ISSN 1870-4069

Selección de atributos en microarreglos mediante un algoritmo wrapper metaheurístico

Miguel Ángel Garrido-Castañeda, Antonio Alarcón-Paredes,
Oscar Camacho-Nieto

Instituto Politécnico Nacional,[1]
Centro de Investigación en Computación,
México

{mgarridoc2021,aalarcon}@cic.ipn.mx, ocamacho@ipn.mx

Resumen. Las tecnologías de expresión genómica han permitido analizar una gran cantidad de información por medio del análisis de microarreglos, cuyos datos se conforman por unas cuantas observaciones y miles de atributos, muchos de ellos irrelevantes o redundantes, lo que puede llevar a un pobre desempeño en la clasificación. Si bien existen soluciones para la selección de atributos, como los wrappers, aunque son muy utilizados suelen ser computacionalmente costosos debido a su método de búsqueda. En este trabajo se presenta un algoritmo que realiza un pre-filtrado de atributos para su posterior uso en un wrapper metaheurístico. Se llevó a cabo una experimentación tomando en cuenta once conjuntos de datos de microarreglos utilizando cuatro metaheurísticas y diez algoritmos de clasificación en el estado del arte. Se demuestra que el desempeño del método propuesto ofrece mejoras en cuanto al desempeño de clasificación y/o tiempo respecto de utilizar el conjunto completo de datos y de los wrappers tradicionales.

Palabras clave: Wrappers, metaheurística, microarrays, clasificación.

Feature selection in microarrays using a metaheuristic wrapper algorithm

Abstract. Genomic expression technologies have made it possible to analyze a large amount of information through microarray analysis, whose data is made up of a few observations and thousands of attributes, many of them irrelevant or redundant, which can lead to poor performance in gene expression. classification. Although there are solutions for the selection of attributes, such as wrappers, although they are widely used, they are usually computationally expensive due to their search method. This paper presents an algorithm that performs a pre-filtering of attributes for later use in a metaheuristic wrapper. An experimentation was carried out taking into account eleven microarray datasets using four metaheuristics and ten state-of-the-art classification algorithms. It is shown that the performance of the proposed method offers improvements in terms of

classification performance and/or time compared to using the full data set and traditional wrappers.

Keywords: Wrappers, metaheuristics, microarrays, classification.

1. Introducción

Los microarreglos son una tecnología de amplio uso en la expresión genética; estos son estructuras de alta dimensionalidad que permiten agrupar un número muy grande de genes. Actualmente es posible encontrar conjuntos de datos (datasets) obtenidos a partir de esta tecnología, que comúnmente se conforman por pocas muestras (siendo m el número total de éstas) y muchos genes (con p como el número total de genes o atributos) representando los valores del dataset con tipo de dato numérico. Esta tecnología tiene la finalidad de monitorear y medir datos relevantes de la información biológica que permita comprender y analizar contextos específicos como el diagnóstico de enfermedades o clasificación de tumores [1,2,3]. Por tal motivo frecuentemente se une con el aprendizaje automático para realizar estas tareas. Sin embargo, han sido detectados algunos inconvenientes al tratar de aplicar algoritmos de clasificación con este tipo de datos específicos.

Dichos inconvenientes caen en la problemática de la maldición de la dimensionalidad, siendo el primero originado por las pocas muestras que se tienen en los conjuntos de datos ocasionando un deficiente entrenamiento del clasificador y posteriormente una pobre capacidad de predicción del mismo, y el segundo siendo ocasionado por presentarse un conjunto de datos que contiene un número muy grande de atributos que ocasionará que se tenga una probabilidad de que muchas de sus características estén correlacionadas o sean irrelevantes lo que al final ocasiona un pobre desempeño de clasificación acompañado de un costo computacional muy elevado [4].

Existen técnicas de pre procesamiento de datos en el estado del arte que ayudan a contrarrestar este tipo de problemáticas, como los métodos de selección de atributos que en concreto se encargan de seleccionar los atributos no correlacionados con la mayor información discriminativa con los cuales se forma un pequeño subconjunto el cual es mostrado al clasificador esperando que no degenere su desempeño de clasificación [4,5,6].

Dependiendo de la forma en que operen, existen 3 categorías principales de métodos para la selección de atributos: métodos de filtrado (conocidos como *filters* en inglés), métodos de envoltura (conocidos como *wrappers* en inglés) y métodos integrados (conocidos como *embedded methods* en inglés).

Los filters eliminan los atributos correlacionados antes de llegar a la fase de clasificación, normalmente evaluando y seleccionando por separado a cada uno de los atributos [7]. Por otro lado, los métodos wrappers se unen al clasificador para evaluar los distintos subconjuntos de atributos, siendo el elegido el de mayor desempeño de clasificación. Este método suele destacar con mejores resultados de clasificación comparado con los otros; sin embargo, tienden a requerir un mayor número de recursos computacionales y por lo tanto un mayor tiempo de procesamiento siendo éste un aspecto importante a tomar en cuenta cuando se trabaja con estos métodos [8,9].

