Un algoritmo genético para la agrupación de documentos aplicado en corpus con características diferentes

Dennis Barreda Morales[1] Roque E. López Condori [2] Javier Tejada Cárcamo [1]
Luis Alfaro Casas[2]

[1] Universidad Católica San Pablo

[2]Universidad Nacional San Agustín

casas@unsa.edu.pe, dennis.barreda@ucsp.edu.pe, jtejadac@ucsp.edu.pe,

rlopez27@gmail.com

Resumen

Este artículo presenta un método de agrupación de documentos basado en algoritmos genéticos; este modelo, introduce nuevos operadores genéticos diseñados específicamente para la resolución del problema del agrupamiento como el cruzamiento y mutación, la cual expande la dirección de busqueda y evita convergencia en un máximo local. Las pruebas, se realizán en dos corpus, los cuales presentan documentos con características diferentes, como el tamaño del documento y la frecuencia de las palabras clave. Los resultados obtenidos demuestran que el algoritmo genético logra obtener resultados aceptables para corpus donde las palabras clave aparecen con mayor frecuencia, mientras que en corpus donde las palabras clave tienen una frecuencia mínima, la calidad del agrupamiento cae.

1. Introducción

Llegar a conseguir grupos de documentos, donde cada grupo está formado por textos que comparten características en común (Kaufmann et al., 1990), es una tarea compleja y además que demanda un consumo de bastante tiempo. Puesto que la cantidad de información que actualmente se tiene a la mano aumenta de forma casi exponencial, se hace necesario hacer una categorización para un acceso más rápido y sencillo. Es por esa razón que nace la necesidad de aplicar algoritmos de categorización (Hearst et al., 1996). Una de las características principales del problema de la categorización de documentos es su naturaleza combinatoria (Duran and Odell, 1974). La teoría de la combinación (Liu, 1974) indica que $C(n, K)$, la cantidad de maneras de agrupar n objetos en K grupos, está dada por 1.

$$C(n, K) = 1/K! \sum_{I=0}^{K} (-1)^{I^k} \mathrm{C}_i (K - I)^n \tag{1}$$

Es decir, si deseamos categorizar 25 textos en 5 grupos, $C(25, 5) = 2 \times (10)^{16}$, existen más de 2000 billones de formas en que se pueden agrupar dichos textos. Entonces un algoritmo que evalúe todo el espacio de soluciones y retorne la mejor agrupación según algún criterio no es la mejor manera de resolver el problema debido al gran espacio de búsqueda. Es por eso que se recurre a heurísticas que tienden a acercarse a la solución óptima (Cutting et al., 1992)(Zamir et al., 1999) (si es que aún no la encontraron), en un tiempo aceptable. En este trabajo se propone el uso de un algoritmo genético adaptado como heurística para poder resolver el problema de la clasificación en el cual no se tiene información a *priori*, dicho trabajo está estructurado

de la siguiente forma: en la sección 2 se describen los trabajos relacionados, en la sección 3 se describe el algoritmo genético y cómo se adapta al problema, en la sección 4 se muestran experimentos realizados y finalmente las conclusiones.

2. Trabajos Previos

Para la agrupación de documentos los recientes estudios muestran que los algoritmos de forma particional son más adecuados para agrupar gran cantidad de documentos, esto se debe a sus bajos requerimientos computacionales y los resultados aceptables que obtienen (Makagonov et al., 2002). En el campo del *clustering*, el algoritmo del *K-means* es el algoritmo más popular y usado para encontrar una partición que minimice la medida de la media cuadrada del error (MSE por sus siglas en inglés *mean square error*). Además *K-means* es un algoritmo de agrupación extensamente útil, aunque éste sufre de varias desventajas. La función objetivo de *K-means* no es convexa, por lo tanto ésta puede contener mínimos locales. En consecuencia, minimizando la función objetivo, existe la posibilidad de quedarse bloqueado en mínimos locales (también en máximos locales y punto de silla)(Selim and Ismail, 1984) . El desempeño del algoritmo *K-means* depende de la elección inicial de los centros de los grupos formados. Además, la norma euclidiana es sensible al ruido o los valores extremos. Por lo tanto el algoritmo *K-means* debería verse afectado por el ruido y los valores atípicos (Wu and Yang, 2002).

