COMPUTACIÓN E INFORMÁTICA
Recibido 2 Sep. 2025 *ReCIBE, Año 14 No. 3, Dic. 2025*
Aceptado 25 Nov. 2025

Entrenamiento de una red neuronal de seguimiento de señales: un enfoque metaheurístico mediante optimización por enjambre de partículas

Training a Signal-Tracking Neural Network: A Metaheuristic Approach Using Particle Swarm Optimization

Alma Aide Sánchez Ramírez

Alvaro Anzueto Ríos[1]
aanzuetor@ipn.mx

[1] Unidad Profesional Interdisciplinaria en Ingeniería y Tecnologías Avanzadas del **IPN**.

RESUMEN
A medida que los sistemas complejos y no lineales aumentan su cantidad de datos; como en la actualidad con el uso de aplicaciones que registran posiciones y otras actividades que desarrollamos diariamente, los sistemas inteligentes deben aprender a partir de grandes volúmenes de datos y adaptarse a entornos cambiantes. En muchos casos la optimización basada en métodos clásicos resulta insuficiente, por lo que los algoritmos metaheurísticos se han vuelto una herramienta para la optimización de modelos complejos, con aplicaciones que van desde el diseño de redes neuronales, sistemas de control y el ajuste de modelos predictivos. Por esta razón, resulta relevante evaluar el desempeño de estos algoritmos en diferentes campos. En este trabajo se aborda el entrenamiento de redes neuronales, para lo cual se presenta una comparación entre dos métodos de entrenamiento de tipo forward propagation para la para la tarea de seguimiento de señales temporales, con el objetivo de ajustar sus parámetros sinápticos y que la neurona intente aproximarse a una señal de referencia. Como alternativa al entrenamiento basado en el algoritmo de Backpropagation, que utiliza el descenso del gradiente para minimizar el error cuadrático medio (MSE), se implementa el algoritmo de Optimización por Enjambre de Partículas (PSO), una técnica de búsqueda global que opera sin necesidad de derivadas y realiza una exploración simultánea del espacio de solución. Para evaluar y comparar el desempeño de ambos métodos, se realizaron 30 ejecuciones para cada algoritmo. El rendimiento se midió con el MSE final, la tasa de éxito (definida como un MSE inferior a 0.005) y el tiempo de cómputo total. Adicionalmente, se analiza la superficie de solución generada por la red neuronal, representando la función de costo en términos de dos pesos seleccionados, lo cual permite ilustrar la presencia de múltiples óptimos locales y su impacto en la eficacia de cada algoritmo. Los resultados obtenidos demuestran que, aunque Backpropagation tiende a converger rápidamente, es susceptible a mínimos locales, mientras que PSO mostró mayor estabilidad frente a la topología no convexa de la superficie de solución, logrando convergencia hacia regiones de bajo error en la mayoría de los casos evaluados. Las observaciones realizadas permiten discutir las ventajas y limitaciones de cada enfoque en contextos donde las funciones objetivo son no diferenciables

Palabras clave: Redes neuronales artificiales, optimización de enjambre de partículas, Backpropagation, Seguimiento de señales, aproximación de funciones, optimización metaheurística, superficie de solución**.**

ABSTRACT
As complex, nonlinear systems generate increasing amounts of data—for instance, through modern applications that continuously record positions and other daily activities—intelligent systems must learn from large data volumes and adapt to changing environments. In many cases, optimization based on classical methods becomes insufficient, so metaheuristic algorithms have emerged as powerful tools for optimizing complex models, with applications ranging from neural network design and control systems to the fine-tuning of predictive models. For this reason, it is important to assess the performance of these algorithms in different domains. In this work, we focus on the training of neural networks and present a comparison between two forward-propagation training methods for the task of tracking temporal signals, with the goal of adjusting their synaptic parameters so that the neuron attempts to approximate a reference signal. As an alternative to training based on the Backpropagation algorithm, which uses gradient descent to minimize the mean squared error (MSE), we implement the Particle Swarm Optimization (PSO) algorithm, a global search technique that operates without requiring derivatives and performs a simultaneous exploration of the solution space. To evaluate and compare the performance of both methods, 30 runs were carried out for each algorithm. Performance was measured using the final MSE, the success rate (defined as an MSE below 0.005), and the total computation time. Additionally, the solution surface generated by

the neural network is analyzed by representing the cost function in terms of two selected weights, which makes it possible to illustrate the presence of multiple local optima and their impact on the effectiveness of each algorithm. The results show that, although Backpropagation tends to converge quickly, it is susceptible to local minima, whereas PSO exhibited greater stability with respect to the non-convex topology of the solution surface, achieving convergence toward low-error regions in most of the evaluated cases. These observations support a discussion of the advantages and limitations of each approach in contexts where the objective functions are non-differentiable.

