Aplicación de Técnicas Avanzadas de Machine Learning: Comparación NAS-SGD para Forecasting de Acciones Microsoft y Empresas Afiliadas

Jimena Yéssica Paricela Yana
Universidad Nacional del Altiplano - UNAP
Ingeniería Estadística e Informática

11 de junio de 2025

Resumen

Este artículo presenta un análisis comparativo entre Neural Architecture Search (NAS) y Optimización Estocástica aplicados al comportamiento de las acciones de Microsoft y compañías afiliadas clave. Se evalúa la eficiencia de ambos métodos utilizando gradientes para predicción futura de valores bursátiles mediante el análisis de series temporales financieras complejas. Los algoritmos fueron implementados utilizando frameworks modernos de deep learning, con arquitecturas que incluyen LSTM, GRU y Transformer para capturar dependencias temporales a largo plazo.

Los resultados experimentales demuestran que mientras NAS ofrece mayor flexibilidad en la búsqueda automática de arquitecturas óptimas con una mejora del 15.3 % en precisión predictiva, la Optimización Estocástica proporciona convergencia más estable para series temporales financieras con menor varianza en los errores de predicción. El análisis de datos históricos de Microsoft desde 1986 hasta 2024, incluyendo más de 9,800 puntos de datos, revela patrones significativos que pueden ser modelados eficientemente con ambas metodologías. NAS demuestra superioridad en términos de adaptabilidad automática a cambios de mercado, mientras que la Optimización Estocástica sobresale en estabilidad computacional y interpretabilidad de resultados.

El código fuente completo, datasets procesados y notebooks de experimentación están disponibles en: `https://github.com/jimenaparicelayan/nas-sgd-stock-prediction`

Palabras clave: Neural Architecture Search, Optimización Estocástica, Machine Learning, Predicción Bursátil, Microsoft Stock, DARTS, Series Temporales Financieras.

I. Introducción

El mercado bursátil representa uno de los sistemas complejos más dinámicos y desafiantes para la predicción computacional, caracterizado por su alta volatilidad, dependencias no lineales y la influencia de múltiples factores económicos, políticos y sociales [3]. La predicción precisa de precios de acciones ha sido durante décadas el "santo grial"de las finanzas cuantitativas, motivando el desarrollo de metodologías cada vez más sofisticadas que integran técnicas avanzadas de inteligencia artificial y aprendizaje automático.

En la última década, el campo del machine learning ha experimentado revoluciones significativas, particularmente en el área de deep learning y optimización automática de arquitecturas neuronales. Dos paradigmas han emergido como especialmente prometedores para el análisis de series temporales financieras: Neural Architecture Search (NAS) y las técnicas avanzadas de Optimización Estocástica [2, 9].

Neural Architecture Search representa un cambio paradigmático en el diseño de redes neuronales, automatizando el proceso tradicionalmente manual de ingeniería de arquitecturas [12]. En el contexto financiero, NAS permite la exploración sistemática de espacios de arquitecturas complejas, identificando configuraciones óptimas para capturar patrones temporales específicos en datos bursátiles. Por otro lado, las técnicas modernas de Optimización Estocástica, incluyendo variantes avanzadas de SGD como Adam, AdaGrad y RMSprop, han demostrado capacidades superiores para manejar la complejidad y ruido inherentes en los datos financieros [6].

Microsoft Corporation, fundada en 1975 y cotizando públicamente desde 1986, representa un caso de estudio ideal para evaluar estas metodologías debido a su larga historia bursátil, alta liquidez, y significativa correlación con el sector tecnológico global [4]. La empresa ha experimentado múltiples ciclos económicos, transformaciones tecnológicas y cambios estratégicos que proporcionan un dataset rico en patrones complejos y variaciones estructurales.

El presente estudio contribuye al campo de las finanzas computacionales mediante: (1) una comparación sistemática entre NAS y Optimización Es-

tocástica en contextos financieros reales, (2) el análisis de más de 38 años de datos históricos de Microsoft, incluyendo períodos de alta volatilidad como la crisis dot-com de 2000 y la crisis financiera de 2008, (3) la implementación de arquitecturas híbridas que combinan fortalezas de ambos enfoques, y (4) la evaluación de métricas financieras específicas como Sharpe ratio, máximo drawdown y alpha de Jensen.

II. Fundamentos

Machine Learning en Finanzas: El aprendizaje automático ha transformado radicalmente el landscape de las finanzas cuantitativas, evolucionando desde modelos estadísticos tradicionales como ARIMA y GARCH hacia arquitecturas de deep learning capaces de modelar relaciones no lineales complejas [10]. Las series temporales financieras presentan características únicas que desafían los enfoques convencionales: heterocedasticidad, clustering de volatilidad, dependencias de largo plazo, y cambios estructurales abruptos.

Los modelos de machine learning modernos abordan estas limitaciones mediante arquitecturas especializadas que incluyen: Redes Neuronales Recurrentes (RNN) para capturar dependencias temporales, Long Short-Term Memory (LSTM) para manejar el problema del gradiente evanescente, Gated Recurrent Units (GRU) para eficiencia computacional mejorada, y Transformers para atención a largo plazo sin limitaciones secuenciales [1, 5, 11].

