ReliefF escalable y eficiente basado en prototipos

Samuel Suárez-Marcote[0009−0004−0228−2678],
Laura Morán-Fernández[0000−0001−6703−1846] y
Verónica Bolón-Canedo[0000−0002−0524−6427]

Universidade da Coruña, CITIC, A Coruña, España
{s.suarez.marcote, laura.moranf, veronica.bolon}@udc.es

Resumen La selección de características es un paso clave en el preprocesado de datos de alta dimensionalidad. En este trabajo se presenta una versión escalable y eficiente de ReliefF basada en prototipos en un espacio latente de baja dimensionalidad. La propuesta separa la búsqueda de los vecinos cercanos, realizada en el espacio reducido, del cálculo de relevancia de cada característica, que se mantiene en el espacio original, reduciendo el coste computacional sin perder interpretabilidad sobre las variables originales. Se proponen dos formas para el entrenamiento de los prototipos, una estrategia por mínimos cuadrados alternantes y otra basada en una capa lineal. A su vez, se presentan dos variantes para la actualización de la relevancia de cada característica, una variante exacta y otra aproximación diagonal. Los experimentos, realizados sobre cuatro conjuntos de datos con distintas propiedades, muestran que el método propuesto mantiene resultados similares en clasificación frente a ReliefF, con reducciones drásticas del tiempo de ejecución y consumos energéticos hasta tres órdenes de magnitud menores.

Keywords: Selección de características · ReliefF · Prototipos · Eficiencia · Algoritmos verdes · Alta dimensionalidad

1. Introducción

La incorporación de tecnologías de sensorización e informatización ha impulsado grandes avances en diversos sectores durante los últimos años. Gracias a la integración de sistemas de recogida continua de datos, unidos a la aplicación de técnicas de inteligencia artificial, se ha mejorado significativamente la capacidad de monitorización, predicción y toma de decisiones basada en los datos disponibles. Sin embargo, dicho crecimiento viene acompañado de una mayor complejidad en el tratamiento de los mismos. En particular, la proliferación de los dispositivos de Internet de las Cosas (IoT) obliga a replantear dichos procesos, poniendo el foco en la privacidad, tratamiento y almacenamiento de grandes cantidades de información [10].

En este contexto y, ante la gran dimensionalidad de los datos actuales, la selección de características se presenta como un paso de preprocesado necesario que permite descartar aquellas características que sean irrelevantes o redundantes, manteniendo las verdaderamente relevantes con el fin de mejorar la eficiencia

computacional y, en muchos casos, también la generalización de los modelos [4]. En función de su interacción con el modelo de inducción, los métodos de selección de características se agrupan en tres categorías principales: filtros, envolventes y embebidos. En entornos IoT, los métodos de filtro resultan especialmente adecuados debido a su bajo coste computacional, ya que no requieren entrenar un modelo de inducción. No obstante, en datos de alta dimensionalidad el problema no se limita a la redundancia. Muchas características presentan interacciones complejas de alto orden, por ejemplo en dominios como el biomédico donde la expresión de un gen puede modificar la expresión de uno o más genes [2]. En estas situaciones, la familia de algoritmos Relief ha sido popular al capturar efectos de interacciones de alto orden sin necesidad de evaluar exhaustivamente todas las combinaciones de atributos disponibles [7]. En particular, ReliefF constituye una de las variantes más conocidas de esta familia al permitir lidiar con problemas multiclase y con datos ruidosos o incompletos.

Sin embargo, ReliefF presenta limitaciones importantes. Por un lado, su coste computacional crece con el número de muestras, lo que dificulta su aplicación en conjuntos de datos de gran tamaño. Por otro, su rendimiento puede verse afectado cuando aumenta el número de características irrelevantes, especialmente en presencia de interacciones de orden superior a cuatro, donde la detección de patrones relevantes puede degradarse de forma notable [2]. Estas limitaciones se agravan en entornos con restricciones de recursos, donde resulta prioritario diseñar métodos eficientes tanto en tiempo como en memoria.

Ante esta situación y para evitar dichas limitaciones, en este trabajo se propone una versión eficiente y escalable de ReliefF basada en prototipos para permitir, por un lado, la aplicación de dicho método en escenarios con un alto número de muestras y que, recurriendo a la proyección de los mismos en un espacio latente de baja dimensionalidad, permita aprovechar las interacciones de alto orden entre características. Los experimentos muestran que las variantes propuestas alcanzan precisiones similares o superiores a ReliefF con reducciones significativas de tiempo y consumo energético de varios órdenes de magnitud.

2. Estado del arte

El método propuesto en este artículo se sitúa en la intersección entre la selección de características basada en muestras y el aprendizaje basado en prototipos.

