Aprendizaje frugal en series temporales mediante técnicas de aprendizaje con pocas muestras

Elena Álvarez[0009−0001−4918−1645], Sebastián Ventura[0000−0003−4216−6378]

Departamento de Ciencia de la Computación e Inteligencia Artificial,
Instituto Andaluz Interuniversitario en Ciencia de Datos e Inteligencia Computacional (DaSCI),
Universidad de Córdoba, 14071 Córdoba, España
{e.alvarez,sventura}@uco.es

Resumen En entornos con restricciones de datos, como la medicina personalizada o las ciudades inteligentes, la inteligencia artificial debe optimizar el aprendizaje con información escasa. Esto es clave para desarrollar sistemas eficientes y sostenibles. Sin embargo, muchos estudios se centran en el análisis de imágenes y, cuando se emplean series temporales, estas suelen convertirse en imágenes. Esto puede dar lugar a una pérdida de información y a la introducción de sesgos, ya que pueden alterarse las dependencias temporales de los datos. Por tanto, este artículo evalúa si transformar las series temporales en imágenes proporciona un mejor rendimiento que trabajar directamente con ellas en problemas de aprendizaje frugal, considerando una aplicación innovadora del aprendizaje con pocas muestras (*few-shot learning*) como estrategia de aprendizaje frugal para el análisis de series temporales. La adaptación propuesta ajusta cómo se procesan los datos y reconfigura los filtros para capturar patrones temporales. El estudio experimental, realizado sobre 25 conjuntos de datos y 4 adaptaciones diferentes de *few-shot learning* (*Relation Network*, *Memory-Augmented Neural Network*, *Meta Long Short-Term Memory* y *Prototypical Network*), concluye que los modelos de aprendizaje frugal que trabajan directamente con series temporales consumen menos recursos que aquellos que consideran imágenes como entrada de datos.

Keywords: Aprendizaje frugal · Aprendizaje con pocas muestras (*few-shot learning*) · conjuntos UCR · Meta-learning · recursos limitados.

1. Introducción

La inteligencia artificial transforma múltiples ámbitos al automatizar tareas, mejorar la toma de decisiones y permitir sistemas más eficientes gracias al uso de datos y algoritmos avanzados. Sin embargo, el desarrollo de modelos cada vez más complejos y de mayor tamaño ha incrementado significativamente el consumo de recursos computacionales y energéticos, generando preocupaciones relacionadas con la sostenibilidad económica y ambiental de estas tecnologías.

En este contexto, el aprendizaje frugal[1] surge y se entiende como la capacidad de desarrollar modelos capaces de generalizar utilizando un número reducido de muestras de entrenamiento y manteniendo un bajo consumo de recursos computacionales. Desde una perspectiva operativa, la frugalidad se evalúa considerando conjuntamente el rendimiento predictivo obtenido en escenarios few-shot y la eficiencia computacional medida mediante tiempo de ejecución, consumo de memoria RAM y utilización de CPU. Es especialmente útil en entornos con limitaciones, como dispositivos con hardware reducido, aplicaciones médicas o agrícolas con pocos datos disponibles, y proyectos de gran escala con infraestructuras limitadas. En estos casos, técnicas como el aprendizaje con pocas muestras (*few-shot learning*)[2] permiten entrenar modelos eficaces a partir de muy pocos ejemplos, ampliando el acceso a soluciones de inteligencia artificial en contextos con restricciones. Sin embargo, muchos estudios en *few-shot learning* se centran en datos de imágenes, ya que facilitan la extracción de características. Cuando se trabaja con series temporales, a menudo se transforman en imágenes[3], pero esto puede provocar pérdida de información importante sobre la dinámica temporal y las dependencias a largo plazo.

En este contexto, se plantea la necesidad de comparar enfoques que convierten series temporales en imágenes frente a aquellos que trabajan directamente con los datos temporales. El artículo compara el rendimiento del *few-shot learning* utilizando 25 conjuntos de datos diferentes de series temporales frente a imágenes generadas a partir de esas mismas series temporales, proporcionando una visión clara de las ventajas y desventajas de trabajar con ambos tipos de datos. Los resultados experimentales, considerando cuatro adaptaciones diferentes de *few-shot learning* (Red Relacional (*Relational Network*, RN), Red Neuronal con Memoria Aumentada (*Memory-Augmented Neural Network*, MANN), Red Prototípica (*Prototypical Network*, PN) y Meta Memoria a Largo y Corto Plazo (*Meta Long Short-Term Memory*, Meta LSTM)), concluyen que los modelos que trabajan directamente con series temporales consumen menos recursos que aquellos que utilizan imágenes como entrada.