Keywords: Artificial Neural Networks, Particle Swarm Optimization, Backpropagation, Signal Tracking, Function Approximation, Metaheuristic Optimization, Error Landscape.

1. INTRODUCCION

La capacidad de aproximar funciones y seguir señales dinámicas ha representado un desafío central en áreas tan diversas como la ingeniería de control, las telecomunicaciones y el procesamiento de señales biomédicas (Ljung, 1999). En este contexto, las Redes Neuronales Artificiales (RNA) han demostrado ser herramientas, capaces de capturar relaciones complejas y no lineales a partir de datos experimentales (Goodfellow et al., 2016; Hagan et al., 2014).

El enfoque más utilizado para entrenar este tipo de redes es el algoritmo de Backpropagation, popularizado por Rumelhart y colaboradores en 1986. Este método, basado en el descenso de gradiente, ajusta los pesos de la red minimizando iterativamente el error. Sin embargo, su eficacia depende de la naturaleza de la superficie de solución, una superficie multidimensional definida por los parámetros de la red. En superficies de solución no convexas, que son comunes en problemas no triviales, los métodos basados en gradiente corren el riesgo de converger a mínimos locales, como bien señalan investigaciones previas (Choromanska et al., 2015)), lo que resulta en un rendimiento subóptimo del modelo.

Frente a esta limitación, han surgido enfoques de optimización alternativos, entre las cuales destacan los algoritmos metaheurísticos, que han sido aplicados con éxito al entrenamiento de redes neuronales (Yudong & Lenan, 2015), ofrecen una estrategia de búsqueda global que no depende de la información del gradiente. Entre ellos, la Optimización por Enjambre de Partículas (PSO), desarrollada por Kennedy y Eberhart (1995), opera con una población de soluciones candidatas (partículas) que exploran el espacio de búsqueda de manera colaborativa. Cada partícula ajusta su trayectoria combinando su propia experiencia con la del mejor individuo del enjambre, equilibrando así la exploración de nuevas áreas del espacio de búsqueda con la explotación de regiones prometedoras. Esta naturaleza estocástica y poblacional le permite escapar de mínimos locales y alcanzar soluciones más robustas (Engelbrecht, 2007).

Este trabajo aborda el entrenamiento de una red neuronal con el algoritmo de PSO para una tarea de seguimiento de señales, con el objetivo de responder las siguientes preguntas de investigación:

1. ¿Puede el algoritmo PSO entrenar de manera efectiva la arquitectura de red neuronal propuesta para esta aplicación?
2. ¿Cómo se compara su rendimiento, en términos de precisión, robustez y velocidad de convergencia, con el enfoque clásico de Backpropagation?

Para responder estas preguntas, se llevó a cabo un análisis comparativo, en una red neuronal con arquitectura 1-2-1 fue entrenada con ambos algoritmos, evaluando su desempeño a través de múltiples ejecuciones considerando tanto métricas cuantitativas como el comportamiento dinámico de los algoritmos.

La estructura del artículo es la siguiente: la sección 2 describe la arquitectura de la red, los algoritmos y la configuración experimental. La sección 3 presenta los hallazgos empíricos, mientras que la sección 4 se dedica a la interpretación y discusión de los resultados. El artículo concluye en la sección 5 con un resumen de las conclusiones del estudio.

2. METODOLOGÍA

En esta sección se describe la estrategia experimental diseñada para comparar los algoritmos de entrenamiento. Se presenta la arquitectura de la red neuronal, la formulación del problema de optimización, los algoritmos de entrenamiento empleados y los parámetros que guiaron la configuración de las simulaciones.

2.1 Arquitectura de la Red Neuronal

Para el desarrollo del estudio se emplea una red neuronal artificial de tipo prealimentada (feed-forward) con una arquitectura 1-2-1, compuesta por una capa de entrada, una capa oculta y una capa de salida. La capa de entrada consta de una única neurona que recibe el valor escalar *p*. La capa oculta contiene dos neuronas, y la capa de salida una sola neurona que produce la aproximación final, *a*. Esta arquitectura, aunque simple, es capaz de aproximar cualquier funcion continua con un grado arbitrario de precisión, un principio conocido como el teorema de aproximación universal (Hornik et al., 1989).

La función de activación para las neuronas de la capa oculta es la sigmoide logística (logsig), definida como:

$$f^1(n) = \frac{1}{1+e^{-n}} \qquad (1)$$

donde *n* es la entrada neta a la neurona. Esta función no lineal es fundamental para la capacidad de la red de aprender relaciones complejas (Hagan et al., 2014)

La capa de salida utiliza una función de activación lineal (*purelin*), que simplemente trasmite su entrada neta sin alteración:

$$f^2(n) = n \qquad (2)$$

Esta elección es estándar para tareas de regresión donde la salida no esta acotada a un rango especifico. La salida completa de la red, $a(p)$, es una composición de estas funciones, tal como se describe en Hagan et al. (2014, Capítulo 11).