Entre los métodos de selección de características de tipo filtro, la familia Relief constituye una de las más influyentes. El algoritmo Relief original, presentado por Kira y Rendell [6], estima la relevancia de cada característica comparando una muestra con instancias de la clase positiva y, a su vez, de la clase negativa. ReliefF amplía esta idea al caso multiclase, incorporando múltiples vecinos más cercanos y un esquema de puntuación más robusto. No obstante, al tratarse de un algoritmo basado en muestras y vecinos más próximos, su complejidad temporal escala linealmente tanto con el número de características como con el número de instancias, lo que lo convierte en un importante cuello de botella computacional para conjuntos de datos de alta dimensionalidad [7].

Para contrarrestar esta limitación, se han propuesto diversas adaptaciones tanto estructurales como algorítmicas. Partiendo de trabajos clásicos, Robnik-Šikonja [14] introduce estructuras de árbol k-dimensional para acelerar la búsqueda de vecinos bajo ciertas condiciones. Más recientemente, Eiras-Franco et al. [1] propusieron ReliefF-LSH, que aproxima el grafo de vecinos mediante *hashing* sensible a la localidad, escalando mediante ejecución distribuida en Spark y sin requerir parámetros adicionales. Asimismo, Palma-Mendoza et al. [11] introducen DiReliefF, una versión distribuida en Spark capaz de procesar un gran volumen de datos obteniendo tiempos de ejecución razonables. Otras adaptaciones se han centrado en integrar ReliefF con técnicas de transformación del espacio para manejar datos de alta dimensionalidad. Por ejemplo, Liu et al. [9] combinaron factorización matricial no negativa con ReliefF, aprendiendo inicialmente representaciones de menor dimensión antes de realizar la selección de características.

Además, las aproximaciones basadas en vecinos más cercanos han sido ampliamente estudiadas en la selección de características basada en *clustering*. Reyes et al. [12] recurren a *k-means* para obtener prototipos de los clústeres, calculando las estadísticas con respecto a estos, reduciendo drásticamente el número de comparaciones por pares. Sin embargo, este enfoque sigue operando en el espacio original de las muestras, de alta dimensionalidad, donde las distancias entre prototipos pueden resultar poco fiables.

En este contexto, el aprendizaje de prototipos ha resurgido recientemente como una alternativa altamente eficiente frente a metodologías dependientes de instancias e incluso con su aplicación a redes neuronales profundas. Al condensar los datos de entrenamiento en un conjunto representativo de puntos en un espacio de menor dimensión, el aprendizaje basado en prototipos constituye una técnica clave para abordar la heterogeneidad estadística en entornos federados sin incrementar de forma significativa los costes de comunicación [13]. En una línea cercana a la de este trabajo, aunque fuera del ámbito de la selección de características, ProtoNN [3] combina proyecciones en baja dimensionalidad y un conjunto de prototipos aprendido para construir un clasificador k-vecinos más cercanos extremadamente compacto, lo que lo hace adecuado para dispositivos con recursos limitados. Más recientemente, FedProto [15] adaptó esta idea al aprendizaje federado, proponiendo una estructura donde los modelos locales comparten únicamente prototipos de clase en lugar de los datos brutos o gradientes de un modelo.

En conjunto, estos trabajos motivan un método de selección de características que utiliza prototipos para resumir la estructura de las clases y, a su vez, recurren a una matriz de proyección para reducir la dimensionalidad en el cálculo de las puntuaciones de las características. Frente a la implementación base de ReliefF, esta estrategia evita comparaciones exhaustivas muestra a muestra con los vecinos y las reemplaza por comparaciones basadas en prototipos en un espacio de baja dimensionalidad, permitiendo el manejo de conjuntos de datos de mayor tamaño y complejidad.

3. Método propuesto

En este trabajo proponemos una implementación eficiente de ReliefF en la que el método aprende una proyección lineal W y un conjunto de prototipos B que representan la estructura de las clases en un espacio latente de baja dimensión. Esta formulación reduce simultáneamente la dimensionalidad de los datos y la cantidad de representantes necesarios para describir cada clase. Sobre este espacio reducido se implementa una adaptación de ReliefF, en la que la relevancia de cada característica se calcula empleando los prototipos de las clases. Para el cálculo de la métrica de relevancia de cada una de las características se proponen dos aproximaciones. Independientemente de la aproximación empleada, el método preserva el comportamiento de ReliefF, pero con una formulación eficiente y adecuada a conjuntos de datos de gran tamaño. La Figura 1 muestra un esquema general de la idea propuesta.

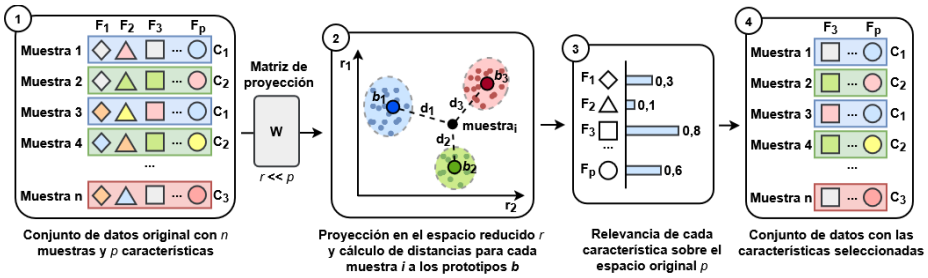


Figura 1. Esquema visual del método propuesto.