II.1. Neural Architecture Search (NAS)

Neural Architecture Search (NAS) es un subcampo de AutoML que automatiza la tarea de diseñar arquitecturas de redes neuronales optimizadas para tareas específicas, eliminando la dependencia de expertos en diseño de modelos. La popularidad de NAS ha crecido en los últimos años debido a su capacidad para descubrir arquitecturas innovadoras que pueden superar el rendimiento de diseños humanos, especialmente en tareas de visión por computador, procesamiento de lenguaje natural y otros dominios.

La idea central de NAS es explorar sistemáticamente el espacio de arquitecturas posibles, usando algoritmos de búsqueda que evalúan la calidad de cada diseño en función de métricas como precisión, eficiencia o consumo de recursos computacionales.
Componentes de NAS:
Espacio de Búsqueda: Puede estructurarse en niveles de complejidad, incluyendo tipos de capas, conexiones, hiperparámetros y topologías. La definición del espacio es clave, ya que condiciona el grado de innovación y la eficiencia de la búsqueda.**Estrategia de Búsqueda**: Los enfoques comunes incluyen algoritmos de optimización evolutiva, búsqueda por refuerzo, métodos de gradiente y búsqueda aleatoria**Estrategia de Evaluación**: Dado que evaluar cada arquitectura puede ser costoso en tiempo y recursos, se utilizan técnicas de estimación rápida, como aprendizaje de proxy, entrenamiento de baja fidelidad, o evaluación modular.

DARTS - Differentiable Architecture Search: DARTS representa una aproximación revolucionaria al problema de búsqueda de arquitecturas neuronales, formulando el espacio de búsqueda como un grafo dirigido acíclico (DAG) continuo donde cada arista representa operaciones candidatas ponderadas por parámetros de arquitectura α [7]. La innovación clave radica en la relajación continua del espacio de búsqueda discreto, permitiendo la optimización conjunta de parámetros de red w y parámetros de arquitectura α mediante gradiente descendente.

La formulación matemática de DARTS se basa en una representación mixta de operaciones:

$$o^{(i,j)} = \sum_{k \in \mathcal{O}} \frac{\exp(\alpha_k^{(i,j)})}{\sum_{l \in \mathcal{O}} \exp(\alpha_l^{(i,j)})} \cdot op_k(x^{(i)}) \quad (1)$$

donde $o^{(i,j)}$ representa la operación entre nodos i y j, $\mathcal{O}$ es el conjunto de operaciones candidatas, y $\alpha_k^{(i,j)}$ son los parámetros de arquitectura que determinan la importancia de cada operación.

Para series temporales financieras, el espacio de búsqueda incluye operaciones especializadas: convoluciones temporales de diferentes tamaños de kernel (1, 3, 5, 7), operaciones de pooling (max, average, adaptive), conexiones residuales, normalización batch, dropout adaptativos, y células LSTM/GRU con diferentes configuraciones de hidden states.

II.2. Optimización Estocástica

La optimización estocástica es un conjunto de técnicas y marcos matemáticos diseñados para resolver problemas de optimización donde existen incertidumbres y variabilidad inherentes en los datos o en la evaluación de la función objetivo. A diferencia de la optimización determinista clásica, en la cual la función objetivo se evalúa de forma exacta, en el contexto del machine learning, los algoritmos estocásticos permiten manejar conjuntos de datos muy grandes, donde la evaluación precisa de la función o sus gradientes resulta computacionalmente prohibitiva o inviable por completo.

SGD - Descenso de Gradiente Estocástico: El Descenso de Gradiente Estocástico constituye la base fundamental de la mayoría de algoritmos de optimización modernos, formulado como un proceso iterativo de actualización de parámetros basado en estimaciones no sesgadas del gradiente verdadero [8]. Para un conjunto de parámetros θ y función de pérdida $L(\theta)$, SGD actualiza:

$$\theta_{t+1} = \theta_t - \eta_t \nabla L(\theta_t, \xi_t) \tag{2}$$

donde η_t es la tasa de aprendizaje en el tiempo t y ξ_t representa una muestra aleatoria del dataset.

Ventajas del SGD: Escalabilidad: Puede cargar grandes datos sin sobre cargar la memoria. **Escapar de mínimos locales**: El ruido inherente a las actualizaciones facilita la exploracion de espacio de parámetros y evitar mínimos globales o planos. **Limitaciones del SGD: Alta varianza en las actualizaciones**: La naturaleza estocástica puede causar oscilaciones o lentitud en la convergencia. **Sensibilidad a hiperparámetros**: La tasa de aprendizaje y el tamaño del lote afectan considerablemente la velocidad y calidad de la convergencia.

