Adaptation, Comparison, and Improvement of Metaheuristic Algorithms to the Part-of-Speech Tagging Problem

Miguel-Alexis Solano-Jiménez; José-Julio Tobar-Cifuentes; Luz-Marina Sierra-Martínez; Carlos-Alberto Cobos-Lozada

Citación: M.-A. Solano-Jiménez, J.-J. Tobar-Cifuentes, L.-M. Sierra-Martínez, C.-A. Cobos-Lozada, "Adaptation, Comparison, and Improvement of Metaheuristic Algorithms to the Part-of-Speech Tagging Problem," *Revista Facultad de Ingeniería*, vol. 29 (54), e11762, 2020.
https://doi.org/10.19053/01211129.v29.n54.2020.11762

Recibido: Julio 13, 2020; **Aceptado:** Septiembre 14, 2020; **Publicado:** Septiembre 15, 2020

Conflicto de intereses: Los autores declaran no tener conflicto de intereses.

Adaptation, Comparison, and Improvement of Metaheuristic Algorithms to the Part-of-Speech Tagging Problem

Miguel-Alexis Solano-Jiménez[1]
José-Julio Tobar-Cifuentes[2]
Luz-Marina Sierra-Martínez[3]
Carlos-Alberto Cobos-Lozada[4]

Abstract

Part-of-Speech Tagging (POST) is a complex task in the preprocessing of Natural Language Processing applications. Tagging has been tackled from statistical information and rule-based approaches, making use of a range of methods. Most recently, metaheuristic algorithms have gained attention while being used in a wide variety of knowledge areas, with good results. As a result, they were deployed in this research in a POST problem to assign the best sequence of tags (roles) for the words of a sentence based on information statistics. This process was carried out in two cycles, each of them comprised four phases, allowing the adaptation to the tagging problem in metaheuristic algorithms such as Particle Swarm Optimization, Jaya, Random-Restart Hill Climbing, and a memetic algorithm based on Global-Best Harmony Search as a global optimizer, and on Hill Climbing as a local optimizer. In the consolidation of each algorithm, preliminary experiments were carried out (using cross-validation) to adjust the parameters of each algorithm and, thus, evaluate them

[1] Universidad del Cauca (Popayán-Cauca, Colombia). miguelsolano@unicauca.edu.co. ORCID: 0000-0003-1936-3488
[2] Universidad del Cauca (Popayán-Cauca, Colombia). josej@unicauca.edu.co. ORCID: 0000-0002-5436-0816
[3] Ph. D. Universidad del Cauca (Popayán-Cauca, Colombia). lsierra@unicauca.edu.co. ORCID: 0000-0003-3847-3324
[4] Ph. D. Universidad del Cauca (Popayán-Cauca, Colombia). ccobos@unicauca.edu.co. ORCID: 0000-0002-6263-1911

on the datasets of the complete tagged corpus: IULA (Spanish), Brown (English) and Nasa Yuwe (Nasa). The results obtained by the proposed taggers were compared, and the Friedman and Wilcoxon statistical tests were applied, confirming that the proposed memetic, GBHS Tagger, obtained better results in precision. The proposed taggers make an important contribution to POST for traditional languages (English and Spanish), non-traditional languages (Nasa Yuwe), and their application areas.

Keywords: computational intelligence; computational linguistics; evolutionary computing; heuristic algorithms; natural language processing; parts of speech tagging; search methods.

Adaptación, comparación y mejora de algoritmos metaheurísticos al problema de etiquetado de partes del discurso

Resumen

La identificación de partes del discurso (Part-of-Speech Tagging, POST) es una tarea compleja en las aplicaciones de procesamiento de lenguaje natural. Ha sido abordada desde enfoques basados en información estadística y reglas, haciendo uso de distintos métodos y, últimamente, se destacan los algoritmos metaheurísticos obteniendo buenos resultados. Por ello, se involucran en esta investigación para asignar la mejor secuencia de etiquetas (roles) para las palabras de una oración, basándose en información estadística. Este proceso se desarrolló en 2 ciclos, donde cada ciclo tuvo 4 fases para la adaptación al problema de etiquetado en los algoritmos metaheurísticos Particle Swarm Optimization, Jaya, Random-Restart Hill Climbing, y un algoritmo memético basado en Global-Best Harmony Search como optimizador global, y en Hill Climbing como optimizador local. Se realizaron experimentos preliminares (utilizando validación cruzada), para ajustar los parámetros de cada algoritmo y luego ejecutarlos sobre los *datasets* completos de los corpus etiquetados IULA (castellano), Brown (inglés) y Nasa Yuwe (Nasa). Los resultados obtenidos por los etiquetadores propuestos se compararon mediante las pruebas estadísticas no paramétricas de Friedman y Wilcoxon, ratificando que el memético propuesto, GBHS Tagger, obtiene mejores resultados

de precisión. Los etiquetadores propuestos se convierten en un aporte muy importante para el POST, tanto para lenguas tradicionales (Inglés y Castellano), no tradicionales (Nasa Yuwe), y sus áreas de aplicación.

Palabras clave: algoritmos heurísticos; computación evolutiva; etiquetado de partes del discurso; inteligencia computacional; lingüística computacional; métodos de búsqueda; procesamiento de lenguaje natural.

