Revista Iberoamericana de Automática e Informática Industrial

Revista Iberoamericana de Automática e Informática Industrial 22 (2025) 21-32

www.revista-riai.org

Optimización y extensión del algoritmo de codificación-decodificación basado en PWM para Redes Neuronales de Impulsos.

Sergio Lucas, Eva Portillo*, Itziar Cabanes

Departamento de Ingeniería de Sistemas y Automática, Escuela de Ingeniería de Bilbao, Universidad del País Vasco (UPV/EHU), C/ Plaza Ingeniero Torres Quevedo nº 1, 48013, Bilbao, España.

To cite this article: Lucas, S., Portillo, E., Cabanes, I. 2025. Optimization and extension of the PWM-based encoding-decoding algorithm for Spiking Neural Networks. Revista Iberoamericana de Automática e Informática Industrial 22, 21-32. https://doi.org/10.4995/riai.2024.20836

Resumen

Las Redes Neuronales de Impulsos (*Spiking Neural Networks*, SNNs) son modelos neuronales que procesan la información en forma de *spikes* o series de impulsos en el dominio del tiempo, posibilitando el consumo ultrabajo. Sin embargo, debido a que la mayoría de los procesos reales manejan magnitudes físicas de tipo real, para emplear este tipo de redes es necesario el uso de algoritmos de codificación y decodificación. El algoritmo de codificación basado en modulación por ancho de pulso (*Pulse Width Modulation*, PWM) es un novedoso algoritmo temporal de codificación que supera con creces la precisión de sus algoritmos predecesores a la hora de construir y reconstruir la señal original. A pesar de sus múltiples ventajas, este algoritmo presenta una serie de limitaciones: (a) requiere de dos valores consecutivos de la serie temporal original para poder codificar, lo cual imposibilita su uso en campos donde no existan relaciones cronológicas, como puede ser el tratamiento de imágenes; y (b) presenta posibilidades de ser optimizado computacional y energéticamente. Así, en este trabajo se presentan dos nuevas propuestas basadas en este algoritmo de codificación y decodificación que solventan las limitaciones mencionadas. Cabe destacar que ambas propuestas permiten reducir en más del doble el coste computacional y energético de los procesos de codificación y decodificación.

Palabras clave: Diseño de metodologías, Validación de modelos, Redes Neuronales, Formulación de modelos, diseño de experimentos, Modelado de series temporales.

Optimization and extension of the PWM-based encoding-decoding algorithm for Spiking Neural Networks.

Abstract

Spiking Neural Networks (SNNs) are neural models that process the information in temporal spike sequences or spike trains, enabling ultra-low energy consumption. However, since most of the real processes handle real value information, SNNs require the use of encoding-decoding algorithms. The Pulse Width Modulation (PWM) based encoding-decoding algorithm is a novel temporal encoding algorithm that overtakes its predecessor algorithms in terms of precision. Despite its many advantages, this algorithm presents some limitations: (a) it requires two consecutive values of the original time-series to encode, which makes it impossible to use it in fields where there are no chronological relationships, such as image processing; and (b) it presents possibilities of being optimized computationally and energetically. In this sense, this paper proposes two new proposals of the PWM-based encoding-decoding algorithm in order to overcome the aforementioned limitations. In addition, the new proposals reduce more than double the computational and energy cost of encoding and decoding processes.

Keywords: Design methodologies, Model validation, Neural Networks, Model formulation, experiment design, Time series modelling.

*Autor para correspondencia: eva.portillo@ehu.eus

1. Introducción

Con la irrupción de la Inteligencia Artificial (IA) y de las técnicas de aprendizaje automático ha aumentado significativamente el interés de la comunidad científica en diseñar y desarrollar modelos cada vez más precisos. Sin embargo, en la mayoría de los trabajos la eficiencia energética de los modelos está considerada como un criterio secundario o incluso en algunos de ellos ni siquiera se ha contemplado en el diseño (García-Martín et al., 2019). Así, trabajos publicados recientemente (de Vries, 2023)(Suetake et al., 2023) han señalado el grave problema que puede llegar a existir entre la IA y el consumo energético global si su uso sigue creciendo de forma incontrolada. Por tanto, resulta apremiante comenzar a emplear algoritmos de IA y arquitecturas hardware que permitan conseguir un compromiso aceptable y sostenible entre precisión y eficiencia.

En este sentido, las Redes Neuronales de Impulsos (*Spiking Neural Networks*, SNNs) son uno de los algoritmos de IA que más popularidad han ganado en los últimos años. Las SNNs, también conocidas como la tercera generación de redes neuronales artificiales, son modelos neuronales que tratan de imitar con mayor fidelidad el comportamiento de las neuronas biológicas. Para ello, a diferencia de las Redes Neuronales Artificiales tradicionales (*Artificial Neural Networks*, ANNs), las SNNs codifican la información mediante series de impulsos o *spikes*. Esta nueva forma de codificar implica que la información no va contenida en la forma de la acción potencial, sino en el número y la distribución temporal de los *spikes* (Gerstner and Kistler, 2002).

A pesar de la gran popularidad de las ANNs (Mesanza et al., 2020)(Vermander et al., 2023)(Nakai and Nishimoto, 2023), trabajos recientes (Deng et al., 2020)(Bu et al., 2022) señalan que bajo ciertas condiciones las SNNs son capaces de mantener o mejorar la precisión en los resultados de las ANNs y, además, presentar ventajas en cuanto a la eficiencia computacional y energética. Una de las principales razones de esta afirmación reside en que las SNNs se pueden considerar como una técnica basada en eventos. Así, en las SNNs las neuronas que no reciben *spikes* permanecen inactivas, haciendo que su consumo computacional y energético sea nulo. Otra de las razones se fundamenta en las implementaciones hardware de las redes neuronales, donde uno de los principales costes computacionales está ligado al proceso de multiplicación de los pesos por las entradas. En el caso de las SNNs, al consistir las entradas en 0 y 1, dicho coste se puede omitir. De hecho, a diferencia de las ANNs, las SNNs presentan importantes ventajas para ser implementadas en hardware neuromórfico con consumo ultrabajo (Sboev et al., 2016).

En lo que respecta a la eficiencia de las SNNs, un aspecto que se debe tener en cuenta es que en su cálculo no sólo influye cómo funciona o esté implementada la propia red, sino que el algoritmo de codificación y decodificación empleado también juega un papel importante. Dado que la mayoría de los datos de los procesos reales son de tipo real, en las SNNs resulta necesario incorporar una etapa previa a la introducción de los datos en la red para codificar los datos de tipo real en *spikes*. Además, en múltiples aplicaciones de ámbitos como el industrial o la bioingeniería, resulta necesario poder obtener la salida de la red en las unidades/magnitudes reales correspondientes por distintos motivos, por ejemplo monitorización o control. Así, resulta interesante que las técnicas o algoritmos de codificación ofrezcan también la posibilidad de decodificar con precisión la salida de las SNNs.

Actualmente, dentro de los algoritmos de codificación existentes para SNNs se pueden diferenciar dos enfoques: codificación basada en frecuencia (*rate coding*) o codificación basada en tiempo (*temporal coding*). La codificación frecuencial se basa en codificar la información basándose en el número de *spikes* dentro de una ventana de tiempo. Dentro de este enfoque destacan los siguientes algoritmos: distribución de Poisson (Xu et al., 2017), *Ben's Spike Algorithm* (BSA) (Brusca et al., 2019) o *Adaptative Threshold-Based* (ATB) (Laña et al., 2018). Por otro lado, la codificación temporal emplea la distancia que hay entre *spikes* para codificar la información. En este caso los algoritmos más comunes son *Time-to-First-Spike* (Chen et al., 2016) o *Rank Code* (Hong et al., 2020).

Pese a que estudios recientes han afirmado que la codificación temporal (a) preserva mayor cantidad de información que la codificación frecuencial (Lopes-dos Santos et al., 2015), y (b) presenta mayores ventajas para aprovechar el consumo ultrabajo de las SNNs (Xu et al., 2020), esta clase de algoritmos suelen estar limitados por la precisión en la localización de los *spikes* durante el proceso de codificación, provocando que a menudo no se contemple el proceso de decodificación o que si lo contemplan la salida decodificada presente errores significativos. Esto conlleva que dentro del marco de las SNNs no haya un algoritmo de codificación y decodificación por excelencia o comúnmente utilizado.

En este sentido, en (Arriandiaga et al., 2020) se presentó un novedoso algoritmo de codificación-decodificación temporal basado en la modulación por ancho de pulso (*PulseWidth Modulation*, PWM) que supera con creces la precisión en la codificación y decodificación de sus algoritmos predecesores. A pesar de sus múltiples ventajas, este algoritmo presenta una serie de limitaciones: i) requiere de dos puntos cronológicamente consecutivos de la serie temporal para codificar un sólo *spike*, lo cual limita su campo de aplicación a problemas donde existan relaciones cronológicas; ii) al requerir de dos puntos consecutivos dificulta su implementación en tiempo real; iii) el algoritmo requiere de la interpolación de toda la serie temporal, lo cual supone un elevado coste computacional y de memoria.