3.1. Planteamiento del problema

Sea $\mathbf{X} \in \mathbb{R}^{n \times p}$ la matriz de datos de entrada, con n muestras y p atributos, y sea $\mathbf{y} \in \{0, \ldots, C\}^n$ el vector de etiquetas. Denotamos por $x_i \in \mathbb{R}^p$ la i-ésima muestra y por y_i su clase correspondiente. Para cada una de ellas, definimos su proyección en un espacio latente de dimensión reducida, r como

$$z_i = W x_i, \qquad W \in \mathbb{R}^{r \times p}, \quad r \ll p.$$

El objetivo del aprendizaje es obtener una proyección que preserve la estructura discriminativa de las clases mediante un conjunto de prototipos $\{b_c\}_{c=0}^{C}$ en el espacio de baja dimensionalidad. Para el entrenamiento de W y B, proponemos dos opciones. La primera de ellas consiste en una estrategia de mínimos cuadrados alternantes (ALS) y la segunda en una estrategia probabilística basada en gradiente.

3.2. Aprendizaje de prototipos

ALS. Como primera aproximación y con el objetivo de obtener una convergencia rápida y determinista, formulamos la proyección como un problema de mínimos cuadrados alterno. El objetivo es minimizar la distancia euclidiana entre las muestras proyectadas y el prototipo de su clase:

$$\min_{W,\{b_c\}_{c=0}^{C}} \sum_{i=1}^{n} \|Wx_i - b_{y_i}\|_2^2 + \lambda \|W\|_F^2, \tag{1}$$

donde $\lambda > 0$ es un parámetro de regularización L_2 para evitar sobreajuste.

Resolvemos el problema alternando la actualización de W y de los prototipos. En primer lugar y, tras inicializar los prototipos, para un B fijo, la actualización de W se obtiene mediante la solución cerrada del siguiente problema de regresión:

$$W = B_{[y]}^{\top} X \left(X^{\top} X + \lambda I_p \right)^{-1}, \tag{2}$$

donde $B_{[y]} \in \mathbb{R}^{n \times r}$ es la matriz cuyas filas contienen el prototipo $b_{y_i}^{\top}$ asociado a la etiqueta de cada muestra. Dado un W actualizado, los prototipos se recalculan como la media de las muestras proyectadas pertenecientes a cada clase:

$$b_c = \frac{1}{|I_c|} \sum_{i \in I_c} Wx_i, \qquad I_c = \{\, i : y_i = c \,\}. \tag{3}$$

Capa lineal. De forma alternativa, especialmente pensada para escenarios federados o donde el número de muestras es reducido, proponemos una variante basada en descenso de gradiente. En esta variante, una capa lineal aprende una proyección $z_i = Wx_i$ y cada clase es representada mediante un pequeño conjunto de prototipos entrenables, a diferencia de la alternativa anterior donde un único prototipo representa a cada una de las clases. El uso de varios prototipos por clase permite modelar variaciones internas dentro de una misma categoría sin necesidad de aumentar la complejidad del modelo.

Dado un conjunto de prototipos $b_{c,k} \in \mathbb{R}^r$, con $c = 0, \ldots, C$ y $k = 1, \ldots, K$, siendo K el número de prototipos por clase, calculamos la distancia entre cada representación latente de la muestra y los prototipos deseados

$$dist_{i,(c,k)} = \|z_i - b_{c,k}\|_2^2. \tag{4}$$

El valor de K se presenta como un hiperparámetro, el cual se estudiará en profundidad en la Sección 5. A continuación, se obtiene una ponderación de los prototipos mediante una función *softmax* aplicada a las distancias negativas:

$$a_{i,(c,k)} = \frac{\exp(-\gamma dist_{i,(c,k)})}{\sum_{c',k'} \exp(-\gamma dist_{i,(c',k')})}, \tag{5}$$

donde $\gamma > 0$ controla el grado de concentración de la distribución resultante.

La probabilidad de clase se obtiene agregando las contribuciones de los prototipos de la misma clase y finalmente se entrena el modelo minimizando la entropía cruzada de la siguiente forma

$$\mathcal{L} = -\sum_{c=0}^{C} \sum_{i \in I_c} \log \hat{y}_{i,\, y_i}, \qquad \hat{y}_{i,c} = \sum_{k=1}^{K} a_{i,(c,k)}. \tag{6}$$

3.3. Cálculo de la relevancia

La principal diferencia entre la propuesta presentada y la formulación clásica de ReliefF radica en la separación entre la búsqueda de vecinos y el cálculo de la relevancia. En lugar de realizar ambas operaciones en el espacio original de los datos, el método desacopla estas dos etapas. La identificación de los elementos equivalentes a los aciertos y fallos del método original se lleva a cabo en el espacio latente de baja dimensionalidad, donde cada clase se encuentra representada mediante sus prototipos. Esta estrategia reduce significativamente el coste computacional asociado a las búsquedas de vecinos más cercanos, especialmente en conjuntos de datos con un elevado número de muestras.