Adaptação, comparação e melhora de algoritmos metaheurísticos ao problema de etiquetado de partes do discurso

Resumo

A identificação de partes do discurso (Part-of-Speech Tagging, POST) é uma tarefa complexa nas aplicações de processamento de linguagem natural. Tem sido abordada desde enfoques baseados em informação estatística e regras, fazendo uso de distintos métodos e, ultimamente, destacam-se os algoritmos metaheurísticos obtendo bons resultados. Por isso, envolvem-se nesta pesquisa para assignar a melhor sequência de etiquetas (papéis) para as palavras de uma oração, baseando-se em informação estatística. Este processo desenvolveu-se em 2 ciclos, onde cada ciclo teve 4 fases para a adaptação ao problema de etiquetado nos algoritmos metaheurísticos Particle Swarm Optimization, Jaya, Random-Restart Hill Climbing, e um algoritmo mimético baseado em Global-Best Harmony Search como otimizador global, e em Hill Climbing como otimizador local. Realizaram-se experimentos preliminares (utilizando validação cruzada), para ajustar os parâmetros de cada algoritmo e depois executá-los sobre os datasets completos dos corpus etiquetados IULA (castelhano), Brown (inglês) e Nasa Yuwe (Nasa). Os resultados obtidos pelos etiquetadores propostos compararam-se mediante as provas estatísticas não paramétricas de Friedman e Wilcoxon, ratificando que o mimético proposto, GBHS Tagger, obtém melhores resultados de precisão. Os etiquetadores propostos convertem-se em um aporte muito importante para o POST, tanto para línguas tradicionais (Inglês e Castelhano), não tradicionais (Nasa Yuwe), e suas áreas de aplicação.

Palavras chave: algoritmos heurísticos; computação evolutiva; etiquetado de partes do discurso; inteligência computacional; linguística computacional; métodos de busca; processamento de linguagem natural.

I. INTRODUCCIÓN

Los algoritmos metaheurísticos están siendo aplicados en una variedad de áreas de conocimiento, por esto, no es extraño su uso en el problema de Identificación o etiquetado de partes del discurso (Part-of-Speech Tagging, POST), el cual es una tarea compleja y de gran importancia en el preprocesamiento de Lenguaje Natural dados los retos, como son: La ambigüedad de las palabras, el tamaño del conjunto de etiquetas y el etiquetado de palabras desconocidas [1, 2].

Los algoritmos metaheurísticos en el problema de etiquetado (POST) se han utilizado para asignar la mejor secuencia de etiquetas (roles) para las palabras de una oración basándose tanto en información estadística como en la transformación de reglas, para solucionar este problema, obteniendo resultados sobresalientes en contraste con los enfoques tradicionales. Algunos trabajos previos son: 1) Alhasan y Al-taani [3], representan el problema de etiquetado como un grafo, los nodos son las posibles etiquetas de una oración y utilizan el algoritmo de optimización por colonia de abejas (Bee Colony Optimization, BCO), las cuales encuentran el mejor camino de solución. 2) Sierra Martínez, Cobos y Corrales [4], proponen un algoritmo memético para el etiquetado basado en Global-Best Harmony Search (GBHS) que incluye conocimiento del problema mediante una estrategia de optimización local basada en el algoritmo Hill Climbing. 3) Forsati & Shamsfard [5], presentan dos mejoras del etiquetador HSTAgger basado en la metaheurística Harmony Search, llamados HSTAgger(I) y HSTAgger(II), que incrementan la eficiencia de búsqueda y mejoran la selección de nuevas soluciones para la memoria armónica. 4) Ekbal y Saha [6], abordan el problema de etiquetado usando optimización mono-objetivo y multiobjetivo basado en Simulated Annealing-Based Multiobjective Optimization Algorithm propuesto en [7], explotando la capacidad de búsqueda del algoritmo de recocido simulado. Estos enfoques metaheurísticos se han aplicado a corpus etiquetados en inglés, Corpus Brown [8], Corpus PennTreebank [9] y otras lenguas no tradicionales como el árabe, corpus KALIMAT [10], el Bengalí (Bangladés) [11], el hindi (India) [12], el Telugu (India) [13] y el Nasa Yuwe (lengua indígena de Colombia) [14]. En general estas propuestas usan el conjunto de etiquetas Petrov [15].

Los algoritmos metaheurísticos solucionan problemas utilizando un proceso de búsqueda (exploración y explotación) de soluciones óptimas para un problema en particular [16], es así, que los algoritmos meméticos[17] utilizan la búsqueda basada en población para la exploración de soluciones y la búsqueda local basada en vecindad para la explotación sobre soluciones prometedoras [18, 19], es decir, adicionan conocimiento del problema a solucionar. A continuación, en la Tabla 1, se describen las metaheurísticas estudiadas en esta investigación para su posterior adaptación al problema de etiquetado:

Tabla 1. Algoritmos metaheurísticos estudiados para adaptar al problema de etiquetado.

Algoritmo Metaheurístico	Descripción
Restart Random Hill Climbing (RRHC) [20]	Metaheurística de estado simple que mejora al Hill Climbing (HC) [17]. Busca evitar que HC quede atrapado en óptimos locales, realiza exploraciones repetitivas en el espacio del problema, que son generadas de forma aleatoria hasta que ocurra el criterio de parada o no se encuentra una mejor solución.
Particle Swarm Optimization (PSO) [21]	Metaheurística poblacional motivada por el comportamiento colectivo inteligente de los enjambres en la naturaleza. Cada solución potencial se denomina partícula, el conjunto de partículas se llama enjambre, y la posición de cada partícula cambia en función de su propia experiencia y la experiencia del enjambre [22].
Jaya [23]	Metaheurística poblacional busca encontrar la mejor solución en el menor tiempo posible, también a su vez siempre está intentando alejarse del fracaso, generando un balance óptimo entre exploración y explotación. Jaya es novedoso, simple y eficiente para resolver problemas de optimización con y sin restricciones.
GBHS Tagger [4]	Algoritmo memético adaptado al problema de etiquetado [4], el cual tiene como base la metaheurística Global-Best Harmony Search (GBHS), la cual tiene como parámetros [17]: HMS (tamaño de la memoria armónica), NI (número de improvisaciones), HCMR (tasa de consideración de la memoria armónica) y PARMin, PARMax (tasa de ajuste de tono). GBHS Tagger incluye conocimiento del problema de etiquetado mediante un optimizador local, adaptado de la metaheurística Ascenso a la Colina (HC) [17], que se aplica a la mejor armonía en la memoria armónica (HM). Adicional a los parámetros de GBHS, se definen tres parámetros más: ProbOpt, que controla el proceso de optimización local, MaxNeighbors, define el número de vecinos que se usan en el proceso de optimización local y el parámetro Alpha controla si los componentes de cada armonía en la población son generados aleatoriamente de sus posibles etiquetas o tomados de la etiqueta con mayor probabilidad.