Así, el objetivo principal de este trabajo es promover el uso del algoritmo de codificación-decodificación basado en PWM como algoritmo a implementar junto a las SNNs. Para ello, en este trabajo se presentan dos nuevas propuestas del algoritmo que solventan las limitaciones mencionadas. La primera propuesta consiste en una optimización del algoritmo original, la cual, manteniendo la precisión en los resultados de su algoritmo predecesor, presenta ventajas significativas en cuanto a eficiencia computacional y energética. Por otro lado, la segunda propuesta se basa en una nueva extensión que permite ampliar el uso del algoritmo de codificación-decodificación a otros campos donde no existan relaciones cronológicas como el procesamiento de imágenes, incorporando también la optimización del coste computacional y energético de la primera propuesta. Ambas propuestas se validan codificando y decodificando dos series temporales y comparando sus resultados con los obtenidos con el algoritmo original en términos de precisión y eficiencia.

Las dos series temporales empleadas son: (1) una señal senoidal totalmente conocida, periódica y con una dinámica lenta, y (2) la serie temporal *ARCTIC*, la cual se ha extraído de un repositorio UCI y cuenta con una dinámica más rápida. Además, la segunda propuesta se valida en la codificación y decodificación de imágenes para verificar que es capaz de codificar y decodificar en campos de aplicación donde no existen relaciones cronológicas.

El resto del artículo está estructurado de la siguiente forma: en la sección 2 se presentan los fundamentos de las SNNs. En la sección 3 se describe el funcionamiento de la versión original del algoritmo de codificación-decodificación basado en PWM. En la sección 4 se presenta tanto la propuesta optimizada del algoritmo para series temporales como la nueva extensión para otros campos de aplicación. En la sección 5 se validan ambas nuevas propuestas. En la sección 6 se realiza la discusión a la vista de los resultados obtenidos. Por último, en la sección 7 se exponen las conclusiones del trabajo.

2. Redes Neuronales de Impulsos

Desde sus inicios las redes neuronales artificiales han tratado de imitar la estructura de las redes biológicas. Sin embargo, el funcionamiento y la eficiencia de las neuronas artificiales empleadas hasta la fecha en los modelos ANNs dista mucho del de las neuronas reales. Con el objetivo de imitar con mayor fidelidad el funcionamiento de las neuronas biológicas han surgido las SNNs, también conocidas como la tercera generación de redes neuronales artificiales.

La principal diferencia entre las ANNs y las SNNs radica en que, en vez de valores reales, en las SNNs la información está codificada en trenes de impulsos o *spikes* al igual que ocurre con las neuronas biológicas. Esta diferencia repercute directamente en los modelos de neuronas artificiales que se emplean y en cómo funcionan.

En el caso de los modelos neuronales de las ANNs, para emitir una salida (Y) las neuronas aplican una función de activación ($f(x)$) al término bias (b) y al sumatorio de las entradas reales (x_i) que recibe cada neurona multiplicada por el peso de sus interconexiones (w_{ij}), tal y como se ve en la Figura 1a.

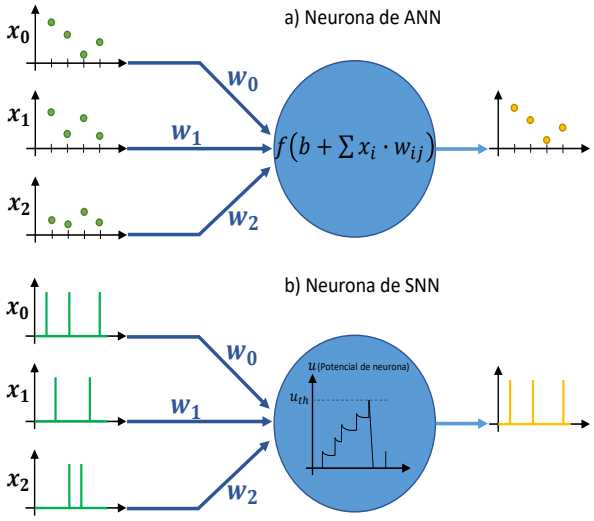

Figura 1: ANNs vs SNNs.

Sin embargo, en el caso de las SNNs la información tanto a la entrada como a la salida de las neuronas viene representada mediante *spikes*. Esta forma de codificar la información no se puede procesar mediante modelos neuronales basados en funciones de activación, sino mediante ecuaciones diferenciales.

Existe una gran variedad de modelos neuronales basados en ecuaciones diferenciales para SNNs (Han and Lee, 2021), siendo los más conocidos los modelos Integrate-and-Fire (IF), Leaky Integrate-and-Fire (LIF), Spike Response Neuron Model (SRNM) o Izhikevich. El modelo LIF (Gerstner et al., 2014) es el modelo más utilizado en SNNs debido a su simplicidad y bajo coste computacional (Yamazaki et al., 2022), por lo que es el modelo escogido para este trabajo.

A pesar de que las ecuaciones diferenciales varían dependiendo del modelo neuronal escogido, en términos generales la dinámica de cualquier modelo neuronal de SNN en tiempo discreto se puede describir mediante las siguientes tres ecuaciones diferenciales:

$$H^t = f(V^{t-1}, X^t) \tag{1}$$

$$S^t = \Theta(H^t - V_{th})) \tag{2}$$

$$V^t = H^t(1 - S^t) + V_{rest} \cdot S^t \tag{3}$$

La Ecuación 1 describe cómo el potencial de cada neurona se carga o descarga en cada instante dependiendo de si se ha recibido como entrada (X^t) un *spike* o no. La forma de cargarse o descargarse lo describe una función (f) que depende de cada modelo neuronal. El modelo LIF tiene un término "*leaky*"que hace que su descarga se realice en forma de exponencial, de manera que su comportamiento es el de un sistema de primer orden ante entrada impulso. De hecho, la representación del modelo LIF se suele basar en el circuito RC (ver Figura 2).

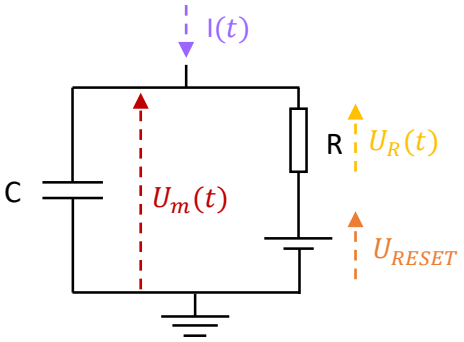

Figura 2: Representación de la neurona LIF mediante circuito RC.

En cada instante de tiempo las neuronas comparan su valor de carga (H^t) con un valor de tensión umbral (V_{th}). En caso de sobrepasar V_{th}, dicha neurona emite un *spike*. En caso contrario emite un 0. Este comportamiento se define mediante la Ecuación 2, donde Θ es la función escalón unitario definida mediante la Ecuación 4.

$$\Theta(x) = \begin{cases} 1, \ if \ x \geq 0 \\ 0, \ if \ x < 0 \end{cases} \tag{4}$$

Por último, la Ecuación 3 define el potencial residual a la salida de la neurona, el cual será el valor potencial para calcular

el valor de carga en la siguiente iteración (ver Ecuación 1). Así, si en una iteración la neurona no dispara, su valor residual es el valor de carga (H^t) alcanzado en dicha iteración. Sin embargo, si la neurona dispara, automáticamente el valor de su potencial se resetea a un potencial de reposo (V_{rest}).

La salida de una SNN se consigue al aplicar estas ecuaciones a todas las neuronas y propagar los *spikes* capa tras capa. En la Figura 3 se muestra una SNN con neuronas LIF tanto en la capa oculta como en la capa de salida. Al igual que ocurre con las ANNs, las neuronas de la capa de entrada de las SNNs sólo se encargan de propagar hacia adelante los datos codificados.

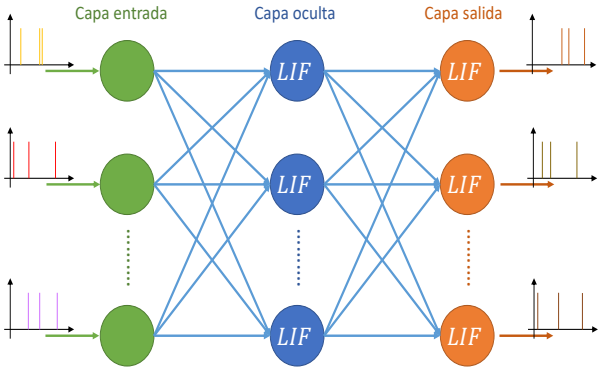

Figura 3: Arquitectura SNN totalmente conectada.

3. Versión original del algoritmo de codificación-decodificación basado en PWM

Tal y como se ha comentado anteriormente, una de las principales características de las SNNs es que se trata de una técnica computacional y energéticamente eficiente. Por tanto, como las SNNs dependen de las etapas de codificación y decodificación para transformar los datos reales en *spikes* es necesario que el algoritmo de codificación-decodificación sea lo más eficiente posible.

El algoritmo de codificación-decodificación basado en PWM (Arriandiaga et al., 2020) es un novedoso algoritmo de codificación y decodificación inspirado en los fundamentos teóricos de la técnica PWM. Durante la codificación, este algoritmo emplea una señal portadora o *carrier* como base temporal para localizar cuándo se dan las intersecciones entre dicha señal y la serie temporal original, dando lugar a los *spikes* que se introducen como entrada en la SNN. Por otro lado, durante la decodificación el algoritmo emplea la misma señal portadora (misma base temporal) para localizar las intersecciones entre dicha señal y una secuencia de *spikes*, lo cual da lugar a la estimación de puntos discretos de la señal original reconstruida. Cabe destacar que el algoritmo se diseñó para series temporales digitalizadas a una frecuencia de muestreo correctamente seleccionada por la persona experta de dominio (Arriandiaga et al., 2020), lo que permite asegurar una buena precisión en las etapas de codificación y decodificación.