2.2 Formulación del Problema de Optimización

El objetivo es entrenar la red para que su salida $a(p)$ siga una señal sinusoidal de referencia, $g(p)$, definida como:

$$g(p) = 1 + \sin\left(\frac{\pi}{4}p\right) \qquad (3)$$

El proceso de entrenamiento se formula como un problema de optimización donde se busca el conjunto de parámetros de la red que minimice el error de aproximación. Todos los parámetros entrenables de la red (4 pesos y 3 polarizaciones) se consolidan en un único vector de parámetros de 7 dimensiones, θ:

$$\boldsymbol{\theta} = [w^1_{1,1}, w^1_{2,1}, b^1_1, b^1_2, w^2_{1,1}, w^2_{1,2}, b^2_1] \qquad (4)$$

La tarea consiste en encontrar el vector óptimo θ^* que reside en un espacio de búsqueda R^7

2.3 Función de Fitness: Error Cuadrático Medio (MSE)

Para cuantificar el rendimiento de un conjunto de parámetros θ dado, se utiliza el Error Cuadrático Medio (MSE) como función de fitness. El MSE es una función de perdida estándar en problemas de regresión con redes neuronales (Godfellow et al., 2016). Mide la diferencia promedio al cuadrado entre la salida deseada t_k es la salida deseada y a_k la salida real de la red para un conjunto de ***N*** puntos de entrenamiento.

$$\text{Fitness}(\boldsymbol{\theta}) = \text{MSE} = \frac{1}{N}\sum_{k=1}^{N}(t_k - a_k)^2 \quad (5)$$

El objetivo de los algoritmos de optimización es encontrar el vector θ^* que minimice el valor de esta función.

2.4 Algoritmos de Optimización

Se implementaron y compararon dos algoritmos de optimización con filosofías fundamentalmente distintas. El primero, *Backpropagation*, es un método basado en gradientes, mientras que el segundo, la optimización por enjambre de partículas (PSO), es un algoritmo metaheurístico sin derivadas.

2.4.1. Optimización por Enjambre de Partículas (PSO)

PSO es un algoritmo metaheurístico poblacional inspirado en el comportamiento social de ciertos animales (Kennedy & Eberhart, 1995). El algoritmo mantiene una población (enjambre) de soluciones candidatas (partículas), donde la posición de cada partícula *i* corresponde a un vector de parámetros θ_i. En cada iteración *t*, la velocidad v_i y la posición θ_i de cada partícula se actualizan de acuerdo con las siguientes ecuaciones:

$$\mathbf{v}_i(t+1) \quad = w\mathbf{v}_i(t) + c_1 r_1(\mathbf{pbest}_i - \boldsymbol{\theta}_i(t)) + c_2 r_2(\mathbf{gbest} - \boldsymbol{\theta}_i(t)) \quad (6)$$

$$\boldsymbol{\theta}_i(t+1) \quad = \boldsymbol{\theta}_i(t) + \mathbf{v}_i(t+1) \quad (7)$$

donde *w* es el coeficiente de inercia, que modula la influencia de la velocidad anterior; c_1 es el coeficiente cognitivo, que pondera la atracción hacia la mejor solución personal de la partícula ($\boldsymbol{pbest_i}$), c_2 es el coeficiente social, que pondera la atracción hacia la mejor solución global del enjambre ($gbest$)**.** Los términos r_1, r_2 son números aleatorios en el intervalo [0,1] que introducen estocasticidad en la búsqueda. Un análisis detallado de la influencia de estos parámetros se encuentra en trabajos como el de Clerc and Kennedy (2002).

2.4.2 Retropropagación del Error (Backpropagation)

Backpropagation es un algoritmo de optimización local que implementa el **descenso de gradiente** (Rumelhart et al., 1986). Utiliza la regla de la cadena para calcular el gradiente de la función de error con respecto a cada parámetro de la red. Este gradiente indica la dirección de máximo crecimiento del error, por lo que los parámetros se actualizan en la dirección opuesta para minimizarlo. La regla de actualización para un parámetro genérico θ_j es:

$$\theta_j(t+1) = \theta_j(t) - \eta\frac{\partial \text{MSE}}{\partial \theta_j} \quad (8)$$

donde η es la tasa de aprendizaje, un hiperparámetro crítico que controla el tamaño del paso en cada iteración. El cálculo del gradiente $\frac{\partial MSE}{\partial \theta_j}$ se realiza propagando las sensibilidades del error hacia atrás, desde la capa de salida hasta la de entrada (Goodfellow et al., 2016, Capítulo 7).