Hay trabajos anteriores que aplican algoritmos genéticos y programación evolutiva para la agrupación de documentos. Algunos de ellos realizan la agrupación de un conjunto de objetos asumiendo que el valor apropiado de *k* es conocido (Goldberg., 2002) (Chu et al., 2002)(Murthy and Chowdhury, 1996)(Merz and Zell, 2002) . Sin embargo un algoritmo de agrupación basado en programación evolutiva (Sarkar et al., 1997) junta un conjunto de datos en un óptimo número de *cluster*. Este está basado en el algoritmo *K-means*. Ellos usan dos funciones objetivo que son minimizadas simultáneamente: una da el número óptimo de *clusters*, mientras que la otra conduce a la identificación adecuada de los centroides de cada *cluster*. (Casillas et al.,) sólo utilizan una función objetivo, al mismo tiempo los dos aspectos de la solución se calcula: una aproximación al óptima valor de k, y la mejor agrupación de los objetos en estos k grupos.

3. El algoritmo genético

En esta sección se presenta el algoritmo genético adaptado al problema del agrupamiento de documentos y la descripción de los operadores a utilizarse para llegar a obtener un resultado aceptable.

3.1. Extracción de características

El objetivo de esta etapa es eliminar las partes de los documentos que no sean importantes, es decir, que no aportan significado. Esta etapa consta de los siguientes pasos:

- Eliminación de palabras vacías (*stopwords)*. Elimina palabras que no transmiten información (pronombres, preposiciones, conjunciones, etc.).
- Eliminación de símbolos de puntuación.
- Reducción de palabras a su raíz. Elimina sufijos y afijos de una palabra de tal modo que aparezca sólo su raíz léxica. Por ejemplo, los vocablos *medicina*, *médico*

y *medicinal* tienen la raíz léxica *medic*. Para este paso se utiliza el Algoritmo de Porter (Porter, 1980)

3.2. Representación

Una población está formada por *N* individuos o cromosomas, donde cada individuo es una posible solución al problema, en este caso la solución es el conjunto de *n* documentos agrupados en *K clusters*. Se tiene una cadena de tamaño *n* (número de documentos), donde en cada posición *i* se encuentra un número que está entre 1 y *K* (Jones and Beltramo, 1991), el cual representa el *cluster* al que pertenece. Un ejemplo de la representación se puede ver en la Figura 1.

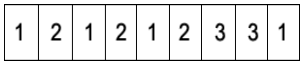

Figura 1: Representación implícita que contiene los grupos {1,3,5,9}{2,4,6}{7,8}

En la Figura 1, se representa la forma implícita del individuo, la forma explícita es como se representa la información a la hora de la programación, esta forma se usa para aprovechar las estructuras que se tienen a la mano. La figura 2 ejemplifica la representación explícita.

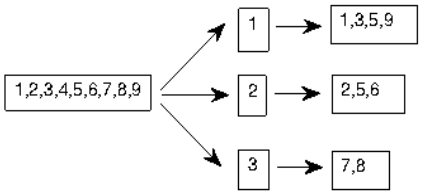

Figura 2: Representación explícita que contiene los grupos {1,3,5,9}{2,4,6}{7,8}

3.3. Población inicial

Los cromosomas de la población inicial se podrían generar de forma totalmente aleatoria, es decir para cada documento del conjunto, generar un número aleatorio *i* que este entre 1 y *K*, y dicho documento se inserta en el grupo que corresponde a *i* (Jones and Beltramo, 1991). Al hacer esto las agrupaciones iniciales son bastante pobres lo que implicaría mayor número de generación de poblaciones para llegar a obtener un agrupamiento óptimo.