En la Figura 4 se muestra de forma resumida el método original de codificación presentado en Arriandiaga et al. (2020), el cual va a ser el objetivo principal a optimizar en este trabajo.

Nótese que en el paso 1 de la Figura 4 la serie temporal original se normaliza dentro del rango $[min, max]$, mientras que en el paso 2 la señal portadora se normaliza dentro del rango $[min - \Delta, max + \Delta]$. Se trata de una contribución respecto a la propuesta presentada en (Arriandiaga et al., 2020) con el fin de asegurar que en todos los instantes se produzca una intersección. En el caso de que ambas señales se normalizaran para tomar valores dentro del mismo rango se podría dar la situación en la que en un instante los dos puntos consecutivos de la serie temporal original coincidiesen exactamente con los valores límite mínimo y máximo de la señal portadora, produciéndose un solapamiento de las señales y, por tanto, no existiendo ninguna intersección ni ningún *spike* en dicho instante.

Teniendo en cuenta cómo funciona el algoritmo, definir correctamente la señal portadora es esencial para la metodología. Para ello, se emplean dos hiperparámetros que son definidos por las personas expertas del dominio de aplicación:

- El número de ondas de la señal portadora (*number of carrier waves*, nc): define el número de intersecciones que pueden ocurrir entre las diferentes señales. Este hiperparámetro está directamente relacionado con la anchura de la señal portadora, de manera que el nc adecuado viene determinado por el número de puntos (N) capturados a la frecuencia de muestreo seleccionada por la persona experta de dominio (Arriandiaga et al., 2020).

- El número de puntos de onda de la señal portadora (*number of points per carrer wave*, npc): define el número de valores por el que está formada cada onda de la señal portadora. Por tanto, define la resolución de la codificación y decodificación.

Ambos hiperparámetros se representan en el paso 2 de la Figura 4. Cabe destacar que el hiperparámetro npc permite definir un compromiso entre la eficiencia y la resolución del algoritmo. En la Figura 5 se muestra la diferencia que se produce al codificar la misma señal empleando dos señales portadoras que sólo difieren en el valor de npc empleado. Se puede ver cómo al emplear un valor de npc menor (caso de npc=4), la señal portadora está formada por un menor número de puntos y, por tanto, trabajar con esta señal conlleva menores costes computacionales, de memoria y energéticos. Sin embargo, también conlleva mayores pérdidas de resolución a la hora de emitir los *spikes*, provocando una codificación y decodificación menos precisa que la que se obtiene empleando npc mayores. Por tanto, se trata de una decisión de compromiso. Si el campo de aplicación requiere de una codificación y decodificación muy precisa es necesario emplear un valor de npc elevado. Si por el contrario no es necesaria una precisión elevada y se quiere premiar la eficiencia se puede optar por valores de npc menores.

El algoritmo de codificación-decodificación basado en PWM presenta importantes ventajas respecto a otros algoritmos temporales en cuanto a la simplicidad del método, la precisión obtenida en la reconstrucción de señales y la oportunidad de fijar por la persona usuaria mediante el hipeparámetro npc un compromiso entre el coste computacional del método y su resolución (Arriandiaga et al., 2020). Además, es posible aplicarlo a cualquier tipo de serie temporal con independencia de sus características (Lucas and Portillo, 2024), así como a series temporales multivariables. Para este último caso, por cada variable es necesario crear una señal portadora que se adapte a sus

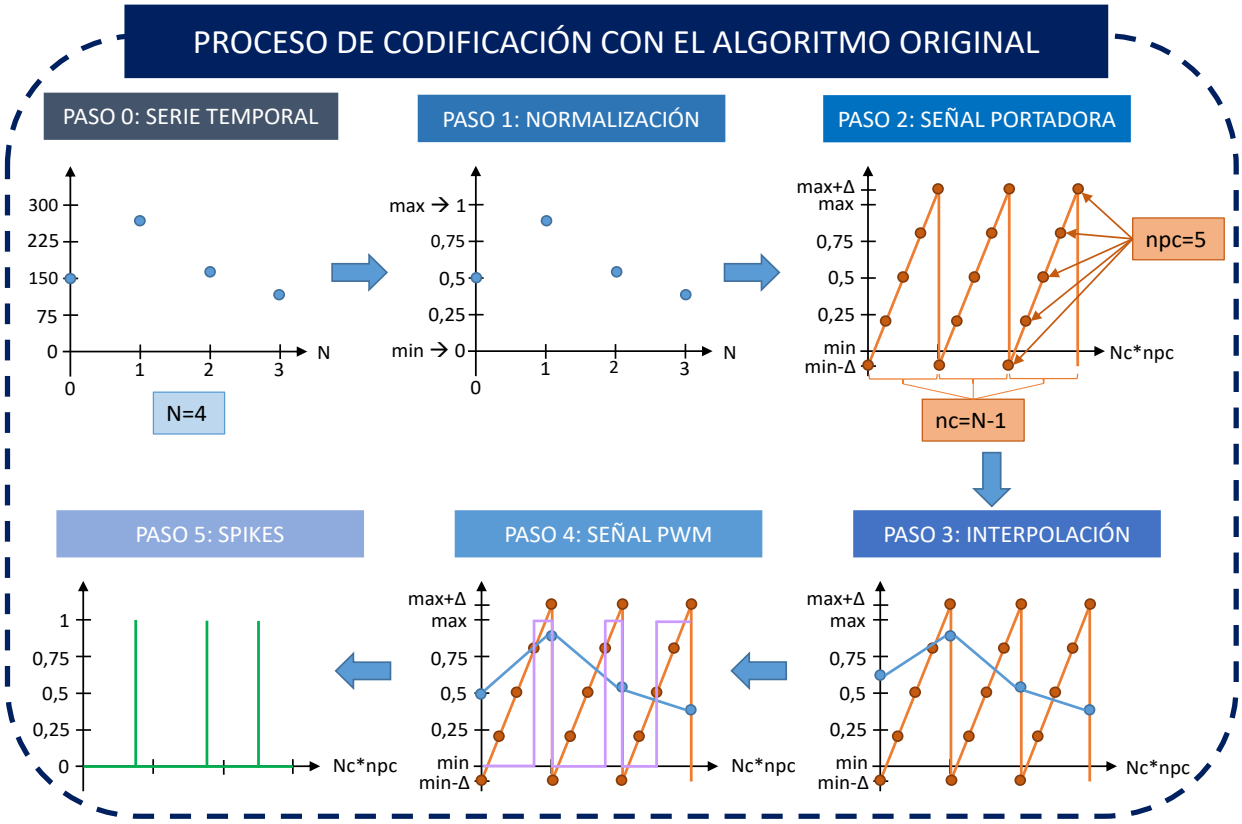

Figura 4: Algoritmo original basado en PWM: método de codificación.

características concretas. Sin embargo, este algoritmo también presenta una serie de limitaciones, especialmente en la parte de la codificación. Tal y como se puede observar en la Figura 4, el algoritmo en su versión original para codificar requiere de la interpolación de dos puntos consecutivos de la serie temporal para poder calcular el instante de intersección con la señal portadora. Este hecho: a) limita el uso del algoritmo a problemas donde exista una relación cronológica; b) requiere que la señal original se normalice en el rango [min, max] y la señal portadora en el rango [min-Δ, max +Δ] para garantizar que se produzcan intersecciones y, por tanto, *spikes* en todos los instantes; c) aumenta el coste computacional del método al tener que realizar una interpolación entre cada dos puntos consecutivos de la serie temporal original para después calcular la intersección entre dos rectas; y d) dificulta la implementación del algoritmo en tiempo real al requerir de 2 valores consecutivos para codificar.

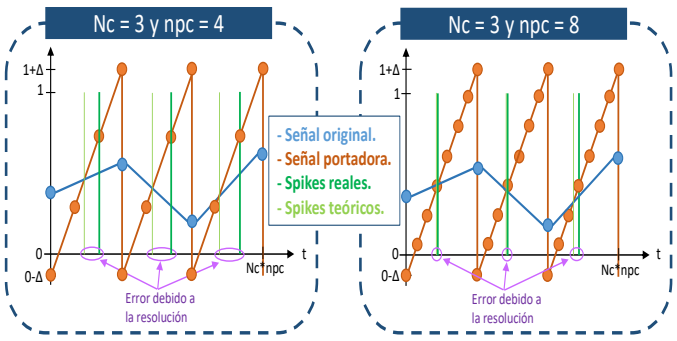

Figura 5: Diferencia al emplear señales portadoras con diferente valor de npc.

4. Optimización y extensión del algoritmo de codificación-decodificación basado en PWM

Con el objetivo de solventar las ya mencionadas limitaciones del algoritmo original de codificación-decodificación basado en PWM (V_{or}), en este apartado se va a presentar: a) una optimización del algoritmo original (V_{opt}) que, manteniendo la precisión de V_{or}, da lugar a menores costes computacionales y energéticos; b) una extensión optimizada (V_{ext}) que se pueda implementar en campos de aplicación donde no existan relaciones cronológicas, como es el caso del tratamiento de imágenes.

4.1. *Optimización de la versión original (V_{opt})*

Originalmente el objetivo principal de V_{or} fue presentar ventajas reseñables en cuanto a la precisión de la codificación y decodificación de señales frente a sus algoritmos predecesores, siendo ésta una contribución significativa en el ámbito de las SNNs y de los algoritmos temporales de codificación-decodificación (Arriandiaga et al., 2020). Sin embargo, esta versión original presenta posibilidades de ser optimizada tanto computacional como energéticamente.