2.5 Configuración Experimental

Todos los experimentos se realizaron utilizando un conjunto de datos de entrenamiento generado a partir de la función objetivo en el rango $p \in [-2,2]$ con un paso de 0.5. Para asegurar la validez estadística de los resultados, cada experimento se repitió 30 veces con inicializaciones aleatorias diferentes. Los parámetros específicos para cada algoritmo se detallan en la **Tabla 1**. El rendimiento se evaluó utilizando el MSE final, la tasa de éxito (proporción de ejecuciones con un MSE final inferior a 0.005) y el tiempo total de cómputo.

Parámetro	Símbolo / Configuración	Valor
Configuración General		
Número de Ejecuciones	–	30
Umbral de Éxito (MSE)	–	0.005
Optimizador PSO		
Número de Partículas	–	40
Número de Generaciones	–	100
Coeficiente de Inercia	w	0.6
Coeficiente Cognitivo	c_1	1.8
Coeficiente Social	c_2	1.8
Optimizador Backpropagation		
Número de Épocas	–	100
Tasa de Aprendizaje	η	0.02

Tabla 1. Parámetros de configuración para los experimentos de optimización

3. RESULTADOS

En esta sección se presentan los hallazgos empíricos obtenidos de la ejecución de los experimentos descritos en la metodología. Se reportan los resultados de 30 ejecuciones de PSO, evaluando la calidad de la aproximación a la señal objetivo, la convergencia del MSE, la evolución de los parámetros sinápticos y la dinámica de exploración-explotación del enjambre. Posteriormente, analiza la superficie de solución y la presencia de múltiples óptimos locales. Finalmente, se comparan PSO y Backpropagation en términos de MSE, desviación estándar, tasa de éxito y tiempo de cómputo, además de las curvas de convergencia que describen el comportamiento global de ambos algoritmos.

3.1 Análisis del Comportamiento del Optimizador PSO

El rendimiento del algoritmo PSO se evaluó a lo largo de 30 ejecuciones independientes. La **Figura 1** ilustra la capacidad de la red entrenada para aproximar la señal objetivo en tres casos representativos: la ejecución con el menor MSE final (mejor caso), la ejecución con el MSE mediano y la ejecución con el mayor MSE final (peor caso). En todos los casos, la red logra capturar la forma sinusoidal de la señal de referencia.

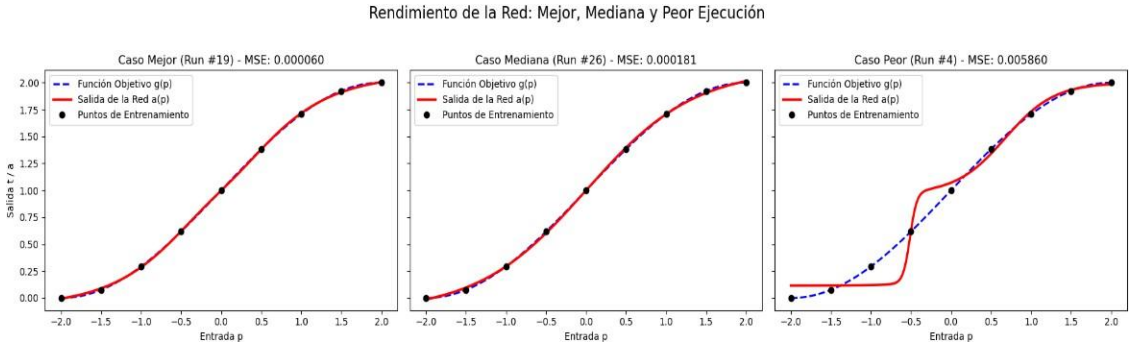

Figura 1. Comparación de la aproximación de la señal para la mejor, mediana y peor ejecución del algoritmo PSO. La línea discontinua azul representa la señal objetivo, la línea continua roja es la salida de la red y los puntos negros son los datos de entrenamiento.

El proceso de convergencia del enjambre se muestra en la **Figura 2**. La figura presenta las curvas de *fitness* (MSE) del mejor individuo a lo largo de las 100 generaciones para cada una de las 30 ejecuciones. La línea roja gruesa indica la trayectoria de convergencia promedio. Se observa una rápida disminución del error en las primeras generaciones, seguida de un ajuste más fino en las etapas posteriores.

