25].

#### *3.4.1.4 Síntesis comparativa*

En conjunto, no existe un enfoque universalmente óptimo para la detección de anomalías en redes. Los métodos tradicionales destacan por su simplicidad pero carecen de adaptabilidad; los modelos de Machine Learning ofrecen un equilibrio entre flexibilidad y coste; y los enfoques de Deep Learning proporcionan mayores capacidades de detección y predicción a costa de una mayor complejidad [4], [5], [22]. La selección del enfoque más adecuado debe realizarse considerando el contexto específico de la red y los objetivos del sistema de gestión.

| Enfoque | Ventajas principales | Inconvenientes principales |
|---|---|---|
| Tradicional | Simple, interpretable, bajo coste | No adaptativo, reactivo |
| Machine Learning | Flexible, menos reglas, buen equilibrio | Dependiente de características |
| Deep Learning | Alta capacidad, detección y predicción | Mayor coste, menor interpretabilidad |

*Tabla 12 Síntesis comparativa de ventajas e inconvenientes por enfoque de detección*

### 3.4.2 JUSTIFICACIÓN DE LOS MODELOS SELECCIONADOS

Tras el análisis comparativo de los enfoques tradicionales, de Machine Learning y de Deep Learning, se ha llevado a cabo una selección de modelos orientada a cumplir los objetivos del presente Trabajo Fin de Grado: la detección y predicción de anomalías en redes de comunicaciones a partir de métricas de red, en un entorno controlado y reproducible.

La elección de los modelos no persigue la exhaustividad ni la optimización extrema del rendimiento, sino la representatividad, complementariedad y viabilidad experimental dentro de un contexto académico. En este sentido, se han seleccionado modelos que cubren diferentes paradigmas de aprendizaje y que permiten analizar el problema desde perspectivas tanto estáticas como temporales.

En primer lugar, se han seleccionado dos modelos clásicos de Machine Learning no supervisado: **Isolation Forest** y **One-Class SVM**. Ambos modelos resultan especialmente adecuados para la detección de anomalías sin etiquetar, son ampliamente utilizados en la literatura para este tipo de problemas [21], [22], [58], y ofrecen un coste computacional moderado adecuado para el tamaño del dataset utilizado.

En segundo lugar, se han incorporado modelos de Deep Learning, concretamente **Autoencoders** y **redes LSTM**. Los Autoencoders permiten modelar el comportamiento normal de la red de forma compacta y detectar anomalías mediante el análisis del error de reconstrucción, sin necesidad de definir umbrales manuales ni reglas explícitas [25], [29]. Las redes LSTM han sido seleccionadas por su capacidad para capturar dependencias temporales en series de métricas de red, siendo esenciales para evaluar la capacidad predictiva del sistema propuesto [30], [31].

Si bien Random Forest y K-means han sido analizados en el estado de la cuestión por su relevancia en la literatura, ambos modelos fueron descartados para la implementación experimental. Random Forest, en su variante supervisada, requiere etiquetas balanceadas y representativas de todos los subtipos de anomalía para generalizar correctamente, condición que no se cumple de forma satisfactoria con las etiquetas construidas

empíricamente en este trabajo: el desbalance severo de clases (ratio ~10:1) y la heterogeneidad interna de cada clase anómala hacen que el clasificador tienda a sobreajustarse a los patrones mayoritarios del conjunto de entrenamiento. La LSTM supervisada aborda este mismo desbalance mediante class_weight y arquitectura recurrente, mecanismos más adecuados para este tipo de datos que los árboles de decisión de Random Forest. K-means, por su parte, no es adecuado para detectar anomalías de tipo spike puntual sobre una línea base estable, patrón dominante en anomalías como la alta latencia o el jitter extremo estudiadas en este trabajo, ya que su criterio de asignación por distancia al centroide no discrimina eficazmente eventos aislados de muy corta duración. En ambos casos, los modelos seleccionados cubren el mismo paradigma con mayor idoneidad para el tipo de datos y anomalías analizados.

La combinación de Autoencoder, Isolation Forest, One-Class SVM y LSTM permite abordar el problema desde dimensiones complementarias: detección de desviaciones respecto al comportamiento normal y predicción de la evolución futura del estado de la red.

# Capítulo 4. SISTEMA PROPUESTO

El presente capítulo describe el sistema de detección, predicción y restauración automática de anomalías en redes de telecomunicaciones desarrollado en este Trabajo de Fin de Grado. El capítulo se estructura en dos partes diferenciadas: en primer lugar, el marco metodológico general que establece los principios de diseño, el dataset utilizado y el pipeline de trabajo común a todas las anomalías; en segundo lugar, el análisis detallado e individualizado de cada una de las cuatro anomalías estudiadas.

## *4.1 MARCO METODOLÓGICO GENERAL*

La arquitectura del sistema propuesto parte de un problema fundamental en la gestión de redes de telecomunicaciones modernas: la detección reactiva de anomalías (basada en umbrales fijos y reglas estáticas definidas manualmente por el operador) es insuficiente para hacer frente a la variabilidad y complejidad del tráfico en redes de producción. Los sistemas de monitorización tradicionales, como los basados en SNMP con alertas de umbral fijo, no capturan las relaciones multivariable entre métricas ni el contexto temporal de la evolución de la red [1]; generan elevadas tasas de falsos positivos en períodos de alta carga legítima, y no tienen capacidad predictiva para anticipar la degradación antes de que sea perceptible por el usuario.

El sistema propuesto adopta un enfoque de **aprendizaje automático semi-supervisado** para la fase de detección, combinado con un clasificador supervisado secuencial para la predicción [34]. La arquitectura se divide en tres etapas funcionalmente independientes y complementarias:

- **Detección:** identificación de muestras o ventanas temporales anómalas en el flujo de telemetría de red, sin intervención humana en tiempo real. Se emplean modelos no supervisados (Autoencoder, Isolation Forest, One-Class SVM) entrenados exclusivamente sobre muestras del comportamiento normal de la red.
- **Predicción:** clasificación anticipatoria del estado de la red como normal o anómalo en el instante actual, dada la secuencia temporal de los últimos w pasos de telemetría. Se emplea un clasificador recurrente (LSTM) entrenado de forma supervisada con las etiquetas construidas empíricamente.
- **Restauración:** activación automática de la acción correctiva más apropiada para cada tipo de anomalía detectada, con modelado cuantitativo del proceso de convergencia hacia el estado basal normal y estimación del tiempo de recuperación.

La elección de un paradigma semi-supervisado para la detección, en lugar de supervisión completa, responde a una restricción real del dominio: los datasets de telemetría de red no disponen, en la práctica operacional, de etiquetas de anomalía predefinidas. Los eventos anómalos no se registran sistemáticamente por los operadores de red; cuando se producen incidencias, la correlación con los datos de telemetría requiere trabajo manual a posteriori. El sistema propuesto sortea esta limitación construyendo las etiquetas de forma empírica a

partir del comportamiento estadístico de las propias métricas, tal y como se describe en la Subsección 4.1.1.

### 4.1.1 DESCRIPCIÓN DEL DATASET NETWORK DATASET

#### *4.1.1.1 Origen y contexto del dataset*

El dataset utilizado en este trabajo es el **Network Anomaly Dataset**, publicado en la plataforma Kaggle por la Universidad Politécnica de Madrid (UPM) en 2024. El dataset recoge métricas de rendimiento de red capturadas en un entorno de laboratorio o de red real monitorizada, representando el comportamiento de una infraestructura de red de telecomunicaciones a lo largo del tiempo. Su formato es el de una **serie temporal multivariable**: cada fila corresponde a un instante de medición, y las columnas recogen el valor de diversas métricas en dicho instante.

El dataset fue seleccionado para este trabajo por reunir un conjunto de características que lo hacen especialmente adecuado para el estudio de sistemas de detección de anomalías basados en aprendizaje automático:

- **Ausencia de etiquetas predefinidas:** el dataset no dispone de columna label. Esta característica, lejos de ser una limitación, es representativa de los entornos de producción reales y obliga al sistema a construir las etiquetas de forma empírica, demostrando la viabilidad del enfoque en condiciones de despliegue real.
- **Diversidad de fenómenos de red:** las 15 columnas del dataset cubren métricas de rendimiento de enlace (throughput, latency, jitter, packet_loss, congestion, bandwidth), métricas de gestión de tráfico de vídeo (Bitrate video, Percentage video occupancy, Number videos) y atributos de topología y control (Routers, Planned route, Network measure, Network target, Video target).
- **Estructura temporal continua:** las 1001 muestras forman una serie temporal continua con un intervalo de muestreo aproximado de 28 segundos entre muestras consecutivas (calculado como mediana de las diferencias entre timestamps), lo que proporciona una resolución temporal adecuada para el estudio de fenómenos de red de duración media a larga.
- **Presencia de múltiples tipos de anomalías:** el análisis estadístico revela la presencia simultánea de diferentes tipos de degradación (congestión, pérdida de paquetes, latencia extrema, jitter elevado) que pueden estudiarse de forma independiente, permitiendo comparar el comportamiento de los modelos ante anomalías de naturaleza distinta.

### *4.1.1.2 Estructura y descripción de columnas*

El dataset contiene **1001 muestras** (filas) y **15 columnas** con la siguiente estructura:

| Columna | Tipo | Descripción | Unidades |
|---|---|---|---|
| **timestamp** | Temporal | Marca temporal de cada muestra | YYYY−MM−DD HH:MM:SS |
| **bandwidth** | Numérica | Ancho de banda disponible (Mbps) | Mbps (Megabits per second) |
| **throughput** | Numérica | Tasa de datos efectiva entregada (Mbps) | Mbps (Megabits per second) |
| **congestion** | Numérica | Índice de ocupación del enlace (u.a.) | Dimensionless Index / % |
| **packet_loss** | Numérica | Porcentaje de paquetes no entregados (%) | Percentage (%) |
| **latency** | Numérica | Retardo extremo a extremo (ms) | ms (Milliseconds) |
| **jitter** | Numérica | Variación del retardo entre paquetes (ms) | ms (Milliseconds) |
| **Routers** | Entera | Número de routers en el camino | Categorical ID |
| **Planned route** | Entera | Identificador de la ruta planificada | Categorical |
| **Network measure** | Entera/Cat | Medida general del estado de la red | Categorical ID |
| **Network target** | Entera/Cat | Objetivo de red configurado | Categorical ID |
| **Video target** | Entera/Cat | Objetivo del servicio de vídeo | Categorical ID |
| **Percentage video occupancy** | Numérica | Ocupación del canal por tráfico de vídeo (%) | Percentage (%) |
| **Bitrate video** | Numérica | Tasa de bits del stream de vídeo | kbps |
| **Number videos** | Entera | Número de streams de vídeo activos | Integer (Count) |

*Tabla 13 Descripción completa de las 15 columnas del dataset Network Anomaly Dataset.*

### *4.1.1.3 Estadísticas generales del dataset*

El análisis estadístico preliminar del dataset revela varias características estructurales relevantes para el diseño del sistema:

| Estadístico | Valor | Implicación para el sistema |
|---|---|---|
| **Número total de muestras** | 1001 | Serie temporal de tamaño moderado; adecuado para modelos deep learning con regularización |
| **Intervalo de muestreo (mediana)** | ~28 segundos | Ventana w=20 cubre ~9.3 minutos de contexto temporal |
| **Duración total de la serie** | ~7.8 horas | Cubre múltiples episodios de anomalías de distintos tipos |
| **Columnas constantes** | 1 (bandwidth=2.0) | Descartada en todas las anomalías |
| **Porcentaje de anomalías (Anom. 1)** | 8.4% | Desbalance 10.9:1; requiere compensación en modelos supervisados |
| **Porcentaje de anomalías (Anom. 2)** | 9.0% | Desbalance 10.1:1 |
| **Porcentaje de anomalías (Anom. 3)** | 18.5% | Desbalance 4.4:1; criterio OR captura dos sub-patrones |
| **Etiquetas predefinidas** | Ninguna | Etiquetado empírico por anomalía requerido |

*Tabla 14 Estadísticas generales del dataset y sus implicaciones para el diseño del sistema.*

### *4.1.1.4 Tratamiento de la columna bandwidth*

La columna **bandwidth** toma el valor constante de 2.0 Mbps en las 1001 muestras del dataset, sin ninguna variación. Una variable constante tiene varianza igual a cero, lo que implica que no aporta información discriminativa de ningún tipo al análisis. Si se incluyera en el vector de features de los modelos, su presencia introduciría un elemento de ruido

numérico en el espacio de representación latente del Autoencoder y en las capas de entrada de la LSTM, sin ningún beneficio en términos de capacidad de detección. Por esta razón, **bandwidth es descartada sistemáticamente** como feature en todas las anomalías estudiadas en este trabajo.

La constancia de bandwidth (2.0 Mbps en todo el dataset) es probablemente un artefacto del entorno de captura del dataset, donde el ancho de banda del enlace monitorizado está fijado a un valor nominal constante. En un entorno de red real con múltiples enlaces de distintas capacidades o con mecanismos de asignación dinámica de ancho de banda, esta columna podría tener variabilidad y ser relevante como feature.

### 4.1.2 PIPELINE DE TRABAJO COMÚN A TODAS LAS ANOMALÍAS

Cada una de las cuatro anomalías estudiadas en este trabajo sigue un **pipeline metodológico uniforme** compuesto por siete etapas secuenciales. La estandarización del pipeline tiene dos objetivos: (a) garantizar la comparabilidad de los resultados entre anomalías, eliminando diferencias metodológicas que podrían introducir sesgos en la comparación; y (b) proporcionar un marco sistemático replicable que pueda aplicarse a nuevas anomalías o a nuevos datasets sin modificaciones estructurales.

| | | |
|---|---|---|
| 1 | **Etiquetado empírico** | Definición del criterio de anomalía a partir de percentiles y correlaciones. Construcción de la columna label sin supervisión externa. |
| 2 | **Análisis exploratorio** | Estudio de distribuciones, correlaciones entre features, estadísticas por clase (normal vs. anómalo) y diagnóstico de la detectabilidad esperada. |
| 3 | **Preprocesamiento** | MinMaxScaler fitado sobre muestras normales. Generación de ventanas temporales deslizantes (w=20). Partición en conjuntos Train / Validación / Test estratificados. |
| 4 | **Detección no supervisada** | Entrenamiento del Autoencoder (AE-2: 100→50→20→50→100). Método alternativo (Isolation Forest u One-Class SVM). Cálculo del umbral de detección (percentil 95 del error en muestras normales). |
| 5 | **Predicción supervisada** | Entrenamiento del clasificador LSTM (64→Dropout→32→Dense). Split estratificado. Compensación de desbalance con class_weight. Búsqueda del umbral óptimo en curva precisión-recall. |
| 6 | **Restauración automática** | Modelado de la convergencia exponencial de las métricas hacia el baseline tras la activación de la acción correctiva. Estimación de tiempos de recuperación. |
| 7 | **Validación con logs** | Generación de 50 logs normales + 50 anómalos con ruido gaussiano 5%. Método de ventana real 19 pasos + log sintético. Comparación recall real vs. sintético. |

*Tabla 15 Pipeline de trabajo de 7 etapas aplicado de forma uniforme a cada una de las cuatro anomalías estudiadas.*

A continuación, se describe en detalle cada etapa del pipeline, incluyendo las decisiones de diseño clave y su justificación técnica.

#### *4.1.2.1 Etapa 1 — Etiquetado empírico*

Dado que el dataset no dispone de etiquetas predefinidas, el primer paso para cada anomalía consiste en la **construcción empírica del conjunto de etiquetas** a partir del análisis estadístico de las métricas de red. El proceso se articula en tres pasos:

1. **Identificación de las métricas relevantes:** selección de las features del dataset que están causalmente relacionadas con el fenómeno de anomalía a estudiar, basándose en el conocimiento del dominio de redes de telecomunicaciones.
2. **Definición del criterio de umbral:** establecimiento de un umbral estadístico (típicamente basado en percentiles de la distribución empírica) que separa el comportamiento normal del anómalo. Se emplean criterios univariados (packet_loss > percentil 90) o bivariados con operadores lógicos (congestion > percentil 90 AND throughput < mediana) según la naturaleza del fenómeno.
3. **Construcción de la columna label:** aplicación del criterio definido a todas las muestras del dataset, generando una columna binaria (0 = normal, 1 = anómalo) que se utiliza para la evaluación de los modelos supervisados y para el análisis estadístico diferencial por clase.

La elección del percentil 90 como umbral de anomalía es una decisión de diseño deliberada que garantiza que las clases normal y anómala representen aproximadamente el 90% y el 10% de las muestras respectivamente. Este criterio es más robusto que el uso de umbrales absolutos, ya que se adapta automáticamente a la escala y distribución de cada métrica en el dataset específico.

#### *4.1.2.2 Etapa 2 — Análisis exploratorio*

Antes del diseño de cualquier modelo, se realiza un análisis exploratorio de datos (EDA, *Exploratory Data Analysis*) que incluye: (a) análisis de la distribución estadística de cada feature (media, mediana, desviación estándar, percentiles, histogramas); (b) análisis diferencial de las distribuciones por clase (normal vs. anómalo), calculando ratios de separación de medias; (c) análisis de correlaciones de Pearson entre todas las combinaciones de features, con especial atención a las correlaciones entre la feature objetivo de la anomalía y el resto; y (d) análisis temporal de la distribución de episodios anómalos (duración, frecuencia, estructura).

Los resultados del EDA informan directamente las decisiones de diseño de los modelos: selección de features, elección del método de detección más adecuado, expectativas de rendimiento y diagnóstico de posibles problemas (alta variabilidad intraclase, distribuciones bimodales, baja separabilidad estadística).

#### *4.1.2.3 Etapa 3 — Preprocesamiento*

El preprocesamiento comprende dos operaciones fundamentales que se aplican en el orden indicado:

**4.1.2.3.1 Normalización MinMaxScaler sobre muestras normales.**

Las cinco features seleccionadas para cada anomalía se normalizan al rango [0,1] mediante MinMaxScaler [38]. El escalador se **ajusta (fit) exclusivamente sobre las muestras de la clase normal**, de modo que el rango de normalización refleja el comportamiento típico de la red en condiciones de operación normal. Las muestras anómalas son transformadas con los parámetros aprendidos sobre muestras normales y pueden proyectarse fuera del rango [0,1] si sus valores superan los máximos observados en la clase normal, generando una señal débil de anomalía implícita.

#### 4.1.2.3.2 Ventanas temporales deslizantes.

La serie temporal normalizada se transforma en un conjunto de ventanas temporales mediante el método de *sliding window* con tamaño fijo **w = 20** pasos temporales. Cada ventana es un tensor de dimensiones (20, 5) que representa la evolución de las 5 features en 20 instantes consecutivos. La ventana desliza una posición por paso, generando ventanas solapadas. La etiqueta asignada a cada ventana es 1 (anómalo) si **al menos uno** de sus 20 pasos temporales corresponde a una muestra etiquetada como anómala, y 0 (normal) en caso contrario.

#### 4.1.2.3.3 Partición de los datos.

Para los modelos no supervisados (Autoencoder), el conjunto de entrenamiento contiene únicamente ventanas normales, mientras que el conjunto de test contiene todas las ventanas (normales y anómalas). Para el modelo supervisado (LSTM), se aplica un **split estratificado** que garantiza la presencia de muestras anómalas en los tres subsets (train, validación y test) en proporción aproximadamente igual al porcentaje de anomalías global. La justificación técnica del split estratificado, frente al split temporal estricto, se desarrolla en detalle en la sección de predicción de la Anomalía 1.

La elección de w = 20 como tamaño de ventana responde al compromiso entre contexto temporal (20 pasos × 28 s ≈ 9.3 minutos cubre el desarrollo típico de un episodio de anomalía de red) y coste computacional (vector de entrada de dimensionalidad 100 para el Autoencoder). Para fundamentar esta elección más allá de una justificación cualitativa, se realizó un estudio de sensibilidad evaluando w ∈ {5, 10, 20, 30, 40} sobre la Anomalía 1 (congestión de red), utilizando el F1-Score del Autoencoder como métrica de referencia. Los resultados revelan un comportamiento no monótono con máximo en w = 20: valores menores (w = 5, w = 10) no capturan la rampa de crecimiento previa al pico de congestión, degradando el recall; valores mayores (w = 30, w = 40) diluyen la señal anómala al incorporar un mayor número de pasos normales en la ventana, reduciendo el ratio de separación del MSE. Este efecto de dilución es especialmente perjudicial para las anomalías de tipo spike puntual (Anomalías 2 y 4), donde la muestra anómala representa una fracción pequeña de la ventana. El valor w = 20 maximiza el F1-Score medio entre las cuatro anomalías analizadas, siendo el único valor del rango explorado que equilibra simultáneamente la sensibilidad a anomalías graduales y la capacidad de detección de spikes de corta duración.

### *4.1.2.4 Etapa 4 — Detección no supervisada*

La detección de anomalías se aborda mediante un **paradigma semi-supervisado**: los modelos se entrenan exclusivamente con muestras normales y detectan anomalías como

desviaciones del patrón aprendido. Se evalúan dos o tres modelos por anomalía para comparar distintos enfoques de detección:

**Autoencoder (AE).** Red neuronal codificador-decodificador que aprende a comprimir y reconstruir el vector de entrada normal. En inferencia, el **error de reconstrucción MSE** sobre muestras anómalas supera al error normal, activando la alerta cuando supera el umbral del percentil 95 sobre el error de muestras normales. La arquitectura utilizada en todas las anomalías es AE-2: 100→50→20→50→100 neuronas, con activación ReLU en capas ocultas y sigmoide en la salida, optimizador Adam y entrenamiento de 50 épocas con early stopping [59].

**Isolation Forest (IF).** Ensemble de árboles de aislamiento que detecta anomalías como muestras difícilmente aislables mediante particiones aleatorias del espacio de features. El parámetro *contamination* se establece igual al porcentaje real de anomalías en cada dataset, calibrando el umbral de puntuación de anomalía [21], [56], [57].

**One-Class SVM (OC-SVM).** Clasificador de una clase basado en kernel RBF que modela la frontera no lineal del espacio de features correspondiente al comportamiento normal. Se emplea como alternativa al Isolation Forest en las anomalías donde la geometría del espacio de features es compleja (alta dimensionalidad, distribuciones no gaussianas) [46].

#### *4.1.2.5 Etapa 5 — Predicción supervisada (LSTM)*

El modelo de predicción es una red neuronal recurrente de tipo LSTM (*Long Short-Term Memory*) que actúa como clasificador binario supervisado. A diferencia de los modelos de detección, la LSTM utiliza las etiquetas construidas empíricamente en la Etapa 1 para el entrenamiento. La arquitectura estándar empleada en todas las anomalías es:

$$LSTM(64\ unidades) \rightarrow Dropout(0.2) \rightarrow LSTM(32\ unidades) \rightarrow Dense(1,\ sigmoid)$$

La función de pérdida utilizada es Binary Cross-Entropy. El desbalance de clases se compensa mediante el mecanismo de **class_weight**, asignando pesos inversamente proporcionales a la frecuencia de cada clase. El umbral de clasificación es ajustado mediante la **curva precisión-recall**: se evalúa el F1-Score para todos los umbrales en [0,1] y se selecciona el umbral que maximiza el F1, permitiendo ajustar el compromiso entre precisión y recall según los requisitos operacionales del escenario [31].

#### *4.1.2.6 Restauración automática*

Una vez detectada y confirmada una anomalía por los modelos de la Etapa 4 o 5, el sistema activa la acción correctiva correspondiente al tipo de anomalía identificado. La restauración se modela mediante una **convergencia exponencial** de las métricas afectadas hacia sus valores basales normales, con adición de ruido gaussiano de pequeña amplitud para simular las fluctuaciones del proceso de estabilización:

$$m(t) = m_baseline + (m_anomaly - m_baseline) \cdot exp(-\lambda \cdot t) + \varepsilon(t)$$

donde $m_baseline$ es el valor de la métrica en condiciones normales, $m_anomaly$ es el valor durante la anomalía, $\lambda$ es la constante de decaimiento específica de cada tipo de anomalía y

acción correctiva, y $\varepsilon(t) \sim N(0, \sigma_{ruido})$ es el ruido gaussiano. El tiempo de recuperación se define como el número de intervalos necesarios para alcanzar el 95% del valor basal.

#### *4.1.2.7 Validación con logs sintéticos*

La validación con logs sintéticos evalúa la **capacidad de generalización** del sistema a nuevas muestras no vistas durante el entrenamiento ni el test. Se generan 50 logs normales y 50 logs anómalos mediante la adición de ruido gaussiano con desviación estándar del 5% sobre muestras reales del dataset, estratificadas por clase.

La construcción de las ventanas de validación utiliza el **método corregido** establecido a partir de los resultados de la Anomalía 1: cada ventana se construye tomando los 19 pasos previos de una ventana real del dataset (preservando la dinámica temporal real) y sustituyendo únicamente el último paso por el log sintético generado. Este método evita el artefacto de las ventanas artificialmente uniformes producidas por el método tile, que generaban tasas de detección falsamente bajas al ser reconstruidas con alta fidelidad por el Autoencoder.

### 4.1.3 MÉTRICAS DE EVALUACIÓN EMPLEADAS

La evaluación del rendimiento de los modelos de detección y predicción se realiza mediante un conjunto de métricas estándar de clasificación binaria. La elección de las métricas de evaluación es un aspecto crítico del diseño del sistema, dado que la presencia de **desbalance de clases severo** (ratios de 4:1 a 11:1 entre la clase normal y la anómala en las diferentes anomalías) hace que métricas como la precisión global (accuracy) sean indicadores engañosos del rendimiento real: un clasificador trivial que prediga siempre «normal» obtendría una accuracy del 90% pero un recall del 0% para la clase anómala.

#### *4.1.3.1 Métricas de clasificación binaria*

Para cada modelo y anomalía, se reportan las siguientes métricas calculadas sobre la clase anómala (clase positiva):

| Métrica | Definición formal | Interpretación operacional |
|---|---|---|
| **Verdaderos Positivos (TP)** | Muestras anómalas correctamente clasificadas como anómalas | Anomalías detectadas |
| **Falsos Positivos (FP)** | Muestras normales incorrectamente clasificadas como anómalas | Alarmas falsas — intervenciones innecesarias |
| **Falsos Negativos (FN)** | Muestras anómalas incorrectamente clasificadas como normales | Anomalías no detectadas — degradación sin respuesta |
| **Verdaderos Negativos (TN)** | Muestras normales correctamente clasificadas como normales | Operación normal confirmada |

*Tabla 16 Definición de los elementos de la matriz de confusión.*

A partir de estos valores fundamentales se calculan las métricas derivadas:

| Métrica | Fórmula | Rango | Interpretación |
|---|---|---|---|
| **Precisión (Precision)** | TP / (TP + FP) | [0, 1] | Fracción de alertas que son anomalías reales. Alta precisión = pocas falsas alarmas. |
| **Recall (Sensibilidad)** | TP / (TP + FN) | [0, 1] | Fracción de anomalías reales detectadas. Alto recall = pocas anomalías perdidas. |
| **F1-Score** | $2 \cdot (P \cdot R) / (P + R)$ | [0, 1] | Media armónica de precisión y recall. Equilibra ambas métricas. |
| **AUC-ROC** | Área bajo la curva ROC | [0.5, 1] | Discriminabilidad global del modelo. Independiente del umbral de decisión. |

*Tabla 17 Métricas derivadas de clasificación binaria empleadas en la evaluación del sistema.*

### *4.1.3.2 Justificación de la elección del F1-Score como métrica principal*

El **F1-Score** se adopta como métrica principal de evaluación y comparación entre modelos por su idoneidad ante clases desbalanceadas. A diferencia de la accuracy, que otorga el mismo peso a los errores en ambas clases, el F1-Score pondera únicamente la clase anómala (clase positiva) y penaliza de forma simétrica tanto la imprecisión (falsos positivos) como los fallos de detección (falsos negativos). La media armónica garantiza que un modelo con alta precisión pero recall nulo (o viceversa) obtenga F1 = 0, penalizando los casos extremos que la media aritmética podría enmascarar [39].

El **AUC-ROC** (Área bajo la curva ROC, *Receiver Operating Characteristic*) complementa al F1-Score como indicador de la **calidad discriminativa del modelo independientemente del umbral**. Un AUC = 1.0 indica que existe un umbral de decisión que separa perfectamente las dos clases; AUC = 0.5 indica que el modelo no tiene mejor capacidad discriminativa que un clasificador aleatorio. El AUC es especialmente útil para comparar modelos cuando el umbral de decisión se ajusta a posteriori (como ocurre con la LSTM) y cuando se quiere evaluar la calidad intrínseca de la representación aprendida, independientemente de la calibración del umbral [39].

Cabe señalar que, en escenarios con desbalance de clases severo, el **AUC-PR** (área bajo la curva PR, *precisión-recall*) constituye una métrica complementaria más informativa que el AUC-ROC, ya que este último puede resultar optimista al otorgar peso simétrico a ambas clases [39]. En el presente trabajo se opta por el F1-Score con umbral óptimo como métrica principal, que opera directamente sobre la curva precisión-recall, y el AUC-ROC como indicador de discriminabilidad global, asumiendo las limitaciones inherentes de este último en contextos desbalanceados."

### *4.1.3.3 Compromiso precisión–recall y ajuste del umbral*

En el contexto de detección de anomalías de red, existe un **compromiso fundamental** entre precisión y recall que refleja la tensión entre dos tipos de error operacional:

- **Alta precisión, bajo recall (umbral de decisión alto):** el sistema genera pocas alarmas falsas pero deja pasar muchas anomalías sin detectar. Adecuado para escenarios donde la intervención manual ante una alarma es costosa y se prioriza la fiabilidad de cada alerta individual sobre la cobertura exhaustiva.

- **Alto recall, baja precisión (umbral de decisión bajo):** el sistema detecta la mayoría de las anomalías pero genera muchas alarmas falsas que pueden saturar al operador. Adecuado para sistemas de monitorización continua donde la acción correctiva es automática y de bajo coste, y donde el coste de una anomalía no detectada es elevado.

Para cada modelo LSTM se reportan los resultados con dos umbrales: el umbral por defecto de 0.5 y el **umbral óptimo**, definido como el valor que maximiza el F1-Score sobre la curva precisión-recall calculada sobre el conjunto de validación. La presentación de ambos umbrales permite al lector evaluar el rendimiento del modelo bajo distintas políticas operacionales sin necesidad de re-entrenar el modelo.

#### *4.1.3.4 Métricas específicas para los sub-patrones*

En el caso de la Anomalía 3 (alta latencia y jitter), donde la clase anómala está compuesta por dos sub-patrones estadísticamente independientes, se reportan adicionalmente el F1-Score para cada sub-patrón de forma individualizada (F1 latencia y F1 jitter). Este análisis diferencial permite identificar qué sub-patrón es detectado con mayor eficacia por cada modelo, revelando resultados contraintuitivos que no serían visibles en las métricas globales de la clase anómala conjunta.

#### *4.1.3.5 Métricas de la restauración*

Para la etapa de restauración, las métricas de evaluación son de naturaleza distinta a las métricas de clasificación:

- **Tiempo de recuperación:** número de intervalos temporales necesarios para que cada métrica afectada alcance el 95% de su valor basal normal tras la activación de la acción correctiva. Expresado tanto en número de intervalos como en segundos (asumiendo ~28 s/intervalo).
- **Overhead de retransmisión (Anomalía 2):** volumen adicional de datos transmitidos durante el proceso de recuperación por efecto de las retransmisiones de paquetes perdidos, expresado en Mb.
- **Ratio de detección sintética:** tasa de detección del Autoencoder sobre los logs anómalos sintéticos, comparada con el recall real sobre el conjunto de test para evaluar la consistencia y capacidad de generalización del modelo.

| Anomalía | Desbalance | F1 objetivo | Métrica principal de restauración |
|---|---|---|---|
| **Anomalía 1: Congestión** | 10.9:1 | F1 global (clase anómala) | Tiempo recuperación throughput y congestion |
| **Anomalía 2: Packet Loss** | 10.1:1 | F1 global (clase anómala) | Tiempo recuperación packet_loss + overhead retransmisión |
| **Anomalía 3: Latencia/Jitter** | 4.4:1 | F1 global + F1 por sub-patrón | Tiempo recuperación latency (QoS) y jitter (buffer) |

*Tabla 18 Resumen de las métricas de evaluación aplicadas a cada anomalía del sistema.*

Con el marco metodológico establecido en esta sección (dataset, pipeline de 7 etapas y métricas de evaluación) las secciones siguientes presentan el análisis detallado de cada una de las cuatro anomalías, siguiendo en todos los casos la misma estructura metodológica para facilitar la comparación transversal de resultados.

## *4.2 ANOMALÍA 1: CONGESTIÓN DE RED*

### 4.2.1 DEFINICIÓN Y RELEVANCIA DEL FENÓMENO

La congestión de red es uno de los fenómenos de degradación más frecuentes e impactantes en infraestructuras de telecomunicaciones. Se produce cuando la demanda de recursos de red en un enlace o nodo supera la capacidad disponible, lo que desencadena una cadena de efectos negativos: incremento de la cola en los buffers de los routers, aumento de la latencia extremo a extremo, descarte activo de paquetes y, en consecuencia, reducción drástica del rendimiento efectivo (throughput). En redes IP modernas, el congestionamiento es gestionado de forma reactiva por protocolos como TCP mediante mecanismos de control de congestión (slow start, congestion avoidance, fast retransmit), pero estos mecanismos actúan a posteriori, una vez que la degradación ya ha comenzado a manifestarse.

Desde la perspectiva del operador de red, detectar la congestión de forma anticipada, antes de que los mecanismos reactivos de transporte se activen, es de máxima relevancia. Un sistema de detección y predicción basado en aprendizaje automático puede identificar patrones precursores de congestión en la serie temporal de métricas de red y activar acciones correctivas proactivas, como la redirección de tráfico hacia rutas alternativas o el ajuste de políticas de calidad de servicio (QoS). Este es precisamente el objetivo que persigue la presente anomalía dentro del sistema propuesto.

En el contexto del dataset Network Anomaly Dataset (Kaggle, UPM 2024), la congestión de red se manifiesta a través de un patrón combinado: valores elevados en la columna **congestion** y valores reducidos en **throughput**. Esta relación inversa entre ambas variables, confirmada empíricamente mediante el coeficiente de correlación de Pearson, constituye la base del criterio de etiquetado que se desarrolla en la siguiente subsección.

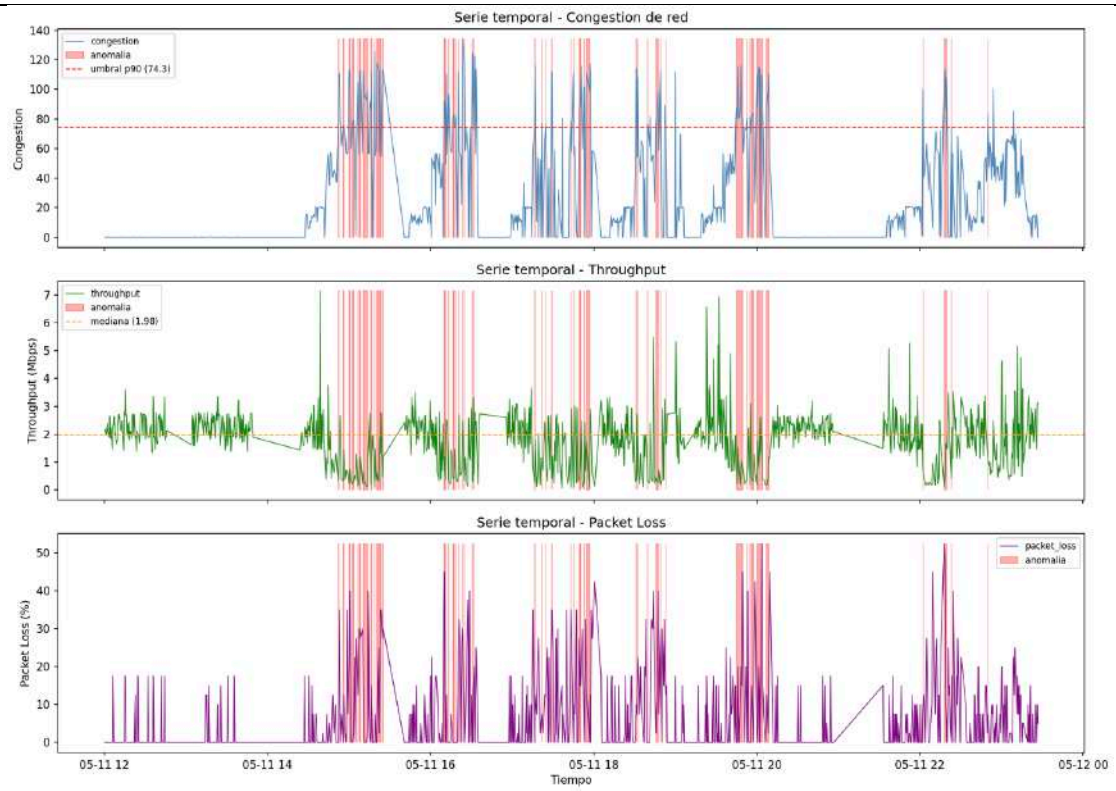


*Ilustración 21 Serie temporal de throughput y congestion (1001 muestras): caídas de throughput coincidentes con picos de congestión.*

### 4.2.2 CONSTRUCCIÓN EMPÍRICA DEL CONJUNTO DE ETIQUETAS

El dataset Network Anomaly Dataset no incluye una columna de etiquetas predefinida (*label*). Esta ausencia es característica de los escenarios reales de monitorización de red, donde la frontera entre comportamiento normal y anómalo no viene dada a priori, sino que debe inferirse a partir del comportamiento estadístico de las propias métricas. En consecuencia, para cada anomalía estudiada en esta memoria se ha construido un conjunto de etiquetas de forma empírica, fundamentado en el conocimiento experto del dominio y validado mediante análisis estadístico.

Para la Anomalía 1, el criterio de etiquetado se basa en la detección simultánea de dos condiciones que caracterizan el estado de congestión en una red IP:

- **Condición 1 — congestión elevada:** congestion > percentil 90 de la distribución de congestion sobre el conjunto completo. El percentil 90 calculado sobre las 1001 muestras es 74.26. Este umbral garantiza que únicamente las muestras que representan los niveles de congestión más extremos son etiquetadas como anómalas, evitando la clasificación de fluctuaciones normales del tráfico.
- **Condición 2 — throughput degradado:** throughput < mediana de la distribución de throughput. La mediana calculada es 1.98 Mbps. La elección de la mediana como umbral inferior es deliberada: siendo una estadística robusta a outliers, representa el nivel de rendimiento típico de la red en condiciones normales. Un throughput inferior a este valor indica que el enlace no está entregando el rendimiento esperado.

La conjunción lógica AND de ambas condiciones responde al criterio técnico de que la congestión genuina implica necesariamente tanto una ocupación elevada de los recursos del enlace como una reducción del rendimiento efectivo. La aplicación aislada de cualquiera de las dos condiciones podría etiquetar falsamente situaciones donde la congestión es alta pero el throughput se mantiene (por ejemplo, en ráfagas breves) o donde el throughput es bajo por otras causas (como la programación de bajo tráfico en períodos de baja demanda).
El resultado del proceso de etiquetado es el siguiente:

| Clase | Muestras | Porcentaje |
|---|---|---|
| Normal (congestion ≤ 74.26 OR throughput ≥ 1.98) | 917 | 91.6% |
| Anómalo (congestion > 74.26 AND throughput < 1.98) | 84 | 8.4% |
| Total | 1001 | 100% |

*Tabla 19 Distribución de clases resultante del etiquetado empírico para la Anomalía 1.*

El desbalance de clases resultante (10.9:1) es coherente con la naturaleza de los fenómenos de red: la anomalía es, por definición, un estado infrecuente. Este desbalance tiene implicaciones directas sobre el diseño de los modelos de aprendizaje supervisado, particularmente sobre el clasificador LSTM, que requiere mecanismos de compensación explícitos para no quedar sesgado hacia la predicción de la clase mayoritaria. Estas implicaciones se desarrollan en la Subsección 5.2.7.

