Selección de Rutas BGP Basada en Latencia:

Un Motor de Failover Inteligente Optimizado con Aprendizaje Supervisado



Ernesto Sánchez
Universidad Católica de Salta. Argentina
Henri Alves de Godoy
Universidade Estadual de Campinas. Brasil

La necesidad: Latencia como métrica en BGP

- La comunidad técnica reconoce formalmente la necesidad de incorporar métricas de desempeño (latencia) en la selección de rutas BGP.
- El draft introduce el uso de un nuevo campo del atributo AIGP, (NETWORK_LATENCY TLV), que codifica la latencia acumulada de red entre el BGP speaker y el prefijo anunciado.
- Recomienda un threshold configurable para evitar actualizaciones disparadas por fluctuaciones menores.
- La latencia de enlace es más estable que la latencia de colas/procesamiento del router, justifica no reaccionar a ruido puntual.

Fuente: IETF Draft: BGP Performance-aware Routing Mechanism

De estático a dinámico: failover con detección contextual

El sistema integra monitorización continua (captura, procesamiento de latencia, jitter y pérdida), más almacenamiento en TimescaleDB con failover automático inteligente basado en scoring ponderado.

1

¿Tendencia sostenida o pico aislado?

Distinguir degradación real de ruido de un solo ciclo

2

¿Crece la variabilidad (σ) en la ventana histórica?

Coeficiente de variación como señal de inestabilidad

3

¿Desviación del percentil 95 vs. media móvil?

