

PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR
FACULTAD DE HÁBITAT, INFRAESTRUCTURA Y CREATIVIDAD

CARRERA: INGENIERIA EN SISTEMAS DE LA INFORMACIÓN

Trabajo de Titulación como requisito previo para la obtención del título de Ingeniero en Sistemas de la Información

IMPLEMENTACIÓN DE UN SISTEMA INTELIGENTE PARA LA DETECCIÓN DE MENSAJES OCULTOS EN IMÁGENES MEDIANTE TÉCNICAS DE ESTEGANOGRAFÍA Y MACHINE LEARNING

Autor: ALAM SAMAEL ROVALINO HERMOSA

Tutor: EDISON JAVIER GUAÑA MOYA

QUITO DM, MARZO 2026

Agradecimiento

Primero agradezco a Dios por darme todas las buenas oportunidades que he tenido en mi vida, le agradezco por permitirme culminar mi tesis, por permitirme culminar mis estudios y estar conmigo en mi corazón día a día permitiéndome seguir adelante sin rendirme. Quiero agradecer a mis padres que son pieza fundamental en mi vida pues sin ellos yo no sería nada y no estaría aquí culminando mi carrera, gracias por nunca abandonarme y gracias por nunca soltarme. Agradezco a mi hermana que a pesar de ser tan complicado nunca me ha dejado de querer, nunca me ha dejado de apoyar y siempre ha estado para mi cuando más la necesito. A mi abuelita Gloria por ser una persona importante en mi vida desde que nací y a mi abuelito Alfredo que se está muy feliz por mis logros desde el cielo.

A mis compañeros por aguantarme y apoyarme a pesar de todos los momentos difíciles que hemos tenido en la carrera. A mis profesores por transmitirme sus conocimientos.

Por último, pero no menos importante quiero agradecer a Danaé por haberme apoyado y ayudado en las buenas y en las malas, ser una compañera incondicional y ser parte fundamental de mi proceso para convertirme en una mejor persona y en un buen profesional, gracias por brindarme todo tu amor y gracias a la vida por haberte puesto a mi lado.

Contenido

Capítulo I: Introducción 1

1.1 Planteamiento del Problema 1

1.2 Antecedentes 3

1.3 Justificación 6

1.4 Objetivos 8

1.4.1 Objetivo General 8

1.4.2 Objetivos Específicos 8

1.5 Alcance y Limitaciones 9

Capítulo II: Fundamentación Teórica y Estado del Arte 12

2.1 Esteganografía Digital 12

2.1.1 Técnicas espaciales y de transformación 14

2.1.2 LSB y otras técnicas comunes 15

2.2 Estegoanálisis 17

2.2.1 Métodos estadísticos y basados en características 18

2.3 Machine Learning para Detección de Esteganografía 19

2.3.1 Algoritmos Clásicos (SVM, Random Forest) 21

2.3.2 Redes Neuronales Convolucionales (CNN) 21

2.4 Aplicaciones Web en Seguridad de la Información 23

Capítulo III: Análisis y Diseño del Prototipo 25

3.1 Análisis de Requerimientos 25

3.2 Arquitectura del Sistema 29

3.3 Diseño del Modelo de Machine Learning 34

3.4 Diseño de la Interfaz Web 39

Capítulo IV: Desarrollo e Implementación 45

4.1 Configuración del Entorno 45

4.2 Implementación del Modelo 48

4.3 Desarrollo de la Ampliación Web 53

4.4 Pruebas y Validación 60
Capitulo V: Conclusiones y Recomendaciones 68
5.1 Conclusiones 68
5.2 Recomendaciones 68
Bibliografía 70

Tabla de Figuras

Figura 1. Proceso general de ocultamiento de información en imágenes mediante técnicas de esteganografía. 2

Figura 2. Ilustración del método atribuido a Heródoto, donde se oculta información en el cuero cabelludo de un mensajero. 3

Figura 3. Representa cómo los bits menos significativos de los píxeles son modificados para ocultar información. 4

Figura 4. Clasificación de técnicas de esteganografía en dominio espacial y de transformación. 5

Figura 5. Arquitectura simplificada de una red neuronal convolucional aplicada al estegoanálisis. 6

Figura 6. Diagrama conceptual del sistema inteligente de detección de esteganografía. 7

Figura 7. Arquitectura general del sistema de detección de esteganografía. 10

Figura 8. Proceso general de la esteganografía digital. 13

Figura 9. Comparación entre técnicas de esteganografía en dominio espacial y dominio de frecuencia 15

Figura 10. Muestra como los bits menos significativos de los pixeles son reemplazados para ocultar información. 16

Figura 11. Pipeline general del proceso de estegoanálisis. 18

Figura 12. Flujo de Machine Learning aplicado a la detección de esteganografía. 20

Figura 13. Arquitectura de una red neuronal aplicada a imágenes, en rojo la capa de salida de una Red Convolucional. 22

Figura14. Arquitectura web del sistema de detección de esteganografía. 23

Figura 15. Diagrama general de casos de uso del sistema. 29

Figura 16. Arquitectura general del sistema. 32

Figura 17. Flujo de procesamiento de una imagen. 33

Figura 18. Pipeline de Machine Learning. 37

Figura 19. Arquitectura simplificada de la CNN. 37

Figura 20. Flujo de interacción del usuario. 41

Figura 21. Estructura de la interfaz web. 42

Figura 22. Pantalla principal del sistema StegoDetect. 43

Figura 23. Interfaz para la inserción de información oculta en imágenes. 44

Figura 24. Interfaz de extracción de payload oculto. 44
Figura 25. Arquitectura tecnología y entorno de desarrollo de StegoDetect. 46
Figura 26. Arquitectura en la nube dentro de Google Drive ejecutado mediante Colab. 47
Figura 27. Organización estructural del proyecto “StegoDetect”. 48
Figura 28. Pipeline general de entrenamiento del modelo CNN. 49
Figura 29. Arquitectura simplificada de ResidualSRNetLite. 50
Figura 30. Análisis de una imagen estaganografiada en el sistema StegoDetect. 52
Figura 31. Flujo general de desarrollo de la aplicación web. 54
Figura 32. Interfaz principal del sistema StegoDetect. 55
Figura 33. Interfaz de inserción de información oculta. 56
Figura 34. Interfaz de análisis integrado. 57
Figura 35. Resultado de análisis de imagen con payload PDF. 62
Figura 36. Reporte PDF integrado generado por StegoDetect. 64
Figura 37. Resultado de imagen externa fuera del dominio del modelo. 65

CAPÍTULO I: INTRODUCCIÓN

1.1 Planteamiento del Problema

En la era digital actual, el desarrollo acelerado de las tecnologías de la información y la comunicación ha generado un entorno digital altamente interconectado, en el cual el intercambio de datos se realiza de forma constante como uno de los medios más utilizados para la transmisión de información, gracias a su facilidad de distribución, almacenamiento y aparente inocuidad. No obstante, este uso generalizado ha introducido nuevas vulnerabilidades en términos de seguridad de la información, muchas de las cuales no son evidentes a simple vista.

Una de las principales amenazas emergentes en este ámbito es la esteganografía digital, técnica que permite ocultar información dentro de los archivos multimedia sin alterar perceptiblemente su apariencia. Al contrario que la criptografía, que pretende proteger el contenido del mensaje, la esteganografía pretende ocultar su existencia, lo que dificulta significativamente su detección (Moya, 2023; Rosas, 2024). Este enfoque ha sido muy estudiado en trabajos como Esteganografía digital de imágenes: Encuesta y análisis de métodos de la actualidad, se hace evidente aquí la variedad de técnicas disponibles y su creciente sofisticación.

Una de las técnicas más usadas es el método LSB (Least Significant Bit) que permite insertar información en los bits de menor valor de los pixeles de una imagen. Este tipo de técnicas en un medio ideal para la transmisión encubierta de información (Romero et al., 2018). En consecuencia, una imagen aparentemente inofensiva puede contener datos sensibles, credenciales o incluso código malicioso, lo que representa un riesgo para la seguridad digital.

Para comprender de manera general este proceso, se presenta el flujo básico de ocultamiento de información mediante técnicas esteganográficas. En esta representación se observa como un mensaje secreto puede ser insertado dentro de una imagen portadora, generando una imagen modificada o estegoimagen que mantiene una apariencia visual a la imagen original.

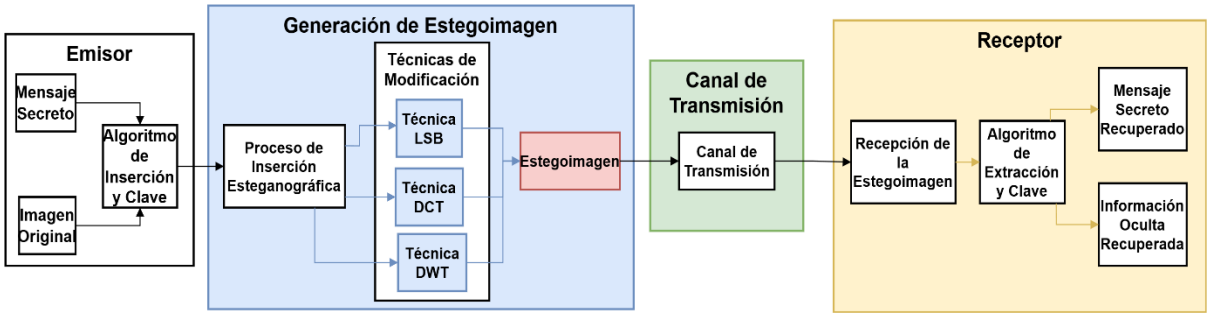


Figura 1. Proceso general de ocultamiento de información en imágenes mediante técnicas de esteganografía.

Este problema llega a hacerse mucho más grande debido a que las herramientas tradicionales de seguridad, como antivirus o sistemas de detección de intrusos, no están diseñadas para identificar este tipo de amenazas. Estas soluciones se enfocan principalmente en patrones conocidos de malware, sin analizar las características internas de los archivos multimedia. Según lo expuesto en *Rich models for stenganalysis of digital images*, los métodos clásicos de detección basados en análisis estadístico presentan limitaciones frente a técnicas modernas de ocultamiento, modelos clásicos como los propuestos por Fridrich y Kodovsky (2012) basados en características estadísticas fueron superados por técnicas más complejas que dificultan la detección mediante análisis convencional. Sabiendo todo esto, cuando se usa modelos de aprendizaje profundo, como Redes Neuronales Convolucionales (CNN), se ha probado que es una alternativa prometedora para la mejora de precisión en detección de contenido oculto (Boroumand et al., 2019; Xu et al., 2016).

Sin embargo, todavía existen desafíos significativos en la implementación de sistemas robustos, eficientes y generalizables en donde se pueda detectar mensajes ocultos en imágenes de forma automatizada. Entre estos desafíos están la variabilidad de las técnicas esteganográficas, la calidad de las imágenes, el formato del archivo, la compresión, el dominio del entrenamiento del modelo y la capacidad del sistema para identificar entre una sospecha estadística y una evidencia directa.

Por lo tanto, surge la necesidad de desarrollar un sistema inteligente que integre técnicas de Machine Learning, mecanismos de extracción LSB y validación de integridad, con el fin de detectar, analizar y recuperar información oculta en imágenes digitales cuando esta sea compatible con el formato del sistema.

1.2 Antecedentes

La esteganografía, entendida como el arte y la ciencia de ocultar información dentro de un medio aparentemente seguro, tiene sus orígenes en la antigüedad, mucho antes del surgimiento de los sistemas digitales. Históricamente, se atribuyen sus primeras manifestaciones a civilizaciones como la griega, donde se utilizaban métodos rudimentarios para transmitir mensajes secretos sin levantar sospechas. Un ejemplo clásico corresponde a la técnica descrita por Heródoto, en la que se raspaba la cabeza de un mensajero, se tatuaba un mensaje en su cuero cabelludo y luego de que el cabello ya volvía a crecer el contenido seguía oculto hasta su recepción (Rosas, 2024). Estas prácticas evidencian que, desde sus inicios, la esteganografía ha estado orientada a garantizar la confidencialidad mediante ocultamiento, en contraste con la criptografía que protege el contenido, pero no disimula su existencia.

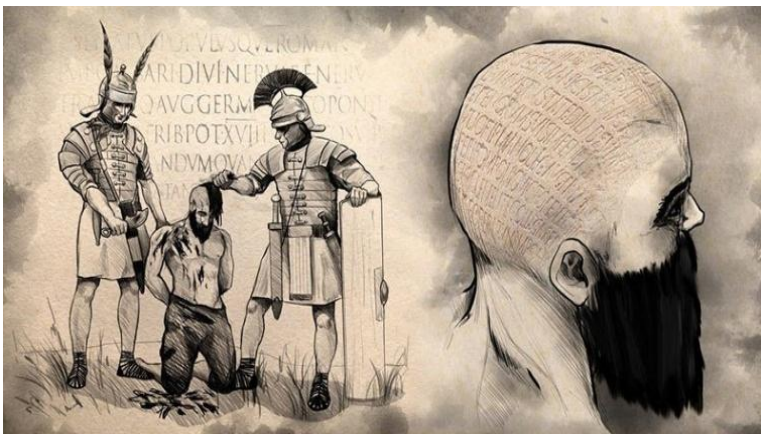


Figura 2. Ilustración del método atribuido a Heródoto, donde se oculta información en el cuero cabelludo de un mensajero.
Tomado de: (Zegarra, 2017).

Con el paso del tiempo, estas técnicas fueron evolucionando hacia métodos más sofisticados durante la Edad Media y el Renacimiento, en donde se empezó a emplear tintas invisibles, mensajes ocultos en pinturas o escritos codificados dentro de textos aparentemente normales. Durante algunos conflictos bélicos, especialmente en la Segunda Gerra Mundial, la esteganografía adquirió mayor relevancia mediante el uso de microdots, que consiste en reducir documentos completos a tamaños microscópicos e insertarlos en imágenes o impresos (Sánchez-Rinza, 2025). Esto marco un punto de transición hacia técnicas más cercanas a la manipulación de medios visuales, sentando las bases para su posterior digitalización.

La llegada de la era digital transformó radicalmente la esteganografía, permitiendo la incorporación de información oculta dentro de archivos multimedia como imágenes, audio y

video. Gracias a esto, las imágenes digitales se convirtieron en el medio más utilizado debido a su alta redundancia de datos y tolerancia a modificaciones imperceptibles (Cheddad et al. 2010). A partir de ese momento, se desarrollaron múltiples técnicas de ocultamiento, destacándose principalmente aquellas en el dominio espacial, como el método de los bits menos significativos (LSB), pues este método consiste en modificar los bits menos relevantes de los pixeles para insertar información sin alterar perceptiblemente la imagen original (Romero et al. 2019).

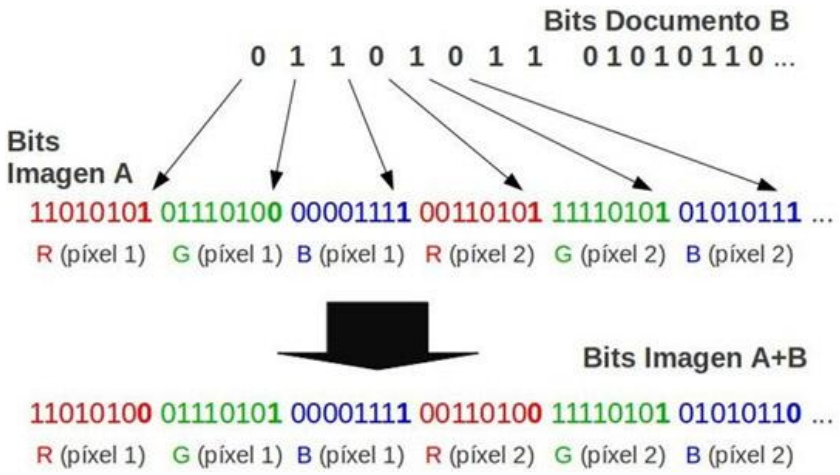


Figura 3. Representa cómo los bits menos significativos de los píxeles son modificados para ocultar información.

Tomado de: (Páez, 2012).

Posteriormente, surgieron técnicas más avanzadas en el dominio de la transformación, como aquellas basadas en la Transformada Discreta del Coseno (DCT) y la Transformada Discreta Wavelet (DWT), que permiten insertar información en los coeficientes de frecuencia de la imagen, logrando mayor resistencia frente a otros procesos de compresión y manipulación (Cheddad et al., 2010; Romero et al., 2018). Estas técnicas representan un avance significativo en términos de robustez y seguridad pues dificultan la detección con solo análisis simples.

En la imagen se sintetiza una clasificación general de técnicas esteganográficas en imágenes digitales, diferenciando los métodos que operan en el dominio espacial de aquellos que trabajan en el dominio de transformación o frecuencia. Esta clasificación resulta relevante porque permite ubicar el método LSB dentro de las técnicas espaciales y comprender su relación con otros enfoques más robustos.

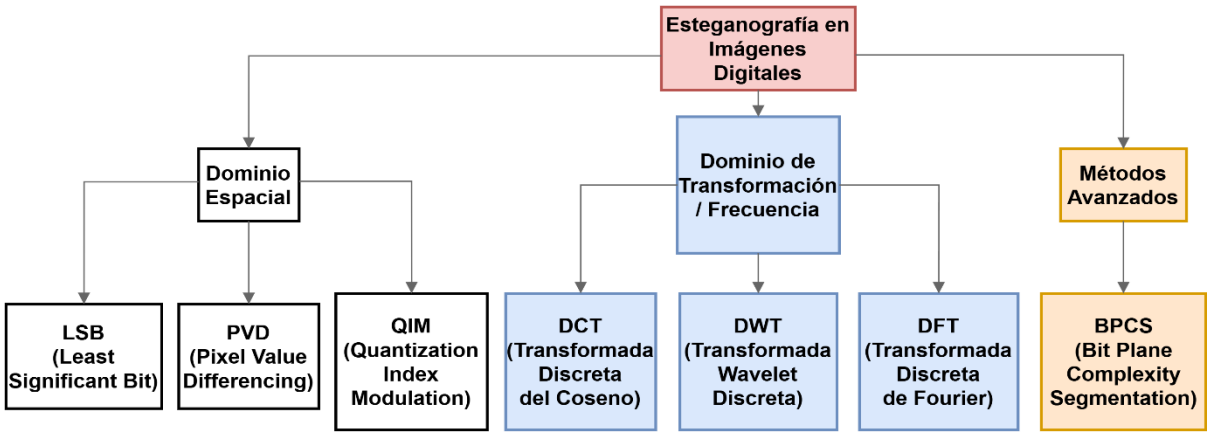


Figura 4. Clasificación de técnicas de esteganografía en dominio espacial y de transformación.

Junto con el desarrollo de técnicas de ocultamiento, surgió el estegoanálisis como disciplina encargada de detectar la presencia de información oculta en medios digitales. Inicialmente, este campo se basó en métodos estadísticos y heurísticos en donde se analizaban características específicas de las imágenes para identificar anomalías. Uno de los avances más relevantes en este ámbito fue el desarrollo de los denominados “rich models”, propuestos por Fridrich y Kodovsky (2012), los cuales permiten extraer un conjunto amplio de características que capturan patrones complejos introducidos por algoritmos esteganográficos.

Sin embargo, el incremento de la sofisticación de las técnicas de ocultación dio lugar la necesidad de métodos de detección más sofisticados. En este sentido, el aprendizaje automático empezó a ser fundamental, facilitando la automatización de los algoritmos de detección que son capaces de aprender patrones discriminativos a partir de grandes cantidades de datos (Ramírez Valle, 2021). Este criterio marco el comienzo de una nueva etapa del estegoanálisis, con el uso de modelos inteligentes.

El punto de inflexión real llegó cuando se incorporaron redes neuronales profundas, especialmente las redes neuronales convolucionales (CNN), ya que éstas han demostrado una gran capacidad para reconocer patrones sutiles en las imágenes. Xu et al. (2016) sugirieron arquitecturas específicas para estegoanálisis, mientras que Boroumand et al. (2019) mejoraron estos modelos con redes residuales profundas, logrando resultados significativamente mejores que los métodos tradicionales. Más recientemente, investigaciones como las de Shehab y Alhaddad (2025) han incorporado modelos híbridos que combinan CNN con redes LSTM, mejorando la capacidad de detecciones en escenarios complejos.

.

La figura muestra una arquitectura simplificada de una red neuronal convolucional para esteganálisis. Esta representación te permite entender de manera general cómo se puede procesar una imagen a través de preprocesamiento, convolución, extracción de características y finalmente clasificación.

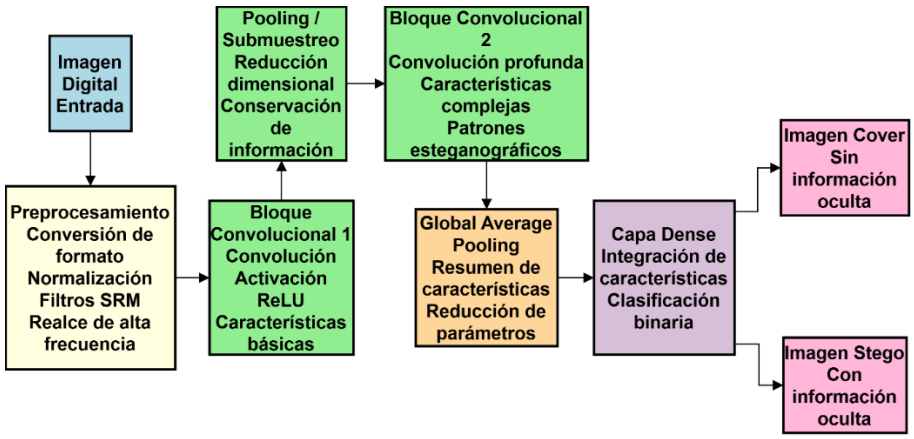


Figura 5. Arquitectura simplificada de una red neuronal convolucional aplicada al estegoanálisis.

Además, trabajos recientes han investigado el uso de técnicas de inteligencia artificial más sofisticadas, como modelos generativos y aprendizaje profundo en espacios latentes, tanto para mejorar la esteganografía como para hacerla más difícil de detectar (Zhou et al., 2025). Este escenario muestra una evolución continua y una especie de "carrera tecnológica" entre los métodos de ocultamiento y las técnicas de detección.

Actualmente, la investigación está en una etapa donde la aplicación el Machine Learning no solo es relevante, sino también necesaria para enfrentar los retos de la esteganálisis moderna. Sin embargo, también se reconoce que los modelos inteligentes deben interpretarse dentro del alcance de su dominio de entrenamiento, ya que una puntuación de clasificación no siempre representa una verdad absoluta al analizar imágenes externas, comprimidas o generadas de otra manera. Por eso, los sistemas actuales tienden a reforzar la detección con enfoques híbridos, combinando análisis estadístico, extracción determinista y validación de integridad.

1.3 Justificación

La construcción de un sistema inteligente para identificar mensajes en imágenes digitales está justificada desde varios puntos de vista. La investigación contribuye al fortalecimiento del conocimiento en áreas como la esteganografía, el esteganálisis y el aprendizaje automático desde un punto de vista teórico, incorporando conceptos y técnicas avanzadas que permiten resolver problemas complejos en seguridad de la información (Rosas,

2024; Fridrich & Kodovsky, 2012). El proyecto también ofrece un enfoque híbrido al problema, ya que no se limita únicamente a la detección estadística usando aprendizaje automático, sino que también incluye técnicas de extracción LSB, validación y análisis contextual de los resultados obtenidos usando SHA-256.

Desde un punto de vista práctico, usar este sistema permitirá el análisis automatizado de información oculta en imágenes. Esto es vital en áreas como la ciberseguridad, la investigación forense digital y la prevención del crimen informático. La capacidad de detectar contenido oculto puede ayudar mucho en la identificación de amenazas y en el análisis de evidencias digitales (Moya, 2023; Sánchez-Rinza, 2025). Sin embargo, los beneficios prácticos del sistema van más allá de solo emitir alertas. También permite recuperar cargas útiles que funcionen con el formato StegoDetect, como textos, documentos PDF, imágenes o archivos binarios, siempre que hayan sido insertados usando el método compatible del sistema.

A continuación, se presenta una visión conceptual del sistema inteligente propuesto. Esta representación nos permite observar la relación entre el análisis basado en aprendizaje automático, la extracción LSB-R, la validación SHA-256 y la decisión integrada del sistema, elementos que diferencian a StegoDetect de un detector convencional basado únicamente en clasificación binaria.

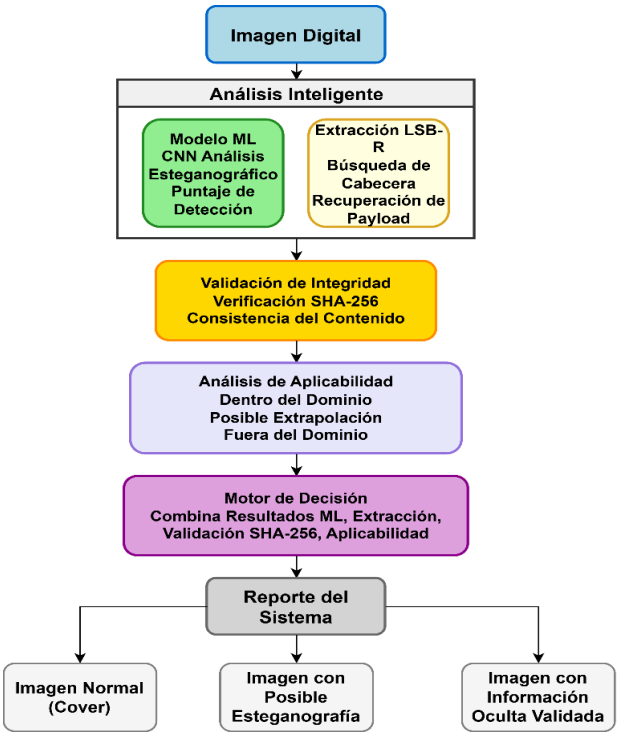


Figura 6. Diagrama conceptual del sistema inteligente de detección de esteganografía.