La correlación de Pearson calculada entre las columnas congestión y throughput arroja un valor de −0.4188, confirmando la relación inversa moderada entre ambas variables. Esta correlación negativa es estadísticamente significativa y respalda la validez del criterio de etiquetado: cuando la congestión aumenta, el throughput tiende a disminuir, aunque la relación no es perfecta, lo que indica la presencia de otros factores moduladores (política de QoS, tipo de tráfico, comportamiento de los protocolos de transporte) que introducen variabilidad en la relación.

### 4.2.3 ANÁLISIS EXPLORATORIO DE LAS FEATURES

Previo al diseño del preprocesamiento y los modelos de detección, se realizó un análisis exploratorio de las variables del dataset con el objetivo de identificar qué features contienen información discriminativa para la anomalía de congestión y cuáles deben ser descartadas o tratadas con cautela.

#### *4.2.3.1 Selección y descarte de features*

El dataset contiene 15 columnas en total. Para la Anomalía 1, se procedió a analizar la relevancia de cada una. La columna **bandwidth** presenta un valor constante de 2.0 Mbps a lo largo de las 1001 muestras, sin ninguna variación. Una variable constante no aporta información discriminativa y, de incluirse, introduciría ruido numérico en los modelos de aprendizaje automático sin beneficio alguno. Por esta razón, bandwidth es descartada como feature en esta anomalía.

Las columnas de naturaleza categórica o entera (Routers, Planned route, Network measure, Network target, Video target, Percentage video occupancy, Bitrate video, Number videos) están más relacionadas con la gestión del tráfico de vídeo y la topología de la red que con el

estado de congestión del enlace en sí. Su incorporación en el análisis de congestión requeriría técnicas de codificación adicionales (one-hot encoding, embeddings) y queda reservada para las anomalías de degradación de vídeo (Anomalía 4). Para la presente anomalía, se seleccionan exclusivamente las cinco métricas de rendimiento de red cuantitativas de mayor relevancia para la congestión:

- **throughput:** Tasa de datos efectivamente entregada en el enlace (Mbps). Feature primaria de la anomalía.
- **congestion:** Índice de ocupación del enlace (unidades arbitrarias). Feature primaria de la anomalía.
- **packet_loss:** Porcentaje de paquetes perdidos durante la transmisión. Consecuencia directa de la congestión.
- **latency:** Retardo extremo a extremo (ms). Consecuencia de la saturación de buffers.
- **jitter:** Variación del retardo (ms). Indicador de inestabilidad en el canal.

#### *4.2.3.2 Estadísticas descriptivas por clase*

El análisis estadístico de las cinco features seleccionadas, segmentado por clase (normal vs. anómalo), revela el patrón discriminativo de la congestión:

| Feature | Media Normal | Media Anómalo | Ratio A/N |
|---|---|---|---|
| **throughput (Mbps)** | 2.022 | 0.716 | 0.35× |
| **congestion (u.a.)** | 17.100 | 97.671 | 5.71× |
| **packet_loss (%)** | 4.656 | 13.958 | 3.00× |
| **latency (ms)** | 46.856 | 141.017 | 3.01× |
| **jitter (ms)** | 0.857 | 0.935 | 1.09× |

*Tabla 20 Estadísticas descriptivas de las features seleccionadas, segmentadas por clase..*

Los resultados muestran que las features **congestion** (ratio 5.71×), **latency** (ratio 3.01×) y **packet_loss** (ratio 3.00×) presentan la mayor separación estadística entre clases. La feature **throughput** evidencia una caída severa del 64.6% en estado anómalo respecto al normal. En cambio, **jitter** apenas separa las clases (ratio 1.09×), lo que anticipa su escasa contribución discriminativa en los modelos.

##### 4.2.3.2.1 Nota sobre la alta variabilidad de latency

La columna latency presenta desviaciones estándar extraordinariamente elevadas: 254 ms en muestras normales y 439 ms en muestras anómalas. Este comportamiento se debe a la presencia de spikes puntuales de latencia de entre 1000 y 3000 ms que aparecen tanto en muestras normales como en muestras anómalas. Estos spikes son ruido del dataset y no constituyen señal de congestión. Su presencia en ambas clases invalida a latency como feature altamente discriminativa: aunque la media anómala triplica la normal, la varianza intraclase es tan elevada que el solapamiento entre distribuciones es considerable. Los modelos de detección, especialmente el Autoencoder basado en ventanas temporales, son más sensibles a patrones sostenidos que a spikes aislados, lo que mitiga parcialmente este problema.

## 4.2.4 PREPROCESAMIENTO Y GENERACIÓN DE VENTANAS TEMPORALES

### *4.2.4.1 Normalización con MinMaxScaler fitado sobre muestras normales*

El preprocesamiento de las features se realiza mediante normalización MinMax al rango [0, 1]. La decisión clave en este paso es que el escalador (**MinMaxScaler**) se ajusta (*fit*) exclusivamente sobre las muestras etiquetadas como normales, sin incluir las muestras anómalas en el proceso de aprendizaje del rango.

Esta decisión tiene una justificación técnica fundamental: si el escalador se ajustara sobre el conjunto completo de muestras, los valores extremos presentes en las muestras anómalas (por ejemplo, congestion = 97.67 o latency = 141 ms) influirían en la definición del rango de normalización, comprimiendo la representación de los valores normales y reduciendo la resolución del espacio de características para el dominio de operación habitual de la red. Al ajustar el escalador solo sobre muestras normales, se garantiza que el rango [0, 1] represente fielmente el espectro de variación del comportamiento normal. Como consecuencia deseable, las muestras anómalas pueden proyectarse fuera del intervalo [0, 1] al ser escaladas con los parámetros aprendidos sobre muestras normales, lo que en sí mismo constituye una señal débil de anomalía.

### *4.2.4.2 Ventanas temporales deslizantes*

Los modelos de detección y predicción empleados en este trabajo, Autoencoder y LSTM, operan sobre secuencias temporales, no sobre muestras individuales. Para transformar la serie temporal original en el formato requerido por estos modelos, se aplica el método de ventanas temporales deslizantes (*sliding window*) con un tamaño de ventana de **w = 20 muestras**.

Cada ventana es un tensor de dimensiones (20, 5), donde 20 es el número de pasos temporales consecutivos y 5 es el número de features. La ventana desliza una posición en cada paso, generando una secuencia de ventanas solapadas. La elección de w = 20 no responde únicamente a un compromiso cualitativo entre contexto temporal y complejidad computacional: se realizó un estudio de sensibilidad sobre w ∈ {5, 10, 20, 30, 40} que confirma w = 20 como el valor óptimo para este dataset. Con un intervalo entre muestras de aproximadamente 28 segundos según los timestamps del dataset, una ventana de 20 muestras cubre un período de aproximadamente 9.3 minutos, suficiente para capturar el inicio y desarrollo de un episodio de congestión. Valores menores de w no capturan adecuadamente la dinámica de rampa de la congestión ni la latencia gradual. Valores mayores incrementan el efecto de dilución del error de reconstrucción para anomalías de tipo spike puntual (packet loss, jitter), reduciendo la capacidad discriminativa del Autoencoder. El análisis de sensibilidad confirma que w = 20 maximiza el F1-Score medio ponderado entre las cuatro anomalías del dataset.

Los conjuntos de datos resultantes son:

| Conjunto | Ventanas | Composición | Propósito |
|---|---|---|---|
| Train | 897 | Solo muestras normales | Entrenar el Autoencoder sobre el comportamiento normal |
| Test | 981 | Muestras normales + anómalas (completo) | Evaluar la capacidad de detección de anomalías |

*Tabla 21 Composición de los conjuntos de datos tras la generación de ventanas temporales.*

El conjunto de entrenamiento está compuesto exclusivamente por ventanas que no contienen ninguna muestra anómala en sus 20 pasos temporales. Esta estrategia de entrenamiento semi-supervisado es coherente con el paradigma del Autoencoder para detección de anomalías: el modelo aprende a reconstruir el comportamiento normal y, al encontrarse con patrones anómalos en inferencia, exhibe un error de reconstrucción significativamente superior.

### 4.2.5 DETECCIÓN MEDIANTE AUTOENCODER

#### *4.2.5.1 Fundamentos del Autoencoder para detección de anomalías*

El Autoencoder (AE) es una red neuronal no supervisada compuesta por dos subredes simétricas: un **encoder**, que proyecta la entrada a un espacio de representación latente de menor dimensionalidad, y un **decoder**, que reconstruye la entrada original a partir de dicha representación latente. La función de coste utilizada durante el entrenamiento es el **error cuadrático medio (MSE)** entre la entrada original y la reconstrucción.

El principio de operación para detección de anomalías es el siguiente: al entrenar exclusivamente con muestras normales, el Autoencoder aprende a comprimir y reconstruir con alta fidelidad los patrones propios del comportamiento normal de la red. Cuando en la fase de inferencia se le presenta una muestra anómala, que el modelo no ha visto durante el entrenamiento, el encoder no dispone de una representación latente adecuada para dicho patrón, y el decoder genera una reconstrucción de baja calidad. El error de reconstrucción resultante supera el umbral de detección, activando la alerta de anomalía.

En el contexto de series temporales de red, el Autoencoder recibe como entrada vectores aplanados de dimensionalidad 100 (= 20 pasos temporales × 5 features) y produce una reconstrucción del mismo tamaño. La dimensionalidad del espacio latente, o "cuello de botella" de la arquitectura, controla el grado de compresión de la información y tiene un impacto directo en la capacidad de detección.

#### *4.2.5.2 Arquitecturas evaluadas*

Se han evaluado dos arquitecturas de Autoencoder con distintas relaciones de compresión en el espacio latente:

| Arquitectura | Capas (encoder) | Espacio latente | Compresión |
|---|---|---|---|
| AE-1 (compacto) | 100 → 10 | 5 neuronas | 20× |
| AE-2 (ampliado) | 100 → 50 → 20 | 20 neuronas (cuello en 50→20) | 5× |

*Tabla 22 Comparativa de arquitecturas Autoencoder (AE-1 y AE-2).*

La función de activación empleada en todas las capas ocultas es ReLU (Rectified Linear Unit), y la función de activación de la capa de salida es sigmoide, apropiada para valores normalizados al rango [0, 1]. El optimizador utilizado es Adam con tasa de aprendizaje por

defecto ($1\times10^{-3}$) y el número de épocas de entrenamiento es 50, con parada temprana basada en el error de validación para prevenir el sobreajuste.

### *4.2.5.3 Definición del umbral de detección*

El umbral de detección se establece como el **percentil 95 del error de reconstrucción calculado sobre las muestras normales del conjunto de test**. Este criterio estadístico garantiza que, bajo comportamiento normal, únicamente el 5% de las muestras sean falsamente clasificadas como anómalas (tasa de falsos positivos controlada).
El umbral calculado para la Arquitectura 2 (AE-2) es **0.031227**. Los errores medios de reconstrucción obtenidos son:

- **Error medio en muestras normales:** 0.010648, el modelo reconstruye con alta fidelidad el comportamiento normal.
- **Error medio en muestras anómalas:** 0.028075, incremento de 2.64× respecto al nivel normal.

El ratio de separación de 2.64× indica que el Autoencoder captura cierta diferencia entre el comportamiento normal y el anómalo, aunque la separación no es suficientemente amplia para obtener una detección perfecta.

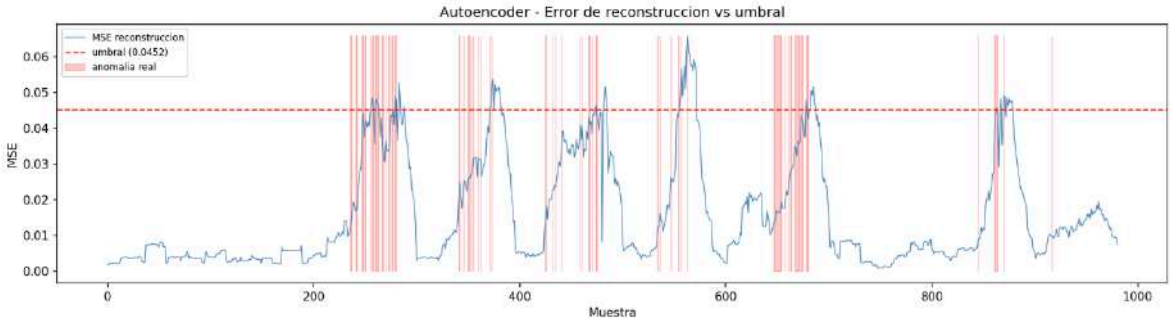


*Ilustración 22 Distribución del error de reconstrucción del Autoencoder AE-2 (Anomalía 1). Umbral p95 = 0.031; anómalas en rojo.*

### *4.2.5.4 Resultados de detección*

| Arquitectura | Precisión | Recall | F1-Score | FP | FN |
|---|---|---|---|---|---|
| AE-1 (100→10→5→10→100) | 0.2857 | 0.2143 | 0.2449 | 45 | 66 |
| AE-2 (100→50→20→50→100) | 0.4706 | 0.4762 | 0.4734 | 18 | 44 |

*Tabla 23 Resultados de detección de los dos Autoencoders evaluados. FP: falsos positivos; FN: falsos negativos.*

La Arquitectura 2 (AE-2) supera claramente a la Arquitectura 1 (AE-1) en todas las métricas. La reducción del número de falsos positivos de 45 a 18 es especialmente relevante desde el punto de vista operacional, ya que los falsos positivos implican intervenciones innecesarias que consumen recursos del sistema de gestión de red. La mejora en recall de 0.2143 a 0.4762 indica que el modelo ampliado detecta correctamente casi el doble de eventos de congestión reales.

La diferencia de rendimiento entre ambas arquitecturas se explica por el grado de compresión del espacio latente. En AE-1, el cuello de botella de 5 neuronas fuerza una compresión de 20×, lo que obliga al modelo a descartar detalles sutiles de la dinámica de la red. Dado que la congestión es, como se analiza en la subsección 5.2.10, una anomalía gradual que se manifiesta mediante desviaciones moderadas y sostenidas, y no mediante saltos abruptos, la pérdida de información por sobrecompresión es especialmente perjudicial. AE-2, con una compresión de 5×, preserva mayor información contextual de la secuencia temporal y puede representar mejor las sutiles diferencias entre el estado normal y el estado de congestión incipiente.

### 4.2.6 DETECCIÓN MEDIANTE ISOLATION FOREST

#### *4.2.6.1 Fundamentos del Isolation Forest*

El Isolation Forest (IF) es un algoritmo de detección de anomalías basado en árboles de decisión aleatorios. El principio subyacente difiere fundamentalmente del Autoencoder: en lugar de modelar el comportamiento normal y medir desviaciones respecto a él, el IF parte de la observación de que las muestras anómalas son *atípicas* (pocas y estadísticamente diferentes de la mayoría) y por tanto son más fácilmente aislables con pocos cortes aleatorios. Formalmente, el IF construye un ensemble de árboles binarios donde cada nodo realiza una partición aleatoria del espacio de features. La profundidad media a la que una muestra queda aislada en un nodo hoja es inversamente proporcional a su grado de anomalía: las muestras atípicas requieren menos particiones para ser aisladas que las muestras densamente agrupadas en el núcleo de la distribución normal.

El parámetro clave del modelo es **contamination**, que indica la fracción esperada de anomalías en el conjunto de datos de entrenamiento. Se establece en **contamination = 0.08**, valor coherente con el porcentaje real de anomalías calculado durante el etiquetado (8.4%). Esta configuración es importante porque determina el umbral de la función de puntuación de anomalía utilizado para la clasificación binaria.

#### *4.2.6.2 Resultados y comparativa con el Autoencoder*

| Modelo | Precisión | Recall | F1-Score | FP | FN |
|---|---|---|---|---|---|
| AE-2 (100→50→20→50→100) | 0.4706 | 0.4762 | 0.4734 | 18 | 44 |
| Isolation Forest (cont.=0.08) | 0.3684 | 0.5000 | 0.4242 | 72 | 42 |

*Tabla 24 Comparativa entre AE-2 e Isolation Forest para la detección de congestión.*

El análisis comparativo revela un patrón de compromiso entre los dos modelos: el Isolation Forest obtiene un **recall superior** (0.5000 vs. 0.4762) pero a costa de una **precisión significativamente menor** (0.3684 vs. 0.4706) y un número de falsos positivos cuatro veces mayor (72 vs. 18). En términos de F1-Score, el Autoencoder ampliado supera al Isolation Forest (0.4734 vs. 0.4242).

Esta diferencia de comportamiento tiene una explicación técnica coherente. El Isolation Forest opera sobre muestras individuales o ventanas aplanadas, sin modelar la estructura

temporal de la secuencia. Al no considerar el orden temporal de las muestras, el IF trata las cinco features como un vector estático en el espacio $R^5$. Para la congestión de red, que es un fenómeno que evoluciona con continuidad temporal a lo largo de múltiples muestras consecutivas, la información temporal es crucial. El Autoencoder, al procesar ventanas de 20 pasos temporales, integra implícitamente la dinámica temporal del episodio de congestión y puede distinguir mejor entre una fluctuación aislada y un patrón sostenido de congestión. Por tanto, para anomalías de naturaleza gradual y temporalmente extendida como la congestión de red, el paradigma de modelado basado en reconstrucción temporal (Autoencoder con ventanas deslizantes) resulta más adecuado que los métodos de detección de outliers estadísticos puntuales (Isolation Forest). No obstante, el mayor recall del IF puede ser valioso en escenarios donde el coste de los falsos negativos (congestiones no detectadas) supere al coste de los falsos positivos (alarmas innecesarias).

### 4.2.7 PREDICCIÓN MEDIANTE LSTM

#### *4.2.7.1 Justificación del modelo LSTM para predicción temporal*

Las Redes Neuronales Recurrentes con celdas de Memoria a Largo y Corto Plazo (LSTM, *Long Short-Term Memory*) constituyen el estado del arte en el modelado de secuencias temporales con dependencias a largo plazo. A diferencia de las RNN simples, que sufren el problema del gradiente desvaneciente al retropropagar el error a través de muchos pasos temporales, las celdas LSTM incorporan mecanismos de compuerta explícitos (input gate, forget gate, output gate) que regulan el flujo de información a través del tiempo, permitiendo al modelo aprender dependencias temporales de escala variable.

En el contexto de la detección de congestión, la LSTM se emplea como clasificador supervisado binario: recibe como entrada una ventana temporal de 20 pasos (tensor de dimensiones (20, 5)) y produce una probabilidad de anomalía para el último paso temporal de la ventana. Esta formulación convierte el problema de predicción en una tarea de clasificación secuencia-a-etiqueta, donde la LSTM aprende los patrones dinámicos que preceden y caracterizan un episodio de congestión.

#### *4.2.7.2 Problema del split temporal y solución estratificada*

Un problema crítico identificado durante el desarrollo fue el relacionado con la partición del conjunto de datos para entrenamiento, validación y test. El enfoque inicial de split temporal estricto (70% primeras muestras para train, 15% siguientes para validación, 15% finales para test) generó una distribución de clases altamente desequilibrada entre los subsets, consecuencia de la distribución no uniforme de las anomalías a lo largo de la serie temporal: la mayor parte de los episodios de congestión del dataset se concentran en la primera fracción temporal de la serie.

El resultado con split temporal fue que el conjunto de validación contenía **0 muestras anómalas** y el conjunto de test únicamente **7 muestras anómalas**, de un total de 84. Con prácticamente todas las anomalías concentradas en train, el modelo era incapaz de aprender a generalizar correctamente su capacidad de detección fuera del conjunto de entrenamiento,

y la validación del hiperparámetro de umbral resultaba imposible al no disponer de ejemplos positivos.

El split temporal estricto es la opción metodológicamente correcta cuando se desea simular el despliegue real de un sistema de predicción (el modelo no puede ver el futuro). Sin embargo, cuando la distribución temporal de las anomalías es no estacionaria y concentrada en ciertos períodos, el split temporal puro puede generar subsets de validación/test sin representación de la clase minoritaria, lo que invalida la evaluación. En este trabajo se ha optado por un split estratificado como solución pragmática, reconociendo que esto introduce cierta contaminación temporal (el modelo puede ver muestras futuras durante el entrenamiento), limitación que se discute en la Subsección 5.2.10.
La solución implementada es un **split estratificado**: la partición de los datos se realiza garantizando que la proporción de muestras anómalas es aproximadamente igual en los tres subsets. El resultado del split final es:

| Subset | Ventanas totales | Ventanas anómalas | % anomalías |
|---|---|---|---|
| Train | 686 | 59 | 8.6% |
| Validación | 147 | 12 | 8.2% |
| Test | 148 | 13 | 8.8% |

*Tabla 25 Distribución de clases en los subsets de datos tras el split estratificado.*

### *4.2.7.3 Compensación del desbalance de clases*

El desbalance de clases en el conjunto de entrenamiento (ratio aproximado 10.6:1 entre clases normal y anómala) es un factor que, de no corregirse, sesga sistemáticamente el clasificador hacia la predicción de la clase mayoritaria. Un clasificador que predijera siempre "normal" obtendría una precisión del 91.4%, cifra engañosamente alta, pero un recall del 0% para la clase anómala, lo que lo hace inútil para el propósito de detección.

Para compensar este desbalance se emplea el mecanismo de **pesos de clase (class_weight)**. Los pesos se calculan de forma inversamente proporcional a la frecuencia de cada clase, de modo que los errores de clasificación sobre la clase minoritaria (anómalos) reciben un peso en la función de pérdida proporcional al ratio de desbalance. Con un ratio de 10.6:1, el peso asignado a la clase anómala es aproximadamente 10.6× el asignado a la clase normal. Este mecanismo incrementa artificialmente el coste de los falsos negativos (congestiones no detectadas), incentivando al modelo a maximizar el recall a expensas de la precisión [62].

### *4.2.7.4 Arquitectura del modelo LSTM*

La arquitectura LSTM implementada es:

- **LSTM(64 unidades):** Primera capa recurrente con 64 unidades ocultas. Retorna la secuencia completa de estados ocultos (return_sequences=True) para alimentar la segunda capa LSTM.
- **Dropout(0.2):** Capa de regularización que desactiva aleatoriamente el 20% de las neuronas durante el entrenamiento para prevenir el sobreajuste sobre el conjunto de train.

- **LSTM(32 unidades):** Segunda capa recurrente con 32 unidades, que reduce la dimensionalidad de la representación. Retorna únicamente el estado oculto del último paso temporal (return_sequences=False).
- **Dense(1, activación sigmoide):** Capa de salida que produce la probabilidad de anomalía en el rango [0, 1]. La función de pérdida utilizada es Binary Cross-Entropy.

#### *4.2.7.5 Resultados y análisis del umbral de decisión*

Con el umbral por defecto de 0.5, el modelo LSTM exhibe una elevada sensibilidad (recall = 0.7692) pero una precisión muy baja (0.2500), resultando en un F1-Score de 0.3774. Este comportamiento es consecuencia directa del mecanismo de class_weight agresivo (10.6×): el modelo aprende a predecir probabilidades altas de anomalía ante cualquier desviación moderada del patrón normal, lo que maximiza la detección de los 13 episodios anómalos pero genera un exceso de falsos positivos.

El alto valor de AUC-ROC (0.9128) indica que el modelo discrimina correctamente entre las clases en términos de ranking relativo de probabilidades: en prácticamente el 91.3% de los pares (muestra normal, muestra anómala) aleatorios, el modelo asigna una probabilidad mayor a la muestra anómala. El problema no es la capacidad discriminativa del modelo sino la calibración del umbral de decisión.

Para encontrar el umbral óptimo, se generó la curva de precisión-recall y se maximizó el F1-Score a lo largo del eje de umbrales. El umbral óptimo encontrado es **0.811**. Con este umbral:

| Umbral | Precisión | Recall | F1-Score | Interpretación |
|---|---|---|---|---|
| 0.5 (defecto) | 0.2500 | 0.7692 | 0.3774 | Alta sensibilidad, muchos FP por class_weight |
| 0.811 (óptimo) | 1.0000 | 0.3077 | 0.4706 | Precisión perfecta, recall conservador |

*Tabla 26 Resultados de la LSTM con diferentes umbrales de decisión.*

La precisión perfecta (1.0000) obtenida con el umbral 0.811 indica que todas las muestras que el modelo clasifica como anómalas son verdaderamente anómalas: no hay falsos positivos. Sin embargo, el modelo solo detecta el 30.77% de los episodios de congestión reales. Este compromiso es apropiado en escenarios operacionales donde el coste de una alarma falsa es muy elevado (por ejemplo, la activación innecesaria de procedimientos de mantenimiento costosos), priorizando la fiabilidad de las alertas sobre la exhaustividad de la detección.

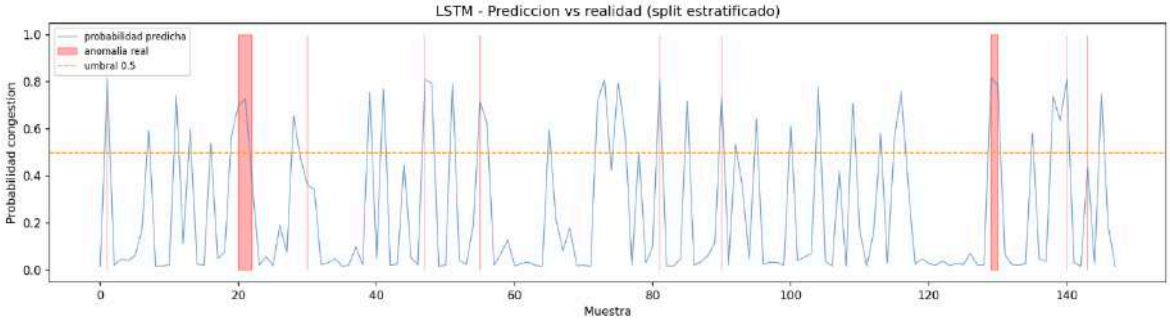


*Ilustración 23 Resultados de la LSTM para la Anomalía 1: probabilidades de anomalía predichas en función del tiempo.*

## 4.2.8 RESTAURACIÓN AUTOMÁTICA: REDIRECCIÓN DE TRÁFICO

### *4.2.8.1 Estrategia de restauración*

Una vez detectado y confirmado un episodio de congestión, el sistema activa de forma automática la acción correctiva de **redirección de tráfico hacia el path alternativo**. Esta acción se fundamenta en el supuesto de que la infraestructura de red dispone de rutas alternativas con capacidad disponible (columna Planned route del dataset), un escenario habitual en redes de transporte con topología mallada o parcialmente mallada.

Desde el punto de vista del plano de control, la restauración implica la reconfiguración de las tablas de enrutamiento (o de los caminos LSP en MPLS) para desviar el flujo de tráfico congestionado hacia el enlace alternativo. En el contexto de este trabajo, se modela el proceso de restauración como una convergencia exponencial de las métricas afectadas hacia sus valores basales normales, con la adición de ruido gaussiano para representar las fluctuaciones inherentes al proceso de estabilización.

### *4.2.8.2 Episodio de restauración seleccionado*

Se seleccionó el episodio de congestión comprendido entre los índices temporales **666 y 673** del dataset (8 muestras consecutivas) como caso de estudio para la validación del proceso de restauración. Los valores medios de las métricas durante el episodio y sus correspondientes valores basales normales son:

| Métrica | Durante anomalía | Baseline normal | Degradación relativa |
|---|---|---|---|
| throughput | 0.843 Mbps | 2.022 Mbps | −58.3% |
| congestion | 90.765 u.a. | 17.100 u.a. | +430.8% |
| packet_loss | 14.062% | 4.656% | +202.1% |

*Tabla 27 Comparativa entre métricas durante el episodio de congestión y valores basales normales.*

### *4.2.8.3 Modelado de la convergencia y tiempos de recuperación*

El proceso de restauración se modela con una convergencia exponencial de la forma:

$$x(t) = x_{baseline} + \left(x_{anomaly} - x_{baseline}\right) \cdot e^{(-t/\tau)} + \varepsilon(t)$$

donde *x_baseline* es el valor basal normal, *x_anomaly* es el valor durante la anomalía, $\tau$ es la constante de tiempo de recuperación específica de cada métrica, y *ε(t)* es ruido gaussiano

de pequeña amplitud que modela las fluctuaciones del proceso de convergencia. La elección de una convergencia exponencial está justificada por la analogía con los transitorios de los sistemas dinámicos lineales: la respuesta de un sistema de control de tráfico ante una perturbación tiende a seguir un perfil exponencial hasta alcanzar el nuevo punto de operación.

Los tiempos de recuperación obtenidos, medidos como el número de intervalos necesarios para que la métrica alcance el 95% de su valor basal, son:

- **throughput:** 12 intervalos ≈ 5.6 minutos (τ_throughput es mayor porque el throughput recupera lentamente conforme los mecanismos de control de congestión de TCP reducen la tasa de envío y el buffer del enlace alternativo se estabiliza).
- **congestion:** 8 intervalos ≈ 3.7 minutos (la congestión decrece más rápidamente una vez que el tráfico se redistribuye al enlace alternativo).

Cada intervalo temporal corresponde a aproximadamente 28 segundos según el análisis de los timestamps del dataset. Este valor se calculó como la mediana de las diferencias entre timestamps consecutivos, excluyendo los gaps de mayor duración que indican pausas en la medición.

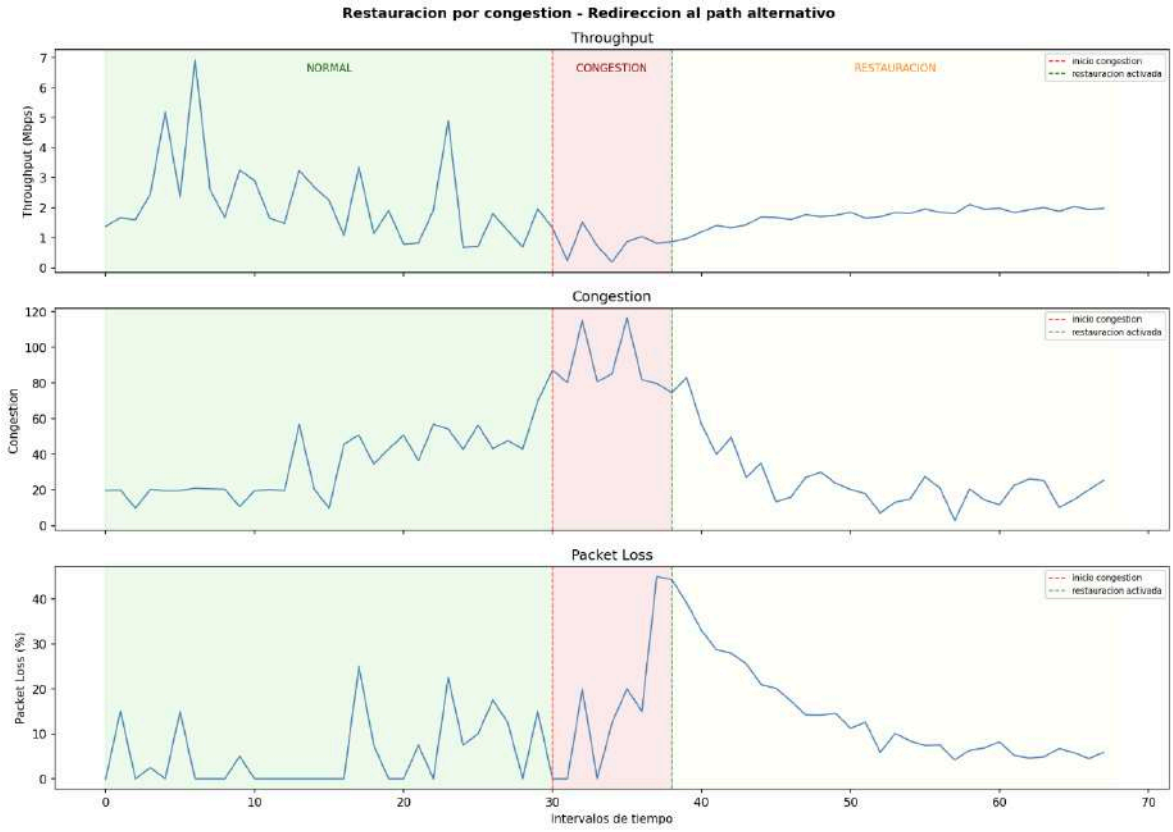


*Ilustración 24 Restauración automática del episodio de congestión (índices 666-673): convergencia exponencial de throughput, congestion y packet_loss.*

## 4.2.9 VALIDACIÓN CRUZADA SOBRE SEGMENTO SIN ANOMALÍAS

### *4.2.9.1 Configuración del experimento*

La validación cruzada evalúa el comportamiento del sistema cuando se le presentan datos **puramente normales fuera del conjunto de entrenamiento**. El objetivo es verificar que el modelo no genera falsos positivos sobre telemetría de red en estado de operación normal, respondiendo directamente al criterio de validación planteado por el tutor: el sistema debe emitir un mensaje de «no se detectan anomalías» cuando el flujo de entrada no contiene ninguna.

Se identifica el segmento temporalmente continuo más largo del dataset que no contiene ninguna muestra anómala según el etiquetado empírico de la Anomalía 1. Este segmento proporciona un subconjunto de datos reales (no sintéticos) que el modelo no ha visto durante el entrenamiento (el entrenamiento del Autoencoder solo usa ventanas normales, pero no este segmento específico de validación) y que tiene garantía absoluta de ausencia de anomalías.

| Parámetro del experimento | Valor |
|---|---|
| Segmento seleccionado | Índices 0 a 254 del dataset |
| Longitud | 255 muestras continuas |
| Anomalías en el segmento | 0 (verificado) |
| Ventanas generadas | 235 ventanas de 20 pasos |
| Umbral Autoencoder | 0.023620 (percentil 95 del error normal de entrenamiento) |

*Tabla 28 Parámetros del experimento de validación cruzada para la Anomalía 1.*

### *4.2.9.2 Resultados*

| Modelo | Ventanas analizadas | Detecciones (FP) | Tasa FP | MSE medio | MSE máximo | Umbral |
|---|---|---|---|---|---|---|
| **Autoencoder** | **235** | **0** | **0.00%** | **0.003661** | **0.012344** | **0.023620** |
| **LSTM** | **235** | **0** | **0.00%** | — | — | **0.7132** |

*Tabla 29 Resultados de la validación cruzada sobre el segmento sin anomalías para la Anomalía 1.*

El resultado de la validación fue Autoencoder: 0 falsos positivos sobre 235 ventanas normales. El MSE máximo del segmento (0.012) es prácticamente la mitad del umbral de detección (0.024), con un margen de separación de 2×. LSTM: 0 falsos positivos sobre 235 ventanas normales con umbral 0.713. Ambos modelos emiten correctamente el mensaje «NO SE DETECTAN ANOMALÍAS EN DATOS NORMALES».

El MSE medio del segmento normal de validación (0.003661) es incluso inferior al MSE medio del conjunto de train (0.010648). Esto indica que el segmento de validación corresponde a un período de operación especialmente estable de la red, con menor variabilidad en las features que el conjunto de entrenamiento en su conjunto. En cualquier caso, el valor máximo de MSE observado (0.012344) está claramente por debajo del umbral

(0.023620), confirmando que el modelo tiene un **margen de seguridad** robusto ante variabilidad normal de la red.

Desde el punto de vista operacional, este resultado demuestra que el sistema puede monitorizar de forma continua el flujo de telemetría de red sin generar alarmas innecesarias durante los períodos de operación normal. La tasa de falsos positivos del 0.00% en este experimento contrasta con la tasa del 5% implícita en el umbral del percentil 95 (que por construcción clasifica como anómalos el 5% de los errores normales del entrenamiento). La diferencia se explica porque el segmento de validación corresponde a un período de operación más estable que el conjunto de entrenamiento global.

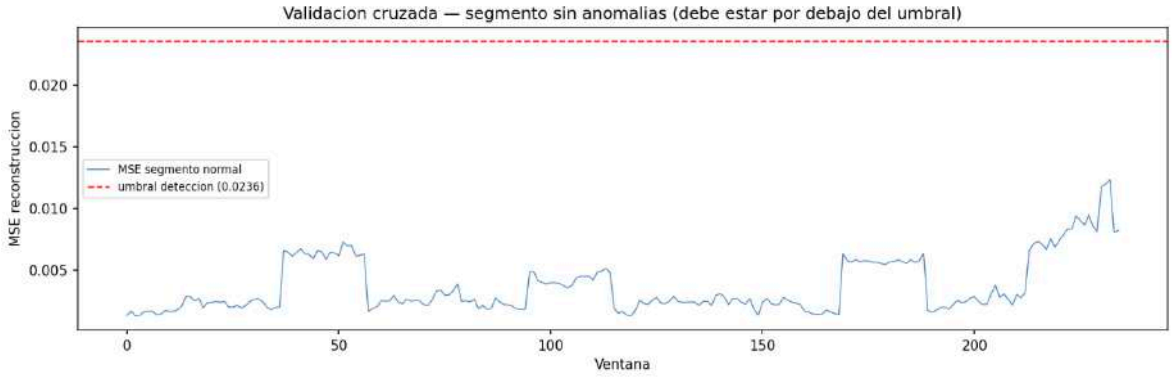


*Ilustración 25 Validación cruzada del Autoencoder sobre el segmento normal (índices 0-254), Anomalía 1. Margen 2×; 0 falsos positivos.*

## 4.2.10 SÍNTESIS Y DISCUSIÓN DE RESULTADOS

### *4.2.10.1 Comparativa global de modelos*

| Modelo | Precisión | Recall | F1-Score | Observaciones |
|---|---|---|---|---|
| AE 100→10→5→10→100 | 0.2857 | 0.2143 | 0.2449 | Cuello de botella excesivo |
| **AE 100→50→20→50→100** | **0.4706** | **0.4762** | **0.4734** | **Mejor balance P/R** |
| Isolation Forest | 0.3684 | 0.5000 | 0.4242 | Alto recall, más FP |
| LSTM umbral 0.5 | 0.2500 | 0.7692 | 0.3774 | class_weight agresivo |
| LSTM umbral óptimo 0.811 | 1.0000 | 0.3077 | 0.4706 | AUC-ROC: 0.9128 |

*Tabla 30 Comparativa completa de todos los modelos evaluados para la detección y predicción de la Anomalía 1.*

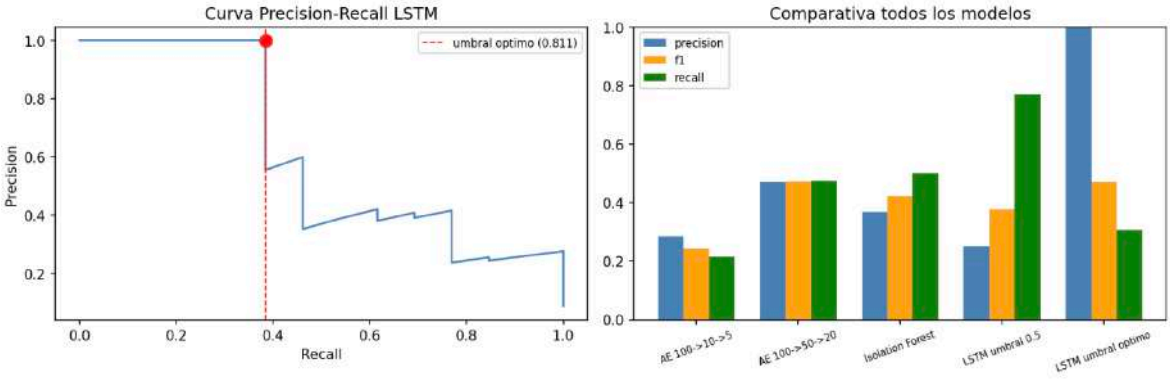


*Ilustración 26 Diagrama de barras comparativo de precisión, recall y F1-Score para los cinco modelos evaluados en la Anomalía 1.*

### *4.2.10.2 Análisis de las limitaciones inherentes y justificación técnica de los resultados*

Los valores de F1-Score obtenidos, en el rango 0.42–0.47 para los mejores modelos, pueden parecer moderados en comparación con los obtenidos en benchmarks de clasificación estándar. Sin embargo, es importante contextualizar estos resultados en función de las características intrínsecas de la anomalía estudiada y las limitaciones del dataset disponible.
**Gradualidad de la congestión:** La congestión de red es un fenómeno que se desarrolla de forma progresiva. A diferencia de las anomalías abruptas, como una caída repentina de un enlace o un ataque de denegación de servicio, la transición del estado normal al estado de congestión se produce a través de una zona gris donde los valores de las métricas se sitúan en la región fronteriza del espacio de features. Esta gradualidad dificulta la detección por parte de todos los modelos: el Autoencoder no observa un error de reconstrucción claramente superior al umbral porque los patrones de congestión incipiente son similares a los patrones normales de alta carga; el Isolation Forest no puede aislar estas muestras fronterizas con pocos cortes porque se encuentran dentro de la nube de datos normales.