Por último, en los métodos embedded la selección de atributos son parte de la construcción del modelo de clasificación, donde atributos fuertemente correlacionados están fuera o dentro del modelo por completo mediante algún tipo de penalización, donde comúnmente suelen utilizar la norma L1, L2, alguna combinación de ellas o bien un análisis discriminante disperso [10,11,12].

Otro tipo de complejidad que se suele encontrar al trabajar en conjuntos de datos de microarreglos es el desbalance de clases, para estos casos obtener el desempeño de un clasificador entrenado con este tipo de datos no basta con calcular el *accuracy* (número de aciertos entre total de datos) ya que la desproporción de las clases no permitiría que se refleje el verdadero desempeño del clasificador.

Para estos casos se hace una distinción entre las clases del conjunto de datos en clase positiva (P) y negativa (N) (para conjuntos de datos de dos clases). Dicha distinción permite crear la matriz de confusión conformada por los verdaderos positivos: el número de datos P clasificados como clase P (TP); los falsos negativos: número de datos P clasificados como clase N (FN); los verdaderos negativos: número de datos N clasificados como clase N (TN); y los falsos positivos: el número de datos N clasificados como clase P (FP).

Con la obtención de estos valores es posible implementar medidas de desempeño, que sean útiles en conjuntos de datos desbalanceados, como pueden ser: la sensibilidad (*sensitivity*: $\frac{TP}{TP+FN}$), especificidad (*specificity*: $\frac{TN}{TN+FP}$) y el *balanced accuracy* (promedio obtenido por las dos métricas anteriores).

2. Método propuesto

El presente método se propone con la finalidad de atacar el problema de la maldición de la dimensionalidad con un evidente objetivo primordial: disminuir la cantidad de atributos necesarios para la clasificación de conjuntos de datos altamente dimensionales, como son los microarreglos. Sin embargo, la correcta elección de un subconjunto de atributos lleva consigo una solución en dos vertientes: en primera instancia, se disminuye el tiempo requerido para su clasificación, y en segunda, los algoritmos de clasificación pueden obtener un mejor desempeño. Una vista general del algoritmo puede verse en la Figura 1.

El presente método retoma una idea propuesta en [13], donde se propone iniciar con un ranking de atributos y un pre-filtrado de forma que se tomen los primeros $n<p$ atributos en el orden de importancia dado por el ranking.

Tras ese pre-filtrado, se propone un método wrapper metaheurístico (WMH), donde se llevan a cabo experimentos utilizando cuatro algoritmos, a saber: Bat Optimization (BAT) [14], Grey Wolf Optimization (GWO) [15], Firefly Optimization Algorithm (FFA) [16] y el conocido Particle Swarm Optimization (PSO) [17]. Las partes constituyentes de la propuesta se explican a detalle a continuación.

2.1. Pre-filtrado

Esta primera etapa es muy útil cuando se trabaja con datos altamente dimensionales. Su propósito es el de discriminar tempranamente aquellos atributos que son menos representativos, de forma similar a como trabaja un método filter de selección de

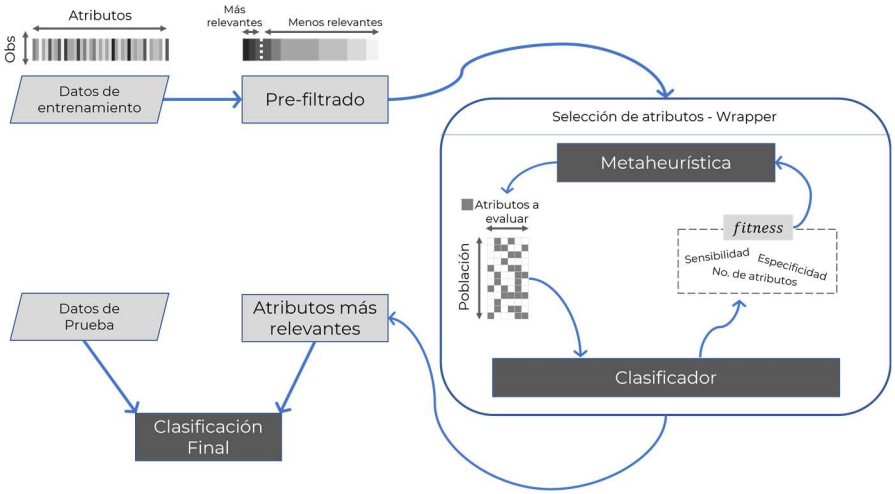

Fig. 1. Modelo propuesto.

atributos. Se lleva a cabo al evaluar individualmente la relevancia de cada atributo con respecto del vector de clase. Para este propósito, el criterio de evaluación de cada atributo consiste en clasificar el conjunto de datos con ese atributo únicamente; la métrica de rendimiento utilizada en este trabajo es el balanced accuracy.

