



BENEMÉRITA UNIVERSIDAD AUTÓNOMA DE PUEBLA
FACULTAD DE CIENCIAS DE LA COMPUTACIÓN
MAESTRÍA EN CIENCIAS DE LA COMPUTACIÓN

SISTEMA DE RECUPERACIÓN DE INFORMACIÓN BASADO EN
CODIFICACIÓN DE IMAGEN-TEXTO PARA LA IDENTIFICACIÓN
DE ESCENAS DE INTERÉS EN VIDEOS DE LARGA DURACIÓN

T E S I S

PARA OBTENER EL GRADO DE:

MAESTRO EN CIENCIAS DE LA COMPUTACIÓN

P R E S E N T A :

LIC. EDGAR ABIDÁN PADILLA LUIS

ASESOR:

DR. DAVID EDUARDO PINTO AVENDAÑO

COASESOR:

DR. RIGOBERTO CERINO JIMÉNEZ

HEROICA PUEBLA DE ZARAGOZA, PUEBLA, MÉXICO.

SEPTIEMBRE 2024

Dedicado a mis padres:
Roberto Padilla y Graciela Luis

Agradecimientos

Al culminar esta etapa en mi vida me es imposible no ver hacia el pasado y notar cuanto he crecido personal y profesionalmente. Puedo afirmar que esta etapa de mi vida ha sido un periodo de mucho aprendizaje, ya que he ampliado mis conocimientos al involucrarme en diversos proyectos académicos y profesionales. Sin embargo, también ha sido un periodo de perdida ya que lamentablemente mis abuelas fallecieron y no lograron ver que he desbloqueado este logro llamado maestría en este juego titulado vida. Por ende, al final de esta etapa no me quedan más que palabras de agradecimientos a todas las personas que me apoyaron.

Agradezco infinitamente a mi compañera de vida, Elizabeth, a quien amo y amaré eternamente. Este logro también es tuyo, ya que has estado a mi lado en cada segundo de esta etapa y gracias a tu apoyo, a tus palabras tiernas, a tus palabras duras y a tu amor he podido concluir este proyecto.

A mi madre y padre, les agradezco su amor y apoyo incondicional, todo lo que soy se los debo a ustedes, pues sus consejos y palabras me acompañan en cada paso que doy.

Le doy gracias a mis hermanos y hermana, aunque seamos tan diferentes, se que tengo su apoyo y cariño, al igual que ustedes siempre tendrán mi apoyo y cariño.

A mis directores de tesis, Dr. David Eduardo Pinto Avendaño y Dr. Rigoberto Cerino Jiménez, les agradezco enormemente su guía y apoyo en este proceso. Su paciencia, enseñanzas y comentarios han conducido a este proyecto a una conclusión más que satisfactoria.

Le agradezco a mi comité revisor, Dra. Beatriz Beltrán Martínez, Dr. José Andrés Vázquez Flores y M.C. Francisco José López Cortés, gracias a sus correcciones y acertadas observaciones esta tesis se ha pulido para presentar un gran trabajo.

Agradezco especialmente al Consejo Nacional de Humanidades Ciencia Y Tecnología (CONAHCYT) por el apoyo económico brindado, este apoyo fue destinado para cubrir mis

gastos de manutención, sin esta beca no hubiera sido posible estudiar la maestría.

A mis compañeros Alejandro y Alberto por compartir su conocimiento conmigo, por la solidaridad, amistad y el apoyo brindado en estos últimos dos años.

Finalmente, agradezco a la Benemérita Universidad Autónoma de Puebla, a su Facultad de Ciencias de la Computación, a los docente que me dieron clases y al personal administrativo por ayudarme a crecer profesionalmente.