Esta investigación recomienda el uso de técnicas modernas de aprendizaje automático, especialmente redes neuronales profundas, en el campo tecnológico para lograr una mayor precisión y eficiencia en la esteganálisis. El uso de estas tecnologías en sistemas reales es un gran paso en el desarrollo de herramientas de seguridad digital (Boroumand et al., 2019; Shehab & Alhaddas, 2025). Sin embargo, también se considera necesario interpretar la puntuación del modelo dentro de su dominio de entrenamiento, ya que imágenes externas, comprimidas o de alta resolución pueden producir resultados inconclusos. Por lo tanto, se incluye en el sistema una decisión genérica basada en la combinación de la puntuación del ML, la extracción LSB, la validación SHA-256 y el análisis de aplicabilidad del modelo.

La validación SHA-256 es una parte esencial del sistema porque permite verificar la integridad del contenido recuperado. Esto es lo que diferencia una suposición estadística realizada por el modelo de aprendizaje automático de la evidencia real que se obtiene mediante la extracción determinista. En otras palabras, la prueba de información oculta es más sólida si el sistema encuentra un encabezado coincidente, extrae una carga útil y verifica correctamente su hash, en comparación con solo una predicción del modelo. Esto hace que el sistema sea más técnico y permite resultados más comprensibles para el usuario.

Por último, visto desde un punto de vista social, fortalecer los mecanismos para la detección y validación de información oculta ayuda a proteger la privacidad, la seguridad de los usuarios y la integridad de los sistemas en línea, mientras que medida que las amenazas cibernéticas se vuelven más sofisticadas, el uso seguro de la tecnología requiere herramientas de detección avanzadas. Además, el proyecto contribuye a la formación académica en seguridad informática mostrando cómo las imágenes digitales pueden usarse como un medio de ocultamiento y como objetos de análisis técnico.

1.4 Objetivos

1.4.1 Objetivo General

Diseñar un sistema inteligente híbrido para la detección, análisis y recuperación de mensajes ocultos en imágenes digitales basado en técnicas de esteganografía LSB-R, esteganálisis basado en Machine Learning, extracción de carga útil, validación SHA-256 y generación de una decisión del sistema integrada.

1.4.2 Objetivos Específicos

- Identificar las principales técnicas de esteganografía en imágenes digitales, incluyendo métodos en el dominio espacial y de transformación.

- Analizar modelos de Machine Learning adecuados para el estegoanálisis, con énfasis en redes neuronales convolucionales.
- Diseñar un modelo de detección capaz de identificar patrones asociados a la inserción de información oculta.
- Implementar un sistema que permita el procesamiento de imágenes y la clasificación automática de información oculta.
- Evaluar el rendimiento del sistema mediante métricas como precisión, recall y exactitud.

1.5 Alcance y Limitaciones

La presente investigación se enfoca en el desarrollo de un sistema inteligente capaz de detectar, analizar y recuperar mensajes ocultos en imágenes digitales, utilizando técnicas de esteganografía LSB-R, estegoanálisis basado en Machine Learning y mecanismos de validación de integridad. El sistema trabajara principalmente con imágenes en formato digital (PNG o JPEG, etc.) y se centrara en técnicas de esteganografía ampliamente utilizadas, como LSB, así como en el análisis de patrones estadísticos generados por inserción de información oculta (Cheddas et al. 2010; Romero et al.,2018).

Se presenta la arquitectura general que referencia a un sistema de detección basados en Machine Learning. Esta figura permite contextualizar el rol del procesamiento de imágenes, el entrenamiento del modelo y la generación de predicciones dentro del enfoque general de estegoanálisis; sin embargo, en el presente proyecto dicho enfoque se complementa con módulos adicionales de extracción LSB-R, validación SHA-256 y decisión integrada.

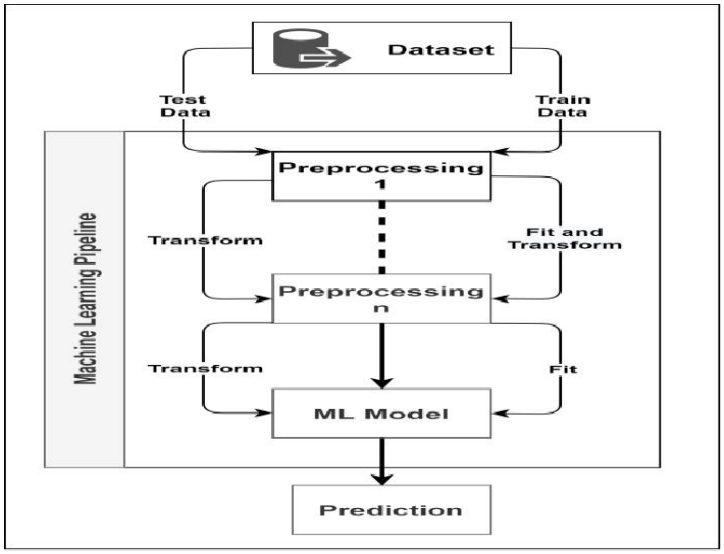


Figura 7. Arquitectura general del sistema de detección de esteganografía.
Tomado de: (Akash, 2022).

El alcance del sistema será analizar imágenes digitales, identificar posibles patrones de esteganografía y recuperar el contenido oculto siempre que haya sido incrustado utilizando el formato compatible con StegoDetect. A diferencia de una herramienta limitada a la detección de la presencia de información oculta, el sistema cuenta con módulos para la inserción y extracción de la carga útil, lo que le permite manejar texto, documentos PDF, imágenes y archivos binarios. Además, emplea la verificación SHA-256 para garantizar la integridad del contenido recuperado.

El sistema también incluye la generación de informes técnicos donde se distinguen la decisión general del sistema, el resultado del modelo de Aprendizaje Automático, la prueba obtenida mediante la extracción de LSB y la aplicabilidad del modelo a la imagen analizada. Esto permite al usuario no solo confiar en una predicción de ML, sino interpretar el resultado basado en diferentes tipos de evidencia. En este sentido, la puntuación del modelo se toma como una señal estadística, mientras que la extracción de una carga útil con un encabezado válido y un SHA-256 correcto es evidencia directa de información oculta compatible con el sistema.

Entre las principales limitaciones están que la extracción del contenido oculto solo puede realizarse cuando la imagen contiene un payload compatible con el formato StegoDetect. Por lo que, el sistema no asegura la recuperación de mensajes insertados por sistemas externos a StegoDetect, algoritmos desconocidos, métodos cifrados o técnicas esteganográficas no

contempladas en el diseño. En esos casos, el modelo de Machine Learning puede generar un puntaje de compatibilidad, pero este no debe interpretarse como una confirmación absoluta.

Otra limitación importante está relacionada con el dominio de entrenamiento del modelo ML. El modelo fue entrenado con un conjunto de datos particular, por lo que su rendimiento podría diferir al trabajar con imágenes del mundo exterior, como: fondos de pantalla, fotos de celular, imágenes JPG comprimidas, capturas de pantalla o archivos de alta resolución. Para reducir la interpretación errónea, el sistema cuenta con un análisis de aplicabilidad del modelo que es capaz de identificar si una imagen pertenece, quizá, fuera o dentro del dominio de entrenamiento. De esta manera, se evita que el sistema proporcione un resultado como concluyente que podría verse afectado por características externas al conjunto de datos utilizado.

La investigación también se limita al análisis de imágenes fijas y no incorpora directamente otros medios como audio o video. La esteganografía se puede aplicar a varios tipos de archivos multimedia, sin embargo, este proyecto se centra en imágenes digitales. Las imágenes digitales son una elección práctica para la esteganografía, existen conjuntos de datos disponibles y son compatibles con las técnicas LSB y viables de implementar dentro de una aplicación web.

No obstante, dichas limitaciones, el sistema propuesto constituye una contribución significativa en el campo de la esteganálisis y la seguridad informática, ya que combina técnicas modernas de inteligencia artificial con mecanismos deterministas de extracción y validación. Esta combinación permite construir una herramienta más completa que un clasificador CNN tradicional, ya que no solo genera un puntaje de detección, sino que también puede insertar, recuperar, validar y documentar información oculta en imágenes digitales.

CAPÍTULO II: FUNDAMENTACIÓN TEÓRICA Y ESTADO DEL ARTE

2.1 Esteganografía Digital

La esteganografía digital es un subconjunto de la seguridad de la información que se ocupa de ocultar información en medios digitales como imágenes, audio o video. La criptografía se trata de encriptar el contenido del mensaje para que no pueda ser leído. La esteganografía, por otro lado, se trata de ocultar la existencia de la comunicación para que no haya sospechas de que se está transmitiendo información sensible (Cheddad et al., 2010; Rosas, 2024). De esta manera, la esteganografía puede considerarse como una técnica complementaria a los sistemas de seguridad modernos, especialmente en caso de necesidad de discreción.

En la era actual del mundo digital con el rápido aumento de los datos multimedia y la conectividad global, la esteganografía ha adquirido una importancia significativa tanto como para aplicaciones legítimas como para fines maliciosos. Por un lado, puede ser utilizada para proteger derechos de autor, implementar marcas de agua digitales y transmitir información confidencial de manera segura; por otro lado, también se ha utilizado en actividades ilegales, como ocultar malware, facilitar comunicaciones clandestinas dentro de redes criminales y evadir sistemas de monitorización (Moya, 2023; Sánchez-Rinza, 2025). Esta dualidad expone un problema importante para la ciberseguridad, ya que la misma técnica puede emplearse tanto para proteger información como para ocultar actividades ilícitas.

La esteganografía digital, desde un punto de vista técnico, es la técnica de modificar un objeto portador mediante un proceso de incrustación que introduce el mensaje secreto, generando así un objeto modificado llamado objeto estego. Este proceso debería garantizar tres propiedades básicas: imperceptibilidad, capacidad de carga útil y robustez frente a ataques o transformaciones (Romero et al. 2018; Oliva Paz, 2020). Optimizar estos parámetros es uno de los principales desafíos en el diseño de algoritmos esteganográficos.

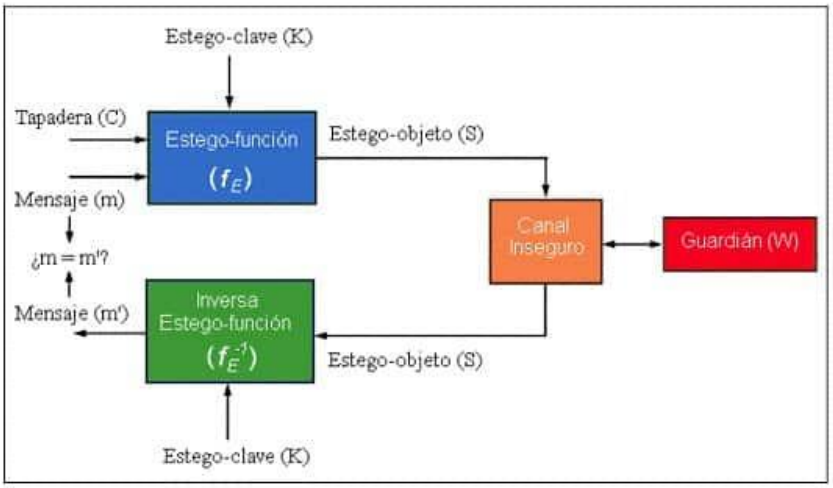


Figura 8. Proceso general de la esteganografía digital.
Tomado de: (AlDr0id, 2016).

En la práctica, los mensajes ocultos no tienen que estar restringidos a texto plano. El diseño del sistema puede hacer que la carga útil consista en texto, documentos, imágenes u otros archivos binarios. Para que esta información se pueda extraer correctamente, debe implementarse una estructura de identificación, como un encabezado, una pequeña cantidad de metadatos y comprobaciones de integridad. Por lo tanto, un sistema de esteganografía debe insertar bits en una imagen y debe permitir al usuario reconocer si la información extraída es de un formato válido, y si esta información fue alterada o no durante el proceso de recuperación.

Para los sistemas contemporáneos enfocados en la esteganografía aplicada, incorporar funciones de hash como mecanismos de validación es deseable. Es posible verificar la integridad de la carga útil extraída mediante el uso del hash SHA-256 en la carga útil después de la extracción y compararlo con el hash registrado desde la inserción de esa carga útil. Este método de validación es crucial, ya que permite determinar si la extracción de la carga útil fue válida, en lugar de una lectura incorrecta de la carga útil. Este principio, en el contexto de este proyecto, proporciona el valor técnico considerable de una extracción LSB validada en contraste con una mera conjetura estadística del modelo de Machine Learning.

El desarrollo de métodos de detección más sofisticados se volvió necesario para contrarrestar técnicas de ocultación cada vez más sofisticadas. Este campo de estudio se llama esteganálisis. La competencia entre estos dos campos se ha convertido en una guerra en constante cambio de tecnología, donde el desarrollo de la inteligencia artificial ha comenzado a desempeñar un papel crítico (Boroumand et al., 2019; Zhou et al., 2025). El enfoque actual, sin embargo, no debe limitarse a identificar si ha ocurrido una alteración. También debe

considerar si la alteración es recuperable, si el contenido es verificable y si la salida del modelo es confiable en el dominio dado.

2.1.1 Técnicas espaciales y de transformación

Los métodos de esteganografía digital pueden clasificarse en dos categorías principales: métodos que funcionan en el dominio espacial y métodos que funcionan en el dominio de la frecuencia. Los métodos que trabajan en el dominio espacial manipulan directamente los valores de los píxeles de la imagen cubierta. Por otro lado, los métodos que trabajan en el dominio de la frecuencia se ocupan de representaciones modificadas de la imagen cubierta en dicho dominio, como los coeficientes de frecuencia (Cheddad et al., 2010). Esta clasificación permite comprender cómo difieren los métodos de inserción simples de los métodos de inserción de alta capacidad y baja complejidad, en comparación con los métodos complejos que requieren más desde el punto de vista computacional.

De los enfoques espaciales disponibles, el método del bit menos significativo (LSB) es quizás el más reconocido debido a su sencillez y alta capacidad de incrustación de datos. El método LSB implica la modificación de los bits menos significativos de los componentes de los píxeles para ocultar datos, y la información puede ser oculta al usuario sin producir cambios notorios en la imagen. Sin embargo, existen debilidades en el método que son el resultado de varias manipulaciones diferentes, incluyendo ataques estadísticos y transformaciones como compresión y filtrado (Rosas, 2024; Oliva Paz, 2020).

Al considerar la resistencia a manipulaciones y ataques, los métodos que utilizan la Transformada Discreta del Coseno (DCT) o la Transformada Discreta de la Wavelet (DWT), pertenecen al dominio de las técnicas de transformación y son más ventajosos. Estos métodos se utilizan en imágenes comprimidas (por ejemplo, en formato JPEG), y debido a que la información está incrustada en los coeficientes de frecuencia, detectar la información embebida resulta más difícil (Romero et al., 2018). Pero su complejidad computacional es mayor y estos demandan procesos más sofisticados de implementación.

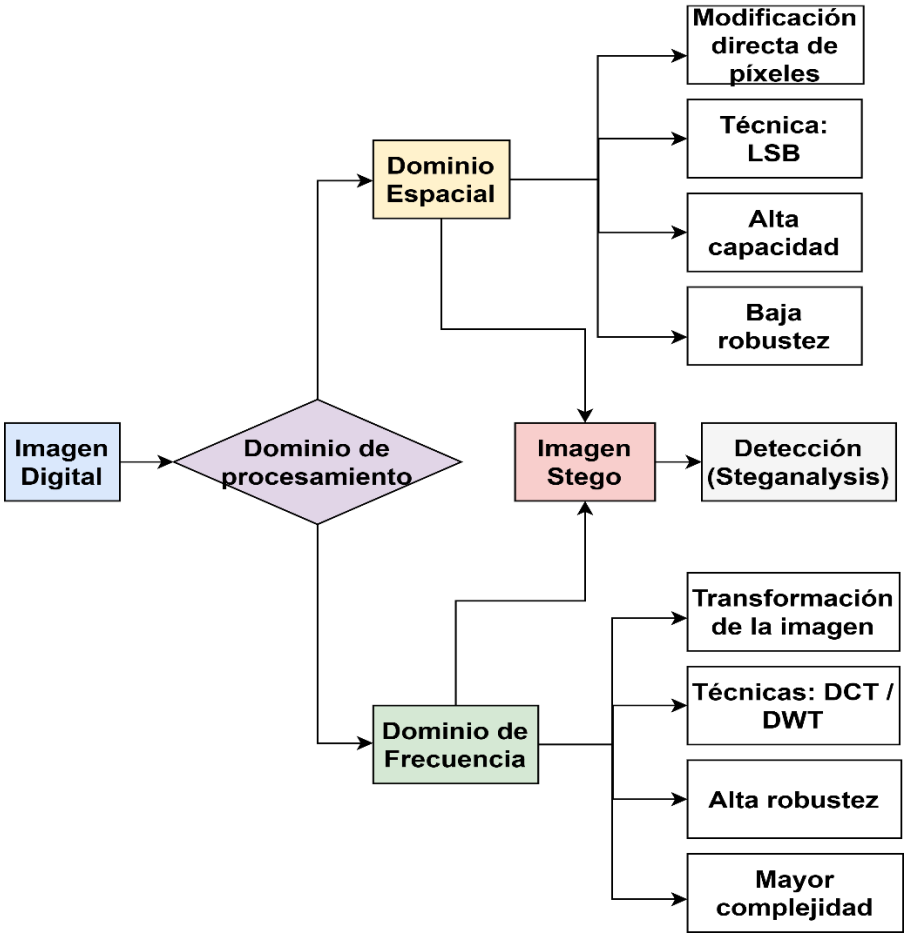


Figura 9. Comparación entre técnicas de esteganografía en dominio espacial y dominio de frecuencia

Al comparar las dos técnicas, las técnicas espaciales se enfocan en la capacidad y la simplicidad, mientras que las técnicas de transformación se enfocan en la robustez y la resistencia a ataques. Esta diferencia inspira la necesidad de una mezcla de técnicas, en la que las aplicaciones actuales ocupen una posición equivalente de importancia para la seguridad y la eficiencia (Xu et al., 2016; Zhou et al., 2025).

Este proyecto implica un enfoque de esteganografía espacial que incorpora el uso de técnicas de inserción y extracción (en este caso, LSB-R) directamente dentro de una aplicación web funcional. Esta elección permite la trazabilidad de los procesos, ya que el sistema tiene la capacidad de registrar el formato de la carga útil, realizar una verificación de integridad y reconstruir los datos cuando se insertan en un formato compatible. También debe indicarse que el uso de esta técnica no puede cubrir todo el espectro de técnicas esteganográficas existentes, y, por lo tanto, el análisis de Machine Learning se incluye como una fuente adicional de evidencia estadística.

2.1.2 LSB y otras técnicas comunes

El algoritmo más emblemático de la esteganografía digital en imágenes es el método del bit menos significativo (LSB). El principio de este método consiste en sustituir el bit menos significativo de un píxel por los bits del mensaje secreto. Este método es muy efectivo para insertar información con una visibilidad muy baja, lo que explica su prevalencia en aplicaciones (Cheddad et al., 2010).

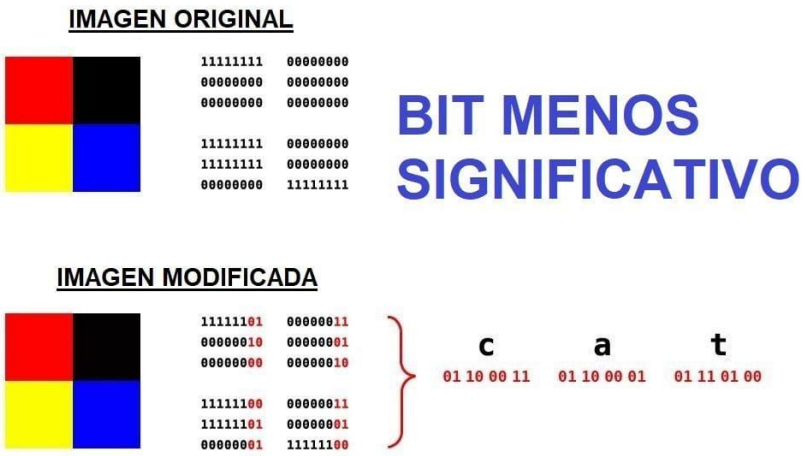


Figura 10. Muestra como los bits menos significativos de los pixeles son reemplazados para ocultar información.

Tomado de: (Esgeeks, 2018).

El LSB clásico sufre de vulnerabilidades que permiten la detección mediante análisis estadísticos. Para contrarrestar esto, se desarrolló el ajuste LSB. La diferencia clave con el ajuste LSB es que, en lugar de reemplazar el bit, se ajusta el valor del píxel de forma aleatoria (Fridrich & Kodovsky, 2012). Este enfoque aumenta la seguridad del método a costa de introducir un poco más de complejidad en el algoritmo.

Otro método pertinente es la Diferencia de Valor de Píxel (PVD). PVD determina cuánto se puede insertar de información en una ubicación dada basada en las diferencias entre píxeles adyacentes. Como resultado, esta técnica maximiza la capacidad de inserción de información mientras mantiene la imperceptibilidad (Ramírez Valle, 2021). Similar a PVD, las técnicas adaptativas contemporáneas utilizan modelos estadísticos y aprendizaje automático para seleccionar regiones para la inserción de información, lo que minimiza la probabilidad de detección (Boroumand et al., 2019).

Para los sistemas aplicados, LSB puede combinarse con otras configuraciones, como la selección de canales RGB y el número de bits utilizados por canal, con el fin de insertar diferentes cargas útiles. Sin embargo, el sistema de extracción debe ser capaz de identificar la configuración utilizada durante la inserción. Consideremos, por ejemplo, una imagen en la que uno o más canales contienen información oculta en uno o más bits. El análisis automático, en caso de que la configuración sea fija, puede determinar la existencia de una carga útil válida. Un sistema más robusto sería capaz de incorporar pruebas controladas de diferentes configuraciones de LSB para la autoextracción y podría concluir, solo después de una prueba exhaustiva, que no existe contenido adecuado.

Además, la presencia de un encabezado distinguible y una suma de verificación SHA-256 en la carga útil hace que el proceso de extracción sea más complejo que solo leer bits. De hecho, esto convierte la extracción en una especie de validación técnica de la información oculta. Este es un aspecto crucial del proceso de validación, pues esto se debe a que permite al recuperador asegurar que el archivo recuperado, de hecho, se relaciona con un mensaje que fue colocado por el sistema y no es ruido aleatorio. Por lo tanto, la extracción de LSB validada significa la primera y única prueba en el análisis, mientras que el modelo de Machine Learning actúa como evidencia estadística de apoyo.

Los métodos más avanzados de esteganografía incluyen la aplicación de inteligencia artificial, técnicas criptográficas y procesos validados de extracción determinista. Es probable que las mejoras en estas técnicas incluyan la creación de sistemas más eficientes y rastreables para el ocultamiento y recuperación de información (Zhou et al., 2025).

2.2 Estegoanálisis

La esteganálisis se refiere a las técnicas utilizadas para identificar el ocultamiento de información en objetos digitales, especialmente en imágenes. Mientras que el objetivo de la esteganografía es ocultar información, el objetivo de la esteganálisis es detectar estructuras anómalas y/o medios electrónicos donde se presume que se ha añadido datos (Fridrich & Kodovsky, 2012). En ese sentido, la esteganálisis es una disciplina complementaria, ya que es una respuesta técnica al problema del ocultamiento.

La esteganálisis es fundamental en la disciplina de la ciberseguridad, ya que aborda las contramedidas del uso nefasto de la esteganografía. El descubrimiento de comunicaciones ocultas es fundamental para la mitigación del ciberdelito, el espionaje digital y la difusión de

malware (Moya, 2023; Sánchez-Rinza, 2025). En consecuencia, es un campo de estudio en auge en el dominio de la seguridad de la información.

Se ha proporcionado un esquema básico del proceso de esteganálisis digital. Indica que el análisis pasará por las etapas de recepción del archivo, preprocesamiento, extracción de características, análisis estadístico, clasificación e interpretación, y generación de informes.

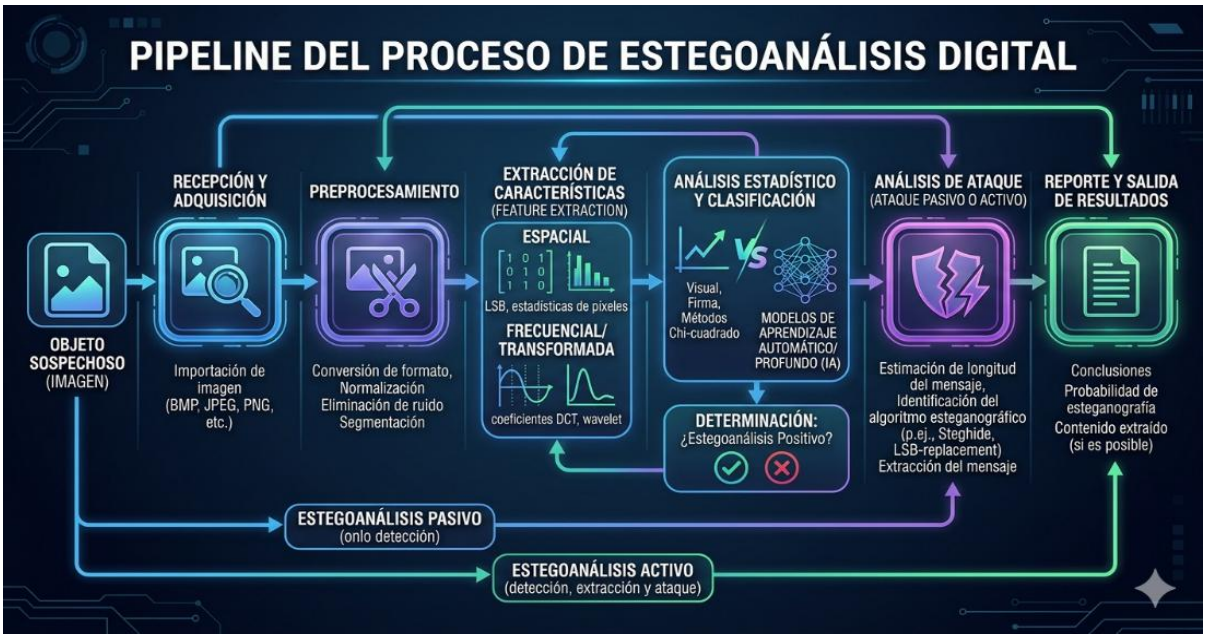


Figura 11. Pipeline general del proceso de estegoanálisis.

