Optimización de la carga de trabajo en computación en la nube usando el algoritmo Grey Wolf Optimizer en dataset Alibaba Cluster (2017 y 2018)

Juan Muñoz, Karen Rincón y Santiago Sotelo
Escuela de Ciencias Exactas e Ingeniería, Universidad Sergio Arboleda, Bogotá, Colombia
{*juan.munoz09, karen.rincon01, carlos.sotelo01*}*@usa.edu.co*

Keywords: Planificación de Tareas, Grey Wolf Optimizer, Computación en la nube, Optimización, Asignación de Recursos, Servidores, Dataset

Abstract: Efficient task allocation in cloud environments is crucial for optimizing resource usage and reducing operational costs. This study evaluates the performance of the Grey Wolf Optimizer (GWO) applied to task scheduling using real-world datasets from Alibaba Data Cluster (2017 and 2018). A robust preprocessing pipeline was designed to clean, scale, and normalize the data, ensuring compliance with strict resource utilization constraints. Experiments demonstrated significant improvements in resource distribution, with CPU and memory utilization maintained below 90 % on average. The GWO converged effectively across iterations, validating its suitability for cloud task scheduling. Results indicate potential for scalability and adaptability in real-world applications.

1. Introducción

La computación en la nube es una infraestructura esencial para los servicios digitales modernos, proporcionando recursos escalables para diversas cargas de trabajo computacionales. Sin embargo, la planificación de tareas sigue siendo un desafío debido a las restricciones de recursos, la alta variabilidad en las cargas de trabajo y la necesidad de optimizar tanto el desempeño como los costos. La optimización de recursos en infraestructuras de computación en la nube es crucial para maximizar el rendimiento y minimizar costos operativos. En los sistemas de computación en la nube, la asignación de tareas de procesamiento y el uso eficiente de los recursos de servidor son aspectos fundamentales que afectan directamente el rendimiento, la disponibilidad y los costos operativos. A medida que los servicios en la nube se expanden, la administración efectiva de los recursos se vuelve cada vez más compleja, pues requiere ajustarse dinámicamente a las variaciones en la demanda y las condiciones de los recursos. Cuando la carga de trabajo no se distribuye adecuadamente, los servidores pueden experimentar problemas como la sobrecarga de CPU, falta de memoria, saturación de red y, en casos críticos, fallos de hardware, los cuales comprometen la continuidad del servicio y aumentan los gastos operativos. [1]

Este proyecto aborda el desafío de asignación de recursos en clústeres de servidores, donde una gestión ineficiente puede llevar a saturación o subutilización de los mismos, afectando la calidad del servicio. Diversos enfoques han sido propuestos para abordar el problema de la asignación de tareas en entornos virtualizados.

Entre las estrategias más destacadas se encuentran:

- Métodos de asignación sencillos como Round-Robin y First-Come, First-Served, que distribuyen las tareas de manera equitativa o en orden de llegada. Sin embargo, estos métodos presentan limitaciones, ya que no consideran las características específicas de las tareas ni las variaciones en la capacidad de los servidores, lo que puede llevar a asignaciones subóptimas y sobrecarga en ciertos nodos. [2]
- Algoritmos Genéticos (GA): Utilizados para optimizar la asignación de datos multimedia en la nube, mostrando su eficacia en escenarios heterogéneos. Sin embargo, su desempeño se ve limitado por una convergencia más lenta en comparación con otros métodos [3].
- Optimización por Colonias de Hormigas (ACO): Esta técnica bioinspirada ha sido aplicada al problema del viajante y otras tareas de asignación. Aunque efectiva, su complejidad computacional puede ser prohibitiva en entornos dinámicos. [4]

- Optimización por Enjambre de Partículas (PSO): PSO se basa en la inteligencia colectiva de las partículas para encontrar soluciones óptimas. Sin embargo, la capacidad de exploración puede ser limitada en problemas de alta dimensionalidad. [5]

En este contexto, el **Grey Wolf Optimizer (GWO)** ha emergido como una solución innovadora para la asignación de recursos en la nube. Inspirado en la estructura jerárquica y el comportamiento de caza de los lobos grises, el GWO organiza las soluciones candidatas en una jerarquía compuesta por lobos alfa, beta, delta y omega. Este enfoque facilita un equilibrio efectivo entre exploración y explotación al dirigir la búsqueda hacia soluciones óptimas mediante un proceso iterativo de aproximación a la presa (la solución óptima) [6].

