Influencia de los Algoritmos de Entrenamiento de RNAs en la Predicción del Nivel de Embalse de Agua en una Estación Hidroeléctrica

Influence of Training Algorithms of ANNs on the Prediction of Reservoir Water Level of an Hydroelectric Station

E. Chafla[1,a], G. Asqui-Santillán[2,b], J. Paucar[3,c], D. Olmedo-Vizueta[3,d]
[1]Ministerio del Interior, 17105, Quito, Ecuador
[2]Transelectric CELEC E.P, 180250, Baños, Ecuador
[3]Escuela Superior Politécnica de Chimborazo, 060150, Riobamba, Ecuador

Email: aedivax@hotmail.es, bleirbaggeo06@gmail.com, cjlpaucar@espoch.edu.ec, ddiana.olmedo@espoch.edu.ec

Resumen- El presente artículo reporta los resultados obtenidos del análisis de la influencia de los algoritmos de entrenamiento de redes neuronales artificiales (RNAs) en el error de predicción del nivel de embalse de agua de una represa hidroeléctrica. Los algoritmos estudiados son los contenidos en la librería Keras, la misma que usa el back-end de TensorFlow. Los datos utilizados son los registros históricos (2005-2016) del nivel de embalse, caudal y potencia activa de una central hidroeléctrica en el Ecuador. Tales datos fueron divididos en datos de entrenamiento, validación y pruebas. La plataforma hardware fue una unidad de procesamiento gráfico (GPU) Nvidia 1050Ti, la misma que permitió explotar el cálculo altamente paralelo de TensorFlow. Un total de 7 algoritmos fueron evaluados. La prueba de Tukey reveló que el algoritmo Nadam obtuvo la menor diferencia significativa respecto al resto, comprobando que es el más eficiente. El modelo RNA de la planta, entrenado con el algoritmo Nadam, permitió lograr predicciones del nivel de embalse de agua hasta un umbral de 48 horas. Los resultados alcanzados favorecerán optimizar la planificación de producción energética en una central hidroeléctrica a través de una predicción más precisa de los recursos hídricos para cotas de producción deseadas.

Palabras Clave— Inteligencia Artificial, Redes Neuronales, Predictor, Keras, Tensorflow.

Abstract- The present paper reports on the obtained results from the analysis of the influence of training algorithms, for artificial neural networks (ANNs), on the prediction error of the reservoir water level of a hydroelectric station. The studied algorithms are those forming the Keras library, which uses the back-end of TensorFlow. Data for this study are the historical records (2005-2016) of reservoir level, streamflow, and active power from an Ecuadorian hydroelectric plant. Such data was divided for the training, validation, and test stages. The hardware platform was a graphic processing unit (GPU) Nvidia 1050Ti, which allowed for exploiting the highly parallel computing capability of TensorFlow. Seven algorithms were evaluated. The Tukey test revealed that the Nadam algorithm obtained the lowest significative difference respect to its counterparts, engaging it as the more efficient. The respective obtained RNA plant model reached effective prediction thresholds up to 48 hours. The obtained results allow for optimization of the planification of energy production on the hydroelectric station trough an accurate prediction of the hydric resources for quotas of desired production.

Keywords— Artificial Intelligence, Artificial Neural Networks, Predictor, Keras, Tensorflow.

Fecha de Recepción: 06 - Nov - 2018 *Fecha de Aceptación: 24 - Nov - 2018*

I. INTRODUCCIÓN

La energía es fundamental para el crecimiento económico y la sostenibilidad ambiental, y se ha descrito como "el hilo " que une el crecimiento económico, la equidad social y la sostenibilidad ambiental. La producción y consumo de electricidad de un país son indicadores básicos de su tamaño y nivel de desarrollo. De acuerdo con cifras publicadas por el Banco Mundial, en abril de 2017, 1060 millones de personas aún vivían sin electricidad, lo que representó tan solo una leve mejora desde 2012. Según el mismo estudio, el mundo alcanzaría el 92 % de electrificación para el año 2030 [1].

Además, de acuerdo al Banco Mundial, el consumo de energía eléctrica por habitante (kWh per cápita) en el año 2014 se ubicó en 3.126,3 kWh/hab a nivel mundial [1]. Por otro lado, en el Ecuador, para el año 2016 este índice fue de 1.143,31 kWh/hab con un consumo de 18.897,43 GWh, para una población de 16`528.730 habitantes [2][3]. Como es posible notar, en el Ecuador este índice, mismo que está muy relacionado con el nivel de desarrollo de una sociedad o país, no representa ni la mitad de la media mundial.

En este sentido, y de acuerdo a políticas de Estado referente a la eficiencia energética, y tomando en cuenta que la proyección del índice de población del país para el año 2020 sería de 17´512.663 habitantes [3], es imperativo la implementación de centrales hidroeléctricas que aseguren el abastecimiento normal de energía eléctrica para la población. De hecho, el Ecuador ha realizado fuertes inversiones económicas en infraestructuras hidroeléctricas durante la última década buscando alcanzar su autonomía energética, aunque con resultados nada alentadores influenciados por crisis políticas y sociales [4], [5]. Sin embargo, actualmente el Ecuador aún se encuentra ejecutando proyectos para la generación de energía renovable, como son: Coca Codo Sinclair, Minas San Francisco, Delsitanisagua, Manduriacu, Masar Dudas, Toachi Pilatón, Quijos, Sopladora y Villonaco [2].