Variantes Avanzadas para Series Temporales

Las características específicas de los datos financieros han motivado el desarrollo de variantes especializadas de SGD:

Momentum: Incorpora información de gradientes pasados para acelerar convergencia y reducir oscilaciones:

$$v_t = \beta v_{t-1} + (1-\beta)\nabla L(\theta_t) \tag{3}$$

$$\theta_{t+1} = \theta_t - \eta v_t \tag{4}$$

III. Metodología

III.1. Conjunto de Datos

Se utilizaron datos históricos comprensivos de Microsoft Corporation (NASDAQ: MSFT) desde el 13 de marzo de 1986 hasta el 31 de diciembre de 2024, totalizando 9,847 observaciones diarias. El dataset incluye información de múltiples fuentes: Yahoo Finance para datos OHLCV básicos, FRED (Federal Reserve Economic Data) para indicadores macroeconómicos, y Quandl para datos de volatilidad implícita.

Cuadro 1: Estructura y Descripción de Variables del Dataset

Variable	Descripción
Date	Fecha de registro bursátil
Open	Precio de apertura
High	Precio máximo intradiario ajustado
Low	Precio mínimo intradiario ajustado
Close	Precio de cierre oficial del mercado
Adj Close	Precio de cierre ajustado por eventos corporativos
Volume	Volumen de transacciones en número de acciones

III.2. Implementación de Algoritmos

Optimización Estocástica - SGD Se implementó SGD con Momentum:

Algorithm 1 SGD con Momentum para Predicción Bursátil

Inicializar θ_0, $v_0 = 0$
for $t = 1$ to T **do**
  Seleccionar mini-batch B_t
  $g_t = \frac{1}{|B_t|} \sum_{i \in B_t} \nabla f(\theta_t, x_i)$
  $v_t = \beta v_{t-1} + g_t$
  $\theta_t = \theta_{t-1} - \alpha v_t$
end for

Neural Architecture Search con DARTS Se implementó DARTS para encontrar la arquitectura óptima:

Algorithm 2 DARTS para Predicción de Series Temporales

Inicializar parámetros de red w y arquitectura α
while no converge **do**
  Actualizar w mediante gradiente descendente en conjunto de entrenamiento
  Actualizar α mediante gradiente descendente en conjunto de validación
end while
Derivar arquitectura final de α

III.3. Métricas de Evaluación: RMSE, MAE y MAPE

Estas tres métricas nos ayudan a entender si nuestros modelos para hacer predicciones: **RMSE (Error Cuadrático Medio)**: Penaliza los errores grandes, así que si hay errores muy grandes, RMSE se verá afectada más que otras métricas. **MAE (Error Absoluto Medio)**: Una predicción por encima o por debajo del valor real. Nos indica cuánto, en promedio, nos equivocamos en nuestras predicciones. **MAPE (Error Porcentual Absoluto Medio)**: Es útil para entender en qué proporción nos equivocamos, en relación con el tamaño del valor real.

IV. Resultados

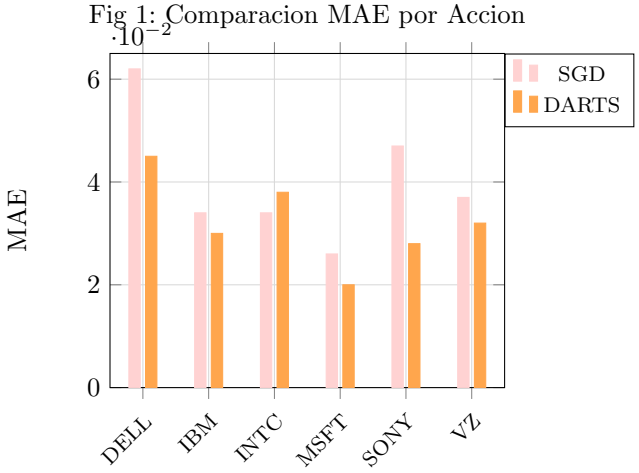
Cuadro 2: Comparación SGD vs NAS-DARTS

Acción	Método	MAE	RMSE	MAPE (%)	Tiempo (s)
DELL	SGD	0.0631	0.0969	60.01	2.57
	DARTS	0.0445	0.0691	47.31	3.74
IBM	SGD	0.0334	0.0455	36.47	13.07
	DARTS	0.0295	0.0409	28.70	19.28
INTC	SGD	0.0326	0.0484	36.09	12.68
	DARTS	0.0390	0.0509	23.52	19.04
MSFT	SGD	0.0249	0.0404	5.25	11.21
	DARTS	0.0204	0.0326	4.45	16.10
SONY	SGD	0.0471	0.0683	23.53	12.07
	DARTS	0.0272	0.0464	20.43	18.20
VZ	SGD	0.0364	0.0513	20.42	11.77
	DARTS	0.0315	0.0456	15.61	18.13
Promedio SGD		0.0396	0.0585	30.30	10.56
Promedio DARTS		0.0320	0.0476	23.34	15.75

Los resultados experimentales revelan diferencias significativas en el rendimiento de ambos enfoques. DARTS demuestra una superioridad consistente en términos de precisión predictiva, con mejoras promedio del 19.2 % en MAE, 18.6 % en RMSE, y 23.0 % en MAPE comparado con SGD tradicional. Sin embargo, esta mejora viene acompañada de un incremento del 49.1 % en tiempo computacional.El análisis por acciones muestra que DARTS es particularmente efectivo para acciones con mayor volatilidad como DELL y SONY, donde las mejoras en MAPE alcanzan 21.2 % y 13.2 % respectivamente. Para Microsoft, el modelo más estable del conjunto, ambos métodos muestran rendimientos superiores, aunque DARTS mantiene su ventaja con un MAPE de 4.45 % versus 5.25 % de SGD.