La implementación del GWO en este trabajo se adaptó para abordar los desafíos específicos de la asignación de recursos en la nube. En este caso, se diseñó una función de fitness que evalúa el consumo de CPU y memoria, penalizando fuertemente las asignaciones que exceden límites críticos (90 % de uso). Además, el algoritmo ajusta dinámicamente las posiciones de las soluciones candidatas para garantizar que las restricciones de recursos sean respetadas durante todo el proceso de optimización.

El GWO destaca como una solución adecuada para la distribución de carga en la nube debido a su capacidad de adaptarse a las fluctuaciones dinámicas en la demanda de recursos. Su diseño bioinspirado permite una exploración eficiente del espacio de soluciones, mientras que su capacidad para equilibrar la asignación de tareas minimiza la sobrecarga de servidores y maximiza la utilización eficiente de recursos. Estos atributos hacen del GWO una herramienta poderosa y flexible en entornos donde la redistribución constante de recursos es fundamental. [6]

2. Metodología

2.1. Descripción de los datos

Este estudio utiliza dos conjuntos de datos reales proporcionados por Alibaba Cluster Trace correspondientes a los años 2017 y 2018. Ambos datasets contienen información detallada sobre el uso de recursos y la programación de tareas en clústeres de servidores, pero presentan diferencias clave en su estructura y contenido.

Dataset de 2017:

Batch Task (batch_task.csv):

Incluye información de las tareas asignadas a los servidores. Columnas relevantes:

- job_id y task_id: Identificadores únicos para cada tarea y su trabajo asociado.
- instance_num: Número de instancias que requiere la tarea.
- status: Estado actual de la tarea (e.g., Running, Terminated, Failed).
- plan_cpu y plan_mem: Recursos de CPU y memoria solicitados por la tarea.
- start_time: Tiempo de inicio de la tarea
- end_time: Tiempo de finalización de la tarea

Machine Utilization (server_usage.csv):
Detalla el uso de recursos de los servidores durante la ejecución de las tareas. Columnas relevantes:

- util_CPU y util_memory: Porcentajes de uso de CPU y memoria.
- util_disk: Uso de disco por servidor.
- load_1, load_5, load_15: Promedios de carga del CPU en intervalos de tiempo específicos.

Dataset de 2018:

Batch Task (batch_task18.csv):
Columnas relevantes:

- task_name, job_name: Nombres únicos para tareas y trabajos.
- instance_num: Número de instancias solicitadas.
- task_type: Tipo de tarea, útil para clasificaciones adicionales.
- start_time y end_time: Marcas de tiempo para la planificación.
- plan_cpu y plan_mem: Requerimientos de CPU y memoria para las tareas.

Diferencias con dataset de 2017: Incluye tiempos y clasificaciones que permiten mayor granularidad en el análisis.

Machine Usage (server_usage18.csv):
Detalla el uso de recursos de los servidores en 2018. Columnas relevantes:

- machine_id: Identificador único de cada servidor.
- cpu_util_percent, mem_util_percent: Porcentajes de uso de CPU y memoria.

- net_in y net_out: Tráfico de red entrante y saliente normalizado.
- disk_io_percent: Porcentaje de uso de entrada/salida de disco.

Diferencias con 2017: Incluye métricas de red y tráfico que no estaban en el dataset anterior.

Diferencias Clave entre 2017 y 2018:

El dataset de 2018 incluye métricas adicionales como net_in, net_out, y disk_io_percent, lo que permite un análisis más detallado de las cargas de trabajo. El dataset de 2017 incluye métricas de carga (load1, load5, load15), mientras que en 2018 estas no están presentes. El tamaño de los datasets finales fue reducido para mayor manejabilidad, seleccionando tareas y servidores representativos. Ambos conjuntos de datos fueron sometidos a un riguroso proceso de limpieza y escalado para garantizar que los valores de consumo estuvieran normalizados entre 0 y 100. Esto aseguró la comparabilidad y mejoró la eficiencia del algoritmo Grey Wolf Optimizer (GWO) aplicado para la optimización de asignación de tareas.

2.2. Pipeline de preprocesamiento

Imputación de datos faltantes:

2017: No se detectaron valores faltantes significativos en las columnas principales (plan_cpu, plan_mem, util_CPU, util_memory). Sin embargo, en las columnas cpus_requested memory_requested se estimo el valor de las tareas que estaban en espera con la mediana.