2. Antecedentes

El aprendizaje frugal es un enfoque del aprendizaje automático que busca reducir la necesidad de grandes volúmenes de datos, tiempo de entrenamiento y recursos computacionales, con el objetivo de construir modelos más eficientes y aplicables en contextos donde estos recursos son limitados. Dentro de este paradigma se encuentra el *few-shot learning*, una técnica que permite a los modelos aprender nuevas clases a partir de muy pocos ejemplos[4], normalmente entre uno y cinco, apoyándose en la generalización del conocimiento previo y en estrategias como el meta-aprendizaje, en el que los modelos aprenden a aprender[5]. Este enfoque ha mostrado buenos resultados en tareas de visión por computador, especialmente en clasificación de imágenes, gracias a su capacidad para extraer patrones relevantes a partir de datos escasos[6]. Sin embargo, su aplicación a series temporales es más compleja, ya que estos datos tienen una

estructura secuencial y dependiente del tiempo que obliga a considerar no solo cada observación, sino también su evolución temporal. Aun así, esta característica puede aportar información adicional útil para mejorar la generalización con pocos datos[7]. En este contexto, *few-shot learning* resulta especialmente útil en ámbitos donde los datos etiquetados son difíciles o costosos de obtener, como la predicción financiera, la monitorización industrial o la detección de anomalías en medicina, permitiendo identificar patrones temporales relevantes a partir de pocos ejemplos y ofreciendo soluciones eficientes en situaciones de escasez de información.

3. Propuesta

El artículo propone adaptar modelos originalmente diseñados para clasificación de imágenes para que puedan trabajar directamente con series temporales, sin necesidad de convertirlas previamente en imágenes u otras representaciones intermedias. Para ello, se modifican algunas arquitecturas de *few-shot learning* de forma que puedan procesar datos temporales manteniendo su estructura y características inherentes. Primero se describen dichas arquitecturas y después se explica la adaptación propuesta para su uso con datos de series temporales.

3.1. Arquitecturas

Se seleccionaron cuatro arquitecturas de *few-shot learning* ampliamente utilizadas: RN, MANN, Meta LSTM y PN. Estas arquitecturas son reconocidas por su eficacia en tareas de clasificación de imágenes y poseen características que resultan útiles para el procesamiento de series temporales, tal como se describe a continuación.

- **PN**[8]: Enfoque basado en métricas que codifica las muestras en un espacio latente para calcular un prototipo (vector promedio) por clase. La clasificación de nuevas instancias se realiza mediante el cálculo de la distancia Euclídea al prototipo más cercano.
- **RN**[9]: Arquitectura compuesta por un codificador de características y un módulo relacional. Este último aprende una función de similitud no lineal entre pares de muestras.
- **Meta LSTM**[10]: Sistema de meta-aprendizaje de dos niveles que usa una red LSTM como meta-aprendiz para actualizar parámetros mediante gradientes episódicos.
- **MANN**[11]: Red neuronal con memoria externa integrada en una estructura recurrente LSTM que permite almacenamiento dinámico de información.

3.2. Adaptación de las arquitecturas

Las arquitecturas empleadas en este trabajo fueron diseñadas originalmente para el procesamiento de imágenes; sin embargo, se adaptaron para operar

directamente sobre series temporales, evitando su conversión previa a imágenes en escala de grises, una estrategia comúnmente utilizada para aprovechar las capacidades de las redes convolucionales. En el enfoque basado en imágenes, cada serie temporal se representa mediante una imagen monocromática de un solo canal, de 640 $\times$ 480 píxeles, en la que se proyectan directamente los puntos de la serie sobre un plano bidimensional, sin recurrir a transformaciones adicionales. Se aplicó el mismo procedimiento de transformación sobre los 25 conjuntos de datos empleados. De este modo, se preservan los principios y la estructura original de los modelos, al tiempo que se optimiza su aplicación a datos temporales. En cuanto a las series temporales, estas se procesan en su formato original y atraviesan diversas etapas de preparación antes de ser utilizadas por los modelos.

En primer lugar, las series temporales se convierten en tensores, lo que permite representarlas de manera estructurada y compatible con las redes neuronales. Posteriormente, se reorganizan sus dimensiones mediante un proceso de *reshaping* (una reestructuración de la forma del tensor sin modificar sus valores) adoptando una estructura de cuatro dimensiones (N, E, T, C), donde (N) representa el tamaño del lote, (E) el número de series por lote, (T) la longitud temporal y (C) el número de características. Esta representación ofrece una mayor capacidad para modelar la información temporal. A continuación, los datos se normalizan utilizando *MinMaxScaler*, lo que permite escalar los valores al rango [0,1]. Esta etapa homogeneiza las diferentes magnitudes presentes en las series temporales y mejora la estabilidad y convergencia durante el entrenamiento.

