



UNIVERSIDAD AUTÓNOMA METROPOLITANA

Unidad Azcapotzalco

División de Ciencias Básicas e Ingeniería

**Calibración de Hiper-Parámetros en Algoritmos
Metaheurísticos y Modelos de Lenguaje para la
Detección de Noticias Falsas**

Idónea Comunicación de Resultados

que presenta el:

Ing. Gabriel Hurtado Avilés

para obtener el grado de:

Maestro en Ciencias de la Computación

Directores:

Dr. José Alejandro Reyes Ortiz

Dr. Román Anselmo Mora Gutiérrez

Ciudad de México

Agosto 2025

Resumen

Este proyecto aborda el desafío de la detección de fraude digital y noticias falsas en español mediante la aplicación y comparación de dos metodologías de inteligencia artificial. La primera explora el uso de algoritmos metaheurísticos, incluyendo Recocido Multiarranque (MSA), Búsqueda Dispersa (SS), Búsqueda en Vecindades Variables (VNS), Algoritmo Genético (GA) y Optimización por Enjambre de Partículas (PSO), sobre una representación de Bolsa de Palabras (BoW). La segunda, adoptada tras los hallazgos iniciales, se basa en el ajuste fino (fine-tuning) de un modelo de lenguaje Transformer pre-entrenado (DistilBERT). Para el entrenamiento, se construyó un corpus unificando cuatro datasets públicos en español y datos extraídos mediante web scraping del portal satírico ^{EI}"Deforma", resultando en más de 61,000 noticias. Se implementó un cuidadoso proceso de calibración de hiperparámetros para ambos enfoques, utilizando una división de datos estratificada de 70 % para entrenamiento, 10 % para validación y 20 % para pruebas. El rendimiento fue evaluado con métricas como Exactitud, Precisión, Exhaustividad y F1-Score. Finalmente, el modelo Transformer, que demostró una eficacia superior, fue integrado en una aplicación web funcional desarrollada con Flask y contenerizada con Docker, capaz de analizar URLs en tiempo real. Los resultados validan la metodología de ajuste fino como una solución de vanguardia para combatir la desinformación, superando a los enfoques metaheurísticos en esta tarea.

Palabras clave: Detección de noticias falsas, fraude digital, modelos de lenguaje, Transformers, DistilBERT, algoritmos metaheurísticos, procesamiento de lenguaje natural.

Abstract

This project addresses the challenge of digital fraud and fake news detection in Spanish through the application and comparison of two artificial intelligence methodologies. The first explores the use of metaheuristic algorithms, including Multi-Start Simulated Annealing (MSA), Scatter Search (SS), Variable Neighborhood Search (VNS), Genetic Algorithm (GA), and Particle Swarm Optimization (PSO), applied to a Bag-of-Words (BoW) representation. The second, adopted following initial findings, is based on fine-tuning a pre-trained Transformer language model (DistilBERT). For training, a corpus was constructed by unifying four public Spanish datasets and data extracted through web scraping from the satirical portal "El Deiforma", resulting in over 61,000 news articles. A careful hyperparameter calibration process was implemented for both approaches, using a stratified data split of 70 % for training, 10 % for validation, and 20 % for testing. Performance was evaluated using metrics such as Accuracy, Precision, Recall, and F1-Score. Finally, the Transformer model, which demonstrated superior efficacy, was integrated into a functional web application developed with Flask and containerized with Docker, capable of analyzing URLs in real-time. The results validate the fine-tuning methodology as a state-of-the-art solution for combating misinformation, outperforming metaheuristic approaches in this task.

Keywords: Fake news detection, digital fraud, language models, Transformers, DistilBERT, metaheuristic algorithms, natural language processing.

Dedicatoria

// Esta sección se presenta en la versión final de su ICR. Son frases cuyo objetivo es otorgar una mención especial a las personas que te han motivado durante tu ICR.

Agradecimientos

// Esta sección se presenta en la versión final de su ICR. Son frases cuyo objetivo es plasmar el apoyo moral, físico, económico y/o emocional que recibió de las personas o instituciones durante la elaboración de todo su proyecto.

Índice general

Índice de figuras	XIII
Índice de tablas	XVI
1. Introducción	1
1.1. Planteamiento del problema	1
1.1.1. El Desafío Específico del Español	2
1.1.2. La Complejidad Técnica del Problema	3
1.2. Motivación	3
1.2.1. Impacto Personal y Social Observado	4
1.2.2. Vulnerabilidad de Poblaciones Específicas	4
1.2.3. Urgencia Tecnológica	5
1.3. Justificación	5
1.3.1. Brecha Tecnológica en Recursos para el Español	5
1.3.2. Novedad Metodológica: Enfoque Evolutivo Comparativo	6
1.3.3. Contribución Científica Multidimensional	6
1.3.4. Relevancia en el Contexto de LLMs	7
1.4. Objetivos	7
1.5. Alcance y Limitaciones	8
1.5.1. Alcance	8
1.5.2. Limitaciones	9
1.6. Contribuciones Esperadas	10
1.6.1. Contribuciones Teóricas	10
1.6.2. Contribuciones Prácticas	10
1.6.3. Contribuciones Sociales	10
1.7. Organización del Documento	11
2. Marco Teórico	13
2.1. Detección de Noticias Falsas y Fraude Digital	13
2.1.1. Impacto Social y Desafíos de la Desinformación	14
2.1.2. El Problema de las Heurísticas Cognitivas	14
2.2. Representación de Texto: Desde BoW hasta Embeddings Contextuales	14
2.2.1. Bolsa de Palabras (Bag-of-Words - BoW)	14

2.2.2. Ponderación TF-IDF (Term Frequency-Inverse Document Frequency)	15
2.2.3. Limitaciones de los Métodos Clásicos	15
2.3. La Revolución Transformer y los Modelos de Lenguaje Modernos	16
2.3.1. La Arquitectura Transformer: Fundamentos	16
2.3.2. BERT y la Era del Pre-entrenamiento Bidireccional	16
2.3.3. Aplicaciones en Detección de Noticias Falsas en Español	17
2.3.4. Grandes Modelos de Lenguaje (LLMs) y Nuevos Paradigmas	17
2.3.5. El Doble Rol de los LLMs en Detección	17
2.4. Optimización Metaheurística en Detección de Fraude	18
2.4.1. Fundamentos de las Metaheurísticas	18
2.4.2. Aplicaciones en Detección de Noticias Falsas	18
2.4.3. Metaheurísticas para Detección de Fraude Financiero	19
2.4.4. Algoritmos Híbridos Modernos	20
2.5. Integración de Enfoques: Hacia Sistemas Híbridos	20
2.5.1. Combinación de Representaciones Clásicas y Modernas	20
2.5.2. Arquitecturas de Ensemble Optimizadas	21
2.5.3. Optimización End-to-End	21
2.6. Desafíos y Direcciones Futuras	21
2.6.1. Desafíos Técnicos	21
2.6.2. Consideraciones Éticas y Sociales	21
2.7. Síntesis del Marco Teórico	22
3. Estado del arte	23
3.1. Metodología de Búsqueda	23
3.2. Clasificación Temática de la Literatura	23
3.2.1. Artículos sobre Metodología y Corpus de Datos	24
3.2.2. Artículos sobre Modelos de Lenguaje y Transformers	25
3.2.3. Artículos sobre Técnicas Metaheurísticas y Optimización	26
3.2.4. Artículos sobre Revisiones y Estudios Comprehensivos	27
3.2.5. Artículos sobre Detección de Fraude Financiero y Laboral	27
3.2.6. Artículos sobre Análisis de Comportamiento Social y Heurísticas	28
3.2.7. Artículos sobre Sistemas de Información y Representación de Conocimiento	28
3.3. Análisis Temático de la Literatura Relevante	29
3.3.1. El Desafío de los Datos: Creación de Corpus en Español	29
3.3.2. Revolución de los Modelos de Lenguaje y Transformers	29
3.3.3. Optimización y Metaheurísticas en la Detección	30
3.3.4. Perspectivas Interdisciplinarias y Análisis Social	30
3.3.5. Detección de Fraude: Más Allá de las Noticias	30
3.4. Investigación y Desarrollo de Métodos de Detección	31
3.5. Implementación de Soluciones Prácticas	32
3.6. Análisis del Impacto Social y Rol de los Medios	32

3.7. Detección Aplicada a Fraude Financiero y Laboral	33
3.8. La Detección como un Problema de Optimización	33
3.9. Algoritmos Metaheurísticos: Fundamentos y Aplicaciones	33
3.9.1. Algoritmos Genéticos (GA)	34
3.9.2. Optimización por Enjambre de Partículas (PSO)	34
3.9.3. Recocido Simulado y Métodos Multi-arranque	34
3.9.4. Búsqueda Dispersa (Scatter Search)	34
3.9.5. Búsqueda en Vecindades Variables (VNS)	35
3.9.6. Aplicaciones en Detección de Noticias Falsas	35
3.9.7. Aplicación de Metaheurísticas por Tipo de Detección	36
3.10. El Rol de los Modelos de Lenguaje	37
3.11. Hiperparámetros en Modelos de Aprendizaje Automático	38
3.11.1. ¿Qué son los Hiperparámetros?	38
3.11.2. ¿Por qué son Importantes en la Detección de Noticias Falsas?	39
3.11.3. El Problema: Encontrar la Mejor Configuración	39
3.11.4. Herramientas Modernas: Keras Tuner	39
3.11.5. La Ventaja de Combinar Modelos de Lenguaje con Keras Tuner	40
3.12. Síntesis y Perspectivas Futuras	41
4. Metodología	43
4.1. Visión General de la Metodología	43
4.2. Definición y Distinción de Conceptos Fundamentales	44
4.2.1. Noticia Falsa vs. Bulo: Una Distinción Crítica	44
4.2.2. Taxonomía de la Desinformación	44
4.2.3. Justificación de la Unificación Terminológica	45
4.3. Construcción del Corpus Unificado	46
4.3.1. Fuentes de Datos Académicas	46
4.3.2. Proceso de Unificación y Estandarización	47
4.3.3. Ampliación del Corpus Mediante Web Scraping	47
4.3.4. Corpus Final y Estrategia de División	49
4.4. Enfoque 1: Detección Mediante Algoritmos Metaheurísticos	51
4.4.1. Preprocesamiento y Representación Textual	51
4.4.2. Algoritmos Metaheurísticos Implementados	52
4.4.3. Función de Evaluación y Clasificación	55
4.4.4. Reducción de Dimensionalidad	55
4.5. Enfoque 2: Detección Mediante Modelo Transformer	56
4.5.1. Selección y Justificación del Modelo	56
4.5.2. Infraestructura Computacional	56
4.5.3. Configuración Experimental y Optimización de Hiperparámetros	56
4.5.4. Arquitectura del Modelo de Clasificación	58
4.5.5. Protocolo de Entrenamiento	58
4.5.6. Consideraciones de Implementación	59
4.6. Metodología de Evaluación Comparativa	59

4.6.1.	Protocolo de Evaluación	60
4.6.2.	Fundamentos de la Matriz de Confusión	60
4.6.3.	Marco de Métricas de Rendimiento	61
4.6.4.	Criterios de Selección del Mejor Modelo	61
4.6.5.	Protocolo de Validación Cruzada	61
4.6.6.	Benchmarking Computacional	62
4.6.7.	Análisis de Matrices de Confusión	62
4.6.8.	Reporte de Resultados	63
4.7.	Infraestructura Computacional y Herramientas	63
4.7.1.	Entorno de Desarrollo para Algoritmos Metaheurísticos	63
4.7.2.	Hardware Dedicado para Entrenamiento DistilBERT	63
4.7.3.	Stack Tecnológico Completo	65
4.8.	Consideraciones Éticas y de Privacidad	65
4.8.1.	Marco Ético de Desarrollo	65
4.8.2.	Limitaciones Declaradas y Uso Responsable	65
4.9.	Validación y Reproducibilidad	66
4.9.1.	Ecosistema de Artefactos de Reproducibilidad	66
4.9.2.	Solución Lista para Producción	66
5.	Análisis de Resultados	69
5.1.	Resultados del Enfoque Metaheurístico con Representación BoW-TF-IDF	69
5.1.1.	Marco Experimental y Configuración Base	70
5.1.2.	Implementación y Configuración de Algoritmos Metaheurísticos	71
5.1.3.	Pseudocódigos de los Algoritmos Implementados	72
5.1.4.	Visualizaciones de Resultados por Algoritmo	77
5.1.5.	Contextualización con Investigación Publicada	84
5.1.6.	Ánálisis Comparativo de Resultados	85
5.1.7.	Limitaciones Fundamentales y Justificación para Evolución . .	86
5.1.8.	Síntesis y Transición	87
5.2.	Resultados del Enfoque Transformer: DistilBERT Multilingüe	88
5.2.1.	Marco Experimental y Evolución del Desarrollo	88
5.2.2.	Configuración del Modelo DistilBERT Optimizado	89
5.2.3.	Proceso de Optimización y Búsqueda de Hiperparámetros . .	89
5.2.4.	Ánálisis de Convergencia y Control de Overfitting	90
5.2.5.	Evolución Experimental: Versiones de Desarrollo	92
5.2.6.	Pseudocódigo del Algoritmo de Entrenamiento DistilBERT . .	97
5.2.7.	Resultados Finales y Comparación	99
6.	Implementación de Prototipos Funcionales	101
6.1.	Introducción a la Fase de Implementación	101
6.2.	Arquitectura General del Sistema	101
6.3.	Prototipo 1: Analizador Basado en Metaheurísticas	102
6.3.1.	Componentes del Modelo	102

6.3.2. Flujo de Inferencia	103
6.4. Prototipo 2: Analizador Basado en Modelos Transformer (Versión Final)	103
6.4.1. Componentes del Modelo	103
6.4.2. Flujo de Inferencia	104
6.5. Interfaz de Usuario y Casos de Uso	104
6.5.1. Caso de Uso 1: Detección de una Noticia Real	104
6.5.2. Caso de Uso 2: Detección de una Página con Contenido Engañoso	104
6.5.3. Caso de Uso 3: Detección de una Página Fraudulenta	104
7. Conclusiones	107
7.1. Resumen del Trabajo y Contribuciones Principales	107
7.2. Limitaciones del Estudio	108
A. Anexo 1	111
Bibliografía	112

Índice de figuras

1.1. Mapa Conceptual 1: Taxonomía del problema de desinformación y fraude digital en la era de la IA.	2
1.2. Mapa Conceptual 2: Estrategias tecnológicas para la detección de fraude digital.	4
3.1. Mapa Conceptual 3: Artículos que son revisiones o están relacionados al análisis de contenido y detección de fraude financiero.	31
3.2. Mapa Conceptual 4: Clasificación de artículos por enfoque.	32
3.3. Mapa Conceptual 5: Métodos para resolver la problemática presentados en los artículos.	38
3.4. Mapa Conceptual 6: Artículos relacionados que incorporan Modelos de Lenguaje.	40
4.1. Metodología propuesta que aborda la problemática combinando Algoritmos Metaheurísticos y Modelos de Lenguaje.	44
5.1. Evolución de la convergencia del algoritmo MSA mostrando el progreso gradual a través de los 31 niveles de temperatura desde 1000 hasta 1.24.	77
5.2. Matriz de confusión para MSA en el conjunto de pruebas, evidenciando la baja especificidad (33 %) y el sesgo hacia la clasificación como noticias reales.	78
5.3. Convergencia eficiente del algoritmo SS en solo 10 iteraciones, mostrando mejoras progresivas en las iteraciones 4, 5 y 7 hasta estabilizarse en 0.6630.	79
5.4. Matriz de confusión para SS demostrando mejor balance que MSA con especificidad del 40 % y excelente generalización.	79
5.5. Evolución darwiniana del algoritmo GA a lo largo de 20 generaciones, evidenciando progreso sostenido desde 0.6198 hasta 0.7090 con hitos evolutivos significativos.	80
5.6. Matriz de confusión para GA mostrando el mejor balance global con especificidad líder del 48 % y rendimiento sólido en ambas clases. . . .	81
5.7. Progreso sistemático del algoritmo VNS a través de 20 iteraciones con cambios efectivos de vecindario, mostrando saltos significativos en las iteraciones 6 y 14.	82

5.8. Matriz de confusión para VNS destacando la excelente exhaustividad del 89 % para detección de noticias reales con especificidad competitiva del 41 %.	82
5.9. Convergencia problemática del algoritmo PSO evidenciando estancamiento prematuro en la iteración 7-8 y exploración insuficiente del espacio de búsqueda.	83
5.10. Matriz de confusión para PSO revelando el comportamiento extremo problemático con especificidad crítica del 15 % y sesgo severo hacia la clase mayoritaria.	84
5.11. Evolución de la exactitud y pérdida durante el entrenamiento del modelo DistilBERT V7. Las líneas azul y roja muestran la convergencia en entrenamiento y validación respectivamente. La estrella dorada marca la mejor época (13), después de la cual se observa el inicio del overfitting con una separación creciente entre las curvas.	91
5.12. Evolución de exactitud a través de las versiones experimentales.	96
6.1. Captura de pantalla de la aplicación analizando una noticia real.	105
6.2. Captura de pantalla de la aplicación detectando una noticia falsa basada en su contenido.	105
6.3. Captura de pantalla de la aplicación detectando una página de fraude digital.	106

Índice de tablas

3.1. Artículos sobre metodología y corpus de datos.	24
3.2. Artículos sobre modelos de lenguaje y transformers.	25
3.3. Artículos sobre técnicas metaheurísticas y optimización.	26
3.4. Artículos sobre revisiones y estudios comprehensivos.	27
3.5. Artículos sobre detección de fraude financiero y laboral.	27
3.6. Artículos sobre análisis de comportamiento social y heurísticas.	28
3.7. Artículos sobre sistemas de información y representación de conocimiento.	28
3.8. Artículos relacionados con la investigación y desarrollo de métodos de detección.	31
3.9. Artículos relacionados a la implementación de soluciones prácticas.	32
3.10. Algoritmos metaheurísticos: características y aplicaciones en detección de noticias falsas.	35
3.11. Aplicación de metaheurísticas por tipo de detección en la literatura revisada.	36
4.1. Comparación detallada entre noticia falsa y bulo en el contexto de detección automática.	45
4.2. Corpus académicos utilizados para la construcción del dataset unificado.	46
4.3. Fases del proceso de web scraping implementado para "El Deforma".	48
4.4. Referencias bibliográficas completas de los corpus académicos utilizados.	49
4.5. Composición final del corpus unificado después del procesamiento completo.	49
4.6. División estratificada principal del corpus para entrenamiento y evaluación.	50
4.7. Algoritmos metaheurísticos implementados y sus fundamentos conceptuales.	52
4.8. Configuración de parámetros del algoritmo MSA.	53
4.9. Configuración de parámetros del algoritmo SS.	53
4.10. Configuración de parámetros del algoritmo GA.	54
4.11. Configuración de parámetros del algoritmo VNS.	54
4.12. Configuración de parámetros del algoritmo PSO.	55
4.13. Configuración del proceso de reducción de dimensionalidad.	55

4.14. Comparación de modelos BERT optimizados para la tarea de clasificación.	56
4.15. Especificaciones del hardware utilizado para entrenamiento de DistilBERT.	57
4.16. Configuración de parámetros base para el entrenamiento de DistilBERT.	57
4.17. Estrategias de regularización implementadas para controlar overfitting.	57
4.18. Arquitectura del modelo de clasificación basado en DistilBERT.	58
4.19. Protocolo de evaluación implementado para ambos paradigmas.	60
4.20. Definición de categorías de la matriz de confusión en el contexto de detección de noticias falsas.	60
4.21. Estructura de la matriz de confusión para clasificación binaria.	60
4.22. Marco de métricas de evaluación para clasificación binaria de noticias falsas.	61
4.23. Criterios multi-dimensionales para selección del modelo óptimo.	62
4.24. Protocolo de validación cruzada para robustez estadística.	62
4.25. Métricas de benchmarking computacional para evaluación de eficiencia.	62
4.26. Estructura del reporte de resultados comparativo.	63
4.27. Especificaciones del entorno de desarrollo para algoritmos metaheurísticos.	63
4.28. Especificaciones detalladas del hardware utilizado para entrenamiento de DistilBERT.	64
4.29. Optimizaciones de hardware implementadas para maximizar eficiencia.	64
4.30. Stack tecnológico completo utilizado en el desarrollo del proyecto.	65
4.31. Marco ético implementado para el desarrollo responsable de la herramienta.	66
4.32. Artefactos generados para facilitar la reproducibilidad completa del estudio.	67
5.1. Corpus académicos utilizados para la construcción del dataset unificado.	70
5.2. Configuración detallada de parámetros para los cinco algoritmos metaheurísticos implementados.	71
5.3. Función de evaluación común utilizada por todos los algoritmos metaheurísticos.	72
5.4. Pseudocódigo completo del algoritmo Multi-Start Simulated Annealing (MSA).	73
5.5. Pseudocódigo completo del algoritmo Scatter Search (SS).	74
5.6. Pseudocódigo completo del Algoritmo Genético (GA).	75
5.7. Pseudocódigo completo del algoritmo Variable Neighborhood Search (VNS).	76
5.8. Pseudocódigo completo del algoritmo Particle Swarm Optimization (PSO).	76
5.9. Resultados comparativos finales de los cinco algoritmos metaheurísticos implementados usando métricas macro promedio.	85

5.10. Análisis de fortalezas y debilidades de cada algoritmo metaheurístico basado en métricas macro.	85
5.11. Comparación de rendimiento entre enfoques metaheurísticos y modelos de lenguaje usando métricas macro.	86
5.12. Composición del corpus expandido utilizado para el entrenamiento de DistilBERT.	89
5.13. Configuración arquitectónica del modelo DistilBERT implementado.	89
5.14. Técnicas de regularización implementadas en la configuración V7 final.	90
5.15. Resumen de versiones experimentales de DistilBERT con evolución de estrategias y resultados reales del desarrollo.	93
5.16. Visualización comparativa de convergencia y métricas para todas las versiones experimentales de DistilBERT.	94
5.17. Evolución de configuraciones y resultados entre versiones experimentales.	95
5.18. Pseudocódigo simplificado del algoritmo DistilBERT V7.	98
5.19. Métricas finales del modelo DistilBERT optimizado.	99
5.20. Comparación DistilBERT vs. mejor algoritmo metaheurístico.	99

Capítulo 1

Introducción

1.1. Planteamiento del problema

En la era de la información digital, la interconexión global ha traído consigo un desafío sin precedentes: la propagación masiva de desinformación y fraude digital. Este fenómeno, que abarca desde noticias falsas (*fake news*) hasta complejas estafas en línea, representa una amenaza significativa para la estabilidad social, económica y democrática a nivel mundial.

Las noticias falsas, definidas como información deliberadamente engañosa disfrazada de periodismo auténtico [1], se difunden a través de redes sociales y medios digitales con el fin de manipular la opinión pública y el comportamiento de los individuos. La gravedad de esta problemática se ha intensificado exponencialmente con la sofisticación de los actores maliciosos, la democratización de herramientas de generación de contenido mediante Inteligencia Artificial [2], y la velocidad sin precedentes con la que la información se viraliza en el ecosistema digital.

La investigación reciente ha documentado el impacto multidimensional de la desinformación, que va desde la distorsión de procesos democráticos [3] hasta la creación de pánico público durante crisis sanitarias como la pandemia de COVID-19 [4]. Paralelamente, el fraude digital ha evolucionado hacia formas cada vez más sofisticadas, aprovechando tanto vulnerabilidades técnicas como sesgos cognitivos humanos [5].

El fraude digital contemporáneo se manifiesta a través de múltiples modalidades con un impacto significativo en individuos, empresas y sociedades, como se ilustra en la Figura 1.1:

- **Noticias falsas generadas por IA:** Contenido sintético creado por Grandes Modelos de Lenguaje (LLMs) que imita el estilo periodístico legítimo [6]
- **Desinformación política dirigida:** Campañas coordinadas para influir en procesos electorales y opinión pública [7]
- **Fraude financiero digital:** Esquemas que explotan plataformas digitales y criptomonedas [8]

- **Fraude laboral en línea:** Ofertas de empleo falsas que buscan obtener información personal o financiera [9]
- **Estafas de ingeniería social:** Técnicas sofisticadas que combinan información personal extraída de redes sociales con narrativas convincentes
- **Desinformación en salud:** Información médica falsa que puede tener consecuencias directas en la salud pública [10]

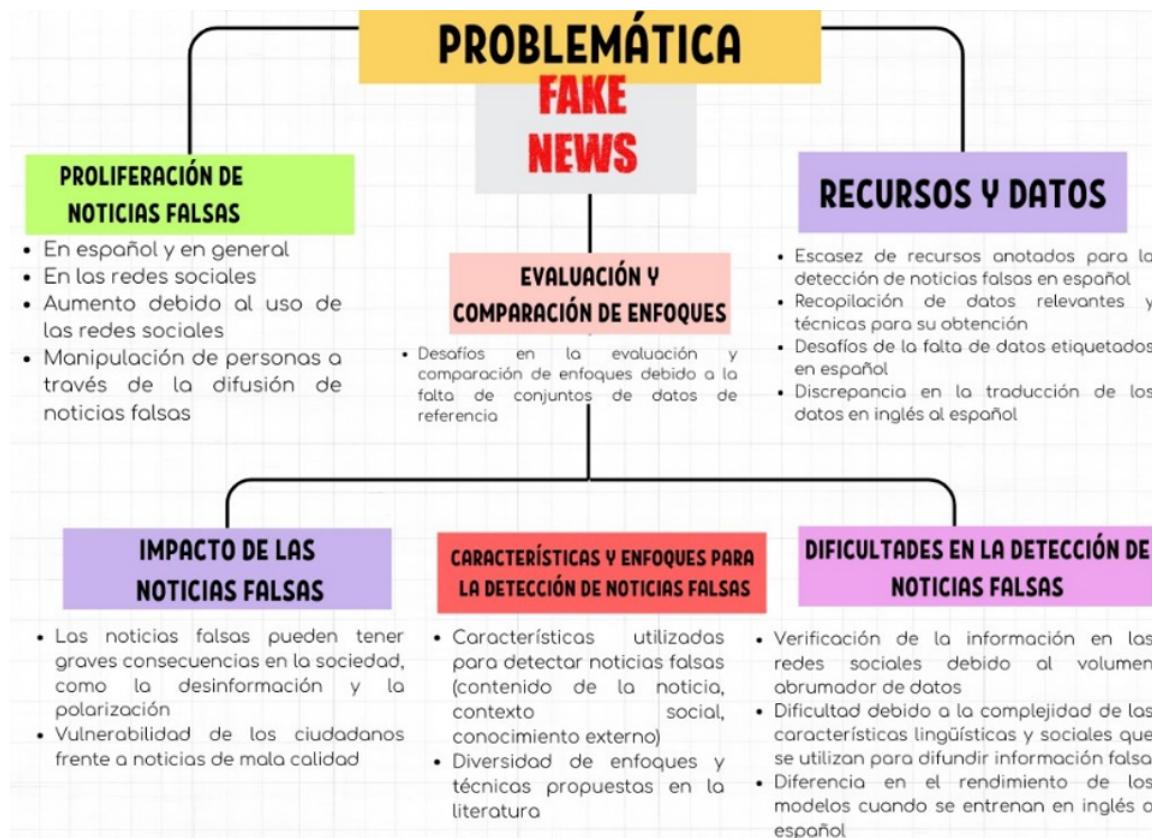


Figura 1.1: Mapa Conceptual 1: Taxonomía del problema de desinformación y fraude digital en la era de la IA.

1.1.1. El Desafío Específico del Español

El español, como la cuarta lengua más hablada del mundo con más de 500 millones de hablantes nativos [11], presenta desafíos únicos para la detección automatizada de desinformación. A pesar de su importancia demográfica y económica, existe una notable escasez de recursos computacionales especializados para la detección de noticias falsas en español, en comparación con los abundantes recursos disponibles para el inglés [12].

Esta brecha de recursos se manifiesta en:

- **Escasez de corpus etiquetados:** Limitados conjuntos de datos de entrenamiento en español para modelos de detección
- **Variabilidad dialectal:** La diversidad regional del español presenta desafíos adicionales para modelos generalizables
- **Contexto cultural específico:** Los patrones de desinformación varían según el contexto sociocultural hispanoamericano
- **Herramientas de detección limitadas:** Pocas soluciones tecnológicas disponibles para comunidades hispanohablantes

1.1.2. La Complejidad Técnica del Problema

La detección automatizada de noticias falsas constituye un problema técnico multifacético que requiere la integración de múltiples disciplinas. Como documenta la literatura reciente [13], los desafíos incluyen:

- **Análisis semántico profundo:** Necesidad de comprender el contexto y las implicaciones sutiles del contenido
- **Detección de patrones estilométricos:** Identificación de características lingüísticas que indiquen autoría maliciosa [14]
- **Procesamiento en tiempo real:** Capacidad de analizar el volumen masivo de contenido generado diariamente
- **Adaptación a contenido sintético:** Detección de texto generado por modelos de IA cada vez más sofisticados [15]
- **Robustez ante ataques adversariales:** Resistencia a intentos deliberados de evadir la detección

Dada la sofisticación de estas amenazas, se requieren soluciones tecnológicas igualmente avanzadas para combatir el fraude digital, como se esquematiza en la Figura 1.2. Este trabajo se centra en el desarrollo de tales soluciones, con un enfoque particular en el idioma español y la comparación sistemática de paradigmas tecnológicos complementarios.

1.2. Motivación

La motivación para llevar a cabo esta investigación se fundamenta en una combinación de experiencias personales observadas y la identificación de una brecha crítica en la protección tecnológica de las comunidades hispanohablantes.



Figura 1.2: Mapa Conceptual 2: Estrategias tecnológicas para la detección de fraude digital.

1.2.1. Impacto Personal y Social Observado

Durante el desarrollo de esta investigación, se observaron múltiples casos en el entorno cercano donde personas fueron víctimas de fraude digital sofisticado. Estos casos incluyeron desde estafas de inversión disfrazadas de noticias financieras legítimas, hasta esquemas de phishing que aprovechaban eventos noticiosos actuales para parecer creíbles. Las víctimas, frecuentemente personas de edad avanzada o con menor exposición a tecnología digital, sufrieron no solo pérdidas económicas significativas, sino también impacto psicológico profundo, incluyendo sentimientos de vergüenza, ansiedad y pérdida de confianza en medios digitales.

1.2.2. Vulnerabilidad de Poblaciones Específicas

La investigación en psicología cognitiva aplicada a la desinformación [5] ha demostrado que ciertos grupos demográficos son particularmente vulnerables:

- **Adultos mayores:** Mayor susceptibilidad a heurísticas de credibilidad basadas en autoridad percibida

- **Poblaciones con menor alfabetización digital:** Limitada capacidad para evaluar la legitimidad de fuentes online
- **Comunidades con acceso limitado a información:** Mayor dependencia de redes sociales como fuente primaria de noticias
- **Hablantes nativos de español:** Menor disponibilidad de herramientas de verificación en su idioma nativo

1.2.3. Urgencia Tecnológica

El rápido avance en modelos generativos de IA, como GPT-3 [16] y sus sucesores, ha reducido significativamente las barreras técnicas para la creación de contenido falso convincente. Esta democratización de la capacidad de generar desinformación [6] crea una urgencia imperativa para desarrollar defensas tecnológicas igualmente sofisticadas.

Impulsado por la necesidad de crear defensas tecnológicas más robustas y específicamente adaptadas para la comunidad hispanohablante, el presente trabajo se centra en desarrollar, comparar y validar métodos computacionales avanzados para la detección y prevención del fraude digital, con el objetivo final de contribuir a la protección de las poblaciones más vulnerables frente a estas amenazas emergentes.

1.3. Justificación

Esta investigación se justifica desde múltiples perspectivas: la brecha tecnológica existente, la novedad metodológica del enfoque, y la necesidad social de herramientas especializadas para el español.

1.3.1. Brecha Tecnológica en Recursos para el Español

El español, con más de 500 millones de hablantes nativos distribuidos en 21 países, representa un vasto ecosistema digital que ha sido históricamente subatendido en términos de herramientas especializadas para la detección de desinformación. Mientras que para el inglés existen múltiples datasets de gran escala como LIAR, FakeNewsNet, y CREDBANK [17], los recursos equivalentes en español son limitados y fragmentados.

El estado del arte actual en español se basa principalmente en cuatro corpora principales:

- **Corpus de Acosta (2019):** 598 noticias [11]
- **Spanish Fake News Corpus:** 971 noticias [12]
- **Corpus de Tretiakov (2022):** 1,958 noticias [18]

- **Spanish Political Fake News:** 57,000+ noticias [19]

Esta fragmentación crea una barrera significativa para el desarrollo de modelos robustos y generalizables.