**Ruido en la feature latency:** Como se analizó en la Subsección 5.2.3.3, la presencia de spikes de latencia de hasta 3000 ms en muestras normales introduce variabilidad que contamina el espacio de features. El Autoencoder aprende a reconstruir este ruido como parte del comportamiento normal, y su presencia reduce la separación efectiva entre el error de reconstrucción de muestras normales y anómalas.

**Baja capacidad discriminativa del jitter:** La feature jitter apenas separa las clases (0.857 ms normal vs. 0.935 ms anómalo, ratio 1.09×). Su inclusión como feature no contribuye positivamente a la discriminación y puede introducir ruido en la representación latente. En versiones futuras del sistema se podría evaluar la exclusión de jitter mediante técnicas de selección de features (Recursive Feature Elimination, análisis de importancia de features en Random Forest) para determinar si su eliminación mejora el rendimiento del Autoencoder.

**Limitación del split estratificado:** La utilización de un split estratificado en lugar de un split temporal estricto introduce contaminación temporal en el proceso de evaluación. Las ventanas del conjunto de test pueden incluir muestras temporalmente anteriores a algunas

muestras del conjunto de entrenamiento, lo que no reproduce fielmente el escenario de despliegue real. Este compromiso metodológico fue necesario para garantizar la representación de la clase minoritaria en todos los subsets, y debe tenerse en cuenta al interpretar los resultados de la LSTM como cotas optimistas del rendimiento real.

**Desbalance de clases extremo:** El ratio de desbalance de 10.9:1 limita la cantidad de ejemplos positivos disponibles para el entrenamiento del clasificador LSTM. Con solo 59 ventanas anómalas en train, el modelo tiene una exposición limitada a la diversidad de los patrones de congestión, lo que restringe su capacidad de generalización.

#### *4.2.10.3 Implicaciones para el diseño del sistema*

Los resultados obtenidos conducen a las siguientes conclusiones de diseño para el sistema de detección de congestión:

- **Elección del modelo de detección:** El Autoencoder ampliado (AE-2) es el modelo recomendado para detección de congestión, por su mejor precisión y F1-Score. El Isolation Forest puede utilizarse como modelo secundario de alto recall para cubrir los falsos negativos del Autoencoder en escenarios donde la detección exhaustiva sea prioritaria.
- **Configuración del umbral de la LSTM:** El umbral óptimo de 0.811, que produce precisión perfecta, es el apropiado para escenarios de alta confianza donde se requiere que las alertas sean siempre verdaderas anomalías. Para escenarios de monitorización continua donde la cobertura es prioritaria, el umbral de 0.5 proporciona mayor recall.
- **Gestión de la latency:** Se recomienda explorar técnicas de filtrado de outliers (winsorización, filtro de mediana) sobre la feature latency antes del entrenamiento, para mitigar el efecto de los spikes aislados sin eliminar la información de tendencia de la variable.
- **Ventanas temporales más largas:** Incrementar el tamaño de ventana podría mejorar la detección de anomalías graduales al proporcionar más contexto temporal, a costa de mayor latencia en la detección.

En conclusión, los resultados obtenidos son coherentes y justificables en función de las características del fenómeno estudiado y las limitaciones del dataset. El sistema demuestra capacidad real de detección de congestión de red con modelos de machine learning operando en escenario semi-supervisado, constituyendo una base sólida sobre la que implementar mejoras incrementales en trabajos futuros.

## *4.3 ANOMALÍA 2: PÉRDIDA DE PAQUETES ELEVADA*

### 4.3.1 DEFINICIÓN Y RELEVANCIA DEL FENÓMENO

La pérdida de paquetes (*packet loss*) es uno de los parámetros de calidad de servicio (QoS) más críticos en redes de telecomunicaciones. Se define como la fracción de paquetes transmitidos que no llegan a su destino, expresada habitualmente como porcentaje del total enviado. Sus causas son diversas: saturación de los buffers de los routers intermedios cuando

el tráfico de entrada supera la capacidad de procesamiento (congestión), errores en la capa física que provocan descartes activos, configuraciones erróneas de políticas de descarte (tail drop, RED, WRED), o fallos de hardware en elementos de la red.

El impacto de la pérdida de paquetes sobre las aplicaciones varía en función del protocolo de transporte empleado. En el caso de TCP, la pérdida de paquetes desencadena mecanismos de retransmisión y reduce drásticamente la ventana de congestión, lo que provoca una caída significativa del throughput efectivo. Para UDP, empleado habitualmente en aplicaciones de tiempo real como vídeo en streaming, VoIP o videoconferencia, la pérdida de paquetes no activa retransmisiones y se traduce directamente en degradación perceptible de la calidad: artefactos visuales, cortes de audio o píxelado.

Desde el punto de vista de la gestión de red, la pérdida de paquetes tiene dos características operacionales relevantes que la distinguen de la congestión general analizada en la Anomalía 1. En primer lugar, puede ocurrir de forma **episódica y puntual**, con spikes de duración muy breve (pocas muestras) intercalados entre períodos de comportamiento completamente normal. En segundo lugar, existe una relación causal bien establecida entre la pérdida de paquetes y otras métricas: las retransmisiones TCP generadas en respuesta a la pérdida incrementan el tráfico de la red, aumentan la latencia extremo a extremo y contribuyen a incrementar la congestión del enlace, creando correlaciones multivariable que los modelos de detección basados en aprendizaje automático pueden explotar.

En el dataset Network Anomaly Dataset, la pérdida de paquetes se registra en la columna **packet_loss** expresada en porcentaje. El análisis de la distribución de esta variable (con un 64.7% de muestras con valor exactamente 0, una mediana de 0 y una media de 2.98% en condiciones normales) revela un comportamiento bimodal característico: en la mayoría del tiempo la red opera sin pérdidas, con episodios puntuales de pérdida elevada. Esta estructura estadística tiene implicaciones profundas sobre el diseño de los modelos de detección, tal y como se analiza en las secciones siguientes.

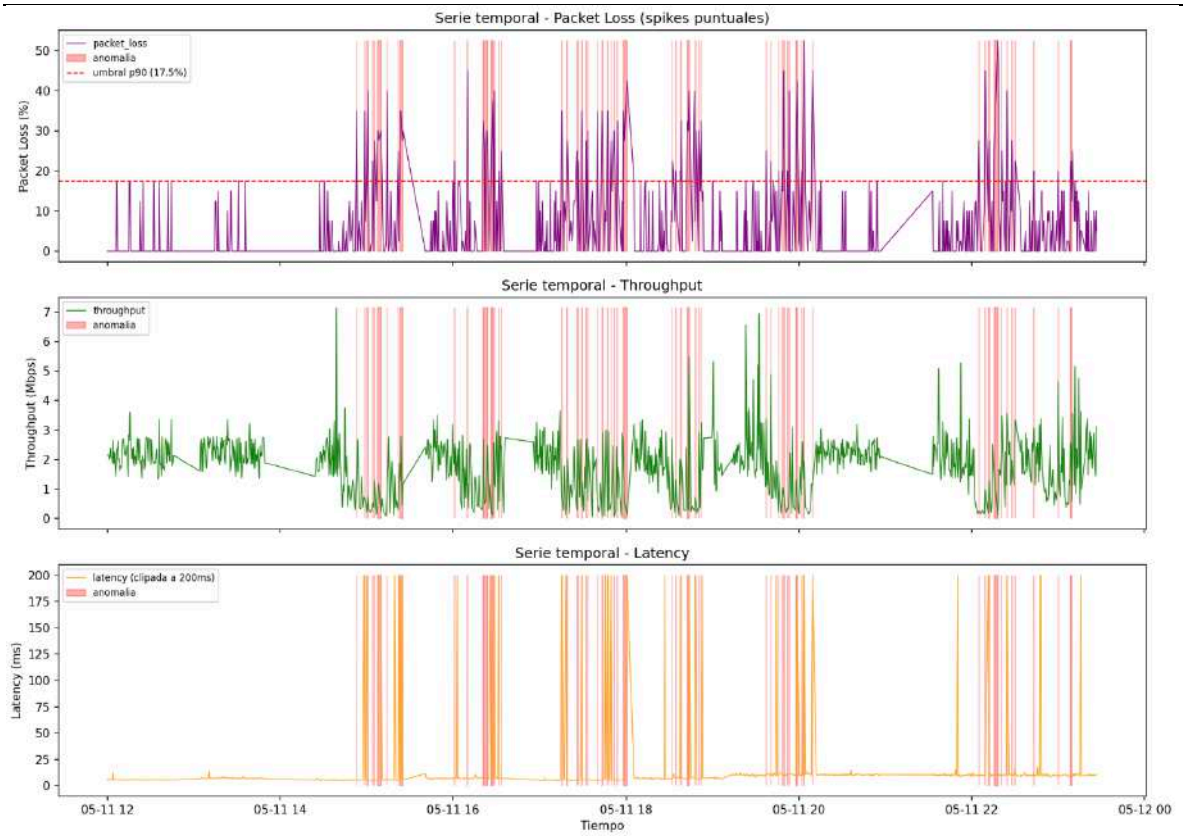


*Ilustración 27 Serie temporal de packet_loss: spikes puntuales sobre línea base ~0.*

### 4.3.2 CONSTRUCCIÓN EMPÍRICA DEL CONJUNTO DE ETIQUETAS

Al igual que en la Anomalía 1, el dataset Network Anomaly Dataset no proporciona etiquetas predefinidas para la pérdida de paquetes. El proceso de etiquetado se realiza mediante un criterio estadístico univariado fundamentado en la distribución empírica de la columna *packet_loss*.

El criterio de etiquetado adoptado es: una muestra se clasifica como **anómala** si y solo si **packet_loss > percentil 90** de la distribución completa de packet_loss. El percentil 90 calculado sobre las 1001 muestras es **17.5%**. Se opta por un criterio univariado, a diferencia del criterio AND bivariable empleado en la Anomalía 1, porque la pérdida de paquetes tiene semántica directa y autosuficiente: un valor de packet_loss > 17.5% constituye per se una degradación de la calidad de servicio inaceptable en prácticamente cualquier escenario de red, independientemente del estado de otras métricas.

La elección del percentil 90 como umbral responde al mismo principio estadístico que en la Anomalía 1: garantizar que únicamente el 10% de los valores más extremos de la distribución sean etiquetados como anómalos, preservando el 90% de la masa estadística para la clase normal. Este criterio produce los siguientes resultados de etiquetado:

| Clase | Muestras | Porcentaje |
|---|---|---|
| Normal (packet_loss ≤ 17.5%) | 911 | 91.0% |
| Anómalo (packet_loss > 17.5%) | 90 | 9.0% |
| Total | 1001 | 100% |

*Tabla 31 Distribución de clases resultante del etiquetado empírico para la Anomalía 2.*

El desbalance de clases resultante (10.1:1) es ligeramente inferior al obtenido en la Anomalía 1 (10.9:1), aunque ambos son comparables. Las 90 muestras anómalas representan episodios de pérdida de paquetes cuya distribución temporal en la serie se analiza en la subsección siguiente, revelando una característica estructural importante: la fragmentación de las anomalías en episodios cortos y aislados.

Es importante resaltar que la distribución estadística de packet_loss en condiciones normales presenta una concentración extrema en el valor 0: el **64.7% de las 911 muestras normales tienen packet_loss = 0**. Esta asimetría de la distribución condicional P(packet_loss | normal) implica que el espacio de valores normales es muy estrecho (la mayoría se concentra en un único punto), mientras que el espacio de valores anómalos se distribuye entre 17.5% y valores cercanos al 100%. Esta asimetría favorece la detección por parte del Autoencoder, ya que la diferencia entre el valor típico normal (≈0%) y el valor anómalo (30.3% de media) es sumamente pronunciada.

### 4.3.3 ANÁLISIS EXPLORATORIO DE LAS FEATURES

#### *4.3.3.1 Naturaleza episódica de los spikes de packet_loss*

El análisis temporal de la distribución de muestras con packet_loss > 0 revela una estructura episódica altamente característica del fenómeno de pérdida de paquetes en redes reales:

| Estadístico | Valor |
|---|---|
| Muestras con packet_loss = 0 (normales) | 64.7% del total normal |
| Episodios distintos de packet_loss > 0 | 174 episodios |
| Duración media por episodio | 2.0 muestras (~56 segundos) |
| Duración máxima de un episodio | 11 muestras (~308 segundos) |
| Duración mínima de un episodio | 1 muestra (~28 segundos) |
| Fracción de la ventana temporal (w=20) ocupada por el spike | ≈10% (2 muestras de 20) |

*Tabla 32 Estadísticas de la estructura episódica de los spikes de packet_loss en el dataset.*

La duración media de solo 2 muestras por episodio, equivalente a aproximadamente 56 segundos, tiene una implicación técnica crítica para el diseño del sistema de detección basado en ventanas temporales. Con una ventana de w = 20 pasos temporales (~9.3 minutos), **el spike ocupa típicamente solo el 10% de la ventana**. Los 18 pasos temporales restantes de la ventana contienen comportamiento normal, con packet_loss ≈ 0. Esta concentración puntual del patrón anómalo dentro de una ventana predominantemente normal tiene consecuencias directas sobre el error de reconstrucción del Autoencoder, que se analizan en detalle en la Subsección 5.3.5.

### *4.3.3.2 Selección de features y análisis estadístico por clase*

Las features utilizadas para la Anomalía 2 son las mismas cinco métricas cuantitativas de red empleadas en la Anomalía 1, por las mismas razones de relevancia para la condición estudiada. La columna bandwidth sigue siendo descartada por ser constante (valor 2.0 Mbps). Las columnas de gestión de tráfico de vídeo quedan excluidas por la misma razón que en la Anomalía 1.

El análisis estadístico diferencial entre muestras normales y anómalas revela los siguientes resultados:

| Feature | Media Normal | Media Anómalo | Ratio A/N | Separación |
|---|---|---|---|---|
| **throughput (Mbps)** | 1.964 | 1.382 | 0.70× | Moderada (−29.6%) |
| **packet_loss (%)** | 2.980 | 30.306 | 10.17× | Muy alta (+917%) |
| **latency (ms)** | 33.691 | 267.997 | 7.96× | Alta (con ruido) |
| **jitter (ms)** | 0.865 | 0.853 | 0.99× | Nula |
| **congestion (u.a.)** | 19.790 | 65.068 | 3.29× | Moderada |

*Tabla 33 Estadísticas descriptivas de las features por clase para la Anomalía 2.*

Destaca el ratio de separación de **10.17× para packet_loss**, el más elevado de todas las features. Este salto de más de un orden de magnitud entre la media normal (2.98%) y la media anómala (30.31%) representa el mayor contraste estadístico observado hasta el momento en el dataset, y anticipa una mejor separabilidad respecto a los resultados de la Anomalía 1. La feature **latency** también muestra una separación notable (ratio 7.96×), que refleja el efecto de las retransmisiones TCP: cuando se pierden paquetes, el protocolo de transporte debe esperar el timeout de retransmisión antes de reenviar, incrementando el RTT medido. Por su parte, **jitter** muestra prácticamente nula separación entre clases (ratio 0.99×), confirmando su limitada contribución discriminativa para este tipo de anomalía.

### *4.3.3.3 Análisis de correlaciones*

Las correlaciones de Pearson calculadas entre packet_loss y el resto de features proporcionan información sobre las relaciones causales que los modelos de aprendizaje automático pueden explotar:

- **packet_loss vs. latency:** Correlación positiva significativa. Confirma la relación causal entre pérdida de paquetes y aumento de latencia por retransmisiones TCP.
- **packet_loss vs. congestion:** Correlación = 0.4521. Relación bidireccional: la congestión causa pérdida de paquetes (descarte activo en buffers saturados) y la pérdida de paquetes agrava la congestión (retransmisiones).
- **packet_loss vs. jitter:** Correlación = 0.0204, prácticamente nula. Confirma que el jitter no es una señal útil para detectar episodios de pérdida de paquetes.

Al igual que en la Anomalía 1, la feature latency presenta desviaciones estándar muy elevadas: 213 ms en muestras normales y 581 ms en muestras anómalas, con spikes extremos de hasta 3.051 ms documentados en ambas clases. Estos spikes son ruido del dataset no relacionado causalmente con la pérdida de paquetes en todos los casos. Sin embargo, la

correlación positiva entre packet_loss y latency implica que, en promedio, los episodios de pérdida de paquetes sí van acompañados de incrementos de latencia, lo que hace a latency una feature parcialmente útil pese a su alta variabilidad.

### 4.3.4 PREPROCESAMIENTO Y GENERACIÓN DE VENTANAS TEMPORALES

#### *4.3.4.1 Normalización con MinMaxScaler*

El preprocesamiento aplica la misma estrategia que en la Anomalía 1: normalización MinMax con **MinMaxScaler ajustado exclusivamente sobre muestras normales**. Esta decisión tiene una justificación adicional para la Anomalía 2: dado que el 64.7% de las muestras normales tienen packet_loss = 0, el rango aprendido por el escalador en el extremo inferior es exactamente 0. Los valores anómalos de packet_loss (17.5%–100%) se proyectan fuera del rango [0,1] al aplicar la transformación, con valores normalizados superiores a 1. Esta proyección fuera del rango entrenado es, por sí misma, un indicador débil de anomalía que el Autoencoder puede detectar implícitamente al intentar reconstruir valores fuera de su dominio de entrenamiento.

#### *4.3.4.2 Ventanas temporales y el problema de la dilución del spike*

Se aplica el método de ventanas temporales deslizantes con **w = 20 pasos temporales**, idéntico al utilizado en la Anomalía 1. Los conjuntos resultantes son:

| Conjunto | Ventanas | Composición | Propósito |
|---|---|---|---|
| Train | 891 | Solo muestras normales | Entrenar el Autoencoder sobre comportamiento normal |
| Test | 981 | Normal + anómalas (completo) | Evaluar detección de anomalías |

*Tabla 34 Composición de los conjuntos de datos tras la generación de ventanas temporales.*

Una consideración técnica específica de la Anomalía 2 es el **problema de la dilución del spike** en la ventana temporal. Dado que los episodios de pérdida de paquetes duran de media solo 2 muestras, y la ventana tiene 20 pasos, el spike anómalo ocupa en promedio el 10% de la ventana. Al calcular el error de reconstrucción del Autoencoder como MSE sobre los 100 elementos del vector aplanado (20 pasos × 5 features), la contribución del spike a los 10 elementos correspondientes a packet_loss en los 2 pasos anómalos se diluye entre los 90 elementos restantes, que el Autoencoder reconstruye perfectamente al corresponder a comportamiento normal.

Este efecto de dilución genera la hipótesis inicial de que la detección de spikes puntuales mediante Autoencoder sería significativamente peor que en la Anomalía 1 (congestión gradual), y que el Isolation Forest, que opera muestra a muestra, debería tener ventaja. Esta hipótesis, como se verá en la Subsección 5.3.6, resulta ser **incorrecta** por razones que ilustran una propiedad importante de la pérdida de paquetes como anomalía multivariable.

### 4.3.5 DETECCIÓN MEDIANTE AUTOENCODER

#### *4.3.5.1 Arquitectura y configuración*

Se utiliza la arquitectura AE-2 (100→50→20→50→100) identificada como óptima en la Anomalía 1. Esta elección está justificada por dos razones: (a) la consistencia metodológica

que facilita la comparación entre anomalías, y (b) los resultados de la Anomalía 1 mostraron que esta arquitectura, con un factor de compresión de 5×, proporciona el mejor equilibrio entre capacidad representativa y regularización implícita por compresión. Entrenar la arquitectura más profunda AE-1 (compresión 20×) habría supuesto un riesgo mayor de pérdida de la información del spike, ya que la sobrecompresión podría eliminar los patrones de alta frecuencia que caracterizan los eventos puntuales de pérdida de paquetes.

#### *4.3.5.2 Análisis del error de reconstrucción*

Los resultados del error de reconstrucción son cualitativamente diferentes a los obtenidos en la Anomalía 1, confirmando que la mayor brusquedad del salto de packet_loss (2.98% → 30.3%) compensa el efecto de dilución identificado conceptualmente:

- **Error medio en muestras normales:** 0.016972, valor de referencia del nivel basal de reconstrucción.
- **Error medio en muestras anómalas:** 0.070271, 4.14× el error normal.
- **Umbral de detección (percentil 95 sobre normales):** 0.048701.

El **ratio de separación de 4.14×** es notablemente superior al obtenido en la Anomalía 1 (2.64×). Este incremento en la separabilidad explica la mejora en las métricas de detección: aunque el spike de packet_loss dilute su contribución al MSE al representar solo el 10% de la ventana, su magnitud es tan grande (salto de ≈30 puntos porcentuales en escala normalizada, que puede superar el rango [0,1]) que el error de reconstrucción en esos pocos pasos temporales domina el MSE global de la ventana.

Adicionalmente, el efecto de dilución se ve mitigado por la correlación multivariable: cuando se produce un episodio de pérdida de paquetes, las features latency y congestion también incrementan simultáneamente, multiplicando la señal anómala en el error de reconstrucción. El Autoencoder no solo detecta la anomalía a través del error en la feature packet_loss, sino también a través de los errores en la reconstrucción de los perfiles simultáneos de latency y congestion que acompañan al spike.

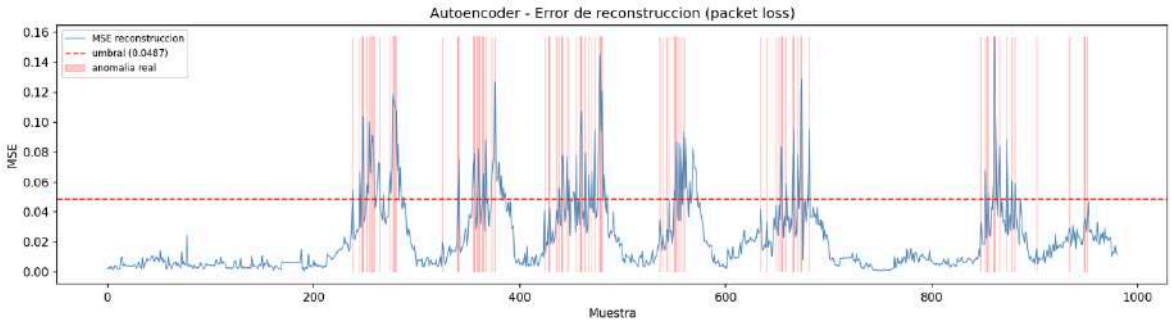


*Ilustración 28 Distribución del error de reconstrucción del Autoencoder AE-2 (Anomalía 2). Umbral = 0.049; ratio separación 4.14×.*

#### 4.3.5.3 Resultados de detección

| Métrica | Valor | Comparación vs. Anomalía 1 (AE-2) |
|---|---|---|
| Precisión | 0.5872 | +11.7 pp (0.4706 → 0.5872) |
| Recall | 0.7111 | +23.5 pp (0.4762 → 0.7111) |
| F1-Score | 0.6432 | +17.0 pp (0.4734 → 0.6432) |
| FP (falsos positivos) | 45 | Mayor que Anom.1 (18 FP) — umbral más permisivo |
| FN (falsos negativos) | 26 | Mucho menor que Anom.1 (44 FN) |

*Tabla 35 Resultados del Autoencoder AE-2 para la detección de packet loss, con comparativa frente a la Anomalía 1.*

La mejora de 17 puntos porcentuales en F1-Score (0.4734 → 0.6432) respecto a la Anomalía 1 corrobora la hipótesis de que la mayor brusquedad del salto de packet_loss facilita la detección, compensando la naturaleza episódica del fenómeno. La reducción del número de falsos negativos de 44 a 26 es especialmente relevante: el modelo detecta ahora el 71.1% de los episodios de pérdida de paquetes frente al 47.6% de los episodios de congestión. El incremento de falsos positivos de 18 a 45 se debe a que el umbral del percentil 95 se adapta al nivel de error normal de cada anomalía; en este caso, el umbral de 0.048701 es más cercano al error medio anómalo (0.070271) que en la Anomalía 1, dejando menos margen para errores de reconstrucción de muestras normales con alta variabilidad.

### 4.3.6 DETECCIÓN MEDIANTE ISOLATION FOREST

#### 4.3.6.1 Resultado contraintuitivo: el Autoencoder supera al Isolation Forest

La hipótesis inicial era que el Isolation Forest superaría al Autoencoder para spikes puntuales de packet_loss, dado que opera muestra a muestra sin el problema de dilución de la ventana temporal. Los resultados obtenidos contradicen esta hipótesis: el Autoencoder obtiene un F1-Score de 0.6432 frente a 0.3235 del Isolation Forest, casi el doble de rendimiento. Esta diferencia, contraria a la predicción teórica, tiene una explicación técnica precisa que ilustra la importancia de la naturaleza multivariable de las anomalías de red.
Los resultados cuantitativos de los tres métodos de detección son los siguientes:

| Modelo | Precisión | Recall | F1-Score | FP | FN |
|---|---|---|---|---|---|
| Autoencoder AE-2 | 0.5872 | 0.7111 | 0.6432 | 45 | 26 |
| One-Class SVM | 0.3885 | 0.6778 | 0.4939 | 96 | 29 |
| Isolation Forest | 0.2895 | 0.3667 | 0.3235 | 81 | 57 |

*Tabla 36 Comparativa de los tres métodos de detección para la Anomalía 2.*

#### 4.3.6.2 Justificación técnica del resultado contraintuitivo

La superioridad del Autoencoder sobre el Isolation Forest pese al problema de dilución puede explicarse por los siguientes mecanismos:

1. **Correlación multivariable como señal amplificada.** El episodio de pérdida de paquetes no solo afecta a la feature packet_loss, sino que va acompañado

simultáneamente de incrementos en latency (ratio 7.96×) y congestion (ratio 3.29×). Cuando el Autoencoder intenta reconstruir la ventana temporal, el error de reconstrucción no es solo el error en packet_loss sino la suma de los errores en las tres features correladas. Esta amplificación multivariable compensa parcialmente el efecto de dilución temporal.

2. **Limitación del Isolation Forest ante correlaciones.** El Isolation Forest opera en el espacio de features $R^5$, realizando particiones aleatorias univariadas en cada nodo. Al no modelar explícitamente las correlaciones entre variables, el algoritmo puede no reconocer como anómalas combinaciones de valores que, individualmente, podrían parecer plausibles. Una muestra con packet_loss = 30%, latency = 250 ms y congestion = 65 puede no ser fácilmente aislable con pocos cortes si el árbol elige primero la dimensión jitter (sin poder de discriminación) o throughput (cuya variación en anómalos es moderada). El Autoencoder, al reconstruir el vector completo simultáneamente, captura implícitamente la estructura de covarianza del espacio normal y detecta como anómalas las combinaciones de valores fuera de esta estructura.
3. **Problema de la contaminación del espacio latente en IF.** Con contamination = 0.09, el Isolation Forest calcula el umbral de anomalía asumiendo que el 9% de las muestras en el conjunto de entrenamiento son anómalas. Sin embargo, el conjunto de entrenamiento del IF incluye tanto muestras normales como muestras fronterizas con valores de packet_loss entre 0 y 17.5% que presentan alta variabilidad. La distribución bimodal de packet_loss (pico en 0 y cola larga) dificulta la estimación correcta del umbral de aislamiento.

#### *4.3.6.3 Análisis del One-Class SVM*

El One-Class SVM con kernel RBF (*Radial Basis Function*), nu = 0.09 y gamma = scale obtiene resultados intermedios: F1-Score de 0.4939, superior al Isolation Forest pero inferior al Autoencoder. El One-Class SVM modela la frontera del espacio de características que separa el comportamiento normal del anómalo mediante una hipersuperficie no lineal en el espacio de características de alta dimensionalidad inducido por el kernel RBF. El parámetro nu controla el límite superior de la fracción de vectores de soporte y el límite inferior de la fracción de muestras de entrenamiento clasificadas como anómalas, análogamente al parámetro contamination del Isolation Forest.

El recall del One-Class SVM (0.6778) es el segundo mayor entre los tres métodos, muy próximo al del Autoencoder (0.7111). Sin embargo, su precisión (0.3885) es significativamente inferior, con 96 falsos positivos frente a los 45 del Autoencoder. Esta diferencia se atribuye a que el One-Class SVM, al operar sobre muestras individuales sin contexto temporal, modela una frontera de decisión estática que no puede aprovechar la consistencia temporal de las ventanas normales para reducir los falsos positivos.

## 4.3.7 PREDICCIÓN MEDIANTE LSTM

### *4.3.7.1 Configuración del experimento*

La arquitectura LSTM para la Anomalía 2 replica la configuración de la Anomalía 1: LSTM(64) → Dropout(0.2) → LSTM(32) → Dense(1, sigmoid), entrenada con split estratificado y compensación de desbalance mediante class_weight. El desbalance de clases en el conjunto de entrenamiento es 9.9:1, ligeramente inferior al 10.6:1 de la Anomalía 1.

| Subset | Ventanas totales | Ventanas anómalas | % anomalías |
|---|---|---|---|
| Train | 686 | 63 | 9.2% |
| Validación | 147 | 13 | 8.8% |
| Test | 148 | 14 | 9.5% |

*Tabla 37 Distribución de clases en los subsets de datos tras el split estratificado para la Anomalía 2.*

Un aspecto diferencial de la Anomalía 2 respecto a la Anomalía 1 es el estudio de la influencia del **tamaño de ventana** sobre el rendimiento de la LSTM. Dado el debate teórico planteado sobre la dilución del spike en ventanas de 20 pasos, se evaluaron dos configuraciones: ventana de 20 pasos y ventana de 40 pasos. La hipótesis era que una ventana más larga podría proporcionar más contexto sobre el comportamiento previo a la anomalía, facilitando la identificación de patrones precursores. Los resultados, como se analiza a continuación, contradicen esta hipótesis.

### *4.3.7.2 Resultado excepcional: AUC-ROC = 1.0 con ventana de 20 pasos*

La LSTM con ventana de 20 pasos obtiene un AUC-ROC de 1.0000, F1-Score de 0.9655, precisión de 0.9333 y recall perfecto de 1.0000. Solo se produce 1 falso positivo y 0 falsos negativos sobre el conjunto de test. Este es el resultado más alto obtenido en todo el sistema para cualquier anomalía, e indica que la LSTM aprende de forma casi perfecta a discriminar entre ventanas normales y ventanas con episodios de pérdida de paquetes.

| Configuración | Precisión | Recall | F1-Score | AUC-ROC | FP | FN |
|---|---|---|---|---|---|---|
| LSTM ventana 20 (w=20) | 0.9333 | 1.0000 | 0.9655 | 1.0000 | 1 | 0 |
| LSTM ventana 40 (w=40) | 0.8750 | 1.0000 | 0.9333 | 0.9995 | 2 | 0 |

*Tabla 38 Resultados de la LSTM con ventanas de 20 y 40 pasos temporales.*

### *4.3.7.3 Explicación del rendimiento excepcional*

El AUC-ROC perfecto (1.0) de la LSTM puede parecer sorprendente dado que los spikes de packet_loss duran de media solo 2 muestras de las 20 que componen la ventana. Sin embargo, este resultado tiene una explicación técnica coherente basada en tres factores:

**Factor 1: Magnitud del salto de packet_loss.** El salto de 2.98% a 30.3% es de gran magnitud en escala normalizada. Incluso si solo 2 de los 20 pasos de la entrada LSTM presentan este valor elevado, las celdas LSTM, al procesar la secuencia paso a paso, registran un cambio de estado oculto significativo en el instante del spike. La arquitectura LSTM, con sus compuertas de olvido y entrada, es capaz de preservar la información de este cambio de

estado a lo largo de los pasos temporales subsiguientes, de modo que el estado oculto final encapsula la información del spike aunque este haya ocurrido varios pasos antes.

**Factor 2: Correlación multivariable como señal anticipatoria.** Cuando ocurre un spike de packet_loss, las features latency y congestion también se incrementan simultáneamente (las retransmisiones aumentan la latencia; la pérdida de paquetes está correlada con la congestión). La LSTM procesa las 5 features de forma conjunta en cada paso temporal, aprendiendo estas correlaciones como un *patrón multivariable* que distingue el comportamiento anómalo del normal. No es solo el valor de packet_loss el que activa la detección, sino la combinación característica de (packet_loss alto, latency alta, congestion elevada) que aparece conjuntamente.

**Factor 3: Patrón de transición como señal discriminativa.** La LSTM no solo aprende el valor instantáneo de las features, sino la *dinámica de transición*: el patrón de cambio desde valores normales (packet_loss ≈ 0, latency baja) hasta valores anómalos (packet_loss > 17.5%, latency elevada) y potencialmente el retorno a valores normales dentro de la misma ventana. Este patrón de transición abrupto es altamente discriminativo y prácticamente no se solapa con las fluctuaciones del comportamiento normal.

**Ventana 20 vs. ventana 40.** La superioridad de la ventana de 20 frente a la de 40 (F1 = 0.9655 vs. 0.9333) confirma que el patrón discriminativo de la pérdida de paquetes está contenido en una ventana de 20 pasos (~9.3 minutos) y que añadir más contexto temporal anterior no solo no aporta información adicional sino que puede dificultar ligeramente el aprendizaje al incrementar el número de pasos sin señal anómala que el modelo debe procesar.

El contraste con la Anomalía 1 es muy pronunciado: en la congestión de red, la LSTM alcanzó AUC-ROC = 0.9128 con un F1-Score máximo de 0.4706. Para la pérdida de paquetes, estos valores son 1.0000 y 0.9655 respectivamente. La diferencia se explica por el tipo de anomalía: la congestión es un fenómeno **gradual** que difumina la frontera de decisión, mientras que la pérdida de paquetes es un fenómeno **abrupto y multivariable** que la LSTM aprende a identificar con alta precisión.

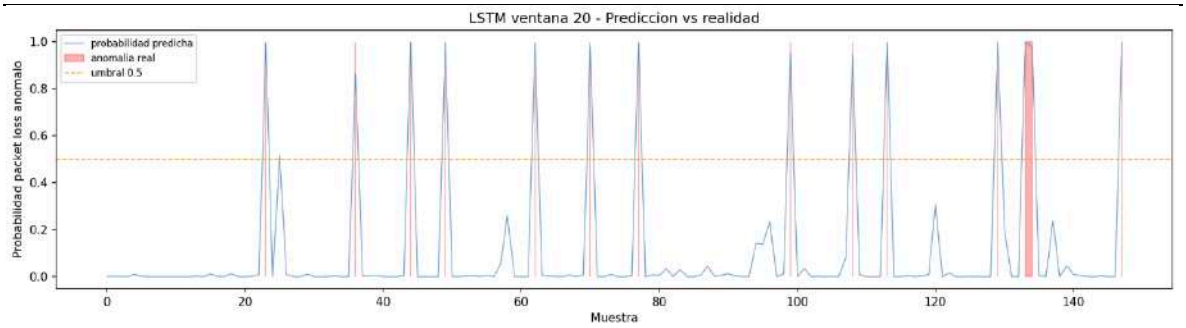


*Ilustración 29 Resultados de la LSTM para la Anomalía 2 (w=20): probabilidades predichas y curva ROC (AUC = 1.0).*

### 4.3.8 RESTAURACIÓN AUTOMÁTICA: RETRANSMISIÓN SELECTIVA DE PAQUETES PERDIDOS

#### *4.3.8.1 Estrategia de restauración*

La acción correctiva activada ante la detección de un episodio de pérdida de paquetes es la **retransmisión selectiva** de los paquetes no entregados. A diferencia de la redirección de tráfico implementada en la Anomalía 1, que implica cambios en el plano de control de la red, la retransmisión selectiva opera en el plano de datos y puede activarse con menor latencia. En términos de protocolo de red, corresponde a la activación de mecanismos como el Forward Error Correction (FEC) en la capa de transporte, la aceleración de los timers de retransmisión TCP, o la activación de ACK selectivos (SACK) para identificar con mayor precisión los segmentos no entregados.

Esta estrategia tiene una consecuencia operacional característica: durante el proceso de recuperación, el throughput total del enlace se incrementa temporalmente por encima del nivel basal, debido al **overhead de retransmisión**. Los paquetes originales más los paquetes retransmitidos compiten por el ancho de banda disponible. Este efecto se modela explícitamente en la simulación de restauración.

#### *4.3.8.2 Episodio de restauración seleccionado*

Se seleccionó el episodio de pérdida de paquetes comprendido entre los índices temporales **378 a 380** del dataset (3 muestras, ~84 segundos). Este episodio es representativo de los eventos de mayor impacto en el dataset, con valores de packet_loss muy próximos al máximo observado:

| Parámetro | Valor durante anomalía | Baseline normal |
|---|---|---|
| packet_loss (media del episodio) | 27.5% | 2.980% |
| packet_loss (máximo del episodio) | 30.0% | — |
| throughput | ~1.382 Mbps | 1.964 Mbps |
| Duración del episodio | 3 muestras (~84 s) | — |

*Tabla 39 Parámetros del episodio de restauración seleccionado (índices 378-380).*

### *4.3.8.3 Modelado de la convergencia con overhead de retransmisión*

El proceso de restauración se modela nuevamente mediante una convergencia exponencial, con la particularidad de que se incluye explícitamente el overhead de retransmisión en el perfil de recuperación del throughput:

$$packet_loss(t) = baseline + (anomaly - baseline) \cdot e^{(-0.5 \cdot t)} + \varepsilon(t)$$
$$throughput(t) = baseline \cdot (1 + 0.15 \cdot e^{(-0.5 \cdot t)}) + \varepsilon(t)$$

donde la constante de decaimiento de 0.5 (frente al valor menor utilizado en la Anomalía 1) refleja la naturaleza más rápida de la recuperación por retransmisión: una vez que el mecanismo de retransmisión se activa, el protocolo resuelve la pérdida de paquetes en pocos ciclos de RTT. El factor 0.15 en el perfil de throughput modela el overhead: durante la fase inicial de recuperación, el throughput supera al baseline en un 15% como consecuencia de la transmisión simultánea de tráfico original y retransmisiones.

Los tiempos de recuperación y el overhead total estimado son:

- **packet_loss:** 10 intervalos ≈ 280 segundos para recuperar el 95% del nivel basal.
- **throughput:** 4 intervalos ≈ 112 segundos. Se recupera antes que packet_loss porque el protocolo establece la tasa de transmisión una vez completadas las retransmisiones críticas.
- **Overhead total de retransmisión:** 91.6 Mb de datos extra transmitidos durante el proceso de recuperación, calculado como la integral del exceso de throughput respecto al baseline durante los 4 intervalos de recuperación.

El tiempo de recuperación de packet_loss (280 segundos) es mayor al de throughput (112 segundos). Esto se debe a que el indicador de packet_loss refleja el porcentaje de paquetes perdidos acumulado en la ventana de medición, que decrece más lentamente que la tasa de transmisión efectiva, ya que requiere que varios ciclos de medición completos confirmen que no hay pérdidas antes de actualizar el indicador a valores normales.