Para la extracción de características, las arquitecturas originales se modifican aumentando la profundidad de la red, el tamaño de las capas y el número de filtros[1], lo que permite mejorar su capacidad de aprendizaje sobre series temporales. Al incrementar la profundidad, la red puede aprender patrones de mayor complejidad, ya que las primeras capas capturan características simples y las más profundas combinan esa información para detectar estructuras más abstractas. El aumento del tamaño de los filtros permite analizar ventanas temporales más amplias, capturando relaciones entre datos más distantes en la secuencia, mientras que un mayor número de filtros posibilita la detección simultánea de distintos tipos de patrones. En conjunto, estas modificaciones permiten una extracción de características más rica y representativa de la información contenida en los datos. Además, se incorporó la técnica de *Early Stopping*, que detiene automáticamente el entrenamiento cuando el rendimiento deja de mejorar, reduciendo el sobreajuste y optimizando los recursos computacionales.

En conjunto, estas modificaciones permiten una representación más eficiente de las series temporales y una extracción de características más precisa, mejorando la capacidad de generalización y el rendimiento de los modelos en tareas de clasificación.

[1] Los datos, la configuración de los modelos y los resultados están disponibles en la página web: `https://www.uco.es/kdis/frugal-few-shot-learning/`

4. Análisis experimental

En esta sección se evalúa experimentalmente la eficacia de modelos de *few-shot learning* en series temporales, comparando datos transformados en imágenes frente al uso directo de las series temporales. El objetivo es determinar qué enfoque es más eficiente, analizando cuatro arquitecturas en 25 conjuntos de datos mediante distintas métricas de rendimiento y recursos computacionales.

4.1. Configuración experimental

Se diseñó un conjunto de experimentos para comparar el uso de transformaciones de series temporales en imágenes frente al procesamiento directo de las series temporales en escenarios de aprendizaje frugal y *few-shot learning*. Se evaluaron ocho modelos (cuatro basados en imágenes y cuatro en series temporales) utilizando 25 conjuntos de datos del UCR Time Series Classification Archive 2018[12], seleccionados por su diversidad en longitud, número de clases y dominios de aplicación, lo que permite analizar la capacidad de generalización de los modelos en distintos contextos[1].

La evaluación se realizó mediante métricas de rendimiento (*accuracy*, F1-Score y pérdida por entropía cruzada (*loss*)) y métricas de eficiencia computacional (tiempo de ejecución (*runtime*), consumo de memoria RAM y uso de CPU, medidos en segundos, megabytes y porcentaje de uso respectivamente), con el objetivo de analizar tanto la precisión de los modelos como su frugalidad en entornos con recursos limitados[1].

En cuanto a la evaluación de modelos de *few-shot learning*, esta difiere del aprendizaje automático tradicional porque no utiliza particiones fijas ni validación cruzada *k-fold*. En su lugar, los datos se organizan en episodios, que consisten en pequeñas tareas de clasificación con pocos ejemplos por clase. Para cada conjunto de datos se empleó una configuración N-way 2-shot, donde N corresponde al número de clases del conjunto de datos y el conjunto soporte (conjunto de entrenamiento) contiene dos ejemplos por clase, mientras que el conjunto de consulta (conjunto de evaluación) se construye con el resto de muestras disponibles en cada episodio. Aunque los datos se dividen en conjuntos de entrenamiento, validación y prueba (por ejemplo, 60 %, 20 % y 20 %), estas particiones se utilizan únicamente para generar episodios de forma aleatoria, respetando la distribución de clases e independientemente para cada fase. Esta configuración experimental se ejecuta una única vez por modelo, manteniendo la misma semilla aleatoria. En la ejecución se generó una media de 1.000 episodios de entrenamiento, 200 de validación y 100 de prueba, favoreciendo la capacidad de generalización del modelo a partir de un número reducido de ejemplos.

4.2. Resultados experimentales

Los resultados experimentales se analizan en dos categorías: métricas de rendimiento y métricas de recursos computacionales. Para garantizar conclusiones

sólidas, se aplican pruebas estadísticas [13] que permiten comparar de forma rigurosa los distintos algoritmos y modelos. Dado que los resultados son amplios, se ha habilitado una página web para consultarlos junto con los estadísticos de cada métrica[1]. Además, se emplea una nomenclatura específica para distinguir los enfoques utilizados: '_TS' identifica los modelos que trabajan directamente con las series temporales, mientras que '_IM' hace referencia a los modelos que operan sobre imágenes generadas a partir de dichas series temporales.