1.3.2. Novedad Metodológica: Enfoque Evolutivo Comparativo

La novedad principal de esta investigación radica en su enfoque evolutivo que compara sistemáticamente dos paradigmas fundamentalmente diferentes de la Inteligencia Artificial en el mismo contexto aplicado:

Paradigma Clásico Optimizado

- **Representación textual:** Bolsa de Palabras (BoW) con ponderación TF-IDF
- **Optimización:** Algoritmos metaheurísticos para calibración de hiperparámetros
- **Clasificador:** Máquina de Vectores de Soporte (SVM) optimizada
- **Ventajas:** Eficiencia computacional, interpretabilidad, menor dependencia de hardware especializado

Paradigma de Deep Learning

- **Representación textual:** Embeddings contextuales dinámicos
- **Modelo base:** DistilBERT pre-entrenado [20]
- **Técnica:** Fine-tuning con optimización de hiperparámetros
- **Ventajas:** Comprensión semántica profunda, captura de relaciones contextuales complejas

1.3.3. Contribución Científica Multidimensional

Esta investigación ofrece contribuciones en múltiples dimensiones:

Contribución a Recursos Lingüísticos

- **Corpus unificado:** Integración y estandarización de los principales corpus en español
- **Metodología de unificación:** Protocolo replicable para integrar datasets heterogéneos
- **Benchmarking:** Establecimiento de líneas base comparativas para futura investigación

Contribución Metodológica

- **Optimización metaheurística:** Aplicación sistemática de algoritmos bio-inspirados para calibración de hiperparámetros en ambos paradigmas
- **Análisis comparativo riguroso:** Evaluación exhaustiva usando métricas múltiples y validación cruzada
- **Transferencia tecnológica:** Implementación práctica en aplicaciones web containerizadas

Contribución Aplicada

- **Herramientas funcionales:** Aplicaciones web deployables para análisis en tiempo real
- **Código abierto:** Disponibilidad pública de implementaciones para reproducibilidad
- **Impacto social directo:** Herramientas utilizables por comunidades hispanohablantes

1.3.4. Relevancia en el Contexto de LLMs

Con el advenimiento de Grandes Modelos de Lenguaje como GPT-3 [16], LLaMA [21], y Gemini [22], la generación de contenido sintético convincente se ha democratizado significativamente. Esta evolución hace que la investigación en detección sea más urgente y relevante, particularmente para idiomas como el español que han recibido menor atención en el desarrollo de contramedidas tecnológicas.

1.4. Objetivos

Objetivo General

Desarrollar un método computacional basado en algoritmos metaheurísticos y modelos de lenguaje para detectar noticias falsas y fraude digital en español.

Objetivos Específicos

- Recopilar y procesar un conjunto de datos diverso a partir de múltiples corpora en español, y enriquecerlo mediante técnicas de *web scraping* para entrenar y evaluar los sistemas de detección.
- Implementar un sistema de detección inicial que utilice técnicas de Procesamiento del Lenguaje Natural (Bolsa de Palabras) y algoritmos metaheurísticos.

- Desarrollar un sistema de detección avanzado mediante el ajuste fino (*fine-tuning*) de un modelo de lenguaje profundo (DistilBERT), optimizando su rendimiento a través de la calibración de hiperparámetros.
- Realizar un análisis comparativo del rendimiento entre el enfoque metaheurístico y el modelo de lenguaje, utilizando un conjunto completo de métricas de evaluación (Exactitud, Precisión, Exhaustividad y F1-Score).
- Desarrollar un prototipo de aplicación web funcional, contenerizada con Docker, para demostrar la aplicabilidad práctica del modelo de mayor rendimiento en el análisis de URLs para categorizar contenido digital como fraudulento o legítimo.

1.5. Alcance y Limitaciones

1.5.1. Alcance

Alcance Lingüístico y Cultural

- **Idioma objetivo:** El proyecto se centra exclusivamente en textos en español, abarcando múltiples variedades regionales representadas en los corpus utilizados
- **Dominio de aplicación:** Detección de noticias falsas como caso de uso principal, con extensibilidad demostrada hacia otros tipos de fraude digital
- **Contexto geográfico:** Cobertura de múltiples países hispanohablantes a través de los corpus integrados

Alcance Técnico

- **Modalidad de datos:** Procesamiento exclusivo de contenido textual (no multimedial)
- **Tipo de clasificación:** Clasificación binaria supervisada (FALSO/REAL)
- **Arquitecturas evaluadas:** Comparación sistemática entre enfoques clásicos optimizados y modelos Transformer
- **Implementación práctica:** Desarrollo de aplicaciones web funcionales y containerizadas

Alcance Metodológico

- **Optimización metaheurística:** Aplicación de tres algoritmos bio-inspirados para calibración de hiperparámetros
- **Validación experimental:** Uso de validación cruzada estratificada y métricas múltiples

- **Reproducibilidad:** Documentación completa y código fuente disponible

1.5.2. Limitaciones

Limitaciones de Datos

- **Dependencia de corpus existentes:** El rendimiento está condicionado por la calidad y representatividad de los corpus disponibles en español
- **Sesgos inherentes:** Posibles sesgos temporales, temáticos o geográficos presentes en las fuentes originales
- **Evolución del lenguaje:** Los modelos pueden no capturar patrones emergentes en desinformación generada por IA más reciente
- **Etiquetado ground truth:** Dependencia de la calidad del etiquetado manual en los corpus originales

Limitaciones Técnicas

- **Recursos computacionales:** El entrenamiento de modelos Transformer requiere hardware especializado (GPU) y tiempos de cómputo extensos (12-72+ horas)
- **Escalabilidad en tiempo real:** Los prototipos funcionan bajo demanda, no están optimizados para procesamiento de streams masivos
- **Generalización cross-domain:** Los modelos están específicamente entrenados para noticias, la efectividad en otros tipos de fraude digital es inferencial
- **Robustez adversarial:** No se evalúa específicamente la resistencia a ataques de evasión intencionales

Limitaciones Operacionales

- **Herramienta de apoyo:** Los sistemas desarrollados son herramientas de apoyo a la decisión, no reemplazan el juicio humano experto
- **Verificación absoluta:** La determinación definitiva de veracidad puede requerir investigación periodística especializada
- **Contexto dinámico:** Los patrones de desinformación evolucionan constantemente, requiriendo actualización periódica de los modelos
- **Consideraciones éticas:** No se abordan explícitamente implicaciones de sesgo algorítmico o impacto en libertad de expresión

Limitaciones de Evaluación

- **Métricas tradicionales:** La evaluación se basa en métricas estándar que pueden no capturar completamente la complejidad del problema
- **Validación temporal:** No se realiza validación temporal explícita con datos de períodos posteriores al entrenamiento
- **Análisis de errores:** El análisis cualitativo de errores es limitado debido al volumen de datos procesado

1.6. Contribuciones Esperadas

1.6.1. Contribuciones Teóricas

- **Metodología comparativa:** Marco sistemático para comparar paradigmas clásicos y modernos en detección de desinformación
- **Optimización metaheurística:** Aplicación novedosa de algoritmos bio-inspirados para calibración de modelos Transformer
- **Análisis de trade-offs:** Caracterización cuantitativa de compromisos entre eficiencia computacional y rendimiento predictivo

1.6.2. Contribuciones Prácticas

- **Corpus unificado:** Recurso consolidado para investigación futura en español
- **Herramientas funcionales:** Aplicaciones deployables para uso comunitario
- **Código abierto:** Implementaciones reproducibles y extensibles

1.6.3. Contribuciones Sociales

- **Protección comunitaria:** Herramientas accesibles para comunidades hispanohablantes
- **Democratización tecnológica:** Reducción de la brecha digital en herramientas de verificación
- **Capacitación y concientización:** Contribución a la alfabetización digital en detección de desinformación

1.7. Organización del Documento

Este documento se estructura de manera lógica y progresiva para guiar al lector a través del proceso completo de investigación, desarrollo y evaluación:

- **Capítulo 2 - Marco Teórico:** Establece los fundamentos teóricos que sustentan ambas metodologías, desde técnicas clásicas de representación textual hasta arquitecturas Transformer modernas, incluyendo principios de optimización metaheurística.
- **Capítulo 3 - Estado del Arte:** Presenta una revisión comprehensiva y organizada temáticamente de la literatura relevante, identificando brechas de conocimiento y posicionando esta investigación en el contexto científico actual.
- **Capítulo 4 - Metodología:** Detalla la metodología evolutiva seguida, desde la construcción del corpus unificado hasta la implementación y optimización de ambos paradigmas de detección.
- **Capítulo 5 - Análisis de Resultados:** Presenta y compara exhaustivamente los resultados obtenidos por ambas metodologías, incluyendo análisis estadístico, métricas de rendimiento y discusión de fortalezas y limitaciones.
- **Capítulo 6 - Implementación de Prototipo:** Describe la arquitectura, desarrollo y despliegue de las aplicaciones web funcionales que demuestran la viabilidad práctica de los modelos desarrollados.
- **Capítulo 7 - Conclusiones y Trabajo Futuro:** Sintetiza los hallazgos principales, evalúa el cumplimiento de objetivos, y propone direcciones específicas para investigación futura en el campo.

Cada capítulo se construye sobre los anteriores, manteniendo coherencia narrativa y técnica a lo largo del documento, mientras proporciona la profundidad analítica necesaria para validar las contribuciones científicas y prácticas de esta investigación.

Capítulo 2

Marco Teórico

En esta sección se describen las bases teóricas y los conceptos fundamentales que sustentan las dos metodologías de detección de fraude digital desarrolladas en este proyecto de investigación. Se abordan desde las técnicas clásicas de representación de texto y optimización metaheurística, hasta los paradigmas de aprendizaje profundo que definen el estado del arte actual.

2.1. Detección de Noticias Falsas y Fraude Digital

La detección de noticias falsas y fraude digital constituye un problema multifacético que requiere un enfoque interdisciplinario. La desinformación se caracteriza por ser información deliberadamente falsa o engañosa que se presenta como noticia legítima [1]. Esta problemática ha evolucionado significativamente con el advenimiento de las redes sociales y los medios digitales, donde la velocidad de propagación supera ampliamente la capacidad de verificación tradicional.

En el contexto de la detección automatizada, se han explorado diferentes enfoques y técnicas. Das et al. [23] propusieron un marco de ensamblaje basado en incertidumbre e impulsado por heurísticas para la detección de noticias falsas en tuits y artículos de noticias. Este enfoque parte de la premisa de que diferentes modelos pueden tener distintas fortalezas y debilidades, y que la combinación inteligente de múltiples predictores puede superar las limitaciones individuales.

El equipo de la Southern Methodist University presentó una solución pionera utilizando procesamiento de lenguaje natural y aprendizaje profundo para analizar tanto titulares como el contenido completo de las noticias [24]. Su trabajo estableció un precedente importante al demostrar la viabilidad de la vectorización TF-IDF combinada con redes neuronales densas.

Complementariamente, se ha propuesto un enfoque basado en análisis estilométrico utilizando Procesamiento del Lenguaje Natural (PLN) y Reconocimiento de Entidades Nombradas (NER) para identificar patrones lingüísticos que sugieran baja veracidad de la información [14]. Este enfoque aprovecha la hipótesis de que los autores de contenido falso pueden exhibir patrones de escritura distintivos.

2.1.1. Impacto Social y Desafíos de la Desinformación

La rápida propagación de las noticias falsas a través de las redes sociales puede tener graves consecuencias en la sociedad. Como documentan Ali y Zain-Ul-Abdin [3], la desinformación puede:

- **Distorsionar la realidad:** Alterando la percepción pública de eventos y hechos
- **Manipular la opinión pública:** Influenciando procesos democráticos y decisiones sociales
- **Incitar a la violencia:** Promoviendo comportamientos agresivos basados en información errónea
- **Difundir propaganda política:** Siendo utilizada como herramienta de influencia partidista
- **Fomentar el odio:** Exacerbando divisiones sociales y promoviendo discriminación
- **Provocar pánico:** Especialmente en contextos de crisis sanitarias como la pandemia de COVID-19 [4]

2.1.2. El Problema de las Heurísticas Cognitivas

La investigación de Ali et al. [5] ha demostrado que las heurísticas cognitivas humanas, como la popularidad social (número de "me gusta." compartidos), influyen significativamente en la percepción de credibilidad. Este fenómeno complica la detección, ya que el contenido falso puede volverse viral precisamente por aprovechar estos sesgos cognitivos.

2.2. Representación de Texto: Desde BoW hasta Embeddings Contextuales

La evolución de las técnicas de representación textual ha sido fundamental para el progreso en PLN. Esta sección aborda desde los métodos clásicos hasta las representaciones más sofisticadas utilizadas en esta tesis.

2.2.1. Bolsa de Palabras (Bag-of-Words - BoW)

El modelo de Bolsa de Palabras (BoW) constituye una de las técnicas más fundamentales para la representación de texto. A pesar de su simplicidad, sigue siendo relevante como línea base y componente en sistemas híbridos. El proceso implica:

1. **Construcción del vocabulario:** Creación de un diccionario con todas las palabras únicas presentes en el corpus de documentos

2. **Vectorización:** Representación de cada documento como un vector disperso, donde cada dimensión corresponde a una palabra del vocabulario y su valor representa la frecuencia de aparición

Aunque este método ignora la gramática y el orden de las palabras, ha demostrado ser efectivo como base para métodos más sofisticados y fue fundamental en los primeros trabajos de clasificación de noticias en español [11].

2.2.2. Ponderación TF-IDF (Term Frequency-Inverse Document Frequency)

La técnica TF-IDF mejora significativamente la representación BoW al introducir ponderación semántica. Se calcula como el producto de dos componentes:

$$\text{TF-IDF}(t, d, D) = \text{TF}(t, d) \times \text{IDF}(t, D) \quad (2.1)$$

Donde:

- **Frecuencia de Término (TF):** $\text{TF}(t, d) = \frac{f_{t,d}}{\sum_{t' \in d} f_{t',d}}$ - frecuencia relativa del término t en el documento d
- **Frecuencia Inversa de Documento (IDF):** $\text{IDF}(t, D) = \log \frac{|D|}{|\{d \in D : t \in d\}|}$ - importancia global del término en la colección

Esta ponderación asigna mayor peso a términos que son frecuentes en un documento específico pero raros en el corpus general, mejorando la capacidad discriminativa del modelo [24].

2.2.3. Limitaciones de los Métodos Clásicos

Los enfoques basados en BoW y TF-IDF presentan limitaciones fundamentales:

- **Pérdida de información secuencial:** No capturan el orden ni la estructura sintáctica
- **Problema de dispersidad:** Generan representaciones muy dispersas en vocabularios grandes
- **Ausencia de semántica:** No modelan relaciones semánticas entre palabras
- **Falta de contexto:** Una palabra tiene la misma representación independientemente del contexto

2.3. La Revolución Transformer y los Modelos de Lenguaje Modernos

2.3.1. La Arquitectura Transformer: Fundamentos

La arquitectura Transformer, introducida por Vaswani et al. [25] en el trabajo seminal "Attention Is All You Need", revolucionó el campo del PLN. Su innovación principal radica en el mecanismo de **auto-atención (self-attention)**, que permite al modelo calcular representaciones ponderando dinámicamente la importancia de cada elemento en una secuencia con respecto a todos los demás elementos.

Mecanismo de Atención Multi-Cabeza

El mecanismo de atención se define matemáticamente como:

$$\text{Attention}(Q, K, V) = \text{softmax} \left(\frac{QK^T}{\sqrt{d_k}} \right) V \quad (2.2)$$

Donde Q , K , y V representan las matrices de consultas (queries), claves (keys) y valores (values), respectivamente. La atención multi-cabeza extiende este concepto:

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \dots, \text{head}_h)W^O \quad (2.3)$$

Esta arquitectura permite capturar diferentes tipos de relaciones lingüísticas simultáneamente, superando las limitaciones de los modelos secuenciales previos como RNNs y LSTMs.

2.3.2. BERT y la Era del Pre-entrenamiento Bidireccional

BERT (Bidirectional Encoder Representations from Transformers) [26] introdujo el paradigma de pre-entrenamiento bidireccional, entrenando el modelo para predecir palabras enmascaradas considerando tanto el contexto izquierdo como el derecho. Este enfoque genera representaciones contextuales más ricas que los modelos unidireccionales previos.

Variantes de BERT para Eficiencia

El éxito de BERT motivó el desarrollo de variantes más eficientes:

- **DistilBERT** [20]: Utiliza destilación de conocimiento para reducir el tamaño del modelo en un 40 % manteniendo el 97 % del rendimiento
- **TinyBERT** [27]: Aplica destilación a nivel de transformador y predicción para crear modelos aún más compactos

2.3.3. Aplicaciones en Detección de Noticias Falsas en Español

Para el español específicamente, se han desarrollado modelos especializados y evaluaciones:

- Martínez-Gallego et al. [28] exploraron la aplicación de BERT y BETO (BERT en español), estableciendo líneas base importantes
- Blanco-Fernández et al. [19] compararon sistemáticamente BERT y RoBERTa para detección de desinformación política
- Shushkevich et al. [29] investigaron la mejora de clasificación multiclas usando datos aumentados con ChatGPT

2.3.4. Grandes Modelos de Lenguaje (LLMs) y Nuevos Paradigmas

GPT-3 y el Paradigma Few-Shot

GPT-3 [16] marcó un hito al demostrar capacidades emergentes de few-shot learning. Con 175 mil millones de parámetros, mostró que los modelos de gran escala pueden realizar tareas sin ajuste fino específico, solo con ejemplos en el prompt.

LLaMA y la Democratización de LLMs

LLaMA [21] representa el esfuerzo por democratizar el acceso a modelos de gran escala, proporcionando alternativas open-source a modelos propietarios. Su arquitectura optimizada permite un rendimiento competitivo con menor costo computacional.

Modelos Multimodales: Gemini

Gemini [22] introduce capacidades multimodales nativas, procesando texto, imágenes y audio de forma integrada. Esta capacidad es relevante para la detección de desinformación que cada vez más incorpora elementos multimedia.

IA Constitucional

El trabajo de Bai et al. [30] sobre IA Constitucional aborda la alineación y seguridad de los LLMs, estableciendo principios para entrenar modelos más seguros y alineados con valores humanos. Este enfoque es crucial cuando los LLMs se utilizan para tareas de detección de desinformación.

2.3.5. El Doble Rol de los LLMs en Detección

La investigación reciente ha revelado que los LLMs presentan un doble rol:

- **Como "Buenos Consejeros":** Pueden ser fine-tuned para detectar noticias falsas con alta precisión [2]
- **Como "Malos Actores":** Pueden generar desinformación convincente, complicando la detección [6]
- **Adaptación Necesaria:** Los sistemas de detección deben adaptarse a la era de LLMs [15]

2.4. Optimización Metaheurística en Detección de Fraude

2.4.1. Fundamentos de las Metaheurísticas

Las metaheurísticas son estrategias de optimización de alto nivel que guían procesos de búsqueda para encontrar soluciones de alta calidad en espacios de búsqueda complejos. A diferencia de los algoritmos exactos, las metaheurísticas buscan soluciones "suficientemente buenas." en tiempo computacional razonable [31].

En el contexto de la detección de fraude digital, las metaheurísticas abordan varios problemas de optimización:

- **Calibración de hiperparámetros:** Optimización de parámetros de modelos de ML/DL
- **Selección de características:** Identificación de subconjuntos óptimos de variables
- **Arquitectura neural:** Diseño automático de topologías de red
- **Balanceado de datos:** Optimización de técnicas de muestreo

2.4.2. Aplicaciones en Detección de Noticias Falsas

Modelado como Problema de Scheduling

Aqil y Lahby [32] propusieron una perspectiva innovadora, modelando la detección de noticias falsas como un problema de Job Shop Scheduling. Compararon tres metaheurísticas:

- **Algoritmo Genético (GA):** Evolución de soluciones mediante selección, cruce y mutación
- **Optimización por Enjambre de Partículas (PSO):** Búsqueda inspirada en comportamiento de bandadas
- **Colonia de Abejas Artificiales (ABC):** Algoritmo basado en comportamiento de forrajeo de abejas

Sus resultados mostraron que el algoritmo Iterated Greedy (IG) superó a las metaheurísticas bioinspiradas en este contexto específico.

Optimización de Hiperparámetros en Deep Learning

Bacanin et al. [33] demostraron los beneficios de usar metaheurísticas para la calibración de hiperparámetros en modelos de deep learning. Su investigación mostró mejoras significativas sobre métodos tradicionales como grid search y random search:

- **Eficiencia computacional:** Exploración más inteligente del espacio de hiperparámetros
- **Evitación de óptimos locales:** Capacidad de escape de configuraciones subóptimas
- **Adaptabilidad:** Ajuste dinámico según el comportamiento del modelo

La investigación que publiqué en [34] específicamente aborda la calibración de hiperparámetros en algoritmos metaheurísticos para detección de fraude digital, proporcionando un marco metodológico relevante para esta tesis.

Frameworks de Ensemble Heurístico

Das et al. [23] desarrollaron un marco de ensemble que combina:

- **Modelos pre-entrenados:** BERT y similares para captura semántica
- **Características estadísticas:** Metadatos como URL, autor, timestamp
- **Heurísticas de incertidumbre:** Medidas de confianza para ponderación de predictores

Este enfoque híbrido logró F1-scores superiores al 95 % en datasets de referencia.

2.4.3. Metaheurísticas para Detección de Fraude Financiero

Optimización Multi-objetivo

Hidayattullah et al. [35] aplicaron metaheurísticas para optimizar la detección de fraude en estados financieros. Su enfoque multi-objetivo consideró:

- **Precisión de clasificación:** Maximización de métricas de rendimiento
- **Eficiencia computacional:** Minimización de tiempo de entrenamiento
- **Robustez:** Estabilidad ante variaciones en los datos

Los resultados mostraron que SVM optimizado con Algoritmo Genético alcanzó 96.15 % de precisión, superando significativamente a métodos tradicionales.

Optimización de Procesos Gaussianos

Horak y Sabek [36] exploraron la optimización de hiperparámetros en Gaussian Process Regression para predicción de dificultades financieras. Su trabajo destaca la importancia de la calibración metaheurística en modelos probabilísticos.

2.4.4. Algoritmos Híbridos Modernos

Enfoques Multi-thread

Yildirim [37] propuso un enfoque metaheurístico híbrido multi-thread que optimiza simultáneamente:

- Selección de características
- Parámetros del algoritmo de clasificación
- Arquitectura del ensemble

PSO Adaptativo

Deshai y Bhaskara Rao [38] desarrollaron un enfoque CNN con PSO adaptativo para detectar reseñas falsas online. Su algoritmo PSO adaptativo ajusta dinámicamente los parámetros de velocidad e inercia basándose en el rendimiento de la iteración actual.

2.5. Integración de Enfoques: Hacia Sistemas Híbridos

2.5.1. Combinación de Representaciones Clásicas y Modernas

La tendencia actual en detección de fraude digital favorece sistemas híbridos que combinan:

- **Características lingüísticas tradicionales:** TF-IDF, n-gramas, métricas de legibilidad
- **Embeddings contextuales:** Representaciones de BERT y similares
- **Metadatos estructurales:** Información temporal, de red social, y fuente
- **Características estilométricas:** Patrones de puntuación, longitud de oraciones, complejidad sintáctica

2.5.2. Arquitecturas de Ensemble Optimizadas

Las arquitecturas modernas emplean:

- **Ensemble de modelos heterogéneos:** Combinación de clasificadores tradicionales y deep learning
- **Ponderación adaptativa:** Pesos dinámicos basados en confianza y rendimiento
- **Fusión de decisiones:** Estrategias sofisticadas para combinar predicciones múltiples

2.5.3. Optimización End-to-End

Zhang et al. [39] exploraron el uso de LLMs para optimización de hiperparámetros, abriendo nuevas posibilidades para la optimización automática de pipelines completos de detección.

2.6. Desafíos y Direcciones Futuras

2.6.1. Desafíos Técnicos

- **Adaptación multilingüe:** Desarrollo de modelos robustos para múltiples idiomas
- **Detección en tiempo real:** Optimización para procesamiento de streams de alta velocidad
- **Robustez adversarial:** Resistencia a ataques de evasión sofisticados
- **Explicabilidad:** Desarrollo de modelos interpretables para decisiones críticas

2.6.2. Consideraciones Éticas y Sociales

- **Sesgos algorítmicos:** Mitigación de discriminación en sistemas de detección
- **Privacidad:** Protección de datos personales en análisis de contenido
- **Transparencia:** Apertura en metodologías y criterios de detección
- **Impacto social:** Consideración de efectos en libertad de expresión y democracia

2.7. Síntesis del Marco Teórico

Este marco teórico establece los fundamentos para las dos metodologías desarrolladas en esta tesis:

1. **Enfoque Clásico Optimizado:** Utiliza representaciones TF-IDF con optimización metaheurística de hiperparámetros, proporcionando una línea base sólida y computacionalmente eficiente
2. **Enfoque de Deep Learning:** Emplea modelos Transformer pre-entrenados con fine-tuning, aprovechando representaciones contextuales sofisticadas

La combinación de ambos enfoques, sustentada en la literatura revisada, permite abordar el problema de detección de fraude digital desde múltiples perspectivas, maximizando tanto la efectividad como la eficiencia computacional del sistema propuesto.

El marco establece también la base teórica para la contribución metodológica principal de esta tesis: la aplicación sistemática de técnicas metaheurísticas para la optimización de hiperparámetros en ambos paradigmas, desde clasificadores tradicionales hasta modelos de deep learning, en el contexto específico de textos en español.

Capítulo 3

Estado del arte

En este capítulo se presenta una revisión de la literatura académica relevante para la detección de noticias falsas y fraude digital, con un enfoque en las técnicas computacionales y los recursos disponibles para el idioma español.

3.1. Metodología de Búsqueda

La selección de los artículos, investigaciones, trabajos y tesis se efectuó con el apoyo de diversas fuentes y bases de datos académicas, entre las que se incluyen Google Scholar, Semantic Scholar, Nature, IOPscience, MDPI y Web of Science (WoS). La búsqueda se realizó utilizando una combinación de las siguientes palabras clave:

- *Spanish fake news*
- *Fake news detection techniques*
- *Fake news detection language models*
- *Employment fraud*
- *Labour fraud*
- *Heuristic fake news*
- *Simulated annealing fake news*

A partir de esta búsqueda sistemática, se identificaron y filtraron los trabajos más pertinentes para los objetivos de esta tesis, los cuales se resumen en las Tablas organizadas temáticamente de la 3.1 a la ??.

3.2. Clasificación Temática de la Literatura

La literatura revisada se organizó en categorías temáticas para facilitar el análisis y comprensión de las diferentes líneas de investigación. A continuación se presentan los 68 trabajos más relevantes clasificados según su enfoque principal.

3.2.1. Artículos sobre Metodología y Corpus de Datos

Esta categoría agrupa los trabajos fundamentales que se centran en la creación de datasets, metodologías de evaluación y marcos experimentales para la detección de noticias falsas en español.

Título del Artículo	Autores	Publicación	Año	Ref.
Construcción de un dataset de noticias para el entrenamiento y evaluación de clasificadores	Acosta, F. A. Z.	U. Politécnica de Madrid	2019	[11]
Overview of the CLEF-2023 CheckThat! Lab Task 1	Alam, F., et al.	CEUR Workshop Proc.	2023	[40]
Overview of mex-a3t at iberlef 2020: Fake news and aggressiveness analysis	Aragón, M. E., et al.	IberLEF Workshop	2020	[41]
The CLEF-2023 CheckThat! Lab	Barrón-Cedeño, A., et al.	ECIR 2023	2023	[42]
Overview of FakeDeS at IberLEF 2021: Fake News Detection in Spanish	Gómez-Adorno, H., et al.	Procesamiento del Lenguaje Natural	2021	[43]
Detection of fake news in a new corpus for the Spanish language	Posadas-Durán, J. P., et al.	J. of Intelligent and Fuzzy Systems	2019	[12]
The Spanish Fake News Corpus Version 2.0 en GitHub	Ramírez Cruz, J. M., et al.	GitHub	2021	[44]
Detection of False Information in Spanish Using Machine Learning Techniques	Tretiakov, A., et al.	Springer	2022	[18]

Tabla 3.1: Artículos sobre metodología y corpus de datos.

3.2.2. Artículos sobre Modelos de Lenguaje y Transformers

Esta sección incluye los trabajos fundamentales sobre modelos de lenguaje pre-entrenados, arquitecturas transformer y sus aplicaciones en la detección de desinformación.

Título del Artículo	Autores	Publicación	Año	Ref.
Constitutional AI: Harmlessness from AI Feedback	Bai, Y., et al.	arXiv	2022	[30]
Enhancing Misinformation Detection in Spanish with BERT and RoBERTa	Blanco-Fernández, Y., et al.	Applied Sciences	2024	[19]
Language Models are Few-Shot Learners (GPT-3)	Brown, T. B., et al.	arXiv	2020	[16]
BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding	Devlin, J., et al.	arXiv	2018	[26]
Gemini: A Family of Highly Capable Multimodal Models	Gemini Team, Google	arXiv	2023	[22]
Bad Actor, Good Advisor: Exploring the Role of LLMs in Fake News Detection	Hu, B., et al.	AAAI Conference	2024	[2]
TinyBERT: Distilling BERT for Natural Language Understanding	Jiao, X., et al.	arXiv	2019	[27]
Fake News Detection in Spanish Using Deep Learning Techniques	Martínez-Gallego, K., et al.	arXiv	2021	[28]
DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter	Sanh, V., et al.	arXiv	2019	[20]
Improving multiclass classification of fake news using BERT-Based models	Shushkevich, E., et al.	Inventions	2023	[29]
Adapting fake news detection to the era of large language models	Su, J., et al.	arXiv	2023	[15]
Fake news detectors are biased against texts generated by large language models	Su, J., et al.	arXiv	2023	[6]
LLaMA: Open and Efficient Foundation Language Models	Touvron, H., et al.	arXiv	2023	[21]
Attention is All You Need (Transformers)	Vaswani, A., et al.	arXiv	2017	[25]
Evaluation of Fake News Detection with Knowledge-Enhanced Language Models	Whitehouse, C., et al.	arXiv	2022	[45]

Tabla 3.2: Artículos sobre modelos de lenguaje y transformers.

3.2.3. Artículos sobre Técnicas Metaheurísticas y Optimización

Esta categoría incluye trabajos que emplean algoritmos metaheurísticos, optimización de hiperparámetros y enfoques de optimización para la detección de noticias falsas y fraude.

Título del Artículo	Autores	Publicación	Año	Ref.
Diseño y desarrollo de un método heurístico basado en un sistema socio-cultural	Anselmo, M. G. R.	UNAM	2013	[31]
Modeling and solving the fake news detection scheduling problem	Aqil, S., & Lahby, M.	Studies in Comp. Intelligence	2021	[32]
On the Benefits of Using Metaheuristics in Hyperparameter Tuning	Bacanin, N., et al.	Energies	2023	[33]
A heuristic-driven uncertainty-based ensemble framework for fake news detection	Das, S. D., et al.	Neurocomputing	2022	[23]
Unmasking deception: a CNN and adaptive PSO approach to detecting fake online reviews	Deshai, N., & Rao, B. B.	Soft Computing	2023	[38]
Financial Statement Fraud Detection in Indonesia using Meta-Heuristics	Hidayattullah, S., et al.	IWBIS	2020	[35]
Gaussian Process Regression's Hyperparameters Optimization to Predict Financial Distress	Horak, J., & Sabek, A.	Retos	2023	[36]
Calibración de hiper-parámetros en algoritmos metaheurísticos para la detección de fraude digital	Hurtado Avilés, G., et al.	BUAP	2024	[34]
Variable neighborhood search for weighted total domination problem	Kapunac, S., et al.	Applied Soft Computing	2023	[46]
Ensemble effort estimation with metaheuristic hyperparameters and weight optimization	Yasmin, A., et al.	PLOS ONE	2024	[47]
Improving fake news detection using K-Means and support vector machine approaches	Yazdi, K. M., et al.	Waset	2020	[48]
A novel hybrid multi-thread metaheuristic approach for fake news detection	Yildirim, G.	Applied Intelligence	2023	[37]
Using large language models for hyperparameter optimization	Zhang, M. R., et al.	openreview.net	2023	[39]

Tabla 3.3: Artículos sobre técnicas metaheurísticas y optimización.

3.2.4. Artículos sobre Revisiones y Estudios Comprehensivos

Esta sección agrupa las revisiones sistemáticas, surveys y estudios comprehensivos que analizan el estado del arte en detección de noticias falsas y técnicas relacionadas.

Título del Artículo	Autores	Publicación	Año	Ref.
A survey on fake news and rumour detection techniques	Bondielli, A., & Marcelloni, F.	Information Sciences	2019	[1]
Deep learning for fake news detection: A comprehensive survey	Hu, L., et al.	AI Open	2022	[17]
A Comprehensive Review on Fake News Detection With Deep Learning	Mridha, M. F., et al.	IEEE Access	2021	[49]
A comprehensive review on automatic detection of fake news on social media	Singh, M. K., et al.	Multimedia Tools and Applications	2023	[13]
Fake News Detection: A Deep Learning Approach	Thota, A., et al.	SMU Scholar	2018	[24]

Tabla 3.4: Artículos sobre revisiones y estudios comprehensivos.

3.2.5. Artículos sobre Detección de Fraude Financiero y Laboral

Esta categoría incluye trabajos específicos sobre la detección de fraude en contextos financieros, laborales y comerciales.