2018: Para columnas como mem_gps y mkpi contenían más del 75 % de datos faltantes y fueron eliminadas por su falta de relevancia para el análisis. En batch_task, las columnas instance_num, plan_cpu, y plan_mem se imputaron mediante la mediana y utilizando relaciones específicas con otras características.

Eliminación de valores anómalos:

En el dataset de 2018, la columna disk_io_percent contenía valores anómalos (-1 o 101), que fueron eliminados para mantener la consistencia de los datos.

Normalización:

Todos los valores de consumo de CPU, memoria, y métricas relacionadas con el uso de servidores fueron escalados entre 0 y 100 para mantener un formato uniforme y garantizar que el algoritmo GWO no estuviera influenciado por escalas heterogéneas.

Filtrado de servidores y tareas:

Ambos datasets fueron filtrados para seleccionar tareas y servidores representativos con el objetivo de reducir el tamaño y enfoque del análisis. Se eligieron subconjuntos de 2500 tareas y 499 servidores para 2018 y 1500 tareas y 200 servidores para 2017.

Comparación Estadística entre los Datasets:

Distribuciones de Consumo: En el dataset de 2017, el consumo promedio de CPU y memoria era menor en comparación con 2018, indicando cargas de trabajo más equilibradas. Mientras que en 2018, las tareas mostraban mayor variabilidad en plan_cpu y plan_mem, lo que plantea un desafío adicional para la optimización.

Varianza de Recursos: Las métricas adicionales como tráfico de red (net_in y net_out) en 2018 permitieron analizar mejor los cuellos de botella en la conectividad, mientras que 2017 dependía más de promedios de carga (load1, load5, load15).

2.3. Etapas del pipeline de preprocesamiento

1. Carga de Datos: Carga y lectura de los datasets en sus versiones 2017 y 2018, que contienen información sobre tareas y utilización de máquinas en clústeres de servidores. Estos datos incluyen información como la utilización de CPU, memoria, e información relacionada con las tareas en ejecución. [7]

2. Limpieza de Datos: En esta etapa, se eliminan columnas irrelevantes y se corrigen valores extremos o inconsistentes. Para el dataset de 2018, por ejemplo, se eliminaron las columnas mem_gps y mkpi debido al alto porcentaje de valores faltantes. Métodos Utilizados: Eliminación de columnas con valores irrelevantes y conversión de formatos. [8]

3. Imputación de Faltantes: Para las columnas con valores faltantes significativos, como plan_cpu y plan_mem, se aplicaron técnicas de imputación. En el caso de valores categóricos, como instance_num, se utilizó la mediana. Para las columnas numéricas relacionadas con recursos, se aplicaron modelos basados en correlaciones con otras columnas del dataset. [9]

4. Escalado/Normalización: Se escalaron las columnas numéricas, como cpu_util_percent, mem_util_percent, y otras métricas de utilización de recursos, utilizando el método de normalización Min-Max. Esto asegura que todas las columnas estén en un rango uniforme, lo

que mejora la convergencia de los algoritmos de optimización. [10]

Figura 1: Diagrama de flujo pipeline de preprocesamiento

3. Resultados

3.1. Resultados dataset 2017

```
--- Solución Alfa ---
Solución Alfa:
Uso total de CPU: 82.21
Uso total de Memoria: 68.13
Penalización por violaciones de restricciones: 0

--- Solución Beta ---
Solución Beta:
Uso total de CPU: 90.04
Uso total de Memoria: 74.55
Penalización por violaciones de restricciones: 1

--- Solución Delta ---
Solución Delta:
Uso total de CPU: 91.29
Uso total de Memoria: 75.63
Penalización por violaciones de restricciones: 1
```

Figura 2: Uso promedio de CPU, memoria y restricciones en los lobos

Para el dataset de 2017, se evidencia que en algunas de las soluciones se llega a la restricción por el consumo en promedio de CPU y memoria, más específicamente en las soluciones Beta y Delta. Sin embargo, es posible ver que en las tres soluciones (Alfa, Beta y Delta) no se sobrepasa de forma significativa el límite superior de la restricción.