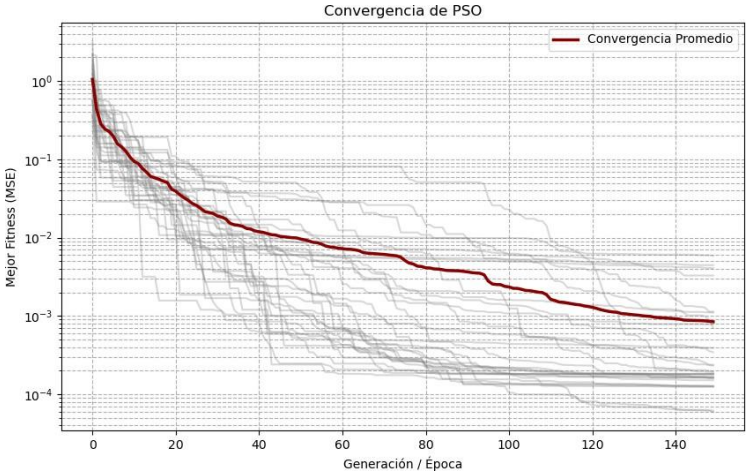

Figura 2. Curvas de convergencia de PSO para 30 ejecuciones. Cada línea representa una ejecución individual, mientras que la línea roja muestra el promedio.

Para comprender cómo el algoritmo encuentra una solución, la **Figura 3** rastrea la evolución de los siete parámetros del vector θ correspondientes a la mejor partícula global (*gbest*) durante la mejor ejecución. Se puede observar cómo los valores de los parámetros se ajustan y estabilizan a medida que avanzan las generaciones.

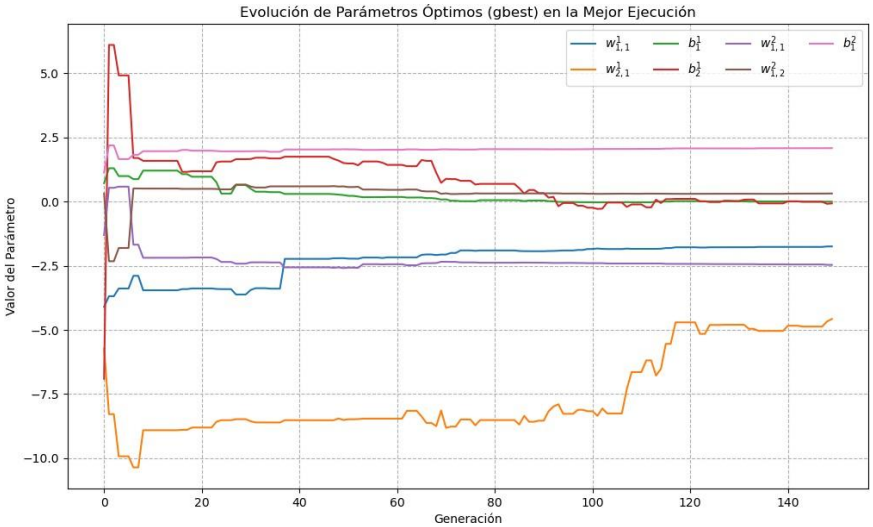

Figura 3. Evolución de los siete parámetros de la red neuronal (pesos y polarizaciones) a lo largo de las generaciones para la ejecución con el mejor rendimiento.

Finalmente, la dinámica interna del enjambre se analiza a través de la métrica de diversidad, como se muestra en la **Figura. 4.** La diversidad, medida como la desviación estándar promedio de las posiciones de las partículas, es alta en las primeras generaciones, lo que indica una fase de **exploración** del espacio de búsqueda. Posteriormente, la diversidad disminuye de manera constante, lo que significa que el enjambre converge hacia una región prometedora en una fase de **explotación.** Este equilibrio dinámico entre exploración y explotación es un comportamiento fundamental y deseable en los algoritmos de optimización poblacionales, ya que previene la convergencia prematura a soluciones subóptimas (Eiben & Smith, 2003).

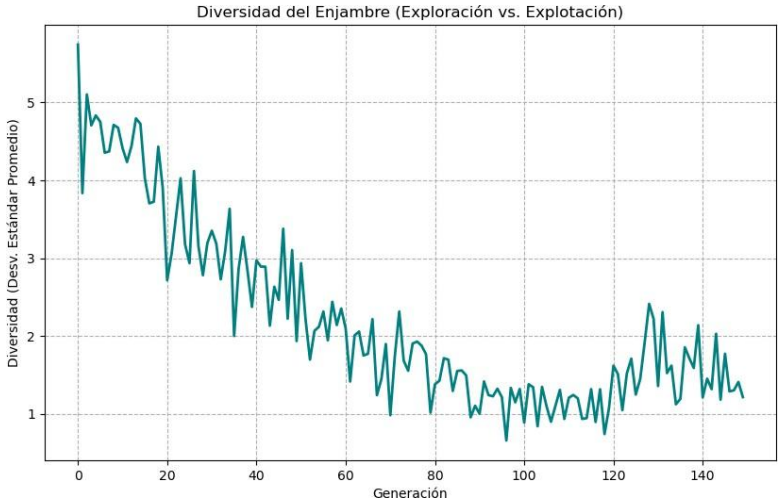

Figura 4. Diversidad del enjambre de PSO a lo largo de las generaciones. Una alta diversidad indica exploración, mientras que una baja diversidad indica explotación y convergencia.