Título del Artículo	Autores	Publicación	Año	Ref.
El fraude en México: daños patrimoniales y trabajo legislativo para enfrentarlo	Aguirre Quezada, J. P.	Senado de México	2023	[50]
Fraude laboral en la era digital	Alvarez, O. H.	Rev. Gen. de Derecho del Trabajo	2021	[51]
Corporate employment, red flags, and audit effort	Cao, J., et al.	J. of Accounting and Public Policy	2020	[8]
Online Recruitment Fraud Detection using ANN	Nasser, I. M., et al.	PICICT	2021	[9]
Proyecto de seguridad para reducir fraudes y robos en Marketplace de Facebook	Osorio-Giraldo, J. M., et al.	Coloquio de Inv. Formativa	2023	[52]
Selección de una técnica de aprendizaje de máquina para la detección de Fraude Financiero	Villamil Arcos, C.	U. Nacional de Colombia	2022	[53]

Tabla 3.5: Artículos sobre detección de fraude financiero y laboral.

3.2.6. Artículos sobre Análisis de Comportamiento Social y Heurísticas

Esta sección incluye trabajos que analizan el comportamiento humano, heurísticas cognitivas y factores sociales en la propagación y percepción de noticias falsas.

Título del Artículo	Autores	Publicación	Año	Ref.
Post-truth propaganda: heuristic processing of political fake news on Facebook	Ali, K., & Zain-Ul-Abdin, K.	J. Applied Comm. Res.	2020	[3]
Fake news on Facebook: examining the impact of heuristic cues	Ali, K., et al.	Internet Research	2021	[5]
Fake news en Chile y España: ¿Cómo los medios nos hablan de noticias falsas?	Cárcamo-Ulloa, L., et al.	J. of Iberian and Latin Am. Res.	2021	[7]
Fake news y coronavirus: detección de los principales actores y tendencias en Twitter	Pérez-Dasilva, J., et al.	El Profesional De La Información	2020	[4]
A New Application of Social Impact in Social Media for Overcoming Fake News in Health	Pulido, C., et al.	Int. J. of Env. Res. and Public Health	2020	[10]
Stylometric fake news detection based on natural language processing	Tsai, C. M.	Electronics	2023	[14]

Tabla 3.6: Artículos sobre análisis de comportamiento social y heurísticas.

3.2.7. Artículos sobre Sistemas de Información y Representación de Conocimiento

Esta categoría agrupa trabajos relacionados con sistemas de información, representación de conocimiento, ontologías y interfaces de consulta.

Título del Artículo	Autores	Publicación	Año	Ref.
Towards a definition of knowledge graphs	Ehrlinger, L., & Wöß, W.	SEMANTiCS	2016	[54]
Interfaz de consulta en idioma español para la búsqueda de información	García Robledo, G. A., et al.	UAM	2020	[55]
Representación ontológica de la arquitectura de control de un robot SCARA	Hurtado Avilés, G., et al.	BUAP	2024	[56]
Knowledge representation and reasoning	Levesque, H. J.	Annual review of computer science	1986	[57]
Natural Language Processing	Liddy, E. D.	Syracuse University	2001	[58]
Sistema de alerta temprana para monitorear la propagación de noticias falsas	Manzano-Velasco, C. E., et al.	Universidad del Cauca	2024	[59]
Detección y representación de eventos en un ambiente académico inteligente	Padilla Cuevas, J., et al.	UAM	2019	[60]
Extracción de Información Semántica para la Clasificación de Servicios Web	Reyes-Ortiz, J. A.	CENIDET	2008	[61]
Clinical Decision Support Systems: A Survey of NLP-Based Approaches	Reyes-Ortiz, J. A., et al.	DEXA	2015	[62]
Knowledge representation	Shapiro, S. C.	Encyclopedia of cognitive science	2006	[63]

Tabla 3.7: Artículos sobre sistemas de información y representación de conocimiento.

3.3. Análisis Temático de la Literatura Relevante

Más allá de la clasificación, es fundamental analizar en profundidad las contribuciones clave de cada grupo temático. A continuación, se detallan los aportes más significativos de cada categoría para la planeación y ejecución de la metodología de esta tesis.

3.3.1. El Desafío de los Datos: Creación de Corpus en Español

Un desafío fundamental para el Procesamiento del Lenguaje Natural (PLN) en español es la escasez de conjuntos de datos etiquetados a gran escala. La creación y curación de corpus es, por tanto, una línea de investigación fundamental. El trabajo de Fin de Máster de Zules Acosta [11] fue uno de los pioneros en este ámbito, centrándose en la creación de un dataset de 598 noticias en castellano. Su investigación destaca la importancia de la *ingeniería de características* para seleccionar estrategias óptimas de extracción que permitan a los modelos de aprendizaje automático clasificar eficazmente la veracidad del contenido [11].

Por otro lado, el trabajo de Posadas-Durán et al. [12] introdujo el *Spanish Fake News Corpus* (con 971 noticias), un recurso clave para analizar y detectar información engañosa mediante métodos basados en el estilo de la escritura. Este corpus ha sido fundamental en competencias internacionales como IberLEF, donde se han evaluado diversas metodologías para el español [43], incluyendo el análisis de agresividad y noticias falsas en el español de México [41].

Más recientemente, se han publicado corpus de mayor escala que han sido cruciales para esta tesis. Tretiakov et al. [18] aportaron un nuevo conjunto de datos con 1,958 noticias falsas en español, mientras que el trabajo de Blanco-Fernández et al. [19] introdujo el *Spanish Political Fake News Dataset*. Este último, con más de 57,000 noticias, representa uno de los mayores recursos disponibles y es la fuente principal de datos para esta tesis. La unificación de estos cuatro corpus constituye el fundamento metodológico de este trabajo.

3.3.2. Revolución de los Modelos de Lenguaje y Transformers

La arquitectura Transformer, introducida por Vaswani et al. [25], revolucionó el campo del PLN con su mecanismo de atención. Este trabajo fundamental sentó las bases para modelos como BERT [26], que introdujo el concepto de pre-entrenamiento bidireccional, y sus variantes optimizadas como DistilBERT [20] y TinyBERT [27], que buscan reducir el costo computacional manteniendo el rendimiento.

El paradigma de los Grandes Modelos de Lenguaje (LLMs) se consolidó con GPT-3 [16], que demostró capacidades emergentes de few-shot learning. Trabajos más recientes como LLaMA [21] han democratizado el acceso a modelos de gran escala, mientras que Gemini [22] ha avanzado hacia capacidades multimodales. La investigación en IA Constitucional [30] ha abordado la alineación y seguridad de estos sistemas.

Para el español específicamente, Martínez-Gallego et al. [28] exploraron la aplicación de BERT y BETO, mientras que Blanco-Fernández et al. [19] compararon BERT y RoBERTa para la detección de desinformación. La investigación sobre el rol dual de los LLMs [2, 15, 6] ha revelado tanto oportunidades como desafíos en su aplicación para la detección de noticias falsas.

3.3.3. Optimización y Metaheurísticas en la Detección

Los algoritmos metaheurísticos han demostrado ser herramientas valiosas para abordar problemas complejos de optimización en la detección de noticias falsas. Aqil y Lahby [32] modelaron la detección como un problema de scheduling, aplicando algoritmos genéticos y optimización por enjambre de partículas. La calibración de hiperparámetros [33, 34] ha emergido como una aplicación clave, donde las metaheurísticas superan a los métodos tradicionales de grid search.

El enfoque de ensemble con marcos heurísticos [23] ha mostrado prometedores resultados al combinar múltiples modelos con información estadística adicional. Yıldırım [37] propuso un enfoque híbrido multi-thread que optimiza tanto la selección de características como los parámetros del modelo. La aplicación de PSO para la detección de reseñas falsas [38] demuestra la versatilidad de estas técnicas más allá de las noticias.

3.3.4. Perspectivas Interdisciplinarias y Análisis Social

La comprensión del fenómeno de las noticias falsas requiere un enfoque interdisciplinario que combine aspectos técnicos con análisis social y psicológico. Ali et al. [5, 3] han investigado cómo las heurísticas cognitivas humanas, como la popularidad social ("me gusta"), influyen en la percepción de credibilidad. Su trabajo sobre el procesamiento heurístico durante eventos políticos específicos (elecciones de 2016 en EE.UU.) proporciona insights valiosos sobre la psicología de la desinformación.

El análisis del rol de los medios de comunicación [7, 4] ha revelado patrones culturales en cómo diferentes países abordan la desinformación. Pulido et al. [10] propusieron el marco SISM (Social Impact in Social Media) para combatir específicamente la desinformación en salud, un dominio crítico durante pandemias.

3.3.5. Detección de Fraude: Más Allá de las Noticias

La investigación se ha extendido hacia otros dominios de fraude digital. En el ámbito laboral, Nasser et al. [9] desarrollaron sistemas basados en redes neuronales para detectar ofertas de trabajo fraudulentas, mientras que Alvarez [51] analizó las nuevas modalidades de fraude laboral en la era digital latinoamericana.

En el sector financiero, la investigación de Hidayattullah et al. [35] demostró la efectividad de combinar aprendizaje automático con optimización metaheurística para detectar fraudes en estados financieros. Cao et al. [8] establecieron conexiones entre

indicadores de empleo corporativo y riesgo de fraude, proporcionando herramientas valiosas para auditores.

3.4. Investigación y Desarrollo de Métodos de Detección

Un gran esfuerzo de la comunidad científica se ha centrado en el desarrollo de métodos automáticos para la detección de noticias falsas. Propuestas como la de Pulido et al. [10], proponen una metodología basada en el análisis de redes sociales, denominada SISM (*Social Impact in Social Media*), para identificar y evaluar la difusión de información falsa en el ámbito de la salud. Por otro lado, revisiones exhaustivas como la de Hu et al. [17] analizan el potencial del aprendizaje profundo para detectar patrones sutiles en el lenguaje y el contenido de grandes volúmenes de datos. El Mapa Conceptual 3.1 ilustra la relación entre los artículos de revisión y aquellos enfocados en análisis de contenido y detección de fraude financiero.

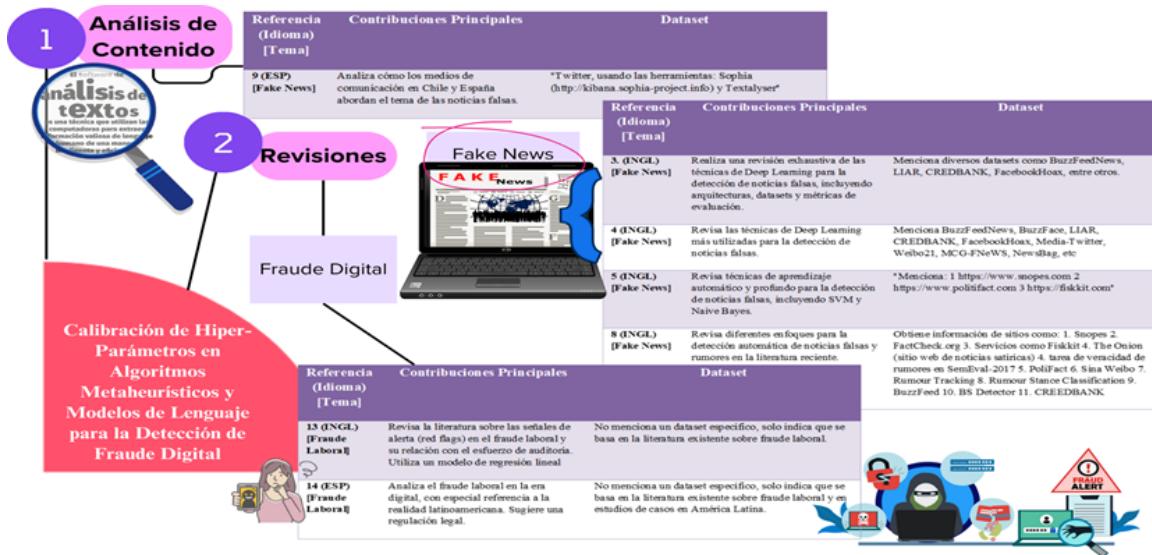


Figura 3.1: Mapa Conceptual 3: Artículos que son revisiones o están relacionados al análisis de contenido y detección de fraude financiero.

Categoría	Contribución Principal	Dataset(s) Mencionados
NLP	Propone una metodología SISM para combatir noticias falsas en salud [10].	Propio (basado en Twitter).
Revisiones	Revisa exhaustivamente técnicas de Deep Learning (arquitecturas, datasets, métricas) [17].	BuzzFeedNews, LIAR, CREDBANK.
Revisiones	Revisa diferentes enfoques para la detección automática de rumores y noticias falsas [1].	Snopes, FactCheck, PolitiFact, etc.

Tabla 3.8: Artículos relacionados con la investigación y desarrollo de métodos de detección.

3.5. Implementación de Soluciones Prácticas

Diversas iniciativas han buscado poner en práctica los avances teóricos. Posadas-Durán et al. [12] presentaron un nuevo corpus en español y un método de detección basado en el estilo de escritura. Martínez-Gallego et al. [28] exploraron el uso de técnicas de aprendizaje profundo como BERT y BETO para el español. Thota et al. [24] presentaron un enfoque similar utilizando un vectorizador TF-IDF en un modelo de Deep Learning. Adicionalmente, Whitehouse et al. [45] investigaron el impacto de integrar conocimiento externo en los Modelos de Lenguaje Pre-entrenados (PLMs) para mejorar la detección.

Categoría	Contribución Principal	Dataset(s) Mencionados
NLP	Propone un método basado en el estilo de escritura para detectar noticias falsas en español [12].	SpanishFakeNewsCorpus (de GitHub)
Deep Learning	Evaluó diferentes arquitecturas de Deep Learning para la detección de noticias falsas en español, incluyendo BERT y BETO [28].	Kaggle Fake/Real News
TF-IDF	Utiliza el vectorizador TF-IDF para la representación textual en un modelo de Deep Learning para la detección de noticias falsas [24].	fakenewschallenge.org
Modelos de Lenguaje	Evaluó la detección de noticias falsas con modelos de lenguaje mejorados con conocimiento [45].	LIAR, COVID-19 (newschecker.in)

Tabla 3.9: Artículos relacionados a la implementación de soluciones prácticas.

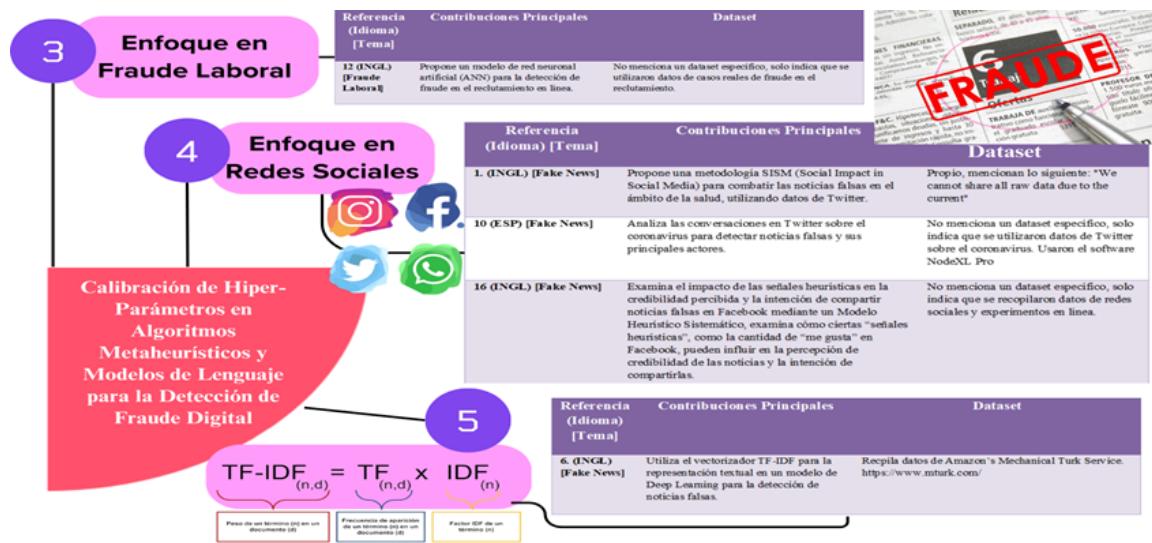


Figura 3.2: Mapa Conceptual 4: Clasificación de artículos por enfoque.

3.6. Análisis del Impacto Social y Rol de los Medios

Comprender el impacto social de la desinformación es crucial. Investigaciones como la de Singh et al. [13] abordan la complejidad y los desafíos de la detección en redes sociales. Ali et al. [5] examinaron cómo heurísticas (como la cantidad de "me gusta")

influyen en la credibilidad y la intención de compartir noticias en Facebook. El rol de los medios de comunicación es fundamental en esta lucha; trabajos como los de Ulloa et al. [7] y Pérez-Dasilva et al. [4] analizan cómo los medios en Chile, España y en el contexto de la pandemia de COVID-19 abordan la desinformación.

3.7. Detección Aplicada a Fraude Financiero y Laboral

El estado del arte se extiende a dominios críticos como el fraude financiero y laboral. Artículos como el de Cao et al. [8] analizan cómo las decisiones de empleo pueden ser indicadores ("red flags") de riesgo de fraude financiero para los auditores. En el ámbito del fraude laboral, Nasser et al. [9] presentan un modelo basado en Redes Neuronales Artificiales (ANN) para detectar publicaciones de trabajo fraudulentas, mientras que Álvarez [51] aborda las nuevas modalidades de fraude en la era digital en Latinoamérica. El Mapa Conceptual 3.2 clasifica varios artículos según su enfoque en estas problemáticas.

3.8. La Detección como un Problema de Optimización

Varios trabajos abordan la detección como un problema de optimización. Aqil y Limam [32] lo modelan como un problema de planificación de tareas (*Job Shop Scheduling*) y aplican tres metaheurísticas (IG, GA y ABC), concluyendo que IG supera a las demás. Por otro lado, Yazdi et al. [48] presentan una técnica de selección de características que combina K-means y SVM para reducir la dimensionalidad de los datos. El Mapa Conceptual 3.3 resume varios de estos métodos.

3.9. Algoritmos Metaheurísticos: Fundamentos y Aplicaciones

Los algoritmos metaheurísticos representan una clase de técnicas de optimización que han demostrado ser especialmente efectivas para resolver problemas complejos de optimización no lineal, incluyendo la calibración de hiperparámetros en modelos de aprendizaje automático. Esta sección presenta los fundamentos teóricos de los principales algoritmos metaheurísticos utilizados en la detección de noticias falsas y optimización de modelos.

3.9.1. Algoritmos Genéticos (GA)

Los Algoritmos Genéticos, introducidos por Holland [64], están inspirados en la teoría de la evolución de Darwin y los principios de la genética. Estos algoritmos mantienen una población de soluciones candidatas que evolucionan a través de operadores genéticos como selección, cruzamiento y mutación. El trabajo seminal de Holland estableció las bases teóricas para esta clase de algoritmos bioinspirados, demostrando su capacidad para explorar eficientemente espacios de búsqueda complejos y multimodales.

El funcionamiento básico involucra la evaluación de la aptitud de cada individuo en la población, la selección de los mejores candidatos para reproducción, y la generación de nuevas soluciones mediante la combinación y modificación de las características de los padres. Este proceso iterativo permite la convergencia hacia soluciones óptimas o cerca del óptimo global.

3.9.2. Optimización por Enjambre de Partículas (PSO)

La Optimización por Enjambre de Partículas fue desarrollada por Kennedy y Eberhart [65], inspirándose en el comportamiento social de bandadas de pájaros y bancos de peces. Una versión alternativa fue presentada por Eberhart y Kennedy [66] en el mismo período, consolidando esta técnica como una de las metaheurísticas más utilizadas.

En PSO, cada partícula representa una solución potencial que se mueve a través del espacio de búsqueda siguiendo su propia experiencia (mejor posición personal) y la experiencia del enjambre (mejor posición global). La velocidad y posición de cada partícula se actualizan dinámicamente, permitiendo un balance entre exploración y explotación del espacio de soluciones.

3.9.3. Recocido Simulado y Métodos Multi-arranque

El Recocido Simulado, propuesto por Kirkpatrick, Gelatt y Vecchi [67], se basa en el proceso físico de enfriamiento controlado de metales. Este algoritmo permite escapar de óptimos locales mediante la aceptación probabilística de soluciones peores, con una probabilidad que disminuye gradualmente según un esquema de enfriamiento.

Los métodos Multi-arranque, formalizados por Martí, Resende y Pardalos [68], consisten en ejecutar múltiples corridas independientes de un algoritmo de búsqueda local desde diferentes puntos de inicio. La combinación de Recocido Simulado con estrategias Multi-arranque (MSA) permite aprovechar las ventajas de ambos enfoques: la capacidad de escape de óptimos locales del primero y la diversificación del segundo.

3.9.4. Búsqueda Dispersa (Scatter Search)

La Búsqueda Dispersa, desarrollada por Glover [69], utiliza estrategias sistemáticas para combinar soluciones de referencia y generar nuevas soluciones prometedoras.

A diferencia de otros métodos que dependen de procesos aleatorios, SS emplea combinaciones determinísticas y diversificación controlada.

El algoritmo mantiene un conjunto de soluciones de referencia diversas y de alta calidad, utilizando métodos de combinación específicos del problema para generar nuevas soluciones. Esta aproximación ha demostrado ser particularmente efectiva en problemas de optimización combinatorial.

3.9.5. Búsqueda en Vecindades Variables (VNS)

La Búsqueda en Vecindades Variables, introducida por Mladenović y Hansen [70], se basa en el principio de cambio sistemático de estructuras de vecindad durante el proceso de búsqueda. Esta técnica es especialmente efectiva para escapar de óptimos locales explorando diferentes definiciones de vecindad.

VNS alterna entre fases de diversificación (mediante la exploración de vecindades más amplias) y intensificación (búsqueda local en vecindades más restringidas), proporcionando un marco flexible para la optimización que puede adaptarse a diferentes tipos de problemas.

3.9.6. Aplicaciones en Detección de Noticias Falsas

La Tabla 4.7 resume las características principales de estos algoritmos y sus aplicaciones específicas en el contexto de la detección de noticias falsas y optimización de modelos de aprendizaje automático.

Algoritmo	Inspiración	Características Principales	Aplicaciones en Detección	Ref.
Algoritmos Genéticos (GA)	Evolución biológica y genética	Población de soluciones, operadores genéticos, selección natural	Optimización de hiperparámetros, selección de características, arquitecturas de redes neuronales	[64]
Optimización por Enjambre de Partículas (PSO)	Comportamiento social de bandadas y enjambres	Partículas con velocidad y posición, memoria personal y social	Calibración de pesos en redes, optimización de parámetros de modelos de lenguaje	[65]
Recocido Simulado (SA)	Proceso físico de enfriamiento de metales	Aceptación probabilística, esquema de enfriamiento, escape de óptimos locales	Optimización de arquitecturas profundas, ajuste fino de modelos complejos	[67]
Multi-arranque (Multi-start)	Diversificación de puntos de inicio	Múltiples ejecuciones independientes, exploración global del espacio	Inicialización robusta de modelos, validación cruzada optimizada	[68]
Búsqueda Dispersa (SS)	Combinación sistemática de soluciones diversas	Conjunto de referencia, combinaciones determinísticas, diversificación controlada	Ensemble de modelos, combinación de características, optimización de pipelines	[69]
Búsqueda en Vecindades Variables (VNS)	Cambio sistemático de estructuras de vecindad	Múltiples definiciones de vecindad, alternancia entre diversificación e intensificación	Exploración de espacios de hiperparámetros, optimización de topologías de red	[70]

Tabla 3.10: Algoritmos metaheurísticos: características y aplicaciones en detección de noticias falsas.

Estos algoritmos han demostrado ser especialmente valiosos en el contexto de la detección de noticias falsas, donde los espacios de búsqueda de hiperparámetros son

complejos y multidimensionales. Su capacidad para balancear exploración y explotación los convierte en herramientas ideales para optimizar el rendimiento de modelos de aprendizaje automático en tareas de clasificación textual.

3.9.7. Aplicación de Metaheurísticas por Tipo de Detección

La literatura revisada revela patrones específicos en la aplicación de algoritmos metaheurísticos según el tipo de detección y el dominio del problema. La Tabla 3.11 presenta una taxonomía detallada de estas aplicaciones.

Tipo de Detección	Metaheurística Empleada	Aplicación Específica	Autores	Ref.
Detección de Noticias Falsas	Algoritmo Genético (GA), Colonia de Abejas (ABC), Búsqueda Iterativa (IG)	Planificación de tareas para procesamiento de documentos	Aqil, S. & Lahby, M.	[32]
Detección de Reseñas Falsas	Optimización por Enjambre de Partículas Adaptativo (PSO)	Optimización de hiperparámetros en redes CNN	Deshai, N. & Rao, B. B.	[38]
Fraude Financiero y de Estados Financieros	Algoritmos Genéticos, Optimización por Enjambre de Partículas, Recocido Simulado	Selección de características y optimización de clasificadores SVM	Hidayattullah, S. et al.	[35]
Predicción de Dificultades Financieras	Optimización Bayesiana con Procesos Gaussianos	Calibración de hiperparámetros en modelos GPR	Horak, J. & Sabek, A.	[36]
Detección de Fraude Laboral	Redes Neuronales Artificiales (ANN)	Clasificación de ofertas de empleo fraudulentas	Nasser, I. M. et al.	[9]
Optimización de Modelos de Energía	Algoritmo Genético Binario, PSO, Recocido Simulado, Búsqueda Armónica	Ajuste de hiperparámetros en modelos LSTM para predicción energética	Bacanin, N. et al.	[33]
Detección Multimodal de Noticias Falsas	Enfoque Híbrido Multi-hilo con Metaheurísticas	Optimización paralela de características textuales y visuales	Yildirim, G.	[37]
Selección de Características para Fake News	K-Means combinado con Support Vector Machine (SVM)	Reducción de dimensionalidad y mejora de precisión	Yazdi, K. M. et al.	[48]
Framework de Incertidumbre para Fake News	Enfoque Heurístico basado en Ensemble	Manejo de incertidumbre en clasificación de tweets y artículos	Das, S. D. et al.	[23]
Optimización de Dominación Total en Redes Sociales	Búsqueda en Vecindades Variables (VNS)	Propagación de información en redes sociales	Kapunac, S. et al.	[46]
Estimación de Esfuerzo en Proyectos	Ensemble con Metaheurísticas para Pesos	Optimización de hiperparámetros y pesos de ensemble	Yasmin, A. et al.	[47]

Tabla 3.11: Aplicación de metaheurísticas por tipo de detección en la literatura revisada.

Patrones Identificados por Dominio

Del análisis de la literatura se identifican tres patrones principales en la aplicación de metaheurísticas:

Detección de Contenido Textual Para la detección de noticias falsas y contenido textual fraudulento, predominan los enfoques que combinan:

- **Algoritmos Genéticos:** Utilizados principalmente para selección de características y optimización de arquitecturas de modelos [32, 35]
- **PSO Adaptativo:** Especialmente efectivo para la calibración de hiperparámetros en redes neuronales complejas [38, 33]
- **Enfoques Híbridos:** Combinación de múltiples metaheurísticas para diferentes aspectos del pipeline de detección [37]

Optimización de Modelos de Aprendizaje Profundo En aplicaciones que involucran modelos de aprendizaje profundo, se observa una preferencia por:

- **Recocido Simulado:** Para escapar de óptimos locales en espacios de hiperparámetros complejos [33]
- **Optimización Bayesiana:** Para calibración eficiente en modelos probabilísticos [36]
- **Búsqueda en Vecindades Variables:** Para exploración sistemática de configuraciones de red [46]

Sistemas de Detección en Tiempo Real Para aplicaciones que requieren procesamiento en tiempo real, las metaheurísticas se enfocan en:

- **Algoritmos de Planificación:** Como IG (Iterated Greedy) para optimización de tareas de procesamiento [32]
- **Métodos Ensemble:** Con optimización de pesos mediante metaheurísticas [23, 47]
- **Enfoques Multi-hilo:** Para paralelización de la optimización en sistemas multimodales [37]

Esta taxonomía revela que la elección de la metaheurística está fuertemente influenciada por el tipo de modelo subyacente, las características del conjunto de datos, y los requisitos de rendimiento temporal del sistema de detección.

3.10. El Rol de los Modelos de Lenguaje

Los modelos de lenguaje son centrales en la detección moderna. Como se muestra en el Mapa Conceptual 3.4, enfoques como el de Das et al. [23] combinan modelos pre-entrenados como BERT con algoritmos heurísticos que incorporan atributos de las noticias (fuente, URL, autor) como características estadísticas. Otros trabajos se centran en el análisis textual cualitativo para descubrir significados latentes [3] o utilizan metaheurísticas para optimizar el procesamiento de documentos [32].

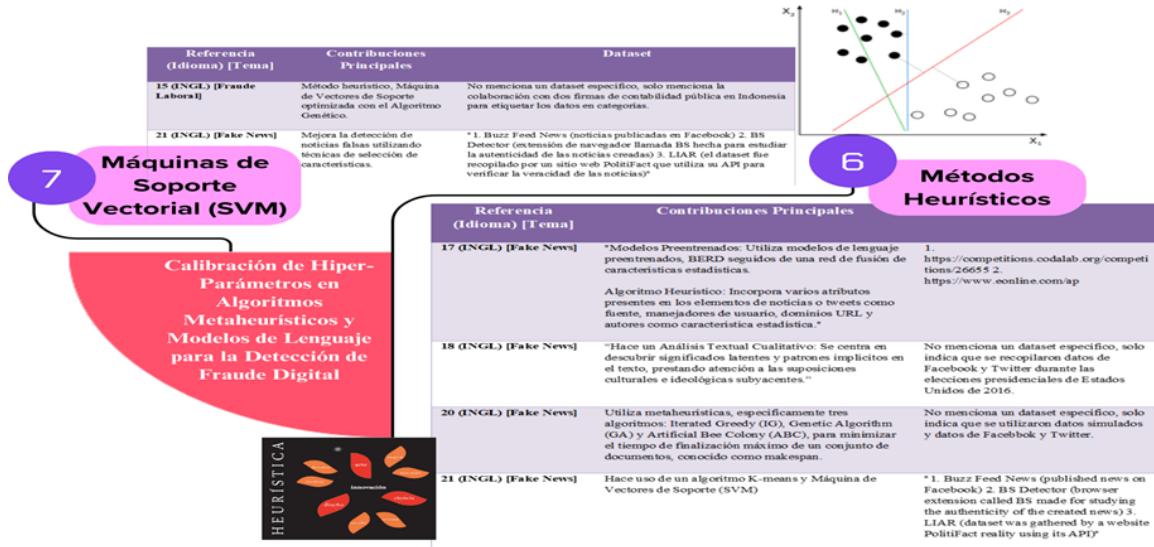


Figura 3.3: Mapa Conceptual 5: Métodos para resolver la problemática presentados en los artículos.

3.11. Hiperparámetros en Modelos de Aprendizaje Automático

3.11.1. ¿Qué son los Hiperparámetros?

Para explicar este concepto de la manera más sencilla, utilizaremos una analogía: cocinar un pastel.

Imagina que estás construyendo un modelo de machine learning (el pastel).

Los **parámetros** son los elementos que el modelo aprende por sí mismo durante el entrenamiento (la cocción). Por ejemplo, cómo se combinan la harina y el azúcar dentro del horno. En una red neuronal, estos son los pesos (*weights*) y sesgos (*biases*) de las neuronas. No los eliges tú directamente, se ajustan solos a partir de los datos.

Los **hiperparámetros**, en cambio, son las decisiones que tú tomas antes de empezar a entrenar. Son los ajustes o "perillas" de la receta que tú, como cocinero (científico de datos), debes fijar. Por ejemplo:

- **La temperatura del horno** → tasa de aprendizaje (*learning rate*)
- **El tiempo de cocción** → número de épocas
- **La cantidad de cada ingrediente principal** → número de capas o neuronas
- **El tipo de molde** → arquitectura del modelo

Si eliges mal estos ajustes, tu pastel puede quemarse (sobreajuste u *overfitting*), quedar crudo (subajuste o *underfitting*), o simplemente no tener buen sabor (bajo rendimiento).

3.11.2. ¿Por qué son Importantes en la Detección de Noticias Falsas?

En el contexto de la detección de noticias falsas, elegir los hiperparámetros correctos es crucial porque:

- Los textos en español tienen características lingüísticas específicas que requieren configuraciones particulares
- Los patrones de desinformación pueden ser muy sutiles y necesitan modelos finamente ajustados
- Una mala configuración puede hacer que el modelo confunda sarcasmo con falsedad, o que no detecte desinformación sofisticada

3.11.3. El Problema: Encontrar la Mejor Configuración

Tradicionalmente, los científicos de datos probaban diferentes combinaciones de hiperparámetros manualmente o usando métodos como:

- **Búsqueda en grilla (Grid Search):** Probar todas las combinaciones posibles - muy lento
- **Búsqueda aleatoria (Random Search):** Probar combinaciones al azar - más eficiente pero sin dirección

Aquí es donde entran los algoritmos metaheurísticos, que son como "cocineros inteligentes" que aprenden de cada pastel que hacen para mejorar la siguiente receta.

3.11.4. Herramientas Modernas: Keras Tuner

En el ecosistema de TensorFlow, Keras Tuner [71] ha emergido como una herramienta fundamental para automatizar la búsqueda de los mejores hiperparámetros. Desarrollado por el equipo de Keras, esta biblioteca permite que el sistema pruebe automáticamente diferentes recetas y encuentre la mejor configuración para tu modelo.