Existen dos categorías principales de esteganálisis: activo y pasivo. El esteganálisis pasivo detecta información oculta sin requerir modificaciones en el archivo original. En cambio, el esteganálisis activo busca eliminar o modificar la información oculta aplicando ciertas transformaciones al archivo (Cheddad et al., 2010). Ambos métodos pueden ser útiles en diferentes contextos. En este proyecto, nos centraremos en la metodología pasiva, ya que nuestro objetivo es detectar, analizar y recuperar información oculta cuando está dentro de los límites del formato del sistema sin perder ninguna información del archivo original analizado.

2.2.1 Métodos estadísticos y basados en características

Los métodos estadísticos para el esteganálisis basado en imágenes se concentran en identificar irregularidades en patrones que ocurren de forma natural. Estos métodos analizan las imágenes a un nivel granular mediante el examen de histogramas, correlaciones entre píxeles y ruidos residuales (Fridrich & Kodovsky, 2012). El objetivo final de estos métodos es determinar si las imágenes han sido sometidas a modificaciones externas que puedan ser el resultado de la incorporación de información oculta en la imagen.

Métodos altamente avanzados para extraer miles de características de alta dimensión han llevado al desarrollo de Modelos Enriquecidos (RM) y el Modelo Enriquecido Espacial (SRM) y representan uno de los avances más importantes en el campo (Fridrich & Kodovsky, 2012). Cuando se utilizan para detectar modificaciones muy leves en una imagen en el dominio espacial, estos métodos han demostrado una extrema robustez contra los métodos esteganográficos clásicos.

Los avances en las estrategias de ocultación revelan las desventajas de los enfoques dependientes de la ingeniería manual de características y, por tanto, han motivado el cambio hacia enfoques de aprendizaje automático que son capaces de comprender y procesar formas complejas de ocultación (Boroumand et al., 2019; Xu et al., 2016). Sin embargo, los métodos no sofisticados de análisis estadístico, que buscan explicar cómo y por qué las distribuciones internas de las imágenes se alteran con la introducción de cargas esteganográficas, siguen siendo de suma importancia.

Los modelos híbridos no desechan el análisis residual ni las herramientas estadísticas; al contrario, los absorben para potenciar el aprendizaje profundo o usarlos como módulos de apoyo. Por ejemplo, los filtros pasa-altos permiten resaltar diferencias de alta frecuencia y reducir el peso del contenido visual semántico. Esto resulta útil porque, en estegoanálisis, el objetivo no es reconocer objetos dentro de la imagen, sino identificar perturbaciones sutiles asociadas a la inserción de información.

2.3 Machine Learning para Detección de Esteganografía

El Machine Learning ha revolucionado el campo del estegoanálisis al permitir la automatización del proceso de detección mediante modelos capaces de aprender patrones complejos directamente de los datos. Aquí, los algoritmos de aprendizaje supervisado clasifican las imágenes en limpias o esteganografiadas a partir de un conjunto de datos previamente etiquetado para su entrenamiento (Ramírez Valle, 2021).

Se ofrece una visión general de la aplicación del Machine Learning a la detección de esteganografía. Es posible identificar dos etapas clave del proceso. La primera es la etapa de entrenamiento, donde el modelo se expone tanto a imágenes limpias como a imágenes que contienen contenido esteganográfico. La segunda es la etapa de inferencia, donde el modelo se utiliza para analizar una imagen y producir una predicción o una puntuación que indique el nivel de compatibilidad de la imagen con patrones esteganográficos conocidos.



Figura 12. Flujo de Machine Learning aplicado a la detección de esteganografía.

La aplicación de Machine Learning ha mejorado las técnicas de detección, particularmente en casos avanzados de ocultación. Las relaciones no lineales y los patrones complejos que el análisis estadístico tradicional no puede aprovechar pueden ser utilizados mediante el Machine Learning (Boroumand et al., 2019).

También es importante entender que los resultados producidos por modelos de aprendizaje automático no deben entenderse como un reflejo de la verdad. El modelo se entrena con un conjunto de datos en particular, por lo que, en este caso, el rendimiento del modelo depende de la representatividad del conjunto de datos, de los tipos de imágenes, de las técnicas esteganográficas empleadas, de los niveles de la carga útil y del preprocesamiento utilizado en la fase de entrenamiento. Si la imagen evaluada cae fuera de estas condiciones, puede desencadenar una reacción en cadena que afectará la confiabilidad del puntaje producido.

Por lo tanto, un marco de esteganálisis aplicado debe incorporar interpretación contextual además de los resultados del modelo. En el caso de StegoDetect, la combinación de la puntuación de aprendizaje automático, la extracción LSB-R, la verificación SHA-256 y un análisis de la aplicabilidad del modelo diferencia aquellos casos en los que el resultado del modelo indica sospecha estadística de esos casos en los que el resultado del modelo es evidencia directa.

2.3.1 Algoritmos Clásicos (SVM, Random Forest)

Al enfrentarse a datos de alta dimensionalidad en esteganálisis, tanto Support Vector Machines (SVM) como Random Forest se han consolidado como herramientas tradicionales de gran eficacia. Estos modelos se basan en características previamente extraídas, como las proporcionadas por los Rich Models o por métodos estadísticos especializados (Fridrich & Kodovsky, 2012).

Las SVM son particularmente efectivas en la separación de clases mediante hiperplanos óptimos, mientras que Random Forest ofrece robustez frente al sobreajuste mediante la combinación de múltiples arboles de decisión (Ramírez Valle. 2021). Sin embargo, ambos enfoques dependen fuertemente de la calidad de las características utilizadas, por lo que su calidad de las características no representa adecuadamente las alteraciones por la técnica esteganográfica.

En aplicaciones tempranas de estegoanálisis, estos algoritmos representaron una solución importante por que permitieron automatizar la clasificación de imágenes a partir de patrones estadísticos. No obstante, su desempeño puede verse limitado cuando las técnicas de ocultamiento se vuelven más sofisticadas o cuando las características manuales no logran capturar adecuadamente las alteraciones generadas por el algoritmo esteganográfico. Por esta razón, los modelos de aprendizaje profundo han ganado relevancia en investigaciones recientes, especialmente en escenarios donde las señales esteganográficas son mínimas y difíciles de modelar manualmente.

2.3.2 Redes Neuronales Convolucionales (CNN)

Las Redes Neuronales Convolucionales (CNN) representan el estado del arte en estegoanálisis, ya que permiten aprender automáticamente características relevantes a partir de las imágenes sin necesidad de ingeniería manual. Estas redes utilizan capas convolucionales, funciones de activación y operaciones de pooling para extraer patrones jerárquicos (Xu et al., 2016). En este contexto de esteganografía, dichas características no se orientan a reconocer objetos visibles, sino a identificar alteraciones estadísticas sutiles generadas por la inserción de información oculta.

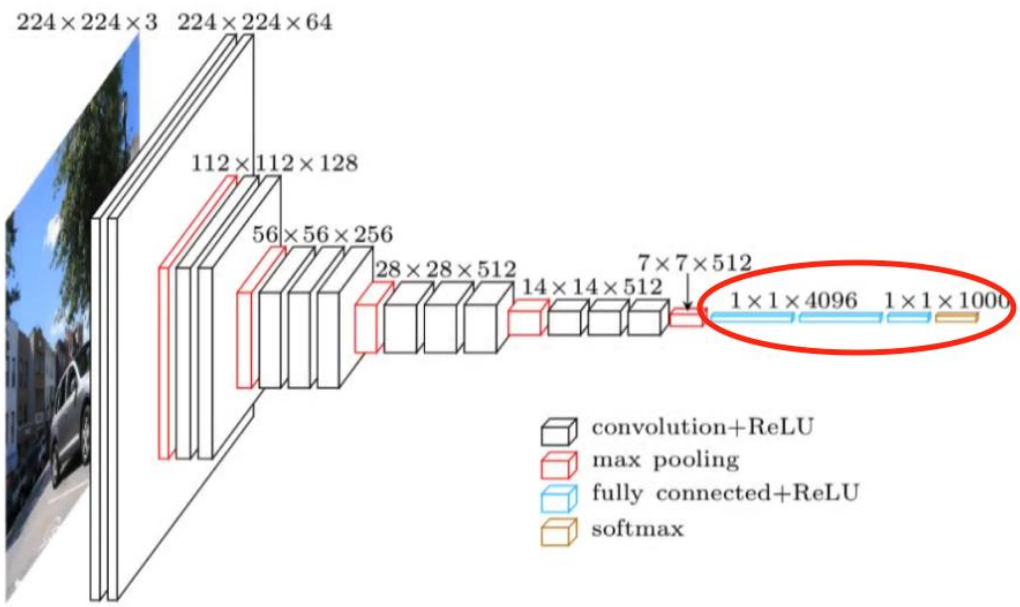


Figura 13. Arquitectura de una red neuronal aplicada a imágenes, en rojo la capa de salida de una Red Convolucional.

Tomado de: (codificandobits, 2019).

Modelos avanzados como las redes residuales han demostrado un rendimiento superior en la detección de esteganografía, logrando identificar modificaciones mínimas en imágenes digitales (Boroumand et al., 2019). Asimismo, enfoques recientes integran arquitecturas hibridas como CNN-LSTM para mejorar la capacidad de detección en contextos complejos (Shehab & Alhaddad, 2025).

En el caso del estegoanálisis, las CNN no se utilizan con el mismo propósito que en la clasificación visual tradicional. Mientras que en tareas comunes de visión artificial se busca identificar objetos, rostros o escenas, en estegoanálisis el objetivo es detectar perturbaciones estadísticas muy pequeñas producidas por la inserción de información oculta. Por esta razón, muchas arquitecturas incorporan filtros residuales, capas de preprocesamiento o estrategias diseñadas para resaltar el ruido de alta frecuencia en lugar del contenido semántico de la imagen.

Al igual que cualquier otro sistema, las CNN también tienen sus limitaciones. Una de ellas es la degradación del rendimiento fuera del dominio del conjunto de datos de entrenamiento. Por ejemplo, un modelo entrenado con conjuntos de datos específicos de imágenes PNG controladas tendrá resultados diferentes cuando se pruebe con imágenes JPG comprimidas o fondos de pantallas de alta resolución. Aunque las CNN tienen una degradación del rendimiento fuera del dominio del conjunto de datos entrenado, es importante mencionar

que sus resultados deben interpretarse como una puntuación, que es una salida condicionada basada en el dominio del conjunto de datos en el que la CNN fue entrenada.

Dado que el enfoque de este proyecto es usar el modelo ResidualSRNetLite como parte de un análisis estadístico dentro de un sistema híbrido, la salida del modelo no está diseñada para ser considerada como una salida concluyente, sino más bien como una puntuación de compatibilidad con los patrones aprendidos durante el entrenamiento. Por esta razón, StegoDetect añade extracción LSB-R, verificaciones SHA-256 y una evaluación de conformidad a la inferencia del modelo, lo cual mejora la claridad de la salida final.

2.4 Aplicaciones Web en Seguridad de la Información

El modelo cliente-servidor de las aplicaciones web contemporáneas es útil para implementar soluciones de seguridad basadas en IA. Permite el despliegue de modelos de aprendizaje automático en el backend y ofrece el front-end a través de aplicaciones web (Oliva Paz, 2021). Este modelo es ventajoso para soluciones que implican el análisis de imágenes, ya que el procesamiento puede realizarse en el backend, mientras que el front-end de la aplicación web permite a los usuarios subir archivos, mostrar resultados y descargar informes.

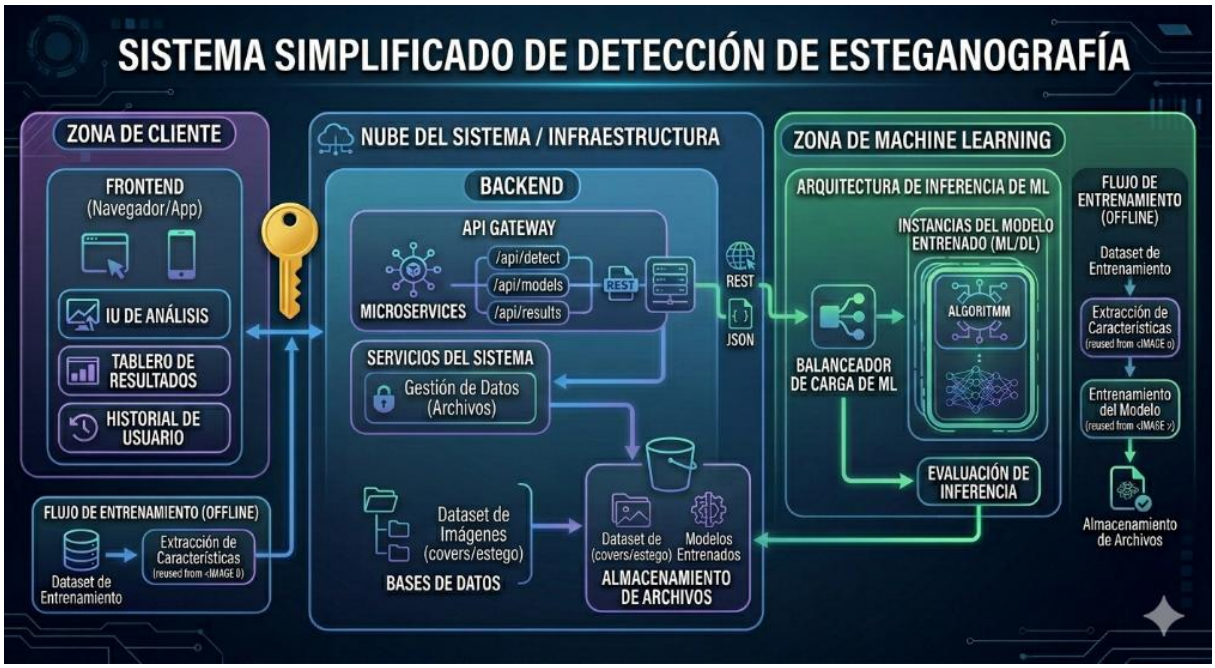


Figura14. Arquitectura web del sistema de detección de esteganografía.

Aquí, las API permiten una interacción rápida entre clientes y servidores. Los marcos contemporáneos permiten a los desarrolladores de aplicaciones construir sistemas escalables y protegidos. Además, técnicas de validación de datos, control de archivos, saneamiento y

manejo de errores, junto con la generación de identificadores únicos de sesión/usuario/tokens de seguridad, ayudan a proteger la información (Romero et al., 2018).

Una buena aplicación web de esteganografía y esteganálisis va más allá de subir imágenes y servicios de predicción simples. Debe categorizar de manera clara los diferentes tipos de evidencia producidos por el análisis. Debe incluir evidencia para la decisión general del sistema, el resultado del modelo de aprendizaje automático, la extracción LSB, la verificación SHA-256, la aplicabilidad del modelo y un informe técnico. Esto ayuda a los usuarios a interpretar mejor la evidencia y reduce la confusión respecto a la puntuación del modelo y su confirmación absoluta.

Desde un punto de vista de ingeniería de software, la arquitectura web ayuda a separar el frontend, backend y los módulos especializados de análisis. El frontend de SPA ayuda al usuario mediante la incorporación de componentes visuales y facilita la inferencia asincrónica del modelo, la extracción de carga útil, las verificaciones SHA-256 y la generación de informes. Este diseño ayuda a la mantenibilidad general del sistema y apoya la implementación de futuras extensiones.

La intersección de esteganografía, esteganálisis y Machine Learning con el desarrollo de aplicaciones web mejorará la capacidad de los sistemas inteligentes para descubrir amenazas ocultas en contextos digitales. Los métodos actuales deben ir más allá de la simple clasificación de cover/stego para incluir extracción de carga útil, verificaciones de integridad, evaluar la aplicabilidad del modelo y proporcionar informes detallados. Esto ayudará a que el sistema cumpla con los requisitos más contemporáneos de seguridad de la información y una perspectiva más amplia del esteganálisis moderno.

CAPÍTULO III: ANÁLISIS Y DISEÑO DEL PROTOTIPO

3.1 Análisis de Requerimientos

Para diseñar herramientas capaces de detectar información oculta en archivos visuales, primero se debe realizar un diagnóstico a fondo que determine qué requisitos técnicos, funcionales y operativos debe cumplir el software. En el contexto de la presente investigación, el sistema "StegoDetect" fue construido como una solución tecnológica especializada en esteganografía y estegoanálisis automático mediante técnicas de Machine Learning, procesamiento digital de imágenes y extracción determinística de información oculta. La necesidad de desarrollar una plataforma de este tipo surge a partir del crecimiento exponencial del intercambio de imágenes digitales en entornos académicos, corporativos y sociales, así como del incremento en el uso de técnicas de esteganografía para ocultar información dentro de archivos multimedia (Cheddad et al; Moya, 2023).

Desde una perspectiva de ingeniera de software, el análisis de requerimientos es una etapa fundamental debido a que permite traducir las necesidades del problema de investigación en especificaciones técnicas concretas y medibles. En este proyecto, el análisis fue realizado considerando tanto los requerimientos funcionales asociados a la detección y extracción de esteganografía como los requerimientos no funcionales relacionados con rendimiento, seguridad, mantenibilidad, usabilidad y escalabilidad del sistema. La arquitectura de la solución se estructuró de forma desacoplada y por módulos. La idea detrás de esto es que se puedan integrar nuevas estrategias de esteganografía, optimizar el componente de Machine Learning y expandir la plataforma a futuro sin tener que rediseñarla por completo.

El sistema propuesto está diseñado como una aplicación web cliente-servidor, que permite a los usuarios subir imágenes digitales en PNG y JPEG para su análisis posterior mediante flujo híbrido. El flujo no se detiene en la ejecución de un modelo de red neuronal convolucional (CNN). Este flujo también abarca la validación de archivos, análisis de la aplicabilidad del modelo, extracción LSB, verificación SHA-256, estimación de cargas útiles compatibles y síntesis de un veredicto del sistema. En este sentido, StegoDetect no es solo un clasificador de cover/stego; funciona como una herramienta para el análisis técnico de la distinción entre evidencia directa y prueba estadística.

El diseño de los requerimientos también nació de los puntos débiles detectados en el software de estegoanálisis actual. La mayoría de las herramientas disponibles hoy en día se quedan cortas porque dependen demasiado de que el usuario configure todo manualmente, baja

escalabilidad, escasa interpretación y poca integración con plataformas web modernas (Fridrich & Kodovsky, 2012; Ramírez Valle, 2021). Por lo tanto, el desarrollo de "StegoDetect" se centró en crear una plataforma única, modular y de acceso sencillo. En lugar de usar herramientas dispersas, el sistema automatiza desde el procesamiento de imágenes y la inferencia con Machine Learning hasta la extracción de los datos ocultos, validando la integridad del archivo y mostrando los resultados en una interfaz integrada.

Además, el sistema fue diseñado para incluir medidas básicas de ciberseguridad. Dado que el sistema procesa imágenes de usuarios externos, existe un riesgo, y por lo tanto se implementaron medidas para gestionar validaciones de tipos MIME, sanitización de nombres de archivo, UUIDs y cálculos de hash SHA-256. Estas medidas se basaron en principios de seguridad defensiva y se centran en los riesgos de exposición de información sensible, corrupción de archivos y manipulación y carga de archivos maliciosos.

El diseño de la interfaz tampoco se dejó al azar; se pensó detalladamente para que la navegación sea intuitiva y la experiencia del usuario resulte cómoda desde el primer momento. Por esta razón, se optó por una arquitectura SPA (Single Page Application) estática basada en HTML5, CSS3, Bootstrap 5 y JavaScript, permitiendo interacciones asincrónicas con el backend mediante Fetch API. Esta aproximación mejora significativamente la fluidez del sistema, reduce tiempos de carga y permite una experiencia de usuario más intuitiva. Además, la interfaz presenta los resultados de forma separada, diferenciando la decisión general del sistema, el resultado del modelo ML, la extracción LSB-R, la validación SHA-256, la aplicabilidad del modelo y la generación del reporte técnico.

Para comprender con mayor claridad la interacción entre los diferentes componentes humanos y técnicos del sistema, en la Tabla 1 se presentan los principales actores que participan en el funcionamiento de StegoDetect, junto con su descripción y la forma en que intervienen dentro del proceso de análisis, extracción y generación de resultados.

Tabla 1. Actores del Sistema.

Actor	Descripción	Interacción principal
Usuario final	Persona que utiliza el sistema para insertar, analizar o extraer información oculta en imágenes digitales.	Carga imágenes, inserta payloads, visualiza resultados y descarga reportes.
Sistema Backend	API REST desarrollada en FastAPI encargada de coordinar validaciones, análisis, extracción y reportes.	Procesa solicitudes, valida archivos, ejecuta análisis ML y gestiona extracción LSB.

Modelo CNN	ResidualSRNetLite entrenado para estegoanálisis dentro de un dominio experimental.	Genera un puntaje ML de compatibilidad con patrones esteganográficos.
Modulo LSB	Servicio encargado de insertar y extraer payloads mediante técnicas LSB-R.	Recupera texto, PDF, imágenes o archivos binarios compatibles con StegoDetect.
Validador SHA-256	Componente de verificación de integridad del payload recuperado.	Confirma que el contenido extraído corresponde a un payload valido.
Generador PDF	Servicio de reportes técnicos del análisis.	Produce reportes integrados con decisión general, ML, LSB y aplicabilidad.

Una vez identificados los actores que intervienen en el sistema, resulta necesario definir las funcionalidades principales que deben cumplir la aplicación. En la tabla 2 se presentan los requerimientos funcionales del sistema, los cuales describen las acciones esenciales que permiten la carga de imágenes, el análisis mediante Machine Learning, la extracción LSB, la validación de integridad y la generación de reportes.

Tabla 2. Requerimientos funcionales.

Código	Requerimiento	Prioridad	Modulo	Justificación
RF-01	Permitir carga de imágenes PNG/JPEG	Alta	Frontend / Upload	Entrada principal del sistema
RF-02	Validar el tipo, tamaño, extensión y contenido real del archivo	Alta	Backend / Seguridad	Prevención de archivos maliciosos o inválidos
RF-03	Generar un identificador único por cada análisis	Alta	Backend	Permite trazabilidad de solicitudes y reportes.
RF-04	Calcular hash SHA-256 del archivo o payload procesado.	Alta	Seguridad / Validación	Garantiza integridad y trazabilidad
RF-05	Ejecutar el modelo CNN para obtener un puntaje ML de esteganografía	Alta	Modelo ML	Permite detectar patrones estadísticos compatibles con esteganografía
RF-06	Insertar información oculta en imágenes mediante LSB-R	Alta	Modulo LSB	Permite realizar funciones de esteganografía, no solo detección
RF-07	Extraer payloads compatibles con el formato StegoDetect.	Alta	Modulo LSB	Permite recuperar texto, PDF, imágenes o archivos binarios ocultos.
RF-08	Validar payloads extraídos mediante SHA-256	Alta	Motor de decisión	Integra evidencia ML, LSB, SHA-256 y dominio del modelo.
RF-10	Evaluar aplicabilidad del	Media	Domain Assessor	Evita interpretar como concluyente

	modelo sobre imágenes externas			un puntaje fuera del dominio
RF-11	Mostrar resultados técnicos interpretables en la interfaz	Alta	Frontend	Mejora compresión del usuario
RF-12	Generar un reporte PDF integrado	Media	Reportes	Documenta evidencia ML, LSB, SHA-256 y conclusión técnica.

Además de las funcionalidades principales, el sistema debe cumplir con condiciones de calidad relacionadas con seguridad, rendimiento, usabilidad, mantenimiento e interpretabilidad. Por ello, en la tabla 3 se detallan los requerimientos no funcionales que orientan el diseño técnico y operativo de StegoDetect.

Tabla 3. Requerimientos no funcionales.

Código	Requerimiento	Categoría	Justificación
RNF-01	Procesamiento eficiente de imágenes	Rendimiento	Reduce tiempos de espera durante análisis e inferencia.
RNF-02	Arquitectura modular	Mantenibilidad	Facilita cambios en modelo, extracción, reportes o frontend.
RNF-03	Validación segura de archivos	Seguridad	Reduce riesgos por archivos corruptos o maliciosos
RNF-04	Separación clara entre ML y evidencia LSB	Interpretabilidad	Evita conclusiones erróneas basadas solo en el score del modelo
RNF-05	Compatibilidad multiplataforma	Usabilidad	Permite acceso desde distintos navegadores y dispositivos.
RNF-06	Manejo controlado de errores	Robustez	Evita fallos visibles y exposición de información técnica sensible
RNF-07	Reportes claros y trazables	Documentación	Permite registrar resultados técnicos del análisis
RNF-08	Escalabilidad futura	Arquitectura	Facilita integración de nuevos modelos o algoritmos esteganográficos

Para representar de manera general las funcionalidades principales del sistema, se presenta en la Figura 15 el diagrama de casos de uso de StegoDetect. Este diagrama permite visualizar cómo el usuario interactúa con los módulos principales de la aplicación, incluyendo la carga de imágenes, la inserción de información oculta, el análisis integrado, la extracción de payloads, la visualización de resultados y la generación de reportes técnicos.