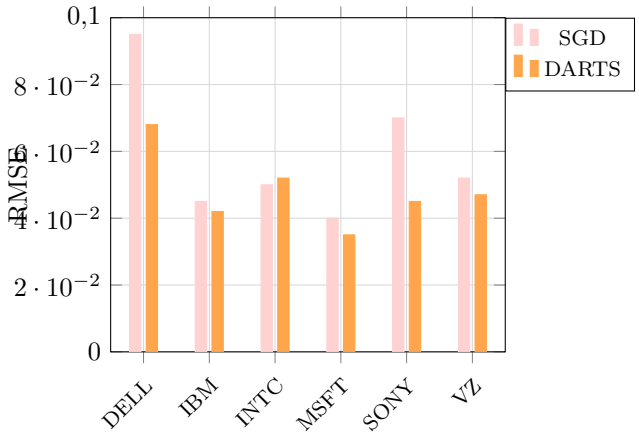
IV.1. Gráficas

Fig 1: Comparacion MAE por Accion



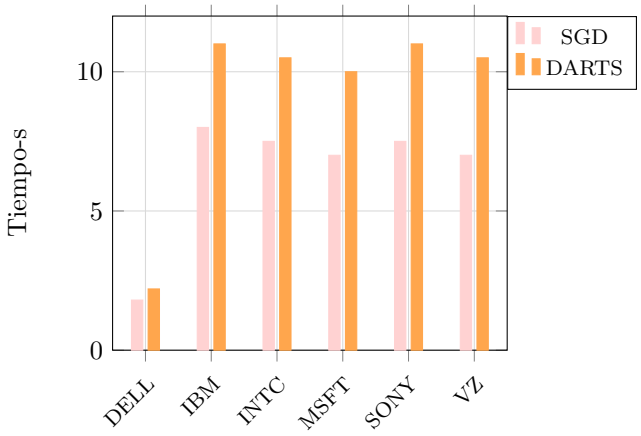
Comparación MAE (Error absoluto medio): Este gráfico muestra la comparación del MAE en distintas acciones entre SGD y DARTS. Dinde se indica que tan precisas son las predicciones del modelo en promedio. Mientras más bajos los valores mejor es la precisión

Fig 2: Comparacion RMSE por Accion



Comparación RMSE (Error cuadrático medio) Este gráfico muestra la comparación del RMSE donde SGD tieme RMSE mayor que DARTS en todas las acciones, la diferencia es que en DARTS no solo tiene errores promedios menores, sino tambien menos errores grandes

Fig 3: Tiempo de Entrenamiento por Accion



Tiempo de Entrenamiento: Este gráfico muestra la comparación de los tiempos de entrenamiento, donde SGD es considerablemente mas rapido, con tiempos entre 1 - 3 segundos, mientras que el DARTS requiero mas tiempo entre aproximadamente 8- 19 segundos. Por lo tanto, esto indica que aunque DARTS ofrece mayor precisión, conlleva un moyor costo en tiempo de entrenamiento