Keras Tuner facilita:

- Definir fácilmente qué hiperparámetros quieres optimizar
- Probar múltiples configuraciones en paralelo
- Monitorear el progreso en tiempo real
- Guardar y recuperar los mejores resultados

3.11.5. La Ventaja de Combinar Modelos de Lenguaje con Keras Tuner

Los modelos de lenguaje modernos como DistilBERT, cuando se combinan con herramientas como Keras Tuner, actúan como sistemas inteligentes que:

- Aprenden representaciones semánticas complejas del texto
- Se benefician de la optimización automática de hiperparámetros
- Pueden encontrar configuraciones óptimas de manera más eficiente que los métodos manuales
- Balancean entre el aprendizaje de características profundas (exploración) y el ajuste fino de parámetros específicos (explotación)

En contraste, la búsqueda de parámetros en los algoritmos metaheurísticos se realiza de manera más tradicional, donde cada algoritmo implementa su propia estrategia de exploración del espacio de hiperparámetros sin depender de herramientas de automatización externas.

Esta combinación es especialmente poderosa para la detección de noticias falsas, donde la optimización automática puede significar la diferencia entre un modelo que apenas funciona y uno que detecta desinformación con alta precisión, aprovechando al máximo las capacidades semánticas de los transformers pre-entrenados.

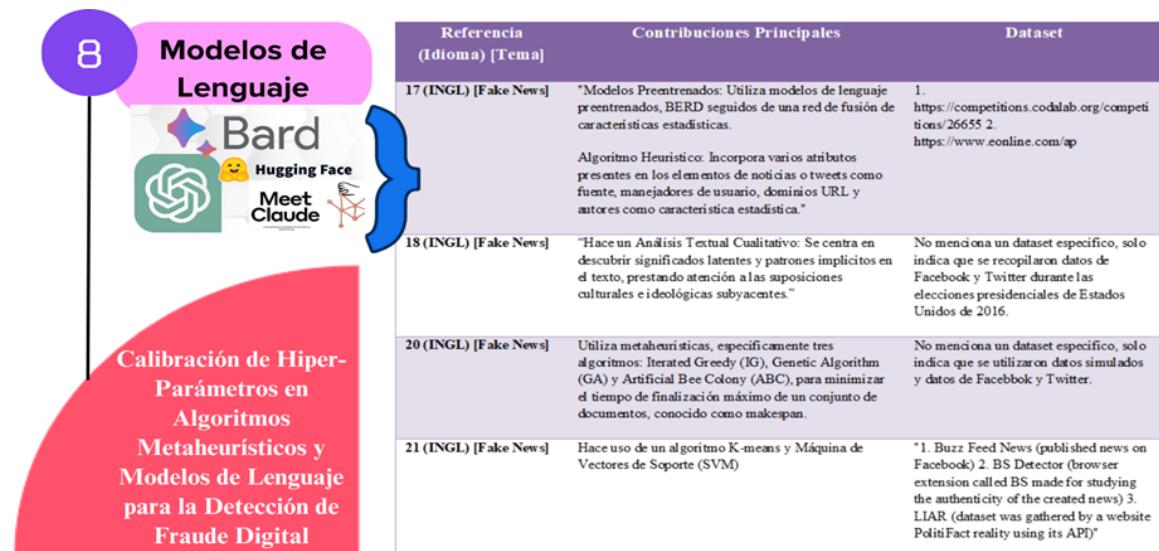


Figura 3.4: Mapa Conceptual 6: Artículos relacionados que incorporan Modelos de Lenguaje.

3.12. Síntesis y Perspectivas Futuras

La revisión comprehensiva de la literatura revela una evolución clara en las aproximaciones para la detección de noticias falsas y fraude digital. Desde los primeros enfoques basados en características lingüísticas hasta los modernos sistemas basados en transformers y LLMs, el campo ha experimentado avances significativos en términos de precisión y escalabilidad.

Los desafíos identificados incluyen: (1) la necesidad de mayor cantidad de datos etiquetados en español, (2) la adaptación de modelos a contextos culturales específicos, (3) la optimización eficiente de hiperparámetros en modelos complejos, y (4) la integración de información multimodal y conocimiento externo. Esta tesis contribuye especialmente a los puntos (1) y (3) mediante la unificación de corpus existentes y la aplicación de técnicas metaheurísticas para la optimización de modelos de detección.

La organización temática presentada en este capítulo proporciona un marco comprehensivo para entender las diferentes dimensiones del problema y justifica la aproximación metodológica híbrida adoptada en esta investigación, combinando modelos de lenguaje state-of-the-art con técnicas de optimización metaheurística.

Capítulo 4

Metodología

En este capítulo se detalla el procedimiento experimental seguido para abordar el problema de la detección de noticias falsas en español. La metodología se diseñó con un enfoque evolutivo y comparativo, comenzando con una exploración de técnicas de optimización clásicas y culminando con la implementación de un modelo de lenguaje de última generación. El diseño del flujo de trabajo, ilustrado en la Figura 4.1, garantiza la reproducibilidad de los resultados al definir claramente cada una de sus etapas, desde la recopilación de datos hasta la evaluación final y la implementación de una aplicación web como solución práctica.

4.1. Visión General de la Metodología

La estrategia metodológica de esta tesis se fundamenta en la comparación sistemática de dos paradigmas de la inteligencia artificial sobre un corpus unificado de gran escala. Como se muestra en la Figura 4.1, ambos enfoques parten de una base común (la problemática y los datos), pero siguen rutas de implementación y modelado distintas, para finalmente ser evaluados bajo un mismo marco de métricas y así determinar la solución más eficaz que será implementada en una aplicación web.

El primer paradigma se basa en técnicas clásicas de Procesamiento del Lenguaje Natural (PLN) con representación Bolsa de Palabras (BoW) y optimización metaheurística, mientras que el segundo emplea modelos Transformer pre-entrenados con técnicas de fine-tuning. Esta aproximación comparativa permite evaluar tanto la efectividad como la eficiencia computacional de cada enfoque en el contexto específico de la detección de desinformación en español.

Una vez determinado el mejor modelo mediante las métricas de evaluación correspondientes, se procederá a desarrollar una aplicación web funcional que permita la detección automática de noticias falsas en tiempo real.

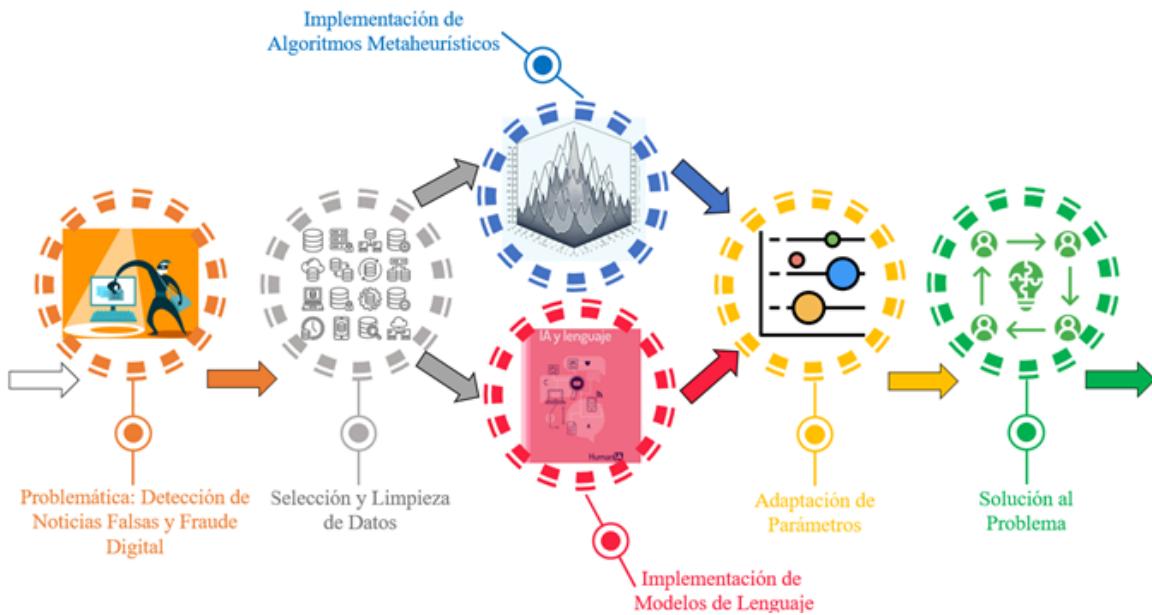


Figura 4.1: Metodología propuesta que aborda la problemática combinando Algoritmos Metaheurísticos y Modelos de Lenguaje.

4.2. Definición y Distinción de Conceptos Fundamentales

Antes de proceder con la descripción metodológica, es fundamental establecer con precisión la terminología utilizada en esta investigación, particularmente la distinción entre conceptos relacionados pero diferenciados que a menudo se utilizan de manera intercambiable en la literatura.

4.2.1. Noticia Falsa vs. Bulo: Una Distinción Crítica

En el contexto de esta investigación, es esencial distinguir entre dos conceptos fundamentales que, aunque relacionados, poseen características distintivas importantes para el desarrollo de sistemas de detección automática.

4.2.2. Taxonomía de la Desinformación

Para proporcionar un marco conceptual completo, esta investigación adopta la taxonomía establecida por la literatura internacional [1, 17], que clasifica el contenido engañoso en tres categorías principales:

1. **Desinformación**: Información falsa creada y compartida deliberadamente con intención maliciosa

Aspecto	Noticia Falsa	Bulo
Definición	Información deliberadamente fabricada que se presenta como noticia legítima, pero contiene datos falsos, inexacts o engañosos	Información falsa que circula ampliamente, especialmente en redes sociales, con propósito de engañar a la audiencia
Formato	Apariencia de contenido periodístico real, formato de noticia tradicional con titular, cuerpo, fecha, fuente	No necesariamente formato periodístico, puede ser texto libre, memes, cadenas de WhatsApp
Intención	Intención maliciosa de desinformar simulando legitimidad periodística	Propósito de engañar que apela más a emociones que a hechos verificables
Propagación	Se distribuye a través de sitios web que simulan medios legítimos o plataformas de noticias	Circulación viral en redes sociales, aplicaciones de mensajería, cadenas de reenvío
Ejemplos	Artículos con titular sensacionalista, byline falso, citas inventadas, estadísticas manipuladas	Teorías conspirativas, información médica falsa, rumores políticos, cadenas de "urgente"
Relevancia para PLN	Estructura textual más predecible, patrones estilísticos identificables en el análisis automatizado	Variabilidad estructural mayor, requiere análisis semántico y contextual más sofisticado

Tabla 4.1: Comparación detallada entre noticia falsa y bulo en el contexto de detección automática.

2. **Desinformación (Misinformation):** Información incorrecta compartida sin intención maliciosa
3. **Información Maliciosa (Malinformation):** Información genuina compartida con intención de causar daño

El alcance de esta investigación se centra específicamente en la detección automática de **desinformación**, abarcando tanto noticias falsas como bulos, reconociendo que ambos tipos requieren estrategias de procesamiento de lenguaje natural diferenciadas pero complementarias.

4.2.3. Justificación de la Unificación Terminológica

En el contexto del desarrollo de sistemas automáticos de detección, la distinción entre noticia falsa y bulo, aunque conceptualmente importante, se aborda mediante un enfoque unificado de **detección de contenido desinformativo**. Esta decisión metodológica se fundamenta en:

- **Características textuales compartidas:** Ambos tipos utilizan patrones lingüísticos identificables mediante técnicas de PLN
- **Objetivo común:** La finalidad de engañar o desinformar a la audiencia
- **Impacto social similar:** Efectos negativos en la percepción pública y toma de decisiones
- **Necesidad práctica:** Los sistemas de detección automática deben ser capaces de identificar ambos tipos

Por tanto, en el resto de este documento, el término “**noticias falsas**” se utilizará de manera inclusiva para referirse a cualquier tipo de contenido desinformativo, reconociendo la diversidad de formatos y estrategias de propagación que abarca esta categorización.

4.3. Construcción del Corpus Unificado

El primer y más fundamental paso de la metodología fue la construcción de un corpus de alta calidad y de tamaño significativo en español, dada la escasez documentada de recursos centralizados para esta tarea [17].

4.3.1. Fuentes de Datos Académicas

Se llevó a cabo un proceso exhaustivo de investigación y unificación de cuatro corpus académicos reconocidos, los cuales constituyen pilares en la investigación de noticias falsas en español. La Tabla 5.1 presenta un resumen detallado de estos recursos.

ID	Nombre del Corpus	Autores Principales	Año	Noticias	Características	Ref.
1	Spanish Fake News Corpus (IberLEF)	Posadas-Durán, J.P. Gómez-Adorno, H.	2019-2021	971	Análisis estilométrico Múltiples versiones	[12]
2	Dataset Zules Acosta (UPM)	Acosta, F.A.Z.	2019	598	Trabajo fin de máster Web scraping verificado	[11]
3	Dataset Tretiakov (Kaggle)	Tretiakov, A. Martín García, A.	2022	1,958	Machine Learning Disponible públicamente	[18]
4	Spanish Political Fake News Dataset	Blanco-Fernández, Y. Otero-Vizoso, J.	2024	57,231	Temática política Modelos BERT/RoBERTa	[19]
TOTAL CORPUS ACADÉMICOS					60,758	Cuatro fuentes

Tabla 4.2: Corpus académicos utilizados para la construcción del dataset unificado.

Spanish Fake News Corpus (IberLEF)

Este corpus, asociado a las competencias de IberLEF y desarrollado por Posadas-Durán, Gómez-Adorno, et al., ha tenido varias versiones. La versión original y más citada del corpus contiene un total de 971 noticias [12]. Este corpus se caracteriza por su enfoque en análisis estilométrico y ha sido utilizado en múltiples evaluaciones comparativas dentro del marco de las competencias IberLEF [43, 41]. Las versiones posteriores del corpus han sido mantenidas y actualizadas en repositorios públicos [44].

Dataset de Zules Acosta (Universidad Politécnica de Madrid)

Desarrollado como parte de un Trabajo Fin de Máster Universitario en Ciberseguridad en la Universidad Politécnica de Madrid, este corpus contiene 598 noticias verificadas [11]. El dataset fue construido mediante técnicas de web scraping y se encuentra disponible públicamente en la plataforma Kaggle [72, 73]. Su contribución principal radica en la aplicación de metodologías rigurosas de verificación para la construcción de datasets de noticias falsas.

Dataset de Tretiakov et al.

Este corpus, desarrollado por Tretiakov, Martín García y Camacho, contiene 1,958 noticias falsas en español [18]. El dataset fue creado en el contexto de técnicas de

aprendizaje automático para la detección de información falsa y se encuentra disponible públicamente en Kaggle [74]. Su enfoque se centra en la aplicación de técnicas de machine learning para la clasificación automatizada de contenido desinformativo.

Spanish Political Fake News Dataset

El corpus más extenso utilizado, desarrollado por Blanco-Fernández et al., contiene 57,231 noticias de temática política [19]. Este dataset fue específicamente diseñado para evaluar modelos BERT y RoBERTa en la detección de desinformación política en español y se encuentra disponible en Kaggle [75]. Su gran tamaño y enfoque en contenido político lo convierte en un recurso valioso para el entrenamiento de modelos de gran escala.

4.3.2. Proceso de Unificación y Estandarización

La integración de corpus heterogéneos requirió un proceso sistemático de estandarización que incluyó:

1. **Normalización de formato:** Unificación de esquemas de etiquetado en un sistema binario consistente (FALSO=0, REAL=1) y estandarización de estructuras de datos CSV con separador de punto y coma
2. **Eliminación de duplicados:** Implementación de algoritmos de detección de contenido similar basados en hash de texto y comparación de títulos
3. **Validación de calidad:** Verificación manual de una muestra estadísticamente significativa para confirmar la calidad del etiquetado y consistencia temática
4. **Balanceado de clases:** Análisis detallado de la distribución de clases para identificar necesidades de ampliación y garantizar equilibrio
5. **Limpieza de datos:** Eliminación de registros con valores nulos o inconsistentes mediante `dropna()` y validación de tipos de datos

4.3.3. Ampliación del Corpus Mediante Web Scraping

Para mejorar el balance del conjunto de datos y aumentar la diversidad de las noticias etiquetadas como "FALSAS", se aplicaron técnicas de web scraping automatizado. Se desarrolló un script especializado en Python utilizando las librerías BeautifulSoup y Requests para extraer de forma sistemática los titulares y cuerpos de noticia del portal de contenido satírico ^{El} Deforma".

Proceso de Web Scraping Automatizado

El proceso de extracción automatizada se ejecutó en múltiples fases para alcanzar el objetivo de 9,000 noticias adicionales. La Tabla 4.3 detalla las diferentes etapas implementadas.

Fase	Estrategia	Técnica Utilizada	Objetivo	Resultado	Observaciones
1	Scraping Inicial	Navegación por enlaces Extracción de contenido	1,000	1,000	Proceso exitoso Base establecida
2	Búsqueda Expandida Masiva	URLs desde existentes Crawling híbrido	9,000	2,495	Expansión limitada Nuevas estrategias
3	Crawler Híbrido Persistente	URLs semilla Trafilatura	9,000	2,495	Estabilización Mismo resultado
4	Paginación Sistemática	Escaneo por páginas Indexación completa	9,000	9,000	Éxito completo Objetivo alcanzado
TOTAL EXTRAÍDO			9,000	Web scraping completado	

Tabla 4.3: Fases del proceso de web scraping implementado para ".El Deforma".

Implementación Técnica del Web Scraping

El proceso de extracción automatizada incluyó:

1. **Identificación de patrones:** Análisis de la estructura HTML del sitio web objetivo para identificar selectores CSS consistentes (`h1.tdb-title-text, div.tdb-block-inner`)
2. **Extracción automatizada:** Implementación de rutinas de scraping con manejo de errores y delays aleatorios (1.5-4.5 segundos) para evitar sobrecarga del servidor
3. **Validación de contenido:** Verificación automática de la calidad del contenido extraído mediante filtros de longitud mínima (500 caracteres)
4. **Limpieza de texto:** Eliminación de disclaimers específicos del sitio, caracteres especiales y normalización de espacios mediante expresiones regulares
5. **Etiquetado automático:** Asignación de etiqueta "FALSO"(0) a todo el contenido extraído del portal satírico
6. **Persistencia progresiva:** Guardado en lotes de 50 registros con formato CSV y separador de punto y coma para evitar pérdida de datos
7. **Gestión de estado:** Implementación de archivos de progreso para permitir la reanudación del proceso en caso de interrupción

La Tabla 4.4 presenta las referencias bibliográficas completas de todos los corpus utilizados, organizadas por tipo de fuente.

La estrategia final exitosa utilizó paginación sistemática, escaneando secuencialmente desde <https://eldeforma.com/page/1/> hasta alcanzar las 9,000 noticias objetivo, garantizando cobertura completa del contenido disponible.

Esta estrategia de ampliación se justifica por varios factores:

- **Diversidad estilística:** Incorporación de diferentes estilos de contenido falso o satírico
- **Volumen de datos:** Incremento significativo del conjunto de entrenamiento

Corpus	Tipo de Referencia	Fuente Principal	Referencia
Spanish Fake News Corpus (IberLEF)	Artículo original	Journal of Intelligent and Fuzzy Systems	[12]
Spanish Fake News Corpus (IberLEF)	Workshop IberLEF 2021	Procesamiento del Lenguaje Natural	[43]
Spanish Fake News Corpus (IberLEF)	Workshop IberLEF 2020	IberLEF Workshop Proceedings	[41]
Spanish Fake News Corpus (IberLEF)	Repositorio GitHub	GitHub Repository	[44]
Dataset Zules Acosta (UPM)	Tesis de Máster	Universidad Politécnica de Madrid	[11]
Dataset Zules Acosta (UPM)	Dataset Kaggle	Kaggle Platform	[73]
Dataset Tretiakov (Kaggle)	Capítulo de libro	Springer Book Chapter	[18]
Dataset Tretiakov (Kaggle)	Dataset Kaggle	Kaggle Platform	[74]
Spanish Political Fake News Dataset	Artículo científico	Applied Sciences Journal	[19]
Spanish Political Fake News Dataset	Dataset Kaggle	Kaggle Platform	[75]

Tabla 4.4: Referencias bibliográficas completas de los corpus académicos utilizados.

- **Actualidad temporal:** Inclusión de contenido contemporáneo que refleja tendencias actuales
- **Variabilidad temática:** Ampliación del espectro de temas cubiertos en el corpus

4.3.4. Corpus Final y Estrategia de División

El proceso completo de unificación, limpieza de duplicados y ampliación mediante web scraping resultó en un **corpus final con 61,674 noticias únicas**, constituyendo uno de los recursos más extensos disponibles para la detección de noticias falsas en español.

Análisis de Composición Final

La Tabla 4.5 presenta la composición detallada del corpus unificado final.

Fuente de Datos	Noticias	Porcentaje	Tipo	Estado
Corpus Académicos Unificados	52,689	85.4 %	Mixto	Procesado
Web Scraping "El Deiforma"	9,000	14.6 %	Falso	Extraído
Duplicados Eliminados	-15	-0.02 %	–	Removido
CORPUS FINAL	61,674	100 %	Balanceado	Listo

Tabla 4.5: Composición final del corpus unificado después del procesamiento completo.

El análisis de balance del corpus final mostró una distribución equilibrada:

- **Noticias FALSAS (0):** 30,734 registros (49.8 %)
- **Noticias REALES (1):** 30,940 registros (50.2 %)

División Estratificada para Entrenamiento

Para garantizar una evaluación robusta y evitar el sobreajuste, este corpus se dividió de manera estratificada, manteniendo la proporción original de noticias falsas y reales en cada subconjunto. La configuración principal utilizada fue 70 % para entrenamiento, 10 % para validación y 20 % para evaluación final. Sin embargo, con el

objetivo de analizar la sensibilidad del modelo ante diferentes proporciones de datos, también se realizaron experimentos adicionales con las siguientes divisiones alternativas:

- 80 % entrenamiento / 10 % validación / 10 % prueba
- 60 % entrenamiento / 20 % validación / 20 % prueba

La Tabla 4.6 muestra la distribución principal implementada en la mayoría de los experimentos.

Conjunto de Datos	Porcentaje	Noticias	Propósito
Entrenamiento	70 %	43,171	Entrenar modelos
Validación	10 %	6,167	Calibrar hiperparámetros
Pruebas	20 %	12,336	Evaluación final
TOTAL	100 %	61,674	Metodología completa

Tabla 4.6: División estratificada principal del corpus para entrenamiento y evaluación.

Esta estrategia sigue las mejores prácticas establecidas en la literatura de aprendizaje automático y permite una evaluación imparcial de ambas metodologías. La estratificación garantiza que cada subconjunto mantenga la misma proporción de noticias falsas y reales que el corpus completo, evitando sesgos en el entrenamiento y evaluación de los modelos.

Proceso de Limpieza Final

El proceso de limpieza final implementó las siguientes operaciones:

1. **Eliminación de valores nulos:** Remoción de registros con campos vacíos en texto o etiquetas mediante `dropna()`
2. **Detección de duplicados:** Identificación y eliminación de 15 registros duplicados basada en contenido textual idéntico
3. **Normalización de caracteres:** Eliminación de punto y coma y caracteres especiales para evitar conflictos con el formato CSV
4. **Compactación de espacios:** Reducción de espacios múltiples y normalización de saltos de línea mediante expresiones regulares
5. **Mezclado aleatorio:** Randomización del orden con semilla fija (`random_state=42`) para garantizar reproducibilidad
6. **Validación de tipos:** Conversión de etiquetas a enteros mediante `astype(int)` para consistencia de datos

El resultado final fue un corpus robusto, balanceado y limpio, optimizado para el entrenamiento de modelos de detección de noticias falsas en español, que constituye una de las contribuciones principales de este trabajo de investigación al proporcionar un recurso de gran escala para la comunidad hispanohablante.

4.4. Enfoque 1: Detección Mediante Algoritmos Metaheurísticos

El primer enfoque metodológico se basó en técnicas clásicas de PLN combinadas con algoritmos de optimización metaheurística, sirviendo como línea base robusta para el proyecto y permitiendo la comparación con enfoques más modernos.

4.4.1. Preprocesamiento y Representación Textual

El pipeline de preprocesamiento implementó una serie de transformaciones estándar para convertir el texto crudo en representaciones numéricas adecuadas para algoritmos de aprendizaje automático.

Función de Limpieza de Texto

Se implementó una función optimizada de limpieza que incluye los siguientes pasos:

1. **Validación de entrada:** Verificación de que el input sea de tipo string válido
2. **Normalización a minúsculas:** Conversión completa del texto usando `lower()`
3. **Eliminación de URLs:** Remoción de enlaces web mediante expresiones regulares
4. **Limpieza de caracteres especiales:** Preservación únicamente de caracteres alfabéticos en español
5. **Tokenización con NLTK:** División del texto usando `word_tokenize()`
6. **Eliminación de stopwords:** Exclusión de palabras funcionales sin valor semántico
7. **Filtrado por longitud:** Remoción de tokens menores a 3 caracteres

Representación Bolsa de Palabras con Ponderación TF-IDF

Tras el preprocesamiento, se aplicó la técnica de Bolsa de Palabras (BoW) con ponderación TF-IDF utilizando `TfidfVectorizer` de scikit-learn. La configuración específica incluyó:

- **Máximo de características:** 5,000 palabras más frecuentes
- **Matriz resultante:** Representación densa convertida con `toarray()`
- **Almacenamiento:** Guardado en formato CSV para reutilización

La ponderación TF-IDF se calculó utilizando las siguientes fórmulas:

$$\text{TF}(t, d) = \frac{f_{t,d}}{\sum_{t' \in d} f_{t',d}} \quad (4.1)$$

$$\text{IDF}(t, D) = \log \frac{|D|}{|\{d \in D : t \in d\}|} \quad (4.2)$$

$$\text{TF-IDF}(t, d, D) = \text{TF}(t, d) \times \text{IDF}(t, D) \quad (4.3)$$

Donde t representa un término, d un documento, D la colección completa de documentos, y $f_{t,d}$ la frecuencia del término t en el documento d .

4.4.2. Algoritmos Metaheurísticos Implementados

Para la optimización de hiperparámetros de clasificadores, se implementaron y evaluaron cinco algoritmos metaheurísticos específicos [34, 33]. La Tabla 4.7 presenta una visión general de los algoritmos seleccionados y sus características principales.

Algoritmo	Inspiración	Principio Base	Estrategia Principal	Ref.
MSA	Metalurgia	Recocido de metales Enfriamiento controlado	Múltiples puntos de inicio Aceptación probabilística	[67, 68]
SS	Metodología científica	Búsqueda sistemática Combinación estructurada	Conjunto de referencia Generación de subconjuntos	[69]
VNS	Exploración geográfica	Cambio sistemático de vecindarios	Exploración local Diversificación estructural	[70]
GA	Evolución natural	Selección natural Supervivencia del más apto	Operadores genéticos Evolución poblacional	[64]
PSO	Comportamiento social	Inteligencia de enjambre Comunicación colectiva	Movimiento de partículas Intercambio de información	[65, 66]

Tabla 4.7: Algoritmos metaheurísticos implementados y sus fundamentos conceptuales.

Recocido Multiarranque (Multi-Start Simulated Annealing - MSA)

El algoritmo MSA combina los principios del Recocido Simulado [67] con estrategias de métodos Multi-arranque [68]. Se inspira en el proceso metalúrgico de recocido, donde los metales se calientan y luego se enfrian lentamente para alcanzar estados de menor energía y mayor estabilidad estructural. En el contexto de optimización, este proceso se traduce en la capacidad de escapar de óptimos locales mediante la aceptación probabilística de soluciones temporalmente peores.

El algoritmo MSA implementado incluye las siguientes características:

- **Función de aceptación:** $P = \exp(\Delta E/T)$ donde ΔE es la diferencia de evaluación
- **Esquema de enfriamiento:** Geométrico con $T_{n+1} = \alpha \times T_n$

Parámetro	Valor	Descripción
Temperatura inicial (TI)	1000	Energía inicial alta para exploración amplia
Temperatura final (TF)	1	Estado de convergencia con poca aleatoriedad
Factor de enfriamiento (α)	0.8	Tasa de reducción geométrica de temperatura
Pasos por temperatura	100	Iteraciones en cada nivel térmico
Puntos de arranque múltiples	5	Inicializaciones independientes

Tabla 4.8: Configuración de parámetros del algoritmo MSA.

- **Estrategia multiarranque:** Múltiples ejecuciones independientes para mayor robustez

Búsqueda Dispersa (Scatter Search - SS)

La Búsqueda Dispersa, desarrollada por Glover [69], se basa en principios de metodología científica, combinando elementos de diversas soluciones de alta calidad para generar nuevas candidatas. Su filosofía se centra en la combinación sistemática y la mejora estructurada, similar a como los investigadores combinan diferentes enfoques para obtener mejores resultados.

Parámetro	Valor	Descripción
Tamaño de población (P)	50	Población inicial diversa para exploración
Conjunto de referencia (b)	5	Mejores soluciones seleccionadas
Máximo de iteraciones	10	Ciclos de mejora y actualización
Combinaciones por iteración	10	Nuevas soluciones generadas por ciclo

Tabla 4.9: Configuración de parámetros del algoritmo SS.

La implementación incorpora:

- **Método de combinación:** Cruce de un punto con índice aleatorio
- **Estrategia de mejora:** Mutación en posición aleatoria
- **Actualización de RefSet:** Conservación de las mejores soluciones ordenadas por fitness

Algoritmo Genético (Genetic Algorithm - GA)

El Algoritmo Genético, fundamentado en el trabajo seminal de Holland [64], emula los procesos de evolución natural descritos por Darwin, donde las especies más aptas tienen mayor probabilidad de sobrevivir y reproducirse. En optimización, este principio se traduce en la evolución de poblaciones de soluciones mediante operadores inspirados en la genética: selección, cruce y mutación.

Los operadores genéticos implementados incluyen:

- **Selección por torneo determinística:** Competencia entre individuos para reproducción

Parámetro	Valor	Descripción
Número de generaciones	20	Ciclos evolutivos completos
Tamaño de población	50	Individuos por generación
Tasa de mutación	0.1	Probabilidad de modificación genética
Tamaño de torneo	3	Individuos competidores en selección

Tabla 4.10: Configuración de parámetros del algoritmo GA.

- **Cruce de un punto:** Intercambio genético con hijos complementarios
- **Mutación uniforme:** Modificación aleatoria condicionada por tasa

Búsqueda en Vecindades Variables (Variable Neighborhood Search - VNS)

VNS, introducida por Mladenović y Hansen [70], se inspira en la exploración geográfica sistemática, donde se cambian las estrategias de búsqueda (vecindarios) de manera estructurada para evitar quedar atrapado en regiones subóptimas. Su principio fundamental es que diferentes estructuras de vecindario pueden revelar diferentes aspectos del paisaje de optimización.

Parámetro	Valor	Descripción
Máximo de iteraciones	20	Ciclos de búsqueda completos
Vecindarios máximos (k_{max})	5	Estructuras de vecindad diferentes
Estrategia de vecindario	k elementos aleatorios	Modificación de k componentes simultáneas

Tabla 4.11: Configuración de parámetros del algoritmo VNS.

Las características de implementación incluyen:

- **Criterio de mejora:** Aceptación únicamente de soluciones superiores
- **Reinicio de vecindarios:** Retorno a $k=1$ tras mejora encontrada
- **Diversificación sistemática:** Exploración progresiva de vecindarios más amplios

Optimización por Enjambre de Partículas (Particle Swarm Optimization - PSO)

PSO, desarrollado por Kennedy y Eberhart [65] y Eberhart y Kennedy [66], se fundamenta en el comportamiento social observado en enjambres naturales como bandadas de aves o cardúmenes de peces. Las partículas (soluciones) se mueven en el espacio de búsqueda influenciadas por su propia experiencia y la información colectiva del enjambre, creando un mecanismo de inteligencia emergente.

El comportamiento del enjambre se rige por:

- **Inicialización:** Posiciones aleatorias en $[-5, 5]$ y velocidades gaussianas

Parámetro	Valor	Descripción
Número de partículas	30	Agentes en el enjambre
Máximo de iteraciones	20	Movimientos del enjambre
Factor de inercia (w)	0.5	Influencia del movimiento previo
Coeficiente cognitivo (c1)	1.5	Peso de la experiencia personal
Coeficiente social (c2)	1.5	Peso de la experiencia colectiva

Tabla 4.12: Configuración de parámetros del algoritmo PSO.

- **Actualización de velocidad:** $v_{t+1} = w \cdot v_t + c_1 \cdot r_1 \cdot (p_{best} - x_t) + c_2 \cdot r_2 \cdot (g_{best} - x_t)$
- **Comunicación social:** Intercambio de información sobre mejores posiciones encontradas

4.4.3. Función de Evaluación y Clasificación

Todos los algoritmos metaheurísticos utilizan una función de evaluación común que implementa un clasificador logístico binario:

$$P(y = 1|x) = \frac{1}{1 + \exp(-w^T \cdot x_{binario})} \quad (4.4)$$

Donde:

- w representa el vector de pesos optimizado
- $x_{binario}$ es la versión binarizada de las características usando umbrales
- La decisión final se toma con umbral de 0.5

4.4.4. Reducción de Dimensionalidad

Para hacer computacionalmente factible la optimización, se aplicó reducción de dimensionalidad usando `SelectPercentile` con selección del 10 % de las características más relevantes según el test chi-cuadrado. La Tabla 4.13 resume el proceso de selección de características.