3.2 Visualización de la superficie de solución.

Para contextualizar el desafío de optimización, se visualizó una sección transversal de la superficie de solución de 7 dimensiones. La **Figura 5** muestra el valor del MSE al variar dos de los pesos de la capa de salida ($w^2_{1,1}$ y $w^2_{1,2}$), manteniendo los otros cinco parámetros fijos en sus valores óptimos encontrados por PSO. La superficie de solución exhibe una topografía compleja y no convexa, con múltiples valles y regiones de bajo error, lo que sugiere la presencia de numerosos mínimos locales.

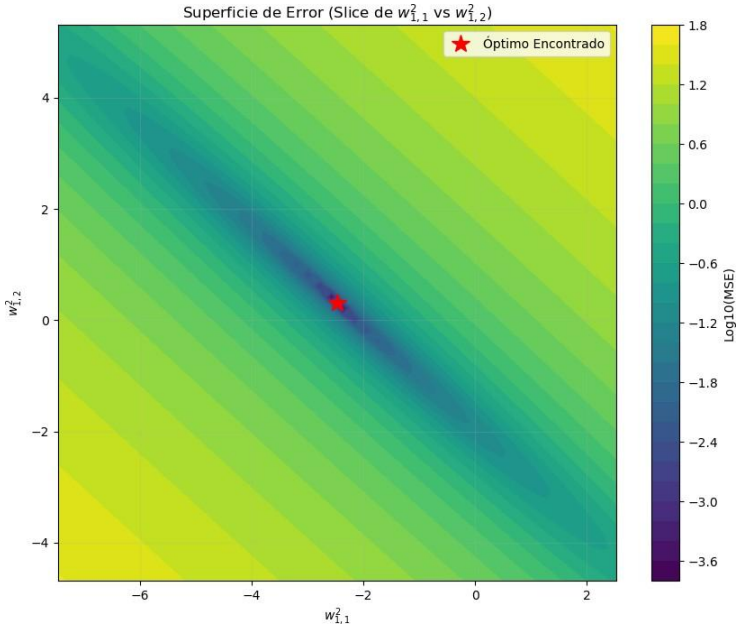

Figura 5 Visualización de una rebanada 2D de la superficie de solución Los colores más oscuros representan un MSE más bajo (mejor fitness). La estrella roja marca la ubicación de la solución óptima encontrada por PSO.

3.3 Comparativa de Rendimiento: PSO vs. Backpropagation

La comparativa directa del rendimiento entre PSO y *Backpropagation* se resume en la **Tabla 2**. Los valores representan el promedio y la desviación estándar de las 30 ejecuciones para cada algoritmo. Los resultados muestran que PSO alcanzó un MSE final promedio más bajo y con menor variabilidad (menor desviación estándar) que *Backpropagation*. Además, PSO logró una tasa de éxito del 100%, mientras que la de *Backpropagation* fue del 86.7%. En contraste, el tiempo de cómputo total para las 30 ejecuciones fue considerablemente menor para *Backpropagation*.

Métrica	PSO	Backpropagation
MSE Final Promedio	0.000378	0.003157
Desviación Estándar del MSE	0.000122	0.005118
Tasa de Éxito (%)	100.0	86.7
Tiempo de Cómputo Total (s)	10.32	1.58

Tabla 2. Resumen comparativo de rendimiento entre PSO y Backpropagation

La diferencia en la dinámica de convergencia se ilustra en la **Figura 6**. PSO muestra una **curva de aprendizaje más estable y consistente**, alcanzando un nivel de error final más bajo. Backpropagation, aunque desciende rápidamente al principio, se estanca en un nivel de error promedio más alto.