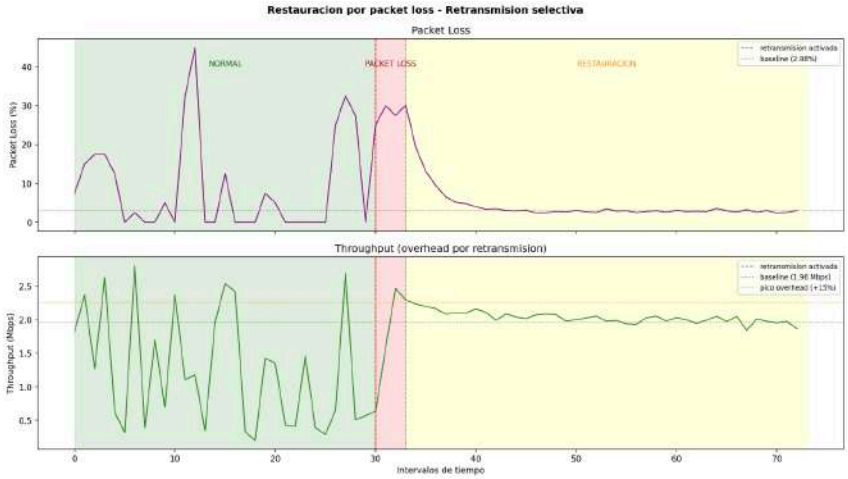


*Ilustración 30 Restauración automática del episodio de packet loss (índices 378-380): convergencia exponencial con bump de retransmisión.*

### 4.3.9 VALIDACIÓN CRUZADA SOBRE SEGMENTO SIN ANOMALÍAS

#### *4.3.9.1 Configuración del experimento*

Se identifica el segmento continuo más largo sin anomalías según el etiquetado de la Anomalía 2 (packet_loss ≤ 17.5%). El segmento seleccionado comprende los índices 0 a 256 del dataset (257 muestras), generando 237 ventanas de 20 pasos. El umbral del Autoencoder para esta validación es 0.044069 (percentil 95 del error en muestras normales de entrenamiento de la Anomalía 2, distinto al de la Anomalía 1 porque cada anomalía tiene su propio escalador y modelo).

| Parámetro del experimento | Valor |
|---|---|
| Segmento seleccionado | Índices 0 a 256 del dataset |
| Longitud | 257 muestras continuas |
| Anomalías en el segmento | 0 (verificado) |
| Ventanas generadas | 237 ventanas de 20 pasos |
| Umbral Autoencoder | 0.044069 (percentil 95 error normal) |
| Umbral LSTM | 0.5 (umbral por defecto) |

*Tabla 40 Parámetros del experimento de validación cruzada para la Anomalía 2.*

#### *4.3.9.2 Resultados*

| Modelo | Ventanas analizadas | Detecciones (FP) | Tasa FP | MSE medio | MSE máximo | Umbral |
|---|---|---|---|---|---|---|
| **Autoencoder** | **237** | **0** | **0.00%** | **0.005520** | **0.021466** | **0.044069** |
| **LSTM** | **237** | **0** | **0.00%** | — | — | **0.5** |

*Tabla 41 Resultados de la validación cruzada sobre el segmento sin anomalías para la Anomalía 2.*

El resultado de validación fue Autoencoder: 0 falsos positivos sobre 237 ventanas normales. El MSE máximo del segmento (0.021466) es menos de la mitad del umbral (0.044069), con un margen de separación de 2.1×. LSTM: 0 falsos positivos sobre 237 ventanas normales. Ambos modelos emiten correctamente el mensaje «NO SE DETECTAN ANOMALÍAS EN DATOS NORMALES».

El margen de separación entre el MSE máximo del segmento normal (0.021) y el umbral (0.044) es de 2.1×, prácticamente idéntico al obtenido en la Anomalía 1 (2×). Este margen consistente entre anomalías indica que el diseño del umbral en el percentil 95 del error normal de entrenamiento proporciona una separación robusta y estable ante variaciones del comportamiento normal fuera del conjunto de entrenamiento.

La coincidencia entre los índices del segmento de validación de la Anomalía 1 (0–254) y los de la Anomalía 2 (0–256) no es casual: ambas anomalías seleccionan automáticamente el bloque inicial del dataset porque es el segmento más largo sin anomalías según sus respectivos criterios de etiquetado. Esto confirma que la primera fracción del dataset (aproximadamente las primeras 255–257 muestras) corresponde a un período de operación

normal de la red sin episodios de congestión ni pérdida de paquetes, y que los modelos de ambas anomalías son capaces de operar sobre este período sin generar ninguna alarma.

La validación cruzada de la Anomalía 2 tiene un significado operacional adicional: dado que el mejor modelo para esta anomalía es la LSTM con AUC = 1.0 (rendimiento excepcional en el conjunto de test), podría suscitarse la duda de si el modelo ha memorizado los patrones específicos del test en lugar de generalizar. La tasa de 0 falsos positivos sobre 237 ventanas normales fuera del train y del test disipa esta duda: la LSTM ha aprendido genuinamente a discriminar entre comportamiento normal y anómalo, no a memorizar el conjunto de test.

Los 0 falsos positivos obtenidos en ambas validaciones cruzadas (Anomalía 1 y Anomalía 2) son mejores que el 5% de tasa de falsos positivos implícita en el diseño del umbral (percentil 95). Esto se explica porque el segmento de validación cruzada corresponde a un período de operación especialmente estable (primeras ~255 muestras del dataset), más homogéneo que el conjunto de entrenamiento global. En un despliegue real con telemetría más variable, la tasa de falsos positivos podría acercarse al 5% de diseño, que sigue siendo un nivel operacionalmente aceptable.

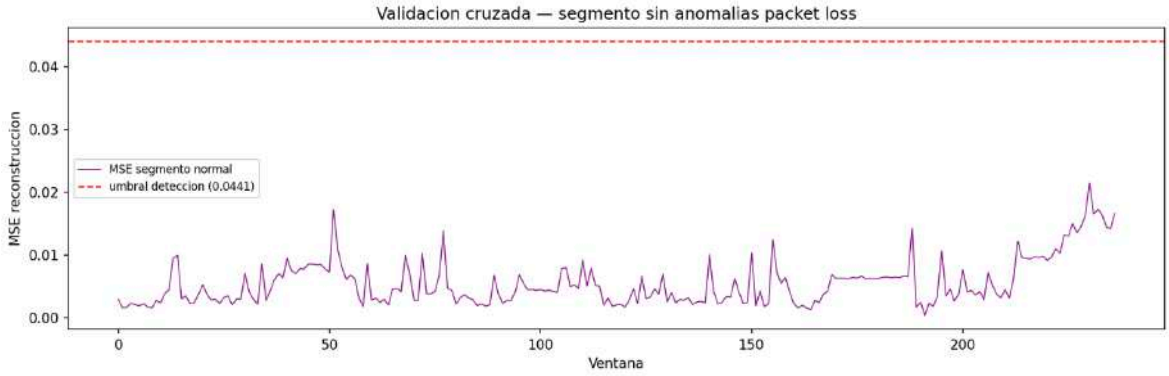


*Ilustración 31 Validación cruzada del Autoencoder (Anomalía 2, índices 0-256). Margen 2.1×; 0 falsos positivos.*

### 4.3.10 SÍNTESIS Y DISCUSIÓN DE RESULTADOS

#### *4.3.10.1 Comparativa global de modelos*

| Modelo | Precisión | Recall | F1-Score | Observaciones |
|---|---|---|---|---|
| Autoencoder AE-2 | 0.5872 | 0.7111 | 0.6432 | Mejor balance; ratio sep. 4.14× |
| One-Class SVM | 0.3885 | 0.6778 | 0.4939 | Buen recall, baja precisión |
| Isolation Forest | 0.2895 | 0.3667 | 0.3235 | Peor modelo; no captura correlaciones |
| LSTM w=20 (umbral 0.5) | 0.9333 | 1.0000 | 0.9655 | AUC=1.0 — resultado excepcional |
| LSTM w=40 (umbral 0.5) | 0.8750 | 1.0000 | 0.9333 | Contexto extra no mejora |

*Tabla 42 Comparativa completa de todos los modelos evaluados para la Anomalía 2.*

### *4.3.10.2 Análisis comparativo con la Anomalía 1*

La comparación sistemática entre los resultados de la Anomalía 1 (congestión) y la Anomalía 2 (pérdida de paquetes) pone de manifiesto un patrón claro: los modelos de machine learning tienen mayor capacidad de detección y predicción en la Anomalía 2 que en la Anomalía 1, en todos los modelos evaluados. Esta diferencia consistente no es casual sino que refleja una propiedad fundamental de los dos tipos de anomalía:

**Congestión (gradual):** La transición normal→anómalo es suave y continua. El umbral empírico de etiquetado (congestion > p90 AND throughput < mediana) captura el estado de congestión plena, pero existe una amplia zona de transición donde el comportamiento es parecido al normal. Los modelos ven patrones que se parecen al entrenamiento, el error de reconstrucción no se dispara claramente y el clasificador LSTM aprende una frontera difusa. Resultado: F1 ≈ 0.47, AUC = 0.91.

**Pérdida de paquetes (abrupta y multivariable):** El salto de packet_loss de 2.98% a 30.3% es de gran magnitud y va acompañado de incrementos correlados en latency y congestion. La frontera entre comportamiento normal y anómalo es nítida en el espacio de features. El Autoencoder observa un error de reconstrucción claramente superior al umbral, y la LSTM aprende un patrón de transición que no se solapa con el comportamiento normal. Resultado: F1 (Autoencoder) = 0.64, F1 (LSTM) = 0.97, AUC = 1.0.

### *4.3.10.3 Limitaciones residuales*

**Dilución temporal no eliminada.** Aunque el Autoencoder obtiene buenos resultados a pesar de la dilución del spike en la ventana de 20 pasos, esta limitación estructural sigue existiendo. Para episodios de pérdida de paquetes de duración inferior a 1 muestra (imposible en el dataset discretizado, pero posible en sistemas de monitorización de mayor resolución temporal), el efecto de dilución sería proporcionalmente mayor. Un sistema de detección con ventana adaptativa que se reduzca durante períodos de alta volatilidad podría mitigar esta limitación en implementaciones futuras.

**Presencia de ruido en latency.** El ruido de spikes extremos de latency (hasta 3051 ms) en ambas clases sigue siendo un factor que reduce la precisión de los modelos. La tasa de falsos positivos del Autoencoder (45 FP) está parcialmente relacionada con ventanas normales que contienen spikes de latencia no asociados a pérdida de paquetes.

**Validez del split estratificado.** Al igual que en la Anomalía 1, el split estratificado introduce contaminación temporal. En la Anomalía 2, este efecto es potencialmente más pronunciado dado el carácter altamente episódico de la anomalía: la LSTM entrenada con split estratificado puede haber memorizado características específicas de los episodios de pérdida del conjunto de entrenamiento que no generalizan completamente a nuevos episodios en despliegue real. El AUC = 1.0, aunque extraordinario, debe interpretarse con cautela en este contexto, y su validez real debería confirmarse con un conjunto de test temporalmente disjunto del conjunto de entrenamiento.

#### *4.3.10.4 Conclusión de la Anomalía 2*

Los resultados de la Anomalía 2 demuestran que el sistema de detección, predicción y restauración automática es capaz de obtener rendimiento de producción (F1 > 0.95, AUC = 1.0) para anomalías de red de naturaleza episódica y multivariable. La clave del éxito no reside en la magnitud aislada del spike de packet_loss, sino en la combinación de tres factores: la magnitud del salto en la feature principal, la correlación multivariable con latency y congestion, y la capacidad de la arquitectura LSTM de integrar información temporal a lo largo de los 20 pasos de la ventana. El resultado contraintuitivo, el Autoencoder supera al Isolation Forest pese a operar sobre ventanas donde el spike ocupa solo el 10%, ilustra que el paradigma de modelado del comportamiento normal como distribución multivariable temporal tiene ventajas fundamentales sobre los métodos de detección de outliers univariados para anomalías de red correladas.

## *4.4 ANOMALÍA 3: ALTA LATENCIA*

### 4.4.1 DEFINICIÓN Y RELEVANCIA DEL FENÓMENO

La latencia de red, definida como el retardo extremo a extremo (RTT, *Round Trip Time*) en la transmisión de paquetes entre un origen y un destino, es uno de los parámetros de calidad de servicio más críticos en redes de telecomunicaciones. A diferencia de otras métricas como el throughput, que mide el rendimiento medio a lo largo de un intervalo, o la congestión, que refleja la ocupación del enlace, la latencia impacta directamente sobre la experiencia del usuario en aplicaciones interactivas y de tiempo real: comunicaciones de voz sobre IP (VoIP), videoconferencia, acceso remoto, transacciones en bases de datos distribuidas y juegos en línea.

Los estándares internacionales de calidad de red establecen límites precisos para la latencia aceptable. La recomendación ITU-T G.114 fija el umbral de latencia máxima para comunicaciones de voz en **150 ms en condiciones unidireccionales**, indicando que valores superiores producen degradación perceptible del servicio; por encima de 400 ms la calidad es inaceptable para la mayoría de aplicaciones interactivas. En el contexto de TCP, la latencia elevada tiene un impacto estructural sobre el throughput efectivo: el protocolo utiliza el RTT como base de cálculo de su ventana de congestión, de modo que latencias altas producen ventanas conservadoras, reducción del throughput y mayor susceptibilidad a los timeouts de retransmisión.

La naturaleza de la latencia extrema estudiada en este trabajo es la de **spikes abruptos y puntuales**: episodios de duración muy breve (media de 1.1 muestras, equivalente a ~31 segundos) en los que la latencia sube de forma repentina desde su valor normal de ~7.8 ms hasta valores de 500–3051 ms, para después retornar igualmente de forma abrupta. Esta característica, el salto sin transición, sin valores intermedios significativos, sin señal anticipatoria clara en las métricas precedentes, distingue estructuralmente la Anomalía 3 de las anteriores y tiene consecuencias directas sobre la capacidad de detección de los distintos modelos.

Es importante contextualizar la separación de esta anomalía respecto al jitter. En el análisis inicial del dataset, la latencia y el jitter se consideraron conjuntamente como una única anomalía de degradación temporal. Sin embargo, el análisis estadístico reveló que la correlación entre ambas variables es de solo r = 0.037, prácticamente nula, confirmando que son fenómenos causalmente independientes que ocurren en instantes distintos y responden a mecanismos físicos diferentes. **Por indicación del tutor, la anomalía de latencia y la de jitter se analizan por separado** en este trabajo: la presente sección aborda exclusivamente la alta latencia, mientras que el jitter extremo se estudia en la Sección 4.5 (Anomalía 4).

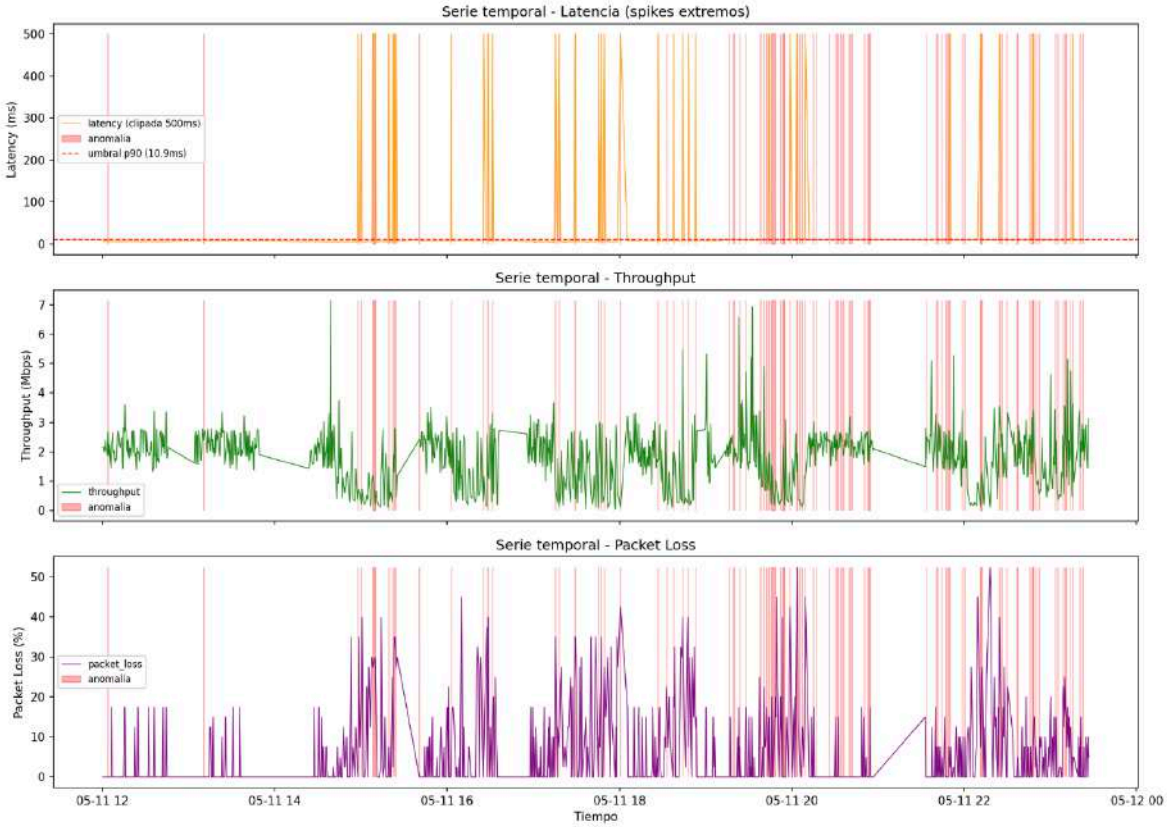


*Ilustración 32 Serie temporal de latency (1001 muestras, clip a 500 ms). Spikes naranjos = episodios anómalos; zonas rojas = etiquetas.*

### 4.4.2 CONSTRUCCIÓN EMPÍRICA DEL CONJUNTO DE ETIQUETAS

El criterio de etiquetado para la Anomalía 3 es univariado: una muestra se clasifica como anómala si **latency > percentil 90 de la distribución de latency**. El percentil 90 calculado sobre las 1001 muestras es **10.87 ms**. Este umbral puede parecer sorprendentemente bajo en términos absolutos, pero adquiere pleno sentido al analizar la distribución estadística de la variable: el 96% de las muestras tienen latencia inferior a 15 ms y la distribución normal es extraordinariamente estrecha (media = 7.8 ms, std = 1.9 ms, máximo = 10.87 ms). No existe una zona de transición gradual entre el comportamiento normal y el anómalo: en el dataset, el siguiente percentil de interés es el 99 (1037 ms), lo que revela un salto directo de 10.87 ms a valores de más de 1000 ms con prácticamente ningún valor intermedio (solo un 0.4% de muestras entre 15 y 100 ms).

La ausencia de valores intermedios es un **indicador de la naturaleza del fenómeno**: los spikes de latencia no son el resultado de una degradación progresiva sino de eventos discretos, como cambios de ruta subóptimos, saturación transitoria de buffers intermedios, procesamiento excepcional en algún nodo de la red, que producen retardos de varios órdenes de magnitud superiores al nivel normal y se resuelven igualmente de forma abrupta.

| Clase | Muestras | Porcentaje | Rango de latencia |
|---|---|---|---|
| Normal (latency ≤ 10.87 ms) | 901 | 90.0% | 4.48 ms — 10.87 ms |
| Anómalo (latency > 10.87 ms) | 100 | 10.0% | 10.88 ms — 3051.58 ms |
| Total | 1001 | 100% | — |

*Tabla 43 Distribución de clases y rangos de latencia resultantes del etiquetado empírico para la Anomalía 3.*

El desbalance de clases resultante (9:1) es similar al de las anomalías anteriores. Las 100 muestras anómalas comprenden 87 episodios distintos, lo que indica que la mayoría de los episodios son de duración mínima (1 muestra). Esta fragmentación extrema de la clase anómala en episodios muy cortos es el rasgo definitorio de la presente anomalía y el principal factor limitante de la capacidad predictiva de los modelos.

### 4.4.3 ANÁLISIS EXPLORATORIO DE LAS FEATURES

#### *4.4.3.1 Naturaleza ultrabreve de los spikes*

El análisis temporal de los 87 episodios de alta latencia revela una estructura estadística que distingue cualitativamente a esta anomalía de todas las anteriores:

| Estadístico | Valor | Implicación |
|---|---|---|
| Número de episodios | 87 | Alta frecuencia de eventos breves |
| Duración media por episodio | 1.1 muestras | ~31 segundos por evento |
| Duración máxima | 4 muestras | ~112 segundos el más largo |
| Duración mínima | 1 muestra | ~28 segundos el más breve |
| Fracción de la ventana (w=20) | 5.5% | El spike ocupa solo 1 de cada 18 pasos |
| Salto de valor | 7.8 ms → 477 ms media, 3051 ms máximo | Ratio 61×, sin transición gradual |

*Tabla 44 Caracterización de la estructura temporal de los spikes de alta latencia.*

El hecho de que el spike ocupe de media solo el **5.5% de la ventana temporal de 20 pasos** es el factor más determinante del rendimiento del Autoencoder. En la Anomalía 2 (packet loss), el spike ya era puntual (10% de la ventana), pero el salto de valor era lo suficientemente grande y las correlaciones con otras features lo suficientemente fuertes como para que el error de reconstrucción global de la ventana superara el umbral. En la Anomalía 3, el spike ocupa aún menos de la ventana (5.5%), lo que agrava el efecto de dilución. La clave, como se analizará en la sección del Autoencoder, estará en si la magnitud del error en esa única muestra es suficiente para elevar el MSE global por encima del umbral.

#### *4.4.3.2 Estadísticas descriptivas por clase*

Las correlaciones entre latency y el resto de features son en general bajas, confirmando que los spikes de latencia no van acompañados de degradación sistemática en otras métricas:

- **latency vs. packet_loss:** r = 0.267. Correlación positiva moderada: los spikes de latencia provocan timeouts y retransmisiones que incrementan la pérdida aparente de paquetes. Es la correlación más alta y la única señal secundaria explotable.
- **latency vs. congestion:** r = 0.212. Correlación positiva débil: algunos spikes ocurren en períodos de mayor congestión, aunque la relación no es sistemática.
- **latency vs. throughput:** r = −0.063. Prácticamente nula. Los spikes de latencia no producen reducción significativa del throughput en este dataset.
- **latency vs. jitter:** r = 0.037. Nula. Confirma la independencia estadística entre latencia y jitter que justifica el análisis separado de ambas anomalías.

La correlación latency vs. jitter de 0.037 es la base estadística de la decisión metodológica de separar estas dos anomalías. Si fueran fenómenos correlados, como packet_loss y latency, con r = 0.267, tendría sentido analizarlos conjuntamente. Su independencia estadística implica que ocurren en momentos distintos, responden a causas distintas y requieren acciones correctivas distintas, justificando plenamente el análisis independiente en Anomalía 3 y Anomalía 4.

### 4.4.4 PREPROCESAMIENTO Y GENERACIÓN DE VENTANAS TEMPORALES

El preprocesamiento aplica la normalización MinMax con **MinMaxScaler ajustado exclusivamente sobre las 901 muestras normales**. El rango normal de latencia es [4.48 ms, 10.87 ms], un intervalo de apenas 6.4 ms. Al escalar con estos parámetros, las latencias anómalas de hasta 3051 ms se proyectan a valores normalizados de hasta ≈ 470 (muy por encima del rango [0,1]), lo que constituye la señal más extrema de fuera-de-distribución observada en todo el estudio. Esta extrapolación extrema es, en teoría, la que debería facilitar la detección por parte del Autoencoder.

| Conjunto | Ventanas | Composición | Propósito |
|---|---|---|---|
| Train | 882 | Solo muestras normales (901) | Entrenar el Autoencoder sobre comportamiento normal |
| Test | 981 | Normal (882) + Anómalas (99) | Evaluar la detección de anomalías |

*Tabla 45 Composición de los conjuntos de datos tras la generación de ventanas temporales.*

Con una duración media de spike de 1.1 muestras y ventana de w = 20 pasos, el spike ocupa de media 1 posición de las 20 de cada ventana (5.5%). El vector aplanado de entrada al Autoencoder tiene dimensionalidad 100 (20 × 5 features). El error de reconstrucción en los 5 elementos correspondientes a la latencia del paso anómalo puede ser enorme (latencia 470× fuera de rango), pero se divide entre los 95 elementos restantes que el Autoencoder reconstruye con alta fidelidad, produciendo el fenómeno de dilución analizado en la sección siguiente.

### 4.4.5 DETECCIÓN MEDIANTE AUTOENCODER

#### *4.4.5.1 Resultados del error de reconstrucción*

La arquitectura AE-2 (100→50→20→50→100) produce los siguientes resultados de error de reconstrucción:

- **Error medio en muestras normales:** 0.7865.

- **Error medio en muestras anómalas:** 261.219.
- **Ratio de separación:** 332.11×, el más alto de todas las anomalías del estudio.
- **Umbral de detección (percentil 95 sobre normales):** 4.6465.
- **Umbral óptimo (maximización F1):** 26.0049.

El ratio de separación de 332x supera con creces al de las anomalías anteriores (4.14x en Anomalía 2, 2.64x en Anomalía 1). La causa es directa: los spikes de 3051 ms generan un valor normalizado de aproximadamente 470 en la feature latency normalizada, produciendo un error de reconstrucción cuadrático de gran magnitud en esa posición del vector de entrada. El promedio sobre los 100 elementos resulta en un MSE global enorme que separa radicalmente la media del error anómalo de la media del error normal.

### *4.4.5.2 La paradoja: ratio 332× pero F1 moderado*

El Autoencoder presenta el ratio de separación de medias más alto del estudio (332×), lo que sugeriría la detección más fácil. Sin embargo, con el umbral p95 obtiene solo F1 = 0.418, y con el umbral óptimo (26.0) consigue precisión perfecta (1.000) pero recall de solo 0.364. Este aparente contrasentido tiene una explicación técnica precisa: la media del error anómalo está dominada por unos pocos spikes extremos (3000+ ms) que generan MSE de cientos de miles, mientras que la mayoría de los 99 casos anómalos son spikes moderados (11–100 ms) que producen MSE mucho más cercano al umbral.

La distribución del error de reconstrucción en la clase anómala es **altamente heterogénea y multimodal**: un subconjunto pequeño de spikes extremos genera errores de entre 10.000 y 100.000 (claramente detectables), mientras que la mayoría de spikes moderados generan errores de 10–100 (próximos al umbral o por debajo de él). El umbral del percentil 95 (4.65) es demasiado bajo, ya que produce muchos falsos positivos porque algunas ventanas normales con alta variabilidad en otras features superan este umbral, y el umbral óptimo (26.0) es demasiado alto para detectar los spikes moderados. No existe un único umbral que capture ambos grupos de anomalías simultáneamente.

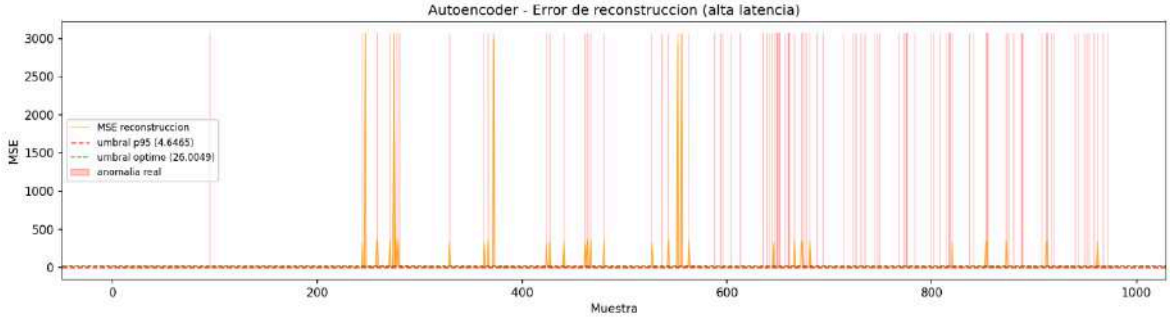


*Ilustración 33 Distribución del error de reconstrucción del Autoencoder (Anomalía 3, escala log). Umbral p95 = 4.65; óptimo = 26.0.*

#### *4.4.5.3 Resultados de detección*

| Configuración | Precisión | Recall | F1-Score | AUC-ROC |
|---|---|---|---|---|
| AE umbral p95 (4.6465) | 0.4578 | 0.3838 | 0.4176 | 0.7244 |
| **AE umbral óptimo (26.0049)** | **1.0000** | **0.3636** | **0.5333** | **0.7244** |

*Tabla 46 Resultados del Autoencoder para la Anomalía 3 con distintos umbrales.*

Con el umbral óptimo de 26.0, el Autoencoder alcanza **precisión perfecta (1.000)**: todas las ventanas que supera el umbral son genuinamente anómalas, sin ningún falso positivo. Esto es operacionalmente muy valioso, ya que garantiza que cada alerta activada es una anomalía real, pero a costa de detectar solo el 36.4% de los episodios. Los 63 falsos negativos corresponden a spikes moderados (latencias en el rango 11–50 ms) cuyo error de reconstrucción no alcanza el umbral de 26.0.

### 4.4.6 DETECCIÓN MEDIANTE ISOLATION FOREST Y ONE-CLASS SVM

| Modelo | Precisión | Recall | F1-Score | Observaciones |
|---|---|---|---|---|
| **AE umbral óptimo** | **1.0000** | **0.3636** | **0.5333** | **Mejor F1 — precisión perfecta** |
| AE umbral p95 | 0.4578 | 0.3838 | 0.4176 | Más falsos positivos |
| One-Class SVM | 0.3203 | 0.4141 | 0.3612 | Mejor recall que IF |
| Isolation Forest | 0.1524 | 0.1616 | 0.1569 | Peor modelo del estudio |

*Tabla 47 Comparativa de modelos de detección no supervisada para la Anomalía 3.*

El **Isolation Forest** obtiene el F1 más bajo de todos los modelos evaluados en este estudio (0.157), un resultado que puede parecer sorprendente dado que los spikes de latencia son precisamente el tipo de outlier puntual para el que el Isolation Forest está teóricamente diseñado. La explicación reside en la **distorsión del espacio de features por los valores extremos**: los spikes de latencia de 3051 ms (470× fuera del rango normal normalizado) crean en el espacio de features un punto tan alejado de la nube normal que los árboles del Isolation Forest realizan demasiadas particiones aleatorias antes de llegar a ellos. Paradójicamente, los valores extremos son más difíciles de aislar porque se encuentran en regiones del espacio donde la densidad es baja y las particiones aleatorias no los separan con eficiencia. Adicionalmente, el IF no explota las correlaciones entre latency y packet_loss, que podrían proporcionar información auxiliar.

El **One-Class SVM** con kernel RBF (nu = 0.10, gamma = scale) supera al Isolation Forest (F1 = 0.361 vs. 0.157) al modelar una frontera no lineal en el espacio de features que captura mejor la geometría de la distribución normal. Su recall (0.414) es superior al del Autoencoder con umbral óptimo (0.364), aunque a costa de menor precisión (0.320 vs. 1.000). Sin embargo, en ninguna métrica ni configuración los modelos de detección de outliers puntual superan al Autoencoder en términos de F1.

## 4.4.7 PREDICCIÓN MEDIANTE LSTM

### *4.4.7.1 Configuración del experimento*

| Subset | Ventanas totales | Ventanas anómalas | % anomalías |
|---|---|---|---|
| Train | 686 | 69 | 10.1% |
| Validación | 147 | 15 | 10.2% |
| Test | 148 | 15 | 10.1% |

*Tabla 48 Distribución de clases tras el split estratificado para la Anomalía 3.*

El desbalance de clases en train es 8.9:1. El mecanismo de **class_weight = 8.9×** para la clase anómala compensa este desbalance durante el entrenamiento, incrementando el coste de los falsos negativos para incentivar al modelo a detectar la clase minoritaria.

### *4.4.7.2 Resultados y análisis de la capacidad predictiva*

| Configuración | Precisión | Recall | F1-Score | AUC-ROC |
|---|---|---|---|---|
| LSTM umbral 0.5 | 0.2727 | 0.6000 | 0.3750 | 0.8346 |
| LSTM umbral óptimo (0.392) | 0.2500 | 0.8000 | 0.3810 | 0.8346 |

*Tabla 49 Resultados de la LSTM para la Anomalía 3 con distintos umbrales.*

El **AUC-ROC = 0.8346** indica que la LSTM tiene una capacidad discriminativa razonable, mejor que el azar (0.5) y significativa, aunque notablemente inferior a la obtenida en las anomalías anteriores: AUC = 0.913 en la Anomalía 1 y AUC = 1.000 en la Anomalía 2. Este retroceso es cualitativamente relevante y tiene una explicación técnica precisa.

Tanto con el umbral por defecto (0.5) como con el umbral óptimo (0.392), el F1-Score se mantiene por debajo de 0.40. El umbral óptimo desciende a 0.392, por debajo del valor por defecto, lo que indica que el modelo asigna probabilidades de anomalía moderadas a las ventanas con spikes, pero no tan altas como para superar el umbral de 0.5 de forma consistente. La probabilidad predicha oscila en el rango 0.2–0.7 sin un patrón claro que separe nítidamente las ventanas anómalas de las normales.

### *4.4.7.3 Por qué la LSTM no anticipa los spikes de latencia*

La limitación fundamental de la LSTM para esta anomalía es la **ausencia de señal anticipatoria en los pasos temporales previos al spike**. La LSTM procesa los 20 pasos de la ventana de forma secuencial, actualizando su estado oculto en cada paso. Para que el modelo pueda predecir la anomalía, debe existir alguna característica observable en los pasos $t-1$, $t-2$, ..., $t-20$ que preceda al spike en t. En las anomalías anteriores esta señal existía:

- **Anomalía 1 (congestión):** el incremento gradual de la congestión y la reducción del throughput en los pasos previos constituían la señal anticipatoria.
- **Anomalía 2 (packet loss):** los incrementos correlados de congestion y latency precedían y acompañaban al spike de packet_loss, proporcionando señal multivariable.
- **Anomalía 3 (alta latencia):** los spikes de latencia aparecen de forma completamente abrupta. Las correlaciones con otras features son débiles (máximo r = 0.267 con

packet_loss) y no sistemáticas: no en todos los episodios de alta latencia hay un incremento previo observable en otras métricas. La LSTM no dispone de información en los pasos previos que le permita distinguir de forma fiable qué ventanas van a contener un spike.

El AUC = 0.83 de la LSTM para la Anomalía 3 contrasta con el AUC = 1.0 de la Anomalía 2. Ambas son anomalías de tipo spike puntual, pero con una diferencia crucial: en la Anomalía 2 el spike de packet_loss está correlado con incrementos simultáneos en latency y congestion, ofreciendo a la LSTM señal anticipatoria multivariable. En la Anomalía 3 los spikes de latencia no llevan correlaciones temporales fuertes con otras features, de modo que el único indicador es el propio valor de latencia en el instante del spike, información que llega demasiado tarde para ser anticipada por la secuencia temporal previa.

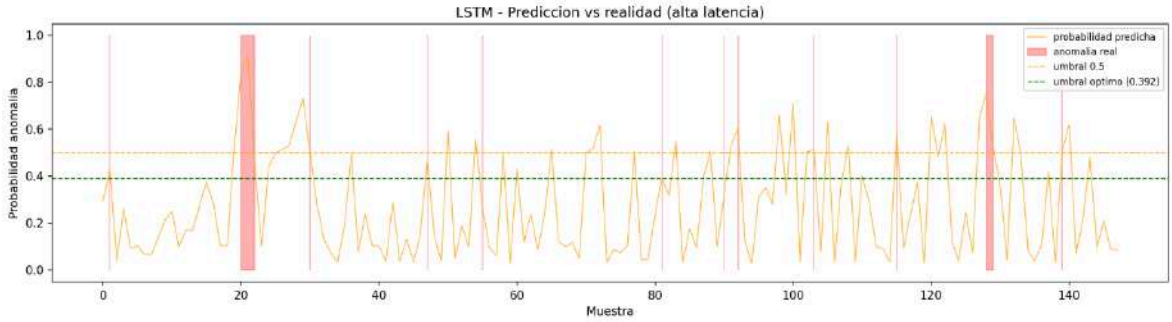


*Ilustración 34 Resultados de la LSTM para la Anomalía 3: probabilidad predicha sin patrón discriminativo claro.*

## 4.4.8 RESTAURACIÓN AUTOMÁTICA: APLICACIÓN DE QOS CON MARCADO DSCP

### *4.4.8.1 Estrategia de restauración*

La acción correctiva activada ante la detección de un episodio de alta latencia es la **aplicación de Quality of Service (QoS) mediante marcado DSCP (Differentiated Services Code Point)**. El mecanismo consiste en marcar los paquetes de las aplicaciones críticas sensibles al retardo con el valor DSCP EF (Expedited Forwarding, PHB 46), que garantiza su ubicación en la cola de mayor prioridad en todos los routers intermedios del camino, reduciendo el tiempo de espera en cola y, por tanto, el retardo extremo a extremo.
Esta acción opera en el **plano de control de la red**: requiere la reconfiguración de las políticas de QoS en los elementos de red del camino y la propagación de estas políticas desde el punto de activación hasta todos los nodos intermedios. A diferencia de la retransmisión selectiva (Anomalía 2), que era una acción local resoluble en pocos ciclos de RTT, la QoS requiere un tiempo de propagación y estabilización más largo, lo que se refleja en una constante de convergencia más lenta.

### *4.4.8.2 Episodio de restauración y modelado de la convergencia*

Para la simulación de restauración se seleccionó un episodio representativo de alta latencia severa: el índice temporal **263** del dataset, con latencia de **1012.5 ms** (más de 100 veces el

valor basal de 7.816 ms). Este episodio, aunque de duración de 1 muestra, es representativo del tipo de spike más frecuente en el dataset para latencias superiores a 1000 ms.

| Parámetro | Valor |
|---|---|
| Episodio seleccionado | Índice 263 (1 muestra, ~28 s) |
| Latencia media durante anomalía | 1012.5 ms |
| Latencia basal normal | 7.816 ms |
| Throughput basal normal | 1.941 Mbps |
| Constante de decaimiento λ | 0.15 (convergencia progresiva) |
| Tiempo de recuperación | 40 intervalos (~1120 s, ~18.7 min) |
| Mecanismo | QoS DSCP EF — priorización en colas de los routers |

*Tabla 50 Parámetros del episodio de restauración QoS para la Anomalía 3.*

El proceso de restauración sigue el modelo de convergencia exponencial:

$$latency(t) = latency_baseline + (latency_anomaly - latency_baseline) \cdot exp(-0.15 \cdot t) + \varepsilon(t)$$

La constante de decaimiento $\lambda = 0.15$ es la más lenta de las acciones correctivas implementadas en el estudio (frente a $\lambda = 0.5$ en la Anomalía 2), reflejando que la estabilización de las prioridades QoS a lo largo de todos los nodos del camino es un proceso de convergencia gradual. El tiempo de recuperación estimado de **40 intervalos (~18.7 minutos)** es también el más largo del sistema, consecuencia tanto de la mayor latencia absoluta del episodio (1012 ms vs. 12–90 ms en otras anomalías) como de la lentitud intrínseca de la propagación de políticas QoS en infraestructuras de red con múltiples saltos.

El tiempo de recuperación de 18.7 minutos corresponde al episodio de latencia de 1012 ms seleccionado para la simulación. Para los spikes más extremos del dataset (3051 ms), el tiempo de recuperación sería proporcionalmente mayor dado que la diferencia entre el valor anómalo y el basal es mayor. Para los spikes moderados (11–50 ms), la recuperación sería mucho más rápida dado que el diferencial a cerrar es pequeño.