Por otro lado, Ecuador cuenta con el Plan Maestro de Electrificación (2013-2022), emitido por el Consejo Nacional de

DOI: https://doi.org/10.47187/perspectivas.vol1iss1.pp16-22.2019

Electricidad (CONELEC). El capítulo 1 de dicho plan, en referencia a los aspectos de sostenibilidad social y ambiental, indica; "La eficiencia energética, basado en las políticas del Ministerio de Electricidad y Energía Renovables (MEER), trata sobre el conjunto de acciones, en ejecución y planificadas, tendientes a optimizar los recursos energéticos renovables y consumir la menor cantidad posible de energía para realizar un proceso determinado, sin disminuir las prestaciones o la calidad final del producto, y con el menor impacto sobre el medio ambiente" [6]. Es evidente que dicho manifiesto sostiene que la optimización de recursos energéticos en el país se convierte en una prioridad.

Entonces, un punto muy importante a considerar es la optimización en la producción de la energía en las centrales en operación. Un primer caso de estudio utilizando RNAs es el presentado en [7], donde los autores utilizaron el algoritmo de entrenamiento Broyden–Fletcher–Goldfarb–Shanno (BFGS) para entrenar una Red Neuronal Artificial (RNA) perceptrón multicapa con el objeto de obtener un predictor de nivel de embalse de agua basado en dos RNAs: la primera utilizada en el modelamiento dinámico de la presa, y la segunda para predecir el caudal de ingreso. Las predicciones alcanzadas lograron un umbral de 8h, sin embargo, el error del resultado final del sistema dependía de la combinación del error de estas dos RNAs[8].

El presente trabajo pretende estudiar la influencia que tienen los algoritmos de entrenamiento de RNAs mediante el análisis del error de predicción del nivel de embalse de agua de una represa hidroeléctrica. El objetivo es determinar cuál de los algoritmos de entrenamiento de RNAs implementados en la librería open-source Keras presenta un mejor rendimiento para este tipo de aplicaciones. La metodología empleada involucra el modelamiento dinámico de una planta hidroeléctrica usando RNAs, considerando que si se logra disminuir el error del modelo también se disminuirá el error del predictor del nivel de embalse de agua. De esta forma, se realiza una comparación de los distintos algoritmos de entrenamiento de RNAs proporcionados por la librería Keras, utilizando el back-end de TensorFlow y una Unidad de Procesamiento Gráfico (GPU). Esto permitirá verificar la influencia del algoritmo de entrenamiento de RNAs durante el modelamiento dinámico de la presa, y por consiguiente en el error de predicción de embalse de agua. Los resultados alcanzados permitirán desarrollar una herramienta que apoye las actividades de planificación de producción energética, optimizando el uso de recursos naturales en centrales hidroeléctricas.

II. METODOLOGÍA

En la Fig. 1 se muestra el diagrama metodológico que permitió el entrenamiento de RNAs, con el uso de los algoritmos de entrenamiento de la librería Keras de Python©, con el objeto de determinar cuál de estos presentaba un mejor desempeño en base a el tiempo y error de entrenamiento, error de validación y umbral de predicción del nivel de embalse de agua de una presa hidroeléctrica.

A. Caso de Estudio.

En este caso de estudio, la información utilizada proviene de la central Hidroeléctrica Agoyán, ubicada en la provincia de Tungurahua a 180 Km al sureste de la ciudad de Quito y a 5 km al este de la ciudad de Baños, en el sector denominado Agoyán de la parroquia Ulba. Dicha central fue concebida para aprovechar el caudal del río Pastaza y cuenta con una producción media anual de 1.010 GWh/año, una potencia efectiva y nominal de 156 MW y 160 MW, respectivamente, y un factor de planta de 73,90 % [6].

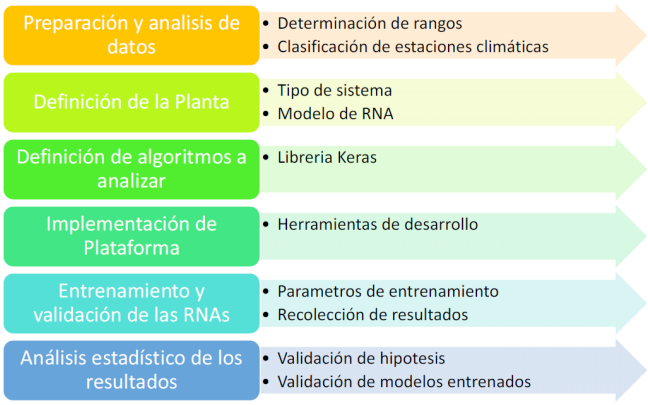

Fig. 1. Fases de la metodología del presente estudio.