Entonces, para que la población pueda converger de manera rápida, se hace una preclasificación, se obtiene un documento *j* cualquiera, que representará al grupo 1 de *K*, se busca otro documento que tenga un valor de similitud menor a un *q* dado, este nuevo documento será el representante del grupo 2, para obtener un representante de un grupo 3, se obtiene un documento que tenga un valor de similitud menor al *q* tanto para el representante del grupo 1 como para el representante del grupo 2, este procedimiento se hace para los *K* grupos.

Una vez obtenidos documentos representantes, los restantes se insertan en el grupo donde el valor de similitud entre éste y el representante del grupo tenga el mayor valor de similitud con respecto a los demás representantes del grupo.

3.4. Funcíon de adaptación *(fitness)*

La función de adaptación debe dar una medida de calidad del cromosoma(Cole, 1998), cromosomas con mayor valor de calidad tienden a sobrevivir a través de las generaciones.

La función *fitness* está definida en el algoritmo 1:

Algoritmo 1

Paso 1: Calcular el vector centroide del grupo, el cual es el promedio de todos los vectores de frecuencia de los documentos que pertenencen a un grupo. La ecuación se muestra en 2.

$$\vec{\mu}(C) = \frac{1}{N_c} \sum_{\vec{x} \in C}^{n} \vec{X} \tag{2}$$

Paso 2: Calcular la distancia coseno, de los vectores de los documentos del grupo contra el centroide. La distancia coseno está definida a continuación. La ecuación de la distancia coseno se muestra en 3

$$cos(\vec{x}, \vec{y}) = \sum_{j}^{n} \frac{x_j \cdot y_j}{\sqrt{\sum_i (x_i)^n} \cdot \sqrt{\sum_i (y_i)^n}} \tag{3}$$

Paso 3: Obtener el promedio de las distancias calculadas.

Paso 4: Sumar los promedios de los K *grupos del cromosoma*.

3.5. Operador de selección

Para la solución propuesta se utiliza el método del Elitismo (Holland, 1975), por los siguientes motivos:

- Siempre escoge los mejores individuos de la población
- Requiere de un ordenamiento previo, pero el tamaño de individuos no suele ser tan grande como para llevar a la solución a ser ineficiente.
- Permite regular la presión de selección.

3.6. Operador de cruzamiento

El operador de cruzamiento está diseñado con los siguientes lineamientos :

- El operador deber ser eficiente.
- El operador debe ser sensible al contexto.
- El operador debe de ofrecer información específica del dominio para generar hijos de buena calidad.

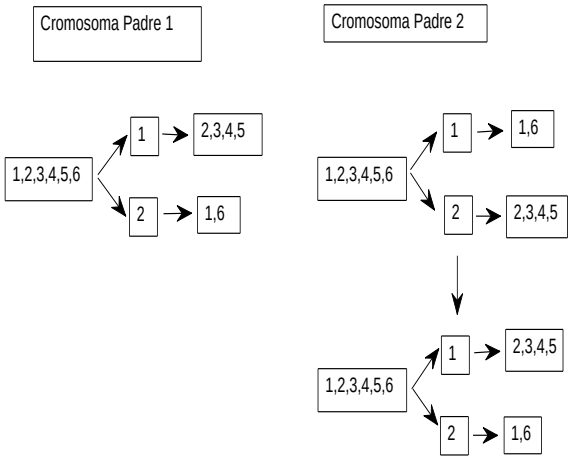

Figura 3: Ordenamiento de un cromosoma

Aquí es donde se aprovecha la representación explícita, para realizar el cruzamiento se hace un ordenamiento previo del segundo cromosoma padre con respecto al primer cromosoma padre. Este ordenamiento se puede apreciar en la figura 3.

En la figura 3, se recorre los *cluster* del cromosoma padre 2, y se les compara con cada *cluster* del cromosoma padre uno, si es que el $cluster_i$ del cromosoma padre 2 tiene el mismo contenido que el $cluster_j$ del cromosoma padre 1, se hace un intercambio para que ambos grupos tengan el mismo número identificador de grupo y ambos grupos similares se muevan al inicio de la enumeración y pasen a la siguiente generación sin modificaciones.