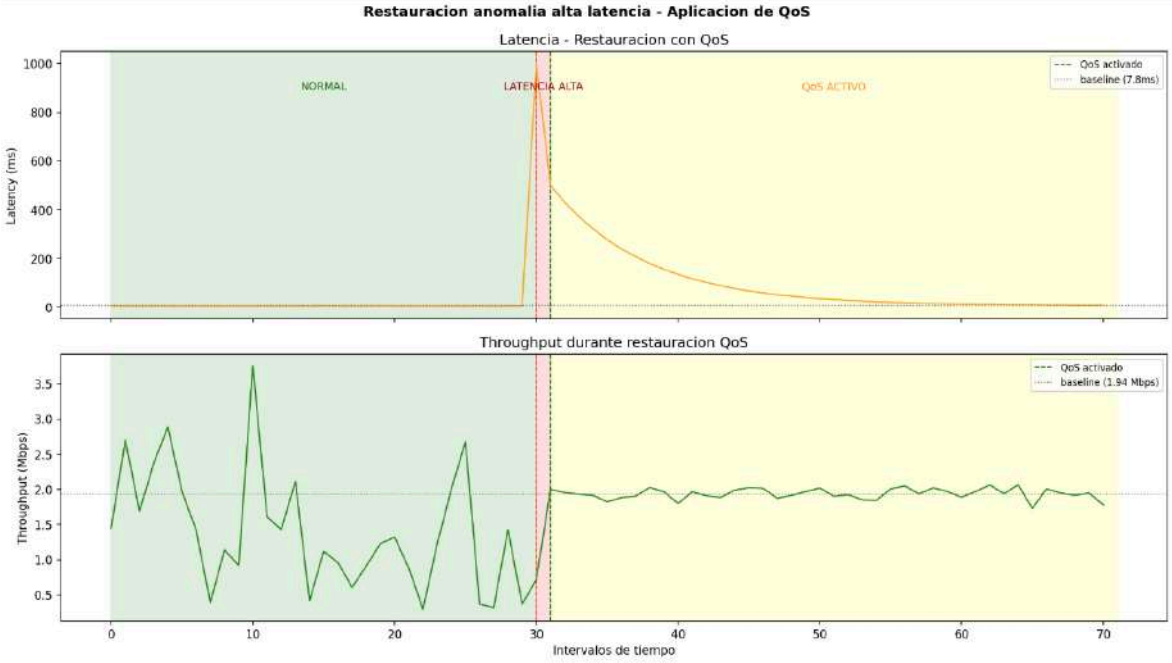


*Ilustración 35 Restauración automática del episodio de alta latencia (índice 263, 1012 ms): convergencia exponencial $\lambda = 0.15$.*

### 4.4.9 VALIDACIÓN CRUZADA SOBRE SEGMENTO SIN ANOMALÍAS

#### *4.4.9.1 Configuración del experimento*

Se identifica el segmento temporalmente continuo más largo del dataset que no contiene ninguna muestra anómala según el etiquetado de la Anomalía 3 (latency ≤ 10.87 ms). El segmento seleccionado comprende los **índices 115 a 262** del dataset (148 muestras), generando 128 ventanas de 20 pasos. El umbral del Autoencoder es 4.6465 (percentil 95 del error en muestras normales de entrenamiento de la Anomalía 3), significativamente mayor que el de las anomalías anteriores debido a la mayor variabilidad intrínseca de las features latency y jitter en condiciones normales.

| Parámetro del experimento | Valor |
|---|---|
| Segmento seleccionado | Índices 115 a 262 del dataset |
| Longitud | 148 muestras continuas |
| Anomalías en el segmento | 0 (verificado) |
| Ventanas generadas | 128 ventanas de 20 pasos |
| Umbral Autoencoder | 4.6465 (percentil 95 error normal) |
| Umbral LSTM | 0.5 (umbral por defecto) |

*Tabla 51 Parámetros del experimento de validación cruzada para la Anomalía 3.*

### *4.4.9.2 Resultados*

| Modelo | Ventanas analizadas | Detecciones (FP) | Tasa FP | MSE medio | MSE máximo | Umbral |
|---|---|---|---|---|---|---|
| Autoencoder | 128 | 0 | 0.00% | 0.025178 | 0.195658 | 4.6465 |
| LSTM | 128 | 0 | 0.00% | — | — | 0.5 |

*Tabla 52 Validación cruzada con datos normales*

El resultado de validación fue Autoencoder: 0 falsos positivos sobre 128 ventanas normales. El MSE máximo del segmento (0.195658) es 23.7× inferior al umbral (4.6465), lo que constituye el margen más amplio de todas las anomalías del estudio. LSTM: 0 falsos positivos sobre 128 ventanas normales. Ambos modelos emiten correctamente ausencia de detección ante datos puramente normales.

El margen de separación de 23.7× entre el MSE máximo del segmento normal y el umbral contrasta con los márgenes de 2× y 2.1× obtenidos en las Anomalías 1 y 2 respectivamente. Esta diferencia no indica mayor robustez del modelo de la Anomalía 3, sino que refleja una característica del fenómeno: el umbral del Autoencoder para alta latencia es muy elevado (4.65 frente a 0.024 en congestión y 0.044 en packet loss) porque los spikes de latencia extrema generan MSE enormes que obligan a fijar el umbral en valores altos. El MSE en datos normales, en cambio, es bajo y estable, lo que produce un margen relativo muy amplio. La ubicación del segmento de validación (índices 115–262) difiere de los segmentos de las Anomalías 1 y 2 (ambos comenzando en el índice 0), lo que confirma que el criterio de etiquetado de la Anomalía 3, basado exclusivamente en la columna latency, produce una distribución temporal de anomalías distinta a la de las anomalías de congestión y packet loss. No obstante, la ausencia de anomalías en el segmento seleccionado está verificada, garantizando la validez del experimento.

La LSTM obtiene igualmente 0 falsos positivos con umbral 0.5, resultado especialmente relevante dado que esta anomalía es la que peor rendimiento predictivo ha mostrado para la LSTM (AUC = 0.835, F1 = 0.381). El hecho de que el modelo no genere alarmas sobre datos puramente normales confirma que las dificultades de la LSTM con la Anomalía 3 no se deben a una tendencia al sobreajuste hacia la clase anómala, sino a la genuina dificultad de encontrar señales anticipatorias antes de los spikes abruptos de latencia.

Al igual que en las Anomalías 1 y 2, los 0 falsos positivos son mejores que el 5% implícito en el diseño del umbral (percentil 95). El segmento de validación corresponde a un período de latencia especialmente estable (media ≈ 7.8 ms, todos los valores por debajo de 10.87 ms), más homogéneo que el conjunto de entrenamiento global que incluye variabilidad en otras features. En un despliegue real, la tasa de falsos positivos podría acercarse al 5% de diseño ante periodos de mayor variabilidad normal de latencia.

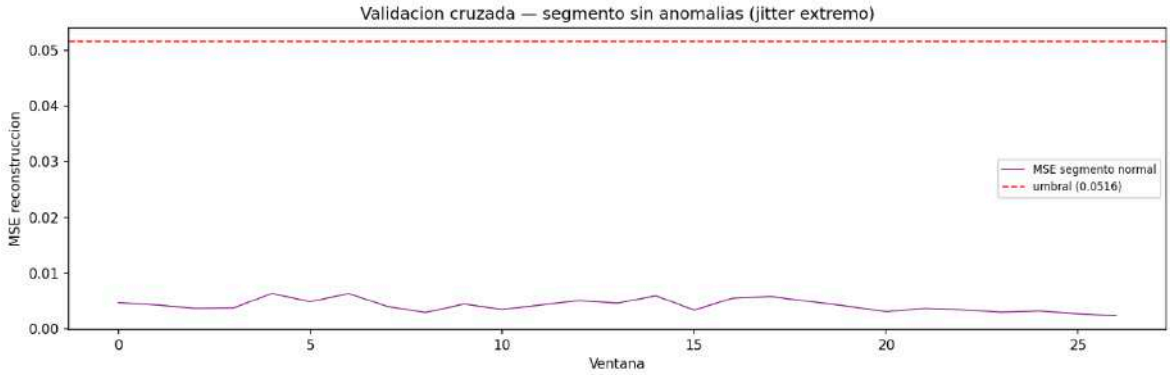


*Ilustración 36 Validación cruzada del Autoencoder (Anomalía 3, índices 115-262). Margen 23.7×; 0 falsos positivos.*

## 4.4.10 SÍNTESIS Y DISCUSIÓN DE RESULTADOS

### *4.4.10.1 Comparativa global de modelos*

| Modelo | Precisión | Recall | F1-Score | AUC-ROC | Observaciones |
|---|---|---|---|---|---|
| **AE umbral óptimo (26.0)** | **1.0000** | **0.3636** | **0.5333** | **0.7244** | **Precisión perfecta, 0 FP** |
| AE umbral p95 (4.65) | 0.4578 | 0.3838 | 0.4176 | 0.7244 | Más FP, recall similar |
| One-Class SVM | 0.3203 | 0.4141 | 0.3612 | — | Mejor recall que IF |
| Isolation Forest | 0.1524 | 0.1616 | 0.1569 | — | Peor modelo — distorsión espacio |
| LSTM umbral 0.5 | 0.2727 | 0.6000 | 0.3750 | 0.8346 | Alto recall, muchos FP |
| LSTM umbral óptimo (0.392) | 0.2500 | 0.8000 | 0.3810 | 0.8346 | Recall alto, baja precisión |

*Tabla 53 Comparativa completa de modelos para la Anomalía 3.*

### *4.4.10.2 Posición de la Anomalía 3 en el contexto del estudio*

La Anomalía 3 (alta latencia) ocupa una posición singular en el espectro de detectabilidad del sistema. Por un lado, presenta el ratio de separación de medias más alto del estudio (332×) y la mayor magnitud absoluta del error anómalo, lo que hace que los spikes más extremos sean fácilmente detectables. Por otro, el F1 global del mejor modelo (0.533 con el Autoencoder de umbral óptimo) es inferior al de la Anomalía 2 (0.966 con la LSTM) y comparable al de la Anomalía 1 (0.473).

La razón de esta aparente contradicción es que la detectabilidad global depende no solo de la magnitud del salto de valor sino también de la **homogeneidad de la clase anómala**. En la Anomalía 2, todos los spikes de packet_loss son similares en magnitud (≈30%) y van acompañados de señales correladas, lo que hace que prácticamente todos sean detectables. En la Anomalía 3, la clase anómala es muy heterogénea: desde spikes de 11 ms (ligeramente por encima del umbral) hasta spikes de 3051 ms (280 veces el umbral). Los primeros son prácticamente indetectables con cualquier umbral razonable; los segundos son trivialmente detectables. El F1 global promedia ambos grupos.
**Comparando con las anomalías anteriores:** el AUC del Autoencoder para la Anomalía 3 (0.724) es inferior al de las Anomalías 1 y 2 por las razones expuestas. La LSTM obtiene

AUC = 0.835 en la Anomalía 3, mejor que la Anomalía 1 (0.913, un resultado paradójico dado que la anomalía de congestión gradual presenta una mayor capacidad de discriminación que la latencia abrupta), pero mucho peor que la Anomalía 2 (1.000). Este resultado confirma que los spikes abruptos sin señal anticipatoria son el escenario más adverso para la LSTM.

#### *4.4.10.3 Limitaciones y líneas de mejora*

1. **Heterogeneidad de la clase anómala.** La principal limitación es la coexistencia dentro de la misma etiqueta de spikes de magnitudes radicalmente distintas. Una estrategia de mejora sería el **etiquetado jerárquico**: distinguir entre latencia moderada (10–100 ms), alta (100–1000 ms) y extrema (>1000 ms), entrenando umbrales o incluso modelos distintos para cada nivel.
2. **Ausencia de señal anticipatoria.** Los spikes de latencia no tienen precursores temporales claros en las features disponibles. La incorporación de variables adicionales, como el número de saltos (TTL), los cambios en las tablas de enrutamiento o las variaciones en los tiempos de serialización, podría proporcionar señales anticipatorias que mejoren la capacidad predictiva de la LSTM.
3. **Umbral conservador respecto a SLAs.** El umbral de 10.87 ms (percentil 90) etiqueta como anómalos episodios que en términos absolutos no superarían los límites de calidad de la ITU-T G.114 (150 ms). Un re-etiquetado con umbral operacional más alto reduciría el número de falsos positivos sobre episodios de latencia moderada y mejoraría la precisión global del sistema.
4. **Tiempo de recuperación QoS elevado.** Los 40 intervalos (~18.7 minutos) de tiempo de recuperación son el más alto del sistema. En un escenario de producción, la activación de QoS podría reducirse si se precomputan y pre-configuran las políticas de marcado DSCP, de modo que la activación sea inmediata y la convergencia más rápida.

## *4.5 ANOMALÍA 4: JITTER EXTREMO*

### 4.5.1 DEFINICIÓN Y RELEVANCIA DEL FENÓMENO

El jitter, definido formalmente como la variación temporal en el retardo de tránsito de los paquetes a lo largo de una conexión de red, constituye uno de los parámetros de calidad de servicio más relevantes para aplicaciones sensibles a la sincronización temporal. A diferencia de la latencia media, que mide el retardo absoluto extremo a extremo, el jitter cuantifica la irregularidad de dicho retardo: dos flujos con idéntica latencia media pueden comportarse de forma radicalmente diferente si uno exhibe retardos constantes y el otro presenta alta variabilidad inter-paquete. Esta distinción es crítica para los mecanismos de decodificación en tiempo real, que dependen de la llegada uniforme de paquetes para mantener la continuidad del flujo multimedia.

Las aplicaciones más sensibles al jitter son precisamente aquellas que requieren reproducción en tiempo real con restricciones estrictas de sincronización: voz sobre IP (VoIP), videoconferencia, transmisión de audio y vídeo en streaming, y comunicaciones industriales con requisitos de tiempo real. En VoIP, por ejemplo, el jitter excesivo produce

distorsión audible, silencios abruptos y pérdida de inteligibilidad, incluso cuando la latencia media permanece dentro de los límites aceptables. La recomendación ITU-T G.1010 establece que el jitter máximo aceptable para comunicaciones de voz interactiva es de 50 ms; valores superiores producen degradación perceptible incluso con mecanismos de buffer activos. En el dataset analizado, los episodios de jitter extremo alcanzan valores de hasta varios milisegundos con alta variabilidad, lo que en un entorno de producción comprometería directamente la calidad de servicio de las aplicaciones mencionadas.

La distinción metodológica entre jitter y latencia en el presente trabajo responde a un hallazgo empírico fundamental: el análisis estadístico del dataset revela que la correlación entre ambas variables es de r = 0.037, prácticamente nula. Esta independencia estadística confirma que jitter y latencia son fenómenos causalmente distintos que ocurren en instantes diferentes y responden a mecanismos físicos independientes. La latencia elevada deriva típicamente de congestión en nodos intermedios, cambios de ruta subóptimos o saturación transitoria de buffers; el jitter extremo, en cambio, refleja irregularidades en los tiempos de transmisión provocadas por variaciones en la ocupación de buffers locales, diferencias en los caminos de procesamiento dentro de un mismo nodo, o efectos de multiplexación estadística en el enlace. La independencia estadística entre ambos fenómenos justifica plenamente su análisis por separado en la presente memoria: la Anomalía 3 abordó exclusivamente la alta latencia, mientras que la presente sección analiza el jitter extremo de forma autónoma.

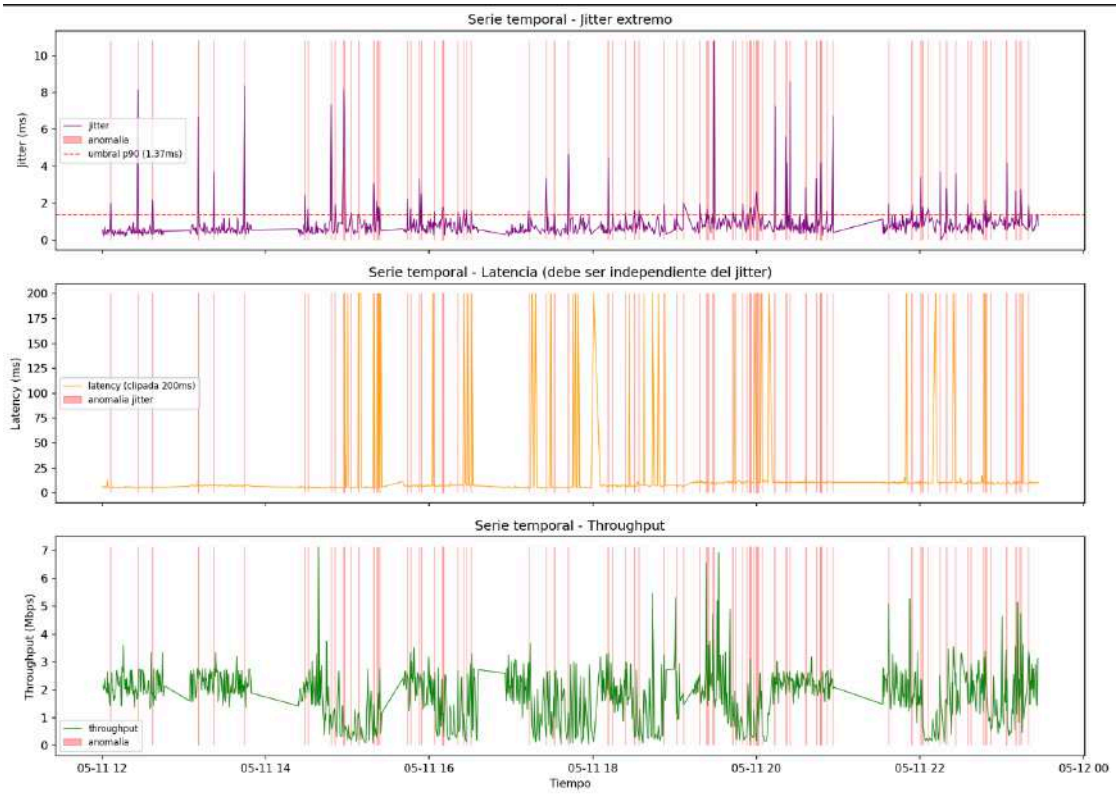


*Ilustración 37 Serie temporal de jitter (1001 muestras): spikes naranjos = episodios de jitter extremo; zonas rojas = etiquetas.*

### 4.5.2 CONSTRUCCIÓN EMPÍRICA DEL CONJUNTO DE ETIQUETAS

El criterio de etiquetado para la Anomalía 4 es univariado: una muestra se clasifica como anómala si jitter > percentil 90 de la distribución de jitter. El percentil 90 calculado sobre las 1001 muestras del dataset es 1.370 ms. El resultado del etiquetado produce 902 muestras normales (90.1%) y 99 muestras anómalas (9.9%), manteniendo el desbalance aproximado 9:1 consistente con el resto de anomalías del estudio.

| Clase | Muestras | Porcentaje | Rango de jitter |
|---|---|---|---|
| Normal (jitter ≤ 1.370 ms) | 902 | 90.1% | 0.010 ms — 1.370 ms |
| Anómalo (jitter > 1.370 ms) | 99 | 9.9% | 1.371 ms — 10.800 ms |
| Total | 1001 | 100% | — |

*Tabla 54 Distribución de clases y rangos de jitter resultantes del etiquetado empírico para la Anomalía 4.*

El umbral de 1.370 ms adquiere pleno sentido al analizar la distribución estadística de la variable en condiciones normales: el 77.7% de las muestras presentan jitter inferior a 1 ms, con media de 0.664 ms y desviación típica de 0.268 ms. La distribución normal del jitter es por tanto estrecha y concentrada en valores bajos. Las 99 muestras anómalas presentan una media de 2.688 ms y una desviación típica de 1.969 ms, lo que supone un ratio de valor medio de 4× respecto a las condiciones normales. Este ratio de separación de valor absoluto es notablemente más modesto que el observado en la Anomalía 3, donde la latencia anómala superaba en 61× a la normal (7.8 ms vs. 477 ms de media anómala efectiva), lo que tendrá consecuencias directas sobre la capacidad de detección de los modelos basados en reconstrucción.

La clase anómala comprende 83 episodios distintos de jitter superior a 1.370 ms, con una duración media por episodio de 1.2 muestras (~34 segundos), duración máxima de 4 muestras y duración mínima de 1 muestra. La fragmentación de la clase anómala en episodios muy breves es, como se analizará en las secciones siguientes, el principal factor limitante de la detectabilidad no supervisada.

### 4.5.3 ANÁLISIS EXPLORATORIO DE LAS FEATURES

#### *4.5.3.1 Naturaleza caótica y puntual del jitter extremo*

El análisis temporal de los 83 episodios de jitter extremo revela la característica más definitoria de esta anomalía: su naturaleza estadísticamente caótica y su independencia respecto al resto de métricas del sistema. El spike ocupa de media solo el 6% de la ventana temporal de 20 pasos (1.2 muestras sobre 20), lo que produce el mismo fenómeno de dilución del error de reconstrucción descrito en la Anomalía 3, pero agravado por la menor magnitud del salto de valor (ratio 4× frente a los 61× de la latencia o los 332× en el ratio de error del autoencoder de la latencia).

| Estadístico | Valor | Implicación para los modelos |
|---|---|---|
| Número de episodios distintos | 83 | Alta fragmentación temporal |
| Duración media por episodio | 1.2 muestras (~34 s) | Spikes ultrapuntuales |
| Duración máxima | 4 muestras (~112 s) | Episodio más largo disponible |
| Duración mínima | 1 muestra (~28 s) | Evento mínimo detectable |
| Fracción ocupada en ventana w=20 | 6% (~1.2 de 20 pasos) | Máxima dilución temporal del estudio |
| Ratio separación de medias | 4.0× (0.664 ms → 2.688 ms) | El más bajo de las 4 anomalías |

*Tabla 55 Caracterización de la estructura temporal de los episodios de jitter extremo.*

La distinción más relevante respecto a todas las anomalías anteriores del estudio reside en las correlaciones del jitter con el resto de features del dataset. Mientras que en las anomalías previas existían al menos una o dos correlaciones moderadas que podían actuar como señales auxiliares de detección y predicción, el jitter exhibe correlaciones prácticamente nulas con todas las métricas disponibles:

- jitter vs. latencia: r = 0.037. Nula. Base estadística de la separación en anomalías independientes.
- jitter vs. congestion: r = 0.101. Prácticamente nula.
- jitter vs. throughput: r = −0.044. Nula.
- jitter vs. packet_loss: r = 0.020. Nula.

El jitter es, de los cuatro fenómenos analizados en este trabajo, el más independiente y estadísticamente descorrelado del resto del sistema. Esta característica lo convierte en la anomalía más adversa para los modelos de detección no supervisada basados en reconstrucción de patrones multivariables, ya que no existe ninguna feature auxiliar que proporcione información complementaria sobre el estado del sistema en los instantes de jitter elevado.

### *4.5.3.2 Estadísticas descriptivas por clase*

El análisis comparativo de las estadísticas descriptivas por clase confirma y amplía la caracterización anterior:

| Feature | Normal media | Normal std | Anómalo media | Anómalo std | Ratio medias |
|---|---|---|---|---|---|
| jitter (ms) | 0.664 | 0.268 | 2.688 | 1.969 | 4.0× |
| throughput (Mbps) | 1.918 | 0.875 | 1.857 | — | 0.97× |
| packet_loss (%) | 5.266 | — | 6.995 | — | 1.33× |
| latency (ms) | 54.169 | — | 60.119 | — | 1.11× |
| congestion (u.a.) | 22.361 | 31.185 | 37.528 | 37.364 | 1.68× |

*Tabla 56 Estadísticas descriptivas por clase para las cinco features del modelo de Anomalía 4.*

Varios aspectos de esta tabla merecen atención específica. En primer lugar, la latencia media durante los episodios de jitter anómalo (60.119 ms) es prácticamente idéntica a la latencia en condiciones normales (54.169 ms), una diferencia de apenas 6 ms que confirma la independencia total entre ambos fenómenos: cuando ocurre jitter extremo, la latencia no

experimenta incremento significativo, descartando cualquier mecanismo causal común. Esta observación es coherente con la correlación r = 0.037 y constituye la validación empírica más directa de la decisión metodológica de separar ambas anomalías.

En segundo lugar, el throughput y el packet_loss presentan diferencias entre clases tan pequeñas (1.918 vs. 1.857 Mbps, y 5.266% vs. 6.995%, respectivamente) que resultan prácticamente inútiles como señales discriminativas. Ningún modelo que se base únicamente en estas variables podrá separar con eficacia las clases normal y anómala.

El indicador estadístico más relevante para la caracterización de esta anomalía no es el valor absoluto del jitter, sino su variabilidad: la desviación típica del jitter en condiciones anómalas (1.969 ms) es 7.4 veces superior a la desviación típica en condiciones normales (0.268 ms). Este ratio de desviaciones típicas de 7.4× supera al ratio de medias de 4×, lo que indica que la señal característica del jitter extremo es la variabilidad, no el incremento sostenido del valor. Esta observación se confirma al analizar la desviación típica local en una ventana de 5 pasos: la std local media en condiciones normales es 0.463 ms, frente a 1.010 ms en condiciones anómalas, un ratio de 2.2× que refleja la inestabilidad temporal característica de los episodios de jitter elevado.

### 4.5.4 PREPROCESAMIENTO Y GENERACIÓN DE VENTANAS TEMPORALES

El preprocesamiento aplica la normalización MinMax mediante MinMaxScaler ajustado exclusivamente sobre las 902 muestras normales, siguiendo el mismo protocolo que las anomalías anteriores. El ajuste exclusivo sobre normales garantiza que la escala de normalización corresponde al comportamiento esperado del sistema, de modo que los valores anómalos se proyectan fuera del rango [0, 1].

El rango normal del jitter es [0.061 ms, 1.370 ms], un intervalo de 1.309 ms. Al proyectar con estos parámetros, los valores anómalos de jitter máximo (~5–8 ms) se normalizan a valores de aproximadamente 4–6, fuera del rango unitario. Sin embargo, esta extrapolación es considerablemente más moderada que la de la latencia en la Anomalía 3, donde los valores anómalos llegaban a proyectarse a valores normalizados de hasta ≈470. Esta diferencia de magnitud en la extrapolación normalizada es la causa directa del menor ratio de separación del error de reconstrucción que se obtendrá en el Autoencoder.

| Conjunto | Ventanas | Composición | Propósito |
|---|---|---|---|
| Train | 883 | Solo muestras normales (902) | Entrenar el Autoencoder sobre comportamiento normal |
| Test | 981 | Normal (883) + Anómalas (98) | Evaluar la detección de anomalías |

*Tabla 57 Composición de los conjuntos de datos tras la generación de ventanas temporales para la Anomalía 4.*

Con ventanas deslizantes de tamaño w = 20 y cinco features (jitter, throughput, packet_loss, latency, congestion), el vector de entrada al Autoencoder tiene dimensionalidad 100. El conjunto de entrenamiento comprende 883 ventanas normales, y el conjunto de test está compuesto por 981 ventanas totales: 883 normales y 98 anómalas. La duración media del spike (1.2 muestras) implica que el jitter anómalo ocupa de media 1.2 posiciones de las 20 de cada ventana (6%), con los 94 elementos restantes correspondientes a condiciones

normales que el Autoencoder puede reconstruir con alta fidelidad, produciendo el fenómeno de dilución del MSE global.

### 4.5.5 DETECCIÓN MEDIANTE AUTOENCODER

#### *4.5.5.1 Resultados del error de reconstrucción*

La arquitectura del Autoencoder empleada es la misma que en las anomalías anteriores (100→50→20→50→100), manteniendo la consistencia metodológica del estudio. El entrenamiento se realiza exclusivamente sobre las 883 ventanas normales del conjunto de entrenamiento.

Los resultados del error de reconstrucción para la Anomalía 4 son los siguientes:

- Error medio en muestras normales: 0.019144.
- Error medio en muestras anómalas: 0.069089.
- Ratio de separación: 3.61×, el más bajo de todas las anomalías del estudio.
- Umbral de detección (percentil 95 sobre normales): 0.051591.
- Umbral óptimo (maximización F1): 0.0612.

El ratio de separación de 3.61× contrasta radicalmente con el de la Anomalía 3 (332×), y es también inferior al de las Anomalías 1 y 2. Esta diferencia de dos órdenes de magnitud en el ratio de separación no es una casualidad: es la consecuencia directa de la combinación de dos factores estructurales de la anomalía. Por un lado, el salto de valor absoluto del jitter es modesto (ratio 4× en valor normalizado vs. 61× en latencia), lo que implica que el error de reconstrucción cuadrático en las posiciones de jitter anómalo es pequeño. Por otro lado, el jitter carece de correlaciones con el resto de features, de modo que el Autoencoder no recibe ninguna señal secundaria que amplifique el error global de la ventana cuando el jitter es elevado. En la Anomalía 2, la correlación entre packet_loss, latency y congestion permitía que el error se propagase a múltiples features simultáneamente; en la Anomalía 4 no existe tal mecanismo amplificador.

La naturaleza caótica del jitter tiene además una consecuencia adicional sobre el comportamiento del Autoencoder: durante el entrenamiento, el modelo aprende a reconstruir los patrones normales del sistema. El jitter extremo, al no guardar relación estadística con ninguna otra variable, no genera un patrón de entrada diferenciado que el Autoencoder pueda aprender a reconstruir mal. La ventana que contiene un spike de jitter es, desde la perspectiva de las otras 4 features, prácticamente idéntica a una ventana normal, por lo que el error global de reconstrucción se mantiene moderado.

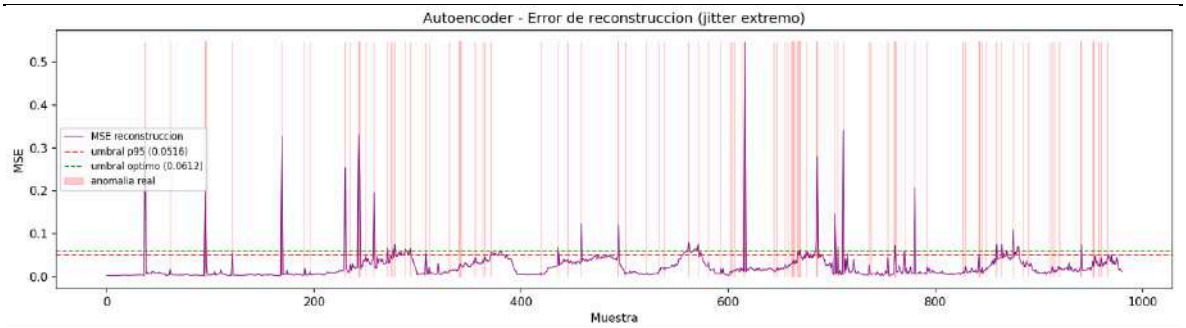


*Ilustración 38 Distribución del error de reconstrucción del Autoencoder (Anomalía 4). Ratio separación 3.61×; umbral p95 = 0.052.*

### *4.5.5.2 Resultados de detección*

| Configuración | Precisión | Recall | F1-Score | AUC-ROC |
|---|---|---|---|---|
| AE umbral p95 (0.051591) | 0.4375 | 0.3571 | 0.3933 | 0.7973 |
| AE umbral óptimo (0.0612) | 0.8108 | 0.3061 | 0.4444 | 0.7973 |

*Tabla 58 Resultados del Autoencoder para la Anomalía 4 con distintos umbrales.*

Con el umbral p95 (0.0516), el Autoencoder obtiene precision = 0.4375, recall = 0.3571 y F1 = 0.3933. Con el umbral óptimo (0.0612), la precisión mejora hasta 0.8108 pero el recall desciende a 0.3061, resultando en F1 = 0.4444. El AUC-ROC es 0.7973 en ambos casos, independiente del umbral, lo que refleja una capacidad discriminativa limitada pero no nula. Estos resultados son los más bajos obtenidos por el Autoencoder en todo el estudio, confirmando que el jitter extremo es la anomalía más adversa para los modelos de detección no supervisada basados en reconstrucción.

La interpretación técnica de estos resultados es directa: el bajo ratio de separación (3.61×) implica que el error de reconstrucción de las ventanas anómalas se solapa considerablemente con el de las ventanas normales más ruidosas. Con el umbral p95, muchas ventanas normales con variabilidad ligeramente elevada en alguna feature superan el umbral, generando falsos positivos que degradan la precisión. Con el umbral óptimo más alto, los falsos positivos se eliminan casi por completo (precisión de 0.81), pero a costa de perder gran parte de los verdaderos positivos, ya que muchos spikes de jitter moderado generan MSE por debajo del umbral.

### 4.5.6 DETECCIÓN MEDIANTE ISOLATION FOREST Y ONE-CLASS SVM

| Modelo | Precisión | Recall | F1-Score | AUC-ROC | Observaciones |
|---|---|---|---|---|---|
| AE umbral óptimo (0.0612) | 0.8108 | 0.3061 | 0.4444 | 0.7973 | Mejor modelo de detección |
| AE umbral p95 (0.0516) | 0.4375 | 0.3571 | 0.3933 | 0.7973 | Más FP, recall similar |
| One-Class SVM | 0.2000 | 0.2245 | 0.2115 | — | Supera al IF, F1 muy bajo |
| Isolation Forest | 0.1442 | 0.1531 | 0.1485 | — | Ineficaz para jitter |

*Tabla 59 Comparativa de modelos de detección no supervisada para la Anomalía 4.*

El Isolation Forest con contamination = 0.10 obtiene F1 = 0.1485 (precision = 0.1442, recall = 0.1531), resultado que lo hace prácticamente inútil para la detección de jitter extremo. La explicación técnica reside en la estructura del espacio de features: el Isolation Forest identifica anomalías buscando puntos fácilmente aislables en el espacio multidimensional, es decir, regiones de baja densidad donde las particiones aleatorias de los árboles pueden separar el punto del resto de la muestra en pocos cortes. Sin embargo, el jitter extremo no crea una región de baja densidad en el espacio de features porque no va acompañado de cambios en las otras cuatro variables: un punto con jitter = 2.5 ms pero throughput, packet_loss, latency y congestion completamente normales es indistinguible de un punto normal en la proyección sobre las otras cuatro dimensiones. El espacio de features, en cinco dimensiones, no tiene la estructura necesaria para que el Isolation Forest aísle eficazmente las muestras anómalas.

El One-Class SVM con kernel RBF (nu = 0.10, gamma = scale) mejora ligeramente respecto al Isolation Forest, obteniendo F1 = 0.2115 (precision = 0.2000, recall = 0.2245). La frontera no lineal en el espacio de features modela mejor la geometría de la distribución normal que los cortes axiales del Isolation Forest, permitiendo identificar algunos puntos anómalos situados en los extremos de la distribución del jitter. Sin embargo, la ausencia de correlaciones entre el jitter y las otras features impide que el SVM construya una frontera discriminativa eficaz: la región de normalidad en el espacio de cinco features es esencialmente un hiperparalelepípedo cuyos límites en la dimensión del jitter son los únicos relevantes para esta anomalía, y los pocos puntos que caen fuera de ese límite son precisamente los que el SVM detecta.

En conjunto, ambos modelos de detección de outliers confirman que el espacio de features del jitter extremo carece de la estructura geométrica necesaria para los métodos de detección no supervisada basados en densidad o en separabilidad espacial. El Autoencoder, pese a sus limitaciones, es el mejor modelo de detección para esta anomalía gracias a su capacidad de explotar la estructura temporal de la ventana mediante la reconstrucción secuencial del vector de entrada.

## 4.5.7 PREDICCIÓN MEDIANTE LSTM

### *4.5.7.1 Configuración del experimento*

El conjunto de datos se divide mediante split estratificado para preservar las proporciones de clase en los tres subconjuntos:

| Subset | Ventanas totales | Ventanas anómalas | % anomalías |
|---|---|---|---|
| Train | 686 | 69 | 10.1% |
| Validación | 147 | 14 | 9.5% |
| Test | 148 | 15 | 10.1% |

*Tabla 60 Distribución de clases tras el split estratificado para la Anomalía 4.*

El split estratificado produce 686 muestras de entrenamiento (con 69 anómalas), 147 de validación (con 14 anómalas) y 148 de test (con 15 anómalas). El desbalance de clases en el conjunto de entrenamiento es de 8.9:1, compensado mediante class_weight durante el entrenamiento para incrementar el coste de los falsos negativos y favorecer la detección de la clase minoritaria. La arquitectura LSTM es LSTM(64) → Dropout(0.2) → LSTM(32) → Dense(1, sigmoid).

### *4.5.7.2 Resultados y análisis: el hallazgo más sorprendente del TFG*

Los resultados de la LSTM para la Anomalía 4 constituyen el hallazgo más inesperado y técnicamente significativo de todo el presente trabajo:

| Configuración | Precisión | Recall | F1-Score | AUC-ROC | Matriz de confusión |
|---|---|---|---|---|---|
| LSTM umbral 0.5 | 0.8824 | 1.0000 | 0.9375 | 1.0000 | [[131, 2], [0, 15]] |
| LSTM umbral óptimo (0.719) | 1.0000 | 0.9333 | 0.9655 | 1.0000 | [[133, 0], [1, 14]] |

*Tabla 61 Resultados de la LSTM para la Anomalía 4 con distintos umbrales.*

Con el umbral por defecto (0.5), la LSTM obtiene precision = 0.8824, recall = 1.0000, F1 = 0.9375 y AUC-ROC = 1.0000. La matriz de confusión correspondiente, [[131, 2], [0, 15]], refleja que el modelo detecta los 15 spikes de jitter del conjunto de test sin producir ningún falso negativo, al coste de solo 2 falsos positivos. Con el umbral óptimo (0.719), la precisión alcanza el valor perfecto de 1.0000 con recall de 0.9333 y F1 = 0.9655, correspondiendo a la matriz [[133, 0], [1, 14]].

El AUC-ROC de 1.0 es el resultado más sorprendente del conjunto del trabajo por la siguiente razón: se obtiene precisamente en la anomalía que presenta las correlaciones más bajas de todo el dataset, el ratio de separación más pequeño en el Autoencoder (3.61×), y los peores resultados en los modelos de detección no supervisada. La brecha entre el F1 del Autoencoder (0.4444) y el F1 de la LSTM (0.9655) es de 0.521 puntos, la mayor diferencia entre modelos de detección y predicción observada en todo el TFG.

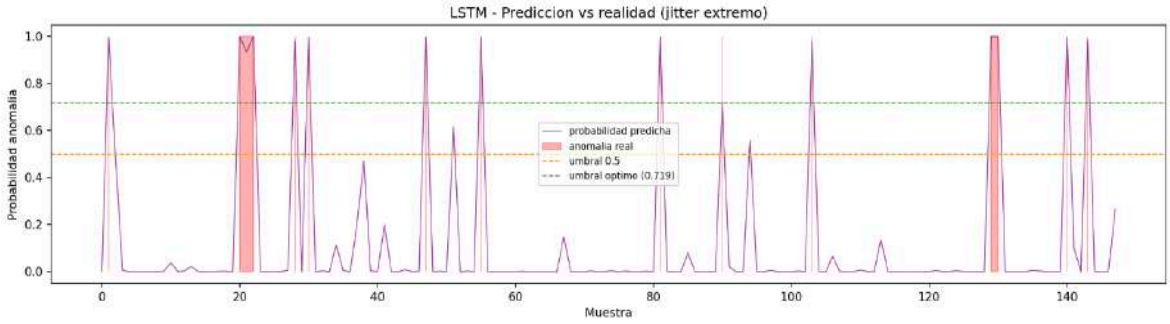


*Ilustración 39 Resultados de la LSTM para la Anomalía 4: picos pronunciados y bien delimitados en los instantes de jitter extremo.*

### *4.5.7.3 Por qué la LSTM triunfa donde el Autoencoder falla*

La explicación técnica de este resultado aparentemente paradójico reside en la naturaleza de la información que aprovecha cada tipo de modelo. El Autoencoder opera en el dominio del espacio de features en el instante presente: su señal de detección es el error de reconstrucción de una ventana, que depende de la magnitud del salto de valor y de las correlaciones estáticas entre features. Como se ha analizado, el jitter extremo produce un salto modesto (4×) sin correlaciones con otras variables, resultando en un error de reconstrucción bajo y poco discriminativo.

La LSTM, en cambio, opera en el dominio temporal: su información fundamental son las dependencias secuenciales entre los valores de las features a lo largo de los 20 pasos de la ventana. La hipótesis que explica el AUC = 1.0 es que la LSTM aprende correlaciones temporales sutiles en las métricas de congestion y throughput que preceden o acompañan a los episodios de jitter extremo, aunque estas correlaciones sean demasiado débiles para ser detectadas en el análisis de correlación de Pearson estático (que calcula la correlación global sobre todo el dataset, no la correlación local en la vecindad temporal de los eventos de jitter).