Sin embargo, la evaluación de los márgenes discriminativos y la actualización de las puntuaciones de relevancia se realizan en el espacio original de características. De este modo, las puntuaciones finales conservan una interpretación directa sobre las variables de entrada, evitando la pérdida de interpretabilidad que habitualmente acompaña a los métodos de extracción de características. En resumen, la representación latente actúa únicamente como un mecanismo para capturar la estructura local de los datos, mientras que la estimación de relevancia continúa expresándose en términos de los atributos originales.

Una vez aprendidos W y los prototipos B, para cada muestra x_i se identifica el prototipo asociado a su clase, $h(i) = b_{y_i}$, y el prototipo de la clase incorrecta más próximo en el espacio latente, $m(i) = \arg\min_{c \neq y_i} \|z_i - b_c\|_2^2$, pudiendo recurrir a la media de los prototipos de la clase en caso de considerarse necesario.

Proponemos dos métodos para calcular la puntuación final de la relevancia de cada característica con el objetivo de estudiar su balance entre exactitud computacional y eficiencia.

Puntuación exacta. La primera de las modificaciones propuestas busca tener en cuenta las interacciones inducidas por la proyección. Para ello se emplea una descomposición completa de la métrica, definiendo la matriz

$$M = W^\top W \in \mathbb{R}^{p \times p}. \tag{7}$$

Como resultado, la actualización del vector de pesos w, a partir de S instancias muestreadas, reforzando la contribución de los atributos que acercan la muestra al prototipo de su clase más cercana y penalizando aquellos que la alejan de los prototipos de las otras clases:

$$w = \frac{1}{S} \sum_{i \in \mathcal{S}} \left[-\Delta x_h^{(i)} \odot \left(M \Delta x_h^{(i)} \right) + \Delta x_m^{(i)} \odot \left(M \Delta x_m^{(i)} \right) \right], \tag{8}$$

donde $\odot$ denota el producto elemento a elemento. Esta expresión aprovecha directamente la distancia de Mahalanobis en el espacio latente para ponderar la contribución de cada atributo.

Aproximación diagonal. Para evitar el coste en memoria y cómputo $O(p^2)$ asociado a la matriz completa M, es posible aproximar dicha matriz únicamente por su diagonal. Podemos definir dicha diagonal como:

$$diag(M)_j = \sum_{k=1}^{r} W_{kj}^2, \qquad j = 1, \ldots, p. \tag{9}$$

Simplificándose la regla de actualización de los pesos a

$$w = \frac{1}{S} \sum_{i \in \mathcal{S}} \Big[- diag(M) \odot (\Delta x_h^{(i)})^2 + diag(M) \odot (\Delta x_m^{(i)})^2 \Big], \tag{10}$$

donde el cuadrado se aplica elemento a elemento. Esta aproximación escala linealmente con p, por lo que resulta especialmente adecuada para conjuntos de datos de alta dimensionalidad, sin perder la información sobre la magnitud de los pesos aprendidos o bien en dispositivos con recursos limitados donde no sea posible mantener dicha matriz en memoria.

4. Configuración de los experimentos

Los experimentos se realizaron sobre cuatro conjuntos de datos. La Tabla 1 resume sus propiedades principales. Para evaluar la propuesta se compararon las variantes obtenidas al combinar las dos estrategias de aprendizaje, ALS y capa lineal (Lin.), con las dos métricas de relevancia, exacta (Ex.) y aproximación diagonal (Dg.), dando lugar a las variantes *ALS+Dg.*, *ALS+Ex.*, *Lin.+Dg.* y *Lin.+Ex.*, todas ellas comparadas frente a ReliefF original, considerando tanto la similitud entre rankings como el rendimiento de clasificación usando las características mejor puntuadas. Además, se midieron el tiempo de ejecución y el consumo energético mediante la librería CodeCarbon. La similitud entre los *rankings* se midió mediante el coeficiente de correlación de Spearman. El clasificador empleado en el paso de evaluación posterior fue k-vecinos más cercanos con $k = 3$ (kNN). Todos los experimentos se ejecutaron con validación cruzada estratificada de 3 *folds* y 3 repeticiones sobre una GPU NVIDIA GeForce GTX Titan y una CPU Intel Core i5-8400.

Se realizaron tres experimentos principales: (i) escalabilidad frente al número de muestras usando MNIST como conjunto de referencia, fijando como $r = 16$ la dimensión, 4 prototipos por clase y empleando las 50 características más relevantes para el clasificador, como configuración de referencia al ofrecer un compromiso adecuado entre representación y coste computacional; (ii) análisis de sensibilidad al número de prototipos, con dimensión latente fija, $r = 16$; y (iii) análisis de sensibilidad a la dimensión del espacio latente, fijando a 2 el número de prototipos. Finalmente se realiza una comparativa en detalle para todas las métricas sobre los cuatro conjuntos de datos empleando 20 características.