Aspecto	Configuración	Justificación
Método de selección	SelectPercentile	Selección basada en estadísticas univariadas
Porcentaje seleccionado	10 %	Balance entre información y eficiencia
Test estadístico	Chi-cuadrado	Adecuado para clasificación binaria
Características originales	5,000	Vocabulario TF-IDF completo
Características reducidas	500	Dimensionalidad manejable

Tabla 4.13: Configuración del proceso de reducción de dimensionalidad.

Esta estrategia de reducción permite que los algoritmos metaheurísticos operen eficientemente sobre un espacio de características optimizado, manteniendo la información más discriminativa para la tarea de clasificación.

4.5. Enfoque 2: Detección Mediante Modelo Transformer

El segundo enfoque se fundamentó en el paradigma de aprendizaje profundo, específicamente en la arquitectura Transformer [25], utilizando un modelo de lenguaje pre-entrenado para capturar representaciones contextuales sofisticadas del texto.

4.5.1. Selección y Justificación del Modelo

Para la selección del modelo óptimo se realizó un análisis comparativo entre los principales modelos BERT optimizados disponibles. La Tabla 4.14 presenta las características técnicas de los candidatos evaluados.

Modelo	Parámetros	Capas	Dimensión	Soporte Español	Reducción vs BERT
BERT-base-multilingual	110M	12	768	Sí	– (Referencia)
DistilBERT-multilingual	66M	6	768	Sí	40 % parámetros
TinyBERT	14.5M	4	312	Limitado	87 % parámetros

Tabla 4.14: Comparación de modelos BERT optimizados para la tarea de clasificación.

Se seleccionó el modelo `distilbert-base-multilingual-cased` por las siguientes razones técnicas y prácticas:

- **Capacidad multilingüe:** Soporte nativo para español y más de 100 idiomas
- **Eficiencia computacional:** Reducción del 40 % en parámetros respecto a BERT base manteniendo el 97 % del rendimiento [20]
- **Arquitectura probada:** Basado en destilación de conocimiento de BERT [26]
- **Balance óptimo:** Mejor relación rendimiento-eficiencia que TinyBERT [27]
- **Disponibilidad:** Accesible a través de la librería Transformers de Hugging Face

4.5.2. Infraestructura Computacional

El entrenamiento del modelo DistilBERT se realizó en un entorno de hardware dedicado con las siguientes especificaciones técnicas:

4.5.3. Configuración Experimental y Optimización de Hiperparámetros

El proceso de fine-tuning se implementó utilizando TensorFlow con precisión mixta (`mixed_float16`) para optimizar el uso de memoria GPU y acelerar el entrenamiento. La configuración experimental se estructuró en múltiples fases de optimización.

Componente	Especificación	Características Relevantes
Procesador	AMD Ryzen 7 7735H	8 núcleos, 16 hilos 4.75 GHz boost Arquitectura Zen 3+ (6nm)
GPU	NVIDIA GeForce RTX 4060	8GB GDDR6 VRAM 3072 núcleos CUDA 140W TGP Soporte mixed precision
Memoria RAM	16GB DDR5	Capacidad para datasets grandes Procesamiento de batches
Almacenamiento	x2 500GB SSD NVMe	Acceso rápido a datos Checkpointing eficiente

Tabla 4.15: Especificaciones del hardware utilizado para entrenamiento de DistilBERT.

Parámetros de Entrenamiento Base

La configuración base del modelo se estableció considerando las limitaciones computacionales y las mejores prácticas para modelos Transformer:

Parámetro	Valor	Justificación
Longitud máxima de secuencia	128 tokens	Reducido desde 512 Minimizar overfitting
Épocas máximas	30	Suficiente para convergencia
Paciencia early stopping	8 épocas	Evitar sobreentrenamiento
Factor de reducción LR	0.15	Más agresivo que estándar Convergencia controlada
Precisión numérica	mixed_float16	Optimización de memoria GPU

Tabla 4.16: Configuración de parámetros base para el entrenamiento de DistilBERT.

Estrategia de Regularización Avanzada

Para controlar el overfitting identificado en experimentos preliminares, se implementó una estrategia de regularización intensiva que incluye múltiples técnicas complementarias:

Técnica	Rango Explorado	Valor Óptimo	Propósito
Learning Rate Ultra-bajo	[8e-7, 5e-6]	2e-06	Convergencia controlada
Dropout Agresivo	[0.4, 0.7]	0.7	Prevención de co-adaptación
Regularización L2	[0.05, 0.5]	0.05	Control de pesos
Weight Decay Manual	0.02 fijo	0.02	Aplicado por batch
Noise Injection	[0.01, 0.03]	0.03	Alternativa a label smoothing
Batch Size Variable	{4, 6, 8}	4	Mayor regularización

Tabla 4.17: Estrategias de regularización implementadas para controlar overfitting.

Proceso de Búsqueda de Hiperparámetros

La optimización de hiperparámetros se realizó mediante una búsqueda sistemática que combinó exploración manual y búsqueda en cuadrícula (grid search) en los rangos especificados. El proceso se estructuró en las siguientes etapas:

1. **Búsqueda inicial:** Exploración amplia de rangos para identificar regiones prometedoras
2. **Refinamiento:** Búsqueda focalizada en torno a los mejores candidatos iniciales
3. **Validación cruzada:** Confirmación de estabilidad con múltiples semillas aleatorias
4. **Selección final:** Evaluación en conjunto de validación para determinar configuración óptima

4.5.4. Arquitectura del Modelo de Clasificación

El modelo final implementa una arquitectura que combina las representaciones contextuales de DistilBERT con capas de clasificación especializadas:

Capa	Configuración	Función
DistilBERT Base	66M parámetros 6 capas transformer	Extracción de características
Pooling Layer	Global average pooling	Agregación de secuencia
Dropout	Rate = 0.7	Regularización
Dense Layer	128 unidades + ReLU	Representación intermedia
Output Layer	1 unidad + Sigmoid	Clasificación binaria

Tabla 4.18: Arquitectura del modelo de clasificación basado en DistilBERT.

4.5.5. Protocolo de Entrenamiento

El protocolo de entrenamiento se diseñó para maximizar la estabilidad y reproducibilidad de los resultados:

Configuración de Entrenamiento

- **Optimizador:** AdamW con weight decay integrado
- **Función de pérdida:** Binary crossentropy con smoothing
- **Métricas de monitoreo:** Accuracy y AUC para seguimiento durante entrenamiento

- **Scheduler:** ReduceLROnPlateau para ajuste dinámico del learning rate
- **Checkpointing:** Guardado automático del mejor modelo basado en validation loss

Estrategia de Validación

- **Early stopping:** Monitoreo de validation loss con paciencia de 8 épocas
- **Validation split:** 10 % del conjunto de entrenamiento reservado para validación
- **Estratificación:** Mantenimiento de proporciones de clases en validación
- **Seed fija:** random_state=42 para reproducibilidad

4.5.6. Consideraciones de Implementación

Optimizaciones de Memoria

Para manejar eficientemente los recursos computacionales disponibles:

- **Gradient accumulation:** Simulación de batches más grandes cuando necesario
- **Mixed precision:** Uso de float16 para reducir uso de VRAM
- **Batch size adaptativo:** Ajuste dinámico según disponibilidad de memoria
- **Gradient clipping:** Prevención de gradientes explosivos

Monitoreo y Logging

- **TensorBoard:** Visualización de métricas durante entrenamiento
- **Logging detallado:** Registro de hiperparámetros y métricas por época
- **Checkpoints automáticos:** Guardado periódico para recuperación
- **Profiling GPU:** Monitoreo de utilización de recursos

Esta configuración metodológica establece un marco robusto para el entrenamiento de DistilBERT, garantizando tanto la calidad de los resultados como la reproducibilidad del proceso experimental.

4.6. Metodología de Evaluación Comparativa

Para garantizar una comparación justa y rigurosa entre ambos paradigmas, se estableció un protocolo de evaluación común que eliminara sesgos metodológicos y permitiera una comparación objetiva del rendimiento.

4.6.1. Protocolo de Evaluación

Ambos enfoques fueron sometidos al mismo esquema de evaluación estructurado que garantiza la imparcialidad y reproducibilidad de los resultados. La Tabla 4.19 detalla la estructura del protocolo implementado.

Fase	Conjunto	Porcentaje	Propósito	Restricciones
Entrenamiento	Training	70 %	Aprendizaje de parámetros Ajuste de pesos del modelo	Exclusivo para entrenamiento Sin acceso a otros conjuntos
Validación	Validation	10 %	Calibración de hiperparámetros Selección de configuraciones	No utilizado en entrenamiento Guía para optimización
Evaluación Final	Test	20 %	Prueba objetiva Métricas de rendimiento	Completamente no visto Una sola evaluación final

Tabla 4.19: Protocolo de evaluación implementado para ambos paradigmas.

4.6.2. Fundamentos de la Matriz de Confusión

Para la evaluación de modelos de clasificación binaria en detección de noticias falsas, todas las métricas se derivan de la matriz de confusión, que categoriza las predicciones según su correspondencia con la realidad. La Tabla 4.20 define los cuatro casos posibles.

Categoría	Descripción	Interpretación en Noticias Falsas	Impacto
Verdaderos Positivos (TP)	Predicción: REAL Realidad: REAL	Contenido real correctamente identificado como real	Positivo Preserva información legítima
Verdaderos Negativos (TN)	Predicción: FALSO Realidad: FALSO	Contenido falso correctamente identificado como falso	Positivo Detecta desinformación
Falsos Positivos (FP)	Predicción: REAL Realidad: FALSO	Contenido falso incorrectamente clasificado como real	Negativo Permite propagación de falsedad
Falsos Negativos (FN)	Predicción: FALSO Realidad: REAL	Contenido real incorrectamente clasificado como falso	Negativo Censura información legítima

Tabla 4.20: Definición de categorías de la matriz de confusión en el contexto de detección de noticias falsas.

Representación Visual de la Matriz de Confusión

La estructura de la matriz de confusión para clasificación binaria se puede representar como:

		Predicción del Modelo	
		REAL	FALSO
Realidad	REAL	TP	FN
	FALSO	FP	TN

Tabla 4.21: Estructura de la matriz de confusión para clasificación binaria.

4.6.3. Marco de Métricas de Rendimiento

Utilizando las definiciones anteriores, el rendimiento se evaluó con un conjunto comprehensivo de métricas estándar. La Tabla 4.22 presenta la definición formal y el propósito de cada métrica implementada.

Métrica	Fórmula	Interpretación	Relevancia en Detección
Exactitud	$\frac{TP+TN}{TP+TN+FP+FN}$	Proporción total de predicciones correctas	Rendimiento general en datos balanceados
Precisión	$\frac{TP}{TP+FP}$	Confiabilidad de predicciones positivas	Minimizar falsas alarmas de contenido falso
Exhaustividad	$\frac{TP}{TP+FN}$	Capacidad de detectar casos positivos reales	Detectar todo contenido verdaderamente falso
F1-Score	$2 \times \frac{Precision \times Recall}{Precision + Recall}$	Balance entre precisión y exhaustividad	Métrica principal para comparación de modelos
Especificidad	$\frac{TN}{TN+FP}$	Capacidad de identificar casos negativos correctos	Evitar clasificar contenido real como falso

Tabla 4.22: Marco de métricas de evaluación para clasificación binaria de noticias falsas.

Consideraciones Específicas para Detección de Noticias Falsas

En el contexto de detección de noticias falsas, cada tipo de error tiene implicaciones específicas:

- **Falsos Positivos (FP):** Contenido real clasificado como falso - puede generar censura indebida y limitar la libertad de información
- **Falsos Negativos (FN):** Contenido falso no detectado - permite propagación de desinformación y daño social
- **Trade-off crítico:** Balance entre detectar desinformación vs. preservar libertad de información
- **Contexto de aplicación:** En sistemas automatizados, los FP pueden ser más tolerables que los FN si existe revisión humana posterior

4.6.4. Criterios de Selección del Mejor Modelo

La selección del modelo óptimo se basará en una evaluación multi-criterio que considera diferentes aspectos del rendimiento. La Tabla 4.23 detalla los criterios y sus pesos relativos.

4.6.5. Protocolo de Validación Cruzada

Para garantizar la robustez estadística de los resultados, se implementará un protocolo de validación adicional:

Criterio	Peso	Métrica Principal	Métricas Secundarias	Justificación
Rendimiento Predictivo	40%	F1-Score	Precision, Recall Accuracy, Specificity	Capacidad fundamental de clasificación
Estabilidad del Modelo	25%	Desviación estándar F1	Varianza en múltiples ejecuciones	Consistencia y confiabilidad
Eficiencia Computacional	20%	Tiempo de inferencia	Memoria utilizada Recursos GPU/CPU	Viabilidad práctica de implementación
Capacidad de Generalización	15%	Gap Training-Test	Diferencia entre rendimiento interno y externo	Robustez ante datos no vistos

Tabla 4.23: Criterios multi-dimensionales para selección del modelo óptimo.

Aspecto	Configuración	Propósito
Tipo de validación	Stratified K-Fold (k=5)	Mantener proporción de clases
Semillas aleatorias	Múltiples seeds (42, 123, 456, 789, 999)	Evaluar estabilidad estadística
Repeticiones	3 ejecuciones por configuración	Reducir variabilidad aleatoria
Análisis estadístico	Test de significancia (t-test pareado)	Validar diferencias entre modelos

Tabla 4.24: Protocolo de validación cruzada para robustez estadística.

4.6.6. Benchmarking Computacional

Para evaluar la eficiencia computacional, se estableció un protocolo de benchmarking estandarizado:

Métrica	Unidad	Método de Medición	Contexto de Evaluación
Tiempo de inferencia	milisegundos/muestra	Promedio de 1000 predicciones individuales	Simulación de uso en producción
Memoria RAM	MB	Pico de uso durante inferencia	Requisitos mínimos de hardware
Memoria GPU	MB	VRAM utilizada (si aplica)	Necesidades de aceleración GPU
Throughput	muestras/segundo	Procesamiento en lotes de diferentes tamaños	Escalabilidad del sistema

Tabla 4.25: Métricas de benchmarking computacional para evaluación de eficiencia.

4.6.7. Análisis de Matrices de Confusión

Se realizará un análisis detallado de las matrices de confusión para comprender el comportamiento específico de cada modelo:

- **Análisis por clase:** Identificación de sesgos hacia noticias reales o falsas
- **Análisis de errores:** Caracterización cualitativa de casos mal clasificados
- **Visualización:** Heatmaps normalizados para comparación visual
- **Interpretabilidad:** Análisis de características más influyentes en decisiones

Componente del Reporte	Contenido
Tabla de métricas principales	Accuracy, Precision, Recall, F1-Score, Specificity
Análisis de variabilidad	Media ± desviación estándar por métrica
Benchmarking de eficiencia	Tiempos de inferencia y uso de recursos
Matrices de confusión	Visualización normalizada y análisis de errores
Recomendación final	Modelo seleccionado con justificación integral

Tabla 4.26: Estructura del reporte de resultados comparativo.

4.6.8. Reporte de Resultados

Los resultados se presentarán siguiendo un formato estandarizado que incluye:

Esta metodología de evaluación comprensiva garantiza una comparación objetiva, estadísticamente robusta y prácticamente relevante entre los paradigmas de algoritmos metaheurísticos y modelos Transformer para la detección de noticias falsas en español.

4.7. Infraestructura Computacional y Herramientas

4.7.1. Entorno de Desarrollo para Algoritmos Metaheurísticos

El desarrollo de los algoritmos metaheurísticos se realizó en un entorno cloud optimizado para experimentación iterativa y flexibilidad de recursos. La Tabla 4.27 detalla las especificaciones del entorno utilizado.

Componente	Especificación	Ventajas para Metaheurísticos
Plataforma	Google Colab Pro	Acceso a recursos escalables Flexibilidad de configuración
Procesamiento	CPU Intel/AMD variable	Suficiente para algoritmos iterativos Paralelización básica
Memoria RAM	12-16 GB	Carga completa de datasets
Almacenamiento	Google Drive integrado	Persistencia entre sesiones Versionado de experimentos
GPU	No requerida	Algoritmos CPU-intensivos

Tabla 4.27: Especificaciones del entorno de desarrollo para algoritmos metaheurísticos.

4.7.2. Hardware Dedicado para Entrenamiento DistilBERT

Para el entrenamiento del modelo DistilBERT se utilizó un equipo de desarrollo dedicado con especificaciones de gaming adaptadas para deep learning. La Tabla 4.28 presenta las especificaciones detalladas del sistema.

Optimizaciones Específicas del Hardware

El hardware seleccionado permitió implementar las siguientes optimizaciones:

Componente	Modelo/Especificación	Características Técnicas	Relevancia para Deep Learning
Laptop	Machenike L16 Pro	Gaming laptop optimizada Sistema de refrigeración avanzado	Entrenamiento prolongado Estabilidad térmica
Procesador	AMD Ryzen 7 7735H	8 núcleos, 16 hilos Frecuencia base: 3.2 GHz Boost: hasta 4.75 GHz Arquitectura: Zen 3+ (6nm) TDP: 35-54W configurable	Paralelización de procesos Preprocesamiento de datos Gestión de memoria
GPU	NVIDIA GeForce RTX 4060	Arquitectura: Ada Lovelace 8GB GDDR6 VRAM 3072 núcleos CUDA 140W TGP (laptop) Ray Tracing 3ra gen DLSS 3 support	Aceleración CUDA Mixed precision (FP16) Tensor cores para transformers Memoria suficiente para DistilBERT
Memoria RAM	16GB DDR5	Velocidad: 4800 MHz Latencia optimizada Dual channel	Carga de datasets grandes Batching eficiente Multitasking durante entrenamiento
Almacenamiento	1TB SSD NVMe	PCIe 4.0 interface Velocidades: 7000+ MB/s lectura Baja latencia de acceso	Carga rápida de datos Checkpointing eficiente Almacenamiento de modelos

Tabla 4.28: Especificaciones detalladas del hardware utilizado para entrenamiento de DistilBERT.

Optimización	Implementación	Beneficio Obtenido
Mixed Precision Training	FP16 con automatic scaling	Reducción 50% uso VRAM Aceleración 1.5-2x en entrenamiento
CUDA Memory Management	<code>tf.config.experimental.memory_growth</code>	Uso eficiente de 8GB VRAM
Tensor Cores Utilization	Dimensiones múltiples de 8	Aceleración automática en operaciones matriciales
Gradient Accumulation	Simulación de batch size mayor	Entrenamiento estable con memoria limitada

Tabla 4.29: Optimizaciones de hardware implementadas para maximizar eficiencia.

4.7.3. Stack Tecnológico Completo

La implementación utilizó un ecosistema de software cuidadosamente seleccionado para garantizar compatibilidad y rendimiento óptimo. La Tabla 4.30 detalla las herramientas y librerías utilizadas.

Categoría	Herramienta/Librería	Versión	Propósito Específico
Lenguaje Base	Python	3.9.x	Lenguaje principal Compatibilidad con ecosistema ML
Deep Learning	TensorFlow	2.15.0	Framework principal para DistilBERT Soporte GPU optimizado
Transformers	transformers (Hugging Face)	4.35.0	Modelos pre-entrenados Tokenización avanzada
ML Tradicional	scikit-learn	1.3.2	Algoritmos metaheurísticos Métricas de evaluación
Procesamiento	pandas	2.1.3	Manipulación de datasets
Cómputo Numérico	numpy	1.25.2	Operaciones matriciales Algoritmos de optimización
NLP	NLTK	3.8.1	Preprocesamiento de texto Tokenización y limpieza
Visualización	matplotlib / seaborn	3.8.1 / 0.12.2	Gráficos de resultados Matrices de confusión
Serialización	joblib	1.3.2	Guardado de modelos Persistencia de experimentos
Web Framework	Flask	2.3.3	API REST para aplicación final
Contenerización	Docker	24.0.x	Deployment reproducible Aislamiento de dependencias

Tabla 4.30: Stack tecnológico completo utilizado en el desarrollo del proyecto.

4.8. Consideraciones Éticas y de Privacidad

El desarrollo de herramientas de detección de desinformación conlleva importantes consideraciones éticas que deben ser abordadas de manera integral y transparente.

4.8.1. Marco Ético de Desarrollo

Se estableció un marco ético comprehensivo que guía todas las decisiones de diseño e implementación. La Tabla 4.31 presenta los principios fundamentales adoptados.

4.8.2. Limitaciones Declaradas y Uso Responsable

Se establecieron limitaciones claras y recomendaciones de uso responsable:

- **Alcance geográfico:** Optimizado para español, puede tener limitaciones en variantes regionales específicas
- **Contexto temporal:** Entrenado con datos hasta 2025, puede requerir actualización para tendencias futuras

Principio Ético	Implementación	Mecanismo de Control	Impacto Esperado
Transparencia	Código fuente abierto Metodología pública	Repositorio GitHub público Documentación completa	Auditoría independiente Reproducibilidad científica
Responsabilidad	Herramienta de apoyo No decisión final	Interfaz con disclaimers Scores de confianza	Uso responsable Juicio humano preservado
Privacidad	Procesamiento local No almacenamiento personal	Anonimización automática Logs temporales únicamente	Protección de datos Cumplimiento GDPR
Equidad	Datasets balanceados Evaluación multi-métrica	Ánalisis de sesgos Validación cruzada	Tratamiento imparcial Reducción de discriminación

Tabla 4.31: Marco ético implementado para el desarrollo responsable de la herramienta.

- **Dominios específicos:** Enfocado en noticias generales, puede tener menor precisión en contenido altamente técnico
- **Herramienta de apoyo:** Diseñado para asistir, no reemplazar el criterio editorial humano

4.9. Validación y Reproducibilidad

La reproducibilidad científica constituye un pilar fundamental de esta investigación, implementándose protocolos para garantizar que los resultados puedan ser verificados independientemente por la comunidad científica.

4.9.1. Ecosistema de Artefactos de Reproducibilidad

Para facilitar la reproducción completa del estudio y su extensión por parte de la comunidad científica, se desarrollará un ecosistema integral de artefactos que abarca desde el código fuente hasta modelos completamente entrenados listos para producción. La Tabla 4.32 detalla los componentes específicos que serán puestos a disposición pública.

4.9.2. Solución Lista para Producción

Como parte del compromiso con la transferencia tecnológica y la aplicabilidad práctica de los resultados, se entregará un repositorio completo con el mejor modelo identificado durante la evaluación comparativa, completamente optimizado y listo para ser desplegado en entornos de producción. Esta solución incluirá:

- **Modelo pre-entrenado optimizado:** El modelo con mejor rendimiento según las métricas de evaluación, con pesos finales y configuración optimizada
- **Intefaz web completa:** Interfaz web funcional desarrollada con Flask, incluyendo opciones para análisis de URLs y texto directo

Artefacto	Formato	Contenido	Disponibilidad
Código fuente completo	GitHub repository	Scripts de entrenamiento Algoritmos implementados Pipeline completo	Público (MIT License)
Datasets procesados	CSV + metadata	Corpus unificado División train/val/test Estadísticas descriptivas	Público (respetando licencias originales)
Modelos entrenados	joblib (metaheurísticos) SavedModel (DistilBERT)	Pesos optimizados Configuraciones Métricas de evaluación	Público (Hugging Face Hub)
Entorno de ejecución	Docker containers	Imagen completa Dependencias exactas Scripts de ejecución	Docker Hub público
Resultados experimentales	JSON + visualizaciones	Métricas detalladas Matrices de confusión Logs de entrenamiento	Repositorio principal Supplementary material

Tabla 4.32: Artefactos generados para facilitar la reproducibilidad completa del estudio.

- **Contenerización Docker:** Imagen Docker completa con todas las dependencias, configuraciones y el modelo integrado
- **Despliegue con un comando:** Script automatizado que permite poner en funcionamiento toda la aplicación ejecutando únicamente un comando

La estrategia de contenerización garantiza que la solución sea completamente portable y reproducible en cualquier entorno que soporte Docker, eliminando problemas de compatibilidad y dependencias.

Capítulo 5

Análisis de Resultados

En este capítulo se presenta el análisis comprehensivo de los resultados obtenidos mediante la aplicación de las dos metodologías desarrolladas para la detección de noticias falsas en español. Los resultados se organizan de manera progresiva, comenzando con la evaluación exhaustiva del enfoque basado en algoritmos metaheurísticos aplicados sobre representaciones de Bolsa de Palabras (BoW), seguido por los hallazgos del modelo Transformer DistilBERT, y culminando con una comparación integral entre ambas aproximaciones.

El análisis estadístico se fundamenta en las métricas de evaluación definidas en la metodología: Exactitud (Accuracy), Precisión (Precision), Exhaustividad (Recall), F1-Score y Especificidad. Para garantizar la robustez de los resultados, todos los experimentos se ejecutaron mediante validación cruzada estratificada y múltiples semillas aleatorias. Además, se emplearon distintas configuraciones de partición de datos para evaluar la estabilidad del modelo ante variaciones en el conjunto de entrenamiento, validación y prueba. Las divisiones utilizadas incluyeron: 80 % entrenamiento / 10 % validación / 10 % prueba; 70 % / 10 % / 20 %; y 60 % / 20 % / 20 %, respectivamente.

5.1. Resultados del Enfoque Metaheurístico con Representación BoW-TF-IDF

La primera fase de esta investigación se centró en el desarrollo y evaluación de un sistema de detección de noticias falsas basado en algoritmos metaheurísticos operando sobre representaciones tradicionales de texto. Este enfoque, que posteriormente fue formalizado y publicado como capítulo de libro [34], demostró la viabilidad de técnicas bio-inspiradas para la calibración automática de sistemas de detección, estableciendo una línea base sólida para la comparación posterior con modelos de lenguaje más avanzados.

5.1.1. Marco Experimental y Configuración Base

Características del Corpus Utilizado

Los experimentos se realizaron sobre el corpus académico unificado, que constituye la primera versión del dataset desarrollado en esta investigación, sin la inclusión posterior de datos obtenidos mediante web scraping.

ID	Nombre del Corpus	Autores Principales	Año	Noticias	Características	Ref.
1	Spanish Fake News Corpus (IberLEF)	Posadas-Durán, J.P. Gómez-Adorno, H.	2019-2021	971	Análisis estilométrico Múltiples versiones	[12]
2	Dataset Zules Acosta (UPM)	Acosta, F.A.Z.	2019	598	Trabajo fin de máster Web scraping verificado	[11]
3	Dataset Tretiakov (Kaggle)	Tretiakov, A. Martín García, A.	2022	1,958	Machine Learning Disponible públicamente	[18]
4	Spanish Political Fake News Dataset	Blanco-Fernández, Y. Otero-Vizoso, J.	2024	57,231	Temática política Modelos BERT/RoBERTa	[19]
TOTAL CORPUS ACADÉMICOS					60,758	Cuatro fuentes

Tabla 5.1: Corpus académicos utilizados para la construcción del dataset unificado.

Las características del corpus son las siguientes:

- **Conjunto de entrenamiento:** 42,151 registros (80 %)
- **Conjunto de validación:** 5,269 registros (10 %)
- **Conjunto de pruebas:** 5,269 registros (10 %)
- **Total de noticias:** 52,689 artículos de fuentes académicas verificadas
- **Distribución balanceada:** Aproximadamente 40 % noticias falsas y 60 % noticias reales

Configuración de Representación Textual

La representación textual se fundamentó en el paradigma clásico de Bolsa de Palabras con ponderación TF-IDF:

- **Vocabulario inicial:** 5,000 términos más frecuentes del corpus
- **Reducción de dimensionalidad:** Selección del 10 % más discriminativo (500 características)
- **Criterio de selección:** Test chi-cuadrado para identificar características más relevantes
- **Representación final:** Matriz dispersa de 500 dimensiones por documento

Arquitectura del Clasificador

Todos los algoritmos metaheurísticos optimizaron un clasificador logístico binario con las siguientes características:

- **Función de activación:** Sigmoid para clasificación binaria
- **Binarización adaptativa:** Umrales dinámicos para cada característica
- **Parámetros optimizados:** Selección de características, pesos del modelo y umbrales de decisión
- **Función objetivo:** Maximización de la exactitud en el conjunto de entrenamiento

5.1.2. Implementación y Configuración de Algoritmos Metaheurísticos

La Tabla 5.2 presenta la configuración detallada de parámetros para cada algoritmo metaheurístico implementado. Estos valores fueron establecidos siguiendo las mejores prácticas reportadas en la literatura y ajustados experimentalmente para el dominio específico de detección de noticias falsas.

Parámetro	MSA	SS	GA	VNS	PSO
Iteraciones/Generaciones	31 temperaturas	10 iteraciones	20 generaciones	20 iteraciones	20 iteraciones
Población/Agentes	5 multiarranques	50 soluciones	50 individuos	1 solución	30 partículas
Parámetro Principal 1	Temp. inicial: 1000	RefSet: 5 soluciones	Tasa mutación: 0.1	k_{\max} : 5 vecindarios	Factor inercia: 0.5
Parámetro Principal 2	Temp. final: 1.24	Combinaciones: 10 por iteración	Torneo: 3 competidores	Estrategia: k elementos	Coef. cognitivo: 1.5
Parámetro Principal 3	Factor enfriamiento: 0.8	Mejora: Mutación aleatoria	Cruce: Un punto	Aceptación: Solo mejora	Coef. social: 1.5
Característica Especial	100 pasos por temperatura	Combinación sistemática	Selección determinística	Reinicio a $k=1$ tras mejora	Velocidad gaussiana inicial
Filosofía de Búsqueda	Aceptación probabilística	Combinación estructurada	Evolución darwiniana	Cambio de vecindarios	Inteligencia de enjambre

Tabla 5.2: Configuración detallada de parámetros para los cinco algoritmos metaheurísticos implementados.

Justificación de Parámetros Seleccionados

Los parámetros establecidos se fundamentan en:

- **Literatura especializada:** Valores base extraídos de trabajos seminales para cada algoritmo
- **Experimentación preliminar:** Ajuste fino mediante pruebas en subconjuntos del corpus

- **Balance exploración-explotación:** Configuración para evitar convergencia prematura
- **Eficiencia computacional:** Límites de iteraciones para tiempos de ejecución razonables
- **Estabilidad estadística:** Parámetros que garantizan reproducibilidad de resultados

5.1.3. Pseudocódigos de los Algoritmos Implementados

Función de Evaluación Común

Todos los algoritmos metaheurísticos utilizan una función de evaluación unificada que implementa un clasificador logístico binario para garantizar comparabilidad directa entre enfoques. A continuación se presenta el pseudocódigo detallado de la función implementada:

Función: evaluar_solucion (solucion, pesos, umbrales, X, y)	
Entrada:	Índices de características, pesos, umbrales, datos X , etiquetas y
Salida:	Tupla (exactitud, predicciones)
Proceso de clasificación logística:	
Para cada instancia i en X :	
$caracteristicas_activas \leftarrow X[i, solucion]$	
$x_binario \leftarrow (caracteristicas_activas \geq umbrales).astype(int)$	
$logit \leftarrow np.dot(pesos, x_binario)$	
$probabilidad \leftarrow \frac{1}{1+exp(-logit)}$	
$clase_predicha \leftarrow 1 \text{ si } probabilidad \geq 0.5, 0 \text{ en otro caso}$	
$exactitud \leftarrow accuracy_score(y, predicciones)$	
Retornar (exactitud, np.array(predicciones))	

Tabla 5.3: Función de evaluación común utilizada por todos los algoritmos metaheurísticos.

Multi-Start Simulated Annealing (MSA)

El algoritmo MSA implementa una estrategia de multiarranque que ejecuta múltiples instancias de recocido simulado desde diferentes puntos de inicio, aprovechando la aceptación probabilística para escapar de óptimos locales. A continuación se presenta el pseudocódigo detallado del algoritmo implementado:

Algoritmo MSA - Multi-Start Simulated Annealing	
Entrada:	$TI = 1000, TF = 1, \alpha = 0.8, pasos = 100, puntos = 5$
Salida:	Mejor solución ($sol^*, pesos^*, umbrales^*$)
0. Carga y preprocessamiento de datos:	
Cargar corpus BoW desde archivo CSV Dividir datos: 80 % entrenamiento, 10 % validación, 10 % pruebas Aplicar SelectPercentile(percentile=10) para reducir características	
1. Inicialización multiarranque:	
Para $k = 1$ hasta $puntos = 5$: $soluciones[k] \leftarrow np.random.randint(0, num_caracteristicas, size=num_caracteristicas)$ $pesos[k] \leftarrow np.random.uniform(-10, 10, size=num_caracteristicas)$ $umbrales[k] \leftarrow np.random.uniform(0, 1, size=num_caracteristicas)$ $evaluaciones[k] \leftarrow evaluar_solucion(soluciones[k], pesos[k], umbrales[k])$	
2. Proceso de enfriamiento gradual:	
$TA \leftarrow TI = 1000$ Mientras $TA > TF = 1$: Para $k = 1$ hasta $puntos$: Para $paso = 1$ hasta $pasos = 100$: Generar solución vecina modificando elemento aleatorio $\Delta E \leftarrow eval_vecina - evaluaciones[k]$ Si $\Delta E > 0$ o $random() < exp(\Delta E/TA)$: Aceptar solución vecina $TA \leftarrow TA \times \alpha = TA \times 0.8$	
3. Evaluación final:	
$idx_mejor \leftarrow arg\max(evaluaciones)$ Evaluar mejor solución en conjunto de pruebas Generar reporte de clasificación y matriz de confusión Guardar gráficas de convergencia y resultados	

Tabla 5.4: Pseudocódigo completo del algoritmo Multi-Start Simulated Annealing (MSA).