Por otro lado, cabe destacar que, tras analizar los resultados obtenidos para cada una de las métricas consideradas, se evaluó la idoneidad de emplear pruebas estadísticas paramétricas o no paramétricas. Con este propósito, se aplicaron las pruebas de normalidad de Shapiro–Wilk[14] y de homogeneidad de varianzas de Levene[15]. Los resultados[1] evidenciaron que, en todas las métricas analizadas, se rechazó la hipótesis nula asociada a dichas pruebas. En consecuencia, no se cumplían los supuestos necesarios para la aplicación de pruebas paramétricas, por lo que fue necesario recurrir a métodos estadísticos no paramétricos (Friedman[16], Nemenyi[17] y Wilcoxon[18]). Por este motivo, los resultados presentados a continuación se centran exclusivamente en este último tipo de pruebas.

Análisis de métricas de rendimiento El análisis se centra en el estudio de las métricas *accuracy*, F1-Score y *loss*. Cabe destacar que debido a restricciones de espacio en dicho artículo, los resultados se encuentran en una página web[1].

Comenzando con la métrica *accuracy*[1], RN_TS y RN_IM son los modelos más precisos, con RN_TS liderando en la mayoría de los conjuntos de datos. MANN_TS y PN_TS también muestran buen rendimiento, mientras que los modelos Meta LSTM, especialmente MetaLSTM_IM, obtienen los peores resultados. Además, las versiones basadas en series temporales superan sistemáticamente a las basadas en imágenes, evidenciando una mejor adaptación a datos temporales.

Debido a que no se cumplen los supuestos de normalidad y homocedasticidad, se emplearon pruebas no paramétricas. La prueba de Friedman confirmó diferencias significativas entre los modelos, con un estadístico de 130,0933 y un valor p de $7{,}4171 \times 10^{-11}$, lo que evidencia diferencias claras en el rendimiento.

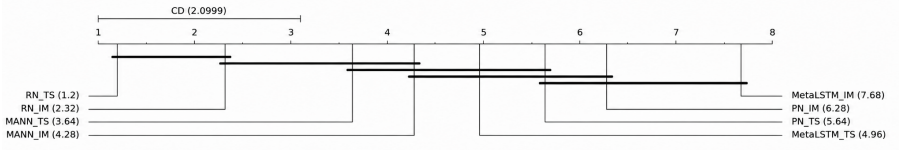


Figura 1: Test de Nemenyi para métrica *accuracy*.

La prueba de Nemenyi mostró que los modelos basados en series temporales superan a los basados en imágenes (véase la Figura 1). RN_TS obtuvo el mejor rendimiento, seguido de RN_IM; MANN y PN mostraron resultados intermedios, mientras que los modelos Meta LSTM presentaron el peor desempeño.

Tabla 1: Resultados de Wilcoxon en *accuracy*: series vs. imágenes

Modelo	R-	R+	p-value	Resultados
PN	7	18	$1{,}00 \times 10^{-3}$	Rechaza H0 ($\alpha = 0,05$) (Distribuciones diferentes)
RN	2	23	$5{,}00 \times 10^{-4}$	Rechaza H0 ($\alpha = 0,05$) (Distribuciones diferentes)
Meta LSTM	0	25	$1{,}22 \times 10^{-5}$	Rechaza H0 ($\alpha = 0,05$) (Distribuciones diferentes)
MANN	5	18	$1{,}26 \times 10^{-2}$	Rechaza H0 ($\alpha = 0,05$) (Distribuciones diferentes)

Nota: Se excluyeron del análisis los valores que no presentaban diferencias.

La prueba de Wilcoxon (Tabla 1) mostró que todos los valores p fueron inferiores a 0,05, evidenciando diferencias estadísticamente significativas y favoreciendo, en general, a los modelos basados en series temporales.

En la métrica F1-Score, los resultados[1] muestran que los modelos RN destacan por su consistencia y alto rendimiento. MetaLSTM_TS también obtiene buenos resultados, aunque con mayor variabilidad, mientras que MANN y PN muestran un desempeño competitivo, siendo PN_TS menos uniforme.