Tabla 1. Características principales de los conjuntos de datos. Se indica con una estrella (*) al trabajar con un subconjunto de muestras.

Conjunto de datos	#Muestras	#Características	#Clases	Repositorio
splice	3175	60	3	[16]
CorrAL100	100 000	100	2	[16]
humanActivity	7352	561	6	[5]
MNIST	70 000*	784	10	[8]

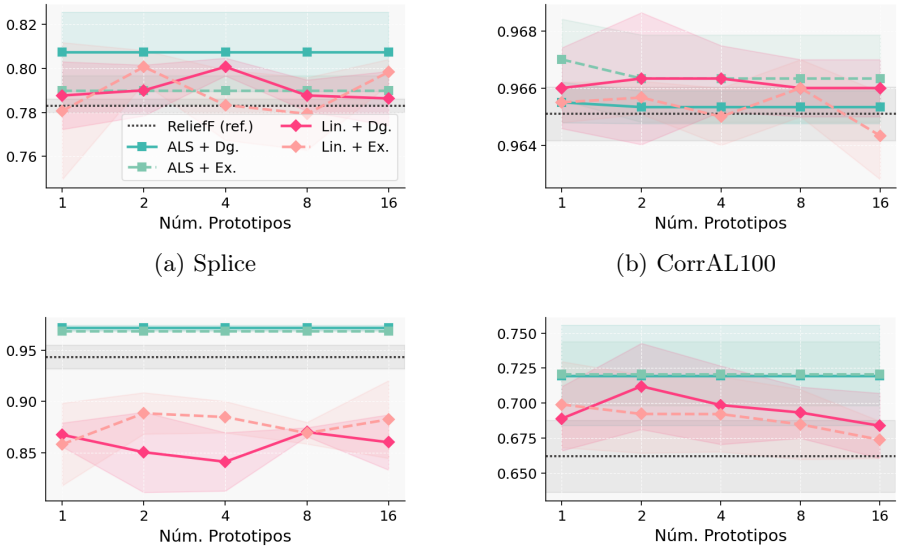
5. Resultados

Para cada uno de los experimentos propuestos anteriormente se presentan y analizan los resultados obtenidos.

Escalabilidad. La Tabla 2 recoge los resultados para distintos subconjuntos de MNIST, manteniendo la misma configuración. En ella podemos observar que existen ciertas diferencias en los *rankings* de características obtenidos, aunque características diferentes pueden contener información similar, por lo que es necesario comprobar los resultados de clasificación. Las cuatro combinaciones propuestas obtienen precisiones de clasificación con las 50 características seleccionadas similares o superiores a las de ReliefF, independientemente del tamaño del conjunto de datos. Las variantes con ALS muestran resultados ligeramente superiores a las de capa lineal. Sin embargo, existe cierto margen de mejora, ya que los hiperparámetros de la capa lineal no fueron optimizados por brevedad.

Tabla 2. Resultados de escalabilidad sobre MNIST. Se muestran la similitud de los *rankings*, exactitud para kNN con 50 características, tiempo de ejecución y consumo.

Muestras	**Método**	**Rankings**	**Exactitud**	**Tiempo (s)**	**Consumo (Wh)**
5000	ReliefF	–	$0.841_{\pm 0.007}$	$547.880_{\pm 0.588}$	$5.640_{\pm 0.006}$
	ALS + Dg.	$0.766_{\pm 0.025}$	$0.851_{\pm 0.011}$	$6.502_{\pm 0.011}$	$0.010_{\pm 0.000}$
	ALS + Ex.	$0.347_{\pm 0.024}$	$0.856_{\pm 0.015}$	$6.504_{\pm 0.013}$	$0.011_{\pm 0.001}$
	Lin. + Dg.	$0.800_{\pm 0.021}$	$0.836_{\pm 0.006}$	$6.951_{\pm 0.066}$	$0.015_{\pm 0.001}$
	Lin. + Ex.	$0.755_{\pm 0.039}$	$0.828_{\pm 0.017}$	$6.953_{\pm 0.068}$	$0.015_{\pm 0.001}$
10000	ReliefF	–	$0.830_{\pm 0.011}$	$1201.116_{\pm 3.488}$	$11.700_{\pm 0.034}$
	ALS + Dg.	$0.805_{\pm 0.030}$	$0.862_{\pm 0.008}$	$6.618_{\pm 0.007}$	$0.011_{\pm 0.000}$
	ALS + Ex.	$0.396_{\pm 0.029}$	$0.876_{\pm 0.006}$	$6.621_{\pm 0.009}$	$0.012_{\pm 0.001}$
	Lin. + Dg.	$0.848_{\pm 0.026}$	$0.850_{\pm 0.016}$	$7.543_{\pm 0.010}$	$0.024_{\pm 0.000}$
	Lin. + Ex.	$0.760_{\pm 0.009}$	$0.847_{\pm 0.002}$	$7.546_{\pm 0.010}$	$0.025_{\pm 0.001}$
20000	ReliefF	–	$0.833_{\pm 0.003}$	$2879.261_{\pm 6.068}$	$28.600_{\pm 0.062}$
	ALS + Dg.	$0.775_{\pm 0.032}$	$0.866_{\pm 0.015}$	$6.791_{\pm 0.010}$	$0.013_{\pm 0.000}$
	ALS + Ex.	$0.441_{\pm 0.023}$	$0.858_{\pm 0.006}$	$6.795_{\pm 0.012}$	$0.014_{\pm 0.001}$
	Lin. + Dg.	$0.817_{\pm 0.019}$	$0.847_{\pm 0.015}$	$8.538_{\pm 0.032}$	$0.051_{\pm 0.001}$
	Lin. + Ex.	$0.698_{\pm 0.033}$	$0.849_{\pm 0.007}$	$8.542_{\pm 0.035}$	$0.053_{\pm 0.001}$