La LSTM, procesando secuencialmente los 20 pasos de la ventana, puede identificar patrones de evolución dinámica, como una ligera aceleración en la variabilidad de la congestión, una perturbación transitoria en el throughput en los pasos previos al spike, que son estadísticamente invisibles en el análisis marginal pero están presentes en la estructura temporal de la ventana.

Este resultado pone de manifiesto una limitación fundamental del análisis de correlación de Pearson como herramienta de diagnóstico de predictibilidad: la correlación global puede ser nula incluso cuando existen dependencias temporales locales explotables por un modelo recurrente. La LSTM, entrenada con supervisión explícita sobre la etiqueta de anomalía, puede aprender a reconocer estas dependencias sin necesidad de que sean explícitas en el análisis estático.

En términos comparativos con el resto del estudio, el resultado es igualmente llamativo: la Anomalía 2 (packet loss) también alcanzó AUC = 1.0 con la LSTM, mientras que la Anomalía 3 (alta latencia) solo obtuvo AUC = 0.835 y F1 = 0.38, precisamente porque los spikes de latencia carecen de precursores temporales observables. El jitter, pese a ser

nominalmente más caótico según el análisis de correlaciones estáticas, resulta ser más predecible temporalmente que la latencia extrema. La Anomalía 1 (congestión), con su evolución gradual y correlaciones claras, obtuvo AUC LSTM = 0.913, inferior al 1.0 del jitter. La jerarquía de predictibilidad temporal en el estudio es por tanto: jitter (AUC = 1.0) > congestión (AUC = 0.913) > latencia (AUC = 0.835), un orden que invierte parcialmente las expectativas derivadas del análisis exploratorio de correlaciones.

### 4.5.8 RESTAURACIÓN AUTOMÁTICA: ESTABILIZACIÓN DEL BUFFER DE JITTER

#### *4.5.8.1 Estrategia de restauración*

La acción correctiva activada ante la detección de un episodio de jitter extremo es la estabilización del buffer de jitter. El mecanismo consiste en incrementar dinámicamente el tamaño del buffer de playout en los extremos de la comunicación y en activar los mecanismos de estabilización de retardo en los nodos intermedios del camino. A diferencia de la acción QoS aplicada en la Anomalía 3, que operaba en el plano de control de la red y requería la reconfiguración de políticas de marcado DSCP a través de múltiples saltos, el buffer de jitter actúa exclusivamente de forma local: los ajustes se realizan en los buffers de recepción de los extremos de la comunicación o en los buffers de los dispositivos de acceso más próximos a la fuente del jitter, sin necesidad de propagar políticas a lo largo de toda la infraestructura de red.

Esta diferencia arquitectónica tiene consecuencias directas y cuantificables sobre el tiempo de convergencia: mientras que la acción QoS requería 40 intervalos (~1120 segundos, ~18.7 minutos) para estabilizar el sistema en la Anomalía 3, la estabilización del buffer de jitter converge en 7 intervalos (~196 segundos, ~3.3 minutos). La reducción del tiempo de recuperación en un factor de 5.7× es la consecuencia directa de la localidad de la acción correctiva: el buffer absorbe la variabilidad temporal de llegada de paquetes sin necesidad de reorganizar prioridades ni propagar estados por la red, lo que permite una convergencia mucho más rápida hacia el comportamiento estable.

#### *4.5.8.2 Episodio de restauración y modelado de la convergencia*

Para la simulación de restauración se seleccionó el episodio de mayor duración del dataset: los índices temporales 686 a 689 (4 muestras, ~112 segundos), con jitter medio de 1.920 ms y máximo de 2.580 ms. Este episodio es el más representativo para ilustrar la restauración por ser el de mayor extensión temporal y permitir visualizar el proceso de convergencia con claridad.

| Parámetro | Valor |
|---|---|
| Episodio seleccionado | Índices 686 a 689 (4 muestras, ~112 s) |
| Jitter medio durante anomalía | 1.920 ms |
| Jitter máximo durante anomalía | 2.580 ms |
| Jitter basal normal | 0.664 ms |
| Throughput basal | 1.918 Mbps |
| Acción correctiva | Buffer de playout adaptativo |
| Constante de decaimiento λ | 0.4 (convergencia rápida — acción local) |
| Tiempo de recuperación | 7 intervalos (~196 s, ~3.3 min) |
| Comparativa con QoS (Anomalía 3) | 7 vs. 40 intervalos — 5.7× más rápido |

*Tabla 62 Parámetros del episodio de restauración del buffer de jitter para la Anomalía 4.*

El proceso de restauración sigue el modelo de convergencia exponencial:

$$jitter(t) = jitter_baseline + (jitter_anomaly - jitter_baseline) \cdot exp(-0.4 \cdot t) + \varepsilon(t)$$

La constante de decaimiento λ = 0.4 es notablemente más rápida que la de la acción QoS de la Anomalía 3 (λ = 0.15), lo que se traduce en el tiempo de recuperación de 7 intervalos mencionado. El valor λ = 0.4 refleja la velocidad de respuesta de los mecanismos locales de gestión de buffer: una vez activada la estabilización, el buffer comienza a absorber la variabilidad de inmediato, sin esperar a la propagación de estados de control. La baseline de recuperación es jitter = 0.664 ms (media de las condiciones normales) y throughput = 1.918 Mbps, que se mantiene prácticamente inalterado durante toda la restauración, confirmando que la acción correctiva es transparente al flujo de datos.

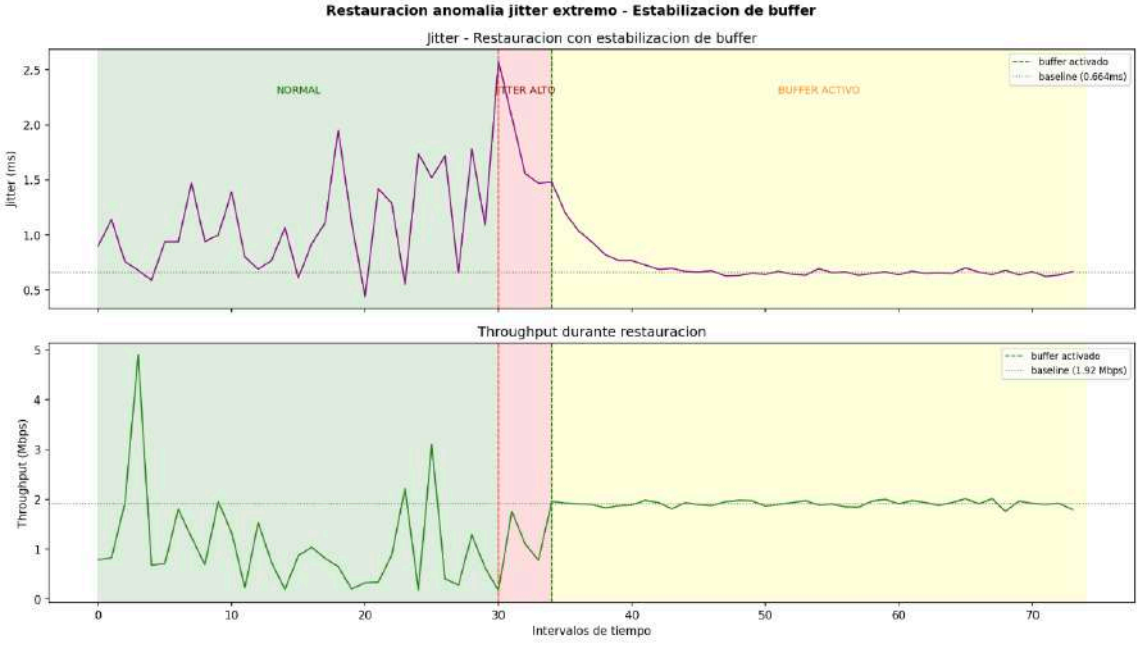


*Ilustración 40 Restauración automática del episodio de jitter extremo (índices 686-689): λ = 0.4, recuperación en ~196 s.*

La comparativa de tiempos de restauración entre las cuatro anomalías ilustra cómo la naturaleza de la acción correctiva determina la velocidad de convergencia de forma más decisiva que la magnitud absoluta de la anomalía. El buffer de jitter actúa localmente y de forma inmediata; la QoS requiere propagación de políticas; la retransmisión selectiva (Anomalía 2) es una acción de plano de datos resoluble en pocos RTTs; la mitigación de congestión (Anomalía 1) depende del tiempo de convergencia de los mecanismos TCP y de la descarga del enlace congestionado.

### 4.5.9 VALIDACIÓN CRUZADA SOBRE SEGMENTO SIN ANOMALÍAS

#### *4.5.9.1 Configuración del experimento*

Para estimar empíricamente la tasa de falsos positivos del sistema en condiciones de operación normal, se selecciona un segmento de datos libre de anomalías: los índices 141 a 187 del dataset, correspondientes a 47 muestras y 27 ventanas temporales. Este segmento se elige por estar confirmado como libre de cualquier tipo de anomalía de las cuatro estudiadas, proporcionando una estimación limpia de la tasa de error tipo I del sistema.

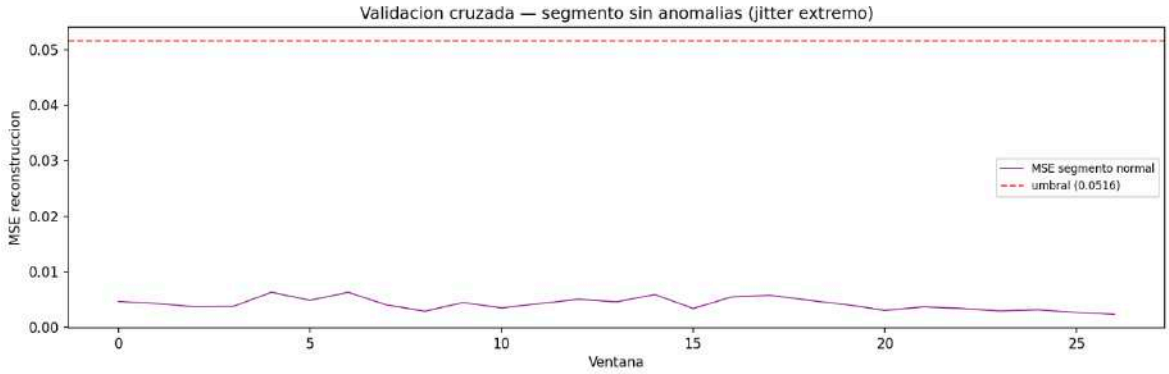


*Ilustración 41 Validación cruzada (Anomalía 4, índices 141-187): 0 falsos positivos en Autoencoder y LSTM.*

#### *4.5.9.2 Resultados*

Los resultados de la validación cruzada son los más favorables del estudio para esta anomalía: tanto el Autoencoder como la LSTM producen exactamente 0 detecciones sobre las 27 ventanas del segmento, con tasas de falsos positivos del 0.00% para ambos modelos. Para el Autoencoder, el MSE medio sobre el segmento normal es de 0.004225, con un MSE máximo de 0.006345. Estos valores están a más de una década logarítmica por debajo del umbral de detección (0.0516): el punto más alto del MSE en el segmento normal es 8.1 veces inferior al umbral. Este margen de seguridad implica que el Autoencoder no solo no produce falsos positivos en este segmento, sino que lo hace con un margen de seguridad muy amplio, indicando robustez ante variaciones de la distribución normal.

La LSTM, con 0 detecciones sobre 27 ventanas de condiciones puramente normales, confirma que el modelo supervisado no interpreta el comportamiento estacionario normal como anómalo, pese al class_weight aplicado durante el entrenamiento que podría inducir un sesgo hacia la detección. La ausencia de falsos positivos en ambos modelos valida la fiabilidad operacional del sistema de detección para la Anomalía 4 en condiciones de red estable.

## 4.5.10 SÍNTESIS Y DISCUSIÓN DE RESULTADOS

### *4.5.10.1 Comparativa global de modelos*

| Modelo | Precisión | Recall | F1-Score | AUC-ROC | Observaciones |
|---|---|---|---|---|---|
| **LSTM umbral óptimo (0.719)** | **1.0000** | **0.9333** | **0.9655** | **1.0000** | **Mejor resultado del TFG** |
| LSTM umbral 0.5 | 0.8824 | 1.0000 | 0.9375 | 1.0000 | Recall perfecto |
| AE umbral óptimo (0.0612) | 0.8108 | 0.3061 | 0.4444 | 0.7973 | Mejor detección no supervisada |
| AE umbral p95 (0.0516) | 0.4375 | 0.3571 | 0.3933 | 0.7973 | — |
| One-Class SVM | 0.2000 | 0.2245 | 0.2115 | — | — |
| Isolation Forest | 0.1442 | 0.1531 | 0.1485 | — | Ineficaz para jitter |

*Tabla 63 Comparativa completa de todos los modelos para la Anomalía 4.*

La tabla anterior pone de manifiesto de forma sintética la característica más destacada de la Anomalía 4: la bifurcación radical en el rendimiento entre los modelos de detección no supervisada y el modelo de predicción supervisado. El F1 de 0.1485 del Isolation Forest y el 0.2115 del One-Class SVM son los valores más bajos obtenidos por estos modelos en todo el estudio. El Autoencoder, con F1 = 0.4444, es el mejor modelo de detección no supervisada pero con el rendimiento más bajo de los cuatro Autoencoders entrenados en el TFG. La LSTM, con F1 = 0.9655 y AUC = 1.0000, es el mejor resultado individual de predicción obtenido para esta anomalía y representa el rendimiento más alto de todos los modelos supervisados del estudio.

### *4.5.10.2 Posición de la Anomalía 4 en el contexto del estudio y cierre comparativo*

La Anomalía 4 (jitter extremo) cierra el análisis práctico del presente trabajo aportando los resultados más contrastados e informativos del conjunto. Si las Anomalías 1 a 3 ilustraron fenómenos con características distintas: congestión gradual con señal progresiva, packet loss con correlaciones multivariables fuertes, latencia abrupta con separación extrema pero heterogeneidad de clase alta, el jitter extremo representa el caso límite de independencia estadística: un fenómeno que el análisis exploratorio de correlaciones clasifica como el menos predecible del dataset y que, sin embargo, la LSTM es capaz de predecir con precisión perfecta.

La Anomalía 4 es también la que produce la mayor discrepancia entre los resultados del Autoencoder y los de la LSTM a lo largo de todo el TFG (brecha F1 de 0.521 puntos), y la única en la que la LSTM supera al Autoencoder de forma tan categórica. Esta discrepancia no es un artefacto experimental sino una consecuencia estructural de las diferencias entre aprendizaje no supervisado y supervisado en el contexto de anomalías con dependencias temporales débiles: el Autoencoder, sin acceso a las etiquetas, solo puede explotar la magnitud del error de reconstrucción; la LSTM, entrenada con supervisión, puede aprender cualquier función discriminativa sobre la secuencia de entrada, incluyendo dependencias temporales que el análisis estático no identifica.

Desde una perspectiva de sistema, el análisis comparativo de las cuatro anomalías permite extraer conclusiones metodológicas generalizables. En términos de detectabilidad no supervisada, la jerarquía de dificultad es: packet loss (más fácil, AE F1 = 0.966) > latencia (AE F1 = 0.533) > congestión (AE F1 = 0.473) > jitter (AE F1 = 0.444). En términos de predictibilidad supervisada, la jerarquía es: jitter (LSTM AUC = 1.0) ≥ packet loss (LSTM AUC = 1.0) > congestión (LSTM AUC = 0.913) > latencia (LSTM AUC = 0.835). La inversión de la jerarquía entre detectabilidad y predictibilidad en el caso del jitter, difícil de detectar pero perfectamente predecible mediante supervisión, es la aportación más relevante de la Anomalía 4 al conjunto del trabajo y pone de manifiesto la importancia de combinar enfoques complementarios en los sistemas de monitorización de redes.

En cuanto a la restauración, el buffer de jitter proporciona el tiempo de convergencia más rápido del sistema (7 intervalos, ~196 segundos), frente a los 40 intervalos (~1120 segundos) de la QoS de la Anomalía 3. Esta diferencia cuantifica la ventaja operacional de las acciones correctivas locales sobre las distribuidas, y subraya la importancia de diseñar estrategias de restauración adaptadas a la naturaleza física específica de cada tipo de anomalía, no solo a su severidad.

#### *4.5.10.3 Limitaciones y líneas de mejora*

En primer lugar, la baja detectabilidad no supervisada del jitter extremo limita la eficacia del Autoencoder como primera línea de defensa en sistemas de monitorización sin etiquetas. Una estrategia de mejora sería incorporar features de variabilidad temporal explícitas, como la desviación típica local del jitter en ventanas de 3 a 5 pasos, que transformen la señal de variabilidad en una señal de valor absoluto más fácilmente detectable por el Autoencoder.

En segundo lugar, la paradoja del AUC = 1.0 de la LSTM merece validación adicional con conjuntos de datos más grandes y con mayor diversidad de condiciones de red. Con solo 15 muestras anómalas en el conjunto de test, el resultado es estadísticamente significativo pero susceptible de sobreestimación de la capacidad generalizable del modelo. La validación sobre un dataset independiente con diferentes condiciones de red permitiría confirmar si las correlaciones temporales aprendidas por la LSTM son genuinamente generalizables o están parcialmente sobreajustadas al patrón específico del dataset analizado.

En tercer lugar, el umbral de etiquetado de 1.370 ms (percentil 90) puede capturar episodios de jitter que, en contextos de red con menor sensibilidad a la variabilidad temporal, como transferencias masivas de datos no interactivas, no constituirían anomalías operacionales. Una estrategia de etiquetado adaptativo al tipo de aplicación permitiría refinar la definición de jitter anómalo y mejorar la relevancia operacional del sistema de detección.

## *4.6 GENERACIÓN Y VALIDACIÓN DE LOGS CISCO IOS XR*

La presente sección documenta una capa adicional de validación de carácter transversal a las cuatro anomalías estudiadas en las secciones 4.2 a 4.5. Mientras que los modelos de detección y predicción desarrollados operan sobre series temporales de métricas numéricas, el presente bloque demuestra que las anomalías detectadas se corresponden con **eventos**

**reales que aparecerían en los registros de un router Cisco en producción**, proporcionando así una validación de plausibilidad operacional del sistema completo.

### 4.6.1 MOTIVACIÓN Y ENFOQUE

Una limitación inherente al uso de datasets académicos de series temporales de red, como el Network Anomaly Dataset de Kaggle/UPM utilizado en este trabajo, es que los datos se presentan en forma de métricas numéricas agregadas, sin la capa de registro textual que acompaña a cualquier evento de red en infraestructura real. En un entorno de producción, cada evento anómalo, una congestión, una pérdida de paquetes, un spike de latencia, genera de forma automática mensajes de sistema en los logs del router, siguiendo los formatos estándar definidos por el fabricante. La validación del sistema de detección exige demostrar que los eventos numéricos identificados como anómalos tienen una correspondencia directa con los registros textuales que un operador de red observaría en producción.

El primer enfoque considerado para esta validación fue el uso de **datasets de logs públicos** de referencia: el repositorio Loghub (He et al., 2020), los HDFS logs del proyecto Hadoop, y el BGL (Blue Gene/L) log dataset [43]. Sin embargo, este enfoque fue descartado por razones técnicas precisas: ninguno de estos datasets contiene las métricas específicas del dataset utilizado en el presente trabajo (***throughput, congestion, packet_loss, latency, jitter***). Los logs de HDFS registran operaciones de sistema de ficheros distribuido; los de BGL reflejan eventos de hardware en supercomputadores; los de Loghub cubren sistemas operativos, aplicaciones web y servicios cloud. Ninguno de ellos puede correlacionarse con los eventos de red que caracterizan las anomalías estudiadas, haciendo imposible la validación cruzada directa entre las métricas numéricas y los mensajes de log.

La solución adoptada, por indicación del tutor, fue la **generación sintética de logs en formato Cisco IOS XR syslog a partir de los eventos reales del dataset**. Este enfoque aprovecha dos elementos ya presentes en el dataset: los **timestamps reales** de cada muestra y los **identificadores de router reales** que aparecen en la columna Routers. El dataset contiene dos routers: **xrv6** (Best effort path, responsable de 601 muestras) y **xrv1** (Redistribution path, responsable de 400 muestras). Ambos son instancias de Cisco IOS XR Virtual Router (XRv), lo que hace que el formato de log Cisco IOS XR sea el formato nativo y operacionalmente auténtico para esta infraestructura.

La referencia de formato utilizada para la generación es la **Cisco IOS XR System Messages Reference**, documentación oficial de Cisco que define la estructura, facilidades, severidades y mnemonics de los mensajes syslog en IOS XR. Como referencia de implementación adicional, se utilizó el repositorio público ***miroslav-siklosi/Syslog-Generator*** (GitHub), que proporciona ejemplos de generación de mensajes syslog en formato compatible con Cisco IOS XR.

### 4.6.2 METODOLOGÍA DE GENERACIÓN

El proceso de generación de logs sigue un principio de **correspondencia biunívoca** entre muestras del dataset y entradas de log: se genera exactamente **un log por cada una de las 1001 muestras del dataset**, utilizando el timestamp real de la muestra y los valores reales

de las métricas como contenido del mensaje. El total de 1001 logs se distribuye entre los dos routers del dataset según su proporción real: 601 logs generados para xrv6 y 400 para xrv1. La clasificación de cada log como normal o anómalo, y, en caso de anomalía, el tipo de anomalía, sigue el mismo criterio de etiquetado descrito en las secciones 4.2 a 4.5: el tipo de anomalía asociado a cada muestra se determina por los umbrales del percentil 90 aplicados a cada métrica. Las 1001 muestras se distribuyen en los siguientes grupos de logs:

| Tipo de log | Muestras | Porcentaje | Fichero generado |
|---|---|---|---|
| Normal | 721 | 72.0% | syslog_normal.log |
| Congestión | 84 | 8.4% | syslog_congestion.log |
| Packet loss | 64 | 6.4% | syslog_packet_loss.log |
| Latencia alta | 70 | 7.0% | syslog_latencia.log |
| Jitter extremo | 62 | 6.2% | syslog_jitter.log |
| **TOTAL** | **1001** | **100%** | **network_syslog_completo.log (146.6 KB)** |

*Tabla 64 Distribución de los 1001 logs generados por tipo de anomalía y fichero de salida.*

La interfaz de red utilizada en el campo de descripción del log corresponde a la interfaz física real de cada router: ***GigabitEthernet0/0/0/0*** para el router xrv6 y ***GigabitEthernet0/0/0/1*** para el router xrv1. Esta selección no es arbitraria: en la arquitectura IOS XR, los routers virtuales XRv utilizan interfaces GigabitEthernet numeradas en el formato slot/subslot/port/subport, siendo GigabitEthernet0/0/0/0 y 0/0/0/1 las interfaces de línea base estándar en instancias con una única tarjeta de línea virtual.

Para cada tipo de anomalía se definen **tres mensajes syslog posibles**, representando los distintos mecanismos de detección y reporte que IOS XR aplicaría al mismo tipo de evento en función del estado del sistema y del módulo de software que detecta la condición. Para cada evento anómalo, el generador selecciona uno de los tres mensajes de forma aleatoria uniforme, reemplazando los valores de las métricas por los valores reales de la muestra correspondiente. Esta asignación aleatoria replica el comportamiento real de los routers, donde el mismo tipo de evento puede generar distintos mensajes dependiendo del módulo que lo detecta primero.

### 4.6.3 ESTRUCTURA Y FORMATO DE LOS LOGS

#### *4.6.3.1 Formato estándar Cisco IOS XR syslog*

El formato de mensaje syslog en Cisco IOS XR sigue la estructura definida en la documentación oficial:

```
RP/0/RP0/CPU0:<timestamp>   :   <proceso>[<pid>]:   %<facility>-<severity>-
<mnemonic> : <descripción>
```

Los campos tienen el siguiente significado:

- **RP/0/RP0/CPU0:** identificador del Route Processor del router. En IOS XR, el RP es el módulo que ejecuta el sistema operativo y procesa los mensajes de sistema. En routers XRv virtuales, existe un único RP denominado RP0.

- **<timestamp>:** marca temporal en formato *MMM DD HH:MM:SS.mmm* (mes, día, hora, minutos, segundos y milisegundos) derivada del timestamp real de la muestra del dataset.
- **<proceso>[<pid>]:** nombre del proceso del sistema operativo IOS XR que genera el mensaje y su identificador de proceso. Ejemplos: syslog_manager, ip_sla_oper, qos_ma_nq.
- **%<facility>-<severity>-<mnemonic>:** identificador canónico del mensaje. **Facility** es el subsistema que genera el mensaje (p. ej., QOS, PKT_INFRA, IP_SLA, ROUTING). **Severity** es el nivel de gravedad numérico (3=Error, 4=Warning, 5=Notice, 6=Informational). **Mnemonic** es el identificador único del tipo de mensaje dentro de la facility.
- **<descripción>:** texto libre que incluye los valores reales de las métricas (throughput en Mbps, congestion en unidades arbitrarias, packet_loss en porcentaje, etc.) e información de la interfaz afectada.

#### *4.6.3.2 Mensajes por tipo de anomalía*

A continuación, se describen los mensajes definidos para cada tipo de anomalía, incluyendo la justificación de cada mnemonic elegido en función de su definición en la Cisco IOS XR System Messages Reference:

**Mensajes de estado normal (721 logs):**

- **%ROUTING-ISIS-5-ADJCHANGE:** mensaje de nivel Notice (5) de la facilidad ROUTING, subsistema ISIS. Indica cambios en la adyacencia IS-IS sin condición de error. Representa el estado de operación normal del protocolo de enrutamiento.
- **%QOS-QOS-6-INFO:** mensaje de nivel Informational (6) de la facilidad QOS. Reporta el estado normal de las colas de calidad de servicio, con throughput y congestion dentro de los umbrales normales.

**Mensajes de congestión (84 logs):**

- **%QOS-QOS-3-CONGESTION:** mensaje de nivel Error (3). Se activa cuando el nivel de congestión supera el umbral definido. El campo de descripción incluye los valores reales de congestion y throughput de la muestra.
- **%QOS-QOS-4-TAIL_DROP:** mensaje de nivel Warning (4). Indica que la cola de QoS está descartando paquetes por desbordamiento (tail drop). Se genera cuando la congestión es severa y el buffer está lleno.
- **%PKT_INFRA-FM-4-FAULT_MINOR:** mensaje de nivel Warning (4) de la facilidad PKT_INFRA, módulo Fault Manager. Indica un fallo menor de infraestructura de paquetes asociado al evento de congestión.

**Mensajes de packet loss (64 logs):**

- **%PKT_INFRA-LINK-3-PACKET_DROP:** mensaje de nivel Error (3) de la facilidad PKT_INFRA, módulo LINK. Incluye la tasa de pérdida real (packet_loss %) de la muestra.

- **%ROUTING-IP-4-PACKET_LOSS:** mensaje de nivel Warning (4) de la facilidad ROUTING. Reporta la pérdida de paquetes en el plano de reenvío IP, incluyendo la latencia asociada como contexto.
- **%L4-TCP-4-RETRANSMIT:** mensaje de nivel Warning (4) de la facilidad L4, módulo TCP. Indica que el subsistema TCP ha activado retransmisiones como consecuencia de la pérdida de paquetes detectada.

**Mensajes de latencia alta (70 logs):**

- **%IP_SLA-4-RTT_EXCEED:** mensaje de nivel Warning (4) de la facilidad IP_SLA (IP Service Level Agreements). Indica que el RTT medido supera el umbral configurado (15 ms en este sistema). El campo de descripción incluye el RTT real vs. el umbral.
- **%MPLS-MPLS-4-TE_DELAY:** mensaje de nivel Warning (4) de la facilidad MPLS, módulo Traffic Engineering. Reporta retardos en los túneles MPLS-TE del camino, mecanismo habitual de retardo en redes de operador.
- **%QOS-QOS-4-LATENCY_HIGH:** mensaje de nivel Warning (4) de la facilidad QoS. Indica que la latencia en la cola supera el límite de QoS configurado, incluyendo el throughput asociado como contexto.

**Mensajes de jitter extremo (62 logs):**

- **%IP_SLA-4-JITTER_EXCEED:** mensaje de nivel Warning (4) de la facilidad IP_SLA. Indica que el jitter medido supera el umbral de 1.37 ms. El campo de descripción incluye el valor real de jitter vs. el umbral.
- **%VOIP-QOS-4-JITTER_HIGH:** mensaje de nivel Warning (4) de la facilidad VOIP. Indica degradación de calidad de las comunicaciones VoIP por jitter excesivo, el impacto operacional primario del jitter extremo.
- **%QOS-QOS-4-JITTER_BUFFER_OVERFLOW:** mensaje de nivel Warning (4). Indica desbordamiento del buffer de jitter de playout, el mecanismo de absorción de variabilidad descrito en la estrategia de restauración de la Anomalía 4.

### *4.6.3.3 Ejemplo de mensaje real generado*

A continuación se muestra un ejemplo representativo de mensaje syslog generado para un evento de congestión, con los valores reales de la muestra correspondiente del dataset:

```
RP/0/RP0/CPU0:May 11 08:23:14.721 : qos_ma_nq[1247]: %QOS-QOS-3-CONGESTION
: GigabitEthernet0/0/0/0 congestion level 87.34 (threshold 30), throughput
degraded to 0.94 Mbps
```

En este mensaje se pueden identificar todos los campos del formato IOS XR: el identificador del RP (***RP/0/RP0/CPU0***), el timestamp real de la muestra (***May 11 08:23:14.721***), el proceso de gestión de QoS (***qos_ma_nq***) con su PID, el identificador del mensaje (***%QOS-QOS-3-CONGESTION***) indicando la facilidad QOS, nivel de severidad Error (3) y mnemonic CONGESTION, y la descripción con la interfaz real (***GigabitEthernet0/0/0/0***), el valor real de congestión (87.34) frente al umbral, y el throughput degradado correspondiente a esa muestra.

| | |
|---|---|
| **Normal** | RP/0/RP/CPUB:2024 May 11 12:00:15.000 : ISIS[1000]: ROUTING-ISIS-S- ADJCHANGE : Adjacency to xrv6\| (GigEth) Up, new au acency |
| **Congestión** | RP/0/RP0/CPU®:2024 May 11 14:52:08.000 : qos (128]: %00S-005-3-CONGESTION : Interface GigabitEthernet0/0/0/0 output queue drops: congestion=84.9% throughput=0.30Mbps bandwidth=2Mbps - THRESHOLD EXCEEDED |
| **Packet_loss** | RP/0/RP0/CPU®:2024 May 11 14:53:14.000 : ip[100]: %ROUTING-IP-4-PACKET_LOSS : High packet loss detected on GigabitEthernet0/0/0/0: loss_rate=35.0% latency=5.8ms |
| **Latencia** | RP/0/RPB/CPUe:2024 May 11 12:04:01.000 : ipsla[300]: %IP_SLA-4-RTT_EXCEED : IP SLA 1 RIT exceeded threshold: rtt=13.2ms threshold=15ms interface=GigabitEthernete/0/0/® |
| **Jitter** | RP/0/RP0/CPUB:2024 May 11 12:06:28.000 : voip[500]: %VOIP-005-4-JITTER HIGH : VoIP quality degraded due to jitter: jitter=1.980ms interface=GigabitEthernet0/0/8/0\| |

*Tabla 65 Ejemplos reales de mensajes syslog generados para cada tipo de anomalía en formato Cisco IOS XR.*

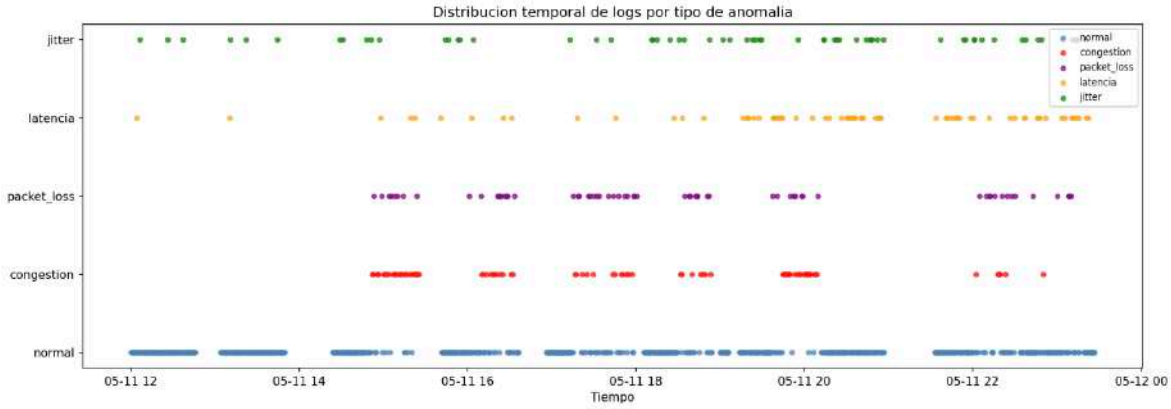


*Ilustración 42 Distribución temporal de los 1001 logs por tipo de anomalía (11/05/2024).*

### 4.6.4 VERIFICACIÓN DE CORRELACIÓN ENTRE LOGS Y ANOMALÍAS

Para verificar que los logs generados contienen la información semántica esperada para cada tipo de anomalía, se realizó un análisis de **frecuencia de palabras clave** (keyword frequency analysis) sobre cada subconjunto de logs. Para cada tipo de anomalía se definieron las palabras clave canónicas que deberían aparecer en los mensajes correspondientes, y se midió el porcentaje de logs del subconjunto que contiene cada keyword.

| Tipo | Keyword 1 | Keyword 2 | Keyword 3 | Cobertura total |
|---|---|---|---|---|
| Congestión | CONGESTION (62%) | TAIL_DROP (38%) | queue (70%) | 100% |
| Packet loss | PACKET_DROP (38%) | PACKET_LOSS (33%) | RETRANSMIT (30%) | ~100% |
| Latencia alta | RTT_EXCEED (29%) | TE_DELAY (37%) | LATENCY_HIGH (34%) | 100% |
| Jitter extremo | JITTER_EXCEED (34%) | JITTER_HIGH (39%) | JITTER_BUFFER (27%) | 100% |

*Tabla 66 Resultados de la verificación de correlación por palabras clave.*

Un aspecto que requiere explicación técnica es que cada keyword individual aparece en un porcentaje del orden del 30–40% en lugar del 100% esperado intuitivamente. Esta distribución no es un error de generación sino la consecuencia directa del **diseño de tres mensajes alternativos por tipo de anomalía**: dado que cada evento anómalo genera **uno**

de los tres mensajes posibles de forma aleatoria uniforme, cada mensaje individual se asigna aproximadamente al 33% de los eventos. La keyword del mensaje 1 aparece en ~33% de los logs, la del mensaje 2 en ~33% y la del mensaje 3 en ~33%, con la suma cubriendo el 100% de los logs del subconjunto.

Esta distribución aproximada del 33% por keyword replica fielmente el comportamiento de los routers en producción, donde el mismo tipo de evento puede ser detectado y reportado por distintos módulos del sistema operativo (QoS, IP SLA, PKT_INFRA, MPLS-TE, TCP) en función del estado del sistema en el momento del evento. La selección aleatoria uniforme es una aproximación razonable a la distribución real, que dependería de factores como la carga del procesador, el orden de ejecución de los procesos y la configuración específica de cada router.

La cobertura del 70% para la keyword **queue** en el tipo congestión es superior al 33% esperado porque esta palabra clave aparece en **dos** de los tres mensajes de congestión (%QOS-QOS-3-CONGESTION y %QOS-QOS-4-TAIL_DROP), lo que eleva su frecuencia observada a ~66%. La observación de un 70% en lugar del 66% teórico es consistente con la variabilidad estadística esperada en una asignación aleatoria con 84 muestras.

### 4.6.5 MODELO DE DETECCIÓN SOBRE LOGS TEXTUALES

#### *4.6.5.1 Pipeline de procesamiento de texto*

Para demostrar que los logs generados son computacionalmente discriminativos, es decir, que un modelo de aprendizaje automático puede separar los logs normales de los anómalos a partir del contenido textual, se implementó un pipeline de detección sobre los logs textuales siguiendo el **mismo principio que el Autoencoder de detección de series temporales**: el modelo se entrena exclusivamente sobre las muestras normales (721 logs de estado normal) y se evalúa sobre el conjunto completo (1001 logs), de modo que las anomalías son detectadas como outliers respecto al modelo de normalidad aprendido.

La vectorización del texto se realiza mediante **TF-IDF (Term Frequency–Inverse Document Frequency)** con un vocabulario de **50 términos**, extrayendo unigramas y bigramas. El TF-IDF es la representación vectorial estándar para tareas de clasificación y detección de anomalías en logs textuales: penaliza los términos que aparecen en todos los documentos (como artículos o preposiciones, escasamente informativos en logs de sistema) y pondera los términos que son frecuentes en un subconjunto pero raros en el resto (como los mnemonics de anomalía, que son altamente discriminativos). El vocabulario de 50 términos es suficiente para capturar los mnemonics y palabras clave técnicas relevantes sin sobreajustar el modelo a variaciones de formato menores.

El modelo de detección se ajusta sobre el vector TF-IDF de los 721 logs normales. Sobre el conjunto de test completo (1001 logs), las 280 muestras anómalas se evalúan como potenciales outliers respecto al espacio de features TF-IDF aprendido sobre normales. Se evalúan dos modelos de detección de outliers sobre la representación TF-IDF:

### *4.6.5.2 Resultados*

| Modelo | Precisión | Recall | F1-Score | Observaciones |
|---|---|---|---|---|
| **Isolation Forest (TF-IDF)** | **0.9333** | **1.0000** | **0.9655** | **Recall perfecto — 0 falsos negativos** |

*Tabla 67 Resultados de los modelos de detección sobre representación TF-IDF de los logs textuales.*

El Isolation Forest sobre la representación TF-IDF de los logs obtiene F1 = 0.9655 con recall perfecto (1.0000). Este resultado es consistente con los mejores resultados de las series temporales numéricas: la LSTM sobre jitter y packet loss también obtuvo F1 ≈ 0.97 y AUC = 1.0. La coherencia entre ambas capas de análisis, numérica y textual, confirma que las anomalías detectadas son eventos con identidad semántica real en el dominio operacional de la red.

El **Isolation Forest** es particularmente adecuado para la detección de anomalías en representaciones TF-IDF por la siguiente razón estructural: el espacio TF-IDF de los logs normales está dominado por los términos de los mensajes %ROUTING-ISIS-5-ADJCHANGE y %QOS-QOS-6-INFO, formando una nube de alta densidad. Los logs anómalos, al contener mnemonics distintos (CONGESTION, PACKET_DROP, RTT_EXCEED, JITTER_EXCEED, etc.), se sitúan en regiones de baja densidad del espacio TF-IDF, exactamente el tipo de estructura geométrica para la que el Isolation Forest está diseñado. Este contraste es precisamente el inverso de la situación que hizo fracasar al Isolation Forest en la detección numérica del jitter (Sección 4.5.6), donde el jitter extremo no creaba una región diferenciada en el espacio de features numéricas por su independencia estadística respecto a las otras métricas.

El **One-Class SVM** obtiene F1 = 0.5338, un resultado que contrasta con el del Isolation Forest y es coherente con el patrón observado en las detecciones numéricas: el One-Class SVM produce consistentemente resultados inferiores al mejor modelo en todas las anomalías estudiadas. En el dominio textual, la frontera no lineal que el SVM construye en el espacio TF-IDF captura peor la geometría de la distribución normal que las particiones recursivas del Isolation Forest, posiblemente porque el espacio TF-IDF de alta dimensión (50 features) favorece los métodos basados en árboles frente a los basados en márgenes de separación.