Por otro lado, la prueba de Friedman confirma diferencias significativas entre los modelos (estadístico = 133.8800, $p = 5{,}8934 \times 10^{-11}$), rechazando la hipótesis nula de igualdad de rendimiento. La prueba de Nemenyi (Figura 2) muestra resultados similares a los obtenidos con *accuracy* y destaca que MetaLSTM_TS supera significativamente a MetaLSTM_IM, ya que la distancia crítica (CD) excede el umbral establecido. Finalmente, la prueba de Wilcoxon (véase Tabla 2) muestra que existen diferencias significativas entre el uso de series temporales e imágenes para todos los modelos, ya que se rechaza la hipótesis nula en todos los casos.

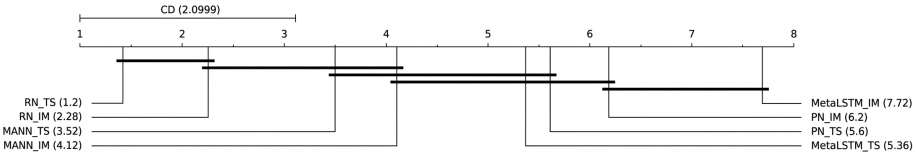


Figura 2: Test de Nemenyi para métrica F1-Score.

Tabla 2: Resultados de Wilcoxon en F1-Score: series vs. imágenes

Modelo	R-	R+	p-value	Resultados
PN	7	18	$7{,}00 \times 10^{-4}$	Rechaza H0 ($\alpha = 0,05$) (Distribuciones diferentes)
RN	2	23	$5{,}00 \times 10^{-4}$	Rechaza H0 ($\alpha = 0,05$) (Distribuciones diferentes)
Meta LSTM	2	23	$1{,}38 \times 10^{-5}$	Rechaza H0 ($\alpha = 0,05$) (Distribuciones diferentes)
MANN	6	19	$6{,}30 \times 10^{-3}$	Rechaza H0 ($\alpha = 0,05$) (Distribuciones diferentes)

Nota: Se excluyeron del análisis los valores que no presentaban diferencias.

Por último, los resultados[1] de la métrica *loss* muestran que RN obtiene, en general, los valores más bajos, lo que indica un mejor rendimiento global. Meta

LSTM y MANN presentan un desempeño más variable según el conjunto de datos, mientras que PN suele registrar pérdidas más elevadas, sugiriendo un ajuste menos eficaz que el de los demás modelos. Los resultados de la prueba de Friedman muestran el rechazo de la hipótesis nula, con un estadístico de prueba de 147.5466 y un valor p de $7{,}9133 \times 10^{-11}$. La prueba de Nemenyi pone de manifiesto la disparidad entre los modelos Meta LSTM aplicados a series temporales y a imágenes (véase Figura 3). Finalmente, la prueba de Wilcoxon (véase Tabla 3) muestra diferencias significativas en la métrica *loss* entre series temporales e imágenes para todos los modelos, ya que la hipótesis nula se rechaza en todos los casos.

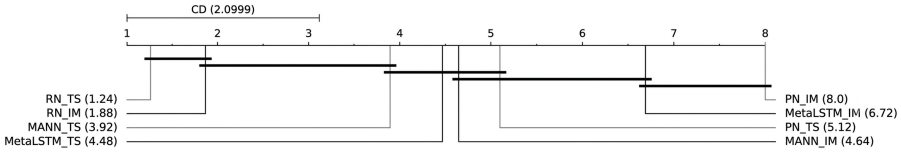


Figura 3: Test de Nemenyi para *loss*.

Tabla 3: Resultados de Wilcoxon en *loss*: series vs. imágenes

Modelo	**R-**	**R+**	**p-value**	**Resultados**
PN	25	0	$1{,}22 \times 10^{-5}$	Rechaza H0 ($\alpha = 0,05$) (Distribuciones diferentes)
RN	19	6	$3{,}20 \times 10^{-3}$	Rechaza H0 ($\alpha = 0,05$) (Distribuciones diferentes)
Meta LSTM	25	0	$1{,}22 \times 10^{-5}$	Rechaza H0 ($\alpha = 0,05$) (Distribuciones diferentes)
MANN	20	5	$1{,}30 \times 10^{-3}$	Rechaza H0 ($\alpha = 0,05$) (Distribuciones diferentes)

Nota: Se excluyeron del análisis los valores que no presentaban diferencias.

En resumen, RN_TS es el modelo con mejor *accuracy* y rendimiento global, superando tanto a los modelos basados en imágenes como al resto de modelos de series temporales. RN_TS y MANN_TS muestran un desempeño sólido y consistente, mientras que los modelos Meta LSTM presentan los peores resultados. Además, las pruebas no paramétricas confirman la superioridad de las series temporales frente a las imágenes en este contexto, destacando a RN_TS como el mejor modelo analizado.