(a) Splice (b) CorrAL100

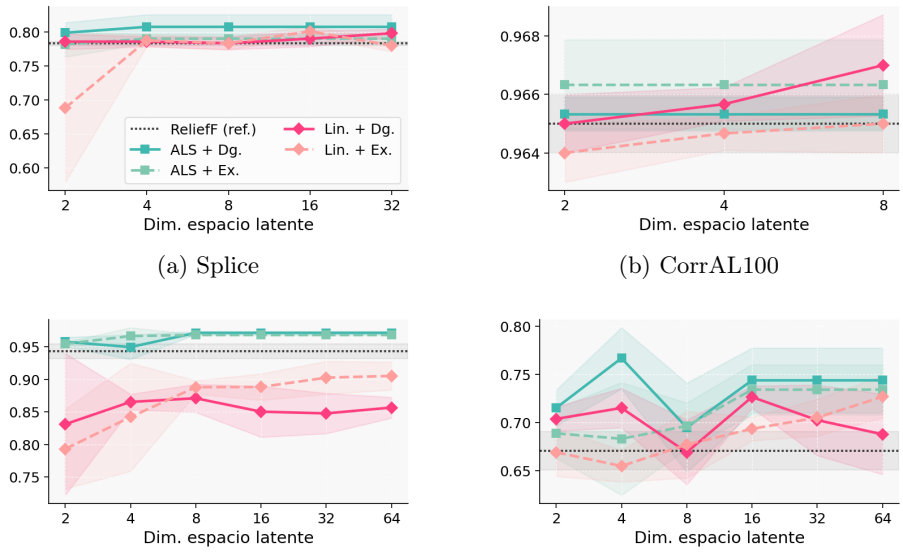
(c) humanActivity (d) MNIST

Figura 2. Exactitud kNN con 50 variables según el número de prototipos por clase.

La ventaja más destacada es la reducción del tiempo de ejecución y del consumo energético. Mientras que ReliefF muestra tiempos que crecen proporcionalmente con el número de muestras, las modificaciones propuestas se mantienen estables en todos los escenarios, lo que supone una aceleración de entre 70 y 400 veces. Dicha ventaja se refleja igualmente en el consumo energético, con reducciones de hasta tres órdenes de magnitud. Las diferencias entre la variante exacta y la diagonal son mínimas tanto en precisión como en tiempo, dado que el coste del cálculo de la puntuación es despreciable frente al entrenamiento de los prototipos y la matriz de proyección.

Sensibilidad: Prototipos. El segundo de los experimentos emplea ya los cuatro conjuntos de datos planteados. La Figura 2 muestra la precisión de clasificación al variar el número de prototipos por clase en el método que emplea la capa lineal. Se observa que, en general, incluso con un único prototipo los métodos propuestos alcanzan resultados comparables o superiores a ReliefF, sin que un mayor número de prototipos aporte mejoras consistentes. La única excepción es *humanActivity*, donde la selección de hiperparámetros de entrenamiento de la capa lineal parece resultar más determinante que el número de prototipos.

Sensibilidad: Espacio Latente. La Figura 3 presenta el impacto de la dimensión r del espacio latente. Para *splice* se exploraron dimensiones hasta $r = 32$, al tener sólo 60 características totales y para *CorrAL100* hasta $r = 8$, dado que sólo



(a) Splice (b) CorrAL100

(c) humanActivity (d) MNIST

Figura 3. Exactitud para kNN con 50 variables según la dimensión del espacio latente.

5 de sus 100 características son informativas. En general, comprimir la proyección a $r = 2$ dimensiones produce una ligera caída de precisión, especialmente en conjuntos con más características. A medida que aumenta, los resultados tienden a mejorar, estabilizándose a partir de $r = 16$, siendo el cálculo de la puntuación exacta la que más se beneficia del incremento de dimensionalidad.