B. Preparación y Análisis de Datos.

La información utilizada corresponde a los datos de nivel, caudal y potencia activa generada por la Hidroeléctrica Agoyán, comprendidos entre el año 2005 y 2016. Los datos fueron adquiridos mediante un sistema SCADA (Supervisory Control and Data Acquisition), y provienen de los sensores de nivel de embalse de la presa y de caudal del río Pastaza. Los datos fueron filtrados, validados y pre-procesados como se indica en [7], antes de ser utilizados en este estudio.
Un análisis de los datos efectuado a la variable caudal, y reportado en [7], permitió identificar que los datos de caudal tienden a repetirse cada año. Por tal motivo se definió el término de "Temporadas Climáticas", y se catalogaron 3 tipos tal como se presenta en la Tabla I.

Tabla I.
TEMPORADAS CLIMÁTICAS

Período	*Temporada*	*Meses*
Invernal Fuerte	*1*	*Marzo – Julio*
Invernal Ligera	*2*	*Agosto – Octubre*
Verano	*3*	*Noviembre - Febrero*

C. Descripción del Modelo Planta basado en RNA.

La configuración del modelo de la presa es un esquema MISO (Multiple Input Single Output), es decir un sistema que tiene varias entradas y salida única [9]. Como ya se mencionó, para el caso de estudio que se reporta, las entradas son: caudal, nivel de embalse y potencia activa, mientras que como salida se obtiene el nivel futuro del embalse de agua en una hora (N+1), como se muestra en la Fig. 2. El modelo de una planta MISO puede ser aproximada mediante un modelo de RNA llamado Perceptrón Multicapa [10].

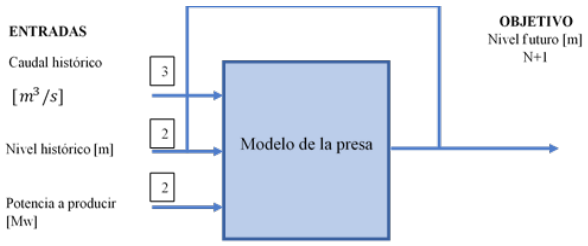

Fig. 2. Diagrama del modelo de la presa en configuración de entrenamiento CLP.

Para poder determinar el error de predicción del nivel de embalse de agua en el futuro (N+1) durante el entrenamiento se utilizó una configuración CLP (predicción a lazo cerrado) según lo reportado en [11]. Además, el número de neuronas de entrada y el número de capas ocultas de la RNA fueron determinadas a partir del PSD (densidad espectral de potencia) de las tres señales de entrada siguiendo los lineamientos establecidos en [7], [12], [13]. En particular, la PSD muestra las frecuencias a las cuales las señales de entrada son más fuertes, y de esta manera se combina el número de neuronas. Para el caso de estudio, se determinó 3 neuronas para la variable caudal, 2 neuronas para la variable nivel, y 2 neuronas para la potencia activa, mientras que el número de capas ocultas de la RNA es 2. Por otro lado, la salida está compuesta de una sola neurona que corresponde al nivel futuro de nivel de embalse, como se muestra en la Fig. 3.

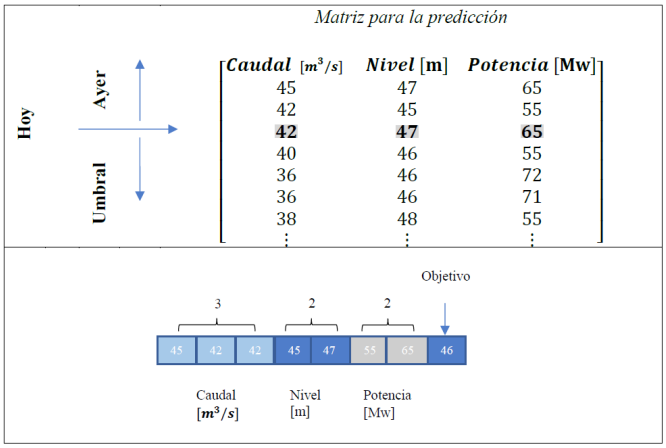

Fig. 3. Configuración de los datos de ingreso a la RNA para la predicción de un valor futuro de nivel de embalse de agua (objetivo).

Como ya se estableció, la predicción del nivel futuro de embalse de agua se realiza a lazo cerrado (CLP) utilizando los mismos valores predichos del nivel de embalse. En este proceso es necesario la definición de un umbral de predicción (u), el mismo que determinará el tamaño de la ventana de deslizamiento de los datos de entrada, como se muestra en la Fig. 3.

D. Algoritmos de Entrenamiento.

Keras es una librería de acceso abierto escrita en Python, en la cual se pueden realizar la creación de modelos de RNAs para aplicaciones de Deep Learnig, utlizando backends como TensorFlow, Theano o CNTK como se muestra en la Fig. 4 [14], [15]. Los algoritmos de entrenamiento de RNAs que serán analizados se encuentran definidos en la librería Keras, los mismos que detallan a continuación:

1. Descenso de Gradiente Estocástico SGD - es simple y bien conocido, en general es un algoritmo de optimización muy robusto, en el cual el gradiente de la función que se minimiza con respecto a los parámetros calculados [16].