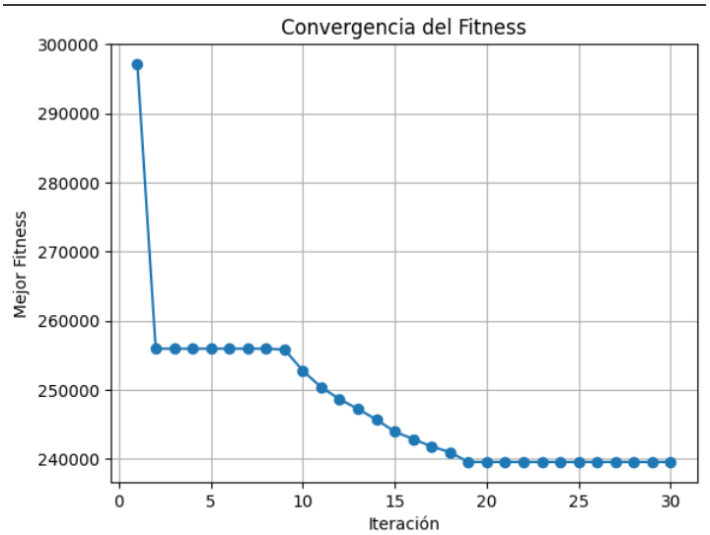

Figura 3: Convergencia del fitness

La convergencia del fitness en las iteraciones indica que el Grey Wolf Optimizer (GWO) logró estabilizarse a partir de la iteración 15. Este comportamiento demuestra la capacidad del GWO para explorar el espacio de búsqueda rápidamente en las primeras iteraciones y ajustar hacia la explotación en las siguientes (Figura 3). A partir de este punto se observa como se convierte en una transición mas lenta y que se acerca más al óptimo.

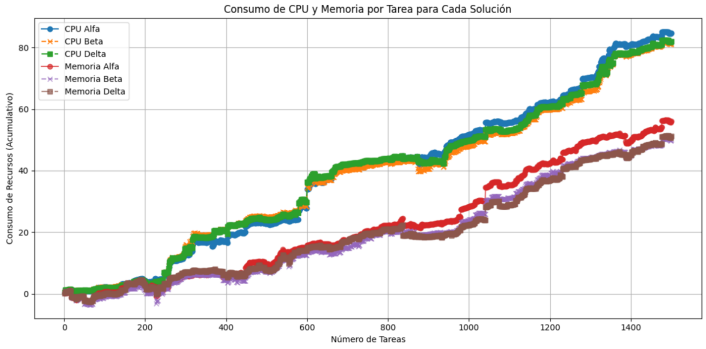

Figura 4: Consumos de CPU y Memoria por tarea para cada solución

En la Figura 4 se visualiza el consumo acumulativo de CPU y memoria por tarea para cada solución. Las tres soluciones tienen un crecimiento similar y controlado, manteniéndose dentro de los límites establecidos por la carga de trabajo. El consumo de CPU muestra una tendencia ligeramente superior para Beta y Delta, mientras que la memoria se comporta de manera uniforme.

3.2. Resultados dataset 2018

```
--- Solución Alfa ---
Solución Alfa:
Uso total de CPU: 59.31
Uso total de Memoria: 29.51
Penalización por violaciones de restricciones: 0

--- Solución Beta ---
Solución Beta:
Uso total de CPU: 65.75
Uso total de Memoria: 32.76
Penalización por violaciones de restricciones: 0

--- Solución Delta ---
Solución Delta:
Uso total de CPU: 63.67
Uso total de Memoria: 32.38
Penalización por violaciones de restricciones: 0
```

Figura 5: Uso promedio de CPU, memoria y restricciones en los lobos

Para el dataset de 2018, las soluciones mejoraron significativamente en términos de uso de CPU y memoria. Las tres soluciones (Alfa, Beta y Delta) muestran consumos por debajo del 70 % en promedio, con cero penalizaciones por restricciones. Estos resultados reflejan la capacidad del GWO para ajustarse a escenarios más complejos (Figura 5).