Sea $X = [f_1, f_2, \dots, f_p] = [X_1, X_2, \dots, X_m]^T$ una matriz de tamaño $m \times p$ con m observaciones y p atributos, donde f_i y $X_i$son la i −ésima columna e i −ésima fila, respectivamente; y sea $y = [y_1, y_2, \dots, y_m]^T$su correspondiente vector de clases. Los atributos son evaluados al medir el desempeño del clasificador *clf* de tal forma que $Balanced\ accuracy(i) = clf(f_i, y)$, y se utiliza el método de validación leave-one-out [18]. Posteriormente, se ordena este resultado de mayor a menor en el vector *SortedBAcc*, y se toman los n mejores atributos, con $n < p$, tal que el nuevo conjunto de datos se construye como:

$$Xs = X\,[:, SortedBAcc[1:n]]. \tag{1}$$

2.2. Selección de atributos con un wrapper metaheurístico

Los métodos wrapper evalúan diferentes subconjuntos de atributos para encontrar aquel que obtenga el mejor desempeño en función de un algoritmo de clasificación. Es claro que para encontrar el óptimo sería necesario probar todas las combinaciones posibles, lo que es igual que aplicar fuerza bruta. Comúnmente estos algoritmos utilizan algún método de búsqueda voraz para encontrar dicho subconjunto.

En su lugar, se propone la utilización de una metaheurística como método de búsqueda. De forma general, las metaheurísticas son algoritmos de optimización que permiten encontrar una solución en un espacio continuo. Sin embargo, el problema de la selección de atributos es un problema en un espacio multidimensional binario, por ello, se consideró una modificación binaria de las metaheurísticas utilizadas.

Tabla 1. Datasets de microarreglos de cáncer, características y descripción.

Dataset	Atributos	Muestras	Clases	Distribución de clases (%)	Descripción
9 tumors	5,726	60	9	15 / 12 / 13 / 10 / 10 / 13 / 13 / 4 / 10	9 tipos de tumor
11 tumors	12,533	174	11	16 / 5 / 15 / 13 / 7 / 6 / 4 / 15 / 3 / 8 / 8	11 tipos de tumor
14 tumors	15,009	308	26	10 / 4 / 6 / 5 / 7 / 4 / 3 / 3 / 5 / 4 / 4 / 5 / 4 / 6 / 2 / 3 / 2 / 4 / 2 / 2 / 2 / 2 / 4 / 3 / 1 / 3	14 tipos de tumor y subclases
Braintumor 1	5,920	90	5	67 / 11 / 11 / 4 / 7	5 tipos de tumor cerebral
Braintumor 2	10,367	50	4	28 / 14 / 28 / 30	4 tipos de gliomas
DLBCL	5,469	77	2	75 / 25	Linfomas B difusos de células grandes
Leukemia 1	5,327	72	3	53 / 13 / 34	3 tipos de leucemia: mieloide aguda y linfocítica aguda de células B y T
Leukemia 2	11,225	72	3	39 / 33 / 28	3 tipos de leucemia aguda: mieloide, linfocítica y mixta
Lung Cancer	12,600	203	5	68 / 9 / 10 / 10 / 3	4 tipos de cáncer de pulmón, y tejido sano
Prostate tumor	10,509	102	2	49 / 51	Tumor de próstata o sanos
SRBCT	2,308	83	4	35 / 30 / 13 / 22	Tumores de células pequeñas y redondas

3. Experimentación

Para la fase experimental se tomaron en cuenta 11 datos de microarreglos [19], que pueden ser consultados en la Tabla 1. Además, 10 clasificadores del estado del arte (3-NN [20], 5-NN[20], Naive Bayes [21], Multilayer Perceptron [22], Random Forest [23], AdaBoost [24], Extra Trees [25], Decision Tree [26], SVM-lineal [27], SVM-RBF

[27]) fueron utilizados con las 4 metaheurísticas previamente mencionadas (PSO, GWO, FFA, BAT). En esta sección se presentan los detalles de la implementación, así como también los resultados obtenidos.

3.1. Datos

Se muestra en la Tabla 1 algunos detalles de los 11 datos de microarreglos usados para probar el método propuesto. Como particularidad de este tipo de datasets el número de atributos contenido en cada uno de estos supera ampliamente al número de observaciones, debido a la naturaleza de los microarreglos (matrices con miles de celdas) al obtenerse el dataset cada microarreglo queda representado como un vector de muy alta dimensionalidad y con valores numéricos representando el contenido de cada una de sus celdas. Además, suele ser común que algunos de ellos presenten la problemática del desbalanceo de clases, como puede verse la distribución de las mismas en la tabla descriptiva de los datos.

3.2. Condiciones experimentales

El algoritmo propuesto fue implementado en Python 3.8 utilizando la biblioteca scikit-learn para los algoritmos de Machine learning, y haciendo una modificación a las metaheurísticas propuestas en [28, 29]. Los parámetros de los clasificadores utilizados son los que se incluyen por defecto en scikit-learn.