2. AdaGrad - es una adaptación del SGD que se enfoca en reducir la velocidad de aprendizaje (gradiente) en zonas que han cambiado significativamente, y acelerarlo en las zonas que han cambiado ligeramente [17].

3. Adadelta - es una extensión del algoritmo Adagrad, que busca reducir su agresividad, disminuyendo monótonamente la tasa de aprendizaje [18], [19].

4. RMSprop - este algoritmo de aprendizaje es una alternativa del algoritmo Adagrad, enfocado en permitir que el modelo continúe aprendiendo indefinidamente [20].

5. Adam - combina el momento clásico con el algoritmo RMSprop para obtener un mejor rendimiento, e incluye términos de corrección del sesgo inicial [21].

6. Adamax - es una variante del algoritmo Adam basada en la norma del infinito [22].

7. Nadam - este algoritmo de aprendizaje es el resultado de la combinación del algoritmo NAG con el Adam [23].

E. Implementación de la Plataforma.

La plataforma de software fue desarrollada en el lenguaje de programación Python© versión 3.6. Para las interfases de usuario se utilizó PyQt versión 5. PyQt5 es un método de la biblioteca gráfica Qt para el lenguaje de programación Python©. Los datos fueron almacenados en el gestor de base de datos PostgreSQL.

Para el entrenamiento de las RNAs utilizamos los algoritmos de entrenamiento de la librería Keras de Python©, la misma que es una biblioteca a nivel de modelo que proporciona bloques de construcción de alto nivel para desarrollar modelos de aprendizaje profundo. Se basa en una biblioteca de manipulación de tensores especializada y bien optimizada, que actúa como el "motor de back-end" de Keras. En lugar de elegir una sola biblioteca de tensores y hacer que la implementación de Keras esté ligada a esa biblioteca, Keras maneja el problema de una manera modular, y varios motores back-end diferentes pueden conectarse sin problemas a Keras [13].

De acuerdo a la Fig. 4, Keras tiene tres implementaciones de back-end disponibles: TensorFlow, Theano y CNTK. Además, Keras puede funcionar sin problemas en tanto CPU o GPU. Cuando se ejecuta en la CPU, TensorFlow utiliza una biblioteca de bajo nivel para las operaciones de tensor llamada Eigen. En GPU, TensorFlow utiliza la biblioteca de operaciones de aprendizaje profundo bien optimizadas llamada biblioteca NVIDIA CUDA Deep Neural Network (cuDNN)[15].

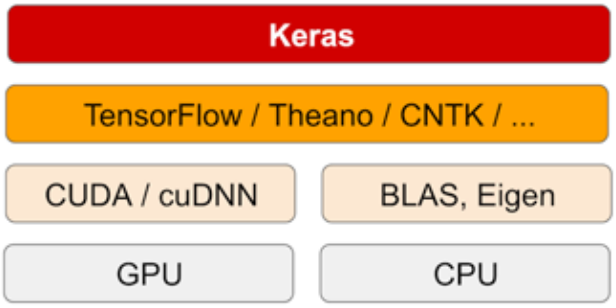

Fig. 4. Estructura de la plataforma hardware-software sobre las que funciona la librería Keras [15].

F. Entrenamiento de los Modelos.

Los valores de época, lote e iteraciones utilizados para el entrenamiento de las RNAs son presentados en la Tabla II. Es importante notar que para un primer acercamiento al entrenamiento de los modelos de RNA se ha definido 5 interacciones, las mismas que podrán ser ampliadas en función de los resultados preliminares.

Tabla III.
PARÁMETROS DE ENTRENAMIENTO

Parámetros	*Valores*
Epoca (epoch)	*15000*
Lote (batch)	*128*
Iteracciones	*5*

Una vez entrenados los modelo de RNA, con los distintos algoritmos u optimizadores de la librería Keras de Python©, para cada una de las estaciones climáticas. A través de un análisis estadístico se puede analizar el comportamiento del error de entrenamiento y error de validación en la predicción de nivel de agua de embalse con cada uno de los modelos y de esta manera poder reconocer cual algoritmo presenta un mejor rendimiento.

Con el fin de determinar si existe un algoritmo de entrenamiento que mantenga una diferencia significativa respecto a los demás algoritmos estudiados, y por lo tanto que presente el mejor rendimiento, se plantea las siguientes hipótesis:

H_0, Hipótesis nula: El promedio del error de entrenamiento (o validación) es igual en los 7 algoritmos de entrenamiento, con el 95% de confiabilidad.

$$H_0: \mu_1=\mu_2=\mu_3...\mu_a=\mu \quad (1)$$

H_1, Hipótesis alterna: Al menos en un algoritmo de entrenamiento la media del error de entrenamiento (o validación) es distinta, con el 95% de confiabilidad.

$$H_1: \quad \text{no es cierto } H_0 \quad (2)$$

III. RESULTADOS Y DISCUSIÓN

A. Comparación de Algoritmos.

La Fig. 5 muestra el comportamiento del error de entrenamiento (EE) y error de validación (EV), en las tres temporadas climáticas (T1: Invernal fuerte, T2: Invernal ligera y T3: Verano), de cada uno de los algoritmos evaluados. Se observa que la varianza tiende a cero (0) por lo que no es necesario realizar más iteraciones que las cinco propuestas. Es importante mencionar que, a primera vista, que los algoritmos RMSprop, Adam, Adamax y Nadam presentan resultados similares y son mejores que los algoritmos SGD, Adagrad y Adadelta.