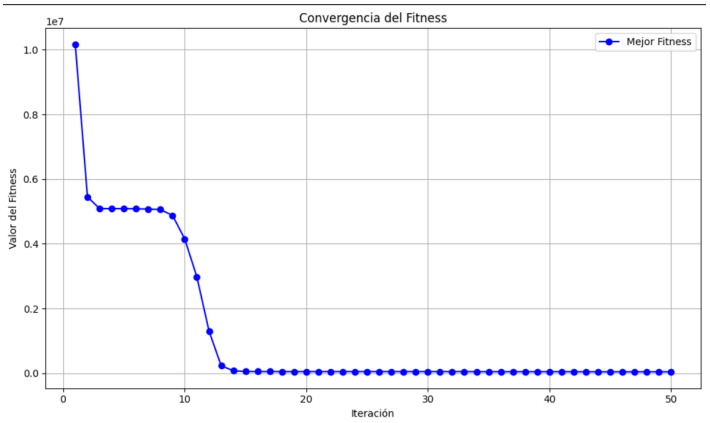

Figura 6: Convergencia del fitness

El comportamiento del fitness para el dataset 2018, representado en la Figura 5, demuestra una mejora en la convergencia, ya que desde la itreración 10 vemos cómo se converge rápidamente al óptimo. Sin embargo, la transición lenta se empieza a evidenciar desde la iteración 16, donde el fitness se estabiliza, indicando una asignación de recursos eficiente y libre de restricciones.

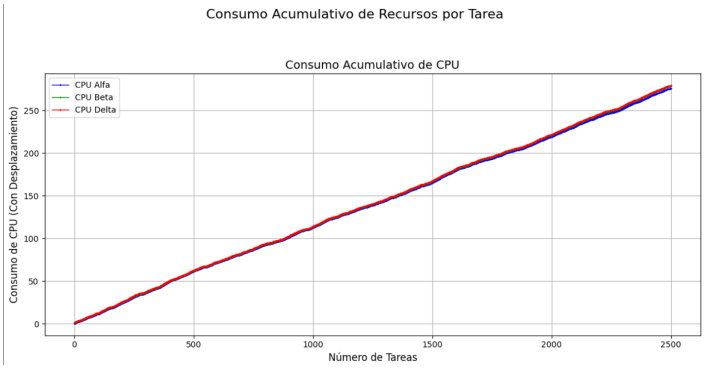

Figura 7: Consumos de CPU por tarea para cada solución

La Figura 6 muestra el consumo acumulativo de CPU por tarea, con Alfa, Beta y Delta alineándose estrechamente, destacando un rendimiento consistente entre las soluciones. La Figura 7, que corresponde al consumo acumulativo de memoria, refleja un comportamiento uniforme entre las soluciones con un uso eficiente de la memoria asignada, consolidando la adaptabilidad del algoritmo para esta versión del dataset.

Cabe resaltar que en esta nueva versión, se introducen penalizaciones más estrictas en la función de fitness, asegu-

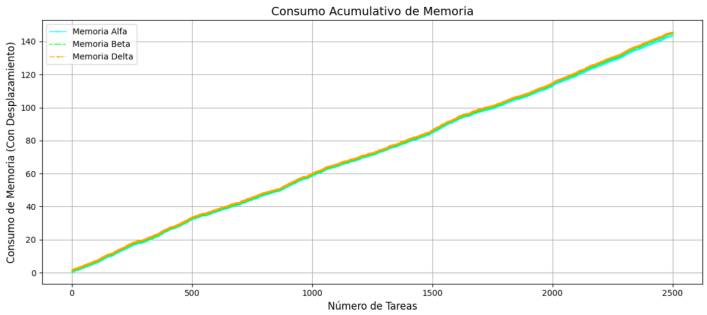

Figura 8: Consumos de Memoria por tarea para cada solución

rando que las soluciones que exceden el 90 % de uso de CPU o memoria sean significativamente penalizadas, mientras que un control explícito en update_positions ajusta dinámicamente las posiciones para evitar violaciones de restricciones. Además, se limita el rango de las posiciones iniciales y actualizadas a [0.2, 0.6], mejorando la distribución inicial de las tareas y la convergencia. También se recalculan dinámicamente los valores aleatorios en cada iteración, lo que equilibra la exploración y explotación del algoritmo. Se incluye un registro estructurado de la evolución del fitness (best_fitness_values) para analizar la convergencia y un cálculo explícito de las penalizaciones en la presentación de los resultados, ofreciendo mayor transparencia y control sobre las soluciones generadas. Estos cambios hacen el algoritmo más robusto, eficiente y adaptado a escenarios reales.

4. Conclusiones

El presente estudio demostró la eficacia del Grey Wolf Optimizer (GWO) como una solución viable y eficiente para el problema de planificación de tareas en entornos de computación en la nube. Al implementar el algoritmo sobre los conjuntos de datos reales de Alibaba Data Cluster 2017 y 2018, se observó que el GWO logró asignar recursos de manera óptima, manteniendo el uso de CPU y memoria por debajo del 90 % en promedio, incluso bajo restricciones estrictas de recursos. Este comportamiento refleja la capacidad del GWO para balancear adecuadamente la exploración y explotación del espacio de búsqueda, adaptándose dinámicamente a las condiciones de carga de trabajo y evitando situaciones de sobrecarga.