Detectar colas pesadas antes de que dominen el promedio

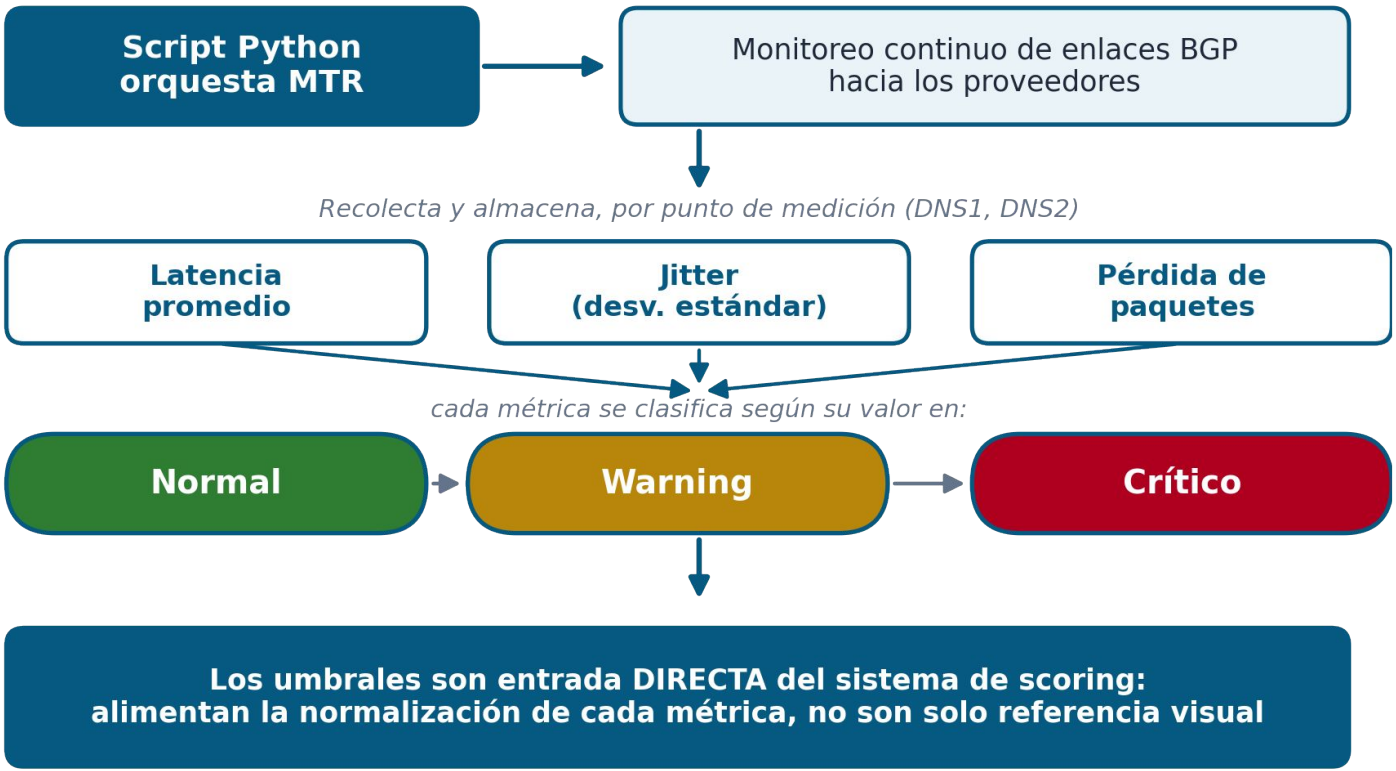
4

¿Degradación progresiva o fluctuación normal?

Derivada del Z-score para ver aceleración del deterioro

Estas cuatro preguntas son exactamente el alcance formal que justifica aplicar el paradigma de Aprendizaje Supervisado (histórico + análisis estadístico) para clasificar y predecir comportamientos de red a partir de datos históricos etiquetados

Herramienta de monitorización y recolección de métricas



Motor de failover BGP: lógica general

1. Normalización
cada métrica ÷ su umbral crítico

→

2. Score base
promedio ponderado (peer/dns/jitter)

→

3. Severidad
multiplicador por pérdida

→

4. Score final
por DNS1 y DNS2

→

5. Decisión
failover / retorno

Fórmula resumida

```
base_score = (peer_norm × 0.27) + (dns_norm × 0.50) + (jitter_norm × 0.23)
score_final = base_score × (1 + loss_norm)
```

Se calcula un score independiente para DNS1 y DNS2; la decisión usa el máximo de ambos.