Para proporcionar una válida experimentación, se dividió el conjunto de datos en dos particiones, de forma que el 75% de ellos se utilizó para llevar a cabo todo el procesamiento de selección de población en las metaheurísticas y dentro de ellas se usó un método de validación cruzada interno con 3 folds (por el desbalance de clases), mientras que el 25% restante de la primera división se utilizó para evaluar el resultado de los mejores subconjuntos de atributos. Debido al desbalance de clases, se eligió el balanced accuracy como métrica para evaluar el desempeño de los clasificadores, que de forma general puede ser expresada como sigue:

$$Balanced\ Accuracy = \frac{1}{K}\sum_{i=1}^{K}\left(\frac{TP_i}{P_i}\right), \tag{2}$$

donde TP_i y P_i representan las observaciones bien clasificadas de la clase i, y el total de observaciones de la misma clase i, respectivamente, mientras que K representa el total de clases del conjunto de datos.

Se usó también la prueba estadística de Friedman, que es una prueba no paramétrica basada en las evaluaciones promedio de los algoritmos de clasificación.

Ésta permite por un lado identificar si existen diferencias significativas en los resultados de clasificación y, por otro lado, establecer un orden de mejor clasificación.

Se tomaron en cuenta diez algoritmos de clasificación, basados en diferentes enfoques y que son de los más utilizados en el estado del arte. Como se mencionó previamente, estos son: 3-NN, 5-NN, Naive Bayes, MLP, Random Forest, AdaBoost, Decision Tree, Extra Trees, SVM-Lineal, SVM-RBF.

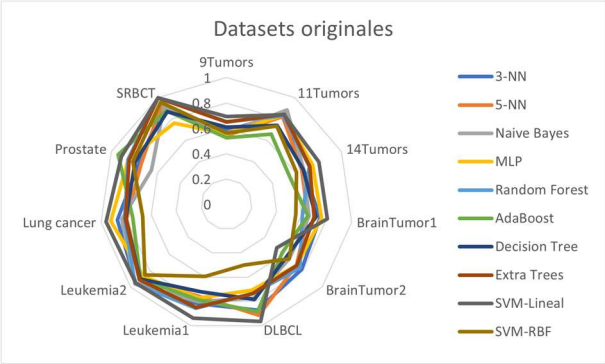

Fig. 2. Resultado datasets con todos los atributos.

Todos los experimentos se corrieron en una computadora con sistema operativo Windows 10, un procesador Intel Core i5 (2.40 GHz) de 11ª generación, con 8GB de memoria RAM instalada.

3.3. Modelo propuesto

3.3.1.Tamaño de la población

Si bien las metaheurísticas referidas tienen un espacio de búsqueda continuo, se lleva a cabo una modificación en la generación de la población, estableciendo un umbral en los valores continuos con la finalidad de binarizar dicho espacio. Uno de los parámetros para utilizar las metaheurísticas es el tamaño de la población, que está íntimamente relacionado con la cantidad de atributos.

Por ejemplo, si un conjunto de datos tiene 3 atributos, existen $2^3 - 2 = 6$ posibles combinaciones para seleccionar (con un 1) o descartar (con un 0) cada uno de esos tres atributos: (0,0,1), (0,1,0), (0,1,1), (1,0,0), (1,0,1), (1,1,0). Sin las combinaciones con todos 0 o todos 1 ya que no seleccionaría ningún atributo o se tomarían en cuenta todos ellos.

Así pues, para un conjunto de datos con 3 atributos, un tamaño de población mayor a 6 podría representar que tome en cuenta todas las combinaciones posibles de atributos. No obstante, esta situación es muy diferente para conjuntos de datos con más atributos, por ejemplo, uno con 10 atributos tendría $2^{10} - 2 = 1022$ posibles combinaciones; el dataset de microarreglos más pequeño de los aquí analizados tiene 2308 atributos, lo que representaría una cantidad impensable de combinaciones.

Por lo anterior, el tamaño de población para datos con atributos $p<9$, es $2\times p$, y en cualquier otro caso el tamaño de población es igual a 200.

3.3.2. Función fitness

Partiendo de que los datos de microarreglos presentan en su mayoría desbalanceo de clases, y que uno de los objetivos es seleccionar la menor cantidad de atributos posible,

para la creación de la función fitness se consideraron valores tales como la sensibilidad (Sen) y especificidad (Spe) además del número de atributos seleccionados (n_s).

La penalización por seleccionar más atributos se considera como la cantidad total de atributos seleccionados n_s sobre los atributos n que ingresan a la metaheurística, y se le asocia un valor de ponderación W_n en caso de que se desee aplicar una mayor o menor penalización; por defecto aquí se tiene un esquema conservador con $W_n = 0.1$.

El hecho de buscar el mejor desempeño con una menor cantidad de atributos constituye un problema de maximización – minimización, y los métodos de optimización aquí usados corresponden a minimización, el desempeño del algoritmo se invierte al restársele al valor máximo si clasificara correctamente todo, que es 1.