Una de las principales áreas de mejora consiste en la simplificación de las operaciones. Tal y como se puede ver en los pasos 2, 3 y 4 de la Figura 4, V_{or} emplea señales completas (señales formadas por $nc \cdot npc$ puntos) durante todo el proceso de codificación. Esta forma de codificar conlleva costes computacionales y energéticos significativos al tener que acceder a grandes cantidades de datos en la memoria.

En V_{opt} se reduce estos costes codificando cada instante por separado. Así, esta optimización del algoritmo original codifica

los valores reales (eje y) en el dominio temporal (eje x) simplemente mediante la localización del instante en que se produce la intersección entre una recta genérica que define a la señal portadora y la recta formada por dos puntos consecutivos de la serie temporal original. En la Figura 6 se muestra el proceso de codificación de V_{opt}, el cual está dividido en los siguientes pasos:

PASO 1: Se define la señal portadora mediante la ecuación genérica de una recta. Esto es posible debido a que la señal portadora que comúnmente se utiliza en el algoritmo de codificación basado en PWM es la señal diente de sierra, la cual para cada instante de tiempo se puede definir mediante la ecuación de una recta ($y = m \cdot x + b$).

La *pendiente de la recta (m)* y el *término independiente (b)* de esta ecuación son totalmente conocidos ya que: i) para cada instante la señal portadora está formada por npc valores ($x \in [0, ..., npc - 1]$); ii) tal y como se puede ver en el paso 1 de la Figura 6, la señal portadora se normaliza dentro del rango de valores máximo y mínimo de la serie temporal original, lo cual permite conocer dos de los valores reales (y) de la recta. Nótese que al igual que en V_{or} es necesario asegurar que en todos los instantes de tiempo se va a producir una intersección entre la señal original y la señal portadora. Con el propósito de perder la mínima resolución posible, a la hora de construir la ecuación genérica de la señal portadora se asegura que exclusivamente un valor de npc quede por encima y por debajo de los valores máximo y mínimo de la serie temporal original, respectivamente (ver paso 1 de la Figura 6). Esta modificación supone una mejora directa en la resolución del algoritmo respecto a V_{or}, ya que el término Δ empleado por V_{or} puede incluir más de un valor de npc por encima y por debajo de los valores máximo y mínimo de la serie temporal.

Así, es posible definir completamente las coordenadas (x,y) de dos puntos de la señal portadora: $(1, minimo)$ y $(npc - 2, maximo)$. Estas coordenadas permiten definir la ecuación genérica de la señal portadora:

$$y = \frac{(maximo - minimo) \cdot (x - 1)}{npc - 3} + minimo \tag{5}$$

PASO 2: Para cada instante (i) se define la recta formada por dos valores consecutivos ($data[i]$, $data[i + 1]$) de la serie temporal original, siendo la ecuación genérica de esta recta:

$$y = \frac{data[i + 1] - data[i]}{npc} \cdot x + data[i] \tag{6}$$

PASO 3: La localización de los *spikes* en el dominio temporal (eje x) se calcula mediante la intersección de la recta genérica que define a la señal portadora (Ecuación 5) con cada una de las rectas calculadas a través de puntos de la serie temporal original (Ecuación 6).

Un aspecto que se debe tener en cuenta en este paso es que en el caso de V_{opt} el *spike* se emite en el valor de npc más cercano a la intersección, mientras que, tal y como se puede ver en los pasos 4 y 5 de la Figura 4, en el caso de V_{or} el *spike* siempre se emite en el valor de npc inmediatamente posterior al punto de intersección. Esta modificación de seleccionar el valor más próximo a priori debería dar lugar a una mayor precisión en la salida y, por tanto, en la decodificación.

PASO 4: Por último, en caso de ser necesario, la señal completa codificada se puede construir a partir de todas las localizaciones de *spikes* calculadas en el paso anterior y sabiendo que para cada instante hay npc valores.

En cuanto al proceso de decodificación con V_{opt}, los puntos discretos reconstruidos de la serie temporal se obtienen directamente al introducir en la Ecuación 5 la localización del *spike* en el dominio temporal (eje x) de cada instante, siendo $x \in [0, ..., npc - 1]$. Señalar que con esta forma de decodificar también se elimina el uso de señales completas.

En resumen, comparando V_{opt} frente a V_{or} se observan las siguientes mejoras: i) se define la señal portadora en función de los valores máximo y mínimo de la serie temporal original, lo cual permite prescindir del proceso de normalización; ii) no se guardan en memoria ni se emplean señales formadas por $nc \cdot npc$ puntos; iii) se elimina el proceso de interpolación durante la codificación; iv) en la codificación, el cálculo de intersecciones punto a punto se sustituye por la intersección entre dos rectas genéricas definidas exclusivamente por dos parámetros; v) la definición del instante donde se dan las intersecciones tanto en la codificación como en la decodificación es más precisa. Todo ello contribuye a que V_{opt} presente importantes reducciones en consumo de recursos (reducción de costes computacionales, de memoria y energéticos) como mejoras en la precisión del algoritmo.

4.2. Extensión a otros campos (V_{ext})

La premisa que debe cumplir la extensión del algoritmo para que pueda ser aplicado en otros campos es que durante la codificación el algoritmo no requiera de dos puntos consecutivos de la serie temporal. Bajo esta condición V_{ext} codifica los valores reales (eje y) en el dominio temporal (eje x) simplemente mediante la localización del instante en el que la señal portadora pasa por el valor original. En la Figura 7 se muestra el proceso de codificación de V_{ext}, el cual está dividido en los siguientes pasos:

PASO 1: Al igual que en V_{opt}, en V_{ext} se prescinde de la etapa de normalización y se comienza calculando la ecuación genérica de la señal portadora. Sin embargo, en este caso es posible definir la señal portadora directamente con una amplitud igual a la de la serie temporal original, tal y como se puede ver el paso 1 de la Figura 7. Nótese que al igualar las amplitudes de la señal original y la señal portadora se está aprovechando al completo la resolución seleccionada mediante el parámetro npc del algoritmo. Esto se debe a que ya no es posible que exista una recta intersección de pendiente igual a la de la señal portadora. Así, la ecuación genérica de la señal portadora para esta nueva extensión queda definida por la Ecuación 7.

$$y = \frac{maximo - minimo}{npc - 1} + minimo \tag{7}$$

PASO 2: Partiendo de la premisa citada anteriormente, la localización de los *spikes* en el dominio temporal (eje x) se puede calcular directamente introduciendo en la Ecuación 7 el valor real (eje y) de la serie temporal en un instante determinado ($data[i]$) y despejando la variable x.

PASO 3: Al igual que en el caso de V_{opt}, en caso de ser necesario, la señal completa codificada se puede construir a partir

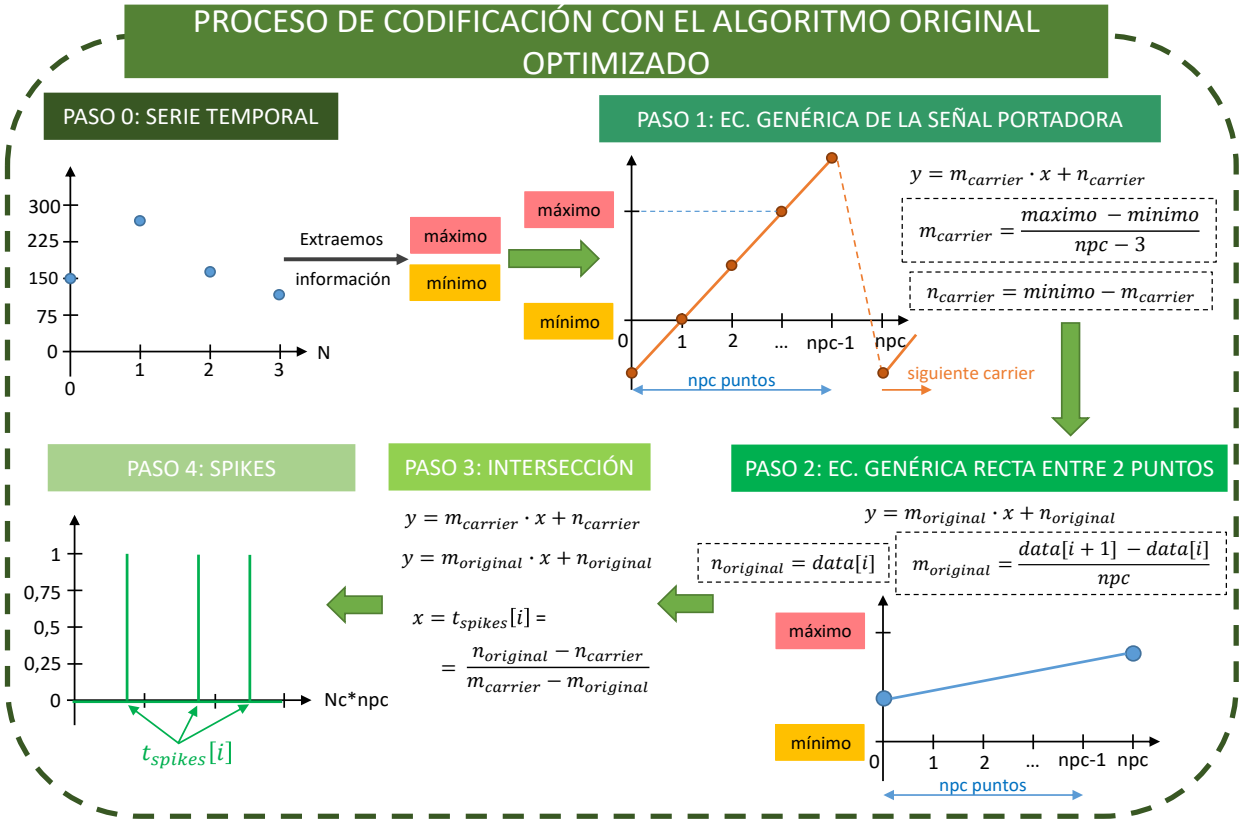

Figura 6: Procedimiento para codificar con el algoritmo original optimizado basado en PWM.

de todas las localizaciones de *spikes* calculadas en el paso anterior y teniendo en cuenta que para cada instante hay npc valores.