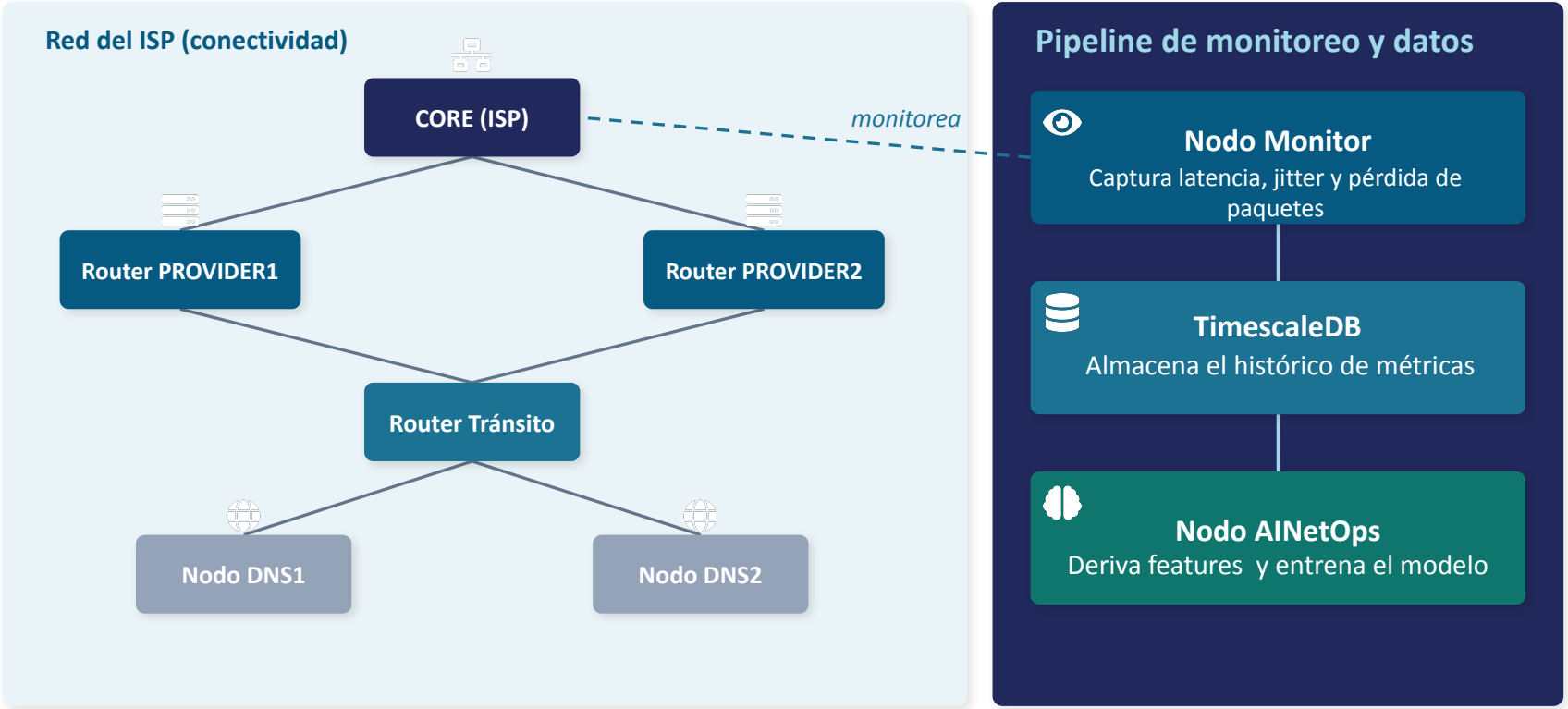
Regla de disparo

max(score_DNS1, score_DNS2) > 1.10 O breach individual (una métrica aislada supera su propio umbral crítico) sostenido durante x ciclos consecutivos ⟶ FAILOVER a provider2

max(score_DNS1, score_DNS2) < 0.80 Y sin breach individual activo, sostenido durante x ciclos consecutivos ⟶ RETORNO a provider1

lacnic

Laboratorio de pruebas: Containerlab



Aprendizaje Supervisado aplicado a BGP

$$f(X) \rightarrow Y$$

X = (peer_norm, dns1_norm, dns2_norm, jitter_norm, loss_norm, ...) en un contexto temporal

Y = decisión de failover (normal / degradación / failover / retorno)

Enfoques de ML en NetOps

- Forecasting: Aprendizaje Supervisado para predecir condiciones futuras de la red.
- Detección de anomalías: Clustering/Clasificación para comportamientos atípicos.
- Optimización de rutas: Aprendizaje por Refuerzo para automatizar políticas.
- Calibración de reglas: Validar/recalibrar los parámetros de una regla de decisión ya existente.

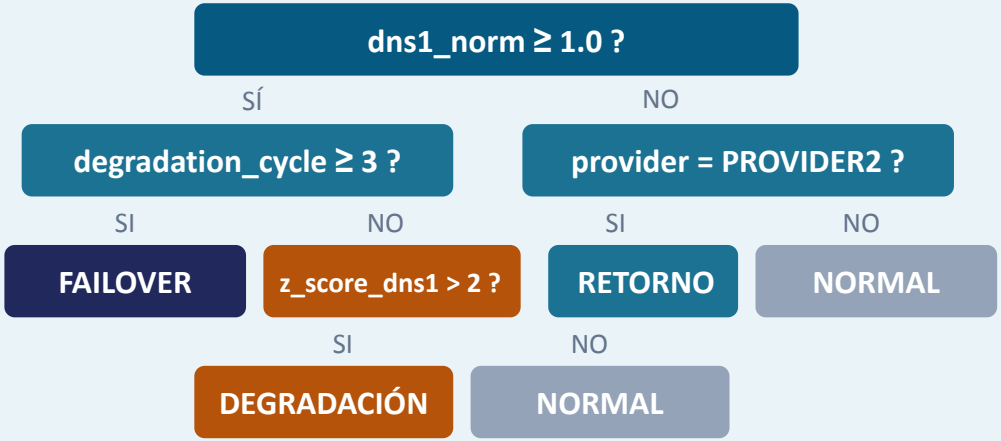
Una vez entrenado, el modelo permite:

- Identificar qué métricas tienen mayor influencia real.
- Estimar nuevas ponderaciones para la función de scoring.
- Reducir falsos positivos y disminuir el flapping.
- Adaptar la política de enrutamiento a condiciones reales.

lacnic

XGBoost: Mecanismo de split y criterios de importancia

Árbol de decisión (simplificado, features reales)



Umbrales reales del motor: breach individual (≥1.0), x ciclos de confirmación, z-score severo (>2σ). El modelo real puede aprender splits distintos, esto es solo ilustrativo del mecanismo.