De tal forma que la función fitness se propone como:

$$fitness = 1 - (Sen + Spe) + W_n \frac{n_s}{n}. \quad (3)$$

3.3.3. Propuesta general

El método propuesto funciona de la siguiente manera:

1) Se evalúa cada atributo del dataset usando como métrica el balanced accuracy, de modo que se realiza un pre-filtrado de atributos seleccionando los m mejores, de acuerdo con el algoritmo de clasificación seleccionado, como se explica en la subsección 2.1,
2) El dataset obtenido, con el número de atributos disminuido se analiza usando una de las metaheurísticas mencionadas, de manera que se genera una población de posibles soluciones iniciales; esto es, una población binaria con valores 0 y 1, indicando cuáles atributos habrán de seleccionarse en cada miembro de la población inicial, como se detalla en la subsección 3.1.1. Se divide el conjunto de datos en dos particiones: el 75% de ellos se dedica al procedimiento de las metaheurísticas para generar poblaciones más aptas, y el 25% restante se utiliza para evaluar el desempeño con la nueva población,
3) Se lleva a cabo el procedimiento de las metaheurísticas generando, en cada una de sus iteraciones, poblaciones más aptas, es decir, va generando soluciones donde se seleccionan diferentes atributos que mejoren el desempeño de la función fitness propuesta para este trabajo, ver subsección 3.1.2,
4) Finalmente se toma en consideración el subconjunto de atributos que obtiene un mejor valor en la función fitness; esto es, mejora el desempeño del balanced accuracy, y a su vez disminuye la cantidad de atributos utilizados en la clasificación del dataset.

4. Resultados y discusión

En esta sección se muestra una comparativa entre los resultados obtenidos con el método propuesto y los métodos wrapper convencionales que utilizan un método de búsqueda voraz para encontrar el mejor subconjunto de datos.

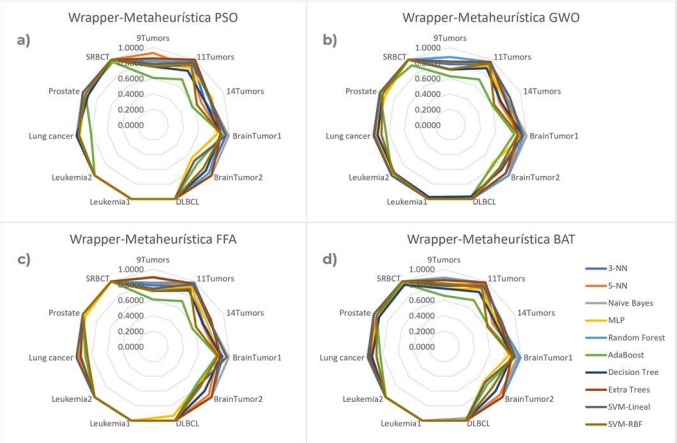

Fig. 2. Resultado datasets con todos los atributos.

Tabla 2. Mejores desempeños en clasificación para cada método.

Dataset	WMH-PSO	WMH-GWO	WMH-FFA	WMH-BAT	Wrapper base	Dataset Original
9 tumors	**0.92** (54)	0.876 (51)	0.896 (32)	0.892 (46)	0.75 (16)	0.691
11 tumors	**1.0** (118)	0.967 (91)	0.967 (103)	0.983 (123)	0.89 (15)	0.887
14 tumors	0.836 (131)	**0.849** (144)	0.824 (130)	0.799 (119)	0.763 (58)	0.802
Braintumor1	0.988 (49)	**1.0** (47)	0.988 (59)	**1.0** (55)	0.933 (10)	0.806
Braintumor2	**1.0** (115)	**1.0** (95)	**1.0** (83)	**1.0** (101)	0.98 (5)	0.786
DLBCL	**1.0** (33)	**1.0** (49)	**1.0** (42)	**1.0** (48)	**1.0** (4)	0.963
Leukemia1	**1.0** (42)	**1.0** (55)	**1.0** (41)	**1.0** (53)	**1.0** (4)	0.937
Leukemia2	**1.0** (103)	**1.0** (118)	**1.0** (108)	**1.0** (96)	**1.0** (6)	0.955
Lung cancer	**1.0** (125)	**1.0** (118)	**1.0** (132)	**1.0** (109)	0.985 (10)	0.962
Prostate tumor	**1.0** (84)	**1.0** (97)	**1.0** (116)	**1.0** (99)	0.951 (5)	0.941
SRBCT	**1.0** (18)	**1.0** (21)	**1.0** (17)	**1.0** (15)	**1.0** (7)	**1.0**
Promedio	**0.977**	0.972	0.970	0.970	0.932	0.885

Un método wrapper convencional inicia el subconjunto tomando en cuenta el primer atributo y aplica una búsqueda voraz para los atributos restantes agregándolos uno a uno y obteniendo el resultado de clasificación, posteriormente toma como inicio del subconjunto al segundo atributo solo y aplica una búsqueda voraz para los atributos restantes (del tercero hasta el último); este proceso se repite para el tercero, cuarto, quinto y todos los atributos restantes del dataset, por lo que en total evalúa un total de $1 + \sum_{k=0}^{p-1}(p-k) = 1 + p(p-1)/2$ subconjuntos [30].