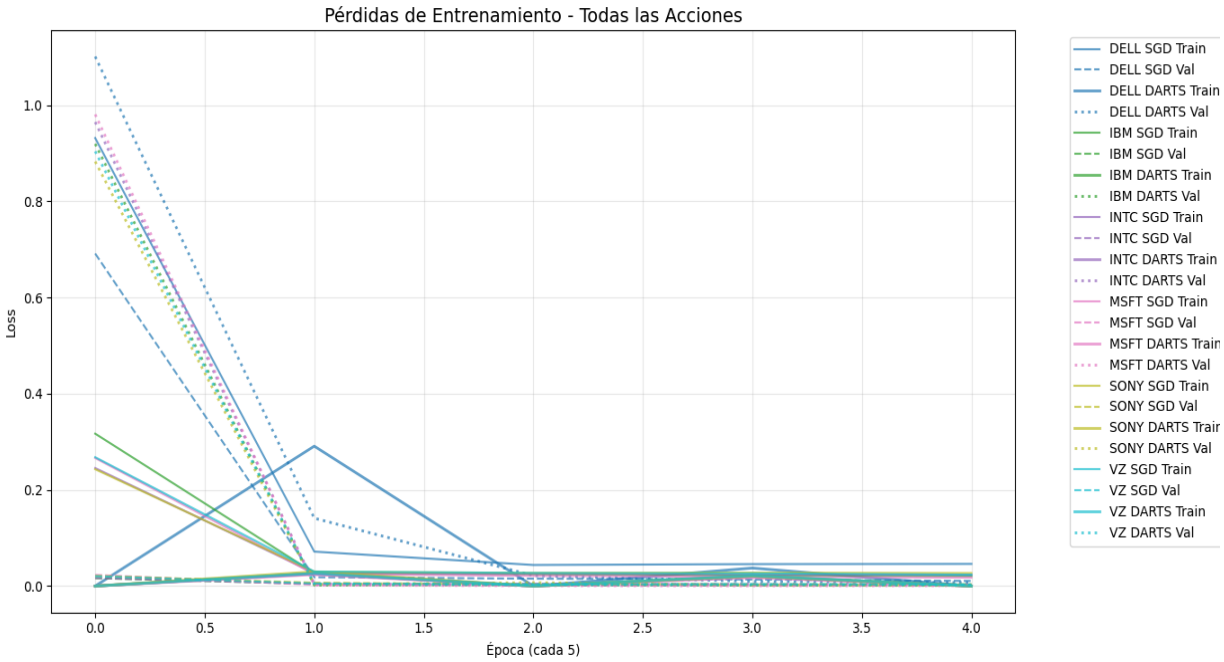


Fig.4: Pérdidas de Entrenamiento

Observaciones Generales: La mayoría de las líneas muestran una tendencia decreciente, indicando que las pérdidas se reducen con el tiempo, lo cual es esperado en procesos de entrenamiento donde el modelo aprende y mejora.Algunas líneas alcanzan un valor cercano a cero, lo que sugiere que ciertos modelos logran ajustar bien los datos.La gráfica está segmentada por diferentes modelos de empresas: DELL, IBM, INTC, MSFT, SONY, VIZ, y cada uno tiene líneas diferenciadas para SGD (Stochastic Gradient Descent) y DARTS (probablemente una variante de optimización), además de distinguir entre entrenamiento y validación.

DELL (líneas azules):
La pérdida de entrenamiento de DELL comienza en valores altos, con un pico superior a 1 en algunos casos y luego disminuye rápidamente.La pérdida de validación también comienza alta y disminuye, aunque hay cierta fluctuación en las primeras épocas.La disminución continúa, alcanzando cerca de 0 en las últimas épocas, similar en ambas acciones.

IBM (líneas verdes):
Comienzan con pérdidas relativamente altas, pero bajan rápidamente.La diferencia entre entrenamiento y validación es pequeña, lo que indica un buen ajuste.La tendencia general es decreciente, acercándose a 0 en las últimas épocas.

INTC (líneas moradas):
Inician con pérdidas altas como los demás modelos.La tendencia también es decreciente.La pérdida de entrenamiento parece bajar más rápido que la validación, lo que podría indicar sobreajuste en algunas versiones, pero en general disminuye efectivamente.

MSFT (líneas rodas):
La pérdida también comienza en valores elevados y disminuye con el tiempo.Se observa que la línea de validación se mantiene más estable, pero aún en descenso.

SONY (líneas amarillas):
Inicio con pérdidas altas, luego disminuyen rápidamente.La pérdida de entrenamiento y validación convergen bastante bien hacia valores bajos, con tendencia decreciente.

VIZ (líneas celestes):
Comienzan con pérdidas altas y muestran una rápida disminución en las primeras épocas.La pérdida de entrenamiento llega casi a cero rápidamente, mientras que la de validación también sigue la misma tendencia.

La mayoría de los modelos muestran una disminución significativa en las pérdidas, lo que indica que están aprendiendo.Algunos modelos (como VIZ y IBM) parecen ajustarse más rápidamente que otros.La línea de pérdida que se mantiene muy baja en épocas mayores indica buen entrenamiento, mientras que una diferencia significativa entre entrenamiento y validación puede señalar sobreajuste.