Análisis de los recursos computacionales Este análisis abarca el estudio de las métricas de *runtime*, uso de memoria RAM y consumo de CPU. La métrica *runtime*[1], muestra que MANN_IM y PN_IM presentan los mayores tiempos de ejecución, indicando una mayor complejidad computacional. Los modelos Meta LSTM exhiben tiempos intermedios, mientras que RN_TS destaca como el más eficiente al registrar los menores *runtimes* en la mayoría de los datos.

Al aplicar pruebas no paramétricas para el *runtime*, Friedman muestra un estadístico de 167.0266 y un valor p de $9{,}2967 \times 10^{-11}$, lo que indica la existencia de diferencias significativas entre los modelos. La prueba de Nemenyi (Figura 4) muestra que los modelos de series temporales superan a los de imágenes, con

RN_TS en primer lugar y diferencias mínimas entre MetaLSTM_TS y MetaLSTM_IM. Finalmente, la prueba de Wilcoxon (Tabla 4) indica diferencias significativas en los *runtimes* de PN, RN y MANN, con valores p muy bajos que evidencian cambios relevantes entre versiones. En cambio, el Meta LSTM presenta un valor p de 0.882, lo que sugiere que no existen diferencias significativas en su *runtime*.

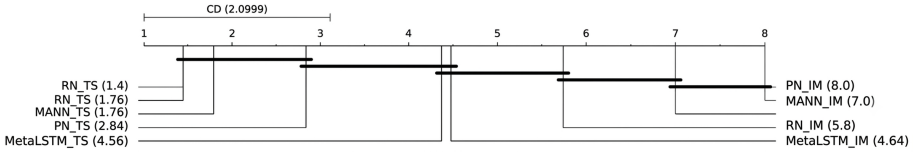


Figura 4: Test de Nemenyi para el *runtime*.

Tabla 4: Resultados de Wilcoxon en *runtime*: series vs. imágenes

Modelo	**R-**	**R+**	**p-value**	**Resultados**
PN	25	0	$1{,}22 \times 10^{-5}$	Rechaza H0 ($\alpha = 0{,}05$) (Distribuciones diferentes)
RN	25	0	$1{,}22 \times 10^{-5}$	Rechaza H0 ($\alpha = 0{,}05$) (Distribuciones diferentes)
Meta LSTM	13	12	$8{,}82 \times 10^{-1}$	No rechaza H0 ($\alpha = 0{,}05$) (Distribuciones similares)
MANN	25	0	$1{,}22 \times 10^{-5}$	Rechaza H0 ($\alpha = 0{,}05$) (Distribuciones diferentes)

Nota: Se excluyeron del análisis los valores que no presentaban diferencias.

En cuanto al uso de memoria RAM[1], MANN presenta el menor consumo, destacando MANN_TS como el más eficiente. El resto de modelos se sitúa alrededor de los 3500 MB, por lo que MANN es el más eficiente en gestión de memoria.

La prueba de Friedman para el uso de RAM muestra diferencias significativas entre modelos (estadístico 167.1600, $p = 9{,}8007 \times 10^{-11}$), por lo que se rechaza la hipótesis nula. La prueba de Nemenyi (ver Figura 5) confirma que el modelo MANN presenta el mejor rendimiento, especialmente MANN_TS, con diferencias significativas entre MANN_TS y RN_TS. Además, los modelos de series temporales continúan superando a los modelos de imágenes. Finalmente, la prueba de Wilcoxon (ver Tabla 5) muestra diferencias significativas en el uso de RAM para PN y RN ($p = 3{,}95 \times 10^{-2}$ y $3{,}70 \times 10^{-2}$), mientras que Meta LSTM y MANN no presentan diferencias significativas ($p = 2{,}87 \times 10^{-1}$ y $2{,}41 \times 10^{-1}$).

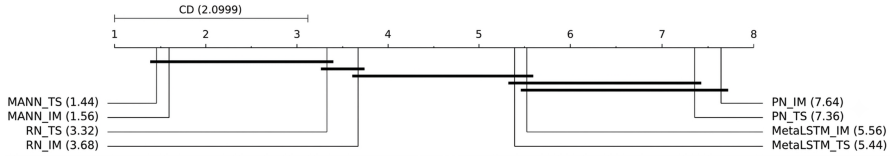


Figura 5: Test de Nemenyi para RAM.