En cuanto a la decodificación, el proceso es idéntico que V_{opt}. Los puntos discretos reconstruidos de la serie temporal original se obtienen directamente al introducir en la Ecuación 7 la localización del *spike* en el dominio temporal (valor x) de cada instante, siendo $x \in [0, ..., npc - 1]$.

Comparando V_{ext} y V_{or} se pueden apreciar importantes ventajas: a) el hecho de sustituir tanto la interpolación como el cálculo de intersecciones entre rectas por el cálculo de una recta que pasa por un punto, permite disminuir el coste computacional del método; b) la codificación de cada *spike* depende exclusivamente del valor real de un único instante de la serie temporal, lo cual hace que esta nueva versión del algoritmo sea fácilmente implementable en tiempo real; y c) el punto codificado sólo depende de su correspondiente valor real, eliminando así cualquier relación cronológica con otros puntos y haciendo que la nueva extensión del algoritmo se pueda emplear no sólo con series temporales sino con imágenes u otro tipo de aplicaciones.

5. Validación

En esta sección se presenta la validación de las dos nuevas propuestas del algoritmo de codificación-decodificación basado en PWM. Por un lado, se analiza la precisión de V_{opt} y V_{ext} para codificar y decodificar series temporales. Por otro lado, se comprueba que V_{ext} es capaz de aplicarse a campos donde no existan relaciones cronológicas. En este caso, V_{ext} se valida codificando y decodificando una imagen. La elección de este campo de aplicación se debe a que el tratamiento de imágenes es un campo de gran interés en las SNNs (Fang et al., 2021) (Yao et al., 2021).

5.1. *Series temporales*

La primera fase de la validación consiste en analizar los procesos de codificación y decodificación de las dos nuevas propuestas (V_{opt} y V_{ext}) en series temporales y comparar sus resultados con los del algoritmo original (V_{or}). Para ello, por cada propuesta se analiza: (a) la precisión de la codificación-decodificación; (b) el coste computacional; (c) el coste energético. Además, con el objetivo de analizar el grado de influencia de la resolución en la decodificación se consideran 3 valores distintos de npc por serie temporal, 32, 64 y 128.

Para cuantificar la precisión de la codificación-decodificación de cada propuesta se emplea la métrica del Error Cuadrático Medio (*Mean Square Error*, MSE). Esta métrica permite analizar las diferencias existentes entre la serie temporal original y la señal decodificada obtenida con cada una de las propuestas (V_{or}, V_{opt} y V_{ext}) mediante la siguiente ecuación:

$$MSE = \frac{1}{nc * npc} \sum_{i=1}^{nc*npc} (y_i - \hat{y}_i)^2 \quad (8)$$

donde y_i es el valor de cada una de las muestras de la serie temporal original e $\hat{y}$ es el valor de cada una de las muestras de la señal decodificada. Para comparar la eficiencia de cada una de las propuestas, se mide el coste computacional (CC) y el coste energético (CE) de codificar y decodificar con cada propuesta 30 veces y se calcula la media. Para ello, en el caso del coste

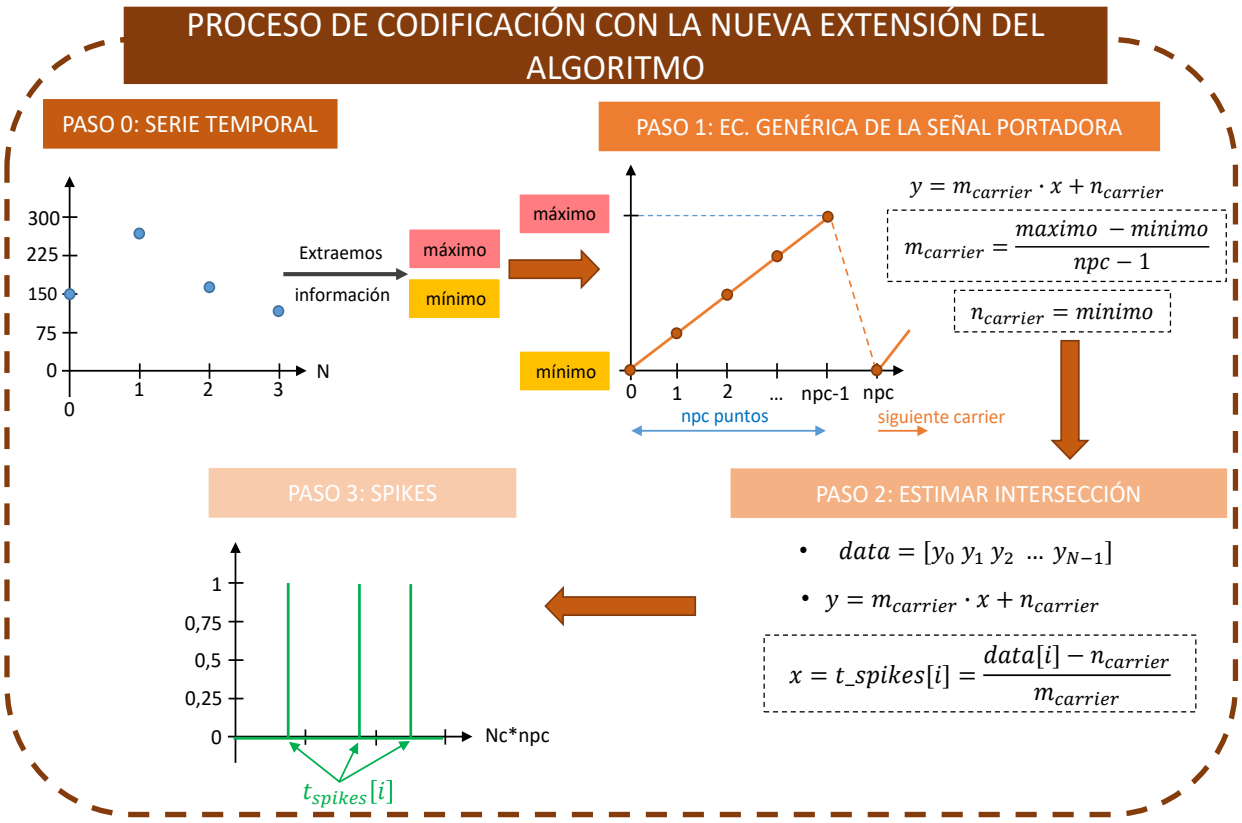

Figura 7: Procedimiento para codificar con la nueva extensión del algoritmo basado en PWM.

computacional se emplean relojes de la librería *time* de python, y para el coste energético se emplea la librería *PyJoules* también de python. Todas estas medidas se realizan en un P64C Intel(R) Core(TM) i7-4770K CPU @ 3.50GHz Linux 6.2.0-32-generic.

Las nuevas propuestas se validan en dos series temporales con dinámicas diferentes. La primera serie temporal utilizada es una señal senoidal periódica de frecuencia $f_o = 100Hz$, amplitud $A = 3$, frecuencia de muestreo $f_s = 60 \cdot f_o$ y número de ciclos 20. La segunda serie temporal es una señal extraída de un repositorio de la base de datos UCI que contiene grabaciones de voz de personas llevadas a cabo en *Carnegie Mellon University*. Esta base de datos es conocida como ARCTIC, la cual está disponible en (Black, 2003) y explicada en (Kominek and Black, 2003). Cada una de las voces de ARCTIC está grabada a una frecuencia de 32kHz, lo cual supone una dinámica relativamente muy superior a la de la primera serie temporal, la señal senoidal. La serie temporal escogida para este trabajo se corresponde a la voz guardada en el fichero "*arctic_a0001.wav*"situado dentro de la carpeta "*US English bdl (made)*". Dado el gran volumen de datos disponibles, se ha seleccionado el rango de datos comprendido entre la muestra 4985 y la muestra 7425 (2440 valores discretos).