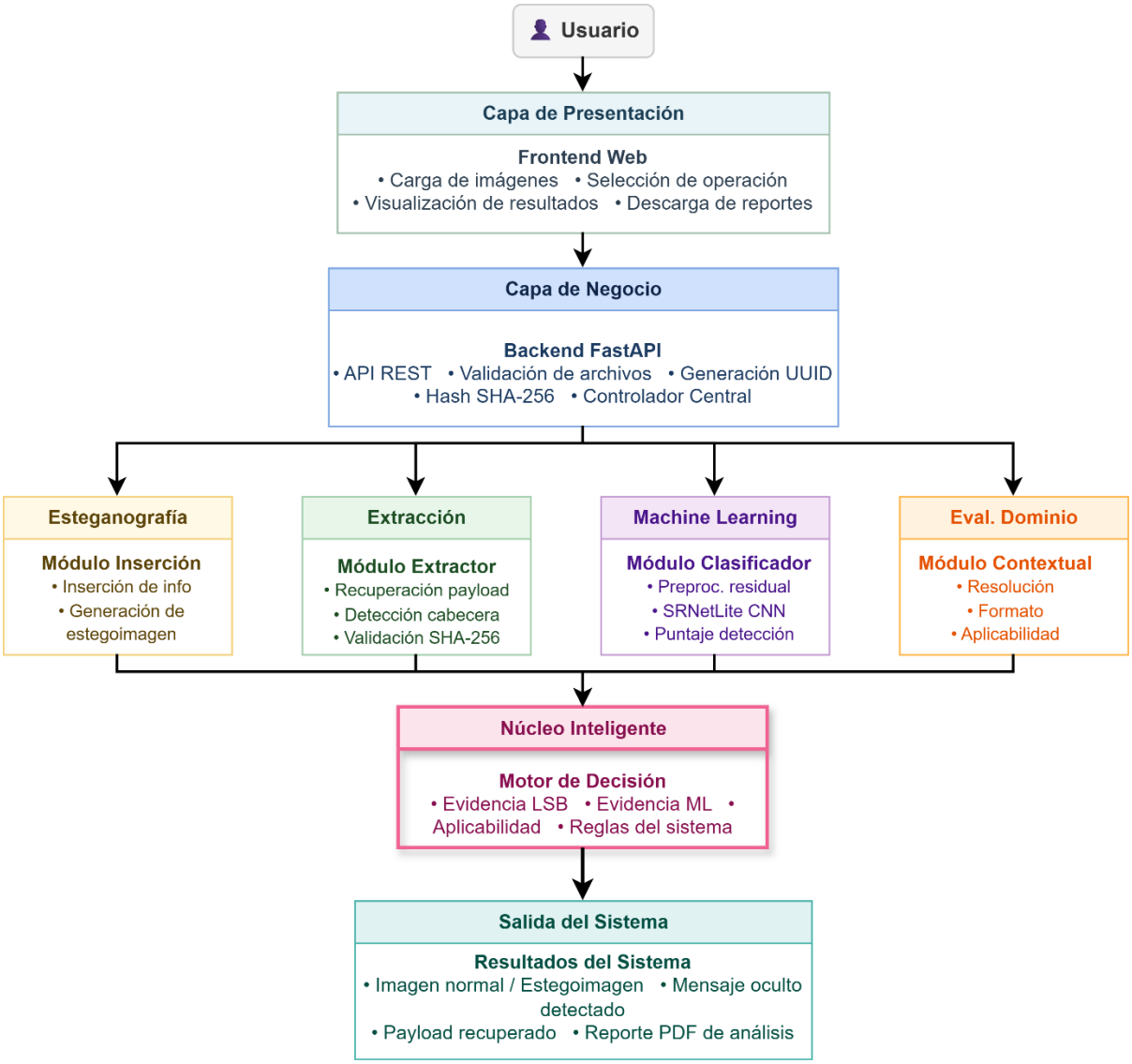


Figura 15. Diagrama general de casos de uso del sistema.

3.2 Arquitectura del Sistema

La arquitectura del sistema "StegoDetect" fue diseñada siguiendo principios de desacoplamiento, modularidad y mantenibilidad propios de arquitecturas empresariales modernas orientadas a servicios. Debido a que la aplicación integra múltiples componentes heterogéneos –frontend web, API REST, procesamiento de imágenes, inferencia de Machine Learning, extracción LSB, validación SHA-256 y generación de reportes– resulta necesario adoptar una arquitectura escalable que permitiera separar responsabilidades de manera clara y organizada.

La solución implementada adopta un enfoque cliente-servidor basado en una API REST desarrollada en FastAPI. El frontend como usa SPA estática encargada de gestionar la interacción con el usuario, mientras que el backend centraliza toda la lógica de negocio,

validación, el almacenamiento temporal, la detección por ML, la extracción de cargas útiles y el reporte final, se logra que cada componente funcione por separado. Esta independencia entre capas asegura que el sistema se pueda actualizar o corregir en el futuro sin alterar toda la estructura.

Uno de los aspectos más importantes de la arquitectura es la implementación de la Arquitectura Limpia. Esto permite la separación de la lógica de negocio de componentes externos como bibliotecas de aprendizaje automático, frameworks o servicios utilizados para la generación de archivos. La estratificación fomenta la reutilización de componentes y mejora la estabilidad del sistema. En el proyecto, la arquitectura de divide en capas de presentación, aplicación, dominio e infraestructura.

La capa de presentación está compuesta por la interfaz web y los Endpoints REST expuestos por FastAPI. La capa de aplicación coordina los casos de uso relacionados con análisis integrado, inserción, extracción y generación de reportes. La capa de dominio contiene las entidades y reglas centrales del sistema, como resultados de análisis, payloads y decisiones integradas. Finalmente, la capa de infraestructura encapsula servicios externos o técnicos, como el modelo ResidualSRNetLite, el módulo de esteganografía LSB, el generador PDF y el almacenamiento de resultados.

En la práctica, el flujo inicia cuando el frontend transfiere la imagen al sistema. Tras comprobar que el formato sea correcto y extraer los metadatos clave, la información pasa al backend, quedando disponible para arrancar las fases de inserción, análisis o extracción según corresponda. En el análisis integrado, el sistema ejecuta el modelo ML para obtener un puntaje de esteganografía, intenta extraer automáticamente payloads compatibles mediante LSB-R, valida la integridad con SHA-256 y evalúa si la imagen pertenece al dominio de entrenamiento del modelo. Finalmente, un módulo de decisión central unifica estos hallazgos para generar un diagnóstico claro y fácil de entender por el usuario.

El diseño contiene funciones de seguridad avanzadas. Para frustrar ataques de recorrido y otros problemas de seguridad relacionados con nombres de archivo, se utilizan UUIDs. Además, el sistema verifica el tipo MIME real e impone un límite de tamaño para contrarrestar ataques de denegación de servicio. El algoritmo de hash SHA-256 se utiliza para preservar la integridad de las cargas útiles y distinguir entre una predicción estadística de un modelo y una prueba legítima.

La organización técnica de StegoDetect se detalla en la Tabla 4, donde se estructuran los componentes arquitectónicos esenciales que dan forma al sistema.

Tabla 4. Componentes arquitectónicos.

Componente	Tecnología	Función
Frontend	HTML5, CSS3, Bootstrap 5, JavaScript	Interfaz para carga, inserción, análisis, extracción y reportes
Backend	FastAPI	API REST y coordinación de lógica del sistema
Modelo ML	PyTorch / ResidualSRNetLite	Genera puntaje ML de compatibilidad con patrones esteganográficos.
Modulo LSB	Servicio de esteganografía	Inserta y extrae payloads mediante técnicas LSB-R
Auto extracción	auto_extract_payload	Evalúa si una imagen pertenece al dominio del modelo ML
Motor de decisión	final_decision	Integra evidencia ML, LSB, SHA-256 y dominio
Reportes	ResportLab / PDF integrado	Generación de reportes técnicos basados en la decisión integrada
Seguridad	UUID, MIME, SHA-256	Garantiza validación, trazabilidad e integridad

La tabla 5 describe los endpoints REST principales que permiten la comunicación entre el frontend y el backend. Esta tabla es relevante porque muestra como cada ruta cumple una función específica dentro del sistema y cómo la API organiza los procesos de carga, análisis, inserción, extracción y generación de reportes.

Tabla 5. Endpoints REST.

Método	Ruta	Entrada	Salida	Propósito
POST	/upload	imagen	upload_id	Carga y validación inicial de imágenes
POST	/analyze	Imagen o ID	Resultado ML	Ejecuta análisis basado en modelo ML
POST	/stego/full-analysis	Imagen	Resultado integrado	Ejecuta ML, extracción LSB, validación SHA-256 y decisión general
GET	/stego/report/{id}	ID de análisis integrado	PDF	Genera reporte integrado
POST	/stego/embed	Imagen + payload	Imagen stego	Inserta texto, PDF, imagen o binario en una imagen.
POST	/stego/extract	Imagen stego + configuración	Payload recuperado	Extrae información oculta usando parámetros indicados

Para complementar la descripción de la arquitectura del sistema, se presenta una vista general del enfoque hibrido de StegoDetect. Esta representación permite observar la interacción entre la interfaz web, el backend FastAPI, los módulos de Machine Learning,

extracción LSB-R, validación SHA-256, evaluación de dominio, motor de decisión integrada y generación de reportes.

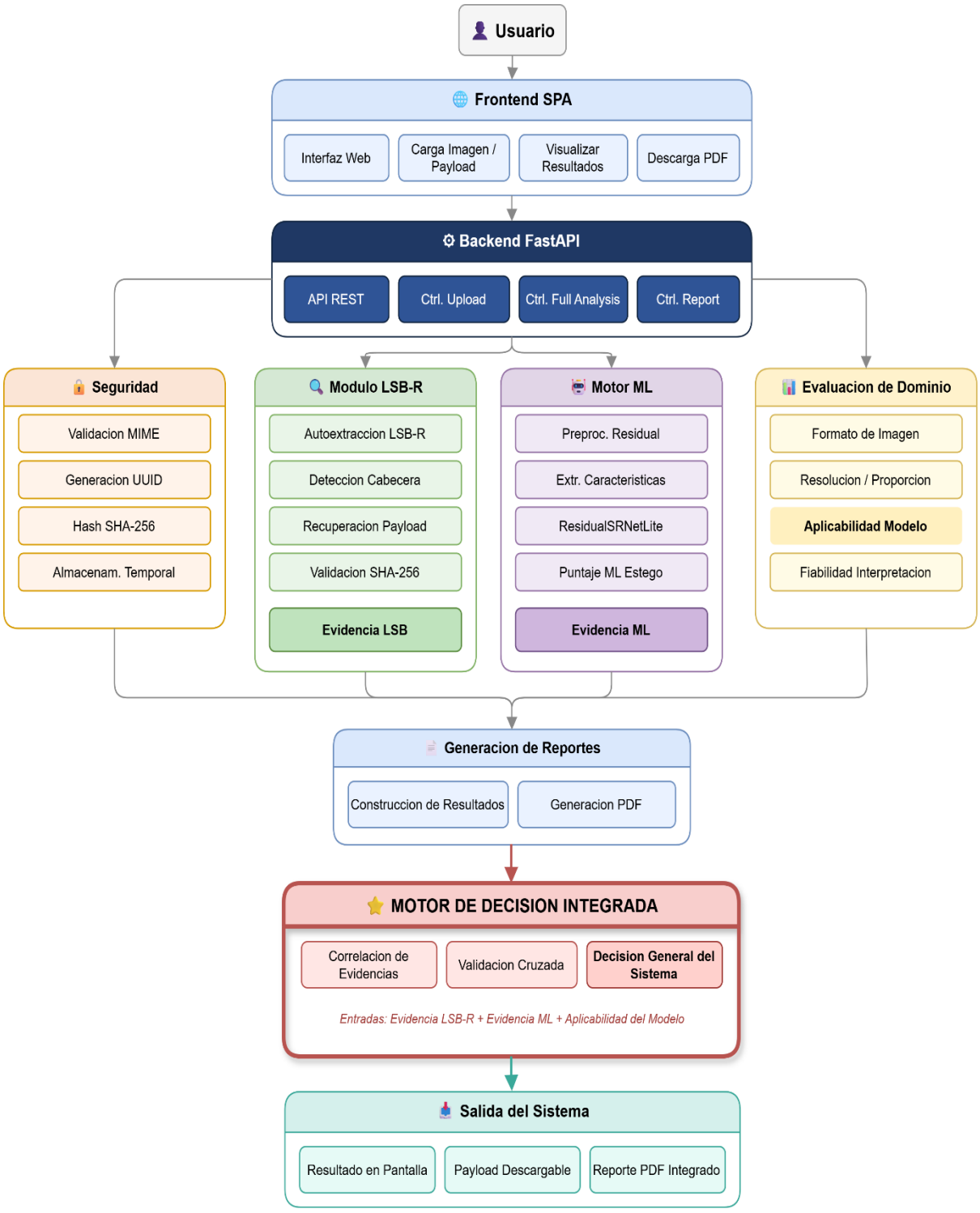


Figura 16. Arquitectura general del sistema.

El flujo de trabajo de procesamiento de imágenes requiere la interacción de múltiples componentes. El flujo de trabajo de procesamiento de imágenes comienza cuando se carga una imagen en la interfaz y termina cuando se genera la decisión general. Esta figura ilustra las

diferentes etapas del procesamiento de imágenes y nos ayuda a entender que el análisis no es solo el resultado del modelo de aprendizaje automático. El análisis también implica la integración de validación, inferencia, extracción, verificación y aplicabilidad.

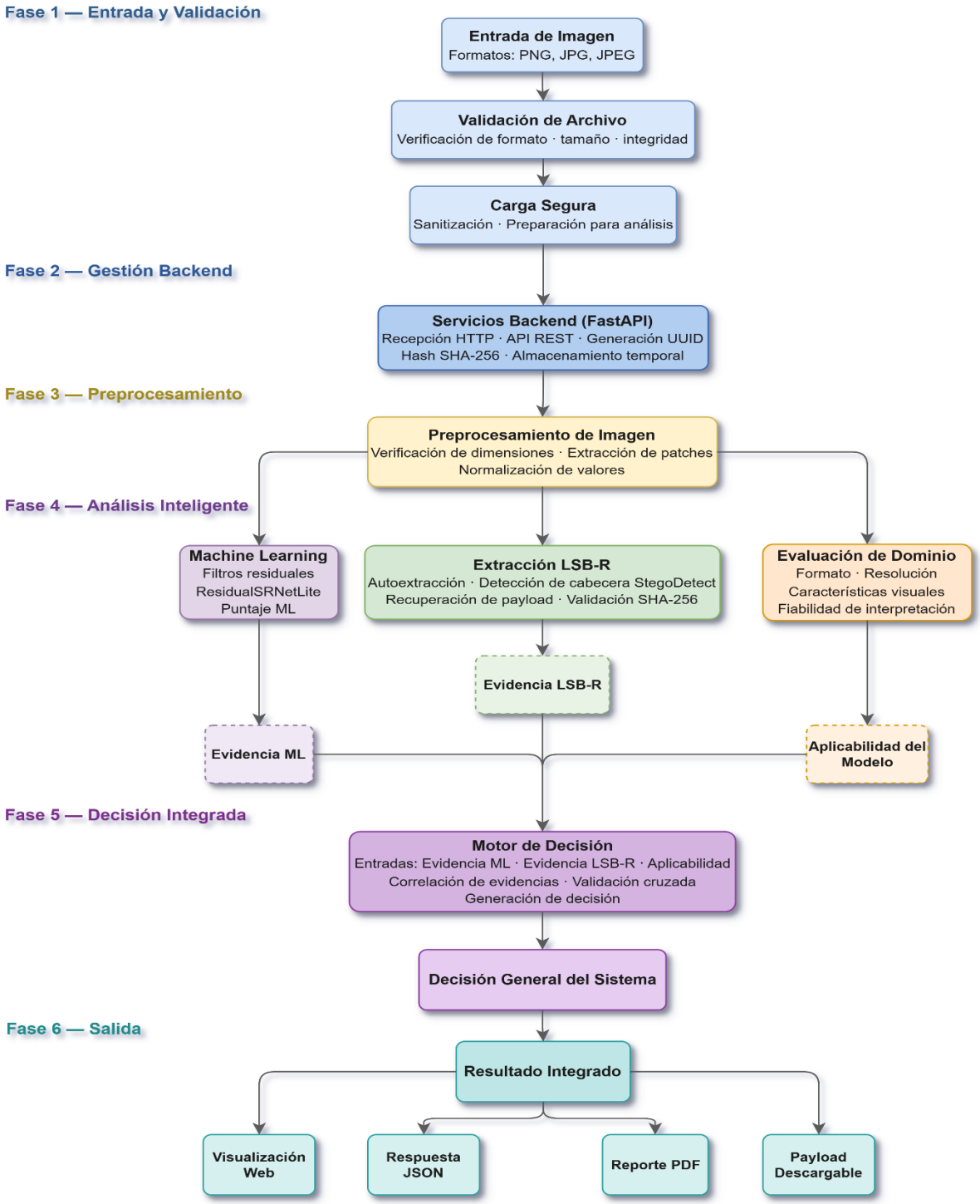


Figura 17. Flujo de procesamiento de una imagen.

El flujo interno del backend comienza con una solicitud HTTP del frontend. El backend procesa la imagen y determina su extensión y tipo MIME y verifica la estructura del archivo

de la imagen. Luego, el backend sanitiza el nombre original del archivo, genera un UUID y calcula el hash SHA-256 del archivo. La sanitización del nombre del archivo, la generación del UUID y el cálculo del hash permiten al backend analizar el archivo y documentar la traza técnica del proceso.

El backend procesa la solicitud según su caso de uso correspondiente después de la primera ronda de validaciones. Para solicitudes de análisis integrados, el backend procesa el modelo ResidualSRNetLite, realiza la extracción automática LSB-R y valida el hash de la carga útil en caso de que haya sido recuperado, además de verificar la aplicabilidad del modelo. Luego, el motor de decisiones integrado serializa la salida al frontend, en caso de un error, el backend genera una respuesta de error controlada que no expone el funcionamiento interno del sistema.

3.3 Diseño del Modelo de Machine Learning

El sistema "StegoDetect" integra un modelo especializado de red neuronal convolucional (CNN) para estegoanálisis en su núcleo analítico. A diferencia de los modelos estándar de CNN utilizados en la clasificación semántica de imágenes, el modelo en cuestión puede identificar cambios sutiles en el ruido residual de la estenografía LSB-R. Este modelo fue desarrollado en respuesta a hallazgos recientes que indican que el análisis de residuos es mucho más efectivo que el análisis visual estándar para los fines del estegoanálisis (Boroumand et al., 2019; Xu et al., 2016).

El problema presentado aquí se refiere a la clasificación binaria supervisada, donde una imagen se asigna a la clase de cubierta (imagen limpia) o stego (imagen que contiene datos ocultos). Este modelo fue desarrollado utilizando imágenes de la base de datos BOSSBase y la inserción LSB-R con diferentes cargas útiles. De esta manera, el modelo es capaz de aprender los patrones estadísticos de los cambios de las modificaciones esteganográficas.

La generación del conjunto de datos se realizó para mantener el equilibrio estadístico en el conjunto y evitar el problema de fuga de datos. Los conjuntos de entrenamiento, validación y prueba se crearon con las imágenes originales intactas, antes de la generación de los fragmentos de imagen. De esta forma, los fragmentos de imagen que se derivaron de la misma imagen no pueden ser, por ejemplo, un conjunto de entrenamiento y validación, ya que esto creará un sobreajuste del modelo y comprometerá la integridad del experimento.

De todos los componentes del sistema, el procesamiento es quizás el más importante. Aquí, se aplica una estrategia de extracción de fragmentos de imagen y análisis de residuo. Una

vez que se han tomado los fragmentos, se aplican filtros de paso alto a los fragmentos para eliminar la mayor parte del contenido semántico y enfatizar los patrones residuales de la manipulación esteganográfica. De esta manera, el modelo puede ser entrenado para detectar la manipulación en lugar del contenido de la imagen.

La arquitectura ResidualSRNetLite del proyecto se basa en modelos Deep Steganalysis SRNet. Esta arquitectura utiliza capas convolucionales, conexiones residuales y normalización por lotes, lo que mejora el flujo de gradientes y ayuda a evitar el problema del gradiente desaparecido. Esto también ayuda al sistema a detectar pequeñas perturbaciones introducidas por las técnicas LSB-R, particularmente en imágenes similares al conjunto de entrenamiento.

No obstante, dentro del diseño actual del sistema, el modelo de Machine Learning no se considera el único responsable de determinar si existe información oculta. Su salida se interpreta como un puntaje ML de compatibilidad con patrones esteganográficos aprendidos. Esto significa que un puntaje alto puede indicar presencia de patrones sospechosos dentro del dominio del modelo, pero no necesariamente constituye una evidencia absoluta en imágenes externas, comprimidas, de alta resolución o con características diferentes a las utilizadas durante el entrenamiento.

Por esta razón, el sistema incorpora una capa de análisis de aplicabilidad del modelo. Este nivel analiza la estructura de la imagen, revisando que el formato, la resolución, las proporciones y la saturación se alineen con los parámetros requeridos para el laboratorio de pruebas. Si la imagen se encuentra fuera del dominio, el resultado del modelo se presenta como no concluyente o de baja fiabilidad. La intención detrás de este diseño es no sentenciar la presencia de información oculta de forma prematura. Esto protege al sistema cuando las alertas del modelo de Machine Learning responden a discrepancias en el dominio de las imágenes analizadas y no a una evidencia real de esteganografía.

Las métricas de evaluación del modelo incluyen precisión, exactitud, recall, puntuación F1 y AUC-ROC, entre otras. Estas métricas son particularmente útiles para la evaluación relacionada con diferentes aspectos del mismo problema y son cruciales en el contexto de la ciberseguridad y problemas de clasificación binaria relacionados. La precisión se refiere al rendimiento en el nivel de predicciones correctas relacionadas con el stego. El recall se relaciona con el rendimiento en la detección de imágenes que ocultan información. La puntuación F1 se centra en el rendimiento en el contexto de la precisión y el recall. El AUC-

ROC se relaciona con el rendimiento respecto a la capacidad del modelo para distinguir entre clases en diferentes umbrales de nivel.

Se presenta a continuación los principales hiperparametros considerados para el entrenamiento del modelo ResidualSRNetLite. Estos parámetros influyen directamente en el aprendizaje de patrones residuales, la estabilidad del entrenamiento y la capacidad de generalización del modelo frente a imágenes del dominio experimental.

Tabla 6. Hiperparametros del modelo.

Parámetro	Valor
Arquitectura	ResidualSRNetLite
Tipo de tarea	Clasificación binaria cover/stego
Dataset base	BossBase
Técnica de inserción	LSB-R
Loss	BCEWithLogitsLoss
Patch size	128x128
Optimizer	AdamW
Métricas	Accuracy, Precision, Recall, F1-score, AUC-ROC
Salida	Puntaje ML de esteganografía

En la tabla 7 se describen las métricas empleadas para evaluar el rendimiento del modelo. Estas métricas son necesarias porque en problemas de ciberseguridad no basta con medir únicamente la exactitud general; también es importante conocer la capacidad del sistema para reducir falsos positivos y falsos negativos.

Tabla 7. Métricas de evaluación.

Métrica	Descripción	Utilidad del sistema
Accuracy	Proporción de predicciones correctas	Evalúa desempeño general del modelo
Precisión	Proporción de positivos predichos correctamente	Reduce falsos positivos en detección stego
Recall	Capacidad de detectar imágenes stego reales	Evalúa sensibilidad del modelo
F1-score	Balance entre precisión y recall	Útil cuando se busca equilibrio entre errores
AUC-ROC	Capacidad de separar clases	Evalúa discriminación entre cover y stego

Debido a que StegoDetect no depende únicamente del modelo ML, la tabla 8 presenta fuentes de evidencia utilizadas por el motor de decisión general. Esta tabla permite diferenciar claramente entre evidencia estadística, evidencia determinística, validación criptográfica y evaluación de dominio.

Tabla 8. Fuentes de evidencia en la decisión general.