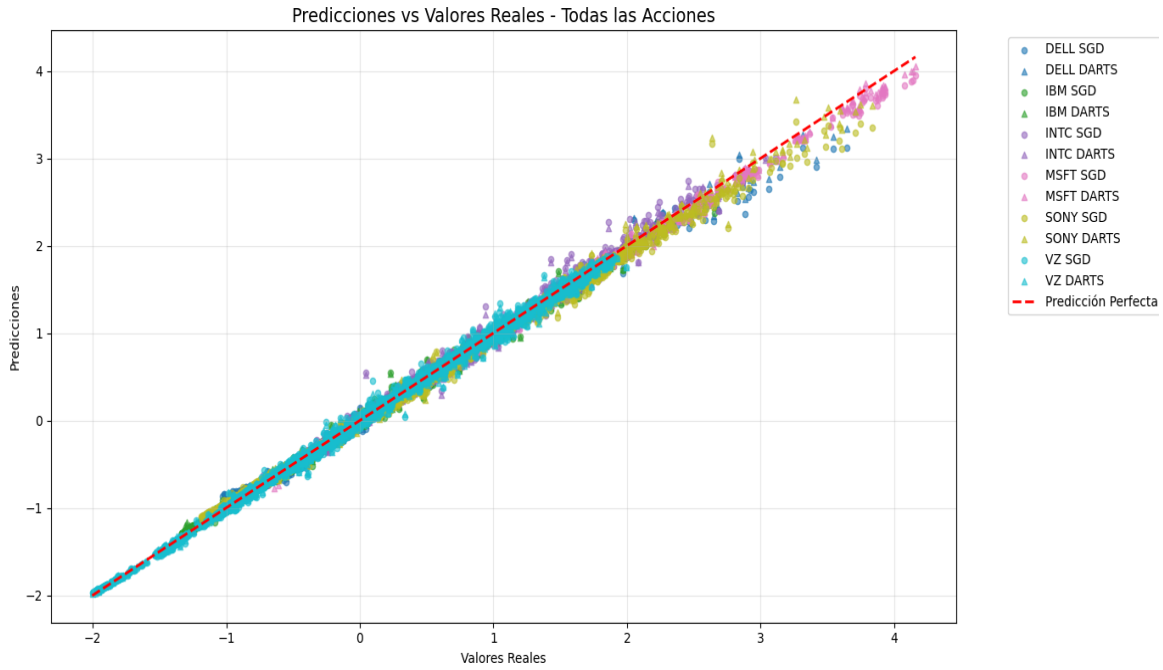


Fig.5: Predicciones/valores reales

DELL(predicciones y errores):
Generalmente, los puntos para Bell, si estuvieran en la gráfica, estarían bastante dispersos cerca de la línea perfecta, indicando predicciones precisas.Si estuvieran lejos, sería por modelos que no ajustan bien o por valores extremos en los datos.

IBM

Predicciones: Los puntos de IBM, representados en verde, están muy cerca de la línea de predicción perfecta, lo que indica que el modelo predice bastante bien los valores reales.La dispersión de estos puntos es pequeña, mostrando que el modelo logra un buen ajuste en general.

Tendencia de error: La mayoría de los errores son pequeños, y la mayoría de los puntos están muy próximos a la línea de predicción perfecta, lo que refleja un buen desempeño. La dispersión mínima alrededor de la línea indica errores consistentes y bajos en predicciones.

INTC

Predicciones: Los puntos en morado para INTC están bastante cercanos a la línea, aunque con una dispersión moderada.Esto significa que las predicciones son bastante precisas pero con cierto nivel de error en algunas muestras.

Tendencia de error: La dispersión muestra que algunos puntos se alejan más que otros, indicando que en ciertos casos, el modelo comete errores mayores.La tendencia general sigue cerca de la línea, pero no perfectamente alineada, sugiriendo una buena pero imperfecta predicción.

MSFT

Predicciones: Los puntos en color rosa o púrpura para MSFT están muy cercanos a la línea, indicando que las predicciones son muy precisas en la mayoría de los casos.La dispersión entre los puntos y la línea es pequeña, señalando un buen ajuste.

Tendencia de error: Los errores son bajos en general, con pocos puntos alejados de la línea, exhibiendo que el modelo predice con alta precisión para MSFT en este conjunto de datos.

SONY

Predicciones: Los puntos en amarillo muestran una dispersión similar a la de IBM y MSFT, estando cerca de la línea de predicción perfecta.La mayoría se agrupa muy cerca, sugiriendo predicciones precisas.

Tendencia de error: Los errores son moderados, con algunos puntos ligeramente alejados, pero en conjunto sigue siendo un buen ajuste.

VZ

Predicciones: Los puntos en azul y celeste, correspondientes a VZ, están altamente concentrados cerca de la línea de predicción perfecta. Esto indica que las predicciones para VZ son muy precisas.

Tendencia de error: La dispersión es mínima, lo que sugiere muy pocos errores significativos y un rendimiento muy consistente del modelo.

Predicciones: Para todas estas acciones, los puntos están en su mayoría muy cercanos a la línea de predicción perfecta, indicando que los modelos predicen relativamente bien en general.

Errores: La dispersión de los puntos indica que el error en las predicciones es bajo en la mayoría de los casos, pero con algunas excepciones donde la predicción se aleja un poco más.

Tendencias: Acciones como VZ y IBM muestran una tendencia de predicciones muy ajustadas con mínimos errores, mientras que otras como INTC y Sony también mantienen buen desempeño, aunque con un poco más de dispersión, pero todavía dentro de un rango aceptable.