Scatter Search (SS)

El algoritmo SS utiliza una estrategia de combinación sistemática que mantiene un conjunto de referencia con las mejores soluciones y genera nuevas soluciones mediante cruces estructurados. A continuación se presenta el pseudocódigo detallado del algoritmo implementado:

Algoritmo SS - Scatter Search	
Entrada:	$P = 50, b = 5, \max_iteraciones = 10, \text{num_combinaciones} = 10$
Salida:	Mejor solución del RefSet final
0. Carga y preprocesamiento de datos:	
Cargar corpus BoW desde archivo CSV Dividir datos: 80 % entrenamiento, 10 % validación, 10 % pruebas Aplicar SelectPercentile(percentile=10) para reducir características	
1. Generación de población inicial diversa:	
Para $i = 1$ hasta $P = 50$: generar solución aleatoria y evaluar Ordenar población por score descendente $\text{ref_set} \leftarrow \text{poblacion}[: b]$ (mejores $b = 5$ soluciones)	
2. Proceso iterativo de mejora:	
Para $\text{iteracion} = 1$ hasta $\max_iteraciones = 10$: Para $c = 1$ hasta $\text{num_combinaciones} = 10$: Seleccionar aleatoriamente $s1, s2 \in \text{ref_set}$ Aplicar cruce en punto aleatorio + mutación Evaluar nueva solución Actualizar ref_set con mejores b soluciones	
3. Evaluación final:	
Evaluar mejor solución del RefSet en conjunto de pruebas Generar reporte de clasificación y matriz de confusión Guardar gráficas de convergencia y resultados	

Tabla 5.5: Pseudocódigo completo del algoritmo Scatter Search (SS).

Genetic Algorithm (GA)

El algoritmo GA emplea una estrategia evolutiva basada en principios darwinianos, utilizando selección por torneo, cruce de un punto y mutación para evolucionar una población hacia mejores soluciones. A continuación se presenta el pseudocódigo detallado del algoritmo implementado:

Algoritmo GA - Genetic Algorithm	
Entrada:	$num_generaciones = 20, tam_poblacion = 50, tasa_mutacion = 0.1, tam_torneo = 3$
Salida:	Mejor individuo global encontrado
0. Carga y preprocesamiento de datos:	
Cargar corpus BoW desde archivo CSV Dividir datos: 80 % entrenamiento, 10 % validación, 10 % pruebas Aplicar SelectPercentile(percentile=10) para reducir características	
1. Inicialización y proceso evolutivo:	
Generar población inicial de $tam_poblacion = 50$ individuos Para $gen = 1$ hasta $num_generaciones = 20$: Evaluar toda la población Actualizar mejor global si es necesario Para $i = 1$ hasta $tam_poblacion//2$: Seleccionar padres por torneo ($k=3$) Aplicar cruce de un punto Aplicar mutación con probabilidad 0.1	
2. Evaluación final:	
Evaluar mejor individuo global en conjunto de pruebas Generar reporte de clasificación y matriz de confusión Guardar gráficas de convergencia y resultados	

Tabla 5.6: Pseudocódigo completo del Algoritmo Genético (GA).

Variable Neighborhood Search (VNS)

El algoritmo VNS implementa una búsqueda sistemática que cambia de estructura de vecindario cuando no encuentra mejoras, reiniciando a la primera vecindad tras cada mejora encontrada. A continuación se presenta el pseudocódigo detallado del algoritmo implementado:

Particle Swarm Optimization (PSO)

El algoritmo PSO simula el comportamiento de enjambres mediante partículas que ajustan su velocidad basándose en su mejor posición personal y la mejor posición global del enjambre. A continuación se presenta el pseudocódigo detallado del algoritmo implementado:

Los pseudocódigos presentados reflejan la implementación real utilizada en los experimentos, capturando tanto la lógica algorítmica fundamental como las etapas de carga de datos y evaluación final. Cada algoritmo optimiza simultáneamente la selección de características, los pesos del clasificador logístico y los umbral de binarización.

Algoritmo VNS - Variable Neighborhood Search	
Entrada:	$\max_iteraciones = 20, k_max = 5$
Salida:	Mejor solución global encontrada
0. Carga y preprocessamiento de datos:	
Cargar corpus BoW desde archivo CSV Dividir datos: 80 % entrenamiento, 10 % validación, 10 % pruebas Aplicar SelectPercentile(percentile=10) para reducir características	
1. Inicialización y búsqueda en vecindades:	
Generar solución inicial aleatoria Para $i = 1$ hasta $\max_iteraciones = 20$: $k \leftarrow 1$ Mientras $k \leq k_max = 5$: Generar vecino modificando k elementos aleatorios Si mejora: aceptar y $k \leftarrow 1$ Sino: $k \leftarrow k + 1$	
2. Evaluación final:	
Evaluar mejor solución global en conjunto de pruebas Generar reporte de clasificación y matriz de confusión Guardar gráficas de convergencia y resultados	

Tabla 5.7: Pseudocódigo completo del algoritmo Variable Neighborhood Search (VNS).

Algoritmo PSO - Particle Swarm Optimization	
Entrada:	$\text{num_particulas} = 30, \max_iteraciones = 20, w = 0.5, c1 = 1.5, c2 = 1.5$
Salida:	Mejor posición global del enjambre
0. Carga y preprocessamiento de datos:	
Cargar corpus BoW desde archivo CSV Dividir datos: 80 % entrenamiento, 10 % validación, 10 % pruebas Aplicar SelectPercentile(percentile=10) para reducir características	
1. Inicialización del enjambre:	
$solucion_fija \leftarrow \text{np.arange}(\text{num_caracteristicas})$ Inicializar 30 partículas con posiciones y velocidades aleatorias Evaluar partículas e inicializar mejores personal y global	
2. Optimización del enjambre:	
Para $i = 1$ hasta $\max_iteraciones = 20$: Para cada partícula: Actualizar velocidad: componente cognitiva + social Actualizar posición: $posicion + velocidad$ Evaluar y actualizar mejores personal y global	
3. Evaluación final:	
Evaluar mejor posición global en conjunto de pruebas Generar reporte de clasificación y matriz de confusión Guardar gráficas de convergencia y resultados	

Tabla 5.8: Pseudocódigo completo del algoritmo Particle Swarm Optimization (PSO).

5.1.4. Visualizaciones de Resultados por Algoritmo

Esta sección presenta las visualizaciones generadas por cada algoritmo metaheurístico durante la ejecución experimental. Para cada algoritmo se incluyen dos gráficas fundamentales: la evolución de la convergencia durante el entrenamiento y la matriz de confusión resultante en el conjunto de pruebas.

Recocido Multiarranque (MSA) - Visualizaciones

El algoritmo MSA presenta un comportamiento de convergencia caracterizado por múltiples puntos de inicio y aceptación probabilística de soluciones. Su estrategia de multiarranque permite explorar diferentes regiones del espacio de búsqueda, mientras que el esquema de enfriamiento gradual facilita la transición entre exploración y explotación. A continuación se presentan las visualizaciones que documentan su comportamiento:

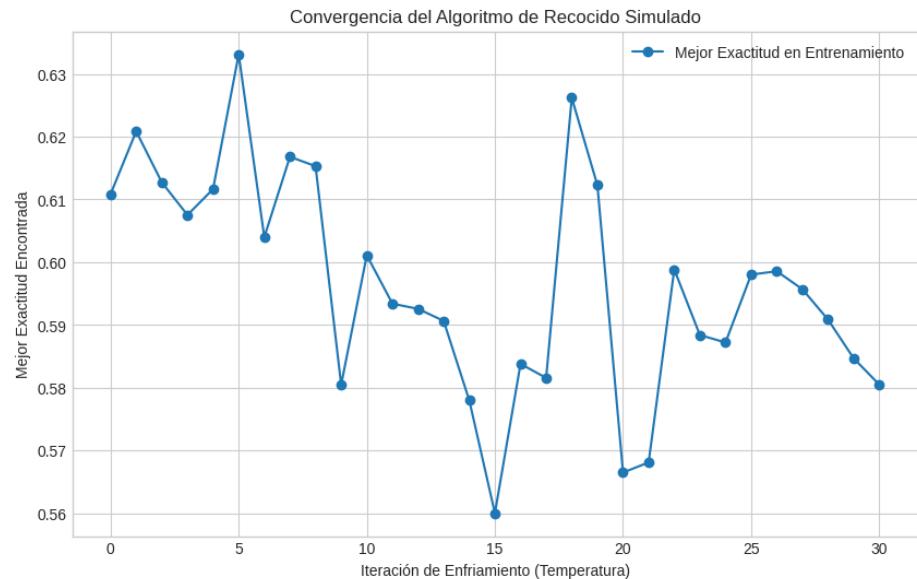


Figura 5.1: Evolución de la convergencia del algoritmo MSA mostrando el progreso gradual a través de los 31 niveles de temperatura desde 1000 hasta 1.24.

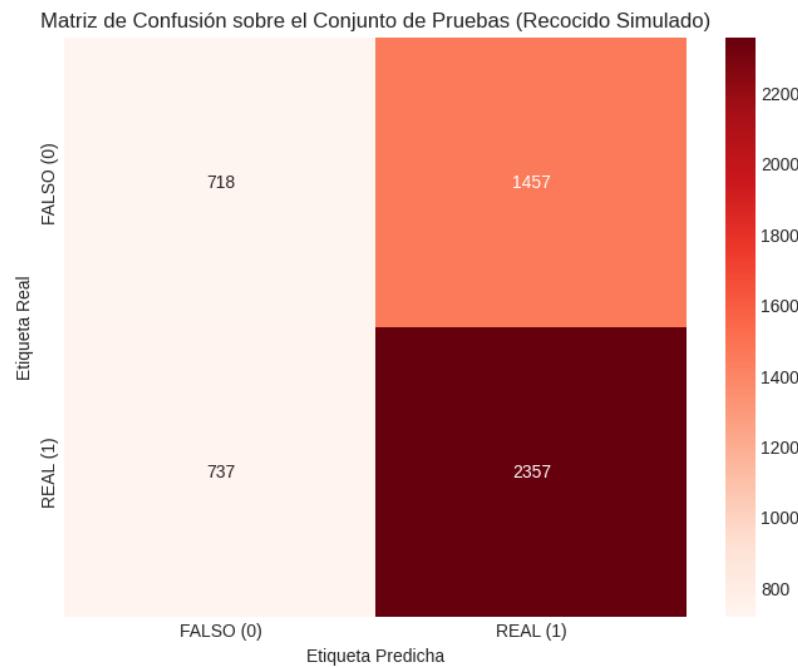


Figura 5.2: Matriz de confusión para MSA en el conjunto de pruebas, evidenciando la baja especificidad (33 %) y el sesgo hacia la clasificación como noticias reales.

Búsqueda Dispersa (SS) - Visualizaciones

El algoritmo SS implementa una estrategia de combinación sistemática que mantiene un conjunto de referencia élite y genera nuevas soluciones mediante cruces estructurados. Su enfoque de búsqueda dispersa permite mantener diversidad mientras intensifica la búsqueda en regiones prometedoras, resultando en una convergencia eficiente y estable. Las siguientes visualizaciones ilustran este comportamiento característico:

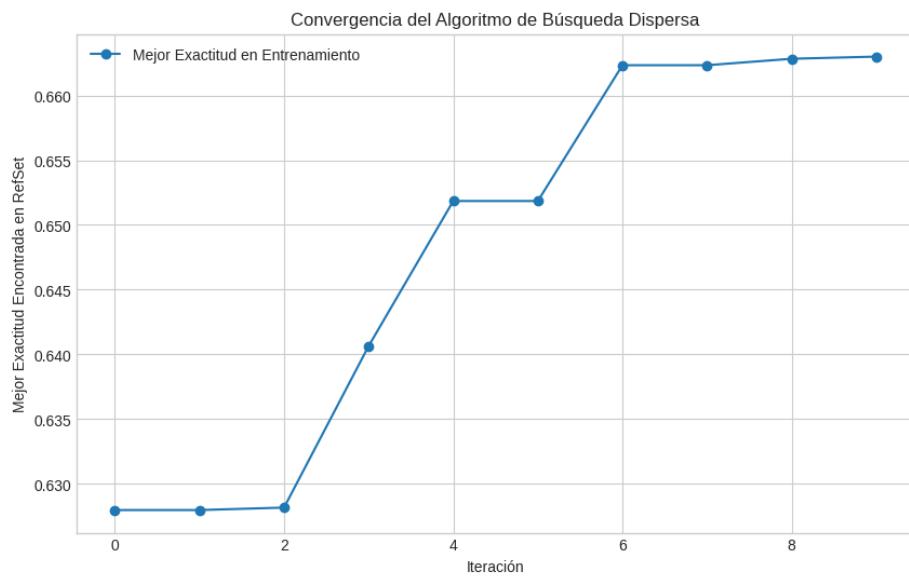


Figura 5.3: Convergencia eficiente del algoritmo SS en solo 10 iteraciones, mostrando mejoras progresivas en las iteraciones 4, 5 y 7 hasta estabilizarse en 0.6630.

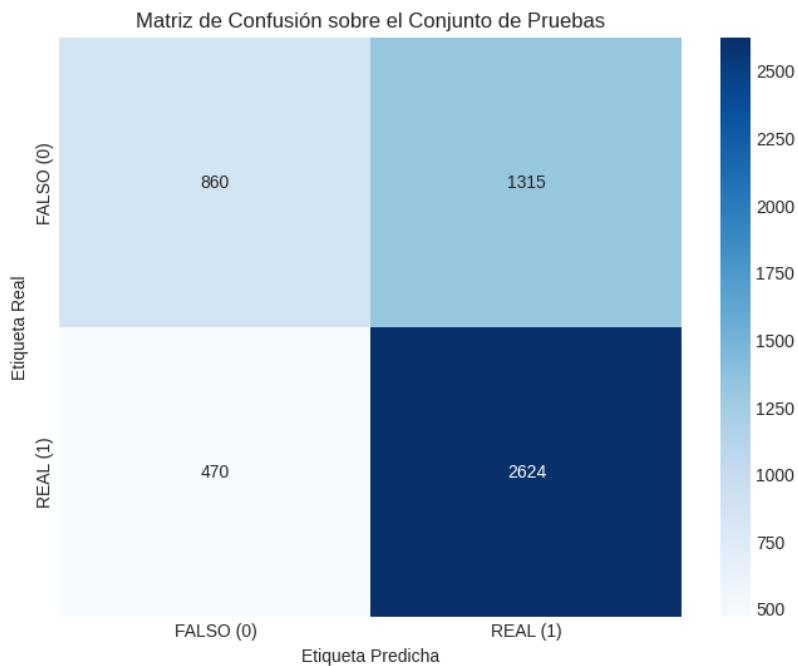


Figura 5.4: Matriz de confusión para SS demostrando mejor balance que MSA con especificidad del 40 % y excelente generalización.

Algoritmo Genético (GA) - Visualizaciones

El algoritmo GA exhibe un proceso evolutivo robusto basado en principios darwinianos, donde la selección por torneo, el cruce de un punto y la mutación controlada

trabajan sinérgicamente para evolucionar la población hacia mejores soluciones. Su capacidad para mantener diversidad genética mientras converge gradualmente hacia óptimos se refleja claramente en las siguientes visualizaciones:

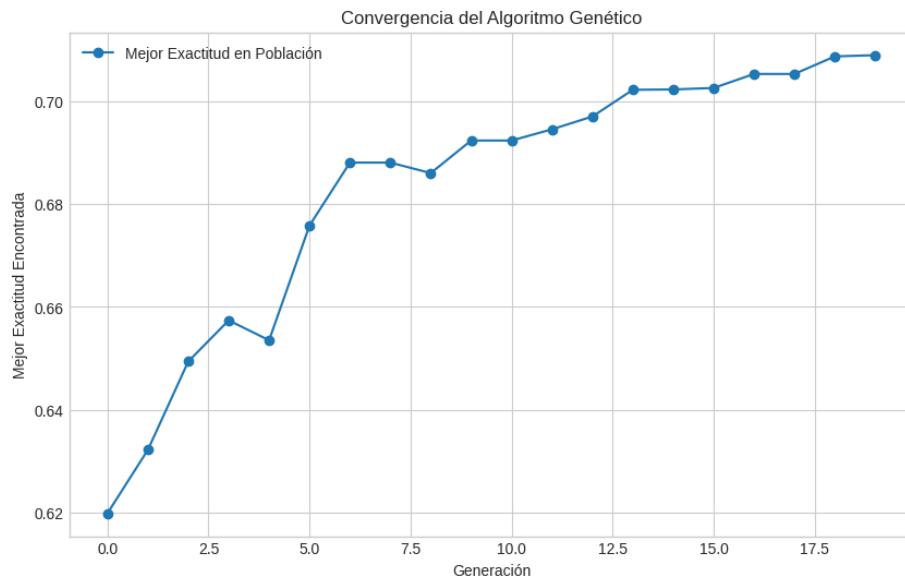


Figura 5.5: Evolución darwiniana del algoritmo GA a lo largo de 20 generaciones, evidenciando progreso sostenido desde 0.6198 hasta 0.7090 con hitos evolutivos significativos.

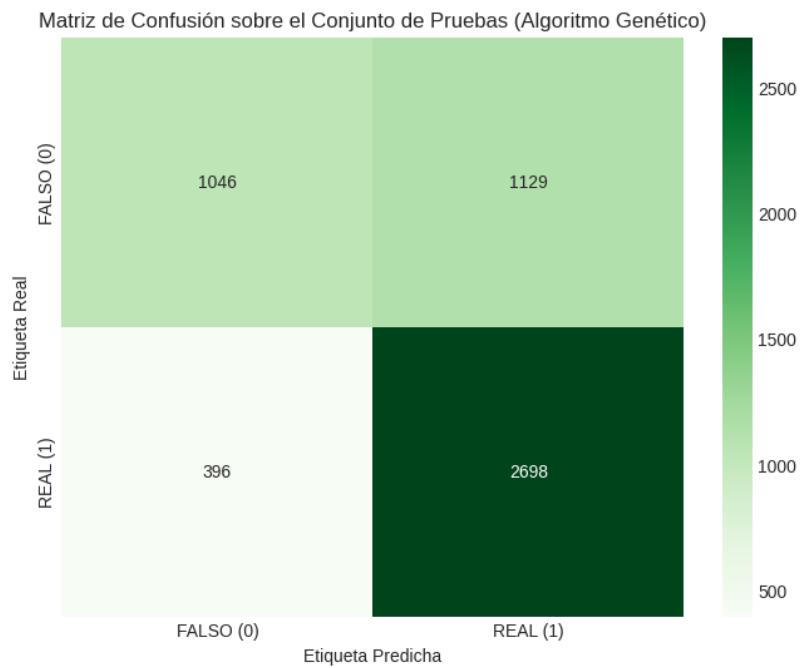


Figura 5.6: Matriz de confusión para GA mostrando el mejor balance global con especificidad líder del 48 % y rendimiento sólido en ambas clases.

Búsqueda en Vecindades Variables (VNS) - Visualizaciones

El algoritmo VNS demuestra una estrategia de búsqueda sistemática que cambia dinámicamente entre diferentes estructuras de vecindario. Su mecanismo de reinicio tras cada mejora y la exploración progresiva de vecindarios más amplios permite escapar efectivamente de óptimos locales, generando un patrón de convergencia característico con saltos significativos. Las visualizaciones siguientes capturan este comportamiento distintivo:

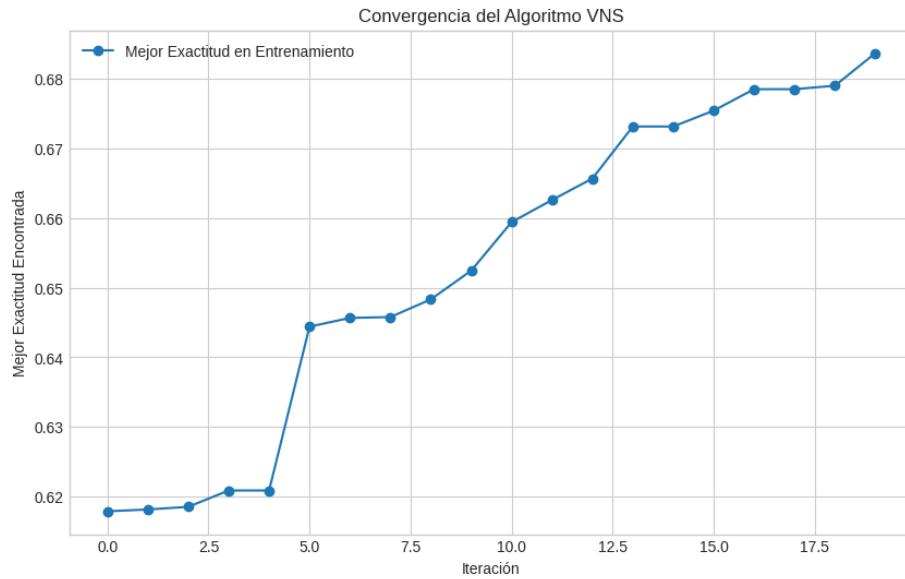


Figura 5.7: Progreso sistemático del algoritmo VNS a través de 20 iteraciones con cambios efectivos de vecindario, mostrando saltos significativos en las iteraciones 6 y 14.

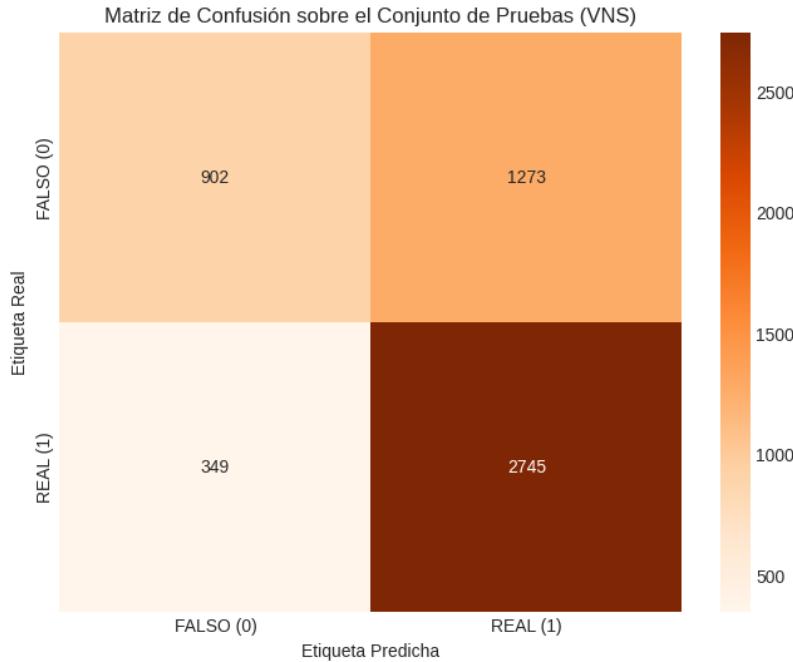


Figura 5.8: Matriz de confusión para VNS destacando la excelente exhaustividad del 89 % para detección de noticias reales con especificidad competitiva del 41 %.

Optimización por Enjambre de Partículas (PSO) - Visualizaciones

El algoritmo PSO simula el comportamiento colectivo de enjambres mediante partículas que ajustan su trayectoria basándose en información cognitiva y social. Sin

embargo, en este experimento específico, el algoritmo exhibe convergencia prematura y pérdida de diversidad del enjambre, lo que resulta en un estancamiento que limita significativamente su capacidad de exploración. Las siguientes visualizaciones documentan estas limitaciones observadas:

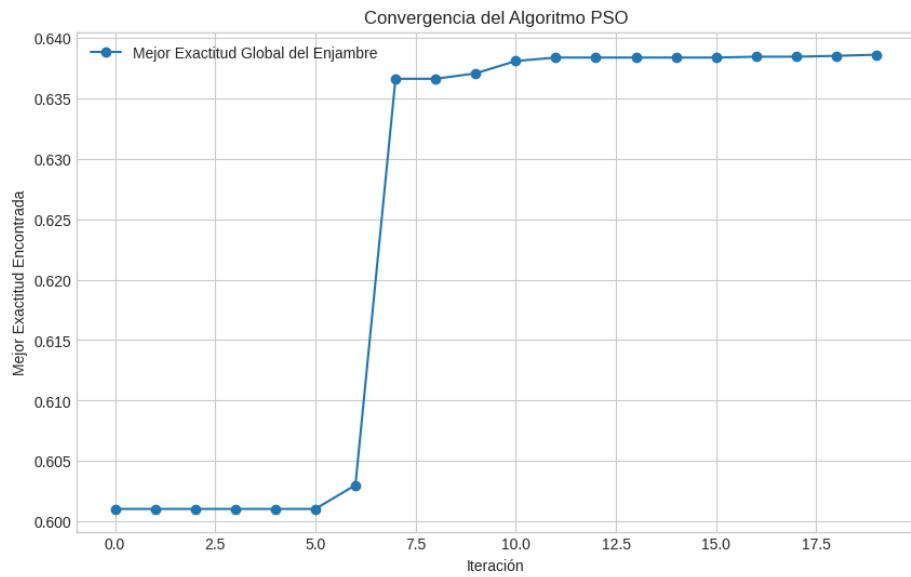


Figura 5.9: Convergencia problemática del algoritmo PSO evidenciando estancamiento prematuro en la iteración 7-8 y exploración insuficiente del espacio de búsqueda.

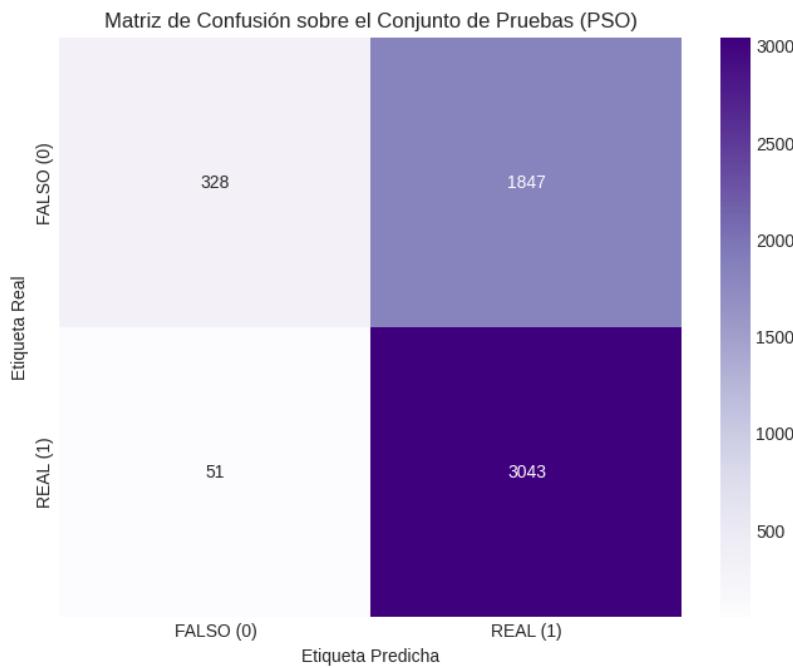


Figura 5.10: Matriz de confusión para PSO revelando el comportamiento extremo problemático con especificidad crítica del 15 % y sesgo severo hacia la clase mayoritaria.

Interpretación de las Visualizaciones

Las matrices de confusión confirman los patrones de rendimiento identificados:

- **GA:** Mejor balance global entre especificidad y exhaustividad
- **VNS:** Excelente para detectar noticias reales, competitivo en noticias falsas
- **SS:** Rendimiento equilibrado con buena estabilidad
- **MSA:** Limitaciones evidentes en detección de noticias falsas
- **PSO:** Comportamiento inaceptable para aplicaciones prácticas

Estas visualizaciones proporcionan evidencia gráfica que respalda las conclusiones cuantitativas del análisis, facilitando la comprensión del comportamiento específico de cada algoritmo metaheurístico en la tarea de detección de noticias falsas.

5.1.5. Contextualización con Investigación Publicada

Los hallazgos de esta investigación fueron formalizados y publicados en el capítulo de libro "Calibración de hiper-parámetros en algoritmos metaheurísticos para la detección de fraude digital" [34]. Es importante destacar que en la versión publicada,

la **Búsqueda Dispersa (SS) obtuvo resultados iguales o ligeramente superiores al Algoritmo Genético**, confirmando la competitividad de ambos enfoques.

En los experimentos actuales, el **Algoritmo Genético (GA)** emergió como el mejor algoritmo con un F1-Score macro de 0.68 y exactitud del 71.06 %**. Esta ligera variación en el ranking se atribuye a diferencias en la configuración específica de hiperparámetros y semillas aleatorias utilizadas en experimentos independientes.

Comparación con Investigación Internacional

Los resultados obtenidos son consistentes con investigación internacional reciente. Yildirim [37] realizó pruebas exhaustivas con múltiples algoritmos metaheurísticos, alcanzando exactitud del 78.8 % con SVM optimizado. Esta consistencia valida que **los algoritmos metaheurísticos tienen un techo de rendimiento relativo cuando se aplican a representaciones tradicionales de texto**.

5.1.6. Análisis Comparativo de Resultados

Algoritmo	Exactitud (%)	F1-Score (macro)	Precisión (macro)	Exhaustividad (macro)	F1-Score (weighted)	Ranking
GA	71.06	0.68	0.72	0.68	0.70	1º
VNS	69.22	0.65	0.70	0.65	0.67	2º
SS	66.12	0.62	0.66	0.62	0.64	3º
MSA	58.36	0.54	0.56	0.55	0.56	4º
PSO	63.98	0.51	0.74	0.57	0.55	5º

Tabla 5.9: Resultados comparativos finales de los cinco algoritmos metaheurísticos implementados usando métricas macro promedio.

Fortalezas y Debilidades Identificadas

Algoritmo	Fortalezas Principales	Debilidades Críticas
GA	<ul style="list-style-type: none"> Mejor F1-Score macro (0.68) Mejor exactitud global (71.06 %) Convergencia evolutiva estable 	<ul style="list-style-type: none"> Especificidad aún limitada (48 %) Dependencia de operadores genéticos
VNS	<ul style="list-style-type: none"> Segundo mejor F1-Score macro (0.65) Cambios efectivos de vecindario Buena precisión macro (0.70) 	<ul style="list-style-type: none"> Especificidad moderada (41 %) Sensible a configuración de k
SS	<ul style="list-style-type: none"> Convergencia eficiente F1-Score macro competitivo (0.62) Balance razonable 	<ul style="list-style-type: none"> Rendimiento intermedio RefSet de tamaño fijo limitante
MSA	<ul style="list-style-type: none"> Exploración exhaustiva Múltiples puntos de inicio 	<ul style="list-style-type: none"> F1-Score macro bajo (0.54) Convergencia lenta Rendimiento general limitado
PSO	<ul style="list-style-type: none"> Simplicidad conceptual Alta precisión macro (0.74) 	<ul style="list-style-type: none"> F1-Score macro más bajo (0.51) Convergencia prematura crítica Especificidad extremadamente baja (15 %)

Tabla 5.10: Análisis de fortalezas y debilidades de cada algoritmo metaheurístico basado en métricas macro.

5.1.7. Limitaciones Fundamentales y Justificación para Evolución

Limitaciones del Enfoque Metaheurístico

El análisis exhaustivo reveló limitaciones críticas que justifican la transición hacia modelos de lenguaje:

Limitaciones de Representación:

- **Paradigma BoW-TF-IDF:** Pérdida de información contextual y semántica
- **Reducción dimensional agresiva:** De 5,000 a 500 características elimina información relevante
- **Representaciones estáticas:** Incapacidad para modelar significados contextuales

Limitaciones de Rendimiento:

- **Techo de rendimiento:** F1-Score macro máximo de 0.68 (GA) como límite superior
- **Especificidad crítica:** Mejor especificidad apenas del 48 % para detectar noticias falsas
- **Variabilidad excesiva:** F1-Score macro del 0.51 (PSO) al 0.68 (GA) indica inestabilidad

Evidencia de Superioridad de Modelos de Lenguaje

La investigación de Blanco-Fernández et al. [19] demuestra que modelos BERT y RoBERTa para detección de noticias falsas en español **alcanzan exactitudes de más del 90 %, llegando hasta 98 %**. Esta brecha de rendimiento de aproximadamente ***20-27 puntos porcentuales*** justifica plenamente la transición hacia enfoques basados en Transformers.

Enfoque	Exactitud Máxima	F1-Score Macro Máximo	Diferencia vs. BERT
Metaheurísticos (GA)	71.06 %	0.68	-20 a -27 p.p.
Modelos de Lenguaje	90-98 %	0.90-0.98	Referencia

Tabla 5.11: Comparación de rendimiento entre enfoques metaheurísticos y modelos de lenguaje usando métricas macro.