Fuente de evidencia	Tipo	Interpretación
Puntaje ML	Estadística	Indica compatibilidad con patrones aprendidos por el modelo

Extracción LSB	Determinística	Recupera payloads compatibles con StegoDetect
SHA-256	Criptografía	Valida integridad del payload extraído
Cabecera StegoDetect	Estructural	Confirma compatibilidad con el formato del sistema
Aplicabilidad del modelo	Contextual	Indica si el puntaje ML debe considerarse confiable

La figura 18 presenta el pipeline general de Machine Learning utilizando en el proyecto. Este flujo inicia con el dataset BOSSBase, continua con la generación de imágenes stego mediante LSB-R, extracción de patches, aplicación de filtros residuales, entrenamiento del modelo ResidualSRNetLite, evaluación mediante métricas y posterior integración con el backend

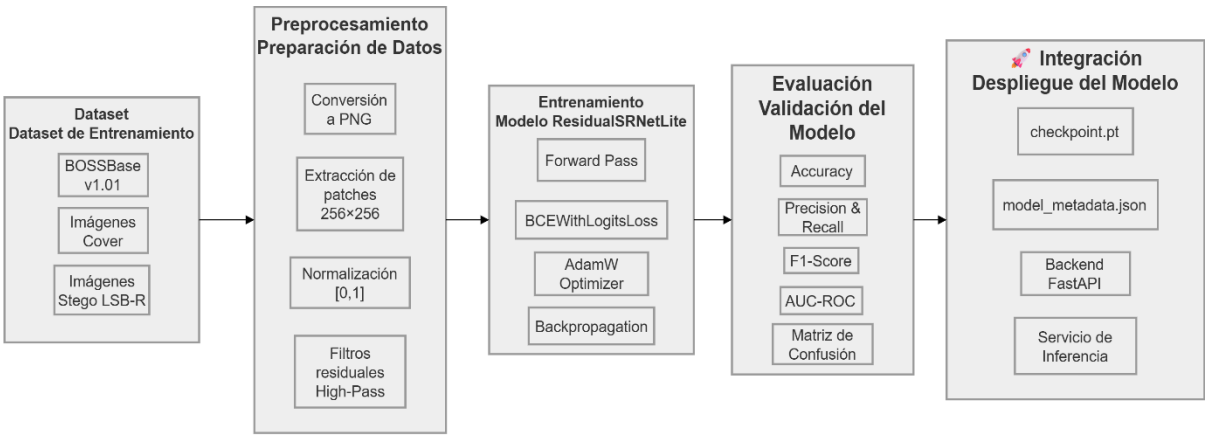


Figura 18. Pipeline de Machine Learning.

La figura 19 presenta una versión simplificada de la arquitectura ResidualSRNetLite. Esta representación permite visualizar las etapas principales del modelo, desde la entrada de la imagen o patch, el preprocesamiento residual, los bloques convolucionales, la reducción de características y la salida como puntaje ML de esteganografía.

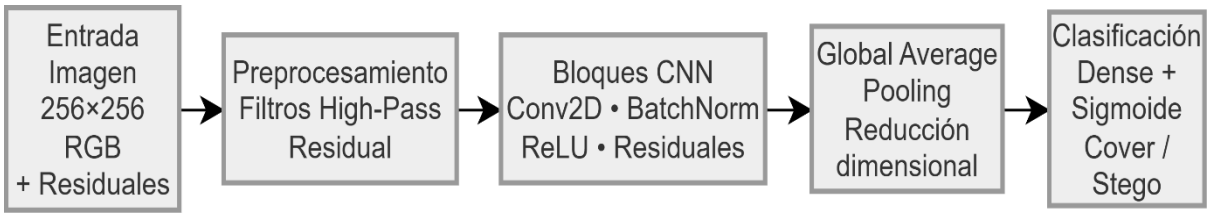


Figura 19. Arquitectura simplificada de la CNN.

Pseudocodigo 1. Flujo de análisis integrado.

Entrada: Imagen PNG/JPEG

Salida: Decisión general del sistema

1. Recibir imagen desde la interfaz web.
2. Validar extensión, MIME y estructura del archivo.
3. Sanitizar el nombre original del archivo.
4. Generar identificador UUID para el análisis.
5. Calcular metadatos básicos y hash SHA-256 del archivo.
6. Ejecutar el modelo ML para obtener puntaje de esteganografía.
7. Evaluar la aplicabilidad del modelo sobre la imagen.
8. Ejecutar auto extracción LSB-R con configuraciones compatibles.
9. Verificar cabecera StegoDetect si existe payload recuperado.
10. Validar integridad del payload mediante SHA-256.
11. Integridad evidencia ML, LSB-R, SHA-256 y domino.
12. Generar decisión general del sistema.
13. Retornar resultado interpretativo y opción de reporte PDF.

Este flujo permite que el sistema produzca un resultado mas robusto que una clasificación binaria tradicional. En particular, cuando se recupera un payload con cabecera válida y SHA-256 correcto, la evidencia determinística tiene mayor peso que el puntaje ML, debido a que confirma la existencia de contenido compatible con StegoDetect.

Pseudocodigo 2. Pipeline de inferencia CNN.

Entrada: Imagen validada

Salida: Puntaje ML de compatibilidad esteganográfica

1. Cargar checkpoint del modelo .pt.
2. Inicializar arquitectura ResidualSRNetLite.
3. Leer imagen de entrada.
4. Convertir imagen a formato compatible.
5. Normalizar valores de píxeles.
6. Extraer patches de tamaño definido.
7. Aplicar filtros residuales y pasa-altos.
8. Ejecutar forward pass en la CNN.
9. Obtener probabilidad por patch.
10. Agregar probabilidades individuales.

11. Calcular puntaje ML final.
12. Retornar probabilidad y clase estimada.

Este pipeline permite que el modelo no analice únicamente la apariencia visual de la imagen, sino los patrones residuales asociados a posibles modificaciones esteganográficas.

3.4 Diseño de la Interfaz Web

La interfaz de “StegoDetect” fue realizada bajo principios modernos de experiencia de usuario y diseño responsivo. Debido a que el sistema está orientado a contextos académicos como demostraciones técnicas, resultaba necesario construir una interfaz simple, intuitiva y altamente funcional. El frontend fue implementado utilizando HTML5, CSS3, Bootstrap 5, JavaScript, manteniendo una estructura SPA estática que permite interactuar con el backend de forma asincrónica.

El diseño SPA estático permite reducir tiempos de carga y mejorar la interacción entre usuario y sistema. Todas las operaciones relacionadas con carga de imágenes, inserción de información, análisis integrado, extracción de payloads y generación de reportes se realizan mediante comunicación asíncrona utilizando Fetch API. Este enfoque elimina recargas completas de página y proporciona una experiencia mucho más fluida para el usuario final.

La interfaz fue diseñada para separar claramente las funciones principales del sistema. A los usuarios se les ofrecen opciones para incrustar datos en imágenes utilizando una carga útil de su elección, ya sea un texto, PDF, imagen o archivo binario. Los usuarios también pueden enviar una imagen al sistema para obtener una respuesta generalizada. Además, los usuarios pueden recuperar los datos ocultos usando la configuración correcta o detectada. Esta capacidad demuestra la flexibilidad de la aplicación, ya que combina tanto la esteganografía como el esteganálisis en una sola plataforma.

El flujo de análisis utiliza un formato diferenciado para mostrar los resultados. La sección principal comunica la conclusión general del sistema, que puede ser "Mensaje oculto encontrado", "Evidencia no detectable", "Patrones posibles compatibles con esteganografía" o "Resultado no concluyente de ML en imagen externa". Los bloques técnicos que aparecen debajo de esta conclusión muestran los resultados del modelo ML, la extracción LSB, la validación SHA-256, la aplicabilidad del modelo y la posibilidad de descargar la carga útil recuperada. Este formato evita que los usuarios asuman que la puntuación del modelo comunica la verdad final.

Incorporar un diseño claro e interpretable fue el principal enfoque desde la perspectiva de UX/UI. Este enfoque incluyó la incorporación de indicadores de progreso y mensajes de error que pueden ser fácilmente corregidos. La interfaz también diferencia entre evidencia estadística y evidencia directa. Por ejemplo, incluso si la puntuación de ML es baja, una carga útil que posee el encabezado StegoDetect junto con un SHA-256 válido se presenta con alta confianza.

En StegoDetect, como se ilustra en la Tabla 9, los principales elementos de la interfaz de la aplicación han sido claramente documentados para que cada componente visual pueda ser correlacionado con las funciones del sistema, incluyendo las opciones y tipos de interacción disponibles para los usuarios.

Tabla 9. Componentes de interfaz.

Componente	Función
Area de carga	Permite seleccionar imágenes PNG, JPG o JPEG
Módulo de inserción	Permite ocultar texto, PDF, imágenes o binarios mediante LSB-R
Panel de análisis	Muestra la decisión general del sistema
Bloque ML	Presenta puntaje ML, umbral y fiabilidad de interpretación
Bloque LSB	Presenta cabecera, payload encontrado, tipo y SHA-256
Bloque de domino	Indica si la imagen está dentro o fuera del dominio del modelo
Botón de descarga	Permite descargar payload recuperado o reporte PDF integrado
Alertas de estado	Muestra errores, advertencias o resultados.

La tabla 10 describe la interacción entre frontend y backend. Esta tabla permite comprender como cada acción visual se traduce en una solicitud HTTP y como la respuesta del backend se refleja en la interfaz de usuario.

Tabla 10. Interacción frontend-backend.

Acción	Endpoint	Resultado
Cargar imagen	/upload	Registro inicial del archivo
Analizar imagen	/stego/full-analysis	Decisión integrada ML + LSB + domino
Generar reporte integrado	/stego/report/{id}	PDF técnico con resultado completo
Insertar payload	/stego/embed	Imagen stego generada
Extraer payload	/stego/extract	Texto, PDF, imagen o binario recuperado
Ejecutar análisis ML base	/analyze	Puntaje ML individual

La figura 20 presenta el flujo de interacción del usuario dentro del sistema. Esta representación muestra las acciones principales disponibles en la interfaz, desde cargar una imagen hasta insertar información, analizar resultados, extraer payloads y descargar reportes.

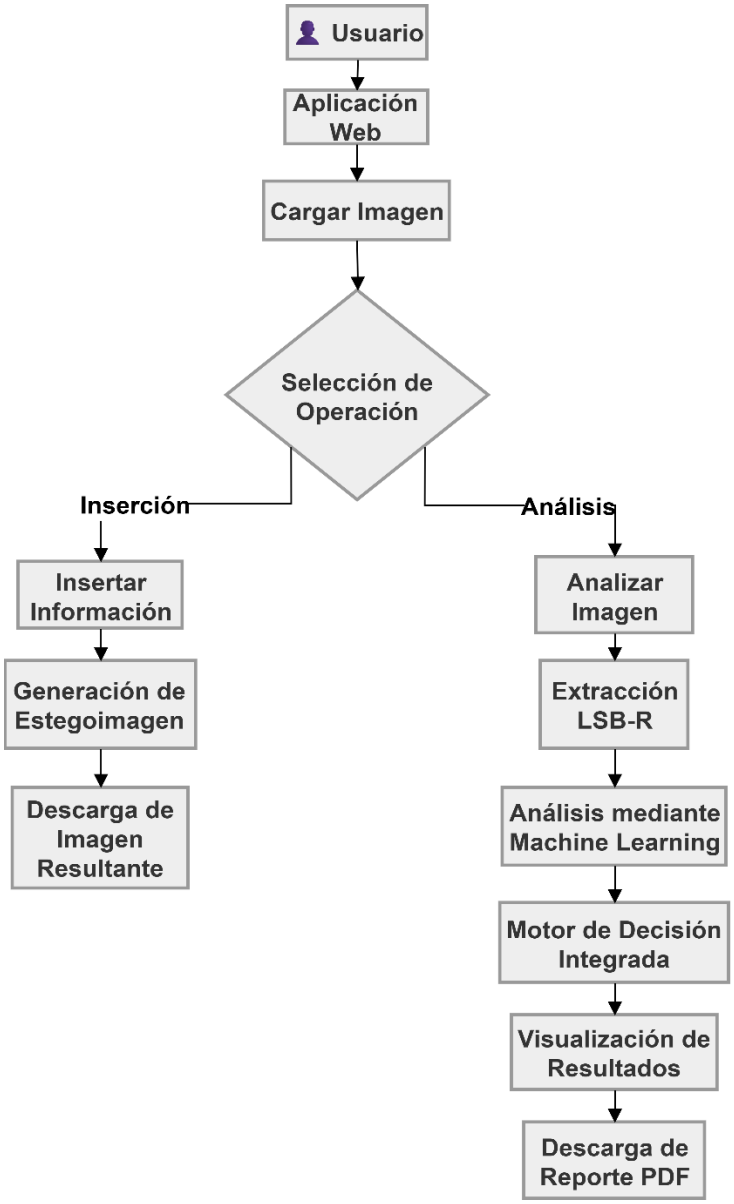


Figura 20. Flujo de interacción del usuario.

La Figura 21 muestra la estructura general de la interfaz web. Esta figura permite observar la organización modular de la pantalla, incluyendo carga, inserción, análisis integrado, resultados ML, resultados LSB-R, aplicabilidad del modelo y reportes.

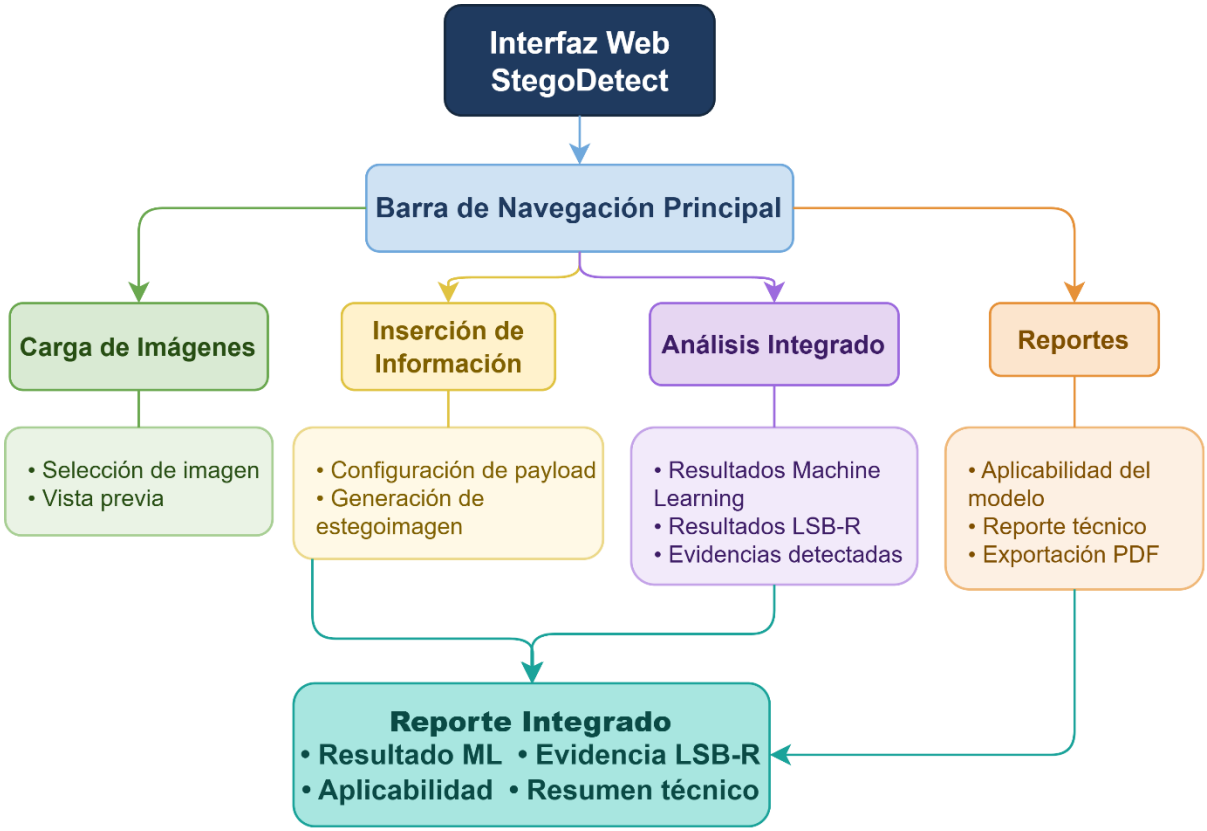


Figura 21. Estructura de la interfaz web.

La trazabilidad del sistema permite relacionar los requerimientos identificados en la dase de análisis con los módulos técnicos que los implementan. En la tabla 11 se presenta la correspondencia entre requerimientos, componentes, funcionalidades y objetivos específicos del proyecto.

Tabla 11. Trazabilidad del sistema.

Requerimiento	Modulo	Funcionalidad	Objetivo especifico
Detectar patrones esteganográficos	Modelo CNN	Puntaje ML de esteganografía	Integrar modelo ML para estegoanálisis
Insertar información oculta	Modulo LSB	Embedding de texto, PDF, imagen o binario	Implementar inserción de información oculta
Extraer información oculta	Auto extracción LSB	Recuperación de payload compatible	Implementar extracción de payloads
Validar integridad	SHA-256	Verificación de payload recuperado	Implementar extracción de payloads.
Interpretar imágenes externas	Domain Assesor	Aplicabilidad del modelo	Evitar conclusiones absolutas fuera de dominio
Presentar resultado integrado	Motor de decisión	Decisión general del sistema	Combinar ML, LSB, SHA-256 y domino
Documentar resultados	Reporte integrado PDF	Generación de informe técnico	Generar evidencia documental del análisis
Facilitar uso del sistema	Frontend SPA	Interacción usuario-sistema	Implementar aplicación web funcional

Se presetna la pantalla principal del sistema StegoDetect. Esta captura permite evidenciar la disposicion general de la interfaz, el acceso a los modulos principales y la orienteación visual de la aplicación hacia un análisis técnico e interpretable.

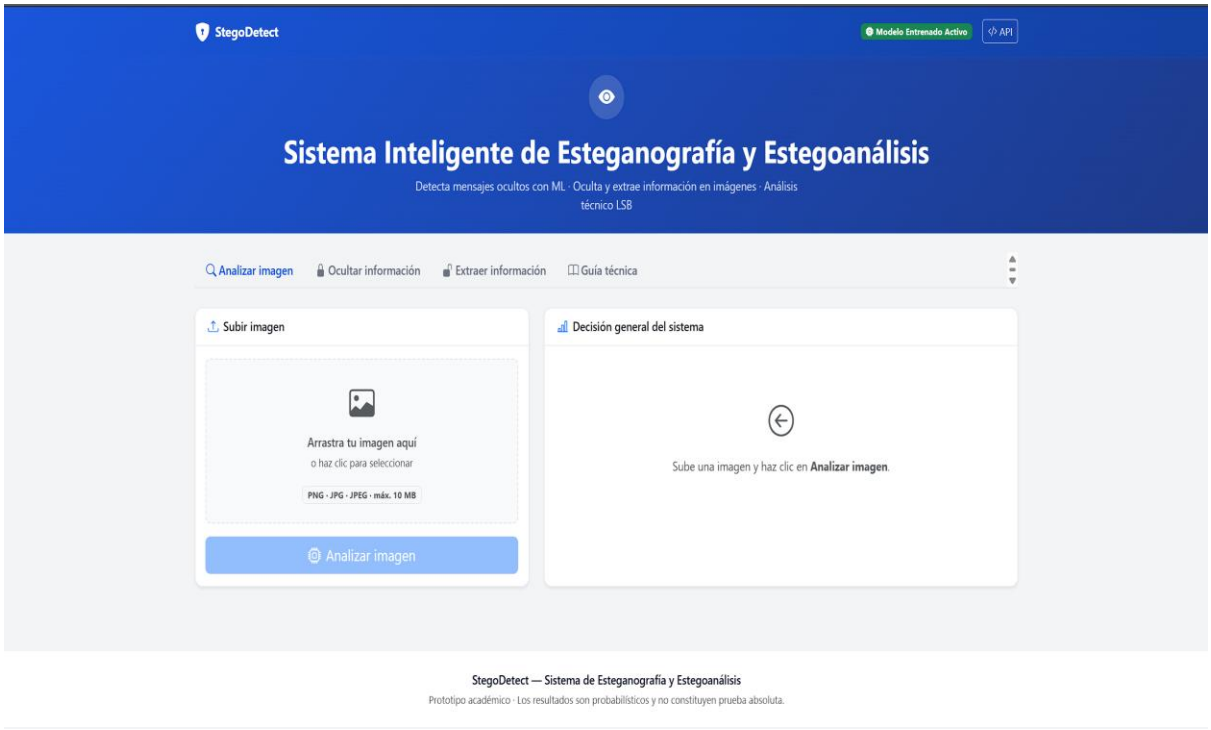


Figura 22. Pantalla principal del sistema StegoDetect.

En la figura 23 se muestra la interfaz destinada a la inserción de información oculta en imágenes. Esta sección permite seleccionar el tipo de payload, configurar la inserción y generar una imagen estaganografiada compatible con StegoDetect.

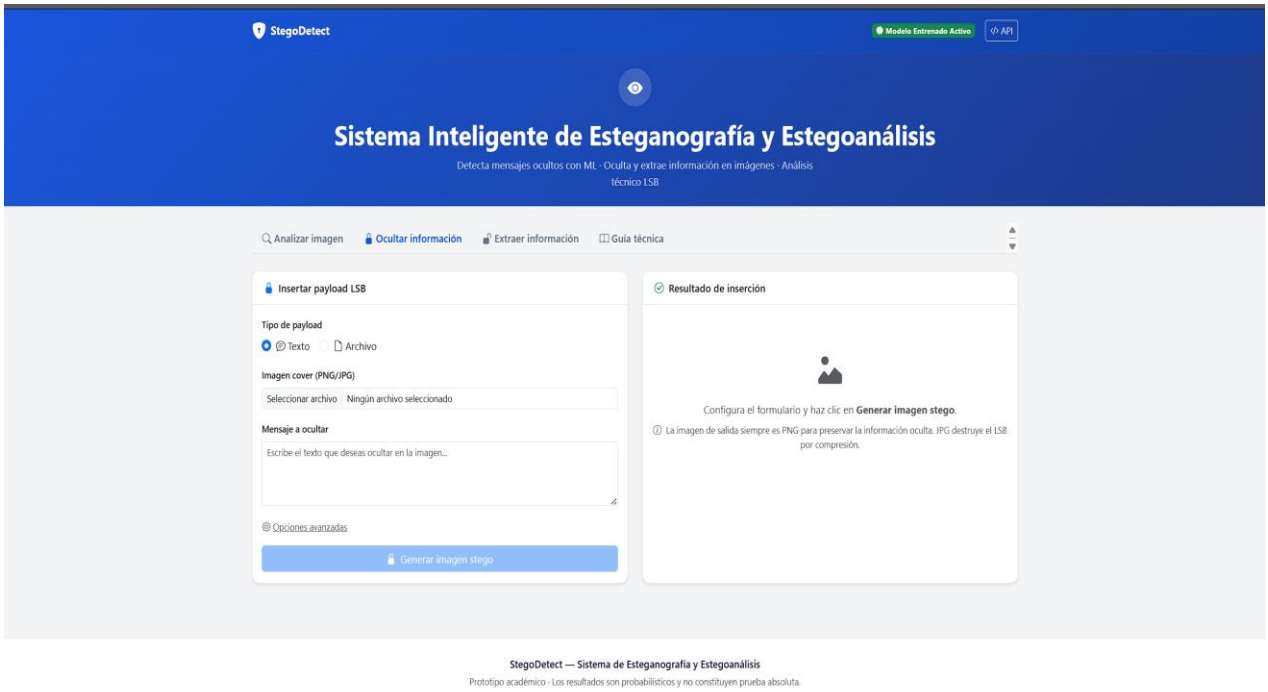


Figura 23. Interfaz para la inserción de información oculta en imágenes.

La figura 24 presenta la interfaz de extracción de payload oculto. Esta sección permite analizar imágenes esteganográficas generadas por el sistema, recuperar el contenido oculto y validar su integridad mediante SHA-256.

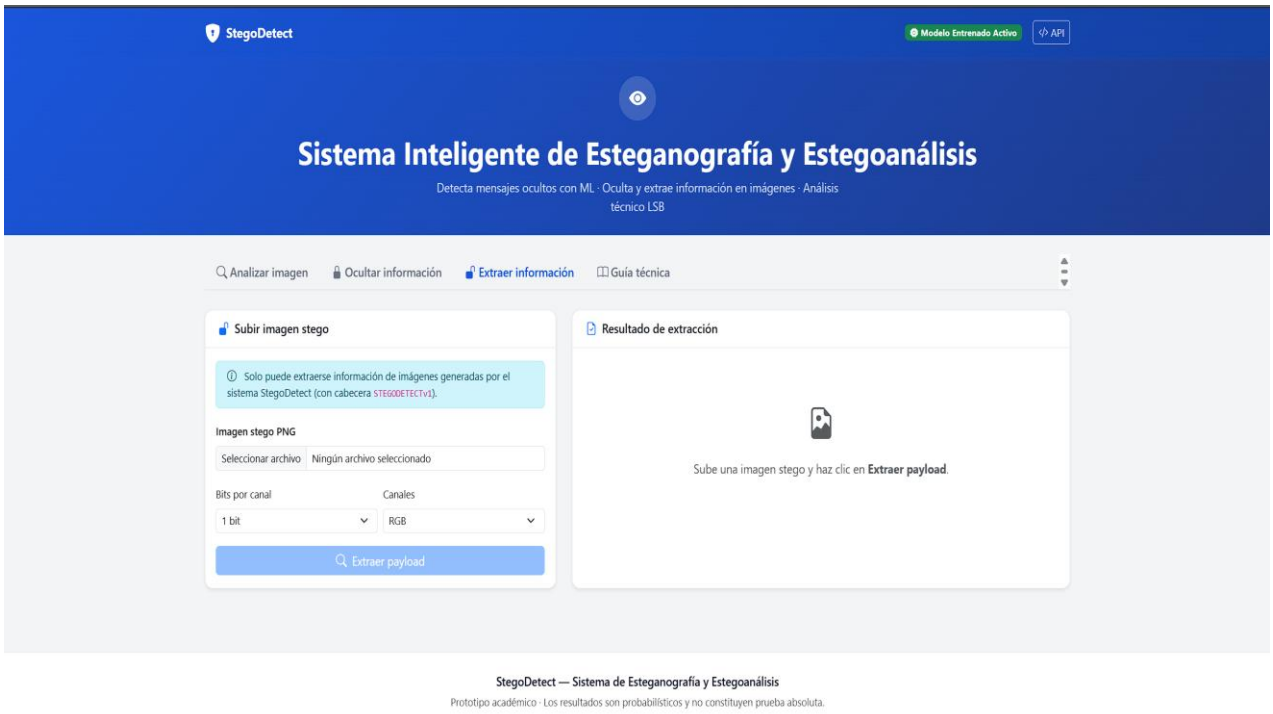


Figura 24. Interfaz de extracción de payload oculto.