En la Figura 1, se muestra la representación de la solución utilizada para esta investigación, que consta de: 1) Un primer vector del tamaño del número de palabras en una oración (una posición por palabra), que contiene las etiquetas asignadas a cada palabra, desde la posición 0 a la posición n-1 ($\mathrm{T}_0, \mathrm{T}_1, \ldots, \mathrm{T}_{\mathrm{n}-1}$); 2) Un segundo vector que contiene la probabilidad acumulada de cada palabra etiquetada, y su relación con su predecesor y su sucesor, calculada como $P_i =$

$P(w_i|t_j)\ x\ P(t_i|t_{i-1}, t_{i+1})$; y 3) Un campo que almacena el valor de la función fitness, calculada como se muestra en la Figura 1, adaptada de [5]. En GBHS Tagger el contexto seleccionado para la palabra a etiquetar es un trigrama (predecesora, palabra a etiquetar, sucesora).

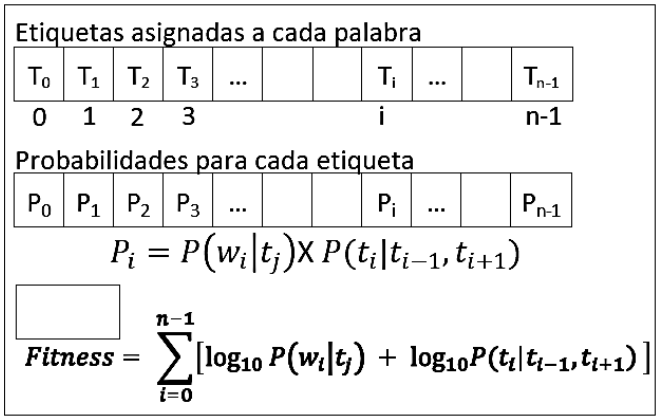

Fig. 1. Representación de la solución [4].

En el presente trabajo se realizó la adaptación de varios algoritmos metaheurísticos al problema de etiquetado, utilizando la representación de la solución propuesta en [4], con el fin de proponer mejoras al memético presentado en el mismo trabajo, al mismo tiempo que se buscó evaluar su desempeño sobre los corpus en castellano IULA [24], inglés Brown [8] y Nasa Yuwe [14].

El resto del articulo está organizado de la siguiente manera: La Sección 2, presenta la metodología utilizada; la Sección 3, detalla la adaptación de las metaheurísticas seleccionadas al problema de etiquetado; la Sección 4, muestra los resultados de los experimentos realizados, y finalmente, la Sección 5, presenta la Discusión, Conclusiones y el trabajo futuro.

II. METODOLOGÍA

Esta sección describe el conjunto de datos utilizado para la evaluación de los algoritmos, las actividades realizadas en cada fase de los ciclos de la metodología Patrón Iterativo de Investigación (Iterative Research Pattern, IRP) [25], usada para el desarrollo de este trabajo y la forma cómo se configuraron los experimentos.

A. Método utilizado

Para esta investigación se utilizaron dos ciclos. El primer ciclo, se enfocó en la adaptación de los algoritmos metaheurísticos al problema de etiquetado y la selección del mejor; el segundo ciclo, en la adaptación al problema de etiquetado de los algoritmos metaheurísticos seleccionados y la propuesta de una nueva versión del algoritmo memético. La Tabla 2, describe las actividades realizadas en cada fase.

Tabla 2. Descripción de la metodología.

<table>
<tr><th></th><th>Observación</th><th>Identificación</th><th>Desarrollo</th><th>Pruebas</th></tr>
<tr><td rowspan="3">Ciclo 1</td><td rowspan="3">Revisión de metaheurísticas e investigación de corpus y conjunto de etiquetas.</td><td>Selección del corpus.</td><td>Diseño e implementación de la base de datos</td><td>Ejecución de experimentos dataset completo.</td></tr>
<tr><td>Mapeo de etiquetas al etiquetado universal.</td><td>Adaptación e implementación de los algoritmos metaheurísticos al POST.</td><td>Ejecución de pruebas estadísticas no paramétricas.</td></tr>
<tr><td>Estudio de algoritmos metaheurísticos seleccionados.</td><td>Configuración y ejecución de experimentos sobre datasets pequeños.</td><td>Análisis y discusión de resultados</td></tr>
<tr><td rowspan="3">Ciclo 2</td><td>Investigación de los algoritmos meméticos e implementaciones discretas</td><td rowspan="2">Adaptación e implementación de los algoritmos PSO, Jaya, Restart Random Hill Climbing (RRHC) y HC al problema de etiquetado.</td><td>Implementación de las versiones finales de los etiquetadores propuestos</td><td>Ejecución de experimentos completos y pruebas</td></tr>
<tr><td rowspan="2">Estudio y comprensión del memético GBHS Tagger.</td><td>Integración de HC Tagger al algoritmo memético.</td><td rowspan="2">Análisis de resultados y emisión de conclusiones.</td></tr>
<tr><td>Configuración de experimentos sobre datasets pequeños y Ajuste de parámetros.</td><td>Ejecución de los experimentos y desarrollo de la nueva versión del memético.</td></tr>
</table>

B. Conjunto de Datos y Configuración de los experimentos

Como parte de este trabajo se realizó la integración de los corpus etiquetados IULA (Castellano), Brown (inglés) y Nasa Yuwe, en una sola base de datos, diseñada y desarrollada en SQL Server. Los experimentos fueron realizados sobre esta base de datos y para su ejecución (tanto preliminares como completos) se utilizó un modelo cliente – servidor, en donde los clientes (máquinas) solicitan las tareas a realizar, cada tarea recibe la frase y el algoritmo que debe ejecutar y evaluar. Cada tarea es ejecutada 30 veces (repeticiones del experimento) en la máquina local, una vez finalizada la tarea se graban los resultados en la base de datos en la nube.