Resultados globales. Con todo lo anterior y, para comprobar cómo se ven afectadas el resto de métricas se muestran los resultados para cada uno de los conjuntos de datos, empleando las métricas que obtuvieron mejores resultados en los casos anteriores. Se fijaron 2 prototipos por clase y una dimensión de proyección de dimensionalidad $r = 16$, salvo en el caso de CorrAL100, donde se empleó $r = 8$. La Tabla 3 resume los resultados para dicha configuración. Nuevamente los *rankings* muestran diferencias frente a ReliefF, especialmente en el caso de la versión que combina ALS con el cálculo exacto. Sin embargo, todas las combinaciones obtienen una exactitud de clasificación igual o superior a ReliefF en tres de los cuatro conjuntos, con la excepción parcial de *humanActivity*, donde la capa lineal sin ajuste de hiperparámetros queda ligeramente por debajo. En cuanto a las métricas de tiempo y consumo, por regla general, se reducen significativamente los tiempos, ligados al número de muestras disponible en cada uno de los conjuntos de datos. En CorrAL100, el conjunto más grande con 100 000 muestras, el tiempo de ReliefF supera los 15 000 segundos, mientras que ALS se ejecuta en aproximadamente 7 segundos y la opción basada en la capa

Tabla 3. Resultados globales con configuración fija de 2 prototipos por clase y dimensión $r = 16$, $r = 8$ para el caso de CorrAL100. Se muestra la similitud de los *rankings*, exactitud para kNN con 20 características, tiempo de ejecución y consumo.

Dataset	Método	Rankings	Exactitud	Tiempo (s)	Consumo (Wh)
splice	ReliefF	–	$0.783_{\pm 0.003}$	$20.450_{\pm 1.455}$	$0.146_{\pm 0.001}$
	ALS + Dg.	$0.269_{\pm 0.074}$	$0.807_{\pm 0.018}$	$6.436_{\pm 0.010}$	$0.009_{\pm 0.000}$
	ALS + Ex.	$0.294_{\pm 0.046}$	$0.790_{\pm 0.007}$	$6.439_{\pm 0.012}$	$0.009_{\pm 0.000}$
	Lin. + Dg.	$0.257_{\pm 0.066}$	$0.801_{\pm 0.004}$	$6.954_{\pm 0.455}$	$0.013_{\pm 0.001}$
	Lin. + Ex.	$0.277_{\pm 0.063}$	$0.783_{\pm 0.016}$	$6.957_{\pm 0.458}$	$0.014_{\pm 0.001}$
CorrAL	ReliefF	–	$0.965_{\pm 0.001}$	$15090.428_{\pm 122.713}$	$141.000_{\pm 1.221}$
	ALS + Dg.	$0.169_{\pm 0.021}$	$0.965_{\pm 0.001}$	$6.914_{\pm 0.012}$	$0.015_{\pm 0.006}$
	ALS + Ex.	$0.120_{\pm 0.155}$	$0.966_{\pm 0.002}$	$6.918_{\pm 0.015}$	$0.016_{\pm 0.007}$
	Lin. + Dg.	$0.184_{\pm 0.027}$	$0.966_{\pm 0.001}$	$15.704_{\pm 0.047}$	$0.245_{\pm 0.002}$
	Lin. + Ex.	$0.140_{\pm 0.107}$	$0.965_{\pm 0.001}$	$15.709_{\pm 0.050}$	$0.247_{\pm 0.003}$
human.	ReliefF	–	$0.943_{\pm 0.012}$	$340.495_{\pm 1.122}$	$3.300_{\pm 0.002}$
	ALS + Dg.	$0.619_{\pm 0.058}$	$0.972_{\pm 0.002}$	$6.753_{\pm 0.410}$	$0.010_{\pm 0.000}$
	ALS + Ex.	$0.164_{\pm 0.009}$	$0.968_{\pm 0.001}$	$6.756_{\pm 0.412}$	$0.010_{\pm 0.000}$
	Lin. + Dg.	$0.502_{\pm 0.086}$	$0.841_{\pm 0.028}$	$7.089_{\pm 0.011}$	$0.017_{\pm 0.000}$
	Lin. + Ex.	$0.398_{\pm 0.013}$	$0.885_{\pm 0.015}$	$7.092_{\pm 0.013}$	$0.018_{\pm 0.000}$
MNIST	ReliefF	–	$0.671_{\pm 0.020}$	$1201.116_{\pm 3.488}$	$11.700_{\pm 0.033}$
	ALS + Dg.	$0.805_{\pm 0.030}$	$0.744_{\pm 0.034}$	$6.618_{\pm 0.007}$	$0.011_{\pm 0.000}$
	ALS + Ex.	$0.396_{\pm 0.029}$	$0.734_{\pm 0.026}$	$6.621_{\pm 0.009}$	$0.011_{\pm 0.000}$
	Lin. + Dg.	$0.848_{\pm 0.026}$	$0.704_{\pm 0.020}$	$7.543_{\pm 0.010}$	$0.024_{\pm 0.000}$
	Lin. + Ex.	$0.760_{\pm 0.009}$	$0.707_{\pm 0.014}$	$7.545_{\pm 0.012}$	$0.025_{\pm 0.001}$

lineal aumenta hasta los 16 segundos. Las diferencias entre las variantes exacta y diagonal son mínimas, tanto en precisión como en eficiencia, confirmando que la aproximación diagonal es suficiente en la mayoría de escenarios.