A continuación se genera un número aleatorio que será el punto de corte, para esto se recorre los grupos que están después de los grupos intocables, seguidamente por cada grupo cruzable se recorren los elementos que tiene y se pregunta si el número es mayor que el número de corte, en caso de ser así se busca dicho elemento en los grupos tocables del cromosoma padre 2, cuando se encuentra el grupo que lo contiene se hace un intercambio de grupos para dicho elemento. Un ejemplo se muestra en la Figura 4.

3.7. Operador de mutación

A la mutación se le entiende como la función encargada de evitar que el algoritmo encuentre su óptimo en un máximo local (Holland, 1975), es por eso que muta a un cromosoma de alguna manera, escogido aleatoriamente con una probabilidad ρ. Pero para no dejar todo al azar es que se guía a la función de mutación por un buen camino. Se tienen dos tipos de mutación las cuales se usarán en la evolución de la población.

- Primera mutación: Se selecciona un cromosoma aleatoriamente, y se selecciona algún grupo del cromosoma también, de este grupo se selecciona el documento que tiene la menor distancia coseno contra el centroide del grupo, seguidamente este documento se inserta en algún otro grupo escogido aleatoriamente.
- Segunda mutación: Es parecida a la primera mutación, la diferencia está a la hora de escoger el grupo donde se insertará el documento que mutará. Una vez escogido el documento que mutará se compara contra los demás vectores centroide

del resto de grupos, y se inserta en el grupo con mayor valor de distancia coseno.

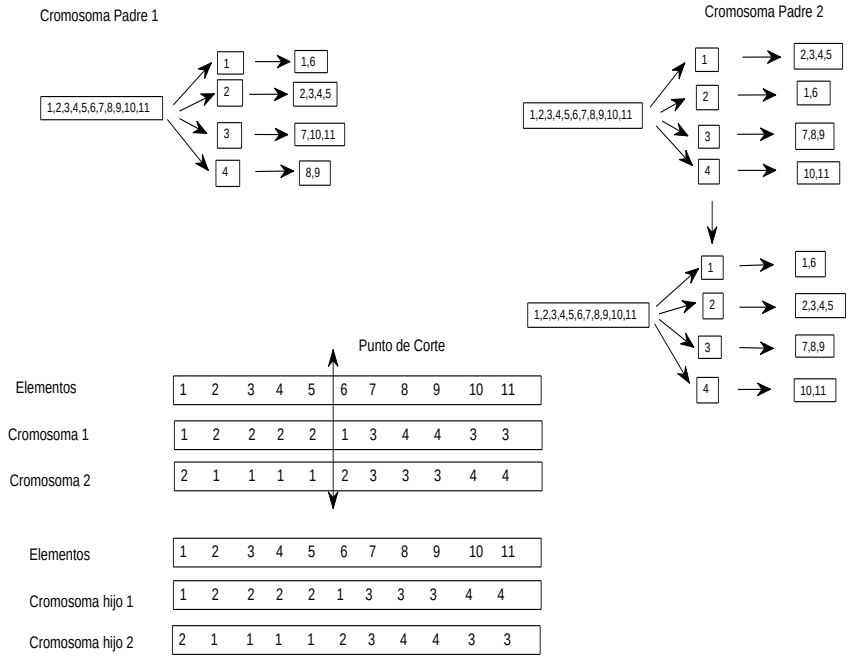

Figura 4: Operador de cruzamiento

4. Pruebas y Resultados

Para la realización de los experimentos se utilizaron datos de la colección de prueba para la categorización de documentos R8* que consta con 2189 documentos, R8 es una subcolección de Reuters-21578 una colección de noticias de la agencia Reuters del año 1987 que son usadas como un estándar para evaluar sistemas. Se eligió R8 ya que presenta gran desbalance con el número de noticias que pertenecen a cada una de las clases y por lo tanto es adecuada para probar si un sistema de clasificación es eficiente. En el cuadro 1 se muestra la cantidad de datos que tiene la colección R8 y como están distribuidos sus documentos, las pruebas se hace con la clase Prueba.