Tabla 5: Resultados de Wilcoxon en RAM: series vs. imágenes

Modelo	R-	R+	p-value	Resultados
PN	16	9	$3{,}95 \times 10^{-2}$	Rechaza H0 ($\alpha = 0,05$) (Distribuciones diferentes)
RN	17	8	$3{,}70 \times 10^{-2}$	Rechaza H0 ($\alpha = 0,05$) (Distribuciones diferentes)
Meta LSTM	14	11	$2{,}87 \times 10^{-1}$	No rechaza H0 ($\alpha = 0,05$) (Distribuciones similares)
MANN	14	11	$2{,}41 \times 10^{-1}$	No rechaza H0 ($\alpha = 0,05$) (Distribuciones similares)

Nota: Se excluyeron del análisis los valores que no presentaban diferencias.

Por último, en consumo de CPU[1], MANN es el modelo más eficiente, destacando MANN_TS como el mejor. Meta LSTM es el más costoso, especialmente en imágenes, mientras que PN y RN muestran un rendimiento intermedio, siendo RN el más consistente. En general, MANN es la mejor opción para minimizar el uso de CPU. La prueba de Friedman muestra diferencias significativas entre las distintas configuraciones evaluadas en el consumo de CPU (estadístico 170.7700, $p = 1{,}2597 \times 10^{-11}$). La prueba de Nemenyi (ver Figura 6) confirma a MANN como modelo más eficiente tanto en imágenes como series temporales, con diferencias significativas respecto a RN. Finalmente, la prueba de Wilcoxon (ver Tabla 6) indica que todos los modelos (PN, RN, Meta LSTM y MANN) presentan diferencias significativas (p <0.05) entre series temporales e imágenes, rechazándose la hipótesis de igualdad.

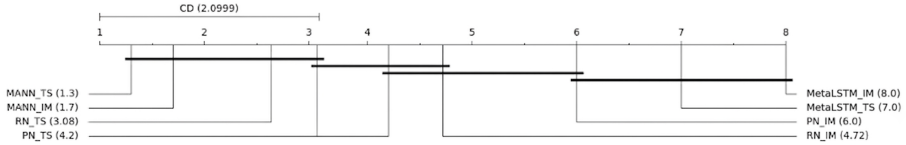


Figura 6: Test de Nemenyi para CPU.

Tabla 6: Resultados de Wilcoxon en CPU: series vs. imágenes

Modelo	R-	R+	p-value	Resultados
PN	25	0	$1{,}18 \times 10^{-5}$	Rechaza H0 ($\alpha = 0,05$) (Distribuciones diferentes)
RN	24	1	$1{,}37 \times 10^{-5}$	Rechaza H0 ($\alpha = 0,05$) (Distribuciones diferentes)
Meta LSTM	25	0	$1{,}22 \times 10^{-5}$	Rechaza H0 ($\alpha = 0,05$) (Distribuciones diferentes)
MANN	17	7	$3{,}84 \times 10^{-2}$	Rechaza H0 ($\alpha = 0,05$) (Distribuciones diferentes)

Nota: Se excluyeron del análisis los valores que no presentaban diferencias.

El análisis muestra que MANN es el modelo más eficiente computacionalmente. MANN_IM y PN_IM presentan los mayores *runtimes*, mientras que RN_TS es el más rápido. En RAM, MANN_TS destaca por el menor consumo, y en CPU, MANN obtiene el mejor rendimiento global. Los análisis estadísticos confirman diferencias significativas entre modelos, consolidando a MANN (especialmente MANN_TS) como la opción más eficiente.

Análisis de ambos grupos de métricas Se ha observado que los modelos con mejor desempeño son RN_TS en las métricas de rendimiento y MANN_TS en las métricas de recursos computacionales. Para determinar cuál presenta el mejor comportamiento global, se compararon sus resultados utilizando los valores obtenidos de las tablas de Nemenyi y CD (véase la Tabla 7). Los resultados muestran diferencias estadísticamente significativas a favor de RN_TS en *accuracy*, F1-Score y *loss*, mientras que las diferencias en *runtime*, uso de RAM y CPU no son significativas. En consecuencia, dentro del enfoque de aprendizaje frugal, RN_TS se posiciona como la mejor alternativa global, al ofrecer un rendimiento predictivo superior sin un incremento significativo en el consumo de recursos computacionales.

Tabla 7: Comparación de alcances medios frente a CD

Métricas	**RN_TS**	**MANN_TS**	**Diferencia rangos medios**
Accuracy	1.2	3.64	2.44
F1-Score	1.2	3.52	2.32
Loss	1.24	3.92	2.68
Runtime	1.4	1.76	0.36
RAM	3.32	1.44	1.88
CPU	3.08	1.3	1.78

5. Conclusión

Este trabajo propone metodologías de *few-shot learning* para el análisis directo de series temporales, evitando su transformación a imágenes con el fin de determinar si preservar la estructura temporal original aporta ventajas en escenarios con pocos datos. Para ello, se adaptan modelos originalmente diseñados para imágenes, modificando su procesamiento y sus filtros convolucionales para capturar patrones temporales. Los experimentos realizados sobre 25 conjuntos de datos demuestran que trabajar directamente con series temporales mejora significativamente el rendimiento y la eficiencia computacional en comparación con los enfoques basados en imágenes.