En primera instancia se presentan los resultados de clasificación sobre el conjunto de datos con todos los atributos, como se ilustra en la Figura 1, donde los mejores

resultados fueron obtenidos por la SVM-Lineal para el 9 Tumors (0.692), 14Tumors (0.802), BrainTumor1 (0.806), DLBCL (0.964), Leukemia1 (0.937), y LungCancer (0.9624); el Naive Bayes obtuvo el mejor resultado para 11Tumors (0.887) y Leukemia2 (0.955); el 3-NN para BrainTumor2 (0.786); AdaBoost para Prostate (0.942); y un empate entre Random Forest, ExtraTrees y SVM-Lineal para SRBCT (1.0).

Para el caso de la propuesta usando PSO (Figura 3a), los mejores resultados fueron obtenidos por el 5-NN para el dataset 9Tumors (0.928), MLP para 14Tumors (0.836), Naive Bayes para BrainTumor1 (0.988); en todos los demás datasets se logró obtener un resultado 1.0 de clasificación: ExtraTrees en 11 Tumors, ExtraTrees y Naive Bayes para BrainTumor2; 3-NN, 5-NN, Random Forest y DecisionTree para LungCancer; Naive Bayes, Random Forest, ExtraTrees y SVM-Lineal para Prostate; y el resto de datasets todos los clasificadores obtuvieron 1.0 con excepción de AdaBoost en SRBCT (0.966).

Cuando se usa GWO (Figura 3b), Random Forest lidera en 9Tumors (0.876) y 11Tumors (0.967), SVM-Lineal en 14Tumors (0.849); en el resto de los datasets al menos tres clasificadores lograron obtener el 1.0 en balanced Accuracy.

Para el caso de FFA (Figura 3c), ExtraTrees obtiene el mejor resultado en 9Tumors (0.897), 11Tumors (0.967) y 14Tumors (0.824); Naive Bayes en BrainTumor1 (0.988), y en el resto de datasets al menos 4 clasificadores obtuvieron 1.0.

Finalmente, cuando se usa WMH-BAT (Figura 3d), Naive Bayes obtiene el mejor resultado para 9Tumors (0.892) y 14Tumors (0.799), Random Forest para 11Tumors (0.984) y BrainTumor1 (1.0); para BrainTumor1 y LungCancer, 3 algoritmos obtuvieron 1.0; y en el resto de los datasets cuando menos 5 clasificadores obtuvieron también la máxima evaluación de balanced Accuracy.

Aunado a lo anterior, también se presenta en la Tabla 2 una comparativa entre los mejores resultados de la propuesta y un método Wrapper tradicional, mostrando los resultados de balanced Accuracy, y la cantidad de atributos seleccionados entre paréntesis.

Hasta este punto es posible ver que los algoritmos reportan resultados competitivos. Sin embargo, en la Tabla 3 se muestra el resultado de la prueba estadística de Friedman con los rankings promedio para los algoritmos que están siendo comparados. Esta prueba, además arroja que los resultados difieren significativamente ($p = 1.17e - 6$) con un 95% de confianza.

En la siguiente tabla se muestran los resultados de tiempo de ejecución para los métodos a comparar. En ella se demuestra que los métodos propuestos son más eficientes que el método wrapper tradicional con búsqueda voraz, con excepción del WMH-FFA cuyos tiempos de ejecución lo superan en todos los datasets exceptuando 11Tumors y 14Tumors.

5. Conclusiones

En este manuscrito se presenta un método wrapper metaheurístico que hace uso de cuatro metaheurísticas y diez algoritmos de clasificación sobre once datasets, para establecer una comparativa en cuanto a desempeño de clasificación, número de atributos seleccionados y eficiencia en tiempo de ejecución.

Tabla 3. Rankings promedio de la prueba de Friedman.

Dataset	Ranking promedio
WMH-PSO	**2.5**
WMH-GWO	2.7273
WMH-FFA	2.9091
WMH-BAT	2.8636
Wrapper tradicional	4.4091
Dataset Original	5.5909

Tabla 4. Tiempos de ejecución (en segundos) de los métodos de clasificación.