En la Tabla 1 se muestran los errores MSE, la media de los costes computacionales en milisegundos (ms) y la media de los costes energéticos en julios (J) obtenidos con la señal senoidal. En general, para todas las propuestas (V_{or}, V_{opt} y V_{ext}) se puede observar que al aumentar la resolución (npc): (a) los errores MSE disminuyen, y (b) el coste computacional y energético aumentan. Es un resultado esperable ya que el parámetro npc define directamente el compromiso entre resolución y eficiencia, por tanto, al aumentar el npc se aumenta la precisión de la decodificación a costa de incrementar tanto el coste computacional como el energético. Comparando V_{opt} con V_{or}, V_{opt} mejora la precisión en la decodificación en el caso de npc igual a 32, y para valores superiores de npc la mantiene. Esto es debido a que para valores iguales o superiores a npc 64 ambos algoritmos convergen en los instantes donde se dan los *spikes*. En cuanto al coste computacional y energético, se puede concluir que V_{opt} presenta ventajas muy significativas, consiguiendo una reducción de más del doble en ambos costes. Por otro lado, V_{ext} presenta ventajas muy significativas en cuanto a eficiencia respecto a V_{or}. Sin embargo, los cambios introducidos en V_{ext} para poder aplicar el algoritmo a otros campos de aplicación reducen la precisión del algoritmo respecto a V_{or} con frecuencias de muestreo adecuadas, siendo la diferencia más notable a medida que se aumenta la resolución (npc) del algoritmo. Por último, comparando V_{ext} con V_{opt} se puede observar que en términos generales se consiguen costes computacionales y energéticos similares, si bien V_{opt} presenta mejores métricas de error. En la Figura 8 se muestra de forma gráfica la señal senoidal codificada y decodificada con todas las propuestas existentes del algoritmo y para un valor de npc igual a 128.

En la Tabla 2 se muestran las métricas de error y los costes computacionales y energéticos obtenidos con la serie temporal ARCTIC. Al igual que con la señal senoidal, independientemente de la versión empleada, al aumentar el valor de npc, el error MSE disminuye y los costes computacional y energético aumentan. Además, en este caso con V_{opt} se logran ventajas re-

Tabla 1: Métricas obtenidas para serie temporal de la señal senoidal.

		SENOIDAL			Diferencia porcentual	
		PWM original (V_{or})	PWM original opt. (V_{opt})	Extensión PWM (V_{ext})	V_{or} vs. V_{opt}	V_{or} vs. V_{ext}
npc = 32	MSE	0.00045	0.00005	0.00046	800 %	-2.17 %
	CC (ms)	245.8	86.6	86.5	183.83 %	184.16 %
	CE (J)	8.97	3.07	2.96	192.32 %	203.08 %
npc = 64	MSE	0.00011	0.00011	0.00044	0 %	-75 %
	CC (ms)	486.1	170.1	168.4	185.77 %	188.66 %
	CE (J)	14.58	5.74	5.10	153.89 %	185.69 %
npc = 128	MSE	0.00003	0.00003	0.00043	0 %	-93.02 %
	CC (ms)	966.3	334.7	333.6	188.71 %	189.66 %
	CE (J)	32.09	10.19	12.40	214.96 %	158.71 %

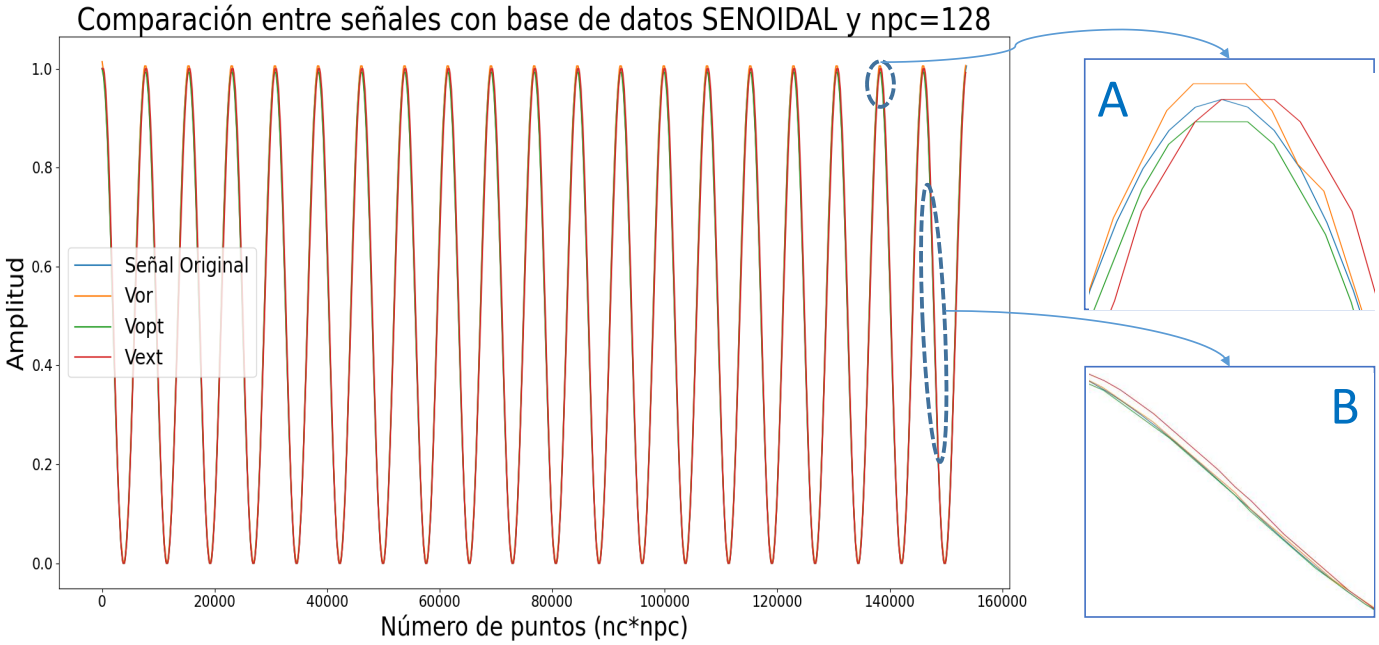

Figura 8: Codificación y decodificación de la señal senoidal con las diferentes versiones del algoritmo basado en PWM y con npc=128.

señables respecto a V_{or} tanto en la métrica de error MSE como en los costes computacionales y energéticos para cualquier valor de npc. En cuanto a V_{ext} se siguen consiguiendo ventajas significativas respecto a V_{or} en cuanto a eficiencia, pero se reduce la precisión en la codificación y decodificación de series temporales. Por último, en líneas generales los costes computacionales y energéticos son similares comparando V_{opt} y V_{ext}, si bien, al igual que con la señal senoidal, con V_{opt} se consiguen los mejores resultados de MSE. En la Figura 9 se muestra la señal ARCTIC codificada y decodificada con todas las propuestas existentes del algoritmo para un valor de npc igual a 32.

5.2. *Imágenes*

La segunda fase de la validación consiste en demostrar que V_{ext} es capaz de codificar y decodificar satisfactoriamente en campos de aplicación donde no existan relaciones cronológicas, como es el caso de las imágenes.

Para comprobar que V_{ext} es capaz de codificar y decodificar satisfactoriamente una imagen se emplea la métrica MSE definida por la Ecuación 8, donde en este caso y_i es el valor del píxel de la señal original e $\hat{y}$ es el valor del píxel de la imagen codificada y decodificada con la nueva extensión del algoritmo. Se debe señalar que en esta parte de la validación no es posible comparar la eficiencia de V_{ext} frente al resto de propuestas, por tanto, en esta ocasión no se miden costes computacionales ni energéticos. Por último, con el objetivo de realizar un análisis más detallado, el parámetro npc se varía entre los valores 2, 4, 8, 16, 32, 64, 128 y 256.

En la Figura 10 se muestra el resultado de la codificación y decodificación de las imágenes para todos los valores de npc mencionados anteriormente. Así, se puede observar que al aumentar la resolución del algoritmo (npc), la calidad de las imágenes mejora sustancialmente, lo cual también se corrobora con los valores MSE recogidos en la Tabla 3. El hecho que para npc igual a 256 se consiga un error nulo quiere decir que el proceso de codificación y decodificación con V_{ext} es correcto para la codificación y decodificación de imágenes, ya que la resolución de un píxel está en el rango [0-255]. Por tanto, V_{ext} permite convertir imágenes en *spikes* satisfactoriamente apro-

Tabla 2: Métricas obtenidas para serie temporal de la señal ARCTIC.

		ARCTIC			Diferencia porcentual	
		PWM original (V_{or})	PWM original opt. (V_{opt})	Extensión PWM (V_{ext})	V_{or} vs. V_{opt}	V_{or} vs.V_{ext}
npc = 32	MSE	0.00047	0.00008	0.00046	487.50 %	2.17 %
	CC (ms)	504.9	186.5	184.5	170.72 %	173.66 %
	CE (J)	18.46	6.82	5.70	170.58 %	223.85 %
npc = 64	MSE	0.00012	0.00002	0.00041	500 %	-70.73 %
	CC (ms)	1003.7	357.4	355.4	180.83 %	182.41 %
	CE (J)	34.38	11.22	13.3	206.35 %	158.61 %
npc = 128	MSE	0.00004	0.00001	0.00039	300 %	-89.74 %
	CC (ms)	2002.4	699.1	696	186.43 %	187.70 %
	CE (J)	69.73	24.07	25.18	189.74 %	176.93 %