El pipeline de preprocesamiento diseñado fue fundamental para garantizar la calidad de los datos, abordando problemas comunes como valores faltantes, escalamiento inconsistente y normalización, lo que permitió al algoritmo operar con información confiable y normalizada. Las técnicas empleadas, como la imputación basada en relaciones entre variables y el escalamiento Min-Max, aseguraron que los datos estuvieran alineados con los requerimientos del modelo, lo cual fue clave para el éxito del proyecto.

Entre los resultados más destacados, se evidenció una mejora notable en la convergencia del algoritmo a lo largo de las iteraciones, con una reducción consistente del valor de fitness en cada etapa. Esta tendencia fue corroborada por los gráficos de convergencia generados, los cuales muestran cómo el GWO converge de manera estable hacia configuraciones de asignación de recursos óptimas. Además, los gráficos de consumo acumulativo y promedio de recursos por tarea proporcionaron información detallada sobre la eficiencia del modelo para distribuir la carga de trabajo, resaltando diferencias en las soluciones Alfa, Beta y Delta.

El análisis de los conjuntos de datos de 2017 y 2018 permitió identificar diferencias clave en la estructura y dimensionalidad de las tareas y recursos. A pesar de estas variaciones, el GWO se mostró altamente adaptable, validando su potencial para generalizarse en contextos reales y diversos. Este proyecto destaca por la validación del rendimiento del algoritmo bajo condiciones realistas, utilizando datos heterogéneos y altamente dinámicos.

Limitaciones y recomendaciones: Aunque el GWO mostró un desempeño satisfactorio, el análisis estuvo limitado por la cantidad de tareas y servidores procesados debido a restricciones computacionales. Para futuras investigaciones, se recomienda evaluar su escalabilidad en clústeres de mayor tamaño, así como comparar su desempeño con otros enfoques modernos como los híbridos entre técnicas heurísticas y aprendizaje automático. Además, se sugiere explorar la incorporación de técnicas de predicción de demanda para mejorar la asignación preventiva de recursos.

En conclusión, el GWO no solo es una herramienta prometedora para la planificación de tareas en la nube, sino que también ofrece flexibilidad y escalabilidad, lo que lo convierte en una opción atractiva para enfrentar los desafíos crecientes en la gestión de recursos computacionales en entornos reales.

5. Referencias

[1] K. Gai, M. Qiu, and H. Zhao, “Cost-aware multimedia data allocation for heterogeneous memory using genetic algorithm in cloud computing,” *IEEE Transactions on Cloud Computing*, vol. 4, no. 1, pp. 34–45, 2016.

[2] Alibaba, “Alibaba cluster trace 2017.” [Online], 2017. Available: https://github.com/alibaba/clusterdata.

[3] R. Buyya, R. Ranjan, and R. N. Calheiros, “Modeling and simulation of scalable cloud computing environments and the cloudsim toolkit: Challenges and opportunities,” in *Proceedings of the 2009 International Conference on High Performance Computing & Simulation*, pp. 1–11, 2009.

[4] M. Dorigo and L. M. Gambardella, “Ant colonies for the traveling salesman problem,” *Biosystems*, vol. 43, no. 2, pp. 73–81, 1997.

[5] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of ICNN'95 - International Conference on Neural Networks*, (Perth, WA, Australia), pp. 1942–1948, 1995.

[6] S. Mirjalili, "Grey wolf optimizer," *Advances in Engineering Software*, vol. 69, pp. 46–61, 2014.

[7] Y. Cao, Q. Ding, and B. Hu, "A multi-objective optimization approach based on grey wolf optimizer for task scheduling in cloud computing," *Applied Soft Computing*, vol. 93, 2020.

[8] S. Kotsiantis, D. Kanellopoulos, and P. Pintelas, "Data preprocessing for supervised learning," *International Journal of Computer Science*, vol. 1, no. 2, pp. 111–117, 2006.

[9] Z. Zhang, "Missing data imputation: Focusing on single imputation techniques," *Annals of Translational Medicine*, vol. 4, no. 1, pp. 9–14, 2012.

[10] A. Géron, *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*. Sebastopol, CA: O'Reilly Media, 2019.