CAPÍTULO IV: DESARROLLO E IMPLEMENTACIÓN

4.1 Configuración del Entorno

El proceso de desarrollo e implementación del sistema "StegoDetect" requirió la configuración de un entorno de trabajo estructurado, modular y orientado a la experimentación reproducible en tareas de esteganografía, estegoanálisis y Machine Learinng. La infraestructura tecnológica fue diseñada con el objetivo de garantizar compatibilidad entre los componentes de Machine Learning, backend y frontend, permitiendo mantener independencia entre módulos y facilitando futuras tareas de mantenimiento, escalabilidad y despliegue. La solución fue implementada utilizando Python debido a su amplio ecosistema de librerías científicas, frameworks de aprendizaje profundo y herramientas para desarrollo web, características que lo convierten en uno de los entornos más utilizados en aplicaciones de ciberseguridad e inteligencia artificial (Boroumand et al., 2019; Shehab & Alhaddad, 2025).

La estructura de este proyecto consiste en tres componentes principales: backend, frontend y módulo de Machine Learning. Cada uno de estos componentes mantiene su propio sistema de arquitectura basado en los principios de Clean Architecture, en el que cada capa tiene su propia responsabilidad y la interacción se controla de manera efectiva a través de interfaces. El backend fue desarrollado usando FastAPI, que es un marco para desarrollar APIs REST de alto rendimiento, y el frontend fue desarrollado como una Aplicación de Página Única (SPA) ligera usando HTML5, CSS3, Bootstrap 5 y JavaScript. Por otra parte, el entrenamiento del modelo CNN fue realizado mediante PyTorch ejecutado en Google Colab aprovechando recursos de aceleración GPU para optimizar el procesamiento de imágenes y el entrenamiento de redes neuronales convolucionales.

La configuración del entorno también considero la evolución real del prototipo durante el desarrollo. Inicialmente, el sistema fue concebido como una aplicación de detección basada en un modelo CNN capaz de clasificar imágenes como cover y stego. Sin embargo, durante la implementación, el sistema evoluciono hacia una plataforma hibrida más completa, incorporando inserción LSB-R, extracción automática de payloads, recuperación de texto, recuperación de imágenes, recuperación de documentos PDF, recuperación de archivos binarios, validación de cabecera STEGODETECTv1, validación SHA-256, análisis de aplicabilidad del modelo, decisión integrada y generación de reportes técnicos en PDF. Por esta razón, el entorno de desarrollo no solo debía soportar inferencia de Machine Learning, sino también de archivos, validaciones criptográficas, manejo de uploads, serialización de resultados y comunicación entre múltiples módulos.

Para complementar la descripción del entorno de desarrollo, la figura 25 presenta la arquitectura general utilizada durante la implementación de StegoDetect. Esta representación permite observar la relación entre el entorno de entrenamiento en Google Colab, la persistencia de modelos y métricas en Google Drive, el backend FastAPI, el frontend SPA, los módulos LSB-R, la inferencia del modelo ResidualSRNetLite y le generación de reportes integrados.

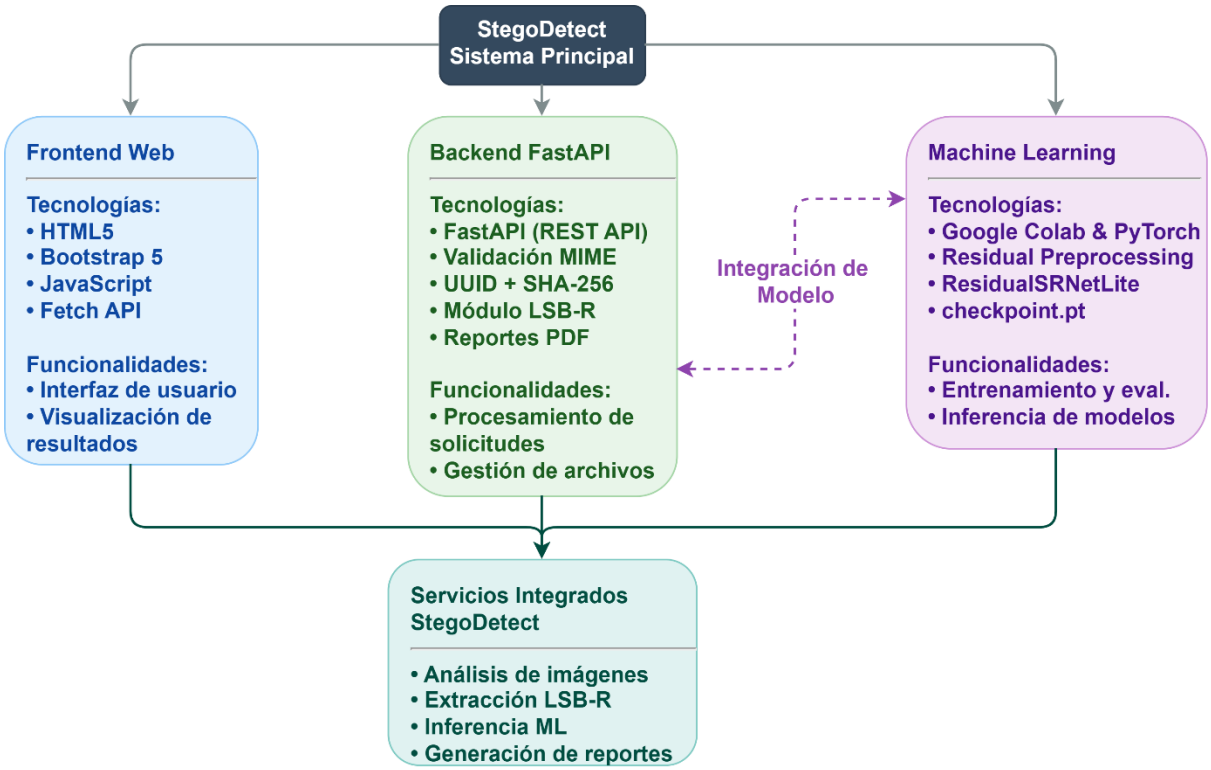


Figura 25. Arquitectura tecnología y entorno de desarrollo de StegoDetect.

La utilización de Google Colab como entorno de entrenamiento respondió principalmente a la necesidad de acceder a recursos computacionales de alto rendimiento, especialmente para el entrenamiento del modelo ResidualSRNetLite. Los procesos de convolución profunda, extracción de características residuales, generación de patches y entrenamiento sobre grandes volúmenes de imágenes requieren una capacidad computacional considerable. Por esta razón, el uso de un entorno con soporte para GPU permitió acelerar las etapas de entrenamiento, validación y ajuste del modelo, manteniendo la posibilidad de documentar el proceso experimental dentro de notebooks reproducibles.

Adicionalmente, Google Colab permitió integrar fácilmente bibliotecas científicas como Numpy, OpenCV, Pillow, TorchVision, PyTorch y Matplotlib, manteniendo compatibilidad directa con el ecosistema de Machine Learning utilizado en el proyecto. Esta integración facilito la implementación de pipelines de preprocesamiento residual, extracción

de patches, generación automatizada de datasets esteganográficos y almacenamiento de métricas. Asimismo, Google Drive fue utilizado como medio de persistencia para datasets, checkpoints, archivos de métrica y metadatos, permitiendo conservar resultados experimentales y reutilizar el modelo entrenado durante la fase de integración con la aplicación web.

```python
# ── Rutas del proyecto ──────────────────────────────────────────────
DRIVE_BASE     = Path("/content/drive/MyDrive/stegadetect_replit")
TRAIN_MANIFEST = DRIVE_BASE / "processed" / "train_manifest.csv"
VAL_MANIFEST   = DRIVE_BASE / "processed" / "val_manifest.csv"
TEST_MANIFEST  = DRIVE_BASE / "processed" / "test_manifest.csv"
CHECKPOINT_DIR = DRIVE_BASE / "checkpoints"
REPORTS_DIR    = DRIVE_BASE / "reports"

CHECKPOINT_BEST      = CHECKPOINT_DIR / "srnet_lite_best.pt"
STATE_DICT_BEST      = CHECKPOINT_DIR / "srnet_lite_best_state_dict.pt"
MODEL_METADATA       = CHECKPOINT_DIR / "model_metadata.json"
TRAINING_HISTORY     = REPORTS_DIR    / "training_history.json"
METRICS_JSON         = REPORTS_DIR    / "metrics.json"
CONFUSION_MATRIX_PNG = REPORTS_DIR    / "confusion_matrix.png"
ROC_CURVE_PNG        = REPORTS_DIR    / "roc_curve.png"
PROB_DIST_PNG        = REPORTS_DIR    / "prob_distribution.png"

CHECKPOINT_DIR.mkdir(parents=True, exist_ok=True)
REPORTS_DIR.mkdir(parents=True, exist_ok=True)

# ── Verificar manifests ─────────────────────────────────────────────
for path, name in [(TRAIN_MANIFEST, "train"), (VAL_MANIFEST, "val"), (TEST_MANIFEST, "test")]:
    if not path.exists():
        raise FileNotFoundError(
            f"Manifest de {name} no encontrado: {path}\n"
            "Ejecuta primero 01_dataset_pipeline_colab.py"
        )
print("Manifests verificados OK.")

# ── Hiperparámetros ─────────────────────────────────────────────────
CONFIG = {
    "lr":              1e-4,       # Learning rate inicial (AdamW)
    "weight_decay":    1e-4,       # L2 regularization
    "epochs":          60,         # Máximo de épocas (early stopping puede cortar antes)
    "batch_size":      32,         # Tamaño de batch (reducir a 16 si hay OOM)
    "num_workers":     2,          # Workers del DataLoader (2 es seguro en Colab)
    "patience":        12,         # Early stopping: épocas sin mejora en val_auc
    "loss_type":       "smooth_bce",  # Con label smoothing para evitar sobreconfianza
    "label_smoothing": 0.05,
    "grad_clip":       1.0,        # Clipping máximo de gradientes
    "scheduler":       "cosine",   # CosineAnnealing LR
    "mixed_precision": True,       # AMP con GPU T4
    "threshold_method":"youden",   # Calcular threshold óptimo con índice de Youden
    "seed":            42,
    "crop_size":       128,        # Crop 128x128 desde imágenes 512x512
    "resume":          False,      # True para reanudar entrenamiento desde checkpoint
}
```

Figura 26. Arquitectura en la nube dentro de Google Drive ejecutado mediante Colab.

Con el objetivo de mantener un entorno reproducible y aislado, el sistema utilizo entornos virtuales mediante venv. Esto permitió encapsular dependencias especificas del proyecto evitando conflictos entre versiones de librerías instaladas globalmente en el sistema operativo. Todas las dependencias requeridas fueron definidas dentro del archivo

requirements.txt, permitiendo replicar el entorno de desarrollo en distintos equipos y facilitando la instalación de componentes como FastAPI, PyTorch, Pillow, OpenCV, Pydantic y ReportLab.

La organización de directorios fue diseñada considerando criterios de mantenibilidad y separación lógica de responsabilidades. El módulo backend contiene rutas API, servicios de inferencia, validaciones, módulos de esteganografía LSB-R, generación de reportes PDF y reportes de decisión integrada. El frontend incluye componentes visuales, hojas de estilo y scripts de interacción dinámica. El módulo ML concentra scripts de entrenamiento, generación de datasets, definición de arquitecturas CNN y archivos relacionados con el checkpoint del modelo entrenado.

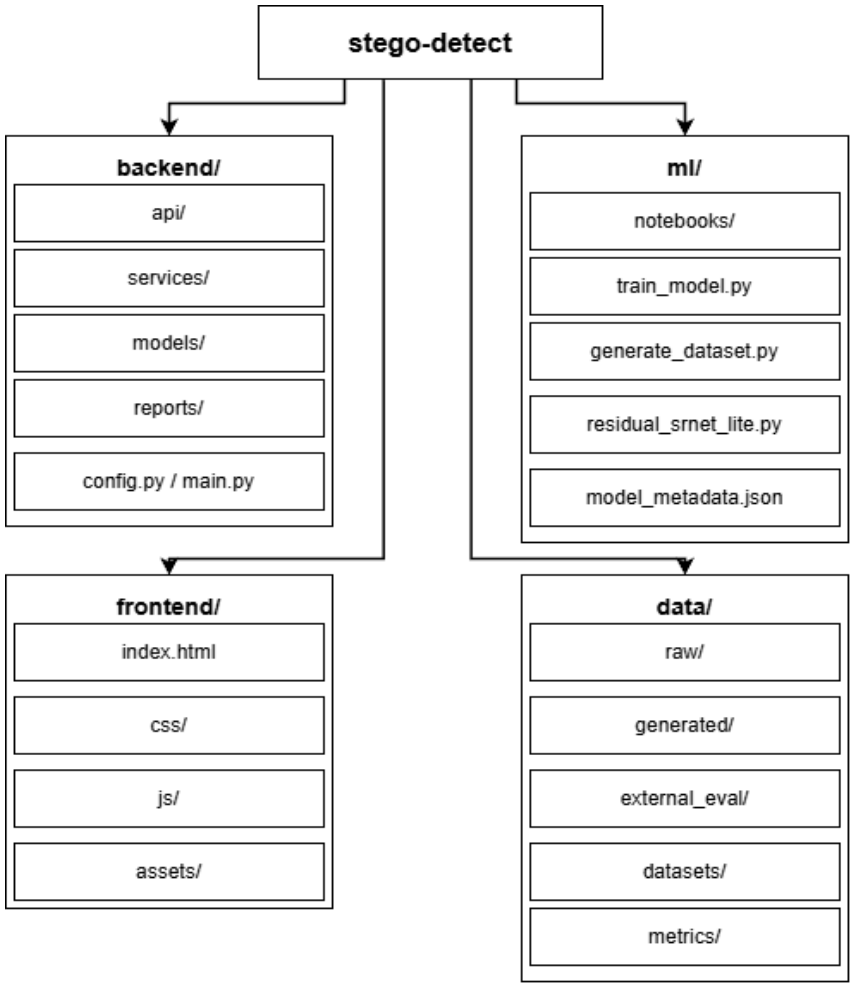


Figura 27. Organización estructural del proyecto "StegoDetect".

4.2 Implementación del Modelo

La implementación del modelo de Machine Learning constituyo uno de los componentes más críticos dentro de la solución StegoDetect, debido a que el proceso de análisis

de mensajes ocultos requiere identificar alteraciones extremadamente sutiles introducidas por técnicas de esteganografía LSB-R sobre imágenes digitales. A diferencia de problemas tradicionales de clasificación visual, el estegoanálisis implica detectar modificaciones imperceptibles a nivel humano, pero estadísticamente relevantes dentro del dominio espacial de los pixeles (Fridrich & Kodovsky, 2012; Boroumand et al., 2019).

El dataset utilizado fue generado a partir de imágenes limpias obtenidas desde BOSSBase, conjunto ampliamente utilizado en investigaciones de estegoanálisis. Las imágenes originales fueron convertidas a formato PNG para evitar perdida de información derivada de compresión con perdida, como JPEG. Posteriormente, se aplicó un algoritmo de ocultamiento LSB-R encargado de insertar mensajes secretos utilizando modificaciones sobre bits menos significativos de los canales RGB.

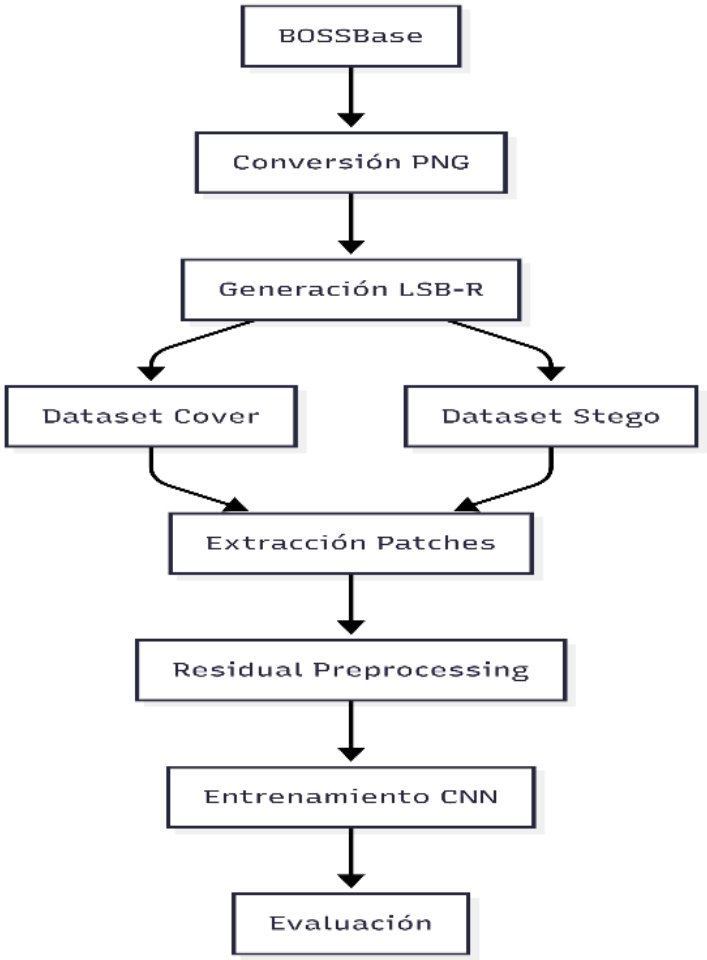


Figura 28. Pipeline general de entrenamiento del modelo CNN.

La generación del dataset fue diseñada de forma balanceada, manteniendo igual número de imágenes cover y stego. Además, los cambios en el tamaño de la carga útil se utilizaron para reducir el sobreajuste al mejorar la capacidad del modelo para manejar la inserción de

información en diferentes niveles. Con esto, el proceso de entrenamiento no dependía de una longitud de mensaje particular ni de una única instancia de modificación en los bits menos significativos.

Durante la implementación, el uso de residual preprocessing se convirtió en una de las decisiones técnicas más importantes, ya que optimiza la calidad de los datos antes de pasarlos al algoritmo. Diversas investigaciones han demostrado que las señales esteganográficas suelen quedar ocultas bajo el contenido visual dominante de la imagen, razón por la cual resulta necesario aplicar filtros high-pass capaces de eliminar información semántica y resaltar patrones residuales relacionados con alteraciones de pixeles (Xu et al., 2016; Shehab & Alhaddad, 2025). La ventaja de este enfoque es que el modelo va más allá de la simple apariencia visual; su verdadero foco es rastrear las alteraciones estadísticas imperceptibles que deja el software de esteganografía en los píxeles.

El modelo utilizado fue una variante ligera llamada ResidualSRNetLite, que fue influenciada por el modelo SRNet para operaciones de esteganálisis profundo. El diseño incorpora capas convolucionales, normalización por lotes, ReLU, bloques residuales y una técnica de reducción de características para capturar patrones residuales de alta frecuencia. Este diseño permite que el modelo aprenda a distinguir los patrones que se encuentran en los múltiples mínimos en la diferencia entre las imágenes limpias y las imágenes esteganografiadas.

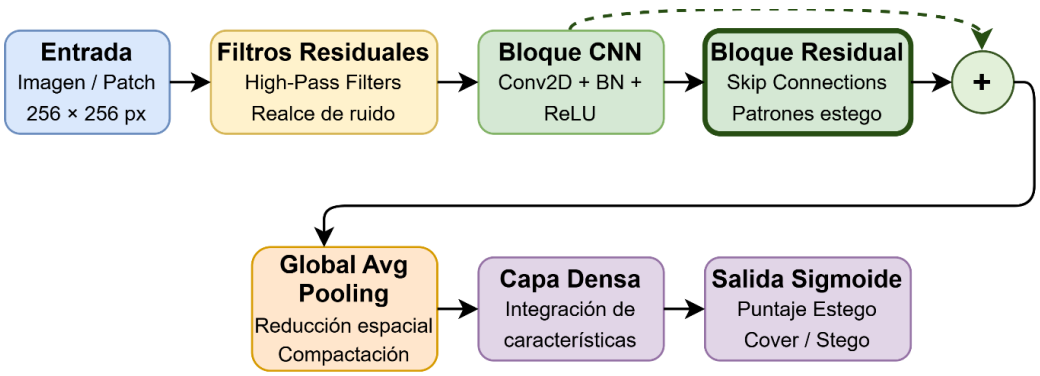


Figura 29. Arquitectura simplificada de ResidualSRNetLite.

Durante el entrenamiento, las métricas de evaluación de precisión, exhaustividad, puntuación F1 y AUC-ROC fueron particularmente útiles dado que la detección de esteganografía no solo debe considerar la precisión general, sino también la capacidad de diferenciar entre clases estrechamente relacionadas. Además, se desarrolló una estrategia para

la búsqueda del umbral de decisión óptima basada en las métricas de clasificación para ajustar el modelo y que sea más sensible a las imágenes esteganográficas.

Cabe mencionar que los resultados del modelo no deben tomarse como una indicación de la presencia absoluta de un mensaje secreto, antes de presentar los parámetros. En la versión de nuestro sistema, el modelo proporciona una puntuación de ML relacionada con la esteganografía, que luego se combina con evidencia LSB-R, SHA-256 y la validez del modelo. Esta decisión fue necesaria debido a que el modelo fue entrenado principalmente sobre un dominio experimental controlado, por lo que imágenes externas como wallpapers, fotografías de celular o archivos JPG con alta compresión pueden generar resultados no concluyentes.

Tabla 12. Configuración del entrenamiento del modelo.

Parámetro	Valor	Justificación técnica
Dataset base	BOSSBase	Conjunto de referencia utilizado en investigaciones de estegoanálisis
Técnica de generación stego	LSB-R	Permite generar imágenes con alteraciones controladas en el dominio espacial
Formato de trabajo	PNG	Evita perdida de información por compresión destructiva
Tamaño de entrada	128x128 o 256x256 según configuración experimental	Permite procesar patches conservando señales locales
Patches por imagen	Configuración definida en entrenamiento	Aumenta la cantidad de muestras y conserva detalles locales
Preprocesamiento	Filtros residuales high-pass	Resalta ruido residual asociado a modificaciones esteganográficas
Modelo	ResidualSRNetLite	Arquitectura CNN residual adaptada al estegoanálisis.
Función de perdida	BCEWithLogitsLoss / Binary Crossentropy	Adecuada para clasificación binaria cover/stego
Optimizador	Adam / AdamW	Permite optimización estable en redes convolucionales
Métricas	Accuracy, precisión, recall, F1-score, AUC-ROC	Evalúan desempeño general, sensibilidad y separabilidad entre clases
Checkpoint	Archivo .pt	Contiene los pesos entrenados utilizados por el backend
Metadatos	model_metadata.json	Registra configuración, umbral, arquitectura y métricas del modelo.

El modelo entrenado fue integrado al backend como servicio de inferencia. Durante el análisis, la imagen es validad, convertida al formato requerido, procesada mediante extracción de patches y evaluada por ResidualSRNetLite. La salida generada corresponde a un puntaje asociado a patrones compatibles con esteganografía, pero la decisión final del sistema no

depende únicamente de este valor. En su lugar, el sistema combina dicho puntaje con evidencia determinística proveniente de la extracción LSB-R y validación SHA-256.

En la figura 30 se muestra la interfaz principal de análisis de imagen en donde se analizó una imagen estaganografiada y se muestra el resultado mediante una barra de porcentaje de predicción del modelo con un mensaje de alerta. De igual forma se muestra un análisis LSB técnico y un mensaje de detección de payload disponible con el sistema StegoDetect.

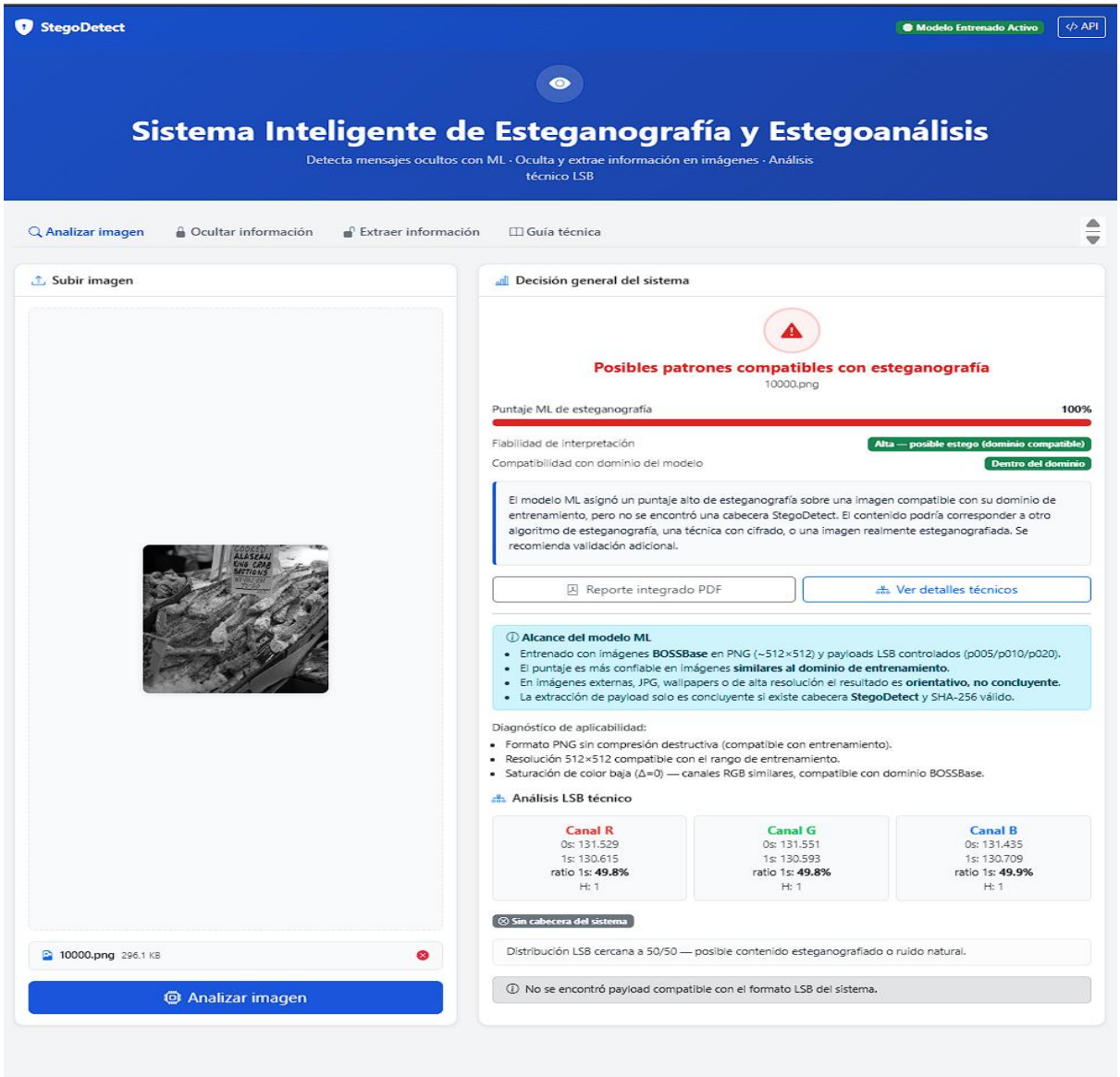


Figura 30. Análisis de una imagen estaganografiada en el sistema StegoDetect.