En la Fig. 6 se muestra el tiempo total de los entrenamientos del modelo con los diferentes algoritmos para las tres temporadas climáticas en total. Como se puede apreciar, el algoritmo Adagrap es el que menos tiempo empleó en el entrenamiento, sin embargo, como ya se mencionó sus resultados de error de entrenamiento y validación no son los mejores. Por otro lado, el algoritmo SGD sigue mostrando su bajo rendimiento al emplear mucho más tiempo que los demás, mientras que la mayor parte de algoritmos comparados emplean tiempo de entrenamiento similares.

A pesar de que las comparaciones realizadas en este apartado permitieron identificar de forma general cuales de los algoritmos analizados presentaron un mejor o peor rendimiento en los errores de predicción durante el entrenamiento y validación, y tiempos de entrenamiento, todavía no está claro cuál de ellos es el mejor de todos. Para ello, es necesario utilizar herramientas estadísticas que permitan determinar si existen diferencias significativas entre los modelos obtenidos.

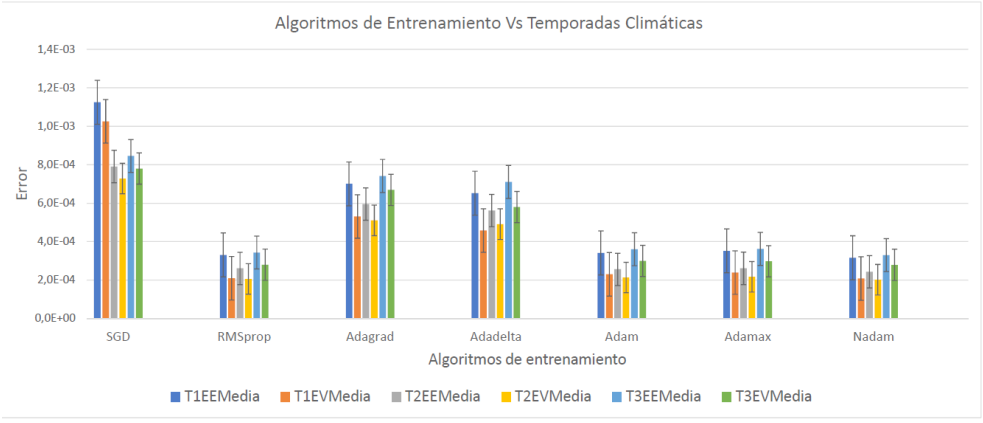

Fig. 5. Resultados de los errores de entrenamiento (EE) y error de validación (EV) de cada uno de los modelos de RNA entrenadas con los diferentes algoritmos estudiados para las tres temporadas climáticas (T1: Invernal fuerte, T2: Invernal ligera y T3: Verano).

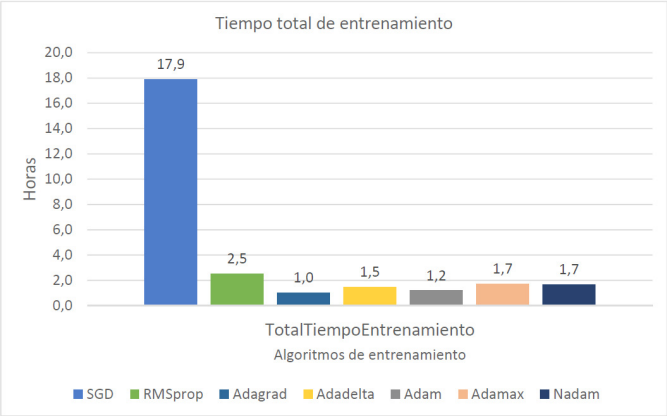

Fig. 6. Tiempo total de entrenamientos de los diferentes algoritmos estudiados de la librería Keras.

Dentro de las herramientas estadísticas utilizadas para la comparación de dos o más elementos o variables esta: la evaluación de significancia con t-tes, el análisis de varianza con la prueba de Tukey, y la eficiencia estadística. En este estudió se utilizó las dos últimas técnicas, debido a que la primera es utilizada para la comparación de un grupo únicamente de dos variables [24]. En el siguiente apartado, la prueba de Tukey es utilizada con un nivel de confianza (n) del 95% para probar las diferencias que existen entre las medias (μ) obtenidas de una experiencia [25].

B. Evaluación de Diferencias Significativas.

Como punto de partida, se realiza un análisis de la varianza (ANOVA) sobre los errores de entrenamiento obtenidos sobre el modelo de RNA con cada uno de los algoritmos analizados en este artículo. En la Tabla III se presenta un resumen de los datos de la media, varianza y el número de muestras de cada uno de los algoritmos de entrenamiento. Es importante tener en cuenta que cada muestra es el resultado de la media obtenida en cada entrenamiento por temporada. Además, la Tabla IV presenta el resultado del análisis de la varianza aplicada al error de entrenamiento, la misma que analiza las variaciones de error presentes entre los grupos y dentro de los mismos grupos.