Para la segunda prueba se usa una colección de *historias clínicas* dicha colección fue proveída por una clínica local. Esta colección está formada por 261 documentos los cuales fueron distribuidos por una persona especialista en 6 clases de manera casi uniforme. En el cuadro 2 se muestra la composición del corpus de *historias clínicas*.

Para obtener el número de generaciones para el algoritmo genético se hizo la prueba que se detalla a continuación.

4.1. Número de generaciones

El algoritmo se ejecuto con un número de 50,100,150 y 200 (eje x de la figura 5,6) generaciones, y se puede notar que el *fitness* de las poblaciones (eje y de la figura 5,6) tienden a

R8		
Clase	Entrenamiento	Prueba
Trade	251	75
Grain	41	10
Acq	1596	696
Earn	2840	1083
Interest	190	81
Money-fx	206	87
Ship	108	36
Crude	253	121
Total	5845	2189

Cuadro 1: Distribución de la colección R8

Historias clínicas	
Clase	Documentos
Gastroenterología	41
Ginecología	48
Neurología	42
Oncología	43
Traumatología	42
Urología	45
Total	261

Cuadro 2: Distribución de la colección de historias clínicas.

aumentar considerablemente hasta las *n* primeras generaciones, luego se estabiliza o aumenta el *fitness* muy poco, lo que comparándolo con el tiempo de ejecución no es conveniente generar nuevas poblaciones. El comportamiento se puede apreciar en la Figura 5 para la colección R8 y en la Figura 6 para la colección de las historias clínicas.

4.2. Resultados Representativos del AG

Una vez aplicados los operadores propuestos al algoritmo genético sin perder noción del problema que se está tratando es que se llega a obtener los resultados que se muestran en el cuadro 3 y el cuadro 4 para R8 e historias clínicas respectívamente.

Clase	Trade	Grain	Acq	Earn	Interest	Money-fx	Ship	Crude	Total	Precisión	Recall
Trade	57	1	0	2	6	0	0	4	70	0.76000	0.81429
Grain	0	8	0	5	0	4	0	0	17	0.80000	0.47059
Acq	0	0	654	46	11	0	3	2	716	0.93965	0.91341
Earn	8	0	38	984	0	0	3	2	1035	0.90858	0.95072
Interest	0	1	4	2	59	0	1	0	67	0.72840	0.88060
Money-fx	0	0	0	6	5	82	0	0	93	0.94252	0.88060
Ship	1	0	0	8	0	1	29	3	42	0.80556	0.69047
Crude	9	0	0	30	0	0	0	110	149	0.90909	0.73826