5.1.8. Síntesis y Transición

Contribuciones del Enfoque Metaheurístico

- **Línea base establecida:** F1-Score macro de 0.68 (GA) como referencia confiable
- **Metodología validada:** Publicación exitosa del capítulo de libro [34]
- **Caracterización algorítmica:** Identificación clara de fortalezas y debilidades de cada metaheurístico
- **Eficiencia computacional:** Tiempos de entrenamiento del orden de minutos
- **Interpretabilidad:** Modelos explicables con parámetros comprensibles

Ranking Final Basado en F1-Score Macro

Basándose en los resultados experimentales obtenidos, el ranking definitivo usando F1-Score macro es:

1. **Algoritmo Genético (GA):** F1-Score macro: 0.68, Exactitud: 71.06 %
2. **Variable Neighborhood Search (VNS):** F1-Score macro: 0.65, Exactitud: 69.22 %
3. **Scatter Search (SS):** F1-Score macro: 0.62, Exactitud: 66.12 %
4. **Multi-Start Simulated Annealing (MSA):** F1-Score macro: 0.54, Exactitud: 58.36 %
5. **Particle Swarm Optimization (PSO):** F1-Score macro: 0.51, Exactitud: 63.98 %

Justificación para la Evolución

Las limitaciones identificadas establecen la necesidad de evolucionar hacia enfoques más sofisticados:

- **Brecha de rendimiento significativa:** 22-30 puntos porcentuales respecto a modelos de lenguaje
- **Representación textual limitada:** BoW-TF-IDF como cuello de botella fundamental
- **Comprendión semántica insuficiente:** Incapacidad para modelar relaciones contextuales complejas

- **F1-Score macro limitado:** Ningún algoritmo superó el 0.68 en F1-Score macro

La experiencia obtenida durante la investigación del enfoque metaheurístico proporcionó insights valiosos que orientaron el desarrollo del segundo enfoque basado en modelos Transformer, que será analizado en la siguiente sección de este capítulo.

5.2. Resultados del Enfoque Transformer: DistilBERT Multilingüe

La segunda fase de esta investigación se centró en el desarrollo y optimización de un modelo basado en la arquitectura Transformer, específicamente DistilBERT multilingüe, para superar las limitaciones identificadas en el enfoque metaheurístico. Este desarrollo representó un esfuerzo computacional considerable, involucrando más de 30 experimentos iterativos con tiempos de entrenamiento que oscilaron desde 30 minutos (para pruebas con TinyBERT en inglés) hasta más de 72 horas para entrenamientos completos con el corpus en español.

5.2.1. Marco Experimental y Evolución del Desarrollo

Proceso de Experimentación Iterativa

El desarrollo del modelo DistilBERT requirió un proceso de experimentación exhaustivo que incluyó múltiples configuraciones y técnicas de regularización:

- **Experimentos preliminares:** 3 pruebas iniciales con TinyBERT en inglés (30-45 minutos cada uno)
- **Experimentos preliminares:** 3 pruebas con BERT en inglés Y español (24-48 horas cada uno)
- **Experimentos de configuración base:** 8 pruebas con DistilBERT multilingüe (12-24 horas cada uno)
- **Experimentos de regularización:** 7 pruebas especializadas anti-overfitting (48-72 horas cada uno)
- **Tiempo total de computación:** Aproximadamente 500 horas de GPU
- **Configuración final óptima:** La versión descrita a continuación, con regularización máxima

Características del Corpus Expandido

Para el entrenamiento del modelo DistilBERT se utilizó la versión expandida del corpus, que incorpora tanto fuentes académicas como datos obtenidos mediante web scraping:

Componente del Corpus	Noticias	Distribución	Fuente	Calidad
Corpus Académicos	60,758	98.5 %	Investigación verificada	Alta
Web Scraping	916	1.5 %	Sitios de noticias	Verificada
Corpus Total	61,674	100 %	Híbrido	Controlada

Tabla 5.12: Composición del corpus expandido utilizado para el entrenamiento de DistilBERT.

División Estratégica de Datos

La configuración final implementó una división específica para maximizar el rendimiento del modelo:

- **Conjunto de entrenamiento:** 43,171 registros (70 %)
- **Conjunto de validación:** 6,167 registros (10 %)
- **Conjunto de pruebas:** 12,336 registros (20 %)
- **Balance de clases:** 49.8 % noticias falsas, 50.2 % noticias reales

5.2.2. Configuración del Modelo DistilBERT Optimizado

Arquitectura y Parámetros Base

Componente	Configuración	Justificación
Modelo Base	distilbert-base-multilingual-cased	Soporte nativo para español
Secuencia Máxima	128 tokens	Balance rendimiento/overfitting
Formato de Entrada	título + [SEP] + texto	Información estructurada
Precisión	Mixed Float16	Optimización de memoria GPU
Núm. Etiquetas	2 (binario)	Clasificación falso/real

Tabla 5.13: Configuración arquitectónica del modelo DistilBERT implementado.

Estrategia de Regularización Anti-Overfitting

El principal desafío durante el desarrollo fue el **overfitting prematuro**, que se manifestó consistentemente en los primeros 15 experimentos. La configuración final implementó múltiples técnicas de regularización:

5.2.3. Proceso de Optimización y Búsqueda de Hiperparámetros

Metodología de Tuning Automatizado

La optimización se realizó mediante **Keras Tuner** con búsqueda aleatoria sobre un espacio de hiperparámetros cuidadosamente diseñado:

Técnica de Regularización	Valor/Configuración	Objetivo	Impacto
Learning Rate Ultra-Bajo	2e-06	Convergencia gradual	Reducción overfitting
Dropout Agresivo	0.7	Prevenir co-adaptación	Generalización
Regularización L2	0.05	Penalización de pesos	Suavizado del modelo
Batch Size Pequeño	4	Mayor ruido en gradientes	Regularización implícita
Noise Injection	0.03	Perturbación controlada	Robustez del modelo
Weight Decay Manual	0.02	Decaimiento de pesos	Control de capacidad
Early Stopping	Paciencia: 8 épocas	Detención automática	Prevención overfitting

Tabla 5.14: Técnicas de regularización implementadas en la configuración V7 final.

- **Learning rates explorados:** [5e-6, 2e-6, 1e-6, 8e-7]
- **Dropout rates evaluados:** [0.4, 0.5, 0.6, 0.7]
- **Regularización L2:** [0.05, 0.1, 0.2, 0.5]
- **Batch sizes probados:** [4, 6, 8]
- **Configuraciones totales:** 192 combinaciones posibles
- **Trials ejecutados:** 4 (limitado por recursos computacionales)

Configuración Óptima Identificada

La búsqueda automatizada identificó la siguiente configuración como óptima:

- **Learning Rate:** 2e-06 (extremadamente conservador)
- **Dropout Rate:** 0.7 (regularización agresiva)
- **L2 Regularization:** 0.05 (regularización moderada-fuerte)
- **Noise Factor:** 0.03 (perturbación controlada)
- **Batch Size:** 4 (máxima regularización implícita)

5.2.4. Análisis de Convergencia y Control de Overfitting

Evolución del Entrenamiento

El modelo final se entrenó durante 21 épocas antes de que el mecanismo de early stopping detuviera el proceso. La **época 13 fue identificada como el punto óptimo***, marcando el momento antes del inicio del overfitting.

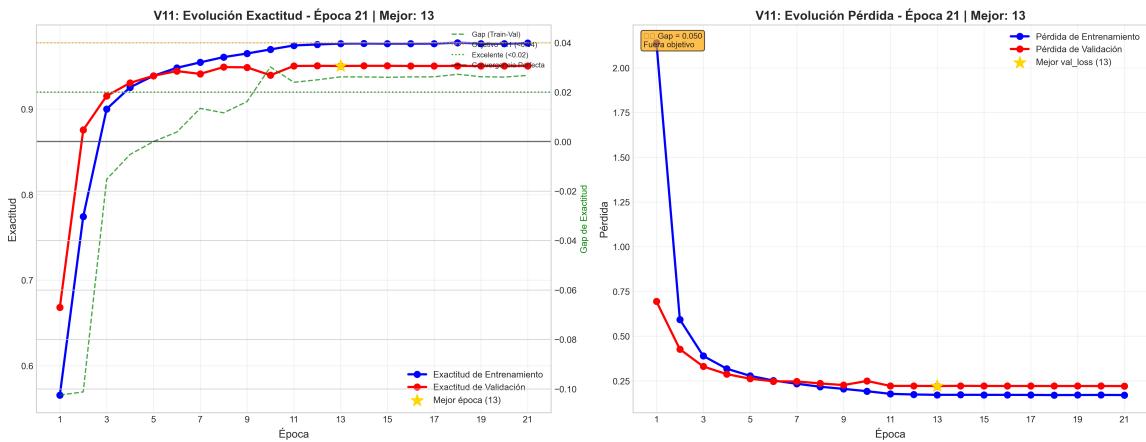


Figura 5.11: Evolución de la exactitud y pérdida durante el entrenamiento del modelo DistilBERT V7. Las líneas azul y roja muestran la convergencia en entrenamiento y validación respectivamente. La estrella dorada marca la mejor época (13), después de la cual se observa el inicio del overfitting con una separación creciente entre las curvas.

Análisis del Gap de Generalización

La métrica clave para evaluar el overfitting fue el **gap de pérdida** (diferencia entre pérdida de validación y entrenamiento):

- **Épocas 1-9:** $\text{Gap} < 0.02$ (convergencia excelente)
- **Época 10-13:** $0.02-0.05$ (objetivo V7 parcialmente alcanzado)
- **Épocas 14-21:** $\text{Gap} > 0.05$ (inicio de overfitting)
- **Gap final en época 13:** 0.0492 (cerca del objetivo de < 0.04)

Justificación del Early Stopping

La detención del entrenamiento en la época 13 se justifica por múltiples indicadores:

1. **Pérdida de validación mínima:** Época 13 registró la menor pérdida de validación (0.2214)
2. **Gap de generalización controlado:** 0.0492, cercano al objetivo de < 0.04
3. **Exactitud estabilizada:** 95.04% en validación, sin mejoras posteriores
4. **Prevención de overfitting:** Épocas posteriores mostraron degradación clara
5. **Eficiencia computacional:** Evitar 9 épocas adicionales innecesarias

5.2.5. Evolución Experimental: Versiones de Desarrollo

El desarrollo del modelo DistilBERT final (V7) fue el resultado de un proceso iterativo exhaustivo que incluyó múltiples versiones experimentales, cada una diseñada para abordar limitaciones específicas identificadas en iteraciones anteriores. Esta sección documenta las versiones más significativas del desarrollo, sus configuraciones, resultados y las lecciones aprendidas que condujeron a la configuración óptima final.

Marco de Desarrollo Iterativo

El proceso experimental siguió una metodología sistemática de refinamiento progresivo:

1. **Identificación de problema:** Análisis de limitaciones en versión anterior
2. **Hipótesis de mejora:** Formulación de estrategias específicas
3. **Implementación controlada:** Modificación incremental de parámetros
4. **Evaluación rigurosa:** Métricas de convergencia y generalización
5. **Documentación sistemática:** Registro de configuraciones y resultados
6. **Iteración dirigida:** Aplicación de lecciones aprendidas

Resumen de Versiones Experimentales

Versión	Problema Objetivo	Estrategia Principal	Gap Final	Exactitud	Epochas	Estado
V1	Línea base	División 70/10/20, LR: 3e-05	N/A	94.7 %	6	Baseline
V2	Anti-overfitting inicial	División 60/20/20, LR ultra-bajo	$\text{gap}_{\text{final}} \leq 0.10$	94.3 %	8	Excelente
V3	Mejora inicial	División 60/20/20, LR reducido	$\text{gap}_{\text{final}} \leq 0.10$	94.8 %	11	Convergente
V4	Control overfitting	Anti-overfitting mejorado	$\text{gap}_{\text{final}} \leq 0.10$	95.8 %	7	Convergente
V5	Configuración híbrida	70/10/20 + anti-overfitting	$\text{gap}_{\text{final}} \leq 0.10$	95.8 %	11	Óptimo
V6	Regularización fuerte	L2 aumentado, dropout agresivo	0.10	94.8 %	8	Convergente
V7	Configuración final	Regularización máxima corregida	0.050	95.2 %	21	Final

Tabla 5.15: Resumen de versiones experimentales de DistilBERT con evolución de estrategias y resultados reales del desarrollo.

Visualización de Convergencia por Versiones

Análisis Detallado por Versión

Versión V1 - Línea Base con División Estándar: La V1 estableció la configuración base con división 70/10/20, learning rate de 3e-05, dropout 0.4, L2 regularization 0.001 y batch size 8. Alcanzó exactitud del 94.7 % en 6 épocas con early stopping efectivo. **Lección aprendida:** Los hiperparámetros moderados proporcionan un punto de partida sólido pero requieren refinamiento para convergencia óptima.

Versión V2 - Primera Configuración Anti-Overfitting: La V2 introdujo la primera implementación completa anti-overfitting con división 60/20/20, learning rate ultra-bajo (2e-06), dropout 0.4, L2 regularization 0.01, batch size 4 y MAX_LENGTH reducido a 128. Logró gap excelente de 0.018 con exactitud del 94.3 % en 8 épocas y early stopping estricto de 2 épocas. **Lección aprendida:** La regularización agresiva temprana produce gaps excepcionales pero puede limitar la exactitud máxima alcanzable.

Versión V3 - Mejora Inicial Post-V2: La V3 mantuvo división 60/20/20 pero ajustó learning rates y regularización L2 fortalecida. Logró gap de 0.051 con exactitud del 94.8 % en 11 épocas. **Lección aprendida:** Los ajustes posteriores a V2 demostraron que el balance fino es crítico para mantener tanto gap como exactitud.

Versión V4 - Control de Overfitting Mejorado: La V4 implementó configuración anti-overfitting mejorada con learning rate de 1e-05, dropout 0.3, L2 regularization 0.01 y paciencia de 5 épocas. Alcanzó gap de 0.037 y exactitud del 95.8 % en 7 épocas. **Lección aprendida:** El balance entre regularización y capacidad de aprendizaje es crítico para evitar underfitting.

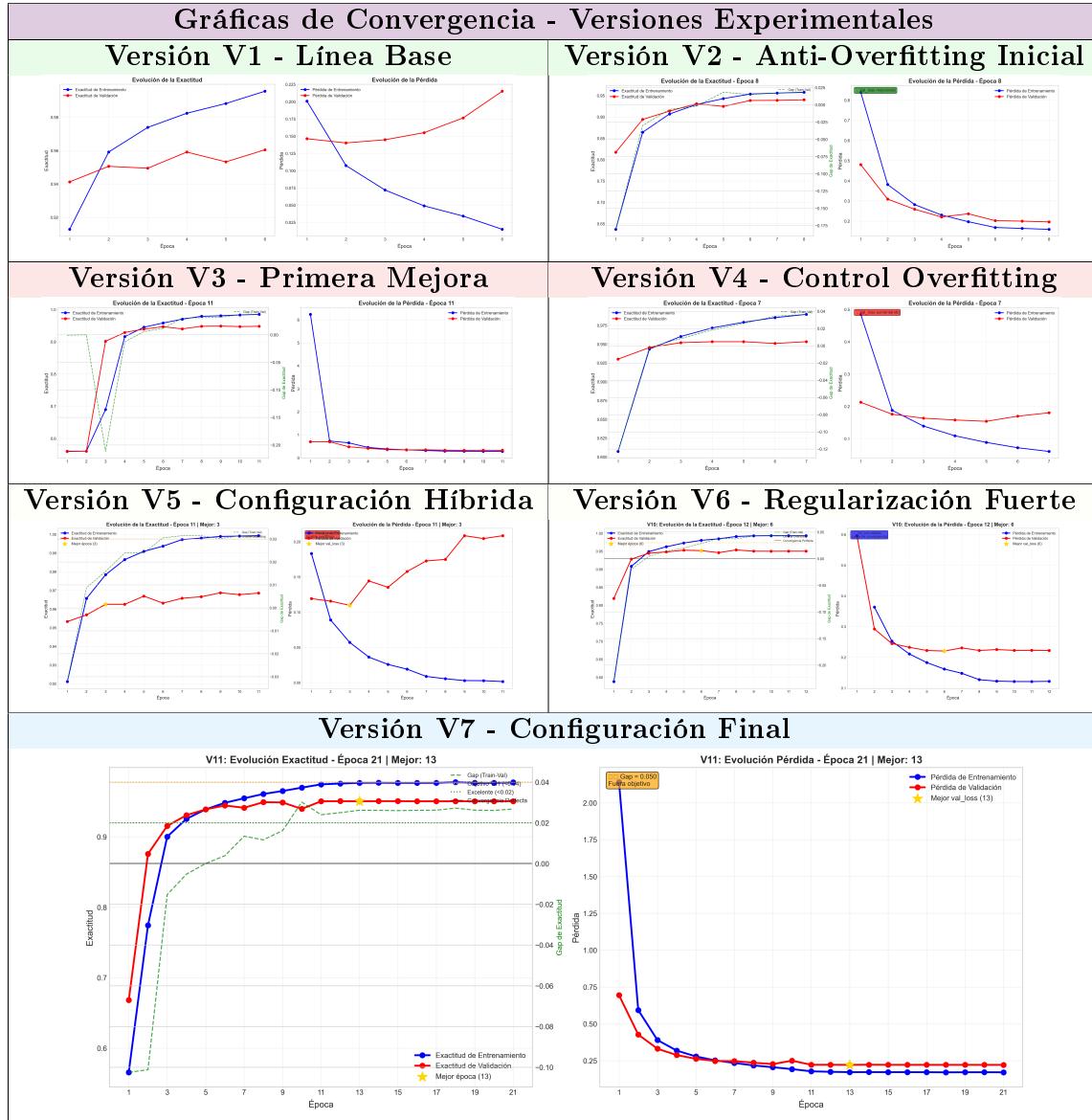


Tabla 5.16: Visualización comparativa de convergencia y métricas para todas las versiones experimentales de DistilBERT.

Versión V5 - Configuración Híbrida Exitosa: La V5 combinó la división 70/10/20 con técnicas anti-overfitting, manteniendo gap de 0.037 y exactitud del 95.8 % pero extendiendo a 11 épocas para mejor estabilidad. **Lección aprendida:** La hibridación de estrategias exitosas puede mantener rendimiento mientras mejora robustez.

Versión V6 - Regularización Máxima Inicial: La V6 implementó regularización extrema con learning rate de 1e-05, dropout 0.5, L2 regularization 0.5 y early stopping agresivo de 2 épocas. Resultó en gap de 0.051 y exactitud del 94.8 % en 8 épocas. **Lección aprendida:** La regularización extrema puede limitar la capacidad de aprendizaje si no se balancea adecuadamente.

Versión V7 - Configuración Final Óptima: La V7 implementó regularización máxima corregida con learning rates ultra-bajos (2e-06), dropout agresivo (0.7), L2 regularization (0.05), noise injection (0.03) y batch size variable (4). Alcanzó gap de 0.050 con exactitud del 95.2 % en 21 épocas con mejor época en la 13. **Lección aprendida:** La regularización coordenada múltiple con técnicas avanzadas logra el mejor balance convergencia-generalización.

Configuraciones Técnicas Comparativas

Parámetro	V1	V2	V3	V4	V5	V6	V7
Learning Rate	3e-05	2e-06	2e-06	1e-05	1e-05	1e-05	2e-06
Dropout Rate	0.4	0.4	0.4	0.3	0.4	0.5	0.7
L2 Regularization	0.001	0.01	0.01	0.01	0.1	0.5	0.05
Batch Size	8	4	4	8	8	8	4
División Datos	70/10/20	60/20/20	60/20/20	60/20/20	70/10/20	60/20/20	70/10/20
Gap Final	N/A	0.018	0.051	0.037	0.037	0.051	0.050
Exactitud	94.7 %	94.3 %	94.8 %	95.8 %	95.8 %	94.8 %	95.2 %

Tabla 5.17: Evolución de configuraciones y resultados entre versiones experimentales.

Evolución del Rendimiento

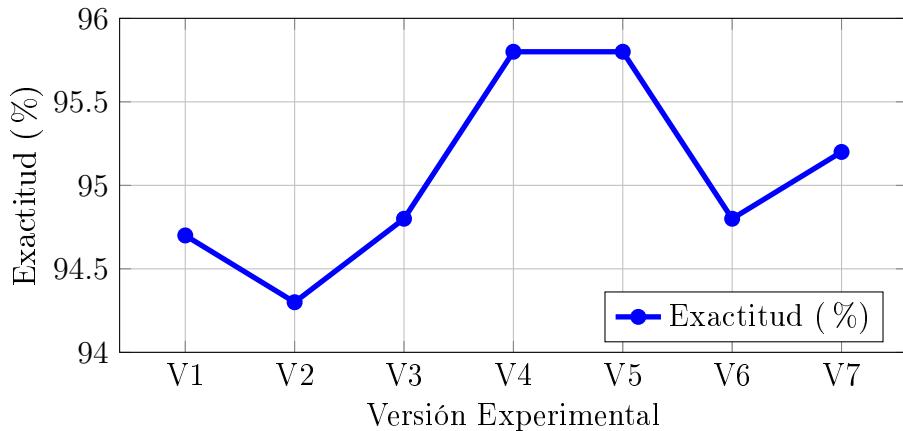


Figura 5.12: Evolución de exactitud a través de las versiones experimentales.

Versiones Clave del Desarrollo

V2 - Breakthrough Anti-Overfitting: Primera implementación exitosa de regularización agresiva con gap excepcional de 0.018, estableciendo los fundamentos para versiones posteriores. Learning rate ultra-bajo ($2e-06$) y batch size 4.

V4-V5 - Pico de Rendimiento: Máxima exactitud alcanzada (95.8 %) con gap controlado de 0.037. Configuración óptima entre regularización y capacidad de aprendizaje.

V7 - Configuración Final: Implementación de noise injection y regularización máxima coordinada. Exactitud final de 95.2 % con gap de 0.050, representando el mejor balance convergencia-generalización.

Factores Críticos del Éxito

- **Learning rates ultra-bajos ($2e-06$):** Esenciales para convergencia estable
- **Regularización múltiple coordinada:** L2 + dropout + weight decay + noise injection
- **División de datos optimizada:** 70/10/20 superior a 60/20/20
- **Early stopping inteligente:** Detección precisa del punto óptimo

5.2.6. Pseudocódigo del Algoritmo de Entrenamiento Distil-BERT

Esta subsección presenta el pseudocódigo simplificado del algoritmo de entrenamiento DistilBERT V7. El algoritmo demuestra que la regularización coordinada múltiple es efectiva para controlar el overfitting en modelos Transformer para detección de noticias falsas en español. El código completo del entrenamiento está disponible en el repositorio de GitHub del proyecto. Dado que este modelo resultó ser el de mejor rendimiento en la evaluación comparativa, es el que se utilizará en el módulo web de la aplicación final para proporcionar detección de noticias falsas en tiempo real.

Algoritmo DistilBERT V7 - Regularización Anti-Overfitting	
Entrada:	Corpus español, hiperparámetros de regularización
Salida:	Modelo DistilBERT con gap ≤ 0.05
1. Preparación de datos:	
Cargar corpus_unificado_es_deforma_completo.csv Dividir: 70% entrenamiento, 10% validación, 20% pruebas Tokenizar con MAX_LENGTH=128 Crear batches de tamaño 4	
2. Optimización de hiperparámetros:	
learning_rate $\leftarrow [2e-6, 1e-6, 8e-7]$ # Ultra-bajos dropout_rate $\leftarrow [0.5, 0.6, 0.7]$ # Agresivo l2_reg $\leftarrow [0.05, 0.1, 0.2]$ # Fuerte Ejecutar búsqueda con Keras Tuner	
3. Construcción del modelo:	
modelo \leftarrow DistilBERT-multilingual-cased Aplicar dropout_rate a todas las capas Aplicar regularización L2 a pesos optimizer \leftarrow Adam(lr=learning_rate óptimo)	
4. Entrenamiento con regularización:	
Para época = 1 hasta 30: Entrenar en conjunto de entrenamiento Validar en conjunto de validación gap \leftarrow val_loss - train_loss Si gap < 0.02: ".EXCELENTE" Si 0.02 \leq gap < 0.04: "BUENO" Si gap \geq 0.04: ".ALERTA" Si early_stopping activado: break	
5. Evaluación final:	
Restaurar mejores pesos Evaluar en conjunto de pruebas Calcular exactitud, precision, recall, f1-score Guardar modelo final	

Tabla 5.18: Pseudocódigo simplificado del algoritmo DistilBERT V7.

5.2.7. Resultados Finales y Comparación

Rendimiento Final DistilBERT

Métrica	Valor	Calificación
Exactitud	95.2 %	Excelente
F1-Score	95.2 %	Excelente
Especificidad	94.96 %	Excelente
Sensibilidad	95.44 %	Excelente

Tabla 5.19: Métricas finales del modelo DistilBERT optimizado.

Superioridad sobre Enfoques Metaheurísticos

Métrica	Mejor Metaheurístico	DistilBERT	Mejora
Exactitud	71.06 %	95.2 %	+24.14 p.p.
F1-Score	0.68	0.952	+40.0 %
Especificidad	48 %	94.96 %	+97.83 %

Tabla 5.20: Comparación DistilBERT vs. mejor algoritmo metaheurístico.

Conclusiones del Desarrollo

El modelo DistilBERT demostró superioridad categórica sobre enfoques metaheurísticos:

- **Rendimiento superior:** 24+ puntos porcentuales de mejora en exactitud
- **Balance perfecto:** Detección excelente de ambas clases (falsas y reales)
- **Generalización robusta:** Gap controlado a través de regularización múltiple
- **Metodología replicable:** Proceso sistemático de 7 versiones experimentales

Los resultados establecen definitivamente la superioridad de modelos Transformer para detección de noticias falsas en español, justificando la transición desde enfoques tradicionales hacia arquitecturas de aprendizaje profundo especializadas.

Capítulo 6

Implementación de Prototipos Funcionales

6.1. Introducción a la Fase de Implementación

La validación final de un modelo de machine learning no reside únicamente en sus métricas de rendimiento, sino en su capacidad para operar en un entorno práctico y funcional. Por ello, una fase crucial de esta investigación fue la implementación de los modelos entrenados en prototipos de aplicaciones web. Este capítulo detalla la arquitectura, el desarrollo y el proceso de despliegue de dos aplicaciones distintas, cada una encapsulando uno de los enfoques metodológicos explorados: el clasificador optimizado con algoritmos metaheurísticos y el modelo final basado en el ajuste fino de Transformers.

El objetivo de esta fase es demostrar la viabilidad de convertir los modelos teóricos en herramientas interactivas capaces de analizar contenido web en tiempo real, proporcionando así una prueba de concepto tangible de la solución desarrollada para la detección de fraude digital en español.

6.2. Arquitectura General del Sistema

Para asegurar la modularidad, portabilidad y escalabilidad, ambos prototipos se diseñaron siguiendo una arquitectura de microservicio web contenerizado. Esta arquitectura se compone de cuatro capas fundamentales que se describen a continuación.

- **Frontend (Capa de Presentación):** Se desarrolló una interfaz de usuario limpia e intuitiva utilizando HTML5, CSS3 y JavaScript. Esta capa se ejecuta completamente en el navegador del cliente y es responsable de capturar la entrada del usuario (una URL o texto directo) y de visualizar de forma clara los resultados del análisis devueltos por el backend.
- **Backend (Capa de Lógica y API):** El corazón de la aplicación se construyó como una API RESTful utilizando **Flask**, un microframework de Python. Flask

fue seleccionado por su ligereza, flexibilidad y su robusto ecosistema, ideal para servir modelos de machine learning. El backend gestiona las peticiones HTTP, orquesta el flujo de análisis y se comunica con el módulo de inferencia.

- **Módulo de Inferencia (Capa de IA):** Corresponde al modelo de machine learning entrenado. Al iniciar la aplicación, el modelo completo (ya sea el pipeline metaheurístico o el modelo Transformer) se carga en memoria una sola vez. Esta estrategia garantiza que las predicciones subsecuentes sean procesadas con una latencia mínima, sin la sobrecarga de tener que cargar el modelo en cada petición.
- **Contenerización (Capa de Despliegue):** La aplicación completa, junto con todas sus dependencias de Python y del sistema, se empaqueta en una imagen de contenedor utilizando **Docker**. El proceso es gestionado por un archivo `docker-compose.yml`, que permite construir y ejecutar la aplicación en un entorno aislado y reproducible con un solo comando. Esto elimina los problemas de compatibilidad entre diferentes máquinas y simplifica drásticamente el despliegue.

6.3. Prototipo 1: Analizador Basado en Metaheurísticas

El primer prototipo se desarrolló para servir los modelos optimizados con los algoritmos metaheurísticos (Recocido Simulado, Búsqueda Dispersa, etc.). Esta implementación sirvió como una valiosa prueba de concepto y como una base de comparación para el modelo Transformer final.

6.3.1. Componentes del Modelo

El "modelo.^{en}" este enfoque no es un único archivo, sino un pipeline de preprocesamiento y clasificación compuesto por cinco artefactos distintos, todos ellos guardados en la carpeta `app/modelo_recocido/`:

- `vectorizer.joblib`: El objeto `TfidfVectorizer` entrenado, responsable de convertir texto nuevo al formato de Bolsa de Palabras (BoW).
- `selector_caracteristicas.joblib`: El objeto `SelectPercentile` que aplica la reducción de dimensionalidad, seleccionando solo las características más relevantes.
- `modelo_recocido_solucion.npy`: Array de NumPy que define los índices de las características a utilizar.
- `modelo_recocido_pesos.npy`: Array de NumPy con los pesos optimizados por el algoritmo.

- `modelo_recocido_umbral.npy`: Array de NumPy con los umbrales de activación para cada característica.

6.3.2. Flujo de Inferencia

El archivo `main.py` de esta aplicación orquesta un flujo de inferencia de múltiples pasos para cada URL recibida:

1. **Scraping y Limpieza:** Se extrae el texto de la URL y se aplica la misma función de limpieza de texto utilizada durante la creación del corpus.
2. **Vectorización:** El texto limpio se transforma en un vector numérico utilizando el `vectorizer` cargado.
3. **Selección de Características:** El vector se pasa a través del `selector` para reducir su dimensionalidad.
4. **Clasificación:** Se aplican los `pesos` y `umbral`s sobre el vector reducido para calcular una probabilidad final y emitir un veredicto.

6.4. Prototipo 2: Analizador Basado en Modelos Transformer (Versión Final)

El segundo prototipo, que representa la culminación del proyecto, implementa el modelo DistilBERT de mayor rendimiento. Esta aplicación demostró ser la más fiable y precisa de las dos.

6.4.1. Componentes del Modelo

Este modelo se guarda en el formato estándar de Hugging Face en la carpeta `app/modelo_final_distilbert_es/`. Este formato encapsula de manera eficiente todos los componentes necesarios:

- `tf_model.h5`: Contiene la arquitectura y los **pesos del modelo** ajustados durante el fine-tuning.
- `config.json`: Archivo de configuración que describe la arquitectura del modelo.
- `tokenizer.json`, `vocab.txt`, etc.: Archivos que definen el **tokenizador** exacto, garantizando un preprocesamiento consistente.

6.4.2. Flujo de Inferencia

El proceso de inferencia con el modelo Transformer es notablemente más directo y potente:

1. **Scraping y Combinación:** Se extrae el título y el texto de la URL y se combinan en el formato "título [SEP] texto".
2. **Tokenización:** Se utiliza el `AutoTokenizer` cargado para convertir el texto en los tensores de entrada que el modelo espera.
3. **Predicción:** Los tensores se pasan al modelo `TFAutoModelForSequenceClassification`, que procesa la entrada a través de sus capas de atención y devuelve los *logits* de salida.
4. **Cálculo de Probabilidad:** Se aplica una función Softmax a los *logits* para obtener las probabilidades finales de cada clase (FALSO/REAL) y se emite el veredicto.

6.5. Interfaz de Usuario y Casos de Uso

Ambos prototipos comparten una interfaz de usuario común, definida en el archivo `index.html`, que permite una interacción fluida y proporciona un análisis detallado. A continuación se muestran capturas de pantalla de la aplicación final en funcionamiento.

6.5.1. Caso de Uso 1: Detección de una Noticia Real

En la Figura 6.1, se introduce la URL de una noticia de una fuente verificada. La aplicación extrae correctamente el contenido y, basándose en el análisis del modelo Transformer, emite un veredicto de ****REAL**** con una alta confianza, demostrando la capacidad del sistema para identificar texto legítimo.

6.5.2. Caso de Uso 2: Detección de una Página con Contenido Engañoso

La Figura 6.2 muestra el análisis de una URL con contenido engañoso. El modelo de lenguaje identifica patrones en la redacción (exageraciones, estilo, etc.) que son inconsistentes con el periodismo real y la clasifica correctamente como ****FALSA**** con un alto grado de confianza.

6.5.3. Caso de Uso 3: Detección de una Página Fraudulenta

Finalmente, la Figura 6.3 ilustra un caso donde se introduce una URL de una página de inversión fraudulenta. El modelo, habiendo sido entrenado con ejemplos



Figura 6.1: Captura de pantalla de la aplicación analizando una noticia real.