Tabla III.
ANÁLISIS ANOVA DEL ERROR DE ENTRENAMIENTO

Algoritmos	Muestras	Suma	Media	Varianza
SGD	15	0,0138037	0,0009202	3,470E-08
RMSprop	15	0,0046720	0,0003114	1,445E-09
Adagrad	15	0,0101849	0,0006789	4,430E-09
Adadelta	15	0,0096177	0,0006411	4,241E-09
Adam	15	0,0047862	0,0003190	2,217E-09
Adamax	15	0,0048743	0,0003249	2,217E-09
Nadam	15	0,0044450	0,0002963	1,577E-09

Tabla IV.
ANÁLISIS DE LA VARIANZA PARA EL ERROR DE ENTRENAMIENTO.

Variaciones	SS	df	MS	F	P Valor	F crítico
Entre grupos	5,53E-06	6	9,22E-07	127	6,02E-44	2,2
Dentro grupos	7,13E-07	98	7,26E-09			
Total	6,26E-06	104				

Debido a que el valor de la probabilidad (p) es menor que el nivel de confianza (n=0.05): p < n, es decir 6,02E-44 <0.05, entonces rechazamos la hipótesis nula H_0 y validamos la hipótesis alterna H_1. Esto significa que al menos en un algoritmo de entrenamiento la media del error de entrenamiento (o validación) es distinta que las demás, con el 95% de confiabilidad. Este mismo análisis fue realizado para el conjunto de datos del error de validación obteniendo que p < n, es decir 1.4024E-42 <0.05, se cumple con un 95% de confiabilidad, resultado en el rechazo de H_0 y validando la hipótesis alterna H_1.

Con estos datos obtenidos del ANOVA, se procede a realizar la prueba de Tukey para encontrar las diferencias significativas entre los algoritmos de entrenamiento. Para esto, fue necesario calcular los factores HSD (diferencia honestamente significativa, en español) como:

$$HSD=VC\sqrt{(MSe/N)} \qquad (3)$$

donde:

VC: es el valor crítico (VC). De acuerdo con la Tabla característica de Tukey [26], para los 98 grados de libertad este valor es VC=4.24

MSe=(Suma de Cuadrados)/(Grados de Libertad)=7.2621E-09 (4)

N: Número de elementos en los grupos, N=15.

Este análisis se realizó tanto para el error de entrenamiento, como para el error de validación, obteniendo una HSD de 9.32936E-05 y 9.2904E-05, respectivamente.

Los valores de HSD obtenidos son comparados con la media (μ) de cada conjunto de datos correspondientes a los algoritmos de entrenamiento y validación para determinar la relación que existe entre cada uno de los algoritmos. Para llevar a cabo este proceso, al valor de la media (μ_1) de cada algoritmo se resta el valor medio (μ_2) del algoritmo con el que se desea comparar. Si el valor de la resta es superior al HSD, entonces existe una diferencia significativa entre ambos algoritmos; y si este valor es menor, entonces no existe diferencia:

$$HSD > \mu_1 - u_2 \quad ; \text{existe diferencia}$$

$$HSD \leq \mu_1 - u_2 \quad ; \text{No existe diferencia}$$

Los valores HSD obtenidos para cada algoritmo de entrenamiento, a partir de los errores de entrenamiento, son resumidos en la Tabla V. La prueba de Tukey determinó que el algoritmo Nadam tiene una mayor diferencia significativa respecto a los algoritmos SGD, Adagrad y Adadelta, y por lo tanto es mejor. Por otro lado, el test de Tukey también indica que el algoritmo Nadam pero tiene menor diferencias significativas con los algoritmos RMSprop, Adam y Adamax, y por lo tanto son muy similares. Sin embargo, la eficiencia estadística determinó que el algoritmo Nadam es el más eficiente dado que la Var(Nadam) es menor que la Var(Adam) y Var(Adamax) según lo indica la Tabla III. Además, según estos mismos datos, el algoritmo RMSprop resultaría más eficiente estadísticamente hablando dado que su Var(RMSprop) es aún menor, pero su tiempo de en-

trenamiento es mayor que el empleado por el algoritmo Nadam (ver Fig. 6). Resultados similares se obtuvieron con el análisis de los errores de validación para cada uno de los modelos de presa entrenados con los diferentes algoritmos estudiados.

Tabla V.
COMPARATIVO DE LOS VALORES HSD OBTENIDOS DE LOS ERRORES DE ENTRENAMIENTO DE CADA MODELO DE PRESA ENTRENADO CON CADA ALGORITMO ANALIZADO.