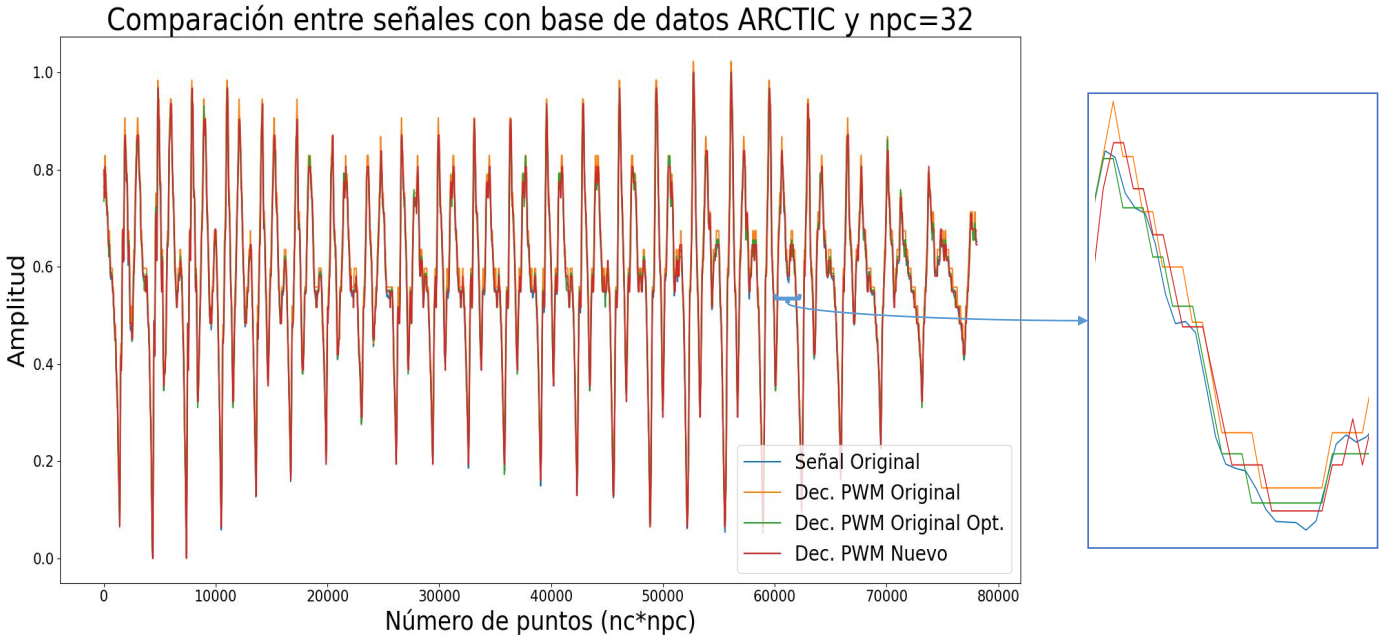

Figura 9: Codificación y decodificación de la señal ARCTIC con las diferentes versiones del algoritmo basado en PWM y con npc=32.

vechando las ventajas que aporta este algoritmo.

Tabla 3: Métricas obtenidas en la codificación y decodificación de imágenes.

npc	**MSE**	**npc**	**MSE**
2	25873.17	32	22.43
4	2476.68	64	5.46
8	450.72	128	1.19
16	88.12	256	0

6. Discusión

Las dos nuevas propuestas del algoritmo de codificación-decodificación basado en PWM presentadas en este trabajo han demostrado ofrecer ventajas muy relevantes frente al algoritmo original tanto en precisión como en eficiencia.

V_{opt} ha demostrado que es capaz de codificar y decodificar series temporales con ventajas muy significativas respecto a V_{or}. Estas mejoras se traducen en una precisión hasta 9 veces mayor y reducciones de costes computacionales y energéticos de más del doble, tal y como se puede ver en la Tabla 1 y Tabla 2. Nótese que la validación en series temporales se ha realizado con señales que presentan diferente dinámica. Mientras que la señal senoidal presenta una dinámica comparativamente lenta ($f_o = 100Hz$), la serie temporal ARCTIC presenta una dinámica significativamente más rápida ($f_o = 32kHz$). Además, en (Lucas and Portillo, 2024) se realiza un análisis de la entropía de Shannon para comparar la aleatoriedad de 5 bases de datos diferentes. Este estudio muestra que la señal senoidal presenta una entropía de Shannon relativamente muy inferior a la que se obtiene con la serie temporal ARCTIC. Por tanto, que V_{opt} haya sido capaz de codificar y decodificar ambas señales demuestra que el algoritmo no se ve influenciado por las características de las series temporales, permitiendo obtener una codificación y decodificación de cualquier serie temporal de forma satisfactoria.

En cuanto a V_{ext} ha demostrado que: (1) es capaz de codificar y decodificar satisfactoriamente imágenes, validando que es posible aplicar el algoritmo a otros campos de aplicación donde no existan relaciones cronológicas; (2) presenta ventajas

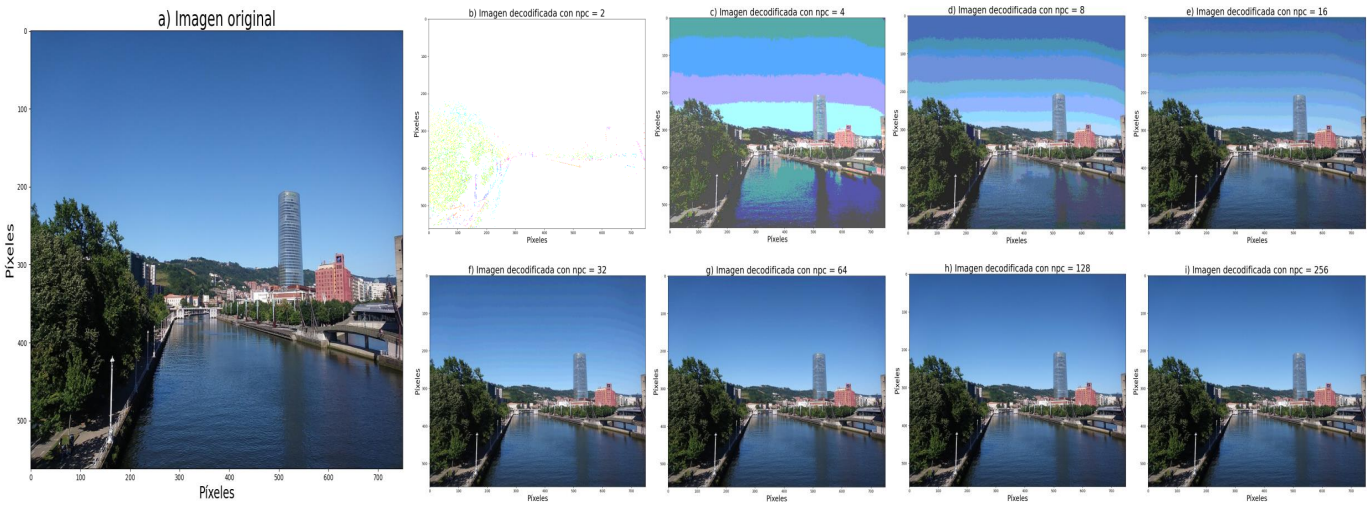

Figura 10: Diferencias en la codificación y decodificación de imágenes variando el parámetro npc.

muy significativas en cuanto a eficiencia a la hora de codificar y decodificar series temporales. En este caso, las ventajas son reducciones de costes computacionales y energéticos de más del doble respecto a V_{or} (ver Tabla 1 y Tabla 2). Asimismo, se debe señalar que V_{ext} no consigue métricas de error tan bajas como V_{or} al codificar y decodificar series temporales.

En caso de comparar V_{ext} con V_{opt}, en líneas generales se puede concluir que en lo que se refiere a eficiencia ambas propuestas presentan costes computacionales y energéticos similares. Sin embargo, en cuanto a precisión V_{opt} presenta mejores métricas en la codificación y decodificación de series temporales.

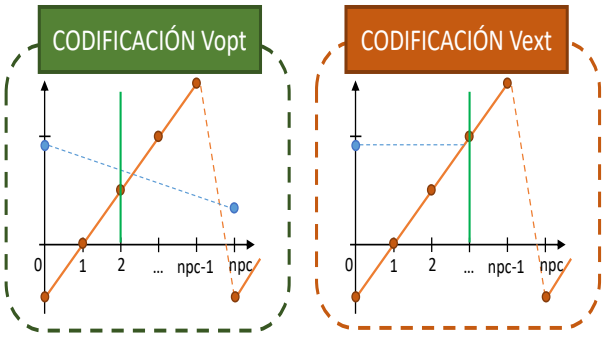

Figura 11: Diferencias codificación con V_{opt} y V_{ext}.

Si se analiza con más profundidad el proceso de codificación y decodificación con V_{ext} se puede apreciar que se introduce un retraso en la señal, tal y como se puede ver en las imágenes ampliadas de las Figuras 8 y 9. Este retraso se debe a la forma de localizar el *spike* durante el proceso de codificación. En la Figura 11 se muestra cómo al codificar el mismo punto, V_{ext} retrasa temporalmente la emisión del *spike* un valor de npc en comparación con V_{opt}. Básicamente, esta circunstancia se debe a que al emplear dos puntos consecutivos en V_{opt} se incluye información sobre el gradiente de la señal a codificar, haciendo que la codificación sea más precisa. Sin embargo, en V_{ext} no es posible tener esta información al utilizar un único punto. Este hecho en campos de aplicación donde no existan relaciones cronológicas no influye en los resultados, sin embargo, en el caso de series temporales puede dar lugar a que los valores de precisión sean inferiores con V_{ext}.

Así, se puede concluir que con el diseño de las dos nuevas propuestas del algoritmo de codificación-decodificación basado en PWM se ha conseguido: a) mejorar la precisión en la codificación y decodificación de series temporales; b) reducir el coste computacional y energético del método al simplificar las operaciones en las que está basado el propio algoritmo; c) tener un algoritmo de codificación susceptible de ser implementado más eficientemente en tiempo real; y d) ampliar el uso del algoritmo a otros campos como el procesamiento de imágenes. En este sentido, el o la usuaria final puede decidir qué propuesta utilizar dependiendo de las características del campo de aplicación objeto de estudio.