III. RESULTADOS

En esta sección, en primera instancia, se presenta la adaptación de los algoritmos PSO Tagger, Jaya Tagger, RRHC Tagger adaptados al problema de etiquetado y una nueva versión del memético GBHS Tagger (GBHS4Tagger). En segunda instancia, se muestran de forma resumida los experimentos y los resultados obtenidos con los etiquetadores propuestos. Se resalta que todos los algoritmos adaptados usan la representación de la solución presentada en [4], descrita en la Figura 1.

A. PSOTagger propuesto

La adaptación propuesta, utilizó una versión discreta de PSO [26] y define los siguientes parámetros: W que selecciona una etiqueta al azar para cada dimensión de una partícula; $C1$ que selecciona la etiqueta del mejor histórico de la partícula, para esa palabra; $C2$ que selecciona la etiqueta del mejor global del enjambre, para cada dimensión de la partícula; P que mantiene los componentes de la partícula actual. Adicionalmente, PSOTagger involucra los parámetros $TamPoblacion$, $MaxGeneraciones$ de su versión original. La afinación de los parámetros $W, C1, C2$ y P en PSO se realizó de forma experimental con validación cruzada de 5 folders y un dataset pequeño a manera de muestra del dataset de evaluación. En la Figura 2, se presenta el pseudocódigo del PSOTagger propuesto.

B. JayaTagger propuesto

Se utilizó una versión discreta de Jaya denominada DJaya, propuesta por [27] que es libre de parámetros. La adaptación consistió en avanzar hacia la solución más conocida y alejarse de la peor solución, se varió el manejo del parámetro de la peor solución. El algoritmo JayaTagger solo maneja tres parámetros $TamPoblacion$, $\mathrm{MaxGeneraciones}$ y $P4$, éste último controla que la nueva solución seleccione una etiqueta de la peor solución (X_W); haciendo que el algoritmo sea sencillo de implementar y evaluar. En la Figura 3, se presenta el pseudocódigo del JayaTagger propuesto.

PSOTagger

1. Definición de parámetros: $TamPoblacion$, MaxGeneraciones, W, C1, C2, P
2. Inicialización aleatoria de la $Población$
3. Definición de la mejor partícula global (G_{best})
4. Definición de P_{best} para cada partícula en el enjambre
5.
6. **para** $i = 1$ to MaxGeneraciones **hacer**
7. **para** $j = 1$ *to* $TamPoblacion$ **hacer** /* para cada partícula */
8. Evaluar la función fitness de la partícula (P_j) as show in Figure 1
9. **si** (fitness (P_j) > fitness (P_{best}))
10. $P_{best} \leftarrow P_j$
11. **fin_si**
12. **fin_para**
13. Definir $G'_{best} \leftarrow Poblacion^{best}$ /* se define el mejor de la población actual*/
14. **si** (fitness (G'_{best}) > fitness (G_{best})) /* mejor actual es mayor a mejor anterior*/
15. $G_{best} \leftarrow G'_{best}$
16. **fin_si**
17. **para** $j = 1$ to TamPoblacion **hacer** /* para cada partícula */
18. **para** $k = 1$ *to* n **hacer** /* para cada palabra en la oración */
19. **si** $(Activo[j] = true)$ **entonces** /*¿la palabra tiene más de una etiqueta? */
20. $aleatorio \leftarrow r$ /* número aleatorio entre 0 y 1*/
21. **si** $(aleatorio < W)$ **entonces**
22. $P_k \leftarrow LB_k + r \times (UB_k - LB_k)$ /*selección de una etiqueta al azar */
23. **fin_si**
24. **sino** $(aleatorio < W + C1)$ **entonces**
25. $P_k \leftarrow P_{best(k)}$ /* se selecciona la etiqueta del mejor histórico */
26. **fin_si**
27. **sino** $(aleatorio < W + C1 + C2)$ **entonces**
28. $P_k \leftarrow G_{best(k)}$ /* se selecciona la etiqueta del mejor Global */
29. **fin_si**
30. **sino**
31. $P_k \leftarrow P_k$ /* se mantiene la etiqueta */
32. **fin_si**
33. **fin_si**
34. **fin_para**
35. **fin_para**
36. **fin_para**
37. **retornar** G_{best}

Fig. 2. Pseudocódigo de PSOTagger.

JayaTagger

1. Definición de parámetros: $TamPoblacion$, MaxGeneraciones, $P4$
2. Inicialización aleatoria de la $Población$
3. Definición de la mejor solución en la población (X_B)
4. Definición de la peor solución en la población (X_W)
5. **para** $i = 1$ to MaxGeneraciones **hacer**
6. **para** $j = 1$ to $TamPoblacion$ **hacer** /* para cada partícula */
7. X_{new} /* generar una nueva solución*/
8. **para** $h = 1$ to n **hacer** /* para cada palabra en la oración */
9. **si** $(Activo[h] = true)$ **entonces** /* ¿la palabra actual tiene más de una posible etiqueta? */
10. $aleatorio \leftarrow r(1, P4)$
11. **según** $(aleatorio)$
12. **caso 1**: /*preserve la etiqueta de la solución*/
13. $X_{new(h)} = P_j(h)$
14. **caso 2**: /*Seleccione la etiqueta de la mejor solución*/
15. $X_{new(h)} = X_B(h)$
16. **caso 3**: /*Seleccione la etiqueta aleatoriamente*/
17. $X_{new(h)} = LB_h + r \times (UB_h - LB_h)$
18. **caso 4**: /*Seleccione la etiqueta de la peor solución*/
19. $X_{new(h)} = X_W(h)$
20. **fin_si**
21. **fin_para**
22. Evaluar la función fitness de la nueva solución (X_{new}) (Figura 1)
23. **si** (fitness (X_{new}) > fitness (P_j))
24. $P_j \leftarrow X_{new}$
25. **fin_si**
26. **fin_para**
27. **fin_para**
28. **retornar** X_B /*Retorna solución de la población*/