Cuadro 3: Matriz de confusión para el AG aplicado a R8

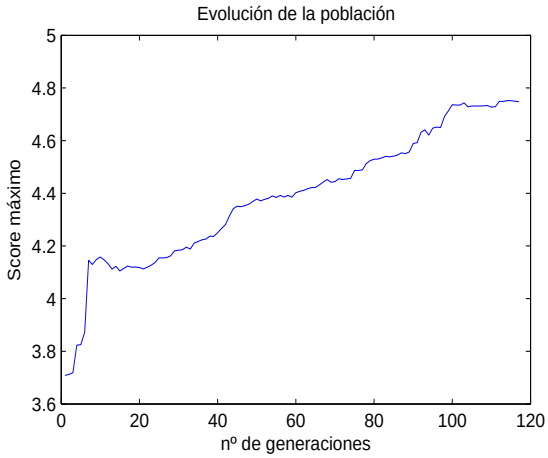

Figura 5: Evolución de la población para R8

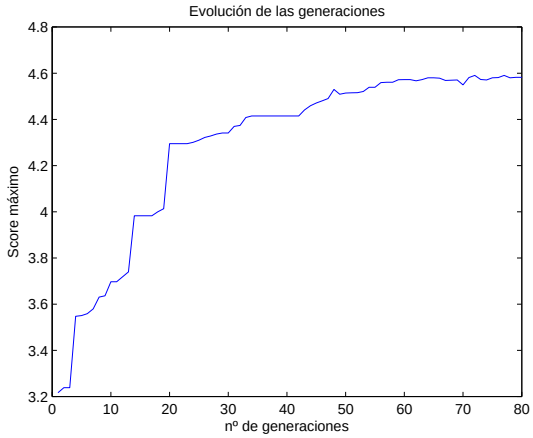

Figura 6: Evolución de la población para historias clínicas

Clase	Gatro	Gineco	Onco	Neuro	Trauma	Uro	Total	Precisión	Recall
Gatro	18	3	0	6	2	4	33	0.43902	0.5454
Gineco	9	15	5	0	0	0	29	0.3125	0.5172
Onco	0	3	18	5	4	1	31	0.4186	0.5806
Neuro	4	7	6	11	3	7	37	0.2619	0.2973
Trauma	7	19	6	17	33	29	112	0.7857	0.2946
Uro	3	1	4	3	0	8	19	0.1778	0.4211

Cuadro 4: Matriz de confusión para el AG aplicado a historias clínicas

5. Conclusiones y Trabajo Futuro

En este documento se propone una adaptación de un algoritmo genético al problema de la categorización automática de documentos, que incluye el diseño de nuevos operadores. Los resultados experimentales obtenidos confirman la tesis que los algoritmos genéticos son una poderosa herramienta para la resolución de problemas en los cuales el espacio de soluciones es amplio y la función de optimización es compleja, siempre y cuadno los operadores sean adaptados al problema. Se ha encontrado también que, tal como lo han afirmado otros autores los algoritmos genéticos no son un método de solución universal de problemas, sino un paradigma que debe adaptarse correctamente al problema a resolver. El algoritmo propuesto logra resultados efectivos para el corpus R8 porque en el diseño del mismo se han adaptado los conceptos que aplican los algoritmos genéticos y se han creado nuevos operadores específicos para el problema a resolver. Pero en el corpus de *historias clínicas* pese a ser el mismo problema el resultado cae debido a la composición de los documentos, es decir los operadores implementados no son específicos para corpus con dichas caracteristicas.

Como trabajo futuro se pretende diseñar nuevos operadores, enfocandonos en corpus con características como el de las *historias clínicas* y adaptarlos a un nuevo algoritmo genético.

Referencias

Casillas, M. T., de Lena, G., and Martinez, R. Document clustering into an unknown number of clusters using a genetic algorithm.

Chu, S., Roddick, J., and Pan, J. (2002). An incremental multi-centroid, multi-run sampling scheme for k-medoids-based algortihms-extended report.

Cole, R. M. (1998). Clustering with genetic algorithms.

Cutting, D. R., Karger, D. R., Pedersen, J. O., and Tukey, J. W. (1992). Scatter/gather: A cluster-based approach to browsing large document collections.

Duran, B. S. and Odell, P. L. (1974). Cluster analysis: A survey.

Goldberg. (2002). D genetic algorithms in search optimization and machine learning, reading.

Hearst, M. A., Pedersen, and O., J. (1996). Reexaminig the cluster hypothesis: Scatter/gather on retrieval results.

Holland, J. (1975). Adaption in natural and artificial systems.

Jones, D. R. and Beltramo, M. A. (1991). Solving partitioning problems with genetic algorithms.

Kaufmann, Leonard, and Rousseuw (1990). Finding groups in data: An introduction to cluster analysis.

Liu, G. (1974). Introduction to combinatorial mathematics.

Makagonov, P., Alexandrov, M., and Gelbukh, A. (2002). Selection of typical documents in a document flow.

Merz, P. and Zell, A. (2002). Clustering gene expresion profiles with memetic algorithms.
Murthy, C. and Chowdhury, N. (1996). In search of optimal clusters using genetic algorithms.
Porter, M. F. (1980). An algorithm for suffix stripping.
Sarkar, M., Yegnanarayana, B., and Khemani, D. (1997). A clustering algorithm using an evolutionary programming-based approach.
Selim, S. and Ismail, M. (1984). K-means type algorithms: *A Generalized Convergence Theorem and Characterization of Local Optimality*.
Wu, K. and Yang, M. (2002). Alternative c-means clustering algorithms.
Zamir, Oren, and Etzioni, O. (1999). Grouper: A dynamic clustering interface to web search results.