	RMSprop	Adagrad	Adadelta	Adam	Adamax	Nadam
SGD	6,0E-04	2,4E-04	2,8E-04	6,0E-04	5,9E-04	6,2E-04
RMSprop		-3,7E-04	-3,3E-04	-7,6E-06	-1,3E-05	1,5E-05
Adagrad			3,8E-05	3,6E-04	3,5E-04	3,8E-04
Adadelta				3,2E-04	3,1E-04	3,4E-04
Adam					-5,9E-06	2,3E-05
Adamax						2,9E-05

**Color naranja: Existen diferencias significativas.*
**Color azul: No existen diferencias significativas.*

C. Validación de los modelos

Finalmente se procede a realizar la evaluación de los modelos determinados como más y menos eficientes (precisos), en el módulo de predicción del nivel de embalse de agua de la plataforma desarrollada. Los modelos seleccionados para esta validación son aquellos entrenados con los algoritmos Nadam y SGD.

Los resultados son presentados en la Fig. 7 que analiza la evolución del RMSE (root-mean-square error) del nivel de agua de embalse de la presa. Cada temporada inicia con pruebas de predicción con umbrales de 4 horas y termina con umbrales de 48 horas. De acuerdo con esto, se puede verificar y confirmar lo analizado en el apartado anterior, en el cual se identifica al algoritmo Nadam como el más eficiente, presentando menores errores durante la fase de predicción del nivel embalse de agua para todos los umbrales de prueba. Como se puede observar, existe un rendimiento superior de este algoritmo en comparación al algoritmo de entrenamiento SGD.

IV. CONCLUSIONES

La evaluación del rendimiento de los algoritmos de entrenamiento de RNAs de la herramienta Open Source Keras, en función del tiempo de entrenamiento, error de entrenamiento y error de validación en la predicción del nivel de embalse de agua de la represa hidroeléctrica Agoyán, permitió identificar que el tipo de algoritmo de entrenamiento si influye en el error de nivel de embalse de agua ya que se identificaron diferencias significativas entre el algoritmo Nadam y varios de los algoritmos probados. En particular, se determinó que el algoritmo SGD, es el menos eficiente y el algoritmo Nadam es el más eficiente del grupo. No se identificaron diferencias significativas entre los algoritmos Nadam, Adam y Adamax, por lo cual se utilizó la eficiencia estadística para determinar cuál es más y menos eficiente. Se determinó que el algoritmo Nadan es el más eficiente de este subgrupo. Los resultados obtenidos en este estudio permitirán afinar las herramientas de predicción de embalse de agua en la central hidroeléctrica, la cual es reportada en [7]. Esto favorecerá una planificación optimizada de la producción energética de la planta al poder predecir hasta con umbrales de 48 horas los recursos hídricos teniendo en cuenta la producción deseada.

REFERENCIAS

[1] BANCO MUNDIAL, "Energía @ www.bancomundial.org." 2017.
[2] L. Galarza and Tecpetrol, "Estadística Anual y Multianual del Sector Eléctrico Ecuatoriano," p. 208, 2017.
[3] Instituto Nacional de Estadística y Censos, "Proyecciones Poblacionales @ www.ecuadorencifras.gob.ec." 2017.
[4] REN21, "Renewables 2017: Global Status Report," Paris, 2017.