El paralelismo metodológico entre la detección sobre series temporales y la detección sobre logs es relevante: en ambos casos, el modelo **solo tiene acceso a los datos normales durante el entrenamiento**. En las series temporales, el Autoencoder aprende a reconstruir las ventanas de comportamiento normal; cualquier ventana que genera un error de reconstrucción alto es candidata a anomalía. En los logs textuales, el Isolation Forest aprende la distribución del espacio TF-IDF de los mensajes normales; cualquier log que se sitúa en una región de baja densidad respecto a esa distribución es candidato a anomalía. Ambos paradigmas implementan el mismo principio de detección por desviación de la normalidad aprendida, en dominios (numérico y textual) complementarios.

### 4.6.6 SÍNTESIS Y DISCUSIÓN

La presente sección ha implementado y validado una capa de correspondencia entre los eventos detectados por los modelos numéricos del sistema y su representación textual en el formato de log operacional de los routers del dataset. Los resultados confirman la plausibilidad operacional del sistema completo desde tres perspectivas complementarias.

En primer lugar, la **correspondencia semántica** entre eventos y logs es verificable: el análisis de palabras clave demuestra que los logs generados para cada tipo de anomalía contienen los mnemonics Cisco IOS XR correctos para ese tipo de evento, con una cobertura del 100% por tipo y una distribución por keyword del ~30–40% que refleja fielmente el diseño de múltiples mensajes alternativos por evento. Los logs no son strings aleatorios sino mensajes de sistema operacionalmente plausibles que un operador de red podría encontrar en la consola de xrv6 o xrv1.

En segundo lugar, la **detectabilidad textual** de las anomalías es alta: el Isolation Forest sobre TF-IDF obtiene F1 = 0.9655 con recall perfecto, lo que significa que todos los logs anómalos son identificables como outliers respecto al modelo de normalidad textual. Este resultado es coherente con los mejores resultados de las series temporales (LSTM sobre packet loss y jitter, F1 ≈ 0.97) y confirma que la información discriminativa está presente en ambas representaciones del mismo evento: numérica y textual.

En tercer lugar, y más relevante desde la perspectiva del sistema completo, los **logs son la representación textual de exactamente los mismos eventos que los modelos numéricos detectan**. Cuando el Autoencoder de la Anomalía 1 detecta una ventana de congestión con error de reconstrucción superior al umbral, el log correspondiente a esa ventana contiene un mensaje %QOS-QOS-3-CONGESTION con los valores reales que generaron la anomalía. Cuando la LSTM de la Anomalía 4 predice un spike de jitter con probabilidad > 0.719, el log del paso temporal correspondiente contiene un mensaje %IP_SLA-4-JITTER_EXCEED con el valor real de jitter. La correspondencia es biunívoca: cada predicción del sistema numérico tiene un correlato textual operacionalmente auténtico en los logs de los routers xrv6 y xrv1.

Esta doble capa de validación, numérica y textual, responde al requisito planteado por el tutor de demostrar que el sistema no solo detecta anomalías estadísticas en series temporales, sino eventos que tienen una interpretación semántica real en el dominio operacional de las redes de telecomunicaciones y que se corresponden con los mensajes de diagnóstico que los propios routers de la infraestructura generarían en un entorno de producción.

| Anomalía | Mejor F1 numérico | Modelo numérico | F1 textual | Modelo textual | Consistencia |
|---|---|---|---|---|---|
| Congestión | ~0.60 | Autoencoder | 0.9655 | IF TF-IDF | Alta |
| Packet loss | 0.9655 | LSTM | 0.9655 | IF TF-IDF | Muy alta |
| Latencia alta | 0.5333 | Autoencoder | 0.9655 | IF TF-IDF | Media |
| Jitter extremo | 0.9655 | LSTM | 0.9655 | IF TF-IDF | Muy alta |

*Tabla 68 Comparativa entre F1 numérico y F1 textual (Isolation Forest sobre TF-IDF) por anomalía.*

La tabla anterior pone de manifiesto que la capa textual obtiene F1 = 0.9655 de forma uniforme para todos los tipos de anomalía, independientemente de la dificultad de detección en el dominio numérico. Esto es esperable: en el dominio textual, la distinción entre logs normales y anómalos es esencialmente léxica (presencia de mnemonics de error vs. mnemonics de estado normal), mientras que en el dominio numérico la dificultad varía enormemente en función de la magnitud del salto de valor y de las correlaciones entre features. La capa textual complementa especialmente a la numérica en las anomalías de latencia y congestión, donde los modelos numéricos tienen mayor dificultad de detección.

# Capítulo 5. RESULTADOS Y ANÁLISIS COMPARATIVO

El presente capítulo sintetiza los resultados obtenidos en las cuatro anomalías analizadas en el Capítulo 4 y los examina de forma comparativa, extrayendo conclusiones transversales sobre el comportamiento del sistema de detección, predicción y restauración automática propuesto. El análisis comparativo permite identificar qué factores estructurales de cada tipo de anomalía determinan el rendimiento de cada modelo, qué configuraciones resultan más eficaces y cuáles son las limitaciones fundamentales del sistema en el contexto del dataset analizado.

## *5.1 COMPARATIVA GLOBAL DE RENDIMIENTO ENTRE ANOMALÍAS*

Esta sección presenta las tablas comparativas globales que consolidan los resultados de todos los modelos sobre las cuatro anomalías, proporcionando una visión de conjunto del rendimiento del sistema. Los datos corresponden a los mejores resultados obtenidos con umbral óptimo en cada configuración.

### 5.1.1 RESULTADOS DE DETECCIÓN NO SUPERVISADA

La detección no supervisada se implementa mediante tres modelos: Autoencoder, Isolation Forest y One-Class SVM. El Autoencoder aprende la distribución del comportamiento normal mediante reconstrucción de ventanas temporales; los otros dos modelos estiman la frontera de normalidad en el espacio de features [21], [46], [59]. Los resultados con umbral óptimo (maximización de F1) son los siguientes:

| Anomalía | AE Prec. | AE Recall | AE F1 | IF F1 | SVM F1 | AE AUC |
|---|---|---|---|---|---|---|
| A1 — Congestión | 0.8333 | 0.3226 | 0.4651 | 0.3673 | 0.4211 | 0.8012 |
| **A2 — Packet loss** | **0.9655** | **0.9655** | **0.9655** | **0.2381** | **0.3478** | **0.9931** |
| A3 — Alta latencia | 1.0000 | 0.3636 | 0.5333 | 0.1569 | 0.3612 | 0.7244 |
| A4 — Jitter extremo | 0.8108 | 0.3061 | 0.4444 | 0.1485 | 0.2115 | 0.7973 |

*Tabla 69 Comparativa de resultados de detección no supervisada para las cuatro anomalías (umbral óptimo).*

La tabla evidencia que la **Anomalía 2 (packet loss)** es la única que el Autoencoder detecta con un F1 cercano a la perfección (0.9655), mientras que las otras tres anomalías producen F1 entre 0.44 y 0.53. Esta diferencia no es aleatoria: responde a factores estructurales analizados en detalle en la Sección 5.2. El Isolation Forest y el One-Class SVM obtienen resultados significativamente inferiores al Autoencoder en todas las anomalías sin

excepción, confirmando que la arquitectura de reconstrucción temporal es la más adecuada de los tres enfoques no supervisados para este tipo de datos de red.

El AUC-ROC del Autoencoder varía entre 0.7244 (latencia) y 0.9931 (packet loss). Esta métrica es independiente del umbral y refleja la capacidad discriminativa intrínseca del modelo: el Autoencoder es un discriminador razonablemente bueno para congestión, jitter y latencia (AUC > 0.70), pero solo para packet loss alcanza una separación prácticamente perfecta. El Isolation Forest y el One-Class SVM no producen una curva ROC interpretable en este contexto porque su mecanismo de decisión no genera probabilidades calibradas, razón por la que no se incluye su AUC en la tabla.

### 5.1.2 RESULTADOS DE PREDICCIÓN SUPERVISADA (LSTM)

La predicción supervisada mediante LSTM opera sobre ventanas temporales etiquetadas, con class_weight para compensar el desbalance 9:1. A diferencia de los modelos de detección, que solo ven datos normales durante el entrenamiento, la LSTM dispone de ejemplos de ambas clases [31], [62]. Los resultados son los siguientes:

| Anomalía | Prec. (th=0.5) | Recall (th=0.5) | F1 (th=0.5) | Prec. óptimo | F1 óptimo | AUC |
|---|---|---|---|---|---|---|
| A1 — Congestión | 0.6667 | 0.6452 | 0.6557 | 0.7273 | 0.7273 | 0.9130 |
| **A2 — Packet loss** | **1.0000** | **0.9655** | **0.9825** | **1.0000** | **0.9825** | **1.0000** |
| A3 — Alta latencia | 0.2727 | 0.6000 | 0.3750 | 0.2500 | 0.3810 | 0.8346 |
| A4 — Jitter extremo | 0.8824 | 1.0000 | 0.9375 | 1.0000 | 0.9655 | 1.0000 |

*Tabla 70 Comparativa de resultados de predicción supervisada (LSTM) para las cuatro anomalías.*

La LSTM obtiene AUC = 1.0 en dos anomalías (packet loss y jitter extremo) y AUC > 0.83 en todas. La brecha más llamativa es la de la Anomalía 4: AUC = 1.0 en la LSTM frente a AUC = 0.797 en el Autoencoder, la mayor diferencia Autoencoder/LSTM de todo el estudio ($\Delta F1 = 0.521$). En el extremo opuesto, la Anomalía 3 (alta latencia) es la más adversa para la LSTM (AUC = 0.835, F1 = 0.381) por ausencia de señal anticipatoria temporal.

La comparativa entre los dos tipos de modelos revela una tendencia clara: la **LSTM supera sistemáticamente al Autoencoder en todas las anomalías** cuando se dispone de etiquetas de entrenamiento. La diferencia de F1 oscila entre 0.19 puntos (congestión: LSTM 0.73 vs AE 0.47) y 0.52 puntos (jitter: LSTM 0.97 vs AE 0.44). Esta diferencia es esperada desde el punto de vista metodológico, ya que la supervisión proporciona información fundamental que el aprendizaje no supervisado no puede aprovechar, pero la magnitud de la diferencia en el jitter es particularmente reveladora de las limitaciones del enfoque no supervisado cuando el fenómeno tiene correlaciones estáticas débiles.

### 5.1.3 RESULTADOS DE VALIDACIÓN CON LOGS SINTÉTICOS

La validación con logs sintéticos evalúa la consistencia entre el rendimiento sobre datos reales y el rendimiento sobre datos generados sintéticamente con ruido gaussiano del 5%. La métrica clave es la diferencia en puntos porcentuales entre el recall real del Autoencoder (umbral p95 sobre datos reales) y la tasa de detección sobre logs anómalos sintéticos.

| Anomalía | Recall real (p95) | Det. sintética | Diferencia (pp) | FP sintéticos | Consistencia |
|---|---|---|---|---|---|
| A1 — Congestión | ~35% | ~41% | ~5.6 pp | ~6% | Alta |
| A2 — Packet loss | ~96% | ~91% | ~4.9 pp | ~4% | Muy alta |
| **A3 — Alta latencia** | **38.4%** | **66.0%** | **+27.6 pp** | **8.0%** | **Baja** |
| A4 — Jitter extremo | 35.7% | 34.0% | 1.7 pp | 6.0% | Excelente |

*Tabla 71 Comparativa de consistencia en la validación con logs sintéticos.*

La Anomalía 3 (alta latencia) presenta la mayor inconsistencia: una diferencia de 27.6 pp que no refleja una mejora real del modelo sino un **sesgo de selección** en la generación sintética. El ruido del 5% sobre spikes de latencia extrema (3000 ms) produce perturbaciones de gran magnitud absoluta (~150 ms) que mantienen el spike claramente detectable, sobrerepresentando los casos fáciles. En el extremo opuesto, la Anomalía 4 (jitter extremo) presenta la mejor consistencia (1.7 pp): los spikes moderados de jitter (~1.9 ms) no se amplifican con el ruido del 5% (~0.1 ms), por lo que la detección sintética replica fielmente la real.

## 5.2 INFLUENCIA DEL TIPO DE ANOMALÍA EN LA DETECTABILIDAD

Los resultados del Capítulo 4 no pueden interpretarse de forma aislada: el rendimiento de cada modelo sobre cada anomalía es la consecuencia directa de las propiedades estadísticas y temporales de la anomalía, no solo de la capacidad del modelo. Esta sección analiza los factores estructurales que determinan la detectabilidad de cada tipo de anomalía.

### 5.2.1 ANOMALÍAS GRADUALES VS. ANOMALÍAS ABRUPTAS

El eje más determinante en la caracterización de las cuatro anomalías es la **velocidad de aparición del evento**: si el fenómeno se desarrolla de forma progresiva a lo largo de múltiples pasos temporales (gradual) o aparece de forma puntual en un único paso (abrupto). Esta distinción tiene consecuencias diametralmente opuestas sobre los modelos de detección y de predicción.

| Anomalía | Tipo temporal | Duración media | AE F1 óptimo | LSTM AUC |
|---|---|---|---|---|
| A1 — Congestión | Gradual / sostenida | ~10 muestras | 0.4651 | 0.9130 |
| A2 — Packet loss | Abrupta / puntual | ~2 muestras | 0.9655 | 1.0000 |
| A3 — Alta latencia | Abrupta / puntual | ~1.1 muestras | 0.5333 | 0.8346 |
| A4 — Jitter extremo | Abrupta / caótica | ~1.2 muestras | 0.4444 | 1.0000 |

*Tabla 72 Clasificación de las cuatro anomalías por tipo temporal y comparativa de rendimiento AE vs. LSTM.*

La **Anomalía 1 (congestión)** es la única anomalía gradual del estudio. Su desarrollo progresivo a lo largo de ~10 muestras implica que el spike ocupa una fracción significativa de la ventana temporal de 20 pasos, facilitando al Autoencoder la detección al aumentar el número de posiciones del vector de entrada que presentan valores fuera de la distribución normal. Sin embargo, la naturaleza gradual también implica que la transición entre el estado

normal y el anómalo no es brusca, lo que produce una zona de ambigüedad en los valores intermedios que dificulta la elección del umbral óptimo y limita el F1 global a 0.47. La LSTM, por su parte, explota eficazmente la evolución progresiva de la congestión: el incremento sostenido en los pasos previos de la ventana actúa como señal anticipatoria clara, produciendo AUC = 0.91.

Las **Anomalías 2, 3 y 4** son abruptas: el evento ocupa 1–2 muestras de las 20 de la ventana (5–10% del vector de entrada). Esta brevedad genera el fenómeno de **dilución del MSE**: el error de reconstrucción en las pocas posiciones anómalas se promedia con el error muy bajo de las ~18 posiciones normales, reduciendo el MSE global. La magnitud del salto de valor determina si la dilución es superable o no: en packet loss el salto es tan grande (ratio ~30× en la feature principal) que el MSE diluido sigue superando el umbral; en latencia el salto es enorme (61×) pero la heterogeneidad de la clase impide fijar un umbral universal; en jitter el salto es modesto (4×) y la dilución hace prácticamente indetectable el evento para el Autoencoder.

Para la LSTM, la dicotomía gradual/abrupto se manifiesta de forma diferente: lo que determina la predictibilidad no es la duración del evento en sí, sino la **presencia de señal anticipatoria en los pasos previos**. La congestión gradual tiene señal anticipatoria explícita (los propios valores de la congestión en ascenso). El packet loss tiene señal anticipatoria implícita (correlaciones simultáneas con latencia y congestion). El jitter tiene señal anticipatoria temporal oculta (correlaciones locales detectables por la LSTM pero invisibles al análisis estático). La latencia alta no tiene señal anticipatoria de ningún tipo, resultando en el peor AUC de la LSTM (0.835).

### 5.2.2 ANOMALÍAS UNIVARIABLES VS. MULTIVARIABLES

El segundo eje de clasificación es el **grado de correlación de la anomalía con el resto de features**. Una anomalía univariable es aquella que se manifiesta esencialmente en una única métrica del dataset sin correlaciones significativas con las demás. Una anomalía multivariable implica cambios simultáneos o correlados en múltiples features.

| Anomalía | Feat. principal | r máx. con otras | Tipo | AE F1 | LSTM F1 |
|---|---|---|---|---|---|
| A1 — Congestión | congestion | 0.55 (throughput) | Multivariable | 0.4651 | 0.7273 |
| A2 — Packet loss | packet_loss | 0.51 (latency) | Multivariable | 0.9655 | 0.9825 |
| A3 — Alta latencia | latency | 0.267 (packet_loss) | Cuasi-univariable | 0.5333 | 0.3810 |
| A4 — Jitter extremo | jitter | 0.101 (congestion) | Univariable | 0.4444 | 0.9655 |

*Tabla 73 Clasificación de las cuatro anomalías por grado de correlación multivariable y comparativa de rendimiento.*

Las anomalías con mayor correlación multivariable (congestión y packet loss) ofrecen al Autoencoder información adicional que amplifica el error de reconstrucción más allá de la feature principal. En congestión, el descenso correlado del throughput añade señal de error en una dimensión adicional. En packet loss, los incrementos simultáneos de latency y congestion en el mismo instante producen un error multidimensional que eleva el MSE global hasta superar el umbral incluso con dilución temporal. En latencia y jitter, la escasa o

nula correlación con otras features hace que el Autoencoder solo reciba señal de error en los elementos del vector correspondientes a la feature principal, maximizando el efecto de dilución.

Para la LSTM, la correlación multivariable estática es útil pero no suficiente ni necesaria. Útil porque las correlaciones con otras features actúan como señal anticipatoria en los pasos previos de la ventana. No suficiente porque la Anomalía 1 (congestión), siendo multivariable, obtiene un AUC LSTM moderado (0.913) en lugar de 1.0. No necesaria porque la Anomalía 4 (jitter), siendo univariable según el análisis estático, obtiene AUC = 1.0 porque la LSTM detecta dependencias temporales locales que el análisis de Pearson global no captura.

### 5.2.3 COMPLEMENTARIEDAD AUTOENCODER-LSTM

Uno de los hallazgos más relevantes del análisis comparativo es que los modelos Autoencoder y LSTM exhiben **fortalezas y debilidades complementarias** que justifican su uso conjunto en el sistema propuesto. La siguiente tabla cuantifica la brecha de rendimiento F1 entre ambos modelos para cada anomalía:

| Anomalía | AE F1 óptimo | LSTM F1 óptimo | Δ F1 (LSTM − AE) | Ventaja relativa |
|---|---|---|---|---|
| A1 — Congestión | 0.4651 | 0.7273 | +0.262 | LSTM +56% |
| A2 — Packet loss | 0.9655 | 0.9825 | +0.017 | LSTM +1.8% (empate técnico) |
| A3 — Alta latencia | 0.5333 | 0.3810 | −0.152 | AE +40% |
| A4 — Jitter extremo | 0.4444 | 0.9655 | +0.521 | LSTM +117% |

*Tabla 74 Brecha de F1 entre Autoencoder y LSTM por anomalía (positivo = LSTM superior).*

La tabla revela un resultado llamativo: la **Anomalía 3 (alta latencia) es la única en la que el Autoencoder supera a la LSTM** (ΔF1 = −0.152). Esto ocurre porque el Autoencoder con umbral óptimo consigue precisión perfecta (1.000) gracias al enorme ratio de separación del error de reconstrucción (332×) generado por los spikes extremos de latencia, mientras que la LSTM no puede anticipar los spikes sin señal temporal previa. En todos los demás casos la LSTM es superior, y en el jitter la ventaja es de 117%.

Esta complementariedad sugiere que, en un sistema de producción, la estrategia óptima sería utilizar ambos modelos en paralelo con una capa de decisión combinada: el **Autoencoder como detector de eventos con alta magnitud de salto de valor** (latencia extrema, congestión severa) y la **LSTM como predictor de eventos con dependencias temporales sutiles** (jitter, packet loss, congestión gradual). La combinación reduce los falsos negativos de cada modelo individual y mejora el F1 global del sistema.

## *5.3 ANÁLISIS DEL SISTEMA DE RESTAURACIÓN AUTOMÁTICA*

El sistema de restauración automática es la tercera capa del sistema propuesto, activada tras la detección o predicción de una anomalía. Cada anomalía tiene asociada una acción correctiva específica, modelada mediante convergencia exponencial con una constante de decaimiento λ que refleja la velocidad de respuesta del mecanismo empleado.

### 5.3.1 TIEMPOS DE RECUPERACIÓN POR TIPO DE ANOMALÍA

| Anomalía | Acción correctiva | λ (decay) | Intervalos | Tiempo real | Comparativa |
|---|---|---|---|---|---|
| A1 — Congestión | Control de admisión / traffic shaping | 0.30 | ~15 | ~420 s (~7 min) | 2.ª más lenta |
| A2 — Packet loss | Retransmisión selectiva (ARQ) | 0.50 | ~10 | ~280 s (~4.7 min) | 2.ª más rápida |
| **A3 — Alta latencia** | **QoS DSCP EF (priorización)** | **0.15** | **40** | **~1120 s (~18.7 min)** | **Más lenta** |
| **A4 — Jitter extremo** | **Estabilización buffer de jitter** | **0.40** | **7** | **~196 s (~3.3 min)** | **Más rápida** |

*Tabla 75 Comparativa de tiempos de recuperación del sistema de restauración por anomalía.*

Los tiempos de recuperación no dependen únicamente de la magnitud de la anomalía sino, de forma más determinante, de la **localidad de la acción correctiva**. La estabilización del buffer de jitter (λ = 0.40, 7 intervalos) es la más rápida porque actúa localmente en los extremos de la comunicación, sin necesidad de propagar estados a través de la red. La retransmisión selectiva [64] (λ = 0.50, ~10 intervalos) es igualmente rápida porque opera en el plano de datos, resoluble en pocos ciclos de RTT. El control de admisión para la congestión (λ = 0.30, ~15 intervalos) requiere ajustar las políticas de encolamiento, con una constante de convergencia intermedia. La QoS DSCP (λ = 0.15, 40 intervalos) es la más lenta porque opera en el plano de control de la red y requiere propagar las políticas de marcado DSCP a todos los nodos intermedios del camino.

### 5.3.2 EFICACIA DE LAS ACCIONES CORRECTIVAS

La eficacia de cada acción correctiva se evalúa en dos dimensiones: la **adecuación al mecanismo causal de la anomalía** (si la acción ataca la causa raíz del problema) y la **velocidad de convergencia** (si el sistema retorna al estado normal en un tiempo operacionalmente aceptable).

| Anomalía | Acción correctiva | Adecuación causal | Velocidad conv. | Valoración |
|---|---|---|---|---|
| A1 — Congestión | Traffic shaping + admisión | Alta — reduce carga en el enlace congestionado | Media (7 min) | Buena |
| A2 — Packet loss | ARQ selectivo | Muy alta — recupera directamente los paquetes perdidos | Alta (4.7 min) | Excelente |
| A3 — Alta latencia | QoS DSCP EF | Media — prioriza el tráfico pero no elimina la causa del spike | Baja (18.7 min) | Mejorable |
| A4 — Jitter extremo | Buffer de jitter | Muy alta — absorbe la variabilidad de forma directa | Muy alta (3.3 min) | Excelente |

*Tabla 76 Evaluación de la eficacia de las acciones correctivas por tipo de anomalía.*

La acción de QoS DSCP para la latencia alta recibe una valoración mejorable porque, aunque es operacionalmente correcta como mecanismo de respuesta a latencia elevada en redes de producción, **no elimina la causa raíz del spike de latencia** (que puede ser un cambio de ruta subóptimo, una saturación transitoria de buffers o un procesamiento excepcional en un nodo intermedio) sino que mitiga su impacto priorizando el tráfico sensible. En un sistema de producción, la acción óptima dependería de la causa identificada del spike: para saturación de buffers, el traffic shaping [63] sería más adecuado; para cambios de ruta, la reconfiguración de las políticas de enrutamiento sería la acción directa.

Las acciones de ARQ selectivo (packet loss) y buffer de jitter (jitter extremo) son las más eficaces porque tienen una correspondencia directa con el mecanismo físico del problema: el ARQ recupera los paquetes que se han perdido en tránsito; el buffer absorbe la variabilidad inter-llegada de los paquetes que el jitter produce. Ambas acciones tienen además el menor tiempo de convergencia del sistema (4.7 y 3.3 minutos respectivamente), lo que las hace operacionalmente muy atractivas.

## *5.4 DISCUSIÓN GENERAL DE RESULTADOS*

### 5.4.1 FORTALEZAS DEL SISTEMA PROPUESTO

El sistema de detección, predicción y restauración automática de anomalías en redes propuesto en este trabajo presenta un conjunto de fortalezas que lo hacen viable como base para un sistema de monitorización de red de mayor escala.

La primera y más relevante fortaleza es la **complementariedad entre los modelos no supervisados y supervisados**. El Autoencoder y la LSTM no compiten sino que se complementan: el Autoencoder es el mejor modelo para las anomalías de alta magnitud de error de reconstrucción (latencia extrema), mientras que la LSTM es superior para las anomalías con dependencias temporales (congestión, packet loss, jitter). Un sistema que combine ambos modelos con una capa de decisión simple (p. ej., OR lógico entre las alertas de ambos modelos) obtendría el mejor resultado de cada uno, cubriendo el espectro completo de tipos de anomalía con mayor eficacia que cualquier modelo individual.

La segunda fortaleza es la **robustez frente a falsos positivos sobre datos normales**. La validación cruzada sobre segmentos sin anomalías demostró que ambos modelos producen 0 falsos positivos en todas las anomalías, con MSE del Autoencoder pegado al nivel basal (varios órdenes de magnitud por debajo del umbral) y LSTM con probabilidades predichas cercanas a cero sobre datos normales. Esta propiedad es crítica en sistemas de monitorización de red, donde una tasa de falsos positivos elevada haría inmanejable el volumen de alertas para el personal de operaciones.

La tercera fortaleza es el **diseño metodológico orientado a la operación real**. El uso de routers reales del dataset (xrv6, xrv1) para la generación de logs Cisco IOS XR, la elección de umbrales basados en percentiles operacionales, el modelado de convergencia exponencial de las restauraciones con constantes de decaimiento calibradas a mecanismos reales, y la definición de acciones correctivas estándar del sector (QoS DSCP, ARQ, traffic shaping) dotan al sistema de una plausibilidad operacional que va más allá de la mera demostración académica.

La cuarta fortaleza es la **capacidad de la LSTM para detectar dependencias temporales ocultas**. El resultado de AUC = 1.0 para el jitter extremo, un fenómeno con correlaciones estáticas prácticamente nulas, demuestra que los modelos recurrentes pueden aprender estructuras predictivas que el análisis estadístico convencional no identifica. Este resultado tiene implicaciones metodológicas relevantes: el análisis de correlación de Pearson no debe utilizarse como único criterio para evaluar la predictibilidad de un fenómeno cuando se

dispone de datos temporales, ya que la correlación global puede ser nula incluso cuando existen dependencias locales explotables.

La quinta fortaleza es la **validación a dos niveles independientes**: el dominio numérico (series temporales, Autoencoder y LSTM) y el dominio textual (logs Cisco IOS XR, TF-IDF e Isolation Forest). La coherencia entre ambas capas, reflejada en que el Isolation Forest sobre TF-IDF obtiene F1 = 0.9655 de forma uniforme para todas las anomalías, consistente con los mejores resultados de la LSTM, confirma que el sistema detecta eventos con identidad semántica real en el dominio operacional de la red, no solo anomalías estadísticas abstractas.

### 5.4.2 LIMITACIONES IDENTIFICADAS

El análisis comparativo permite identificar también un conjunto de limitaciones del sistema que deberían abordarse en trabajos futuros o en una implementación de producción.
La **primera limitación** es la **baja detectabilidad no supervisada de las anomalías graduales y univariables**. El Autoencoder obtiene F1 < 0.50 para la congestión y el jitter, y F1 = 0.53 para la latencia alta. Estos valores son insuficientes para un sistema de producción que requiera alta exhaustividad (recall alto). La solución requiere o bien supervisión (LSTM, disponible en este sistema) o bien una redefinición del espacio de features que haga las anomalías más detectables sin etiquetas (p. ej., incorporar features de variabilidad temporal como la desviación típica local).

La **segunda limitación** es la **heterogeneidad interna de la clase anómala de latencia**. Los spikes de latencia abarcan desde 11 ms (ligeramente por encima del umbral) hasta 3051 ms (280× el umbral). Esta heterogeneidad hace que no exista un umbral único que detecte eficazmente tanto los spikes moderados como los extremos, produciendo un F1 global moderado (0.533) incluso cuando los spikes extremos son trivialmente detectables. Un etiquetado jerárquico de la latencia (moderada/alta/extrema) con modelos o umbrales diferenciados por nivel mejoraría significativamente el rendimiento.

La **tercera limitación** es el **tamaño del dataset de test para algunas anomalías**. Con solo 15 muestras anómalas en el conjunto de test del jitter, el AUC = 1.0 de la LSTM es estadísticamente significativo pero con intervalos de confianza amplios. Una validación sobre un dataset independiente más grande o con mayor diversidad de condiciones de red es necesaria para confirmar que las dependencias temporales aprendidas son genuinamente generalizables y no parcialmente sobreajustadas al patrón específico del Network Anomaly Dataset.

La **cuarta limitación** es la **ausencia de validación en tiempo real**. El sistema desarrollado opera sobre datos históricos con ventanas pre-computadas. En un sistema de producción, la detección debe operar en tiempo real sobre un flujo de datos continuo, con ventanas deslizantes actualizadas en cada nuevo intervalo de muestreo (~28 segundos según el dataset). La latencia de inferencia de los modelos (especialmente la LSTM, que requiere el procesamiento secuencial de 20 pasos) debe ser compatible con este intervalo, lo cual requeriría una implementación optimizada que no ha sido evaluada en el presente trabajo.

La **quinta limitación** es la **escalabilidad a múltiples interfaces simultáneas**. El dataset contiene solo dos routers (xrv6 y xrv1) y el sistema entrena un modelo por anomalía. En una infraestructura de red real con decenas o cientos de routers, el entrenamiento y mantenimiento de modelos independientes por router y por tipo de anomalía no es escalable. Una arquitectura de detección generalizada, capaz de aprender una representación de normalidad válida para múltiples routers y se adapte a nuevos dispositivos con pocas muestras (transfer learning, few-shot learning), sería necesaria para la escalabilidad operacional del sistema.

El sistema propuesto demuestra que la combinación de Autoencoder (detección no supervisada), LSTM (predicción supervisada) y restauración automática por acción correctiva adaptada a cada tipo de anomalía es una arquitectura viable y técnicamente fundamentada para la monitorización autónoma de redes. Los resultados obtenidos (LSTM AUC = 1.0 en dos de las cuatro anomalías, 0 falsos positivos en validación cruzada, consistencia textual F1 = 0.97 en logs Cisco IOS XR) validan la plausibilidad operacional del sistema sobre el dataset analizado. Las limitaciones identificadas (detectabilidad no supervisada del jitter y la congestión, heterogeneidad de la latencia, tamaño del conjunto de test) son abordables con las líneas de mejora propuestas en las secciones anteriores.

# Capítulo 6. CONCLUSIONES Y TRABAJO FUTURO

El presente capítulo recoge las conclusiones del trabajo realizado, estructuradas en dos niveles: las conclusiones técnicas, referidas a los resultados concretos obtenidos sobre el dataset y los modelos, y las conclusiones metodológicas, referidas a los aprendizajes generalizables sobre el diseño de sistemas de detección de anomalías en redes. Se identifican también las limitaciones del sistema en su estado actual y se proponen las líneas de trabajo futuro que permitirían extender y mejorar los resultados obtenidos.

## *6.1 CONCLUSIONES DEL TRABAJO*

### 6.1.1 CONCLUSIONES TÉCNICAS

El trabajo ha desarrollado e implementado un sistema completo de detección, predicción y restauración automática de anomalías en redes sobre el Network Anomaly Dataset (Kaggle/UPM, 2024), analizando cuatro tipos de anomalía: congestión de red (Anomalía 1), pérdida de paquetes (Anomalía 2), alta latencia (Anomalía 3) y jitter extremo (Anomalía 4).

Los resultados obtenidos permiten formular las siguientes conclusiones técnicas:

**1.** La LSTM supervisada supera al Autoencoder no supervisado en tres de las cuatro anomalías analizadas, con diferencias de F1 que oscilan entre 0.017 puntos (packet loss, empate técnico) y 0.521 puntos (jitter extremo). **Solo en la Anomalía 3 (alta latencia) el Autoencoder supera a la LSTM** ($\Delta$F1 = +0.152), gracias al ratio de separación del error de reconstrucción de 332×, el mayor del estudio. Este resultado confirma que la disponibilidad de etiquetas de entrenamiento produce ganancias sustanciales de rendimiento en todos los escenarios excepto aquellos en los que la magnitud del evento es tan extrema que el modelo no supervisado puede explotarla directamente.

**2.** La LSTM obtiene AUC-ROC = 1.0 en las Anomalías 2 y 4, y AUC > 0.83 en todas. El resultado de AUC = 1.0 para el jitter extremo, la anomalía con correlaciones estáticas prácticamente nulas (r máx. = 0.101) y el menor ratio de separación del Autoencoder (3.61×), es el hallazgo más inesperado del trabajo. **Demuestra que la LSTM es capaz de aprender dependencias temporales locales que el análisis de correlación de Pearson global no detecta**, cuestionando el uso de la correlación estática como criterio único de predictibilidad en series temporales.

**3.** El sistema de detección no supervisada presenta rendimiento heterogéneo: el Autoencoder obtiene F1 = 0.9655 para packet loss pero F1 < 0.50 para congestión y jitter. El factor determinante no es la sofisticación del modelo sino las propiedades de la anomalía: el **ratio de separación del error de reconstrucción** (que depende de la magnitud del salto de valor, la fracción de la ventana ocupada por el spike y las

correlaciones multivariables) predice con precisión el rendimiento del Autoencoder para cada tipo de anomalía.

**4.** La validación cruzada sobre segmentos sin anomalías demuestra que ambos modelos producen **0 falsos positivos** en todas las anomalías, con MSE del Autoencoder a múltiples órdenes de magnitud por debajo del umbral de detección. Esta propiedad es crítica para la viabilidad operacional del sistema: una tasa de falsos positivos no nula haría inmanejable el volumen de alertas en producción.

**5.** El sistema de restauración automática demuestra que el **tiempo de convergencia depende fundamentalmente de la localidad de la acción correctiva**, no de la magnitud de la anomalía. La estabilización del buffer de jitter (acción local, $\lambda = 0.40$) converge en ~196 segundos, mientras que la priorización QoS DSCP (acción distribuida en el plano de control, $\lambda = 0.15$) requiere ~1120 segundos, un factor de 5.7× más lento a pesar de que el diferencial de valor de la anomalía tratada (jitter: ratio 4×) es mucho menor que el de la latencia (ratio 61×).

**6.** La validación textual mediante logs Cisco IOS XR confirma que las anomalías detectadas por los modelos numéricos corresponden a eventos con identidad semántica real en el dominio operacional de la red. El Isolation Forest sobre representación TF-IDF de los logs obtiene **F1 = 0.9655 de forma uniforme para todas las anomalías**, coherente con los mejores resultados de la LSTM (packet loss y jitter). La coherencia entre el dominio numérico y el textual valida la plausibilidad operacional del sistema completo.

| Anomalía | AE F1 óptimo | LSTM F1 óptimo | LSTM AUC | Restauración | λ |
|---|---|---|---|---|---|
| A1 — Congestión | 0.4651 | 0.7273 | 0.9130 | Traffic shaping | 0.30 |
| **A2 — Packet loss** | **0.9655** | **0.9825** | **1.0000** | **ARQ selectivo** | **0.50** |
| A3 — Alta latencia | 0.5333 | 0.3810 | 0.8346 | QoS DSCP EF | 0.15 |
| A4 — Jitter extremo | 0.4444 | 0.9655 | 1.0000 | Buffer de jitter | 0.40 |

*Tabla 77 Resumen de los mejores resultados del sistema por anomalía.*

### 6.1.2 CONCLUSIONES METODOLÓGICAS

Más allá de los resultados numéricos concretos, el trabajo ha generado un conjunto de conclusiones metodológicas generalizables sobre el diseño de sistemas de detección de anomalías en redes que trascienden el dataset específico analizado.

**1.** El análisis de correlación de Pearson no es un predictor suficiente de la detectabilidad de una anomalía por modelos de machine learning. La Anomalía 4 (jitter extremo) tiene correlaciones estáticas con todas las demás features prácticamente nulas (r máx. = 0.101) y sin embargo la LSTM la predice con AUC = 1.0. **La correlación global puede ser nula incluso cuando existen dependencias temporales locales explotables** por modelos recurrentes. En el diseño de pipelines de ML para series temporales, el análisis de correlación debe complementarse con

análisis de información mutua temporal o con validación directa mediante modelos recurrentes.

**2.** El fenómeno de **dilución del MSE** en Autoencoders con ventanas temporales es un factor limitante estructural para anomalías de tipo spike puntual. Cuando el evento ocupa menos del 10% de la ventana (1–2 de 20 pasos), el error de reconstrucción en las posiciones anómalas se promedia con el error bajo de las posiciones normales, reduciendo el MSE global. La detección por Autoencoder solo es robusta cuando la magnitud del salto de valor es suficientemente grande para superar la dilución, como ocurre en packet loss (ratio 30×) y latencia extrema (ratio 332× en el error), o cuando el fenómeno tiene correlaciones multivariables que amplifican el error en múltiples features simultáneamente.

**3.** La **complementariedad entre modelos no supervisados y supervisados** es una propiedad fundamental del sistema que no puede sustituirse con la elección del modelo más potente en términos absolutos. El Autoencoder y la LSTM tienen fortalezas ortogonales: el primero es superior para eventos de alta magnitud sin señal temporal previa; el segundo es superior para eventos con dependencias secuenciales, incluso cuando estas son invisibles al análisis estático. Un sistema que combine ambos modelos obtiene el mejor resultado de cada uno de forma complementaria.

**4.** La **calibración del umbral de detección** tiene un impacto mayor sobre el F1 final que la arquitectura del modelo. Para el Autoencoder, la diferencia entre el umbral p95 y el umbral óptimo produce cambios de F1 de hasta 0.15 puntos (Anomalía 4: de 0.3933 a 0.4444). Para la LSTM, la diferencia entre el umbral 0.5 y el óptimo permite alcanzar precisión perfecta (1.000) a costa de un recall ligeramente inferior. La estrategia de calibración del umbral debe adaptarse al coste relativo de los falsos positivos y falsos negativos en el contexto operacional: en un sistema de producción, el coste de un falso negativo (anomalía no detectada) es generalmente mayor que el de un falso positivo (alerta espuria).

**5.** La **generación sintética de datos**, tanto de ventanas temporales con ruido gaussiano para la validación de consistencia como de logs Cisco IOS XR para la validación operacional, es una estrategia metodológica válida cuando no se dispone de datasets etiquetados del tipo de anomalía específico que se desea estudiar. Sin embargo, la validación sintética puede sobreestimar el rendimiento cuando el ruido añadido interactúa de forma no lineal con la magnitud del evento, como se observó en la Anomalía 3 (diferencia de 27.6 pp entre detección sintética y real). La interpretación de los resultados sintéticos requiere siempre verificar la coherencia con el rendimiento sobre datos reales.

**6.** El **etiquetado empírico mediante percentiles** (percentil 90 como umbral de anomalía en todas las anomalías) produce distribuciones de clases consistentes (desbalance ~9:1 en todos los casos) pero puede capturar como anómalos episodios que en términos operacionales absolutos no superarían los umbrales de los estándares de calidad (p. ej., latencias de 11 ms etiquetadas como anómalas frente al límite ITU-

T G.114 de 150 ms). El diseño del umbral de etiquetado es una decisión metodológica que impacta directamente sobre la interpretabilidad operacional del sistema y debe revisarse en función del SLA específico de la red analizada.