Pseudocodigo 3. Pipeline de inferencia CNN

Entrada: Imagen PNG/JPEG

Salida: Puntaje ML de esteganografía.

1. Recibir imagen desde el backend.
2. Validar formato y contenido del archivo.
3. Convertir patches de la imagen.
4. Extraer patches de la imagen.
5. Normalizar valores de pixeles.
6. Aplicar preprocesamiento residual mediante filtros high-pass.
7. Ejecutar forward pass en ResidualSRNetLite.
8. Obtener puntajes por patch.
9. Agregar puntajes parciales en un resultado global.
10. Comparar el puntaje del umbral definido en model_metadata.json.
11. Retornar puntaje ML e interpretación interna.

4.3 Desarrollo de la Ampliación Web

La aplicación web "StegoDetect" fue diseñada bajo una arquitectura cliente-servidor desacoplada, donde el frontend funciona como SPA estática y el backend expone funcionalidades mediante una API REST desarrollada en FastAPI. Esta arquitectura permitió mantener independencia tecnológica entre componentes y facilito la integración directa con el modelo CNN de estegoanálisis, los servicios de esteganografía LSB-R, la extracción de payloads y la generación de reportes técnicos.

El backend implementa Endpoints REST especializados para carga de imágenes, inserción de información oculta, extracción de payloads, ejecución del análisis integrado y generación de reportes PDF. Cada endpoint fue desarrollado utilizando validaciones orientadas a seguridad informática y control de integridad de archivos. Entre estas validaciones se incluye verificación de extensión, análisis MIME, generación de UUID, almacenamiento temporal seguro y cálculo de hash SHA-256.

En la siguiente figura se representa el recorrido completo de una imagen desde la interacción del usuario hasta la generación de resultados y reportes, mostrando la integración entre la aplicación web, los servicio s backend, el análisis basado en Machine Learning y la extracción LSB-R.

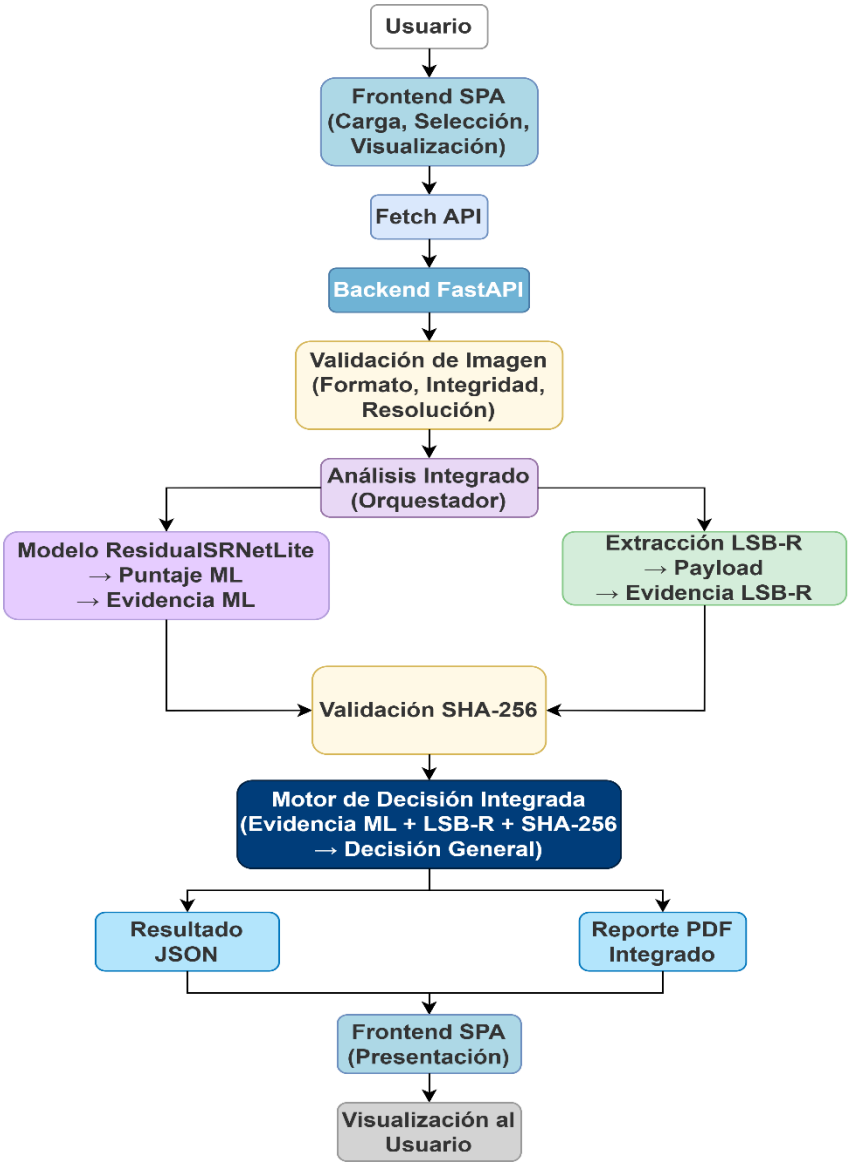


Figura 31. Flujo general de desarrollo de la aplicación web.

La aplicación fue ampliada para incorporar funcionalidad de esteganografía y estegoanálisis. El sistema permite insertar información oculta en imágenes mediante LSB-R, utilizando payloads de tipo de texto, PDF, imagen o archivo binario. Para garantizar la identificación posterior del contenido oculto, el sistema incorpora una cabecera compatible con StegoDetect y un hash SHA-256 del payload insertado.

El módulo de extracción permite recuperar payloads compatibles siempre que la imagen haya sido generada bajo el formato del sistema y conserve una cabecera valida. Esta extracción puede realizarse de forma manual, indicando parámetros como los bits por canal y canales RGB, o de forma automática dentro del análisis integrado. La auto extracción LSB-R prueba configuraciones de 1, 2, 3 y 4 bits por canal en RGB, deteniéndose cuando encuentra una

cabecera valida y una verificación SHA-256 correcta. Este mecanismo permite analizar imágenes generadas por StegoDetect, aunque el usuario no recuerde la configuración utilizada en la inserción.

El frontend fue desarrollado utilizando Bootstrap 5 con el objetivo de proporcionar una interfaz responsive, moderna y compatible con múltiples dispositivos. La comunicación entre frontend y backend se realiza mediante Fetch API enviando solicitudes asíncronas y recibiendo respuestas JSON o archivos descargables. Esta estrategia evita recargas completas de la página y mejora la experiencia del usuario durante procesos como análisis, inserción, extracción y generación de reportes.

La siguiente figura muestra la pantalla principal de la aplicación, como el frontend fue diseñado como un SPA como header se muestra el titulo del proyecto, una breve descripción de lo que es y como primera pestaña está el analizador de imágenes.

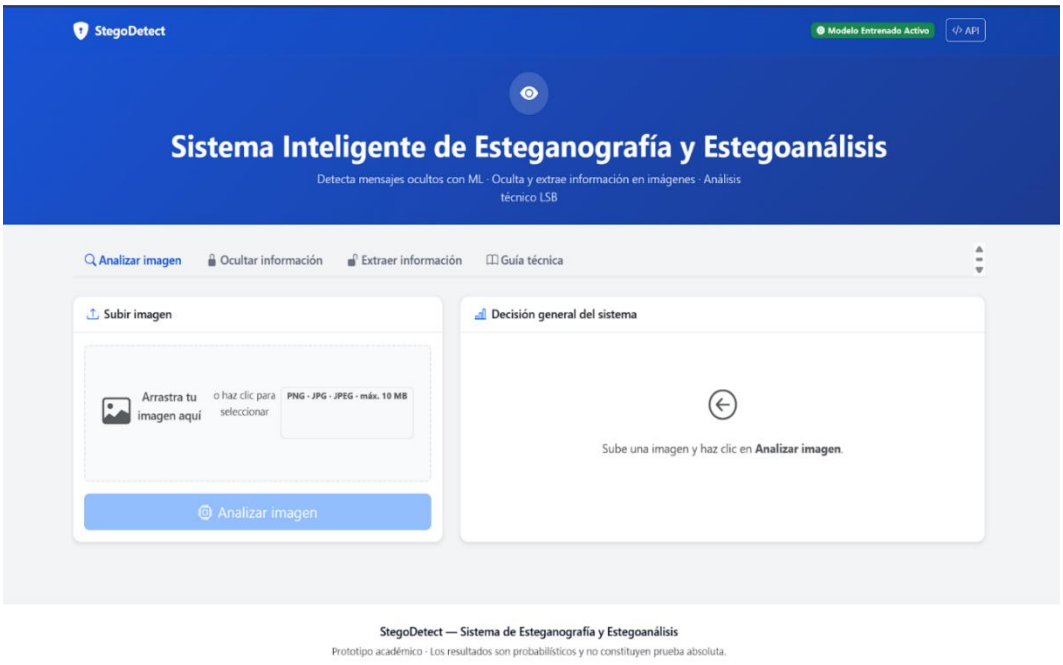


Figura 32. Interfaz principal del sistema StegoDetect.

En la figura 33 se evidencia la carga de imagen, selección de payload, bits por canal, canales RGB y generación de imagen estego.

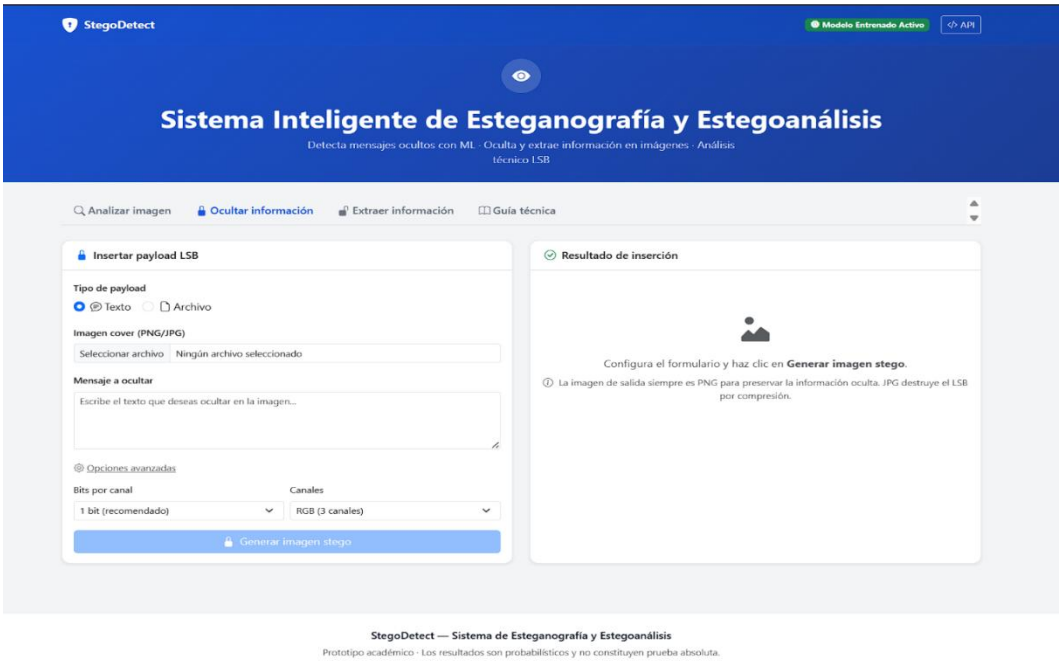


Figura 33. Interfaz de inserción de información oculta.

La interfaz fue diseñada priorizando simplicidad operativa y experiencia de usuario. El flujo principal permite cargar una imagen, insertar información oculta cuando se requiere, ejecutar análisis integrado, visualizar la decisión general del sistema y descargar un reporte PDF técnico. Además, la aplicación permite recuperar payloads compatibles y mostrar información diferenciada sobre el resultado del modelo ML, la evidencia LSB-R, la validación SHA-256 y la aplicabilidad del modelo.

La siguiente imagen muestra la interfaz de análisis integrado en donde esta la decisión general del sistema, puntaje ML, evidencia LSB-R, validación SHA-256, aplicabilidad del modelo y reporte integrado.

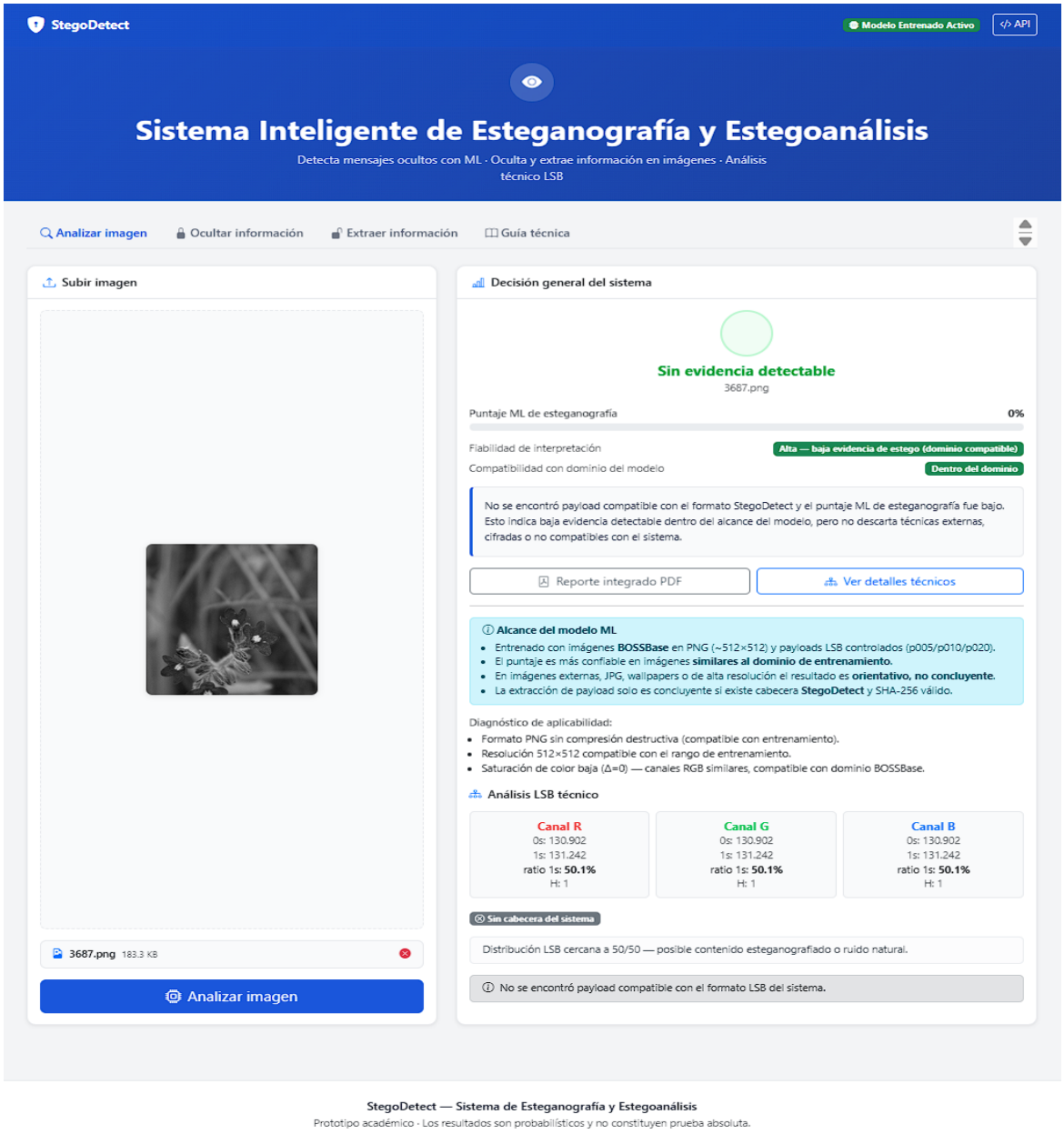


Figura 34. Interfaz de análisis integrado.

El sistema implemento validaciones de seguridad orientadas a prevenir carga de archivos maliciosos, ejecución arbitraria y corrupción de almacenamiento interno. Para ello se utilizó UUID aleatorios, verificación MIME, control de extensiones, cálculo de SHA-256 y manejo controlado de excepciones. Estas medidas son relevantes debido a que el sistema procesa archivos enviados por el usuario, lo que exige controles preventivos para reducir riesgos de seguridad.

El reporte principal y la decisión del sistema se basan en el endpoint de análisis y en el reporte PDF integrado.

En la tabla 13 se presentan los endpoints implementados en el sistema StegoDetect, cuales son las funciones que estos realizan y el resultado esperado de cada uno de ellos.

Tabla 13. Endpoints REST implementados.

Método	Endpoint	Función	Resultado
POST	/analyze	Ejecuta análisis mediante el modelo ML	Puntaje ML de esteganografía y predicción interna
POST	/stego/embed	Inserta información oculta en una imagen mediante LSB-R	Imagen stego generada por el sistema
POST	/stego/extract	Extrae manualmente payloads compatibles	Texto, PDF, imagen o binario recuperado
POST	/stego/full-analysis	Ejecuta análisis integrado mediante ML, LSB-R, SHA-256 y aplicabilidad	Decisión general del sistema
GET	/stego/report/{id}	Genera reporte PDF integrado	Reporte con decisión general, evidencia LSB, ML y conclusión técnica
GET / POST	/report o /report/{id}	Endpoint heredado de reporte basado en análisis ML	Reporte legacy, no usado como reporte integrado principal

La decisión general del sistema se obtiene a partir de una lógica de priorización. Cuando la auto extracción LSB-R encuentra un payload valido y el hash SHA-256 coincide, el sistema presenta la decisión “Mensaje oculto encontrado”, independientemente del puntaje ML. Si no existe payload compatible, pero el modelo detecta patrones dentro del dominio de entrenamiento, el sistema reporta “Posibles patrones compatibles con esteganografía”. En cambio, si la imagen esta fuera del dominio del modelo, el resultado ML se interpreta como no concluyente. Finalmente, cuando no existe payload y el puntaje ML es bajo, el sistema presenta “Sin evidencia detectable”.

Pseudocodigo 4. Flujo de inserción de información LSB-R.

Entrada: Imagen cover, payload, bits por canal y canales seleccionados.

Salida: Imagen stego generada.

1. Recibir imagen cover desde el frontend.
2. Recibir payload de tipo de texto, PDF, imagen o binario.
3. Validar formato y tamaño de la imagen.
4. Validar tamaño del payload frente a la capacidad disponible.
5. Generar cabecera compatible con StegoDetect.
6. Calcular hash SHA-256 del payload.
7. Construir secuencia binaria con cabecera, metadatos, payload y hash.

8. Insertar bits en los canales seleccionados mediante LSB-R.
9. Guardar imagen stego resultante.
10. Retornar archivo generado al usuario.

Este proceso permite que el sistema no solo analice imágenes, sino que también genera imágenes esteganográficas controladas, útiles para validación y pruebas del prototipo.

Pseudocodigo 5. Flujo de análisis.

Entrada: Imagen a analizar.

Salida: Decisión general del sistema.

1. Recibir imagen desde la interfaz web.
2. Validar formato, extensión y contenido real del archivo.
3. Calcular hash SHA-256 de la imagen.
4. Ejecutar inferencia del modelo ResidualSRNetLite.
5. Obtener puntaje ML de esteganografía.
6. Ejecutar auto extracción LSB-R.
7. Verificar existencia de cabecera StegoDetect.
8. Validar SHA-256 del payload recuperado, si existe.
9. Evaluar aplicabilidad del modelo según características de la imagen.
10. Integrar evidencia ML, evidencia LSB y dominio del modelo.
11. Generar decisión general.
12. Retornar resultado JSON al frontend.

Este flujo evita que el sistema dependa únicamente del modelo ML, ya que combina evidencia estadística con evidencia determinística.

Pseudocodigo 6. Auto extracción LSB-R.

Entrada: Imagen a analizar.

Salida: Payload recuperado o resultado sin payload compatible

1. Definir configuraciones de prueba: 1, 2, 3 y 4 bits por canal en RGB.
2. Para cada configuración, intentar extraer secuencia binaria.
3. Busca cabecera compatible con StegoDetect.
4. Si no existe cabecera, continuar con la siguiente configuración.

5. Si existe cabecera, leer metadatos del payload.
6. Recuperar contenido oculto.
7. Calcular SHA-256 del payload recuperado.
8. Comparar hash calculado con hash almacenado.
9. Si el hash coincide, retornar payload valido.
10. Si ninguna configuración es válida, retornar sin payload compatible.

La auto extracción permite que el análisis integrado detecte payloads generados con distintas configuraciones de inserción sin requerir intervención manual del usuario.

Pseudocodigo 7. Generación de reporte PDF integrado.

Entrada: Resultado de /stego/full-analysis

Salida: Reporte PDF integrado.

1. Recibir identificador del análisis integrado.
2. Recuperar datos almacenados del análisis.
3. Leer decisión general del sistema.
4. Agregar sección de evidencia LSB-R.
5. Agregar sección de resultado del modelo ML.
6. Agregar sección de aplicabilidad del modelo.
7. Incluir validación SHA-256 y metadatos del payload.
8. Generar conclusión técnica priorizando evidencia directa.
9. Construir documento PDF.
10. Retornar archivo descargable al usuario.

Este reporte evita contradicciones entre pantalla y documento descargado, debido a que utiliza la misma decisión general generada por el análisis integrado.

4.4 Pruebas y Validación

Las pruebas realizadas sobre el sistema “StegoDetect” estuvieron orientadas a validar tanto el funcionamiento funcional de la aplicación web como el desempeño del modelo CNN y la correcta integración del enfoque hibrido de estegoanálisis. Estas pruebas incluyeron validaciones funcionales, pruebas de integración, pruebas API, pruebas de seguridad, análisis experimental del modelo y pruebas específicas de inserción, extracción y validación SHA-256.

Las pruebas funcionales permitieron verificar el correcto comportamiento de los endpoints REST bajo distintos escenarios de entrada. Se evaluó la carga de imágenes PNG/JPG/JPEG validas, el manejo de archivos corruptos, el rechazo de formatos inválidos y generación adecuada de respuestas JSON. Asimismo, se verifico que los archivos fueran almacenados con identificadores únicos, evitando conflictos de nombres y reduciendo riesgos asociados al manejo de archivos subidos por usuarios.

También se realizaron pruebas sobre el módulo de inserción LSB-R. En estas pruebas, el sistema recibió imágenes cover y payloads de diferente tipo, incluyendo texto y PDF. Posteriormente, genero imágenes stego descargables que conservan la apariencia visual de la imagen original, pero contenían información oculta dentro de los bits menos significativos. Estas pruebas fueron vitales porque confirmaron que StegoDetect funciona no solo como detector de esteganografía, sino también como una herramienta para la esteganografía controlada.

Las pruebas de extracción manual de cargas útiles confirmaron que las cargas útiles se recuperaron para archivos de texto, PDF y compatibles con el sistema cuando el usuario definió correctamente las configuraciones. Además, la validación SHA-256 permitió comprobar que el contenido recuperado coincidía con el payload original, otorgando evidencia técnica de integridad.

Posteriormente, se validó el mecanismo de auto extracción LSB-R dentro del análisis integrado. Este componente fue probado con imágenes generadas mediante distintas configuraciones, incluyendo inserciones con 1 y 2 bits por canal en RGB. El sistema logro identificar automáticamente la configuración valida, detectar la cabecera StegoDetect, recuperar el payload y validad el hash SHA-256. Esta prueba fue relevante debido a que evito falsos negativos cuando el análisis integrado intentaba extraer únicamente con una configuración por defecto.

La siguiente imagen muestra que el sistema identifica un mensaje oculto, valida SHA-256 y permite descargar el archivo recuperado, cabe recalcar que todo esto se debe a que la imagen estaganografiada que fue analizada esta dentro del mismo dominio del sistema StegoDetect.