Criterios de selección de features

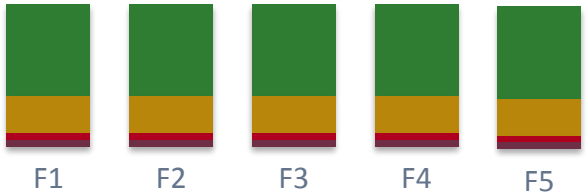
Gain mide relevancia predictiva relativa dentro de un ensamble no-lineal. No son coeficientes lineales intercambiables por los pesos de la fórmula (peer×0.27 + dns×0.50 + jitter×0.23). Por eso se complementa con Regresión Logística.

Dataset histórico etiquetado y generalización

Dataset desbalanceado | **Stratified K-Fold** | **Métricas multiclase**

Normal: (98,7%) la inmensa mayoría





cada fold conserva la misma proporción de clases

- **Precision**
- **Recall**
- **F1 Score**
- **ROC-AUC**

Cross-Validation (K=5)

- El dataset se divide en 5 folds; en cada iteración 4 entrenan y 1 valida, rotando.
- Reduce la dependencia de una única partición del dataset.
- Da una estimación más robusta del rendimiento real.

lacnic

Taxonomía de las features derivadas

1 — 9

Crudas

peer/dns1/dns2: latencia, jitter, loss

2 — ~15

Normalizadas

*_norm, base_score, severity_multiplier, score_dns1/2, max_score, quality_status

3 — ~27

Temporales/estadísticas

z_score_*, cv_*, p95_dev_*, z_deriv_*, trend/velocity/aceleración

4 — 5

Contextuales

hour_of_day, day_of_week, is_business_hours, is_peak_traffic, is_weekend

5 — 2

Contexto de cambios

provider_changes_last_hour, time_since_last_change_min

6

Target

target_decision (normal/degradación/failover/retorno)

El tamaño del espacio de features (~60) es intencional: en vez de decidir a priori qué importa, se deja que XGBoost lo determine empíricamente vía Gain, incluyendo señales que hoy no pesan en la fórmula manual (ej. contexto horario, jitter crudo) para que el análisis de importancia las evalúe también.

lacnic

Regresión Logística: De importancia a pesos interpretables

XGBoost vs. Logistic Regression

XGBoost: estructura no lineal



Gain RANKING de importancia, no son pesos comparables

Logistic Regression: ecuación lineal

$$score = w_1 \cdot peer + w_2 \cdot dns + w_3 \cdot jitter$$

Coeficientes = pesos DIRECTAMENTE comparables entre sí

Rol de cada modelo

XGBoost
¿Qué features importan?
(Gain)

Logistic Regression
¿Cuánto pesa cada una?
(misma escala que la fórmula actual)

Los pesos candidatos se validan contra 0.27 / 0.50 / 0.23 antes de reemplazar nada

lacnic

Pruebas realizadas

Pipeline Validación de captura y almacenamiento de métricas



Pipeline Validación de derivación de features y entrenamiento del modelo



[Ver video demo]

lacnic

Generador de Datos Sintéticos: Fuentes y Calibración

Procesos aleatorios utilizados

Proceso	Dónde se usa	Qué decide
Ornstein-Uhlenbeck	Ruido de fondo (tráfico "normal")	Simula latencia con memoria temporal realista
Bernoulli / Poisson	Arranque de episodios	Si un episodio empieza en este ciclo
Categórico (multinomial)	Tipo de episodio y DNS afectado	Cuál de los 5 tipos de anomalía, y a qué destino
Uniforme (cont./discreta)	Severidad y duración	Pico de latencia (ms) y cuántos ciclos dura cada tramo
Gaussiana sin memoria	Estado 'unstable'	Variabilidad pura, deliberadamente sin estructura temporal

Ruido de fondo: Ornstein-Uhlenbeck

Cada muestra depende parcialmente de la anterior, igual que la latencia real, que no salta al azar sino que se recupera gradualmente.
Fuente: https://rpubs.com/astrojcms/OU_process

Los scripts de este proyecto fueron desarrollados con asistencia de Claude (Anthropic), modelo de IA generativa, bajo la dirección y validación del autor.

lacnic

Resultados obtenidos

Dataset y distribución de clases

Registros	Rango simulado
50,000 ciclos	~26 días (14 jun – 10 jul)

Clase	Registros	%
normal	49,326	98.7%
degradación	562	1.1%
failover	56	0.1%
retorno	56	0.1%

Métricas de evaluación (macro-average)

Modelo	Precision	Recall	F1	ROC-AUC
XGBoost	0.652	0.967	0.701	0.9992
Logistic Reg.	0.717	0.950	0.769	0.9984