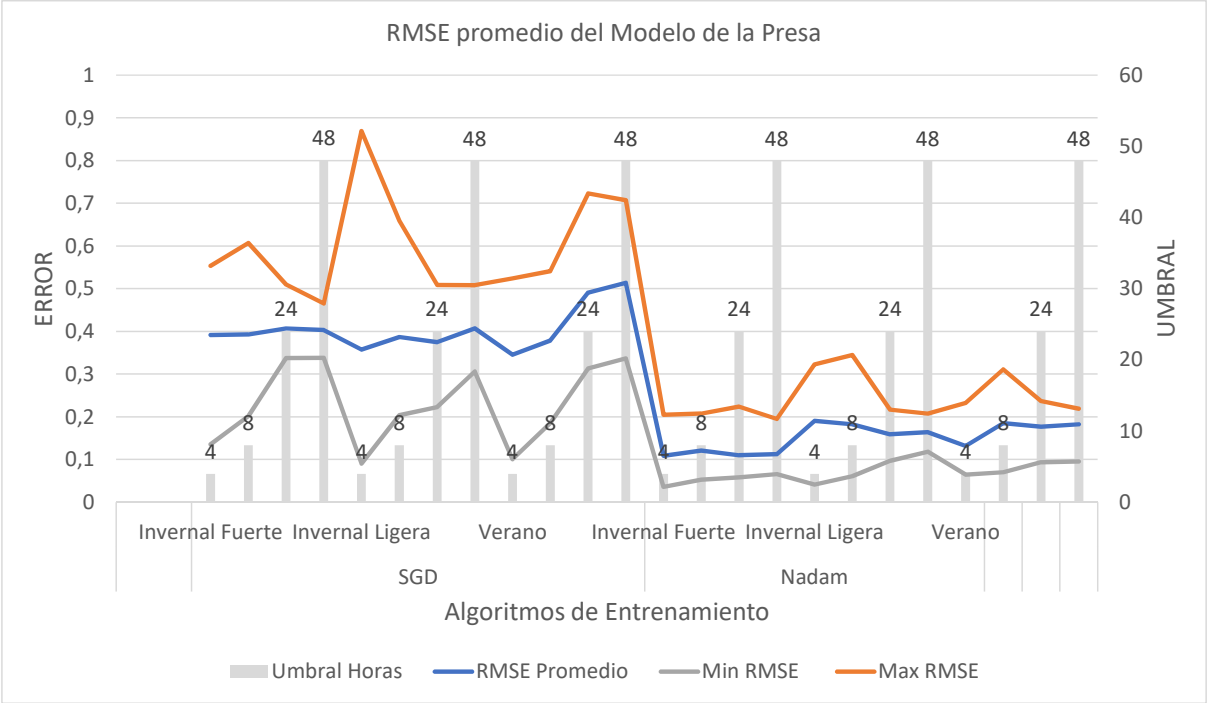

Fig. 7. Evaluación de los algoritmos Nadam vs SGD.

[5] C. A. Villacís Laínez, Y. F. Suarez Nuñez, and X. M. Güillín Llanos, "Análisis de la Responsabilidad Social en el Ecuador," Rev. Publicando, vol. 3, no. 8, pp. 452–466, 2016.

[6] Consejo Nacional de Electricidad, "Plan Maestro de Electrificación 2013-2022," Plan Maest. Electrif. 2013-2022, vol. 1, p. 116, 2013.

[7] J. Hernández-Ambato, G. Asqui-Santillán, A. Arellano, and C. Cunalata, "Multistep-ahead Streamflow and Reservoir Level Prediction Using ANNs for Production Planning in Hydroelectric Stations," in 2017 16th IEEE International Conference on Machine Learning and Applications, 2017, pp. 479–484.

[8] G. E. Asqui Santillán, "Predicción del nivel de agua del embalse, basado en redes neuronales, para la mejora de la planificación de producción de energía en la Central Hidroeléctrica Agoyán.," 2017.

[9] J. R. Azagra, "Control robusto cuantitativo de sistemas con múltiples entradas de actuación y una salida objeto de control," 2017.

[10] P. Isasi Viñuela and I. M. Galván León, Redes de Neuronas Artificiales: Un Enfoque Práctico. Pearson Educación, 2004.

[11] G. Asqui Santillán, D. Olmedo Vizueta, and J. Hernández Ambato, "Modelamiento basado en Redes Neuronales Artificiales para la Predicción de Recursos Hídricos en una Central Hidroeléctrica," in V Congreso Internacional de la Ciencia, Tecnología, Emprendimiento E Innovación, 2018, pp. 532–545.

[12] J. M. Zaldívar, E. Gutiérrez, I. M. Galván, F. Strozzi, and A. Tomasin, "Forecasting high waters at Venice Lagoon using chaotic time series analysis and nonlinear neural networks," J. Hydroinformatics, vol. 2, no. 1, pp. 61–84, 2000.

[13] J. L. Correa-Figueroa, E. Morales-Sánchez, J. A. Huerta-Ruelas, J. J. González-Barbosa, and C. R. Cárdenas-Pérez, "Sistema de adquisición de señales SEMG para la detección de fatiga muscular," Rev. Mex. Ing. Biomed., vol. 37, no. 1, pp. 17–27, 2016.

[14] F. Chollet, "Optimizers - Keras Documentation," 2015.

[15] F. Chollet, Deep Learning with Python, vol. 80, no. 1. 2007.

[16] T. Dozat, "Incorporating Nesterov Momentum into Adam," ICLR Work., no. 1, pp. 2013–2016, 2016.

[17] A. T. Hadgu, A. Nigam, and E. Diaz-Aviles, "Large-scale learning with AdaGrad on Spark," in Big Data (Big Data), 2015 IEEE International Conference on, 2015, pp. 2828–2830.

[18] S. Ruder, "An overview of gradient descent optimization algorithms," pp. 1–14, 2016.

[19] M. D. Zeiler, "ADADELTA: An Adaptive Learning Rate Method," 2012.

[20] M. C. Mukkamala and M. Hein, "Variants of RMSProp and Adagrad with Logarithmic Regret Bounds," CoRR, vol. abs/1706.05507, 2017.

[21] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," CoRR, vol. abs/1412.6, 2014.

[22] X. Zeng, Z. Zhang, and D. Wang, "AdaMax Online Training for Speech Recognition," 2016.

[23] S. J. Reddi, S. Kale, and S. Kumar, "On the Convergence of Adam and Beyond," in ICLR 2018, 2018, pp. 1–23.

[24] E. Morales and J. González, "Aprendizaje Computacional," Inst. Nac. Astrofísica, Óptica y Electrónica, pp. 1–232, 2011.

[25] B. Cecila, "Facultad de Ciencias Forestales," 2002.

[26] J. A. Villalpando-García, A. Castillo-Morales, M. E. Ramírez-Guzmán, G. Rendón-Sánchez, and U. Larqué-Saavedra, Mario, "COMPARACIÓN DE LOS PROCEDIMIENTOS DE TUKEY, DUNCAN, DUNNETT, HSU Y BECHHOFER PARA SELECCIÓN DE MEDIAS," Agrociencia, vol. 35, no. 1, pp. 79–86, 2001.