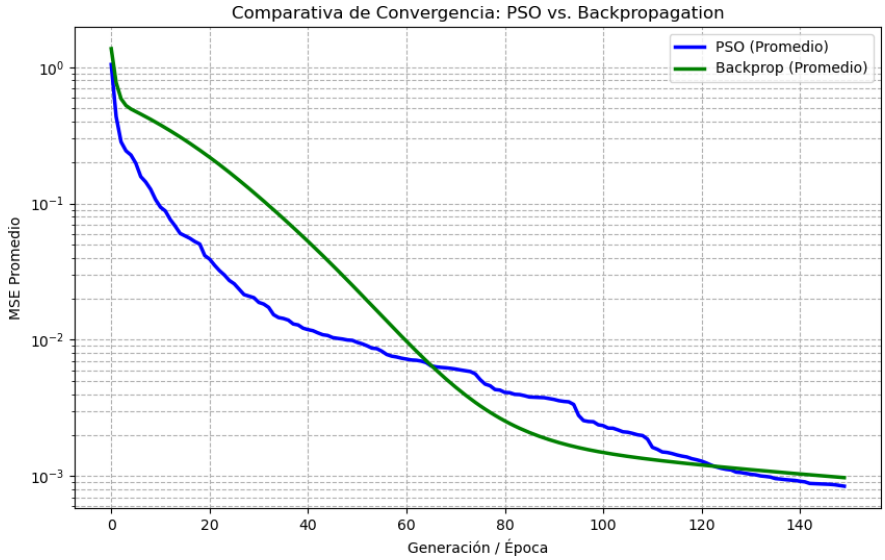

Figura 6 Comparación de las curvas de convergencia promedio para PSO (azul) y Backpropagation (verde) a lo largo de 100 iteraciones.

4. DISCUSIÓN

Los resultados de la sección anterior proporcionan los datos para interpretar el rendimiento del algoritmo de optimización. En esta sección, se analiza el significado de estos hallazgos, se abordan las preguntas de investigación planteadas y se discuten las implicaciones del estudio.

4.1 Analisis del Comportamiento de PSO

Los resultados obtenidos sugieren que el desempeño consistente del optimizador PSO reside directamente de su mecánica de búsqueda poblacional. En la Figura 4 se observa un equilibrio entre la exploración inicial y explotación posterior. Durante las iteraciones iniciales, la elevada diversidad del enjambre le permite muestrear una amplia regio del espacio de parametros, lo que aumenta la probabilidad de que las partículas localicen regiones prometedoras. A medida que el enjambre localiza estas regiones, la comunicación entre las partículas (a través del término *gbest*) hace que la diversidad disminuya, lo que indica una transición hacia una fase de explotación.

Esta fase de ajuste fino permite al algoritmo converger de manera precisa hacia una solución de bajo error, como se observa en la trayectoria de los parámetros en la **Figura. 3**. Este comportamiento dual es fundamental para el éxito de los metaheurísticas en problemas de optimización complejos.

4.2 La Robustez de PSO frente a la Búsqueda Local

La diferencia más notable entre los dos métodos reside en su robustez. La **Tabla 2** muestra que PSO no solo alcanzó un MSE promedio inferior, sino que también tuvo una desviación estándar significativamente menor, junto con una tasa de éxito del 100%. En contraste, Backpropagation mostró una mayor variabilidad en sus resultados y falló en alcanzar el umbral de éxito en aproximadamente el 13% de las ejecuciones.

Esta disparidad en la fiabilidad puede explicarse directamente por la topografía de la superficie o plano de solución visualizado en la **Figura 5**. La presencia de múltiples valles y una estructura no convexa crea trampas potenciales para un algoritmo de búsqueda local como Backpropagation. Al seguir exclusivamente la información del gradiente local, es propenso a quedar atrapado en un mínimo local que, si bien es un punto de bajo error, no representa la solución óptima global. Por otro lado, la naturaleza poblacional de PSO le permite mantener una perspectiva global del espacio de búsqueda. Si una parte del enjambre queda atrapada en un mínimo local, otras partículas que exploran diferentes regiones pueden encontrar una solución superior y, a través del mecanismo de comunicación social, guiar al resto del enjambre hacia esa mejor área.

4.3 El Compromiso entre Velocidad y Fiabilidad

Si bien PSO demostró una mayor robustez, Backpropagation fue considerablemente más rápido en términos de tiempo de cómputo. Esta diferencia se debe a sus fundamentos operativos. Backpropagation realiza un cálculo de gradiente y una actualización de parámetros por cada punto de datos, un proceso computacionalmente eficiente. En cambio, PSO requiere evaluar la función de *fitness* (que implica una pasada completa por todo el conjunto de datos) para cada una de las 40 partículas en cada una de las 100 generaciones.

Esto presenta un compromiso fundamental entre velocidad y fiabilidad. Para problemas donde la superficie de solución es relativamente simple o donde la velocidad de entrenamiento es el factor crítico, Backpropagation sigue siendo una opción muy atractiva. Sin embargo, para problemas "caja *negra"* o aquellos donde se sospecha una alta complejidad del paisaje de error y donde la fiabilidad de la solución final es primordial, el costo computacional adicional de un método como PSO puede estar justificado para garantizar una convergencia a una solución de alta calidad.

4.4 Limitaciones del Estudio

Es importante reconocer las limitaciones de este trabajo. El estudio se centró en una única y relativamente simple arquitectura de red neuronal (1-2-1) y en una tarea específica de seguimiento de señal. El rendimiento comparativo de los algoritmos podría variar en arquitecturas más profundas o complejas, donde el número de dimensiones del espacio de búsqueda (el "*curse of dimensionality"*) se convierte en un desafío mayor para cualquier algoritmo de optimización. Además, solo se evaluó PSO; otras metaheurísticas, como los algoritmos genéticos o la optimización por colonia de hormigas, podrían ofrecer diferentes perfiles de rendimiento.