6. Conclusiones

En este trabajo se ha presentado una versión eficiente y escalable de ReliefF basada en prototipos. El método aprende una proyección lineal y un conjunto de representantes de cada clase en un espacio latente de baja dimensionalidad, desacoplando la búsqueda de vecinos, pasando a realizarse sobre los prototipos en el espacio reducido, del cálculo de relevancia, que se mantiene en el espacio original para preservar la interpretabilidad de las características originales.

Los experimentos realizados sobre cuatro conjuntos de datos de distinto tamaño y naturaleza muestran que las mejoras propuestas obtienen precisiones de clasificación similares o superiores a ReliefF. A su vez, se obtiene una gran reducción tanto del tiempo de ejecución como del consumo energético, destacando en conjuntos de datos con un alto número de muestras.

La variante ALS ofrece la convergencia más rápida y resultados ligeramente superiores. Por su parte, la formulación mediante una capa lineal ofrece una mayor flexibilidad de entrenamiento y constituye una base especialmente adecuada

para futuras extensiones a escenarios federados, donde la optimización mediante gradiente resulta más natural y fácilmente integrable.

Agradecimientos Esta investigación ha sido financiada por la Xunta de Galicia/ FEDER (ED431C 2026/38), el Ministerio de Ciencia e Innovación MCIN/AEI/10.13039/501100011033 bajo la subvención (PID2023-147404OB-I00), junto con "NextGenerationEU"/PRTR del Ministerio para la Transformación Digital y de la Función Pública (TSI-100925-2023-1). CITIC, como miembro de la Red CIGUS, recibe apoyo de la Xunta de Galicia y del Programa Operativo FEDER Galicia 2021-2027 (ED431G 2023/01).

Referencias

1. Eiras-Franco, C., Guijarro-Berdiñas, B., Alonso-Betanzos, A., Bahamonde, A.: Scalable feature selection using ReliefF aided by locality-sensitive hashing. International Journal of Intelligent Systems **36**(11), 6161–6179 (2021)
2. Freda, P.J., Ye, S., Zhang, R., Moore, J.H., Urbanowicz, R.J.: Assessing the limitations of Relief-based algorithms in detecting higher-order interactions. BioData Mining **17**(1), 37 (2024)
3. Gupta, C., Suggala, A.S., Goyal, A., Simhadri, H.V., Paranjape, B., Kumar, A., Goyal, S., Udupa, R., Varma, M., Jain, P.: ProtoNN: Compressed and accurate kNN for resource-scarce devices. In: International conference on machine learning. pp. 1331–1340. PMLR (2017)
4. Guyon, I., Gunn, S., Nikravesh, M., Zadeh, L.A.: Feature extraction: foundations and applications, vol. 207. Springer Science & Business Media (2006)
5. Kaggle, Google LLC: https://www.kaggle.com, accedido el: 2 de Abr. de 2026
6. Kira, K., Rendell, L.A.: A practical approach to feature selection. In: Machine learning proceedings 1992, pp. 249–256. Elsevier (1992)
7. Kononenko, I.: Estimating attributes: Analysis and extensions of Relief. In: European conference on machine learning. pp. 171–182. Springer (1994)
8. LeCun, Y., Cortes, C., Burges, C.: MNIST handwritten digit database. ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist **2** (2010)
9. Liu, K., Chen, Q., Huang, G.H.: An efficient feature selection algorithm for gene families using NMF and ReliefF. Genes **14**(2), 421 (2023)
10. Marengo, A.: Navigating the nexus of AI and IoT: A comprehensive review of data analytics and privacy paradigms. Internet of Things **27**, 101318 (2024)
11. Palma-Mendoza, R.J., Rodriguez, D., De-Marcos, L.: Distributed ReliefF-based feature selection in Spark. Knowledge and Information Systems **57**(1), 1–20 (2018)
12. Reyes, O., Morell, C., Ventura, S.: Scalable extensions of the ReliefF algorithm for weighting and selecting features on the multi-label learning context. Neurocomputing **161**, 168–182 (2015)
13. Shenaj, D., Rizzoli, G., Zanuttigh, P.: Federated learning in computer vision. Ieee Access **11**, 94863–94884 (2023)
14. Šikonja, M.R.: Speeding up Relief algorithm with kd trees. In: Proceedings of Electrotehnical and Computer Science Conference, pages B. pp. 137–140 (1998)
15. Tan, Y., Long, G., Liu, L., Zhou, T., Lu, Q., Jiang, J., Zhang, C.: Fedproto: Federated prototype learning across heterogeneous clients. In: Proceedings of the AAAI conference on artificial intelligence. vol. 36, pp. 8432–8440 (2022)
16. University of California, Irvine, School of Information and Computer Sciences: http://archive.ics.uci.edu/ml, accedido el: 2 de Abr. de 2026