Índice general

Agradecimientos	IV
Índice general	VII
Índice de figuras	VIII
Índice de tablas	X
1. Introducción	1
1.1. Planteamiento del problema	2
1.2. Preguntas de investigación	3
1.3. Objetivos	4
1.3.1. Objetivo general	4
1.3.2. Objetivos particulares	4
1.4. Hipótesis planteada	4
1.5. Estructura de la tesis	5
2. Marco teórico	7
2.1. Sistemas de recuperación de información de texto	7
2.1.1. Vocabulario	8
2.1.2. Índice invertido	10
2.1.3. Recuperación de información booleana	12
2.1.4. Recuperación de información tolerante a fallos	15
2.1.4.1. Búsqueda por sinónimos	15
2.1.4.2. Algoritmo de Levenshtein	17

2.1.5.	Modelo de espacio vectorial	21
2.2.	Recuperación de información multimedia	23
2.2.1.	Recuperación de información multimedia de imágenes	23
2.2.2.	Sistemas de recuperación multimedia de imágenes basados en redes neuronales.	24
3.	Estado del arte	29
4.	Algoritmo de codificación imagen-texto	34
4.1.	Metodología	34
4.2.	Algoritmo de conversión de imagen a texto	35
4.3.	Primer experimento: Conversión a caracteres mediante el promedio de grises	36
4.4.	Segundo experimento: Conversión a cadenas de códigos Hash	42
4.5.	Discusión sobre los experimentos	46
4.6.	Algoritmo de cercanía entre tokens	47
4.7.	Conclusiones.	52
5.	Sistema de recuperación de escenas de interés en videos de larga duración	53
5.1.	Metodología	53
5.2.	Desarrollo del sistema de recuperación de información	55
5.2.1.	Creación del corpus de texto	55
5.2.2.	Sistema de recuperación de información booleano tolerante a fallos con búsqueda por sinónimos	56
5.3.	Particularidades del hardware y del sistema de recuperación de información .	57
5.4.	Resultados obtenidos al implementar el sistema de recuperación de información	59
5.5.	Conclusiones	65
6.	Conclusiones	67
	Bibliografía	69

Índice de figuras

2.1. Sistema IR convencional.	8
2.2. Preprocesamiento en un sistema IR.	10
2.3. Estructura de un índice invertido.	10
2.4. Índice invertido al agregar la palabra <i>Perro</i> al documento 13.	11
2.5. Índice invertido con frecuencia de documentos.	12
2.6. Índice invertido como componente principal de un sistema IR convencional. .	12
2.7. Sistema de recuperación de información booleano.	15
2.8. Sistema de recuperación de información con expansión por sinónimos.	17
2.9. Sistema de recuperación de información booleano con expansión de consul- tas por sinónimos y corrección ortográfica.	20
2.10. Esquema de una red neuronal con una capa de entrada, una capa oculta y una capa de salida.	25
4.1. Metodología propuesta para la identificación de escenas de interés en videos de larga duración mediante un módulo de codificación de imagen a texto. . .	35
4.2. Segmentación del frame 13 obtenido del video.	38
4.3. Representación textual generado del frame 13 al implementar el mapeo de conversión a caracteres mediante el promedio de grises.	39
4.4. Frames cercanos al frame 13	40
4.5. Aplicación del mapeo basado en promedio de grises sobre la franja del sombrero	40
4.6. Segmentación del frame 13 al aplicar el algoritmo basado en códigos Hash. .	43
4.7. Texto asociado al frame 13 al aplicar el mapeo basado en códigos Hash. . . .	44
4.8. Analizando frame 13 con el código Hash.	45

4.9. Estructura de un archivo de texto cuya estructura se asemeja a una matriz de 6 filas y 6 columnas.	47
4.10. Experimento: Analizar la cadena asociada a la franja del sombrero.	49
4.11. Experimento: Analizar la cadena asociada a las piernas de una persona.	50
4.12. Experimento: Analizar la cadena asociada al cabello de una persona.	50
4.13. Experimento: Analizar la cadena asociada a la fecha, en particular al segmento de imagen que muestra los caracteres 2023-Feb.	51
4.14. Experimento: Analizar la cadena asociada a un azulejo en el piso.	51
5.1. Metodología del sistema de recuperación de objetos para una rápida identificación de objetos en videos de larga duración.	54
5.2. Sistema de recuperación de información boleando tolerante a fallos con búsqueda por sinónimos implementado	56
5.3. Interfaz web desarrollada, en ella se presenta un cuadro de búsqueda en donde se escribe la consulta <i>person</i> . En la parte inferior del cuadro de búsqueda la aplicación web posee un de búsqueda y un botón que redirige al inicio. En la parte inferior se presenta los resultados de la consulta.	60
5.4. Primer imagen recuperada de la consulta <i>person</i>	61
5.5. Respuesta del sistema a la consulta <i>perosn</i> . En la interfaz se observa el funcionamiento del modulo tolerante a fallos al devolver resultados que corresponden a la palabra <i>person</i>	62
5.6. Respuesta del sistema a la consulta <i>person with phone</i> . Esta consulta posee el conector <i>with</i> que hace referencia al operador lógico <i>AND</i>	63
5.7. Respuesta del sistema a la consulta booleana especializada <i>person AND phone</i>	64
5.8. Respuesta del sistema a la consulta <i>human</i> . Se observan los mismos resultados a la consulta <i>person</i> , mostrando el funcionamiento del módulo de expansión de consulta por sinónimos.	65