Fig. 3. Pseudocódigo de JayaTagger

C. Restart Random Hill Climbing (RRHC) Tagger propuesto

La adaptación del algoritmo RRHCTagger al problema de etiquetado se realizó así: 1) Los parámetros: $n\_restart$, controla el número de reinicios de la solución S; $Prob$, lista que almacena las probabilidades de las posibles etiquetas de una palabra; $ActiveIndex$, lista que almacena las posiciones de las palabras que tienen más de una etiqueta; $EstadoTrigrama$, lista que almacena las palabras seleccionadas para hacer una mejora estocástica. 2) Se mejora estocásticamente la solución, luego de un número determinado de iteraciones sin obtener mejoras, el algoritmo guarda el resultado actual y se reinicia nuevamente la solución (parámetro n_restart), seleccionando otra palabra de todas las posibilidades. 3) Se implementó una memoria tabú que guarda las palabras que fueron seleccionadas en el reinicio de la solución. En la Figura 4, se presenta el pseudocódigo del RRHCTagger propuesto.

```
Algoritmo RRHCTagger
1.  Definición del parámetro S, iterations_HC, i_restart y n_restart
2.  i_restart = 0
3.  Prob /* Lista de las probabilidades de una palabra de acuerdo a todas las opciones */
4.  Inicialización aleatoria de S
5.  ActiveIndex /* Esta lista guarda las posiciones de palabras que tienen más de una etiqueta*/
6.  EstadoTrigama /* Esta lista guarda la posición de las palabras que fueron seleccionadas*/
7.  j = Aleatorio(ActiveIndex.lenght)        /* j va a ser una posición al azar de activeIndex*/
8.  Prob = posiblesProbabilidades(j)  /*obtiene las posibles probabilidades de j*/
9.  X_new ← Copiar(S)
10. para i = 1 to iterations_HC hacer
11.         si (i_restart > n_restart ) /*Si i_restart supera a n_restart */
12.             EstadoTrigama.add (j)
13.             si (fitness(X_new) > fitness(S))
14.                 S ← X_new
15.                 si (EstadoTrigrama.contains(j − 1)) /*¿Se encuentra vecino de j? */
16.                     EstadoTrigrama.remove(j − 1)/*Se elimina de la lista */
17.                 fin_si
18.                 si (EstadoTrigrama.contains(j + 1)) /* ¿Se encuentra vecino sucesor? */
19.                     EstadoTrigrama.remove(j + 1) /* Se elimina de la lista */
20.                 fin_si
21.                 X_new ← Copiar(S) /*Copia para nueva búsqueda estocástica*/
22.                 pos ← Aleatorio(ActiveIndex.lenght /*Nueva posición→activeIndex*/
23.                 j = ActiveIndex(pos)/*Nueva palabra no debe estar en
                                          EstadoTrigrama*/
24.                 Prob = posiblesProbabilidades (j) /*Probabilidades de la nueva palabra*/
25.                 i_restart = 0
26.             fin_si
27.             Prob.ordenar() /*Se ordena de forma descendente, con la mejor probabilidad */
28.             X2_new ← Copiar(X_new) /*Copia para las modificaciones que se van a realizar*/
29.             pos2 =Prob [0]
30.             X2_new(j) = pos2 /*Se cambia la etiqueta en la posición j*/
31.             si (fitness(X2_new) > fitness(X_new) >))
32.                 X_new ← X2_new
33.                 Prob.remove(Prob [0]) /*Se elimina etiqueta, para no utilizarla más*/
34.             else
35.                 Prob.remove(Prob [0])
36.                 i_restart + +
37.             fin_si
38. fin_para
39. retornar S
```

Fig. 4. Pseudocódigo de RRHCTagger.

D. GBHS4Tagger propuesto

El algoritmo GBHS4 Tagger está basado en el algoritmo memético GBHS Tagger propuesto en [4] y su mejora consta de: 1) Se adaptó el algoritmo Hill Climbing (HC) al problema de etiquetado, involucrando dos vecindarios, el primero selecciona una palabra al azar sin importar la condición, y el segundo, selecciona la palabra con la probabilidad más baja, estos vecindarios se controlan con el parámetro Prob. 2) El HCTagger propuesto, se incorporó al GBHS Tagger 2 [4] como optimizador local y así se consolidó la nueva versión memética denominada GBHS4Tagger. En la Figura 5, se muestra el pseudocódigo de HCTagger propuesto y en la Figura 6, el GBHS4Tagger propuesto.

Algoritmo HCTagger

1. Definición del parámetro $Prob$(Probabilidad para seleccionar la estrategia de selección de la palabra a etiquetar)
2. Definición del parámetro N (Número máximo de evaluaciones de la función objetivo)
3. Inicialización aleatoria de la $Solucion$
4. ListaPosibilidadesEtiquetas /* Esta lista contiene todas las posibilidades de las palabras a etiquetar*/
5. **para** $i = 1$ to N **hacer**
6. $aleatorio \leftarrow r(0,1)$ /* Se genera un numero aleatorio entre 0 y 1*/
7. $X_{new} \leftarrow$ Copiar($Solucion$)
8. **si** ($aleatorio \leq Prob$) **entonces** /*Si el número aleatorio es menor que $Prob$ definido en la configuración del experimento*/
9. $pos \leftarrow$ r(etiquetas) /* Se selecciona la posición de la palabra al azar */
10. **sino**
11. $pos \leftarrow$ indice de la palabra con la probabilidad mas baja /* Se selecciona la posición de la palabra basado en la probabilidad más baja
12. **fin_si**
13. $j \leftarrow LB_{pos} + r \times (UB_{pos} - LB_{pos})$ /*Se selecciona un índice al azar de las posibilidades de la palabra selecciona (pos)
14. $X_{new(pos)}$ = ListaPosibilidadesEtiquetas[pos][j] /*Se asigna la etiqueta a la nueva solución */
15. **si** (fitness (X_{new}) > fitness ($Solucion$)) **entonces**
16. $Solucion \leftarrow X_{new}$ /* Si la nueva solución supera a la actual se reemplaza */
17. **sino**
18. ListaPosibilidadesEtiquetas(pos). eliminar(j) /*sino se elimina la posibilidad de la lista*/
19. **fin_si**
20. **fin_para**
21. **retornar** $Solucion$

Fig. 5. Pseudocódigo de HCTagger.