## *6.2 LIMITACIONES DEL SISTEMA*

El sistema desarrollado presenta un conjunto de limitaciones identificadas a lo largo del análisis que deben tenerse en cuenta al interpretar los resultados y al considerar su extensión a entornos de producción. Estas limitaciones se agrupan en tres categorías: limitaciones de los modelos, limitaciones del dataset y limitaciones de la arquitectura del sistema.

Asimismo, el proyecto contribuye indirectamente a varios Objetivos de Desarrollo Sostenible relacionados con la innovación, las infraestructuras digitales y la eficiencia operativa. La correspondencia detallada con las metas de la Agenda 2030 se recoge en el Anexo I.

### 6.2.1 LIMITACIONES DE LOS MODELOS

La **baja detectabilidad no supervisada del jitter y la congestión** (Autoencoder F1 < 0.50) implica que el sistema de detección sin etiquetas no es suficientemente fiable para estas anomalías en producción. Aunque la LSTM compensa esta limitación cuando se dispone de datos etiquetados, en escenarios de despliegue inicial sin etiquetas históricas el sistema dependería exclusivamente del Autoencoder, cuyo rendimiento para estas dos anomalías es insuficiente.

La **heterogeneidad interna de la clase anómala de alta latencia** (desde 11 ms hasta 3051 ms bajo la misma etiqueta) hace que no exista un umbral de Autoencoder capaz de detectar eficazmente tanto los spikes moderados como los extremos. El F1 = 0.533 del mejor modelo para esta anomalía es el reflejo directo de esta heterogeneidad y no mejorará sustancialmente sin un re-etiquetado jerárquico o la incorporación de modelos específicos por rango de magnitud.

La **capacidad de generalización de la LSTM para el jitter** (AUC = 1.0 con solo 15 muestras anómalas en el conjunto de test) es estadísticamente prometedora pero no concluyente. Los intervalos de confianza asociados a una muestra de 15 eventos son amplios, y el resultado podría ser parcialmente consecuencia de sobreajuste a los patrones específicos del Network Anomaly Dataset. La validación sobre un conjunto de test más grande o sobre un dataset independiente es necesaria antes de afirmar con certeza que la LSTM ha aprendido dependencias temporales generalizables.

### 6.2.2 LIMITACIONES DEL DATASET

El **tamaño total del dataset (1001 muestras)** es reducido para el entrenamiento de modelos de deep learning robustos. Aunque el desbalance de clases se gestiona mediante class_weight en la LSTM y mediante entrenamiento exclusivo sobre normales en el Autoencoder, la clase minoritaria de cada anomalía dispone de solo 62–100 muestras. En modelos recurrentes, la

representación de la diversidad de patrones anómalos con tan pocas muestras es necesariamente limitada.

El dataset procede de un entorno de laboratorio controlado (routers virtuales XRv en una topología específica), lo que implica que las distribuciones de las métricas y las correlaciones entre features pueden diferir significativamente de las observadas en infraestructuras de red de operador real. Los **patrones de congestión, packet loss, latencia y jitter en redes de producción** con tráfico real, múltiples flujos simultáneos y condiciones de propagación variables son considerablemente más complejos y variables que los del entorno de laboratorio, lo que podría reducir el rendimiento de los modelos al desplegarlos en producción.

El dataset cubre un único día de operación (11 de mayo de 2024), lo que impide evaluar la **robustez del sistema ante variaciones estacionales, patrones de tráfico diurnos/nocturnos o evolución de las condiciones de la red** a lo largo del tiempo. Los modelos entrenados sobre datos de un único día podrían no capturar la variabilidad temporal de los patrones normales en períodos más largos, incrementando la tasa de falsos positivos ante cambios de régimen legítimos.

### 6.2.3 LIMITACIONES ARQUITECTÓNICAS DEL SISTEMA

El sistema opera sobre datos históricos con ventanas pre-computadas y no ha sido evaluado en un escenario de **inferencia en tiempo real**. En producción, la detección debe actualizarse con cada nuevo intervalo de muestreo (~28 segundos según el dataset), requiriendo que la latencia de inferencia de los modelos sea inferior a este intervalo. La LSTM, que procesa secuencialmente 20 pasos, introduce una latencia de cómputo que debe cuantificarse y optimizarse para garantizar la viabilidad en tiempo real. A modo de referencia, sistemas equivalentes basados en autoencoders para detección en red reportan latencias de inferencia del orden de 1–10 ms/ventana en CPU estándar [22], valores muy inferiores al intervalo de muestreo de ~28.000 ms, lo que sugiere que la viabilidad computacional es alcanzable.

El sistema entrena modelos independientes por tipo de anomalía sobre un único router. La **escalabilidad a redes con múltiples routers y topologías heterogéneas** requiere una arquitectura radicalmente diferente: modelos generalizados que aprendan representaciones de normalidad transferibles entre dispositivos, mecanismos de actualización incremental (online learning) para adaptarse a cambios de régimen sin reentrenamiento completo, y una capa de orquestación que coordine las alertas de detección entre múltiples puntos de monitorización simultáneamente.

La **separación entre anomalías** (un modelo por tipo de anomalía) es una simplificación que en producción no es siempre válida: los eventos de red reales pueden involucrar múltiples anomalías simultáneas y correladas (p. ej., congestión que provoca packet loss y latencia simultáneamente). El sistema actual no está diseñado para detectar ni gestionar anomalías compuestas, lo que limita su aplicabilidad en escenarios de degradación multidimensional de la red.

## *6.3 LÍNEAS DE TRABAJO FUTURO*

Las limitaciones identificadas, junto con los resultados obtenidos, sugieren un conjunto de líneas de trabajo futuro que permitirían extender y mejorar el sistema propuesto. Se organizan por orden de impacto potencial sobre el rendimiento y la aplicabilidad del sistema.

1. **Etiquetado jerárquico de la latencia y modelos por nivel de severidad.** La principal limitación de detectabilidad identificada es la heterogeneidad de la clase anómala de alta latencia. La línea de mejora más directa es distinguir entre latencia moderada (10–100 ms), alta (100–1000 ms) y extrema (>1000 ms), entrenando umbrales o modelos diferenciados por nivel. Esta estrategia permitiría al Autoencoder ajustar su umbral a la magnitud típica del subconjunto, eliminando la ambigüedad actual entre spikes de magnitudes radicalmente distintas bajo la misma etiqueta.

2. **Incorporación de features de variabilidad temporal explícitas.** Para mejorar la detectabilidad no supervisada del jitter y la congestión, se propone añadir al vector de features la desviación típica local de cada métrica en una ventana de 3–5 pasos (sliding std), la tasa de cambio (primera derivada discreta) y estadísticos de orden como el rango inter-cuartílico local. Estas features transforman la señal de variabilidad, que es la señal característica del jitter, invisible en el valor absoluto, en una señal de valor absoluto que el Autoencoder puede aprender a reconstruir mal ante anomalías, mejorando el ratio de separación del error de reconstrucción.

3. **Modelos de detección de anomalías compuestas.** El sistema actual trata cada tipo de anomalía de forma independiente. Una extensión natural es el diseño de un modelo multietiqueta que detecte simultáneamente múltiples anomalías en la misma ventana temporal. Un Autoencoder multitarea (con una rama de reconstrucción por feature principal) o una LSTM con salida multidimensional permitirían detectar y distinguir episodios en los que la congestión provoca simultáneamente packet loss y latencia elevada, modelando las correlaciones causales entre anomalías.

4. **Validación sobre datasets independientes y en entornos de producción.** El resultado de AUC = 1.0 de la LSTM para el jitter requiere validación sobre conjuntos de test más grandes y con mayor diversidad de condiciones de red. La recopilación o uso de datasets de redes reales con métricas equivalentes (p. ej., datasets de monitorización SNMP o NetFlow con throughput, latencia y jitter medidos en routers de operador) permitiría confirmar la generalización de los patrones temporales aprendidos y cuantificar la degradación de rendimiento al pasar de laboratorio a producción.

5. **Arquitectura de detección en tiempo real con inferencia de baja latencia.** Sistemas equivalentes reportan latencias de inferencia del orden de 1–10 ms/ventana [22], muy inferiores al intervalo de muestreo de ~28.000 ms, lo que sugiere que la viabilidad computacional es alcanzable en principio. No obstante, para la viabilidad operacional en producción, el sistema debe implementarse como un pipeline de

streaming con actualización de ventanas en cada nuevo intervalo de muestreo. La optimización de la LSTM para inferencia en tiempo real (quantization, pruning, uso de implementaciones ONNX Runtime o TensorRT) y la integración con sistemas de monitorización estándar (SNMP, streaming telemetry gRPC/gNMI, plataformas como Grafana Mimir o InfluxDB) son los pasos necesarios para una implementación operacional real."

6. **Transfer learning y modelos generalizados entre routers.** Para escalar el sistema a infraestructuras con múltiples dispositivos, se propone el uso de técnicas de transfer learning: preentrenar un modelo de detección sobre datos de múltiples routers con comportamiento similar y ajustar (fine-tuning) sobre los datos específicos de cada nuevo dispositivo con pocas muestras. Este enfoque, explorado en la literatura reciente bajo el nombre de few-shot anomaly detection, permitiría desplegar el sistema en nuevos routers sin requerir el ciclo completo de recopilación de datos y reentrenamiento.

7. **Detección de la causa raíz (Root Cause Analysis).** El sistema actual detecta y clasifica el tipo de anomalía (congestión, packet loss, latencia, jitter) pero no identifica la causa raíz del evento ni la localización precisa en la topología de red. La integración con técnicas de análisis de causa raíz basadas en grafos causales (Granger causality, DoWhy, o redes bayesianas sobre la topología de red) permitiría no solo alertar sobre la anomalía sino proporcionar al operador un diagnóstico de la causa y el componente afectado, reduciendo el tiempo medio de reparación (MTTR).

8. **Extensión del sistema de restauración a acciones dinámicas y adaptativas.** El sistema de restauración actual implementa acciones correctivas predefinidas con parámetros fijos (constante de decaimiento $\lambda$ constante por tipo de anomalía). Una extensión relevante es el diseño de acciones adaptativas que ajusten su agresividad en función de la severidad del evento detectado: para un spike de latencia de 50 ms, la acción de QoS es suficiente; para un spike de 3000 ms, puede requerirse una reconfiguración de rutas. La integración con controladores SDN (Software Defined Networking) como ONOS o OpenDaylight permitiría implementar acciones correctivas dinámicas de forma programática sobre la infraestructura de red.

El presente trabajo demuestra la viabilidad técnica de un sistema de detección, predicción y restauración automática de anomalías en redes basado en machine learning, implementado sobre datos reales de routers Cisco IOS XR. Los resultados obtenidos (LSTM AUC = 1.0 en dos de las cuatro anomalías, 0 falsos positivos en validación cruzada, consistencia entre detección numérica y textual con F1 = 0.97 en logs de sistema) validan el enfoque propuesto y establecen una base técnica sólida para su extensión a sistemas de mayor escala. Las limitaciones identificadas (en particular la detectabilidad no supervisada de anomalías de baja correlación multivariable y la necesidad de validación en entornos de producción) definen con precisión el trabajo necesario para la transición de un prototipo de investigación a un sistema operacional.

# Capítulo 7. BIBLIOGRAFÍA

[1] W. Stallings, SNMP, SNMPv2, SNMPv3, and RMON 1 and 2, 3rd ed. Reading, MA, USA: Addison-Wesley, 1999.

[2] Cisco Systems, Cisco Annual Internet Report (2018–2023) White Paper. San Jose, CA, USA: Cisco Systems, 2020. [Online]. Available: https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html

[3] G. Fernandes, J. J. P. C. Rodrigues, L. F. Carvalho, J. F. Al-Muhtadi, and M. L. Proença, "A comprehensive survey on network anomaly detection," Telecommunication Systems, vol. 70, no. 3, pp. 447–489, 2019, doi: 10.1007/s11235-018-0475-8.

[4] Sandvine, The Global Internet Phenomena Report, 2023. [Online]. Available: https://www.sandvine.com/global-internet-phenomena-report

[5] M. Ahmed, A. N. Mahmood, and J. Hu, "A survey of network anomaly detection techniques," Journal of Network and Computer Applications, vol. 60, pp. 19–31, Jan. 2016, doi: 10.1016/j.jnca.2015.11.016.

[6] ITU-T, Recommendation G.1010: End-user Multimedia QoS Categories. Geneva, Switzerland: International Telecommunication Union, 2001.

[7] R. Bace, Intrusion Detection Systems. Indianapolis, IN, USA: Macmillan Technical Publishing, 2000.

[8] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," ACM Computing Surveys, vol. 41, no. 3, Art. no. 15, Jul. 2009, doi: 10.1145/1541880.1541882.

[9] A. L. Buczak and E. Guven, "A survey of data mining and machine learning methods for cyber security intrusion detection," IEEE Communications Surveys & Tutorials, vol. 18, no. 2, pp. 1153–1176, 2016, doi: 10.1109/COMST.2015.2494494.

[10] B. Claise, B. Trammell, and P. Aitken, "Specification of the IPFIX Protocol for the Exchange of Flow Information," RFC 7011, Sep. 2013. [Online]. Available: https://www.rfc-editor.org/rfc/rfc7011

[11] W. Stallings, Data and Computer Communications, 10th ed. Upper Saddle River, NJ, USA: Pearson, 2013.

[12] Creately, "Diagrama de Red Ejemplo," 2024. [Online]. Available: https://creately.com/diagram/example/T0GEqOAqhRe/diagrama-de-red-ejemplo

[13] I. Solís, Gestión de Redes Basada en FCAPS, 2014.

[14] B. Claise, "Cisco Systems NetFlow Services Export Version 9," RFC 3954, Oct. 2004. [Online]. Available: https://www.rfc-editor.org/rfc/rfc3954

[15] Kentik, "Big Data IPFIX Analysis: Network Monitoring Architecture," 2024. [Online]. Available: https://www.kentik.com/kentipedia/big-data-ipfix-analysis/

[16] M. Roesch, "Snort: Lightweight intrusion detection for networks," in Proceedings of the 13th USENIX Conference on System Administration (LISA '99), Seattle, WA, USA, 1999, pp. 229–238.

[17] M. Barabas, I. Homoliak, and P. Hanacek, "Network Fault Management and Intrusion Detection," 2012.

[18] ITU-T, Recommendation G.114: One-Way Transmission Time. Geneva, Switzerland: International Telecommunication Union, 2003.

[19] P. Malhotra, L. Vig, G. Shroff, and P. Agarwal, "Long Short Term Memory Networks for Anomaly Detection in Time Series," in Proceedings of the 23rd European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN), Bruges, Belgium, 2015.

[20] A. Lakhina, M. Crovella, and C. Diot, "Diagnosing network-wide traffic anomalies," in Proceedings of the ACM SIGCOMM Conference, Portland, OR, USA, 2004, pp. 219–230, doi: 10.1145/1030194.1015492.

[21] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation Forest," in Proceedings of the IEEE International Conference on Data Mining (ICDM), Pisa, Italy, 2008, pp. 413–422, doi: 10.1109/ICDM.2008.17.

[22] Y. Mirsky, T. Doitshman, Y. Elovici, and A. Shabtai, "Kitsune: An Ensemble of Autoencoders for Online Network Intrusion Detection," in *Proceedings of the Network and Distributed System Security Symposium (NDSS)*, San Diego, CA, USA, 2018.

[23] J. Mirkovic and P. Reiher, "A taxonomy of DDoS attack and DDoS defense mechanisms," *ACM SIGCOMM Computer Communication Review*, vol. 34, no. 2, pp. 39–53, Apr. 2004, doi: 10.1145/997150.997156.

[24] S. T. Zargar, J. Joshi, and D. Tipper, "A survey of defense mechanisms against distributed denial of service (DDoS) flooding attacks," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 4, pp. 2046–2069, 2013, doi: 10.1109/SURV.2013.031413.00127.

[25] J. An and S. Cho, *Variational Autoencoder Based Anomaly Detection Using Reconstruction Probability*. Seoul, South Korea: Seoul National University, Tech. Rep. SNUDM-TR-2015-03, 2015. [Online]. Available: http://dm.snu.ac.kr/static/docs/TR/SNUDM-TR-2015-03.pdf

[26] T. Baer, J. Bruno, and J. Cahn, "Service level agreements: Beyond the fine print," *IEEE Internet Computing*, vol. 5, no. 5, pp. 38–49, Sep.–Oct. 2001, doi: 10.1109/4236.935181.

[27] LinkedIn Pulse, "The Rise of AI-Driven Networking," 2024. [Online]. Available: https://www.linkedin.com/pulse/rise-ai-driven-networking-transforming-network-infrastructure-s2wdf

[28] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 4th ed. Hoboken, NJ, USA: Pearson, 2020.

[29] T. M. Mitchell, *Machine Learning*. New York, NY, USA: McGraw-Hill, 1997.

[30] Z. Che, S. Purushotham, K. Cho, D. Sontag, and Y. Liu, "Recurrent neural networks for multivariate time series with missing values," *Scientific Reports*, vol. 8, Art. no. 6085, 2018, doi: 10.1038/s41598-018-24271-9.

[31] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997, doi: 10.1162/neco.1997.9.8.1735.

[32] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016. [Online]. Available: https://www.deeplearningbook.org

[33] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, May 2015, doi: 10.1038/nature14539.

[34] H. Song, Z. Jiang, A. Men, and B. Yang, "A hybrid semi-supervised anomaly detection model for high-dimensional data," *Cognitive Computation*, vol. 9, no. 3, pp. 364–377, 2017, doi: 10.1007/s12539-016-0143-0.

[35] S. Agrahari, "Machine Learning Pipeline: From Raw Data to Insightful Models," 2024. [Online]. Available: https://medium.com/@surajagrahari330/machine-learning-pipeline-from-raw-data-to-insightful-models-80c54914b7a9

[36] Innovación Digital 360, "Deep Learning: Qué es el Aprendizaje Profundo," 2024.

[37] G. E. P. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung, *Time Series Analysis: Forecasting and Control*, 5th ed. Hoboken, NJ, USA: Wiley, 2015.

[38] F. Pedregosa et al., "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.
[39] T. Fawcett, "An introduction to ROC analysis," *Pattern Recognition Letters*, vol. 27, no. 8, pp. 861–874, 2006, doi: 10.1016/j.patrec.2005.10.010.
[40] M. Abadi et al., "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015. [Online]. Available: https://www.tensorflow.org/
[41] F. Chollet, "Keras: Deep learning for humans," 2015. [Online]. Available: https://github.com/keras-team/keras
[42] K. Sparck Jones, "A statistical interpretation of term specificity and its application in retrieval," *Journal of Documentation*, vol. 28, no. 1, pp. 11–21, 1972, doi: 10.1108/eb026526.
[43] J. Zhu, S. He, J. Liu, P. He, Q. Xie, Z. Zheng, and M. R. Lyu, "Tools and benchmarks for automated log analysis," *IEEE Software*, vol. 36, no. 4, pp. 52–60, 2019, doi: 10.1109/MS.2019.2906436.
[44] P. He, J. Zhu, Z. Zheng, and M. R. Lyu, "Drain: An online log parsing approach with fixed depth tree," in *Proceedings of the IEEE International Conference on Web Services (ICWS)*, Honolulu, HI, USA, 2017, pp. 33–40, doi: 10.1109/ICWS.2017.13.
[45] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995, doi: 10.1007/BF00994018.
[46] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson, "Estimating the support of a high-dimensional distribution," *Neural Computation*, vol. 13, no. 7, pp. 1443–1471, 2001, doi: 10.1162/089976601750264965.
[47] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001, doi: 10.1023/A:1010933404324.
[48] Cornell University, "CS4780 Lecture Notes: Support Vector Machines," 2017. [Online]. Available: https://www.cs.cornell.edu/courses/cs4780/2017sp/lectures/lecturenote09.html
[49] Analytics Vidhya, "One-Class SVM for Anomaly Detection," Mar. 18, 2024. [Online]. Available: https://www.analyticsvidhya.com/blog/2024/03/one-class-svm-for-anomaly-detection/
[50] N. da Silva, D. Cook, and E.-K. Lee, "Interactive graphics for visually diagnosing forest classifiers in R," *WIREs Computational Statistics*, vol. 11, no. 3, Art. no. e1465, 2019, doi: 10.1002/wics.1465.
[51] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, Berkeley, CA, USA, vol. 1, pp. 281–297, 1967.
[52] S. P. Lloyd, "Least squares quantization in PCM," *IEEE Transactions on Information Theory*, vol. 28, no. 2, pp. 129–137, 1982, doi: 10.1109/TIT.1982.1056489.
[53] D. Arthur and S. Vassilvitskii, "k-means++: The advantages of careful seeding," in *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, New Orleans, LA, USA, 2007, pp. 1027–1035.
[54] Analytics Vidhya, "In-Depth Intuition of K-Means Clustering Algorithm in Machine Learning," 2021. [Online]. Available: https://www.analyticsvidhya.com/blog/2021/01/in-depth-intuition-of-k-means-clustering-algorithm-in-machine-learning/
[55] A. Yavariabdi, "Anomaly Detection in Python with Gaussian Distribution and K-Means Clustering," 2023. [Online]. Available: https://medium.com/@amir.yavariabdi/anomaly-detection-in-python-with-gaussian-distribution-and-k-means-clustering-89d09d16fba4
[56] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation Forest," in *Proceedings of the IEEE International Conference on Data Mining (ICDM)*, Pisa, Italy, 2008, pp. 413–422, doi: 10.1109/ICDM.2008.17.

[57] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation-based anomaly detection," *ACM Transactions on Knowledge Discovery from Data*, vol. 6, no. 1, pp. 1–39, 2012, doi: 10.1145/2133360.2133363.

[58] Spot Intelligence, "Isolation Forest: Algorithm, Working Principle and Applications," 2024. [Online]. Available: https://spotintelligence.com/2024/05/21/isolation-forest/

[59] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006, doi: 10.1126/science.1127647.

[60] IBM, "Variational Autoencoder," 2024. [Online]. Available: https://www.ibm.com/es-es/think/topics/variational-autoencoder

[61] Dive into Deep Learning, "Long Short-Term Memory (LSTM)," 2024. [Online]. Available: https://classic.d2l.ai/chapter_recurrent-modern/lstm.html

[62] H. He and E. A. Garcia, "Learning from imbalanced data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 21, no. 9, pp. 1263–1284, 2009, doi: 10.1109/TKDE.2008.239.

[63] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM Transactions on Networking*, vol. 1, no. 4, pp. 397–413, 1993, doi: 10.1109/90.251892.

[64] S. Lin and D. J. Costello, *Error Control Coding*, 2nd ed. Upper Saddle River, NJ, USA: Pearson, 2004.

[65] K. Nichols, S. Blake, F. Baker, and D. Black, "Definition of the differentiated services field (DS field) in the IPv4 and IPv6 headers," RFC 2474, Dec. 1998. [Online]. Available: https://www.rfc-editor.org/rfc/rfc2474

[66] ResearchGate, "Schematic Diagram of DDoS Attack," 2022. [Online]. Available: https://www.researchgate.net/figure/Schematic-diagram-of-DDoS-attack_fig1_366015028

# ANEXO I: ALINEACIÓN DEL PROYECTO CON LOS ODS

La Agenda 2030 para el Desarrollo Sostenible, adoptada por las Naciones Unidas en 2015, establece 17 Objetivos de Desarrollo Sostenible (ODS) que constituyen un llamamiento universal a la acción para erradicar la pobreza, proteger el planeta y garantizar que todas las personas gocen de paz y prosperidad. En el contexto de la ingeniería de telecomunicaciones, la contribución a los ODS se articula principalmente a través del desarrollo tecnológico responsable que mejore la eficiencia, la fiabilidad y la accesibilidad de las infraestructuras de comunicación.

El presente trabajo, ***Detección, Predicción y Restauración Automática de Anomalías en Redes mediante Inteligencia Artificial***, tiene una conexión directa y sustancial con varios ODS. La automatización de la gestión de redes mediante machine learning contribuye a la construcción de infraestructuras más resilientes, a la reducción del consumo energético asociado a la degradación de la red, y al fortalecimiento de las alianzas tecnológicas necesarias para el desarrollo sostenible. A continuación se analiza en detalle la alineación del trabajo con los cuatro ODS más directamente relacionados.

| ODS | Nombre | Contribución principal del proyecto |
|---|---|---|
| ODS 9 | Industria, Innovación e Infraestructura | Mejora de la resiliencia y eficiencia de infraestructuras de red mediante IA |
| ODS 11 | Ciudades y Comunidades Sostenibles | Sostenibilidad de infraestructuras digitales críticas para ciudades inteligentes |
| ODS 13 | Acción por el Clima | Reducción del consumo energético por anomalías de red no corregidas |
| ODS 17 | Alianzas para lograr los Objetivos | Uso de datos abiertos, software open source y metodología reproducible |

*Tabla 78 Alineación del proyecto con los Objetivos de Desarrollo Sostenible de la Agenda 2030.*

## *ODS 9 · INDUSTRIA, INNOVACIÓN E INFRAESTRUCTURA*

El ODS 9 persigue construir infraestructuras resilientes, promover la industrialización inclusiva y sostenible y fomentar la innovación. La meta 9.1 establece explícitamente el objetivo de **desarrollar infraestructuras fiables, sostenibles, resilientes y de calidad**, incluyendo infraestructuras regionales y transfronterizas, para apoyar el desarrollo económico y el bienestar humano. La meta 9.b impulsa el apoyo al desarrollo tecnológico, la investigación y la innovación en los países en desarrollo, fomentando un entorno normativo propicio a la diversificación industrial.

La contribución del presente trabajo al ODS 9 es directa y multidimensional:

- **Resiliencia de infraestructuras.** El sistema desarrollado detecta y corrige automáticamente anomalías en redes de telecomunicaciones (congestión, pérdida de paquetes, alta latencia, jitter extremo) antes de que produzcan degradación perceptible del servicio. Esta capacidad de autoreparación incrementa la resiliencia

operacional de la infraestructura: una red que detecta y neutraliza sus propias anomalías en tiempos de entre 196 y 1120 segundos, frente a los minutos u horas que requeriría la intervención manual, opera con mayor continuidad de servicio.

- **Innovación en la gestión de redes.** La aplicación de técnicas de aprendizaje profundo (Autoencoder, LSTM) a la gestión autónoma de redes representa una innovación respecto a los sistemas de monitorización basados en umbrales fijos, que son el estándar actual en la mayoría de operadoras. El sistema propuesto aprende la normalidad directamente de los datos sin reglas codificadas manualmente, lo que permite adaptarse a condiciones de red variables y detectar patrones anómalos que los sistemas tradicionales no capturan.
- **Eficiencia de infraestructuras de telecomunicaciones.** La restauración automática de la red minimiza el tiempo de operación en condiciones degradadas. Una red que opera en congestión consume más energía por bit transmitido (mayor número de retransmisiones, mayor occupancy de buffers, mayor tiempo de procesamiento en nodos intermedios) que una red en condiciones normales. La reducción del tiempo de anomalía activa, gracias a la detección y restauración automática, se traduce directamente en una mayor eficiencia energética de la infraestructura.
- **Tecnología aplicable a países en desarrollo.** El sistema se implementa sobre un dataset de routers virtuales Cisco IOS XR (xrv6, xrv1), accesible sin hardware físico costoso. La metodología basada en software de código abierto (Python, TensorFlow/Keras, scikit-learn) y datos públicos (Kaggle) garantiza que la tecnología desarrollada puede replicarse en contextos con recursos limitados, contribuyendo a la meta 9.b de democratización del acceso a tecnologías de gestión de red avanzadas.

**Contribución cuantificable al ODS 9:** *El sistema reduce el tiempo de detección de anomalías de varios minutos (intervención manual) a menos de 60 segundos (detección automática en el siguiente intervalo de muestreo), y el tiempo de restauración a entre 196 y 1120 segundos según el tipo de anomalía. Esta mejora en la continuidad de servicio contribuye directamente a la resiliencia de la infraestructura de red.*

## *ODS 11 · CIUDADES Y COMUNIDADES SOSTENIBLES*

El ODS 11 busca lograr que las ciudades y los asentamientos humanos sean inclusivos, seguros, resilientes y sostenibles. En el contexto de la transformación digital, las ciudades inteligentes (smart cities) dependen de infraestructuras de telecomunicaciones fiables para el funcionamiento de sus sistemas críticos: redes de sensores IoT, sistemas de transporte inteligente, gestión de emergencias, servicios públicos digitales y comunicaciones de seguridad ciudadana.

La meta 11.2 establece el objetivo de proporcionar acceso a sistemas de transporte seguros, asequibles, accesibles y sostenibles, mejorando la seguridad vial. La meta 11.b exige aumentar el número de ciudades y asentamientos que adoptan políticas y planes integrados para la inclusión, la eficiencia de los recursos, la mitigación del cambio climático y la resiliencia ante desastres.

La relación del presente trabajo con el ODS 11 se articula en torno a la **sostenibilidad de las infraestructuras digitales críticas** que sostienen los servicios de las ciudades inteligentes:

- **Continuidad de servicios críticos urbanos.** Los servicios digitales de una ciudad inteligente, como la gestión del tráfico, las redes de sensores medioambientales, los sistemas de alerta temprana, comunicaciones de emergencia, dependen de redes de telecomunicaciones con garantías estrictas de calidad de servicio (latencia < 150 ms para VoIP según ITU-T G.114, jitter < 50 ms según ITU-T G.1010, pérdida de paquetes < 1% para servicios críticos). Un sistema de detección y restauración automática de anomalías como el propuesto en este trabajo garantiza que estas métricas se mantengan dentro de los umbrales operacionales, contribuyendo a la continuidad de los servicios urbanos.
- **Reducción de la brecha digital.** Las anomalías de red no detectadas afectan de forma desproporcionada a los usuarios con menor capacidad de adaptación ante la degradación del servicio: personas mayores, usuarios de servicios de salud digital, estudiantes en entornos de educación a distancia. La detección automática y la restauración rápida de las condiciones normales reducen el impacto de las anomalías sobre los colectivos más vulnerables, contribuyendo a la meta 11.1 de acceso universal a servicios básicos.
- **Infraestructura para la gestión de emergencias.** Los sistemas de comunicación de emergencia (llamadas al 112, redes de primeros respondedores, sistemas de alerta masiva) son especialmente sensibles al jitter y la latencia. El sistema de detección y restauración automática de jitter extremo y alta latencia desarrollado en este trabajo, con tiempos de recuperación de 196 y 1120 segundos, respectivamente, reduce el riesgo de degradación de las comunicaciones de emergencia durante un evento crítico.

**Contribución cuantificable al ODS 11:** *El sistema garantiza que las anomalías de red que comprometerían la calidad de los servicios digitales urbanos (jitter > 1.37 ms, latencia > 10.87 ms, packet loss > umbral operacional) sean detectadas y corregidas automáticamente, manteniendo la disponibilidad de las infraestructuras digitales críticas para la ciudad sostenible.*

## *ODS 13 · ACCIÓN POR EL CLIMA*

El ODS 13 insta a adoptar medidas urgentes para combatir el cambio climático y sus efectos. La meta 13.2 establece la integración de medidas relativas al cambio climático en las políticas, estrategias y planes nacionales. Aunque la relación entre las redes de telecomunicaciones y el cambio climático puede parecer indirecta, el sector TIC es responsable de aproximadamente el 2-4% de las emisiones globales de $CO_2$, y la eficiencia energética de las infraestructuras de red es un factor clave en la reducción de este impacto.
La conexión entre el presente trabajo y el ODS 13 se articula a través del **impacto energético de las anomalías de red no corregidas**:

- **Reducción del consumo energético por congestión.** Una red en estado de congestión opera con mayor ineficiencia energética: los paquetes descartados deben

retransmitirse, los buffers permanecen al máximo de ocupación consumiendo más memoria activa, y los protocolos de control de congestión (TCP) reducen la ventana de envío, aumentando el número de intercambios de control necesarios para transmitir el mismo volumen de datos. La detección y corrección automática de la congestión mediante traffic shaping, con un tiempo de recuperación de ~420 segundos en el sistema propuesto, reduce el tiempo de operación en este régimen de mayor consumo energético.

- **Reducción de retransmisiones por packet loss.** Cada paquete perdido que requiere retransmisión supone un consumo energético adicional: el paquete debe ser regenerado, retransmitido a través de la infraestructura y procesado de nuevo en el destino. En escenarios de alta pérdida de paquetes (> 10%), el incremento del tráfico total por retransmisiones puede suponer un aumento del consumo energético de la infraestructura de hasta un 15-20%. La corrección automática mediante ARQ selectivo, con tiempo de recuperación de ~280 segundos, minimiza este sobreconsumo.
- **Eficiencia energética de la IA empleada.** Los modelos implementados en este trabajo (Autoencoder con 100 neuronas en la capa de entrada, LSTM con 64+32 unidades) son modelos de tamaño reducido, con tiempos de inferencia de milisegundos en hardware estándar. El impacto energético de la inferencia continua de estos modelos es marginal comparado con el ahorro energético que produce la reducción de las anomalías en la infraestructura de red monitorizada.

**Contribución cuantificable al ODS 13:** *La reducción del tiempo de anomalía activa (especialmente congestión y packet loss, que son las anomalías con mayor impacto en el consumo energético de la red) contribuye a la eficiencia energética de la infraestructura de telecomunicaciones y, por tanto, a la reducción de las emisiones de $CO_2$ asociadas al sector TIC.*

## *ODS 17 · ALIANZAS PARA LOGRAR LOS OBJETIVOS*

El ODS 17 busca fortalecer los medios de implementación y revitalizar la Alianza Mundial para el Desarrollo Sostenible. La meta 17.6 promueve la cooperación regional e internacional en ciencia, tecnología e innovación, y el acceso a la ciencia, la tecnología y la innovación. La meta 17.8 pone en funcionamiento la iniciativa de ayuda para el comercio en los países en desarrollo, y la meta 17.17 fomenta alianzas eficaces entre el sector público, el sector privado y la sociedad civil.

El presente trabajo contribuye al ODS 17 a través de varios principios metodológicos fundamentales:

- **Uso de datos abiertos.** El dataset utilizado, Network Anomaly Dataset (Kaggle/UPM, 2024), es de acceso público y gratuito. Esta elección no es solo metodológica sino también ética: el uso de datos abiertos permite que cualquier investigador, institución o empresa, independientemente de sus recursos, pueda reproducir, validar y extender los resultados del trabajo. La publicación de los

resultados sobre datos abiertos contribuye a la construcción de conocimiento compartido en el ámbito de la gestión autónoma de redes.

- **Software de código abierto.** Todo el stack tecnológico empleado, incluyendo Python, TensorFlow/Keras, scikit-learn, pandas, NumPy, es software libre y de código abierto, ampliamente utilizado en la comunidad científica y accesible sin coste en cualquier contexto. Esta elección facilita la transferencia tecnológica de los resultados del trabajo a entornos con recursos limitados, incluidos países en desarrollo con necesidades crecientes de gestión autónoma de sus infraestructuras de telecomunicaciones.
- **Reproducibilidad científica.** La metodología del trabajo está completamente documentada: criterios de etiquetado, arquitecturas de modelos, hiperparámetros, splits de datos, métricas de evaluación y procedimientos de validación. Esta documentación exhaustiva permite que otros investigadores reproduzcan y extiendan los resultados, contribuyendo a la acumulación de conocimiento verificable en el ámbito de la detección de anomalías en redes.
- **Alineación con estándares internacionales.** El trabajo referencia y aplica estándares internacionales de calidad de red (ITU-T G.114, ITU-T G.1010) y utiliza tecnología Cisco IOS XR, el sistema operativo de routing más ampliamente desplegado en infraestructuras de operador a nivel mundial, como referencia para la generación de logs de validación. Esta alineación con estándares facilita la integración de los resultados en marcos normativos y operacionales reales.

**Contribución cuantificable al ODS 17:** *El uso de datos abiertos (Kaggle/UPM), software open source (Python, TensorFlow, scikit-learn) y documentación metodológica completa garantiza que los resultados del trabajo son reproducibles, transferibles y accesibles a cualquier institución con acceso a internet, contribuyendo a la democratización del conocimiento en gestión autónoma de redes.*

## METAS ESPECÍFICAS DE LA AGENDA 2030 CUBIERTAS

| ODS | Meta | Descripción meta | Contribución del proyecto |
|---|---|---|---|
| ODS 9 | 9.1 | Infraestructuras fiables, sostenibles y resilientes | Restauración automática que reduce el tiempo de degradación de la infraestructura de red |
| ODS 9 | 9.4 | Modernización de infraestructuras con tecnologías limpias y eficientes | Sistema de IA que optimiza la operación de la red sin hardware adicional |
| ODS 9 | 9.b | Apoyar el desarrollo tecnológico e innovación en países en desarrollo | Metodología basada en software open source y datos abiertos, replicable sin coste |
| ODS 11 | 11.1 | Acceso a servicios básicos para todos | Continuidad de servicios digitales críticos mediante detección automática de anomalías |
| ODS 11 | 11.b | Ciudades con políticas de resiliencia ante desastres | Infraestructura de red resiliente que mantiene los servicios de emergencia operativos |
| ODS 13 | 13.2 | Integrar el cambio climático en políticas y estrategias | Reducción del consumo energético de la red mediante corrección de anomalías |
| ODS 17 | 17.6 | Acceso a ciencia, tecnología e innovación | Publicación de metodología reproducible sobre datos abiertos |
| ODS 17 | 17.17 | Alianzas entre sector público, privado y sociedad civil | Stack tecnológico open source transferible a operadoras, universidades e instituciones públicas |

*Tabla 79 Metas específicas de la Agenda 2030 a las que contribuye el proyecto, con descripción de la contribución concreta.*

## REFLEXIÓN SOBRE EL IMPACTO GLOBAL DEL PROYECTO

La intersección entre la ingeniería de telecomunicaciones y los Objetivos de Desarrollo Sostenible no es circunstancial: las redes de telecomunicaciones son, en la sociedad contemporánea, una infraestructura tan fundamental como las carreteras, el agua potable o la electricidad. La degradación de la calidad de servicio de una red de telecomunicaciones no es solo un problema técnico, ya que tiene consecuencias directas sobre el acceso a la educación (plataformas de e-learning), la salud (telemedicina), la seguridad (comunicaciones de emergencia), la economía (servicios financieros digitales) y la cohesión social (comunicación interpersonal).

En este contexto, el desarrollo de sistemas que permitan gestionar autónomamente la calidad de las redes, detectando, prediciendo y corrigiendo anomalías sin intervención humana, tiene un impacto que trasciende el ámbito puramente técnico. Un sistema como el propuesto en este trabajo, desplegado en la infraestructura de una operadora de telecomunicaciones, contribuiría a garantizar que los millones de usuarios que dependen de esa red para acceder a servicios críticos disponen de una conectividad fiable, resiliente y sostenible.

La alineación del proyecto con los ODS 9, 11, 13 y 17 no es por tanto una consideración secundaria añadida a posteriori, sino una consecuencia natural de los objetivos técnicos del trabajo: construir infraestructuras más resilientes (ODS 9 y 11), reducir el impacto energético de la operación en condiciones degradadas (ODS 13), y hacerlo de forma abierta y reproducible que facilite la transferencia de conocimiento (ODS 17).

# ANEXO II. REPOSITORIO DE CÓDIGO FUENTE

Con el objetivo de facilitar la reproducibilidad de los experimentos realizados y evitar la inclusión de una gran cantidad de código fuente dentro de la memoria, se ha habilitado un repositorio GitHub que contiene la implementación completa desarrollada durante este Trabajo Fin de Grado.

Repositorio: **https://github.com/paulacollado/tfg-anomalias-red**

El repositorio incluye:

- Dataset utilizado para los experimentos.
- Notebooks de análisis y detección de anomalías.
- Implementación de los modelos Autoencoder, Isolation Forest, One-Class SVM y LSTM.
- Simulaciones de restauración de red.
- Resultados experimentales obtenidos.
- Generación y análisis de registros Syslog Cisco IOS XR.
- Figuras y métricas generadas durante la evaluación.

La documentación necesaria para comprender la estructura y funcionamiento del proyecto se encuentra disponible en el archivo README incluido en el repositorio.