Índice de tablas

1.1. Tiempos de búsqueda a velocidad $\times 4$	2
3.1. Revisión del estado del arte.	32
4.1. Cadenas con una distancia de Levenshtein menor o igual a 3.	41
4.2. Documentos recuperados al aplicar recuperación de información tolerante a fallos.	45
5.1. Datos obtenidos al generar el corpus de texto tomando diferentes intervalos de tiempo para obtener un frame del video.	59

Capítulo 1

Introducción

Un viaje de mil millas comienza con un solo paso.

Lao-TSe

En los últimos años, debido al desarrollo y creación de mejores dispositivos tecnológicos se ha visto un aumento en el uso de cámaras digitales (teléfonos celulares, sistemas de video vigilancia, cámaras especializadas, etc.), en consecuencia se crean grandes volúmenes de archivos multimedia. Recolectar información de estos masivos compendios trae consigo diversos desafíos, que dependen del tipo de datos que se quiera recuperar (video, imágenes, audios o texto), algunas de las técnicas que se implementan pertenecen al área de *Machine Learning*, sobresaliendo el aprendizaje profundo, ya que los modelos creados en esta área son capaces de extraer características automáticamente a partir de los datos masivos con los que se entrenan [Dubey (2021)].

A pesar del aumento de nuevos estudios en el área de inteligencia artificial, muchos organismos públicos y privados aún hacen uso de recurso humano para trabajos específicos como búsquedas en videos. Obtener información en videos de larga duración mediante el uso de personal operativo suele ser tardado y consumir muchas horas-hombre, esto se debe al hecho de que identificar escenas en los videos es una tarea difícil y tediosa, especialmente cuando se trata de grandes volúmenes de datos. En contraste, utilizar técnicas de recuperación de información de imágenes ha demostrado ser de ayuda para reducir el costo humano en la identificación de escenas y objetos presentes en videos [Haering et al. (2008)].

Motivados en la tarea de video vigilancia, este trabajo de tesis presenta un sistema de

recuperación en videos de larga duración que codifica imágenes textual para identificar las escenas de interés presentes en el video.

1.1. Planteamiento del problema

Imaginemos la siguiente situación: “Usted es el responsable de la seguridad en una oficina, pero hoy no es un día fácil. Le han informado sobre la desaparición de parte del mobiliario. Ahora, debe revisar las grabaciones del sistema de vigilancia en busca de pistas que expliquen lo sucedido”. De forma inmediata, una persona podría hacerse la siguiente pregunta, ¿Cuánto tiempo le llevará revisar el contenido de las grabaciones para encontrar una respuesta a lo sucedido? En la tabla 1.1 se muestra el tiempo que le tomaría revisar videos de diferente duración a una velocidad $\times 4$. En esta tabla se observa que, si se busca en un video con una duración de 2 días (lo que regularmente se almacena en servidores de video vigilancia), se necesitaría más de una jornada laboral para dar con alguna respuesta sobre lo sucedido, lo cual resulta ser muy costoso. Otra solución es repartir el trabajo de vigilancia en más de una persona, lo cuál no es práctico, ya que el personal de seguridad tiene diversas tareas que deben ser cumplidas en sus jornadas laborales.

Duración del video	Tiempo de búsqueda a velocidad $\times 4$
4 minutos	1 minuto
1 hora	15 minutos
8 horas	2 horas
1 día	6 horas
2 días	12 horas
1 semana	1.75 días

Tabla 1.1: Tiempos de búsqueda a velocidad $\times 4$.