E. Experimentos realizados con los etiquetadores propuestos

Para la realización de los experimentos, en primera instancia, se realizó un ajuste (afinamiento) de los parámetros de los etiquetadores utilizando un dataset pequeño (muestra) de 5000 frases con el fin de seleccionar las mejores combinaciones de parámetros de cada algoritmo, en la Tabla 3, se muestra la distribución de las oraciones en los conjuntos de datos (data sets) de prueba y entrenamiento para cada folder, con los cuales se realizaron los experimentos sobre cada corpus completo, como se puede apreciar en la tabla 5. Todos los experimentos se ejecutaron utilizando validación cruzada de 5 folders, excepto para el corpus Nasa Yuwe, con el cual se utilizó Leave-One-Out, dado que el dataset solo tiene 175 frases.

GBHS adaptado al problema de etiquetado

1. Definir parámetros: HMS, NI, HCMR, PARMin, ParMax, ProbOpt, Alpha, MaxNeighbors, Prob
2. Inicialización aleatoria de HM o inicialización mejorada usando el parámetro Alpha
3. **para** $i = 1$ to NI **hacer**
4. $PAR \leftarrow PARMin + (PARMax - PARMin) \times \left({}^{i}/_{NI}\right)$ /* definición de PAR */
5. **para** $j = 1$ to n **hacer** /* para cada palabra en la oración */
6. **si** (Activo[j] == true) **entonces** /* ¿la palabra actual tiene más de una posible etiqueta? */
7. **si** $(U(0,1) \leq HMCR)$ **entonces** /* memory consideration */
8. $x_j' \leftarrow x_j^p$, donde $p \sim U(1, \ldots, HMS)$
9. **si** $(U(0,1) \leq PAR$ **entonces**
10. $x_j' \leftarrow x_k^{best}$, donde best es el índice de la mejor armonía en HM and $k \sim U(1,n)$
11. **fin_si**
12. **sino** /* selección aleatoria */
13. $x_j' \leftarrow LB_j + r \times (UB_j - LB_j)$
14. **fin_si**
15. **sino**
16. $x_j' \leftarrow$ UnicaEtiquetaParaLaPalabra(j)
17. **fin_si**
18. **fin_para**
19. **mientras** (visitado (x_j')) **hacer** /* Si la solución ha sido visitada antes */
20. **si** (Activo[j] == true) **entonces** /* mutar la nueva armonía (x_j')
21. Se selecciona aleatoriamente una diferente a la actual
22. **fin_mientras**
23. Evaluar la función fitness de la nueva armonía (x_j')
24. **si** (fitness (x_j') > fitness de la peor armonía en HS) **entonces**
25. HM [pos_peor] $\leftarrow x_j'$ /* reemplazar la peor en la memoria armónica */
26. **fin_si**
27. **si** $(U(0,1) < ProbOpt)$ **entonces**
28. Aplicar Optimización Local a la mejor armonía en HM
29. **fin_si**
30. **fin_para**
31. **retornar** la mejor armonía en HM

Fig. 6. Pseudocódigo de GBHS4Tagger. Adaptado de [4].

Tabla 3. Conjunto de datos de prueba y entrenamiento para los experimentos.

Data sets		**Corpus IULA**			**Corpus Brown**		
Datos Prueba	**Folder con datos de Entrenamiento**	**Oraciones en los datos de Prueba**	**Palabras en datos de Prueba**	**Palabras en los datos de Entrenamiento**	**Oraciones en los datos de Prueba**	**Palabras en datos de Prueba**	**Palabras en los datos de Entrenamiento**
1	2, 3, 4, 5	8416	16316	65521	10595	23105	45113
2	1, 3, 4, 5	8416	16357	65480	10600	22852	45199
3	1, 2, 4, 5	8416	16466	65371	10600	23130	45009
4	1, 2, 3, 5	8416	16290	65547	10600	22929	45130
5	1, 2, 3, 4	8415	16408	65429	10603	23111	45025
Corpus Nasa Yuwe		Naya Yuwe uso Leave-One-Out un método que toma como dato de prueba una frase y como datos de entrenamiento las 174 frases restantes. Ese proceso se repite para todas las frases del corpus. Es un método de evaluación comúnmente usado cuando el conjunto de datos es pequeño.					

A continuación, en la Tabla 4, se presenta la configuración de los algoritmos para los experimentos realizados con cada corpus.

Tabla 4. Configuración de los algoritmos etiquetadores para los experimentos.

Algoritmo	**Parámetros definidos para el corpus IULA**	**Parámetros definidos para el corpus Brown**	**Parámetros definidos para el corpus Nasa Yuwe**
PSOTagger	W = 0.3, C1 = 0.15, C2 = 0.45 y P = 0.1	W = 0.3, C1 = 0.15, C2 = 0.45 y P = 0.1	W = 0.3, C1 = 0.2, C2 = 0.4 y P = 0.1.
JayaTagger	P4 = 4	P4 = 3	P4 = 4.
RRHCTagger	n_restart = 6	n_restart = 4.	n_restart = 4.
GBHS4Tagger	Prob = 0.7	Prob = 0.7.	Prob = 0.75.

En la Tabla 5, se presentan los resultados de los experimentos realizados sobre los tres corpus completos. Se puede observar que GBHS4Tagger superó a los demás algoritmos en valor de precisión, en los corpus IULA y Brown, con el corpus Nasa Yuwe fue el segundo mejor. Cabe resaltar que, los algoritmos adaptados obtuvieron muy buenos resultados para este problema, pero existen diferencias entre los valores de precisión obtenidos en cada algoritmo, lo cual permite apreciar que algunos algoritmos se desempeñan mejor que otros, tal y como se establece en el segundo teorema de No Free Lunch Theorems for Optimization (NFLT) [28].