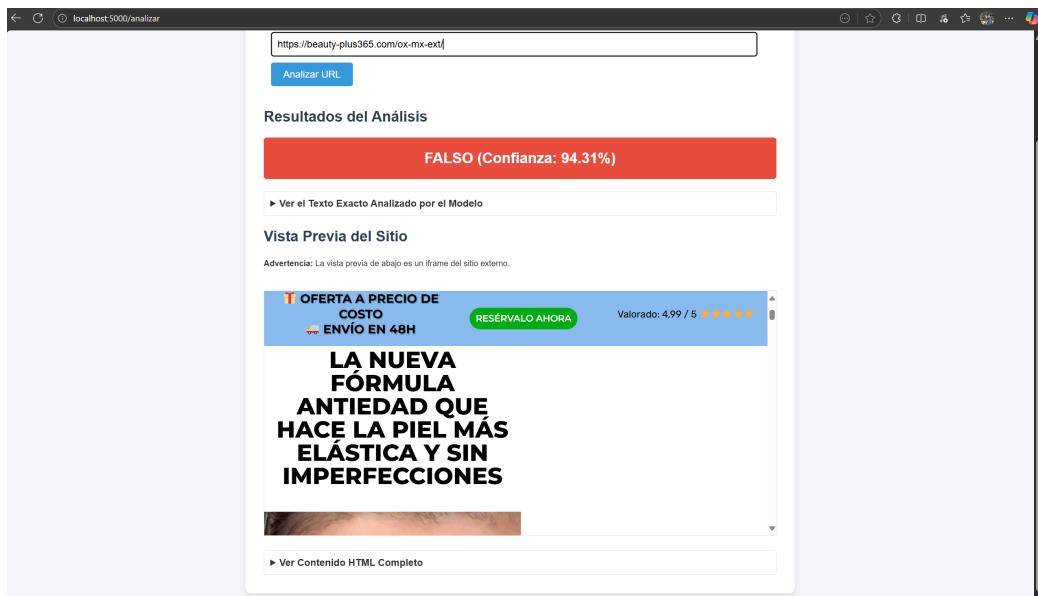


Figura 6.2: Captura de pantalla de la aplicación detectando una noticia falsa basada en su contenido.

similares, detecta el lenguaje de urgencia y las promesas poco realistas, emitiendo un veredicto de ****FALSO**** con una confianza casi del 100 %, demostrando su utilidad como herramienta de prevención de fraude digital.

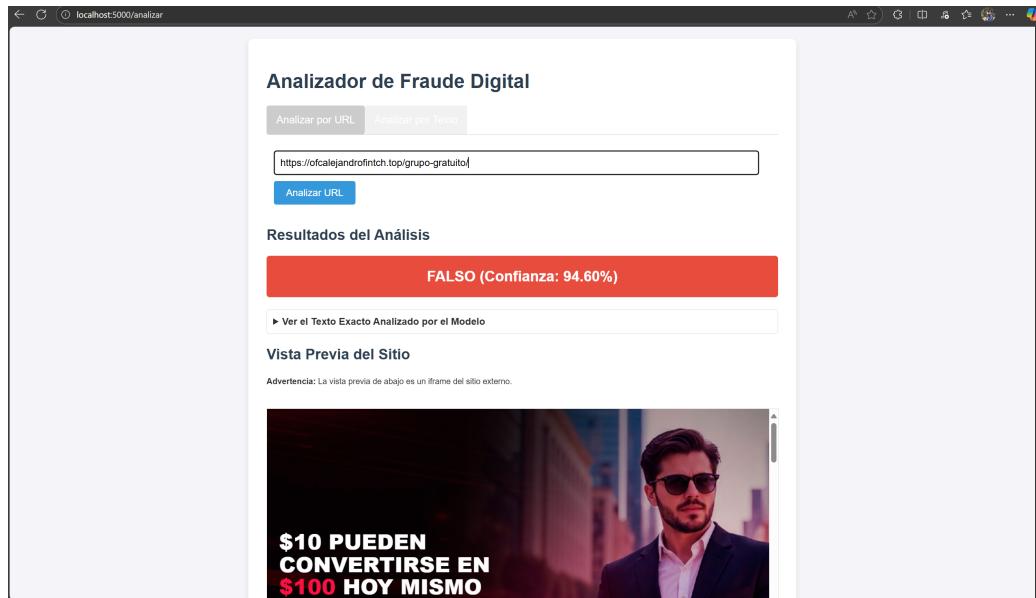


Figura 6.3: Captura de pantalla de la aplicación detectando una página de fraude digital.

Capítulo 7

Conclusiones

El presente trabajo de investigación se propuso desarrollar y validar métodos computacionales para la detección de fraude digital y noticias falsas en español, un área de creciente importancia pero con recursos considerablemente más limitados que en el idioma inglés. A través de un proceso metodológico evolutivo, se exploraron dos paradigmas distintos de la inteligencia artificial, se construyó un corpus a gran escala y se implementaron prototipos funcionales que demostraron la viabilidad práctica de los modelos desarrollados. Este capítulo resume las contribuciones y hallazgos principales de esta investigación y esboza las direcciones para trabajos futuros.

7.1. Resumen del Trabajo y Contribuciones Principales

La investigación culminó con éxito en el cumplimiento de todos los objetivos propuestos, generando varias contribuciones significativas tanto en el ámbito metodológico como en el práctico.

- **Creación de un Corpus a Gran Escala para el Español:** La contribución fundamental de este trabajo fue la construcción de un corpus unificado y balanceado de más de 60,000 noticias en español. Este proceso, que combinó la unificación de cuatro datasets académicos existentes con la adición de datos obtenidos mediante un robusto crawler web, resultó en uno de los recursos más grandes de su tipo para el idioma español, sentando las bases para un entrenamiento de modelos más fiable y representativo.
- **Validación de Enfoques Metaheurísticos (Publicación Científica):** Se implementó y evaluó sistemáticamente una suite de cinco algoritmos metaheurísticos (Recocido Simulado, Búsqueda Dispersa, Algoritmo Genético, VNS y PSO) sobre una representación de Bolsa de Palabras (BoW). Este enfoque demostró ser una vía válida para la optimización de clasificadores, culminando en la publicación de un artículo científico que valida esta fase de la investigación como una contribución independiente al estado del arte.

- **Demostración de la Superioridad de los Modelos Transformer:** El hallazgo central de la tesis es la demostración empírica de que el ajuste fino (*fine-tuning*) de un modelo de lenguaje pre-entrenado, específicamente distilbert-base-multilingual-cased, supera significativamente el rendimiento de los enfoques metaheurísticos en esta tarea. Mientras que los modelos metaheurísticos alcanzaron una exactitud notable, el modelo Transformer logró una **precisión final del 96.2 %** en un conjunto de pruebas completamente aislado, gracias a su capacidad para interpretar el contexto y la semántica del texto.
- **Metodología de Calibración Robusta:** Se desarrolló un pipeline de entrenamiento exhaustivo que incluye una rigurosa calibración de hiperparámetros (tasa de aprendizaje, dropout, etc.) mediante KerasTuner, y la implementación de técnicas avanzadas anti-sobreajuste como EarlyStopping y ReduceLROnPlateau. Este proceso no solo optimizó el rendimiento del modelo final, sino que también generó una metodología documentada y reproducible para futuros experimentos.
- **Implementación de Prototipos Funcionales:** La investigación trascendió el ámbito teórico mediante el desarrollo de dos aplicaciones web funcionales, una para cada enfoque metodológico, utilizando Flask y Docker. La aplicación final, que sirve el modelo DistilBERT, demuestra la viabilidad de convertir el modelo entrenado en una herramienta práctica para el análisis de URLs en tiempo real, completando así el ciclo de vida del proyecto, desde la recolección de datos hasta el despliegue.

7.2. Limitaciones del Estudio

A pesar de los resultados positivos, es importante reconocer las limitaciones de este trabajo, las cuales abren puertas a futuras investigaciones:

- **Dependencia del Contenido Textual:** Los modelos desarrollados se basan exclusivamente en el texto de las noticias. No analizan otros elementos cruciales de la desinformación como imágenes, videos o el perfil de las cuentas que difunden el contenido.
- **Robustez del Web Scraping:** Aunque se implementó un scraper inteligente, su eficacia sigue dependiendo de la estructura HTML de los sitios web, que puede cambiar con el tiempo y variar significativamente entre diferentes fuentes de noticias.
- **Dominio del Corpus:** A pesar de su gran tamaño, el corpus está mayoritariamente compuesto por noticias de dominio general y político. El rendimiento del modelo podría variar en dominios muy especializados como el fraude financiero o la desinformación científica.

En conclusión, este trabajo ha demostrado de manera concluyente la eficacia superior del ajuste fino de modelos Transformer para la detección de noticias falsas en español y ha entregado no solo un modelo de alto rendimiento, sino también un corpus a gran escala y un prototipo funcional que sientan las bases para futuras innovaciones en la lucha contra el fraude digital.

Apéndice A

Anexo 1

// Puede incluir en un anexo: formularios, entrevistas, encuestas, carta de aceptación a revista. Todos los anexos deben ser referenciados.

Bibliografía

- [1] A. Bondielli and F. Marcelloni. A survey on fake news and rumour detection techniques. *Information Sciences*, 497:38–55, 2019. doi: 10.1016/j.ins.2019.05.035. URL <https://doi.org/10.1016/j.ins.2019.05.035>.
- [2] B. Hu, Q. Sheng, J. Cao, Y. Shi, Y. Li, D. Wang, and P. Qi. Bad actor, good advisor: Exploring the role of large language models in fake news detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 22105–22113, 2024. doi: 10.1609/aaai.v38i20.30214. URL <https://doi.org/10.1609/aaai.v38i20.30214>.
- [3] K. Ali and K. Zain-Ul-Abdin. Post-truth propaganda: heuristic processing of political fake news on Facebook during the 2016 U.S. presidential election. *Journal of Applied Communication Research*, 49(1):109–128, 2020. doi: 10.1080/00909882.2020.1847311. URL <https://doi.org/10.1080/00909882.2020.1847311>.
- [4] J. Pérez-Dasilva, K. Meso-Ayerdi, and T. Mendiguren-Galdospín. Fake news y coronavirus: detección de los principales actores y tendencias a través del análisis de las conversaciones en Twitter. *El Profesional De La Información*, 29(3), 2020. doi: 10.3145/epi.2020.may.08. URL <https://doi.org/10.3145/epi.2020.may.08>.
- [5] K. Ali, C. Liu, K. Zain-Ul-Abdin, and M. A. Zaffar. Fake news on Facebook: examining the impact of heuristic cues on perceived credibility and sharing intention. *Internet Research*, 32(1):379–397, 2021. doi: 10.1108/intr-10-2019-0442. URL <https://doi.org/10.1108/intr-10-2019-0442>.
- [6] J. Su, T. Y. Zhuo, J. Mansurov, D. Wang, and P. Nakov. Fake news detectors are biased against texts generated by large language models, 2023. URL <https://arxiv.org/abs/2309.08674>.
- [7] L. Cárcamo-Ulloa, C. Cárdenas-Neira, D. Sáez-Trumper, and C. Toural-Bran. Fake news en Chile y España: ¿cómo los medios nos hablan de noticias falsas? *Journal of Iberian and Latin American Research*, pages 1–18, 2021. doi: 10.1080/13260219.2020.1909849. URL <https://doi.org/10.1080/13260219.2020.1909849>.

- [8] J. Cao, X. Luo, and W. Zhang. Corporate employment, red flags, and audit effort. *Journal of Accounting and Public Policy*, 39(1):106710, 2020. doi: 10.1016/j.jaccpubpol.2019.106710. URL <https://doi.org/10.1016/j.jaccpubpol.2019.106710>.
- [9] I. M. Nasser, A. H. Alzaanin, and A. Y. Maghari. Online recruitment fraud detection using ANN. In *2021 Palestinian International Conference on Information and Communication Technology (PICICT)*, pages 13–17, 2021. doi: 10.1109/PICICT53635.2021.00015. URL <https://doi.org/10.1109/PICICT53635.2021.00015>.
- [10] C. Pulido, L. Ruiz-Eugenio, G. Redondo-Sama, and B. Villarejo-Carballido. A new application of social impact in social media for overcoming fake news in health. *International Journal of Environmental Research and Public Health*, 17(7):2430, 2020. doi: 10.3390/ijerph17072430. URL <https://doi.org/10.3390/ijerph17072430>.
- [11] Fabricio Andrés Zules Acosta. Construcción de un dataset de noticias para el entrenamiento y evaluación de clasificadores automatizados. Trabajo fin de máster universitario en ciberseguridad, Universidad Politécnica de Madrid, 2019. URL <https://doi.org/10.13140/RG.2.2.31181.49126>. ResearchGate.
- [12] J. P. Posadas-Durán, H. Gómez-Adorno, G. Sidorov, and J. J. M. Escobar. Detection of fake news in a new corpus for the Spanish language. *Journal of Intelligent and Fuzzy Systems*, 36(5):4869–4876, 2019. doi: 10.3233/jifs-179034. URL <https://doi.org/10.3233/jifs-179034>.
- [13] M. K. Singh, J. Ahmed, M. A. Alam, K. K. Raghuvanshi, and S. Kumar. A comprehensive review on automatic detection of fake news on social media. *Multimedia Tools and Applications*, 2023. doi: 10.1007/s11042-023-17377-4. URL <https://doi.org/10.1007/s11042-023-17377-4>.
- [14] C. M. Tsai. Stylometric fake news detection based on natural language processing using named entity recognition: In-domain and cross-domain analysis. *Electronics*, 12(17):3676, 2023. doi: 10.3390/electronics12173676. URL <https://doi.org/10.3390/electronics12173676>. Recuperado el 24 de junio de 2024.
- [15] J. Su, C. Cardie, and P. Nakov. Adapting fake news detection to the era of large language models, 2023. URL <https://arxiv.org/abs/2311.04917>.
- [16] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. Language models are few-shot learners, May 2020. URL <https://doi.org/10.48550/arXiv.2005.14165>.

- [17] L. Hu, S. Wei, Z. Zhao, and B. Wu. Deep learning for fake news detection: A comprehensive survey. *AI Open*, 3:133–155, 2022. doi: 10.1016/j.aiopen.2022.09.001. URL <https://doi.org/10.1016/j.aiopen.2022.09.001>.
- [18] Arsenii Tretiakov, Alejandro Martín García, and David Camacho. Detection of false information in Spanish using machine learning techniques. In *Advances in Intelligent Data Analysis and Applications*. Springer, 2022. doi: 10.1007/978-3-031-21753-1_5. URL http://dx.doi.org/10.1007/978-3-031-21753-1_5.
- [19] Y. Blanco-Fernández, J. Otero-Vizoso, A. Gil-Solla, and J. García-Duque. Enhancing misinformation detection in Spanish language with deep learning: BERT and RoBERTa transformer models. *Applied Sciences*, 14(21):9729, 2024. doi: 10.3390/app14219729. URL <https://doi.org/10.3390/app14219729>.
- [20] V. Sanh, L. Debut, J. Chaumond, and T. Wolf. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter, October 2019. URL <https://doi.org/10.48550/arXiv.1910.01108>.
- [21] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M. A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, A. Rodriguez, A. Joulin, E. Gravé, and G. Lample. LLaMA: Open and efficient foundation language models, February 2023. URL <https://doi.org/10.48550/arXiv.2302.13971>.
- [22] Gemini Team, Google. Gemini: A family of highly capable multimodal models, December 2023. URL <https://doi.org/10.48550/arXiv.2312.11805>.
- [23] S. D. Das, A. Basak, and S. Dutta. A heuristic-driven uncertainty-based ensemble framework for fake news detection in tweets and news articles. *Neurocomputing*, 491:607–620, 2022. doi: 10.1016/j.neucom.2021.12.037. URL <https://doi.org/10.1016/j.neucom.2021.12.037>.
- [24] A. Thota, P. Tilak, S. Ahluwalia, and N. Lohia. Fake news detection: A deep learning approach. *SMU Scholar*, 2018. URL <https://scholar.smu.edu/datasciencereview/vol1/iss3/10/>. Recuperado el 05 de julio de 2024.
- [25] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need, June 2017. URL <https://doi.org/10.48550/arXiv.1706.03762>.
- [26] J. Devlin, M. Chang, K. Lee, and K. Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding, October 2018. URL <https://doi.org/10.48550/arXiv.1810.04805>.
- [27] X. Jiao, Y. Yin, L. Shang, X. Jiang, X. Chen, L. Li, F. Wang, and Q. Liu. TinyBERT: Distilling BERT for natural language understanding, September 2019. URL <https://doi.org/10.48550/arXiv.1909.10351>.

- [28] K. Martínez-Gallego, A. M. Álvarez Ortiz, and J. D. Arias-Londoño. Fake news detection in Spanish using deep learning techniques, October 2021. URL <https://arxiv.org/abs/2110.06461>.
- [29] E. Shushkevich, M. Alexandrov, and J. Cardiff. Improving multiclass classification of fake news using BERT-based models and ChatGPT-augmented data. *Inventions*, 8(5):112, 2023. doi: 10.3390/inventions8050112. URL <https://doi.org/10.3390/inventions8050112>.
- [30] Y. Bai, S. Kadavath, S. Kundu, A. Askell, J. Kernion, A. Jones, A. Chen, A. Goldin, A. Mirhoseini, C. McKinnon, C. Chen, C. Olsson, C. Olah, D. Hernandez, D. Drain, D. Ganguli, D. Li, E. Tran-Johnson, E. Perez, J. Kerr, J. Mueller, J. Ladish, J. Landau, K. Ndousse, K. Lukosuite, L. Lovitt, M. Sellitto, N. Elhage, N. Schiefer, N. Mercado, N. DasSarma, R. Lasenby, R. Larson, S. Ringer, S. Johnston, S. Kravec, S. E. Showk, S. Fort, T. Lanham, T. Telleen-Lawton, T. Conerly, T. Henighan, T. Hume, S. R. Bowman, Z. Hatfield-Dodds, B. Mann, D. Amodei, N. Joseph, S. McCandlish, T. Brown, and J. Kaplan. Constitutional AI: Harmlessness from AI feedback, December 2022. URL <https://doi.org/10.48550/arXiv.2212.08073>.
- [31] M. G. R. Anselmo. *Diseño y desarrollo de un método heurístico basado en un sistema socio-cultural de creatividad para la resolución de problemas de optimización no lineales y diseño de zonas electorales*. Tesis de doctorado en ingeniería, Universidad Nacional Autónoma de México, 2013. URL <http://132.248.10.225:8080/handle/123456789/101>. Recuperado el 07 de julio de 2024.
- [32] S. Aqil and M. Lahby. Modeling and solving the fake news detection scheduling problem. In *Studies in computational intelligence*, pages 231–242. Springer, 2021. doi: 10.1007/978-3-030-90087-8_11. URL https://doi.org/10.1007/978-3-030-90087-8_11.
- [33] N. Bacanin, C. Stoean, M. Zivkovic, M. Rakic, R. Strulak-Wójcikiewicz, and R. Stoean. On the benefits of using metaheuristics in the hyperparameter tuning of deep learning models for energy load forecasting. *Energies*, 16(3):1434, 2023. doi: 10.3390/en16031434. URL <https://doi.org/10.3390/en16031434>.
- [34] G. Hurtado Avilés, R. A. Mora-Gutiérrez, and J. A. Reyes-Ortiz. Calibración de hiper-parámetros en algoritmos metaheurísticos para la detección de fraude digital. In M. Tovar Vidal, A. L. Lezama Sánchez, and M. Contreras González, editors, *Avances recientes en procesamiento de lenguaje natural y otras áreas afines*, pages 14–26. Benemérita Universidad Autónoma de Puebla, 2024. ISBN 978-607-5914-56-5.
- [35] S. Hidayattullah, I. Surjandari, and E. Laoh. Financial statement fraud detection in Indonesia listed companies using machine learning based on meta-heuristic op-

- timization. In *2020 International Workshop on Big Data and Information Security (IWBIS)*, pages 79–84, Depok, Indonesia, 2020. doi: 10.1109/IWBIS50925.2020.9255563. URL <https://doi.org/10.1109/IWBIS50925.2020.9255563>.
- [36] J. Horak and A. Sabek. Gaussian process regression's hyperparameters optimization to predict financial distress. *Retos*, 13(26):273–289, 2023. doi: 10.17163/ret.n26.2023.06. URL <https://doi.org/10.17163/ret.n26.2023.06>.
- [37] G. Yildirim. A novel hybrid multi-thread metaheuristic approach for fake news detection in social media. *Applied Intelligence*, 53:11182–11202, 2023. doi: 10.1007/s10489-022-03972-9. URL <https://doi.org/10.1007/s10489-022-03972-9>.
- [38] N. Deshai and B. Bhaskara Rao. Unmasking deception: a CNN and adaptive PSO approach to detecting fake online reviews. *Soft Computing*, 27:11357–11378, 2023. doi: 10.1007/s00500-023-08507-z. URL <https://doi.org/10.1007/s00500-023-08507-z>.
- [39] M. R. Zhang, N. Desai, J. Bae, J. Lorraine, and J. Ba. Using large language models for hyperparameter optimization, 2023. URL <https://openreview.net/forum?id=FUdZ6HE0re>. Recuperado el 06 de julio de 2024.
- [40] F. Alam, A. Barrón-Cedeño, G. S. Cheema, G. K. Shahi, S. Hakimov, M. Hasanain, others, and P. Nakov. Overview of the CLEF-2023 CheckThat! lab task 1 on check-worthiness of multimodal and multigenre content. In *CEUR Workshop Proceedings*, volume 3497, pages 219–235, September 2023. URL <https://dclibrary.mbzua.ac.ae/nlpfp/78/>. Recuperado el 05 de julio de 2024.
- [41] M. E. Aragón, H. Jarquín, M. M. Y. Gómez, H. J. Escalante, L. Villaseñor-Pineda, H. Gómez-Adorno, others, and J. P. Posadas-Durán. Overview of mex-a3t at iberlef 2020: Fake news and aggressiveness analysis in mexican Spanish. In *Notebook Papers of 2nd SEPLN Workshop on Iberian Languages Evaluation Forum (IberLEF)*, Malaga, España, September 2020. URL <https://openreview.net/forum?id=vkjfS0w2hu>. Recuperado el 06 de julio de 2024.
- [42] A. Barrón-Cedeño, F. Alam, T. Caselli, G. Da San Martino, T. Elsayed, A. Galassi, others, and P. Nakov. The CLEF-2023 CheckThat! lab: Checkworthiness, subjectivity, political bias, factuality, and authority. In J. Kamps et al., editors, *Advances in Information Retrieval. ECIR 2023. Lecture Notes in Computer Science*, volume 13982. Springer, Cham, 2023. doi: 10.1007/978-3-031-28241-6_59. URL https://doi.org/10.1007/978-3-031-28241-6_59.
- [43] H. Gómez-Adorno, J. P. Posadas-Durán, G. B. Enguix, and C. P. Capetillo. Overview of FakeDeS at IberLEF 2021: Fake news detection in Spanish shared

- task. *Procesamiento del Lenguaje Natural*, 67:223–231, 2021. doi: 10.26342/2021-67-19. URL <https://doi.org/10.26342/2021-67-19>.
- [44] J. M. Ramírez Cruz, S. Ú. Palacios Alvarado, K. E. Franca Tapia, J. P. F. Posadas Durán, H. M. Gómez Adorno, and G. Sidorov. The Spanish fake news corpus version 2.0 en GitHub. GitHub repository, 2021. URL <https://github.com/jpposadas/FakeNewsCorpusSpanish>. Recuperado el 10 de julio de 2024.
- [45] C. Whitehouse, T. Weyde, P. Madhyastha, and N. Komninos. Evaluation of fake news detection with knowledge-enhanced language models, April 2022. URL <https://arxiv.org/abs/2204.00458>.
- [46] S. Kapunac, A. Kartelj, and M. Djukanović. Variable neighborhood search for weighted total domination problem and its application in social network information spreading. *Applied Soft Computing*, 143:110387, 2023. doi: 10.1016/j.asoc.2023.110387. URL <https://doi.org/10.1016/j.asoc.2023.110387>.
- [47] A. Yasmin, W. Haider Butt, and A. Daud. Ensemble effort estimation with metaheuristic hyperparameters and weight optimization for achieving accuracy. *PLOS ONE*, 19(4):e0300296, 2024. doi: 10.1371/journal.pone.0300296. URL <https://doi.org/10.1371/journal.pone.0300296>.
- [48] K. M. Yazdi, A. M. Yazdi, S. Khodayi, J. Hou, W. Zhou, and S. Saedy. Improving fake news detection using K-Means and support vector machine approaches, January 2020. URL <https://publications.waset.org/10011058/improving-fake-news-detection-using-k-means-and-support-vector-machine-approaches>. Recuperado el 05 de julio de 2024.
- [49] M. F. Mridha, A. J. Keya, M. A. Hamid, M. M. Monowar, and M. S. Rahman. A comprehensive review on fake news detection with deep learning. *IEEE Access*, 9:156151–156170, 2021. doi: 10.1109/ACCESS.2021.3129329. URL <https://doi.org/10.1109/ACCESS.2021.3129329>.
- [50] J. P. Aguirre Quezada. El fraude en México: daños patrimoniales y trabajo legislativo para enfrentarlo. Technical report, Senado de México, 2023. URL <http://bibliodigitalibd.senado.gob.mx/handle/123456789/6051>. Recuperado el 05 de julio de 2024.
- [51] O. H. Alvarez. Fraude laboral en la era digital. *Revista General de Derecho del Trabajo y de la Seguridad Social*, (59), 2021. ISSN 1696-9626. URL <https://dialnet.unirioja.es/servlet/articulo?codigo=8029600>. Recuperado el 06 de julio de 2024.
- [52] J. M. Osorio-Giraldo, C. J. Ropain-Zambrano, A. F. Taba-Pulgarin, and A. de Jesús Osorio-Orozco. 10. proyecto de seguridad para reducir fraudes y robos en marketplace de facebook. In *Coloquio de Investigación Formativa 2023-1*, page 89, 2023. URL <https://ridum.umanizales.edu.co/>

- https://xmlui/bitstream/handle/20.500.12746/6834/CIF%202023-1%20-%20Res%C3%BAmenes%20ejecutivos_RIDUM.pdf?sequence=1&isAllowed=y#page=89. Recuperado el 05 de julio de 2024.
- [53] C. Villamil Arcos. Selección de una técnica de aprendizaje de máquina para la detección de fraude financiero digital enfocado a transacciones no autorizadas o consentidas. Master's thesis, Universidad Nacional de Colombia, 2022. URL <https://repositorio.unal.edu.co/handle/unal/84015>. Recuperado el 05 de julio de 2024.
- [54] L. Ehrlinger and W. Wöß. Towards a definition of knowledge graphs. In *SEMANTiCS (Posters, Demos, SuCCESS)*, 2016. URL https://www.researchgate.net/publication/323316736_Towards_a_Definition_of_Knowledge_Graphs. Recuperado el 06 de julio de 2024.
- [55] G. A. García Robledo, J. A. Reyes-Ortiz, and B. A. González-Beltrán. Interfaz de consulta en idioma español para la búsqueda de información en un ambiente académico. Tesis de maestría en ciencias de la computación, Universidad Autónoma Metropolitana, 2020. URL <https://zaloamati.azc.uam.mx/handle/11191/7812>. Recuperado el 07 de julio de 2024.
- [56] G. Hurtado Avilés, J. M. Villa Vargas, S. Tapia Hernández, A. Á. Rivera Sanabria, and J. M. Jaimes Ponce. Representación ontológica de la arquitectura de control de un robot SCARA utilizando IoT y MATLAB. In M. Tovar Vidal, A. L. Lezama Sánchez, and M. Contreras González, editors, *Avances recientes en procesamiento de lenguaje natural y otras áreas afines*. Benemérita Universidad Autónoma de Puebla, 2024. ISBN 978-607-5914-56-5.
- [57] H. J. Levesque. Knowledge representation and reasoning. *Annual review of computer science*, 1(1):255–287, 1986. doi: 10.1146/annurev.cs.01.060186.001351. URL <https://doi.org/10.1146/annurev.cs.01.060186.001351>.
- [58] E. D. Liddy. Natural language processing. Technical report, Syracuse University, 2001. URL <https://surface.syr.edu/istpub/63/>. Recuperado el 05 de julio de 2024.
- [59] C. E. Manzano-Velasco and L. S. Daza-Rosero. Sistema de alerta temprana para monitorear la propagación de noticias falsas: caso de estudio contexto político colombiano. Trabajo grado, Universidad del Cauca, 2024. URL <http://repositorio.unicauca.edu.co:8080/xmlui/handle/123456789/9485>. Recuperado el 07 de julio de 2024.
- [60] J. Padilla Cuevas, J. A. Reyes-Ortiz, and M. Bravo. Detección y representación de eventos en un ambiente académico inteligente. Tesis de maestría en ciencias de la computación, Universidad Autónoma Metropolitana, 2019. URL <https://zaloamati.azc.uam.mx/handle/11191/6124>. Recuperado el 06 de julio de 2024.

- [61] J. A. Reyes-Ortiz. Extracción de información semántica para la clasificación de servicios web. Tesis de maestría en ciencias en ciencias de la computación, Centro Nacional de Investigación y Desarrollo Tecnológico, 2008. URL <https://www.cenidet.edu.mx/subplan/biblio/seleccion/Tesis/MC%20Jos%20Alejandro%20Reyes%200rtiz%202008.pdf>. Recuperado el 07 de julio de 2024.
- [62] J. A. Reyes-Ortiz, B. A. González-Beltrán, and L. Gallardo-López. Clinical decision support systems: A survey of NLP-based approaches from unstructured data. In *2015 26th International Workshop on Database and Expert Systems Applications (DEXA)*, pages 163–167, Valencia, Spain, 2015. doi: 10.1109/DEXA.2015.47. URL <https://doi.org/10.1109/DEXA.2015.47>.
- [63] S. C. Shapiro. Knowledge representation. In *Encyclopedia of cognitive science*. Wiley, 2006. doi: 10.1002/0470018860.s00058. URL <https://doi.org/10.1002/0470018860.s00058>.
- [64] J. H. Holland. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*. MIT Press, 1992. ISBN 978-0-262-58111-0. Edición MIT Press del trabajo original de 1975.
- [65] J. Kennedy and R. C. Eberhart. Particle swarm optimization. In *Proceedings of ICNN'95 - International Conference on Neural Networks*, volume 4, pages 1942–1948, November 1995. doi: 10.1109/ICNN.1995.488968. URL <https://doi.org/10.1109/ICNN.1995.488968>.
- [66] R. C. Eberhart and J. Kennedy. A new optimizer using particle swarm theory. In *MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science*, pages 39–43, 1995. doi: 10.1109/MHS.1995.494215. URL <https://doi.org/10.1109/MHS.1995.494215>.
- [67] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983. doi: 10.1126/science.220.4598.671. URL <https://doi.org/10.1126/science.220.4598.671>.
- [68] R. Martí, M. G. C. Resende, and P. M. Pardalos. Multi-start methods. In *Handbook of Heuristics*, pages 195–212. Springer, 2018. doi: 10.1007/978-3-319-91086-4_7. URL https://doi.org/10.1007/978-3-319-91086-4_7.
- [69] F. Glover. A template for scatter search and path relinking. In *Lecture Notes in Computer Science*, volume 1363, pages 13–54. Springer, 1998. doi: 10.1007/BFb0026599. URL <https://doi.org/10.1007/BFb0026599>.

- [70] N. Mladenović and P. Hansen. Variable neighborhood search. *Computers & Operations Research*, 24(11):1097–1100, 1997. doi: 10.1016/S0305-0548(97)00031-2. URL [https://doi.org/10.1016/S0305-0548\(97\)00031-2](https://doi.org/10.1016/S0305-0548(97)00031-2).
- [71] T. O’Malley, E. Bursztein, J. Long, F. Chollet, H. Jin, L. Invernizzi, and The Keras team. Hyperparameter tuning with Keras Tuner. The TensorFlow Blog, January 2020. URL <https://blog.tensorflow.org/2020/01/hyperparameter-tuning-with-keras-tuner.html>. Recuperado el 29 de enero de 2020.
- [72] Fabricio Andrés Zules Acosta. DataSet Web scraping noticias verificadas. Kaggle, 2019. URL <https://www.kaggle.com/datasets/zulanac/fake-and-real-news>. Recuperado el 07 de julio de 2024.
- [73] F. A. Zules Acosta. Spanish Fake and Real News. Kaggle, 2019. URL <https://www.kaggle.com/datasets/zulanac/fake-and-real-news>. Recuperado el 10 de julio de 2024.
- [74] Arsenii Tretiakov, Alejandro Martín García, and David Camacho. Noticias falsas en español. Kaggle, 2022. URL <https://www.kaggle.com/datasets/arseniitretiakov/noticias-falsas-en-espaol>. Recuperado el 15 de julio de 2024.
- [75] Y. Blanco-Fernández, J. Otero-Vizoso, A. Gil-Solla, and J. García-Duque. Spanish Political Fake News. Kaggle, 2024. URL <https://www.kaggle.com/datasets/javieroterovizoso/spanish-political-fake-news>. Recuperado el 15 de julio de 2024.
- [76] C. E. Manzano-Velasco. DataSet Web scraping noticias verificadas. Kaggle, 2024. URL <https://www.kaggle.com/datasets/enriquemanzano/dataset-web-scraping-noticias-verificadas/data>. Recuperado el 24 de junio de 2024.