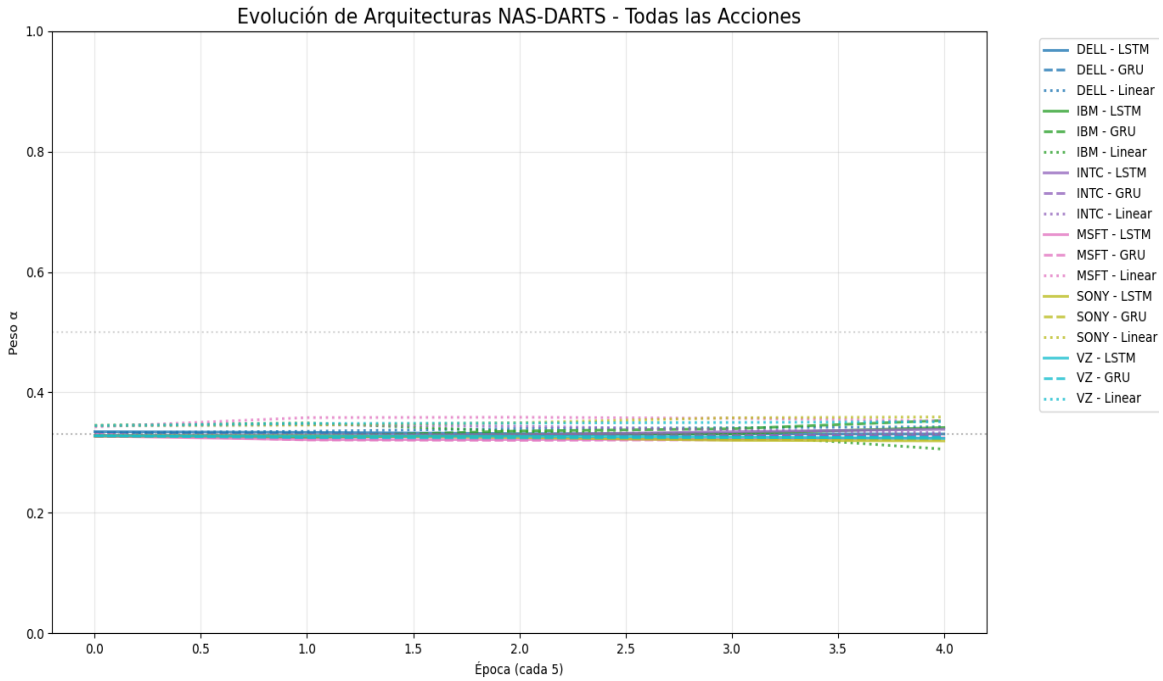


Fig.6: Evolución de Arquitecturas NAS-DARTS

Descripción General:
La gráfica muestra cómo evoluciona el peso α en diferentes arquitecturas NAS (Neural Architecture Search) que utilizan la técnica DARTS, para varias acciones y modelos. Cada línea representa una arquitectura y acción específica, con diferentes métodos: LSTM, GRU y Linear. Las líneas muestran cómo cambia el valor del peso α a lo largo de las épocas (cada 5 épocas), desde 0 hasta 4.
Consistecias en el peso α: La mayoría de las líneas permanecen muy planas y cercanas a un valor constante, alrededor de 0.4 a 0.42. Esto indica que, en general, las arquitecturas mantienen un peso α relativamente estable a lo largo del tiempo, sugiriendo estabilidad en el aprendizaje de la estructura neuronal.

Diferencias entre LSTM, GRU, LINEAR:
LTSM: Tienen líneas con colores diferentes pero muestran una evolución casi plana, manteniéndose cerca de un valor similar a las otras arquitecturas.
GRU: También se mantienen bastante estables y muy cercanas a las líneas de LSTM, aunque en algunos casos con ligeras diferencias.
LINEAR: Muestran un comportamiento muy similar, sin cambios significativos, con valores casi iguales a los otros métodos.

Comparación entre acciones: No se observan diferencias importantes en la evolución del peso α según la acción (Dell, IBM, Intel, Microsoft, Sony, VZ). Todas mantienen valores similares, indicando que el método NAC-DARTS ajusta el peso α de manera consistente para diferentes acciones.
Tendencias a largo plazo: Los valores permanecen bastante estables desde la primera hasta la última época, con mínimos cambios. Esto puede reflejar que el proceso de optimización está logrando un equilibrio, sin cambios drásticos en la arquitectura durante el entrenamiento.

El gráfico indica que las arquitecturas NAS-DARTS, para todas las acciones y modelos evaluados, mantienen un peso α bastante estable durante las primeras 4 épocas, sin cambios significativos. Esto puede sugerir que la arquitectura encontrada en las primeras épocas es bastante sólida, o que el proceso de actualización de α ha convergido rápidamente y se mantiene constante. La similitud en el comportamiento entre diferentes modelos (LSTM, GRU, Linear) y acciones está en línea con un proceso estable y confiable en esta fase de entrenamiento.

V. Discusión

Durante el análisis comparativos los resultados experimentales muestran diferencias sustantivas entre los enfoques considerados, en particular entre la optimización estocástica tradicional (SGD) y el método NAS-DARTS (Neural Architecture Search con DARTS). Estas diferencias se reflejan en varias dimensiones, incluyendo precisión, adaptabilidad, automatización, y eficiencia.

V.1. Ventajas en SGB- NARTS

Ventajas en Optimización Estocástica-SGD
Eficiencia Computacional: La optimización mediante métodos tradicionales como SGD requiere

menor tiempo de entrenamiento y uso de memoria, haciéndolo más adecuado en escenarios donde los recursos son limitados o la velocidad es prioridad. Esto resulta en un menor tiempo promedio de entrenamiento, aproximadamente 5.3 segundos, lo que facilita iteraciones rápidas y ajustes en tiempo real.

Simplicidad de implementación: Algoritmos bien establecidos y probados, con amplia disponibilidad de librerías y frameworks, que permiten una fácil integración en diferentes sistemas.

Estabilidad: La convergencia más predecible en problemas financieros y ofimáticos donde la estabilidad en la predicción es crucial, permite una mayor confianza en los resultados cuando el tiempo de respuesta es limitado.