Tabla 5. Resultados de los experimentos.

Resultados	**Corpus IULA**			**Corpus BROWN**			**Corpus Nasa Yuwe**		
Algoritmo	**Frases**	**Precisión**	**Desv est**	**Frases**	**Precisión**	**Desv. estd**	**Frases**	**Precisión**	**Desv estd**
GBHS4Tagger	**42079**	**97.5403**	**5.4795**	**52998**	**94.8803**	**6.0289**	175	61.4185	15.5199
RRHCTagger	42079	97.2678	5.5006	52998	94.5342	6.0106	**175**	**63.7096**	**16.2249**
GBHS Tagger2	42079	97.4306	5.7844	52998	94.8236	6.2051	175	60.0749	15.5199
JayaTagger	42079	96.8592	5.5963	52998	93.0519	6.3054	175	57.084	15.977
PSOTagger	42079	96.7738	5.7844	52998	89.6933	7.0810	175	55.936	16.052

La Tabla 6, muestra el ranking de cada algoritmo en los experimentos realizados en cada corpus, una vez aplicada la prueba estadística no paramétrica de Friedman NxN, obteniendo un valor de p menor que 0.05, por tanto, hace que el ranking sea estadísticamente significativo, complementando la evaluación de los algoritmos.

Tabla 6. Resultados Test de Friedman – Corpus IULA. Elaborado con software KEEL [29]

Corpus IULA	
Algoritmo	**Ranking**
GBHS4Tagger	1
GBHS Tagger2	2

Corpus Brown	
Algoritmo	**Ranking**
GBHS4Tagger	1
GBHS Tagger2	2

Nasa Yuwe	
Algoritmo	**Ranking**
RRHCTagger	1
GBHS4Tagger 2	2

Corpus IULA	
Algoritmo	**Ranking**
RRHCTagger	3
JayaTagger	4
PSOTagger	5

Corpus Brown	
Algoritmo	**Ranking**
RRHCTagger	3
JayaTagger	4
PSOTagger	5

Nasa Yuwe	
Algoritmo	**Ranking**
GBHS Taggger2	3
JayaTagger	4
PSOTagger	5

Adicionalmente, la prueba de Wilcoxon mostró con un nivel de significancia del 90% que los resultados obtenidos para los algoritmos ganadores, GBHS4Tagger para Castellano e inglés y RRHCTagger para Nasa Yuwe, son mejores en contraste con los otros etiquetadores propuestos.

V. Discusión y conclusiones

Se logró la adaptación de los algoritmos metaheurísticos PSO, Jaya, RRHC al problema de etiquetado de partes del discurso (POST), teniendo en cuenta las características propias de cada algoritmo y realizando el ajuste de parámetros requerido para cada algoritmo sobre cada corpus, obteniendo resultados competitivos con respecto a uno de los algoritmos del estado del arte. También se logró proponer una mejora al algoritmo memético GBHS Tagger 2 del estado del arte, el cual continúo demostrando que el desempeño del etiquetador mejora al incluir conocimiento del problema, como se pudo apreciar en los corpus IULA (castellano) y Brown (inglés). En consecuencia, la investigación reforzó la idea de que los enfoques metaheurísticos obtienen buenos resultados, con recursos y tiempos aceptables. Continuar utilizando algoritmos metaheurísticos para el etiquetado sobre otras lenguas tradicionales y no tradicionales y buscar nuevas mejoras para los etiquetadores propuestos en combinación con otras técnicas de optimización, que mejoren los resultados del etiquetado.

Contribución de los autores

Miguel-Alexis Solano-Jiménez: Análisis Formal, Curación de datos, Investigación, Metodología, Software, Validación, Visualización, Escritura – borrador original, Escritura – revisión y edición.

Jose-Julio Tobar-Cifuentes: Análisis Formal, Curación de datos, Investigación, Metodología, Software, Validación, Escritura – borrador original, Escritura – revisión y edición.

Luz-Marina Sierra-Martínez: Conceptualización, Metodología, Supervisión, Administración del proyecto, Escritura – borrador original, Escritura – revisión y edición.

Carlos-Alberto Cobos-Lozada: Conceptualización, Metodología, Supervisión, Administración del proyecto, Escritura – borrador original, Escritura – revisión y edición.

FINANCIAMIENTO

El trabajo fue parcialmente financiado por la Universidad del Cauca.

AGRADECIMIENTOS

Los autores expresan especial gratitud a Colin McLachlan por las sugerencias relacionadas con el texto en inglés.

REFERENCIAS

[1] T. Güngör, *Handbook of Natural Language Processing (2 Edition)*, 2011.

[2] D. Jurafsky, and J. H. Martin, “Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition,” *Computational Linguistics*, vol. 26(4), pp. 638-641, 2009. https://doi.org/10.1162/089120100750105975

[3] A. Alhasan, and A. T. Al-taani, “POS Tagging for Arabic Text Using Bee Colony Algorithm,” *Procedia Computer Science*, pp. 158-165, 2018. https://doi.org/10.1016/j.procs.2018.10.471

[4] L. M. Sierra Martínez, C. A. Cobos, and J. C. Corrales, “Memetic algorithm based on global-best harmony search and hill climbing for part of speech tagging,” in *International Conference on Mining Intelligence and Knowledge Exploration*, 2017, pp. 198-211. https://doi.org/10.1007/978-3-319-71928-3_20

[5] R. Forsati, and M. Shamsfard, “Novel harmony search-based algorithms for part-of-speech tagging,” *Knowledge and Information Systems*, vol. 42, pp. 709-736, 2014. https://doi.org/10.1007/s10115-013-0719-6

[6] A. Ekbal, and S. Saha, “Simulated annealing based classifier ensemble techniques: Application to part of speech tagging,” *Information Fusion*, vol. 14 (3), pp. 288-300, 2013. https://doi.org/10.1016/j.inffus.2012.06.002