7. Conclusiones

En este trabajo se presentan dos nuevas propuestas del algoritmo de codificación-decodificación basado en PWM presentado en Arriandiaga et al. (2020). Los objetivos principales que se persiguen con estas nuevas propuestas son: (a) mejorar las prestaciones del algoritmo original tanto en eficiencia como en precisión de los resultados, y (b) diseñar una nueva extensión del algoritmo que se pueda implementar en campos de aplicación donde no existan relaciones cronológicas entre los valores de la señal utilizada.

Para la validación de las nuevas versiones se han empleado dos series temporales de diferente dinámica, lo cual ha permitido comparar la precisión de los procesos de codificación y decodificación con la versión original del algoritmo y validar que ambas propuestas se adaptan a las características de las series temporales. Además, se ha verificado que la nueva extensión del algoritmo es capaz de codificar y decodificar imágenes, campo de gran interés para las SNNs y donde no existen relaciones cronológicas entre sus datos.

Los resultados reflejan que ambas propuestas consiguen resultados muy satisfactorios. Por una parte, se ha conseguido obtener una versión optimizada del algoritmo original de codifica-

ción-decodificación basado en PWM que codifica y decodifica series temporales con una precisión hasta 9 veces mayor que el algoritmo original y, además, presenta reducciones de costes computacional y energéticos de más del doble. Por otra parte, se ha conseguido diseñar una nueva extensión capaz de codificar y decodificar satisfactoriamente imágenes y, por tanto, aplicable a otros campos de aplicación. Además, esta nueva extensión al codificar y decodificar series temporales presenta reducciones de costes computacionales y energéticos del doble respecto a la versión original del algoritmo.

Asimismo, se debe señalar que al codificar y decodificar series temporales la nueva extensión presenta menor precisión que tanto el algoritmo original como su nueva propuesta optimizada. Sin embargo, la posibilidad de que la nueva extensión sea aplicable a campos donde no existen relaciones cronológicas así como su mayor simplicidad para ser implementada en tiempo real suponen ventajas reseñables para favorecer su uso. Por tanto, será el o la usuaria final quien decida qué versión escoger dependiendo de las características del campo de aplicación donde quiera implantarlo.

Agradecimientos

Este trabajo ha sido financiado por el Departamento de Educación del Gobierno Vasco (proyectos ref. PIBA_2020_1_0008 y ref. IT1726-22), el FEDER/Ministerio de Ciencia e Innovación - Agencia Estatal de Investigación/Proyecto (ref. PID2020-112667RB-I00) y por el Programa Euskampu de la Fundación Misiones Euskampus (proyecto NEUROTIP).

Referencias

Arriandiaga, A., Portillo, E., Espinosa-Ramos, J. I., Kasabov, N. K., 2020. Pulsewidth Modulation-Based Algorithm for Spike Phase Encoding and Decoding of Time-Dependent Analog Data. IEEE Transactions on Neural Networks and Learning Systems 31 (10), 3920–3931.
DOI: 10.1109/TNNLS.2019.2947380

Black, A. W., 2003. Cmu_arctic speech synthesis databases. http://festvox.org/cmu_arctic/, accessed: 2023-02-07.

Brusca, S., Capizzi, G., Lo Sciuto, G., Susi, G., 2019. A new design methodology to predict wind farm energy production by means of a spiking neural network–based system. International Journal of Numerical Modelling: Electronic Networks, Devices and Fields 32 (4).
DOI: 10.1002/JNM.2267

Bu, T., Ding, J., Yu, Z., Huang, T., 2022. Optimized Potential Initialization for Low-Latency Spiking Neural Networks. Proceedings of the AAAI Conference on Artificial Intelligence 36 (1), 11–20.
DOI: 10.1609/AAAI.V36I1.19874

Chen, T., Sun, G., Wei, Z., Li, H., Cheung, K. W., Sun, Y., 2016. Photovoltaic system power generation forecasting based on spiking neural network. Proceedings of the 2015 Chinese Intelligent Systems Conference 359, 573–581.
DOI: 10.1007/978-3-662-48386-2_59/TABLES/3

de Vries, A., 2023. The growing energy footprint of artificial intelligence. Joule 7 (10), 2191–2194.
DOI: 10.1016/J.JOULE.2023.09.004

Deng, L., Wu, Y., Hu, X., Liang, L., Ding, Y., Li, G., Zhao, G., Li, P., Xie, Y., 2020. Rethinking the performance comparison between SNNS and ANNS. Neural Networks 121, 294–307.
DOI: 10.1016/J.NEUNET.2019.09.005

Fang, W., Yu, Z., Chen, Y., Masquelier, T., Huang, T., Tian, Y., 2021. Incorporating Learnable Membrane Time Constant to Enhance Learning of Spiking Neural Networks. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 2661–2671.

García-Martín, E., Rodrigues, C. F., Riley, G., Grahn, H., 2019. Estimation of energy consumption in machine learning. Journal of Parallel and Distributed Computing 134, 75–88.
DOI: 10.1016/J.JPDC.2019.07.007

Gerstner, W., Kistler, W. M., 2002. Spiking Neuron Models: Single Neurons, Populations, Plasticity. Cambridge University Press.
DOI: 10.1017/CBO9780511815706

Gerstner, W., Kistler, W. M., Naud, R., Paninski, L., jan 2014. Neuronal dynamics: From single neurons to networks and models of cognition. Cambridge University Press.
DOI: 10.1017/CBO9781107447615

Han, C. S., Lee, K. M., 2021. A Survey on Spiking Neural Networks. International Journal of Fuzzy Logic and Intelligent Systems 21 (4), 317–337.
DOI: 10.5391/IJFIS.2021.21.4.317

Hong, C., Wei, X., Wang, J., Deng, B., Yu, H., Che, Y., 2020. Training Spiking Neural Networks for Cognitive Tasks: A Versatile Framework Compatible with Various Temporal Codes. IEEE Transactions on Neural Networks and Learning Systems 31 (4), 1285–1296.
DOI: 10.1109/TNNLS.2019.2919662

Kominek, J., Black, A. W., 2003. CMU ARCTIC databases for speech synthesis.

Laña, I., Capecci, E., Del Ser, J., Lobo, J. L., Kasabov, N., 2018. Road traffic forecasting using neucube and dynamic evolving spiking neural networks. International Symposium on Intelligent and Distributed Computing 798, 192–203.
DOI: 10.1007/978-3-319-99626-4_17

Lopes-dos Santos, V., Panzeri, S., Kayser, C., Diamond, M. E., Quian Quiroga, R., 2015. Extracting information in spike time patterns with wavelets and information theory. Journal of Neurophysiology 113 (3), 1015–1033.
DOI: 10.1152/JN.00380.2014

Lucas, S., Portillo, E., may 2024. Methodology based on spiking neural networks for univariate time-series forecasting. Neural Networks 173, 106171.
DOI: 10.1016/J.NEUNET.2024.106171

Mesanza, A. B., Lucas, S., Zubizarreta, A., Cabanes, I., Portillo, E., Rodriguez-Larrad, A., 2020. A Machine Learning Approach to Perform Physical Activity Classification Using a Sensorized Crutch Tip. IEEE Access 8, 210023–210034.
DOI: 10.1109/ACCESS.2020.3039885

Nakai, T., Nishimoto, S., 2023. Artificial neural network modelling of the neural population code underlying mathematical operations. NeuroImage 270, 119980.
DOI: 10.1016/J.NEUROIMAGE.2023.119980

Sboev, A., Litvinova, T., Vlasov, D., Serenko, A., Moloshnikov, I., 2016. On the Applicability of Spiking Neural Network Models to Solve the Task of Recognizing Gender Hidden in Texts. Procedia Computer Science 101, 187–196.
DOI: 10.1016/J.PROCS.2016.11.023

Suetake, K., ichi Ikegawa, S., Saiin, R., Sawada, Y., 2023. S3NN: Time step reduction of spiking surrogate gradients for training energy efficient single-step spiking neural networks. Neural Networks 159, 208–219.
DOI: 10.1016/J.NEUNET.2022.12.008

Vermander, P., Mancisidor, A., Cabanes, I., Perez, N., Torres-Unda, J., 2023. Intelligent Sitting Posture Classifier for Wheelchair Users. IEEE Transactions on Neural Systems and Rehabilitation Engineering 31, 944–953.
DOI: 10.1109/TNSRE.2023.3236692

Xu, C., Zhang, W., Liu, Y., Li, P., 2020. Boosting Throughput and Efficiency of Hardware Spiking Neural Accelerators Using Time Compression Supporting Multiple Spike Codes. Frontiers in Neuroscience 14, 498784.
DOI: 10.3389/FNINS.2020.00104/BIBTEX

Xu, Y., Tang, H., Xing, J., Li, H., 2017. Spike trains encoding and threshold rescaling method for deep spiking neural networks. 2017 IEEE Symposium Series on Computational Intelligence (SSCI), 1–6.
DOI: 10.1109/SSCI.2017.8285427

Yamazaki, K., Vo-Ho, V. K., Bulsara, D., Le, N., 2022. Spiking Neural Networks and Their Applications: A Review. Brain Sciences 12 (7), 863.
DOI: 10.3390/BRAINSCI12070863

Yao, M., Gao, H., Zhao, G., Wang, D., Lin, Y., Yang, Z., Li, G., 2021. Temporal-wise Attention Spiking Neural Networks for Event Streams Classification. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 10221–10230.