En cuanto a los resultados, se observa que RN_TS es el modelo con mejor desempeño global, alcanzando los valores más altos de *accuracy* y F1-Score, así como las menores en *loss*, superando tanto a los modelos basados en imágenes como a las arquitecturas Meta LSTM. Por su parte, MANN_TS destaca por ser el modelo más eficiente en el uso de CPU y memoria RAM. Aunque los análisis estadísticos confirman diferencias significativas en rendimiento entre los modelos, las diferencias en consumo de recursos son menos acusadas, por lo que RN_TS se posiciona como la mejor alternativa en términos generales al ofrecer el mejor equilibrio entre precisión y eficacia computacional.

Agradecimientos Esta investigación ha sido financiada parcialmente por el proyecto PID2023-148396NB-I00 del Ministerio de Ciencia e Innovación y el Fondo Europeo de Desarrollo Regional, por el proyecto ProyExcel-0069 de la Consejería de Universidad, Investigación e Innovación de la Junta de Andalucía.

Referencias

1. Evchenko, M., Vanschoren, J., Hoos, H. H., Schoenauer, M., & Sebag, M. (2021). Frugal machine learning. *arXiv preprint arXiv*:2111.03731.
2. ZHAO, K. L., JIN, X. L., & WANG, Y. Z. (2021). Survey on few-shot learning. *Journal of software*, 32(2), 349-369.
3. Park, S. H., Syazwany, N. S., & Lee, S. C. (2023). Meta-feature fusion for few-shot time series classification. *IEEE Access*, 11, 41400-41414.
4. Wang, Y., Yao, Q., Kwok, J. T., & Ni, L. M. (2020). Generalizing from a few examples: A survey on few-shot learning. ACM computing surveys (csur), 53(3), 1-34.
5. Gharoun, H., Momenifar, F., Chen, F., & Gandomi, A. H. (2024). Meta-learning approaches for few-shot learning: A survey of recent advances. *ACM Computing Surveys*, 56(12), 1-41.
6. Ji, Z., Chai, X., Yu, Y., Pang, Y., & Zhang, Z. (2020). Improved prototypical networks for few-shot learning. *Pattern Recognition Letters*, 140, 81-87.
7. Li, C., Denison, T., & Zhu, T. (2024). A survey of few-shot learning for biomedical time series. IEEE Reviews in Biomedical Engineering, 18, 192-210.
8. Información y código sobre Red Prototípica `https://github.com/oscarknagg/few-shot/tree/master`
9. Información y código sobre Red Relacional: `https://github.com/floodsung/LearningToCompare_FSL`
10. Información y código sobre Meta LSTM: `https://github.com/stanFurrer/Meta-Learner-LSTM-for-Few-Shot-Learning`
11. Información y código sobre MANN: `https://github.com/shashankvmaiya/Meta-Learning-MANN`
12. Dau, H. A., Bagnall, A., Kamgar, K., Yeh, C. C. M., Zhu, Y., Gharghabi, S., ... & Keogh, E. (2019). The UCR time series archive. IEEE/CAA Journal of Automatica Sinica, 6(6), 1293-1305.
13. C. Luna, A. R. Moya, J. M. Luna and S. Ventura, "StaTDS library: Statistical tests for data science". *Neurocomputing*, vol. 595, p. 127877, 2024.
14. S. Shapiro and M. B. J. B. Wilk, "An analysis of variance test for normality". *Biometrika*, vol. 52, p. 591–611, 1965.
15. H. Levene, "Robust tests for the equality of variances". *Contributions to Probability and Statistics: Essays in Honor of Harold Hotelling*, p. 278–292, 1960.
16. M. Friedman, "The use of ranks to avoid the assumption of normality implicit in the analysis of variance.", *Journal of the American Statistical Association*, vol. 32, p. 675–701, 1937.
17. M. Nemenyi, "Distribution-free multiple comparisons.", In: *Contributions to Probability and Statistics: Essays in Honor of Harold Hotelling*, Princeton University Press, p. 263–272, 1963.
18. F. Wilcoxon, "Individual comparisons by ranking methods.", *Biometrics Bulletin*, vol. 1, p. 80–83, 1945.