Dataset	WMH-PSO	WMH-GWO	WMH-FFA	WMH-BAT	Wrapper base
9 tumors	**49.79**	79.61	669.22	50.39	478.00
11 tumors	123.90	225.22	1692.32	**107.99**	1959.00
14 tumors	182.08	307.73	2070.12	**174.80**	18:56:21
Braintumor1	**71.09**	120.44	509.64	72.01	252.00
Braintumor2	98.55	196.11	695.55	**81.78**	231.00
DLBCL	66.69	112.22	302.53	**63.41**	102.00
Leukemia1	**61.45**	102.48	209.06	62.14	85.00
Leukemia2	97.44	178.94	331.53	**78.06**	278.00
Lung cancer	113.38	213.87	1048.80	**89.42**	903.00
Prostate tumor	106.64	202.74	424.15	**80.28**	296.00
SRBCT	**44.45**	62.63	185.94	57.05	72.00
Promedio	92.31	163.81	739.89	**83.39**	423.34

De acuerdo con los resultados obtenidos es posible ver que, aunque el método wrapper tradicional selecciona en general una menor cantidad de atributos en todos los conjuntos de datos, el método propuesto lo supera en cuanto el desempeño de clasificación, usando para ello el balanced Accuracy, siendo el método que usa PSO el que mejores resultados ofrece en este rubro.

Aunado a lo anterior, los tiempos de ejecución se disminuyen en gran medida con la propuesta, a excepción de aquel que usa el Firefly Optimization Algorithm cuyos tiempos son mayores; el mejor desempeño en tiempo se obtiene con el algoritmo Bat Optimization y en segundo lugar el Gray Wolf Optimization.

Es posible notar además que, si bien los métodos wrapper comúnmente se asocian a tiempos grandes de procesamiento, en este caso, el Wrapper Metaheurístico, depende totalmente del tipo de metaheurística que se elija. Por lo tanto, se piensa en un futuro mejorar algún algoritmo existente [31], probar en primera instancia con metaheurísticas que sean eficientes, o bien proponer una metaheurística simple que permita seleccionar los mejores atributos en un menor tiempo. Por otro lado, se sugiere también modificar la función fitness agregando una mayor penalización a la cantidad de atributos

seleccionados, o bien, realizar un pre-filtrado más estricto para tener un menor espacio de búsqueda y por lo tanto seleccionar también una menor cantidad de atributos.

Referencias

1. Chan, W. H., Mohamad, M. S., Deris, S., Zaki, N., Kasim, S., Omatu, S., Al Ashwal, H.: Identification of informative genes and pathways using an improved penalized support vector machine with a weighting scheme. Computers in biology and medicine, 77, pp. 102–115 (2016) doi: 10.1016/j.compbiomed.2016.08.004
2. Chandrashekar, G., Sahin, F.: A survey on feature selection methods. Computers & Electrical Engineering, vol. 40, no. 1, pp. 16–28 (2014) doi: 10.1016/j.compen leceng.2013.11.024
3. Guyon, I., Weston, J., Barnhill, S., Vapnik, V.: Gene selection for cancer classification using support vector machines. Machine learning, vol. 46, no. 1, pp. 389–422 (2002) doi: 10.102 3/A:1012487302797
4. Brezočnik, L., Fister, I., Podgorelec, V.: Swarm intelligence algorithms for feature selection: A review. Applied Sciences, vol. 8, no. 9, 1521 (2018) doi: 10.3390/app 8091521
5. Blum, A. L., Langley, P.: Selection of relevant features and examples in machine learning. Artificial intelligence, vol. 97, no. 1–2, pp. 245–271 (1997) doi: 10.10 16/S0004-3702(97)00063-5
6. Pavlenko, T.: On feature selection, curse-of-dimensionality and error probability in discriminant analysis. Journal of statistical planning and inference, vol. 115, no. 2, pp. 565–584 (2003) doi: 10.1016/S0378-3758(02)00166-0
7. Bolón-Canedo, V., Sánchez-Maroño, N., Alonso-Betanzos, A.: A review of feature selection methods on synthetic data. Knowledge and information systems, vol. 34, no. 3, pp. 483–519 (2013) doi: 10.1007/s10115-012-0487-8
8. Guyon, I., Elisseeff, A.: An introduction to variable and feature selection. Journal of machine learning research, pp. 1157–1182 (2003)
9. Hancer, E.: An improved evolutionary wrapper-filter feature selection approach with a new initialisation scheme. Machine Learning, pp. 1–24 (2021) doi: 10.10 07/s10994-021-05990- z
10. Clemmensen, L., Hastie, T., Witten, D., Ersbøll, B.: Sparse discriminant analysis. Technometrics, vol. 53, no. 4, pp. 406–413 (2011) doi: 10.1198/TECH.2011.08118
11. Tibshirani, R.: Regression shrinkage and selection via the lasso. Journal of the Royal Statistical Society: Series B (Methodological), vol. 58, no. 1, pp. 267–288 (1996) doi: 10.1111/j.2517-6161.1996.tb02080.x
12. Tay, J. K., Narasimhan, B., Hastie, T.: Elastic net regularization paths for all generalized linear models. arXiv preprint arXiv:2103.03475 (2021) doi: 10.4855 0/arXiv.2103.03475
13. Ventura-Molina, E., Alarcón-Paredes, A., Aldape-Pérez, M., Yáñez-Márquez, C., Adolfo Alonso, G.: Gene selection for enhanced classification on microarray data using a weighted k-NN based algorithm. Intelligent Data Analysis, vol. 23, no. 1, pp. 241–253 (2019) doi: 10.3233/IDA-173720
14. Tripathi, D., Ramachandra Reddy, B., Padmanabha Reddy, Y. C. A., Shukla, A. K., Kumar, R. K., Sharma, N. K.: BAT algorithm based feature selection: Application in credit scoring. Journal of Intelligent & Fuzzy Systems, vol. 41, no. 5, pp. 5561–5570 (2021) doi: 10.3233/JIFS-189876.
15. Faris, H., Aljarah, I., Al-Betar, M. A., Mirjalili, S.: Grey wolf optimizer: A review of recent variants and applications. Neural computing and applications, vol. 30, no. 2, pp. 413–435 (2018) doi: 10.1007/s00521-017-3272-5