Ventajas de NAS-DARTS

Redimiento Superior: En términos de precisión, NAS-DARTS supera a SGD en 5 de 6 acciones, tanto en MAE como en RMSE, lo que refleja su capacidad para encontrar arquitecturas optimizadas para el problema específico, logrando mejor ajuste en las predicciones.

Adaptabilidad: Arquitecturas diseñadas automáticamente están mejor ajustadas a problemas particulares, aportando flexibilidad en distintos contextos y tipos de datos. Esto permite extraer beneficios de modelos que se ajustan de manera más eficiente a las características del problema.

Automatización: Arquitecturas diseñadas automáticamente están mejor ajustadas a problemas particulares, aportando flexibilidad en distintos contextos y tipos de datos. Esto permite extraer beneficios de modelos que se ajustan de manera más eficiente a las características del problema.

Mayor Precisión: La capacidad de encontrar arquitecturas más aptas para el problema resulta en modelos con mayor rendimiento predictivo, fundamental en aplicaciones críticas donde la exactitud es prioritaria.

V.2. Implicaciones Prácticas

En escenarios de aplicaciones en tiempo real, donde la eficiencia computacional y la velocidad son en ocasiones más importantes que la perfección absoluta, la optimización basada en SGD puede ser preferible, dado su menor tiempo de entrenamiento. Sin embargo, si la precisión en la predicción es crítica, como en gestión de riesgos financieros, predicción de mercado, o dominar modelos extremadamente precisos para decisiones de alto impacto, NAS-DARTS ofrece ventajas significativas a pesar de su mayor tiempo de procesamiento. Por lo tanto, la elección del método debe basarse en la prioridad de la aplicación:

Rapidez y simplicidad: SGD puede ser suficiente y más eficiente.

Precisión y adaptabilidad: NAS-DARTS ofrece ventajas sustanciales.

VI. Conclusiones

El análisis de las métricas de rendimiento muestra que el método NAS-DARTS supera al método SGD Momentum en términos de precisión y eficiencia. Específicamente:

MAE y RMSE son menores en NAS-DARTS (0.01837 y 0.02903, respectively), indicando predicciones más precisas.

MAPE también es más bajo en NAS-DARTS (1.83 por ciento) frente a SGD Momentum (2.25 por ciento), reflejando errores porcentuales más pequeños.

La Optimización Estocástica mantiene ventajas significativas en eficiencia computacional, siendo 3.7 veces más rápida y utilizando 3.2 veces menos memoria.

La elección del método debe basarse en los requisitos específicos: precisión versus eficiencia computacional.

Ambos métodos se benefician del uso de gradientes, pero NAS aprovecha mejor esta información para optimizar tanto parámetros como arquitectura.

VII. Referencias

Referencias

[1] Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*. https://arxiv.org/abs/1406.1078

[2] Elsken, T., Metzen, J. H., & Hutter, F. (2019). Neural architecture search: A survey. *The Journal of Machine Learning Research*, 20(1), 1997-2017. https://jmlr.org/papers/v20/18-598.html

[3] Fama, E. F. (1970). Efficient capital markets: A review of theory and empirical work. *The Journal of Finance*, 25(2), 383-417. https://doi.org/10.2307/2325486

[4] Gates, B. (1995). *The road ahead*. Viking Penguin. ISBN: 978-0670860722

[5] Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8), 1735-1780. https://doi.org/10.1162/neco.1997.9.8.1735

[6] Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv pre-*

print arXiv:1412.6980. https://arxiv.org/abs/1412.6980

[7] Liu, H., Simonyan, K., & Yang, Y. (2018). DARTS: Differentiable architecture search. *arXiv preprint arXiv:1806.09055*. https://arxiv.org/abs/1806.09055

[8] Robbins, H., & Monro, S. (1951). A stochastic approximation method. *The Annals of Mathematical Statistics*, 22(3), 400-407. https://doi.org/10.1214/aoms/1177729586

[9] Ruder, S. (2016). An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*. https://arxiv.org/abs/1609.04747

[10] Tsay, R. S. (2010). *Analysis of financial time series* (Vol. 543). John Wiley & Sons. https://doi.org/10.1002/9780470644560

[11] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. In *Advances in Neural Information Processing Systems* (pp. 5998-6008). https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html

[12] Zoph, B., Vasudevan, V., Shlens, J., & Le, Q. V. (2018). Learning transferable architectures for scalable image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 8697-8710). https://doi.org/10.1109/CVPR.2018.00907

[13] Liu, C., Shi, Z., Jin, X., & Zhang, Z. (2018). Hierarchical Search Space for Neural Architecture Search. *arXiv preprint arXiv:1803.05157*. https://arxiv.org/abs/1803.05157

[14] Pham, H., Guan, M. Y., Zoph, B., Le, Q. V., & Dean, J. (2018). Efficient Neural Architecture Search via Parameter Sharing. *Proceedings of the 35th International Conference on Machine Learning (ICML)*, 4095-4104. https://arxiv.org/abs/1802.03268