[7] S. Bandyopadhyay, S. Saha, U. Maulik, and K. Deb, “A Simulated Annealing-Based Multiobjective Optimization Algorithm: AMOSA,” *IEEE Transactions on Evolutionary Computation*, vol. 12 (3), pp. 269-283, Jun. 2008. https://doi.org/10.1109/TEVC.2007.900837

[8] W. N. Francis, and H. Kucera, *Brown Corpus Manual*, 1979. http://clu.uni.no/icame/manuals/BROWN/INDEX.HTM#bc8

[9] M. P. Marcus, M. A. Marcinkiewicz, and B. Santorini, "Building a large annotated corpus of English: the penn treebank," *Computational Linguistics*, vol. 19 (2), pp. 313-330, 1993. https://doi.org/10.1162/coli.2010.36.1.36100

[10] M. El-Haj, and R. Koulali, "KALIMAT a multipurpose Arabic Corpus," in *Second Workshop on Arabic Corpus Linguistics*, 2013.

[11] T. Chakraborty, "Identification of Reduplication in Bengali Corpus and their Semantic Analysis : A Rule-Based Approach," in *Proceedings of the Workshop on Multiword Expressions: from Theory to Applications*, 2010, pp. 73-76.

[12] O. Bojar, V. Diatka, P. Rychly, P. Straňak, V. Suchomel, Al. Tamchyna, and D. Zeman, "HindEnCorp - Hindi-English and Hindi-only corpus for machine translation," in *Proceedings of the Ninth International Conference on Language Resources and Evaluation*, 2014, pp. 3550-3555.

[13] S. S. Mukku, and R. Mamidi, "ACTSA: Annotated Corpus for Telugu Sentiment Analysis," in *Proceedings of the First Workshop on Building Linguistically Generalizable NLP Systems*, 2018, pp. 54-58. https://doi.org/10.18653/v1/w17-5408

[14] L. M. Sierra Martínez, C. A. Cobos, C. J. Muñoz Corrales, T. Curieux Rojas, E. Herrera-viedma, and D. H. Peluffo-ordóñez, "Building a Nasa Yuwe Language Corpus and Tagging with a Metaheuristic Approach," *Computación y Sistemas*, vol. 22 (3), pp. 881-894, 2018. https://doi.org/10.13053/CyS-22-3-3018

[15] S. Petrov, D. Das, and R. McDonald, "A Universal Part-of-Speech Tagset," in *Proceedings of the Eighth International Conference on Language Resources and Evaluation*, 2012, pp. 2089-2096.

[16] X. S. Yang, and S. Deb, "Cuckoo search: Recent advances and applications," *Neural Computing and Applications*, vol. 24 (1), pp. 169-174, 2014. https://doi.org/10.1007/s00521-013-1367-1

[17] J. Brownlee, *Clever Algorithms*, 2011.

[18] F. Neri, and C. Cotta, "A Primer on Memetic Algorithms," in *Handbook of Memetic Algorithm*, pp. 43-52, 2012.

[19] C. Cotta, *Una Visión General de los Algoritmos Meméticos*. http://www.lcc.uma.es/~ccottap/papers/memeticos.pdf

[20] E. R. R. Kato, G. D. de A. Aranha, and R. H. Tsunaki, "A new approach to solve the flexible job shop problem based on a hybrid particle swarm optimization and Random-Restart Hill Climbing," *Computers & Industrial Engineering*, vol. 125, pp. 178-189, Nov. 2018. https://doi.org/10.1016/j.cie.2018.08.022

[21] J. Kennedy, and R. Eberhart, "Particle Swarm Optimization," in *Proceedings of ICNN'95 - International Conference on Neural Networks*, 1995, pp. 1942-1948. https://doi.org/10.1109/ICNN.1995.488968

[22] A. Nickabadi, M. M. Ebadzadeh, and R. Safabakhsh, "A novel particle swarm optimization algorithm with adaptive inertia weight," *Applied Soft Computing*, vol. 11 (4), pp. 3658-3670, 2011. https://doi.org/10.1016/j.asoc.2011.01.037

[23] R. Venkata Rao, "Jaya: A simple and new optimization algorithm for solving constrained and unconstrained optimization problems," *International Journal of Industrial Engineering Computations*, vol. 7 (1), pp. 19-34, Dec. 2016. https://doi.org/10.5267/j.ijiec.2015.8.004

[24] Institut Universitari de Lingüística Aplicada, *IULA Spanish LSP Treebank*, 2012.

[25] K. S. Pratt, "Design Patterns for Research Methods: Iterative Field Research," in *AAAI Spring Symposium: Experimental Design for Real*, 2009, pp. 1-7.

[26] Q. Pan, M. F. Tasgetiren, and Y. Liang, “A discrete particle swarm optimization algorithm for the no-wait flowshop scheduling problem,” *Computers & Operations Research*, vol. 35, pp. 2807-2839, 2008. https://doi.org/10.1016/j.cor.2006.12.030

[27] K. Gao, F. Yang, M. Zhou, Q. Pan, and P. N. Suganthan, “Flexible job-shop rescheduling for new job insertion by using discrete Jaya algorithm,” *IEEE Transactions on Cybernetics*, vol. 49 (5), pp. 1944-1955, 2019. https://doi.org/10.1109/TCYB.2018.2817240

[28] D. H. Wolpert, and W. G. Macready, “No free lunch theorems for optimization,” *IEEE Transactions on Evolutionary Computation*, vol. 1 (1), pp. 67-82, Apr. 1997. https://doi.org/10.1109/4235.585893

[29] J. Alcalá-Fdez, L. Sánchez, S. García, M. J. del Jesus, S. Ventura, J. M. Garrell, J. Otero, C. Romero, J. Bacardit, V. M. Rivas, J. C. Fernández, and F. Herrera, “KEEL: A software tool to assess evolutionary algorithms for data mining problems,” *Soft Computing*, vol. 13, pp. 307-318, 2009. https://doi.org/10.1007/s00500-008-0323-y