16. Fister, I., Fister Jr, I., Yang, X. S., Brest, J.: A comprehensive review of firefly algorithms. Swarm and Evolutionary Computation, vol. 13, pp. 34–46 (2013) doi: 10.1016/j.swevo. 2013.06.001
17. Jain, N. K., Nangia, U., Jain, J.: A review of particle swarm optimization. Journal of The Institution of Engineers (India): Series B, vol. 99, no. 4, pp. 407–411 (2018) doi: 10.1007/ s40031-018-0323-y
18. Kearns, M., Ron, D.: Algorithmic stability and sanity-check bounds for Leave-One-Out cross-validation. Neural Computation, vol. 11, no. 6, pp. 1427–1453 (1999) doi: 10.116 2/089976699300016304
19. Cancer microarray data sets. Plymouth University (2005) http://www.tech.plym .ac.uk/spmc/links/bioinformatics/microarray/microarray_cancers.ht ml
20. Denœux, T.: A k-nearest neighbor classification rule based on Dempster-Shafer theory. IEEE Transactions on Systems, Man, and Cybernetics, vol. 25, no. 5, pp. 804–813 (1995)
21. Yang, F. J.: An implementation of Naive Bayes classifier. In: 2018 International Conference on Computational Science and Computational Intelligence CSCI´18, pp. 301–306 (2018) doi: 10.1109/CSCI46756.2018.00065
22. Hush: Classification with neural networks: A performance analysis, In: IEEE 1989 International Conference on Systems Engineering, pp. 277–280 (1989) doi: 10.11 09/ ICSYSE.1989.48672
23. Pal, M.: Random forest classifier for remote sensing classification. International Journal of Remote Sensing, vol. 26, no. 1, pp. 217–222 (2005) doi: 10.1080/01 431160412331269698
24. Schapire, R. E.: Explaining AdaBoost. In: Schölkopf, B., Luo, Z., Vovk, V. (eds) Empirical Inference. Springer, Berlin, Heidelberg pp. 37–52 (2013) doi: 10.10 07/978-3-642- 411366_5
25. Safavian, S. R., Landgrebe, D.: A survey of decision tree classifier methodology. IEEE Transactions on Systems, Man, and Cybernetics, vol. 21, no. 3, pp. 660–674 (1991) doi: 10.1109/21.97458
26. Sharaff, A., Gupta, H.: Extra-tree classifier with metaheuristics approach for email classification. Bhatia, S., Tiwari, S., Mishra, K., Trivedi, M. (eds) Advances in Computer Communication and Computational Sciences. Advances in Intelligent Systems and Computing, Springer, Singapore vol. 924, pp. 189–197 (2019) doi: 10.1007/978-981-13- 6861-5_17
27. Chauhan, V. K., Dahiya, K. Sharma, A.: Problem formulations and solvers in linear SVM: A review. Artif Intell Rev 52, pp. 803–855 (2019) doi:10.1007/s10462-018-9614-6
28. Khurma, R.A., Aljarah, I., Sharieh, A., Mirjalili, S. (2020). EvoloPy-FS: An open-source nature-inspired optimization framework in python for feature selection. In: Mirjalili, S., Faris, H., Aljarah, I. (eds) Evolutionary Machine Learning Techniques. Algorithms for Intelligent Systems, Springer, Singapore, pp. 131–173 (2020) doi: 10.1007/978-981-32- 9990-0_8
29. Hossam, F, Aljarah, I., Mirjalili, S., Castillo, P. A., Merelo-Guervós, J. J.: EvoloPy: An open-source nature-inspired optimization framework in python. In: IJCCI (ECTA), pp. 171–177 (2016)
30. Tang, J., Alelyani, S., Liu, H.: Feature selection for classification: A review, Data Classif, Algorithms Appl, 37 (2014)
31. Bezdan, T., Cvetnic, D., Gajic, L., Zivkovic, M., Strumberger, I., Bacanin, N.: Feature selection by firefly algorithm with improved initialization strategy. In: 7th Conference on the Engineering of Computer Based Systems, no. 8, pp. 1–8 (2021) doi: 10.1145/3 459960.3459974