4.5 Implicaciones y Futuras Líneas de Investigación

A pesar de sus limitaciones, los resultados de este estudio sugieren que los algoritmos metaheurísticos como PSO son una alternativa viable y robusta para el entrenamiento de redes neuronales, especialmente en contextos donde los métodos basados en gradiente pueden fallar. Este es el caso de arquitecturas con funciones no diferenciable so problemas de optimización con funciones de coste discontinuas.

Sería valioso aplicar esta metodología comparativa a problemas de mayor complejidad, como el seguimiento de señales caóticas o el entrenamiento de redes neuronales recurrentes. Otra área es la investigación de algoritmos híbridos que combinen las fortalezas de ambos enfoques. Por ejemplo, se podría utilizar PSO para una búsqueda global inicial que localice una región prometedora del espacio de búsqueda, y luego emplear Backpropagation para un ajuste fino y rápido dentro de esa región.

5. CONCLUSIONES

Este estudio tuvo como objetivo evaluar la eficacia de la Optimización por Enjambre de Partículas (PSO) como método de entrenamiento para una red neuronal en una tarea de seguimiento de señal, y comparar su rendimiento con el algoritmo de Backpropagation. A través de un riguroso análisis experimental, se han extraído las siguientes conclusiones principales:

En primer lugar, los resultados confirman que PSO constituye un método de entrenamiento notablemente robusto y eficaz para la arquitectura bajo estudio. Su desempeño no solo fue consistente, sino infalible, alcanzando una tasa de éxito del 100% en todas las réplicas experimentales. Esta remarcable confiabilidad puede atribuirse directamente a su mecánica de búsqueda poblacional, la cual, al equilibrar de manera inteligente la exploración del espacio de soluciones con la explotación de regiones prometedoras, demostró una capacidad superior para navegar los complejos paisajes de error típicos de estos problemas.

- El algoritmo PSO es un método de entrenamiento robusto y eficaz para la arquitectura de red propuesta. Demostró una capacidad consistente para encontrar conjuntos de parámetros que minimizan el error de seguimiento, logrando una tasa de éxito del 100% en todas las ejecuciones. Esto se deben a su dinámica de búsqueda, que equilibra la exploración global y la explotación local, que le permite navegar la superficie de solución.
- Mientras que Backpropagation es más rápido, su naturaleza de búsqueda local lo hace susceptible a mínimos locales, lo que resulta en una mayor variabilidad de rendimiento y una menor tasa de éxito. En contraste, PSO proporciona una solución más fiable y precisa, posicionándolo como un método superior para problemas donde la calidad y la consistencia de la solución son críticas, a costa de un mayor tiempo computacional.

En resumen, este trabajo establece que las metaheurísticas, y en particular PSO, no solo son una alternativa viable, sino una herramienta potente y fiable para el entrenamiento de redes neuronales. Ofrecen una solución robusta al desafío de superficies de solución no convexas, abriendo la puerta a su aplicación en problemas de optimización más complejos donde los métodos tradicionales basados en gradiente pueden resultar insuficientes.

REFERENCIAS

Choromanska, A., Bena, M., LeCun, Y., & Arous, G. (2015). The loss surfaces of multilayer networks. Journal of Machine Learning Research, 38, 192–204.

Clerc, M., & Kennedy, J. (2002). The particle swarm – explosion, stability, and convergence in a multidimensional complex space. IEEE Transactions on Evolutionary Computation, 6(1), 58–73.

Eiben, A. E., & Smith, J. E. (2003). Introduction to evolutionary computing. Springer.

Engelbrecht, A. P. (2007). Computational intelligence: An introduction (2nd ed.). John Wiley & Sons.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep learning. MIT Press.

Hagan, M. T., Demuth, H. B., Beale, M. H., & De Jesús, O. (2014).Neural network design (2nd ed.). [Disponible en: http://hagan.okstate.edu/NNDesign.pdf]. Martin Hagan.

Hornik, K., Stinchcombe, M., & White, H. (1989). Multilayer feedforward networks are universal approximators. Neural Networks, 2(5), 359–366.

Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. Proceedings of ICNN'95 – International Conference on Neural Networks, 4, 1942–1948. https://doi.org/10.1109/ICNN.1995.488968

Ljung, L. (1999). System identification: Theory for the user (2nd ed.). Prentice Hall.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. Nature, 323, 533–536.

Yudong, Z., & Lenan, W. (2015). A survey on swarm intelligence for training artificial neural network. International Journal of Machine Learning and Cybernetics, 6, 257–271.

Esta obra está bajo una licencia de Creative Commons
Atribución-NoComercial-Compartirlgual 4.0 Internacional.