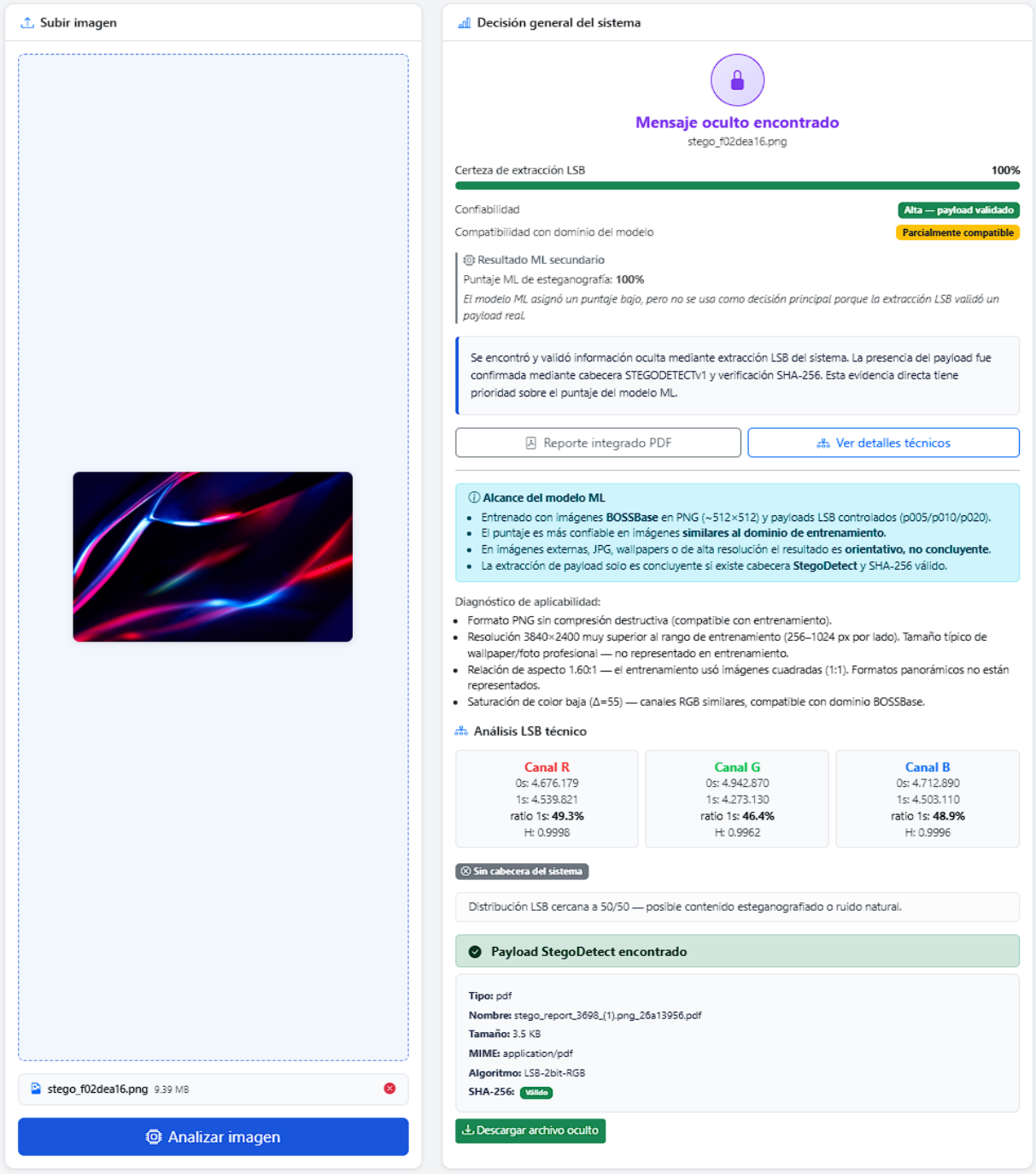


Figura 35. Resultado de análisis de imagen con payload PDF.

Una prueba importante correspondió a la comparación entre la decisión general del sistema y el resultado del modelo ML. En imágenes generadas por StegoDetect con payload válido, el modelo ML podía asignar un puntaje bajo o alto según las características estadísticas de la imagen; sin embargo, cuando la extracción LSB-R recuperaba un payload con SHA-256 valido, la decisión general debía ser "Mensaje oculto encontrado". Esto confirma que la evidencia determinística tiene prioridad sobre el puntaje ML en los casos donde existe recuperación verificable de información.

Asimismo, se realizaron pruebas con imágenes BOSSBase cover y stego. En imágenes cover, el sistema debía presentar “Sin evidencia detectable” cuando no existía payload compatible y el puntaje ML era bajo. En imágenes stego pertenecientes a dominio de entrenamiento, el sistema podía reportar “Posibles patrones compatibles con esteganografía”, aclarando que la evidencia proviene del modelo ML y no de la recuperación de un payload StegoDetect.

También se probaron imágenes externas, como wallpapers, imágenes JPG comprimidas o archivos provenientes de fuentes distintas al dataset de entrenamiento. Estas pruebas permitieron identificar la importancia del módulo de aplicabilidad del modelo. Cuando una imagen externa no contenía cabecera StegoDetect y el modelo asignaba un puntaje alto, el sistema no debería reportar una detección concluyente, sino un resultado ML no concluyente en imagen externa. Esta distinción evita interpretar el puntaje del modelo como una verdad absoluta fuera del dominio experimental.

La figura muestra la decisión general del sistema, evidencia LSB-R, resultado ML, aplicabilidad del modelo y conclusión técnica.

REPORTE INTEGRADO DE ESTEGANOGRAFÍA Y ESTEGOANÁLISIS

Decisión combinada: extracción LSB StegoDetect + modelo SRNet-lite + aplicabilidad de dominio

Información del Análisis

ID del Análisis:	779cc155-2347-47d5-aa3e-942c649186ea
Archivo analizado:	stego_3684527e.png
Fecha y hora (UTC):	2026-05-20 00:00:47
Versión del modelo:	srnet-lite-v1.0
Modo de operación:	Modelo real

1. Decisión general del sistema

■ Mensaje oculto encontrado

Fuente principal:	Extracción LSB StegoDetect (evidencia directa)
Confiabilidad:	Alta

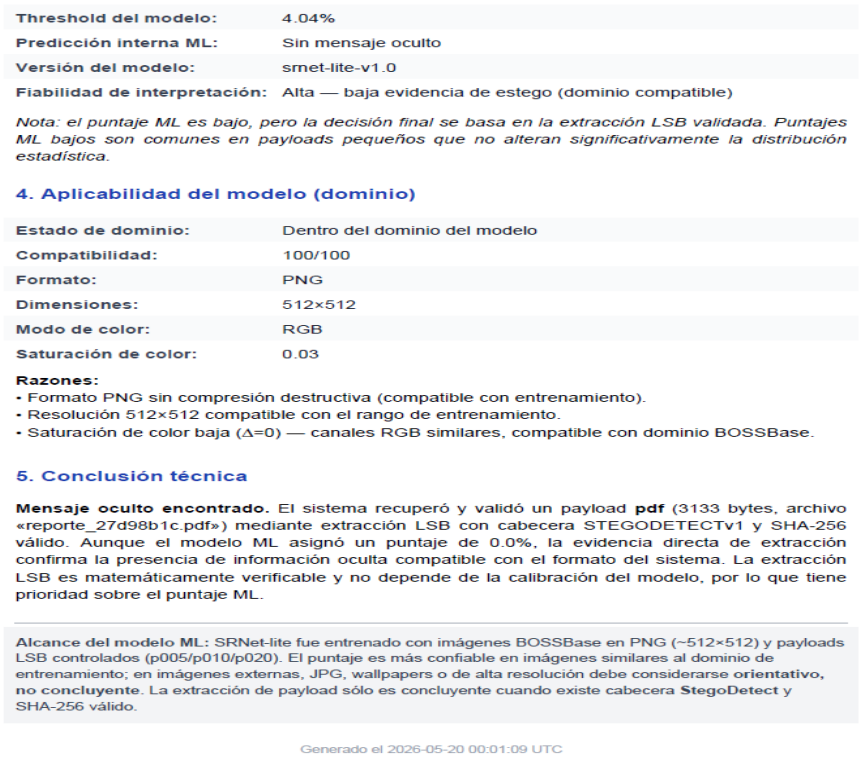
Se encontró y validó un payload mediante extracción LSB del sistema (cabecera STEGODETECTv1 detectada, SHA-256 verificado). Evidencia directa — no depende del modelo ML.

2. Resultado de extracción LSB

Cabecera StegoDetect:	Detectada
Payload encontrado:	Sí
SHA-256 válido:	Sí
Tipo de payload:	pdf
Nombre archivo:	reporte_27d98b1c.pdf
Tamaño (bytes):	3133
MIME:	application/pdf
Algoritmo:	LSB-1bit-RGB

3. Resultado del modelo ML

Puntaje ML de esteganografía	0.0%



Threshold del modelo:	4.04%
Predicción interna ML:	Sin mensaje oculto
Versión del modelo:	srnet-lite-v1.0
Fiabilidad de interpretación:	Alta — baja evidencia de estego (dominio compatible)

Nota: el puntaje ML es bajo, pero la decisión final se basa en la extracción LSB validada. Puntajes ML bajos son comunes en payloads pequeños que no alteran significativamente la distribución estadística.

4. Aplicabilidad del modelo (dominio)

Estado de dominio:	Dentro del dominio del modelo
Compatibilidad:	100/100
Formato:	PNG
Dimensiones:	512×512
Modo de color:	RGB
Saturación de color:	0.03

Razones:
- Formato PNG sin compresión destructiva (compatible con entrenamiento).
- Resolución 512×512 compatible con el rango de entrenamiento.
- Saturación de color baja (Δ=0) — canales RGB similares, compatible con dominio BOSSBase.

5. Conclusión técnica

Mensaje oculto encontrado. El sistema recuperó y validó un payload **pdf** (3133 bytes, archivo «reporte_27d98b1c.pdf») mediante extracción LSB con cabecera STEGODETECTv1 y SHA-256 válido. Aunque el modelo ML asignó un puntaje de 0.0%, la evidencia directa de extracción confirma la presencia de información oculta compatible con el formato del sistema. La extracción LSB es matemáticamente verificable y no depende de la calibración del modelo, por lo que tiene prioridad sobre el puntaje ML.

Alcance del modelo ML: SRNet-lite fue entrenado con imágenes BOSSBase en PNG (~512×512) y payloads LSB controlados (p005/p010/p020). El puntaje es más confiable en imágenes similares al dominio de entrenamiento; en imágenes externas, JPG, wallpapers o de alta resolución debe considerarse **orientativo, no concluyente**. La extracción de payload sólo es concluyente cuando existe cabecera **StegoDetect** y SHA-256 válido.

Generado el 2026-05-20 00:01:09 UTC

Figura 36. Reporte PDF integrado generado por StegoDetect.

El reporte PDF integrado también fue sometido a validación. Inicialmente, el reporte basado únicamente en el análisis ML podía contradecir la interfaz cuando el sistema detectaba un payload valido mediante LSB-R, pero el modelo ML mostraba puntaje bajo. Para corregir esta inconsistencia, el flujo fue actualizado para que el reporte integrado use la misma decisión general generada por /stego/full-analisys y almacenada para su descarga mediante /stego/report/{id}. De esta forma, el reporte separa decisión general, evidencia LSB, puntaje ML, aplicabilidad del modelo y conclusión técnica.

La siguiente imagen evidencia que el sistema interpreta el puntaje ML como no concluyente cuando la imagen puesta a análisis no pertenece al dominio de entrenamiento.

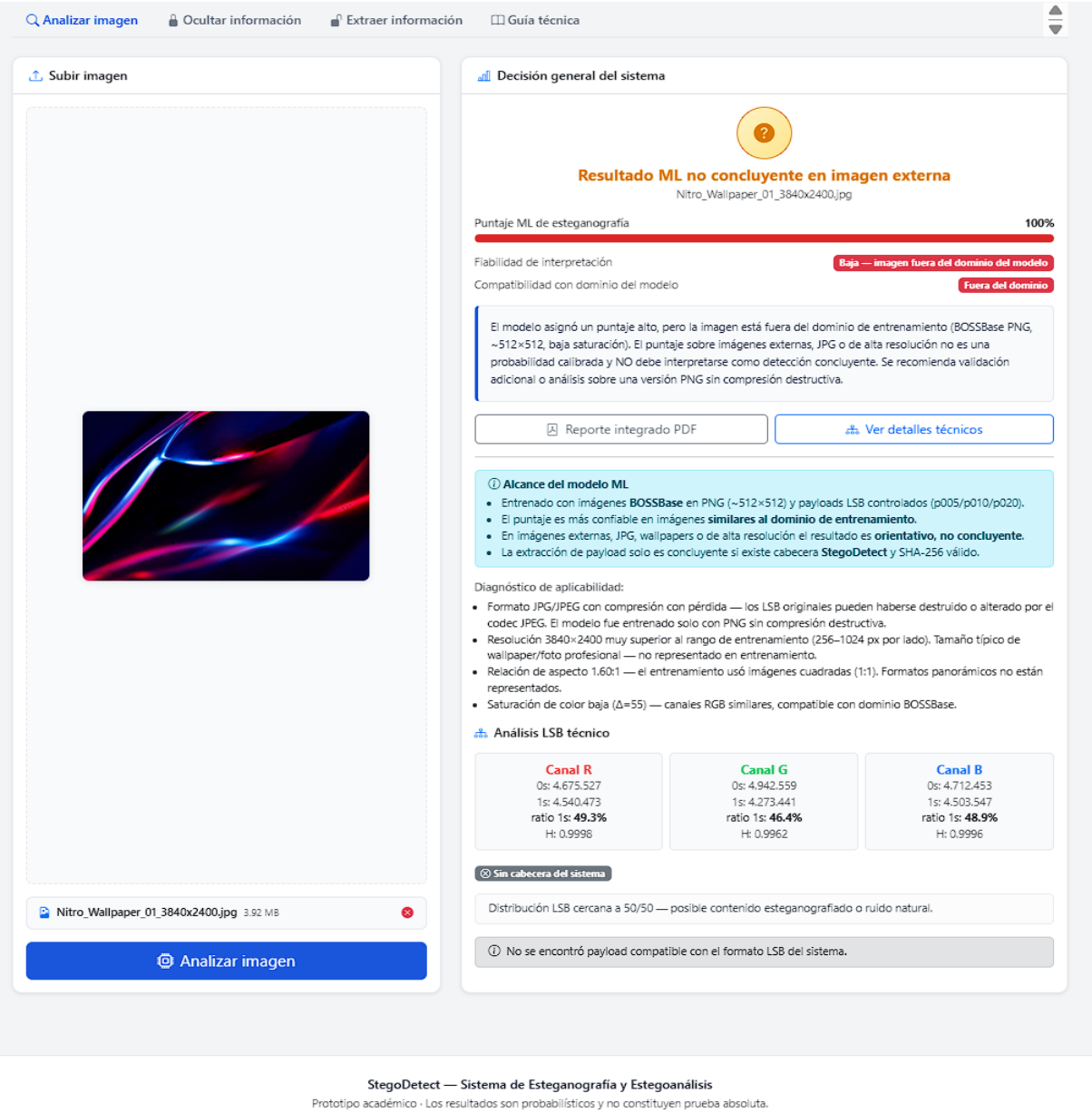


Figura 37. Resultado de imagen externa fuera del dominio del modelo.

Antes de presentar los resultados de prueba, se debe señalar que la validación del sistema no se limitó a comprobar si una imagen era clasificada como cover o stego. Debido a la evolución del prototipo, las pruebas se enfocaron en comprobar si el sistema podía insertar, detectar, extraer, validar y reportar evidencia de forma coherente.

Con el objetivo de verificar el correcto funcionamiento de cada uno de los módulos implementados en StegoDetect, se diseñó un conjunto de pruebas funcionales orientadas a validar los principales procesos del sistema. Estas pruebas funcionales orientadas a validar los principales procesos del sistema. Estas pruebas abarcan desde la carga y validación de imágenes hasta las funcionalidades de inserción, extracción y recuperación de información

oculta, así como la generación de reportes y la validación de integridad mediante SHA-256. Los resultados obtenidos permitieron comprobar que cada componente opera de acuerdo con los requerimientos establecidos durante la fase de diseño e implementación.

Tabla 14. Casos de prueba funcionales.

Caso	Entrada	Resultado esperado	Resultado obtenido	Estado
Carga de imagen valida	Imagen PNG/JPG/JPEG	Archivo aceptado y procesado	Imagen validada correctamente	Aprobado
Archivo invalido	Archivo no compatible	Rechazo del archivo	Error controlado	Aprobado
Inserción de texto	Imagen cover + texto	Imagen stego generada	Payload de texto insertado	Aprobado
Inserción de PDF	Imagen cover + PDF	Imagen stego generada	PDF insertado correctamente	Aprobado
Extracción manual	Imagen stego + configuración correcta	Payload recuperado	Contenido extraído	Aprobado
Auto extracción LSB-R	Imagen stego sin indicar configuración	Configuración detectada automáticamente	Payload recuperado	Aprobado
Validación SHA-256	Payload recuperado	Hash valido	Integridad confirmada	Aprobado
Reporte integrado	Resultado de análisis integrado	PDF coherente con pantalla	Reporte generado correctamente	Aprobado

Una vez comprobado el correcto funcionamiento individual de los distintos módulos del sistema, se realizaron pruebas de validación hibrida con el propósito de evaluar el comportamiento de la estrategia de decisión integrada implementada con StegoDetect.

Tabla 15. Pruebas de validación hibrida.

Imagen	Payload	Resultado ML	Resultado LSB-R	SHA-256	Dominio del modelo	Decisión general
StegoDetect con texto	Texto	Puntaje secundario	Payload encontrado	Valido	Puede ser externo	Mensaje oculto encontrado
StegoDetect con PDF	PDF	Puntaje secundario	Payload encontrado	Valido	Puede ser externo	Mensaje oculto encontrado
BOSSBase cover	Ninguno	Bajo	Sin cabecera	No aplica	Compatible	Sin evidencia detectable
BOSSBase stego	No compatible con StegoDetect	Alto	Sin cabecera	No aplica	Compatible	Posibles patrones compatibles con esteganografía
Wallpaper externo limpio	Ninguno	Puede ser alto	Sin cabecera	No aplica	Fuera del dominio	Resultado ML no concluyente
Imagen externa con payload StegoDetect	PDF/texto/binario	Secundario	Payload encontrado	Valido	Puede ser externo	Mensaje oculto encontrado

Los resultados obtenidos muestran que el sistema responde adecuadamente ante distintos escenarios de uso. Cuando existe payload compatible con StegoDetect y validación SHA-256 correcta, la decisión general es “Mensaje oculto encontrado”. Cuando no existe payload, pero el modelo identifica patrones en imágenes compatibles con su dominio de

entrenamiento, el sistema presenta una alerta basada en evidencia ML. Por otro lado, cuando la imagen esta fuera del dominio del modelo, el resultado ML se considera orientativo y no concluyente.

En síntesis, el desarrollo e implementación de StegoDetect permitió construir un sistema funcional de esteganografía y estegoanálisis hibrido, capaz de insertar, detectar, extraer y validar información oculta en imágenes digitales. La combinación de técnicas LSB-R, Machine Learning, validación SHA-256, automatización y reportes técnicas integrados permitió superar la visión inicial de un detector CNN simple, consolidando un prototipo más completo, coherente y alineado con las necesidades actuales de seguridad de la información.

CAPITULO V: CONCLUSIONES Y RECOMENDACIONES

5.1 Conclusiones

La investigación desarrollada sobre esteganografía, estegoanálisis y Machine Learning aplicado a imágenes digitales permite establecer las siguientes conclusiones:

- La validación experimental realizada durante el desarrollo de StegoDetect permitió comprobar que la evidencia obtenida mediante extracción LSB-R y verificación SHA-256 constituye un mecanismo mas concluyente para demostrar la existencia de información oculta que una predicción basada únicamente en Machine Learning.
- Las pruebas realizadas con imágenes provenientes de BOSSBase evidenciaron que ResidualSRNetLite es capaz de identificar patrones compatibles con modificaciones esteganográficas introducidas mediante LSB-R. Sin embargo, la evaluación con imágenes externas a este dataset mostros que el desempeño del modelo disminuye cuando las características visuales difieren significativamente del dominio de entrenamiento utilizado.
- Durante el proceso de integración del sistema se identificó que una decisión basada exclusivamente en el resultado del modelo podía producir interpretaciones incorrectas cuando existía un payload recuperable. La incorporación de un motor de decisión integrada permitió resolver este problema y priorizar la evidencia determinística obtenida mediante extracción LSB-R y validación criptográfica.
- La investigación permitió demostrar la viabilidad de integrar técnicas de esteganografía, estegoanálisis, aprendizaje profundo y desarrollo web dentro de una única plataforma funcional. El resultado fue un sistema diseñado para insertar, detectar, extraer y verificar información encubierta incrustada dentro de imágenes digitales.

5.2 Recomendaciones

Teniendo en cuenta los resultados alcanzados, las pruebas realizadas y las limitaciones observadas en el desarrollo de StegoDetect, se sugieren los siguientes avances:

- Para fortalecer el marco experimental de la plataforma, se debe probar la capacidad del sistema para ocultar información de diversas maneras mediante la incorporación de otros métodos esteganográficos además de LSB-R.
- Para mejorar la aplicabilidad del modelo, el desarrollo del mismo debe continuar utilizando técnicas más avanzadas que permitan estimar automáticamente el nivel de confianza de cada predicción realizada por la red neuronal.

- Para apoyar el análisis forense digital, StegoDetect debe ser desarrollado aún más para soportar el uso de infraestructura en la nube incorporando mecanismos de almacenamiento persistente, auditoría y autenticación.
- Para mejorar el rendimiento del modelo, se debe realizar una validación externa formal para medir métricas específicas de resultados falsos positivos y falsos negativos en casos del mundo real para imágenes que no formaron parte de los datos de entrenamiento.

Bibliografía

Alrusaini, O. A., Alqahtani, H. M., & Alzubi, J. A. (2025). Deep learning for steganalysis: Evaluating model robustness against common image transformations. *Frontiers in Artificial Intelligence, 8*, Article 1532895. https://doi.org/10.3389/frai.2025.1532895

Bas, P., Filler, T., & Pevný, T. (2011). "Break our steganographic system": The ins and outs of organizing BOSS. En *Proceedings of the 13th International Conference on Information Hiding* (pp. 59–70). Springer. https://doi.org/10.1007/978-3-642-24178-9_5

Boroumand, M., Chen, M., & Fridrich, J. (2019). Deep residual network for steganalysis of digital images. *IEEE Transactions on Information Forensics and Security, 14*(5), 1181–1193. https://doi.org/10.1109/TIFS.2018.2871749

Cheddad, A., Condell, J., Curran, K., & Mc Kevitt, P. (2010). Digital image steganography: Survey and analysis of current methods. *Signal Processing, 90*(3), 727–752. https://doi.org/10.1016/j.sigpro.2009.08.010

Fridrich, J., & Kodovský, J. (2012). Rich models for steganalysis of digital images. *IEEE Transactions on Information Forensics and Security, 7*(3), 868–882. https://doi.org/10.1109/TIFS.2012.2190402

Guaña-Moya, J. (2023). Usos y aplicaciones de la esteganografía en la era digital. *Revista Retos para la Investigación, 2*(1), 61–75. https://editorialscientificfuture.com/index.php/rri/article/view/21

Oliva Paz, A. E. (2021). *Creación de una herramienta para ocultamiento de información a través de imágenes* [Tesis de pregrado, Pontificia Universidad Católica del Ecuador]. Repositorio PUCE. https://repositorio.puce.edu.ec/items/0029a6ef-deb9-4d43-937e-fbb39a0fd04f

Ramírez Valle, S. A. (2021). *Estudio de algoritmos de IA aplicables al estegoanálisis de imágenes digitales* [Tesis de maestría, Tecnológico Nacional de México]. Repositorio Nacional TecNM. https://rinacional.tecnm.mx/handle/TecNM/4168

Romero, A. F., Fajardo, A. M. C., & Lorente, A. S. (2018). Seguridad de imágenes digitales mediante un algoritmo esteganográfico en el dominio de la frecuencia. *Roca: Revista*

Científico-Educacional de la Provincia de Granma, 14(2), 179–190. https://dialnet.unirioja.es/servlet/articulo?codigo=6759661

Rosas, M. (2024). Esteganografía: El arte de ocultar información en medios digitales. *Polo del Conocimiento, 9*(2), 180–200. https://polodelconocimiento.com/ojs/index.php/es/article/view/6626

Sánchez-Rinza, B. E., & Hernández-Flores, M. R. (2025). Esteganografía lingüística en texto en español. *Revista ELECTRO, 47*, 13–16. https://itchihuahua.mx/revista_electro/2025/A16_13-16.html

Shehab, D., & Alhaddad, M. (2025). Image steganalysis using LSTM fused convolutional neural networks for secure telemedicine. *Frontiers in Medicine, 12*, Article 1619706. https://doi.org/10.3389/fmed.2025.1619706

Xu, G., Wu, H.-Z., & Shi, Y.-Q. (2016). Structural design of convolutional neural networks for steganalysis. *IEEE Signal Processing Letters, 23*(5), 708–712. https://doi.org/10.1109/LSP.2016.2548421

Ye, J., Ni, J., & Yi, Y. (2017). Deep learning hierarchical representations for image steganalysis. *IEEE Transactions on Information Forensics and Security, 12*(11), 2545–2557. https://doi.org/10.1109/TIFS.2017.2710946

Zhou, Y., Wang, N., Zhang, H., & Shao, S. (2025). Deep learning-based image steganography with latent space embedding and smart decoder selection. *Entropy, 27*(12), Article 1223. https://doi.org/10.3390/e27121223