Coeficiente de Variación: estabilidad de features entre folds

Estables (CV<0.3)	Moderadas (0.3-0.7)	Inestables (CV≥0.7)
41 / 62 features	3 / 62 features	18 / 62 features

Interpretación Recall alto + Precision moderada:
El modelo prioriza detectar la mayor cantidad posible de casos reales de estas clases, a costa de más falsos positivos.

lacnic

Resultados obtenidos

Clasificación de importancia por familia de features

Familia	XGBoost	LR (failover)	LR (retorno)
dns	29.9%	52.3%	45.6%
peer	7.5%	11.0%	8.7%
jitter	0.6%	5.6%	3.3%

Comparación final: pesos peer/dns/jitter

	peer	dns	jitter
Actual	0.270	0.500	0.230
XGBoost	0.197	0.788	0.016
LR failover	0.160	0.759	0.082
LR retorno	0.151	0.792	0.057

Pesos candidatos sugeridos

`peer=0.16  dns=0.78  jitter=0.07`

Interpretación: La consistencia de dirección entre XGBoost y LR es la conclusión más sólida y generalizable de este tipo de comparación. Recomendación: Validar contra históricos de meses reales.

lacnic

Conclusiones

Un aporte concreto a la comunidad de Internet

Este trabajo presenta una solución de software para la captura, almacenamiento y análisis de métricas de latencia, jitter y pérdida de paquetes, integrada con un motor inteligente de failover BGP capaz de automatizar políticas de enrutamiento basadas en dichas métricas.

La implementación, configuración y validación de la solución sobre una topología de laboratorio con Containerlab demuestran su viabilidad técnica y proporcionan una plataforma reproducible que facilita su adaptación y posterior evaluación en entornos operativos reales.

Conclusiones

Evidencia empírica del valor de IA-NetOps y la ingeniería de features

El análisis de datos históricos e ingeniería de features, posibilitó optimizar de manera empírica las ponderaciones de una función de scoring, reduciendo la dependencia de configuraciones manuales y proporcionando un criterio cuantitativo para la automatización de políticas de enrutamiento BGP.

La metodología adoptada representa un paso hacia redes interdominio más inteligentes, adaptativas y orientadas al desempeño, alineándose con las tendencias actuales de IA aplicada a NetOps y la automatización de redes.

Referencias

- Draft IETF “BGP Performance-aware Routing Mechanism”: https://www.ietf.org/archive/id/draft-ietf-idr-performance-routing-06.html
- Measuring Broadband America - Open Methodology: https://www.fcc.gov/oet/mba/measuring-broadband-america-open-methodology
- Cross-Validation and Model Evaluation in XGBoost: https://www.researchgate.net/publication/390141367_Cross-Validation_and_Model_Evaluation_in_XGBoost
- Measuring loss, latency and jitter based on application types to improve application routing decisions: https://www.tdcommons.org/cgi/viewcontent.cgi?article=4903&context=dpubs_series
- LACNIC Blog “La inteligencia artificial llega al corazón de los ISP”: https://blog.lacnic.net/inteligencia-artificial-isp/
- AI-Driven Networking: Machine Learning for BGP Optimization: https://eureka.patsnap.com/article/ai-driven-networking-machine-learning-for-bgp-optimization
- Generating Balanced and Realistic BGP Traffic for Machine Learning-Based Anomaly Detection: http://arantxa.ii.uam.es/~jlopezv/publicaciones/noms26.pdf
- Topologia, script BGP Failover y scripts entrenamiento del modelo: https://github.com/ernestosv73/bgp-failover.git
- What is z score? Meaning, Architecture, Examples, Use Cases, and How to Measure It: https://aiopsschool.com/blog/z-score/
- Detecting Anomalies in Network Latency Time Series: From Statistical Filters to Machine Learning: https://medium.com/thousandeyes-engineering/detecting-anomalies-in-network-latency-time-series-from-statistical-filters-to-machine-learning-b8098dc61109
- Cross-validation: evaluating estimator performance: https://sklearn.org/stable/modules/cross_validation.html
- https://www.lacnic.net/es/webinars/2026/del-despliegue-a-la-automatizacion-ripe-atlas-software-probes-apis-y-casos-de-uso

¡Gracias por su atención!

Ernesto Sánchez
esanchez@ucasal.edu.ar

Henri Alves de Godoy
henri.godoy@fca.unicamp.br

www.lacnic.net