Actualmente, el avance en el aprendizaje profundo (deep learning, por sus siglas en inglés) nos ofrece diversas herramientas que pueden ayudar a resolver esta problemática. Sin embargo, estas técnicas analizan imágenes, por lo que es necesario tener en cuenta los siguientes aspectos si se requiere implementar este tipo de soluciones:

- Se necesita un gran corpus para entrenar los modelos de deep learning.
- El entrenamiento de estos modelos suele tardar demasiado tiempo.
- Se necesita un modelo especializado.
- Los recursos computacionales son altos; se requieren computadoras con gran capacidad de RAM, GPU, procesador y espacio en disco.

Este proyecto de tesis utiliza un enfoque basado en la recuperación de información de texto para identificar escenas de interés en videos de larga duración. Estas técnicas tienden a requerir menos recursos computacionales en comparación a las técnicas basadas en deep learning, lo cual las hace ideales para su adaptación en muchos servidores que almacenan videos o imágenes.

1.2. Preguntas de investigación

Naturalmente, surgen las siguientes preguntas de investigación que se desprenden del problema planteado:

- P1** ¿Qué tipos de algoritmos existen en la literatura para lograr la codificación de una imagen a una representación textual, no necesariamente en lenguaje natural?
- P2** ¿Qué características debe tener un algoritmo de indexación para ser utilizado en un sistema de recuperación de información en video?
- P3** ¿Qué tan eficaces son las representaciones textuales de imágenes que existen en la literatura para efectuar la búsqueda de escenas dentro de videos?
- P4** ¿Cuáles son las ventajas y desventajas de utilizar estas técnicas frente a las metodologías que realizan comparaciones entre imágenes?
- P5** ¿Qué tan eficaz puede llegar a ser el sistema de recuperación multimedia propuesto para identificar escenas en video que contengan personas y objetos específicos?

En este documento se abordan estas cuestiones mediante una investigación del estado del arte y la creación e implementación de dos sistemas de recuperación basados en codificación de imagen a texto.

1.3. Objetivos

Para responder las preguntas de investigación planteadas, se formularon el objetivo general y los objetivos específicos, los cuales se presentan a continuación.

1.3.1. Objetivo general

Crear un modelo de recuperación de información basado en codificación textual de imágenes para identificar escenas de interés en videos de larga duración.

1.3.2. Objetivos particulares

- Identificar en la literatura algoritmos de representación de imágenes a texto que existen actualmente.
- Utilizar la comprensión adquirida para proponer un algoritmo que genere representaciones de imágenes a texto en lenguaje no necesariamente natural.
- Implementar una plataforma para la adquisición de videos que puedan ser utilizados posteriormente como un corpus para el sistema de recuperación que se ha planteado que se pretende desarrollar.
- Adaptar el modelo de recuperación de información para que utilice las representaciones textuales obtenidas.
- Diseñar un algoritmo de búsqueda por cercanía entre tokens que representen a cada una de las imágenes.
- Implementar la integración del mecanismo de cercanía para la identificación de escenas de interés en videos de larga duración.

1.4. Hipótesis planteada

Se define la siguiente hipótesis, que se buscará aceptar o refutar a lo largo de esta tesis:

- **Hipótesis:** las representaciones textuales de imágenes, obtenidas mediante el algoritmo de codificación imagen a texto, son eficientes al realizar búsquedas de escenas de interés en vídeos de larga duración en comparación a los métodos que empatan imagen a imagen. La mayor ventaja que se espera obtener es el tiempo en recuperación de las escenas de interés las cuales deberán mostrarse en milisegundos. En contra parte, se piensa que esta técnica tenga una menor precisión.

1.5. Estructura de la tesis

Este documento está estructurado de la siguiente manera:

- El Capítulo 2 aborda el marco teórico. En este capítulo se presenta la teoría de los sistemas de recuperación y se explican de manera detallada los conceptos esenciales en la recuperación de información textual, que sirven como base para la creación de sistemas de recuperación, como el famoso motor de búsqueda Google. Posteriormente, se detalla cómo se crean sistemas de recuperación de información multimedia mediante redes neuronales.
- En el Capítulo 3 se expone el estado del arte, en este apartado se presentan los trabajos relacionados con el tema de tesis así como las aplicaciones que se han creado al rededor de la teoría de recuperación de información multimedia y la video vigilancia. Se presenta un análisis de las técnicas ocupadas en los artículos presentados en el estado del arte, de esta forma podemos apreciar el impacto de nuestro proyecto de tesis al proponer un método innovador en la recuperación de información multimedia.
- El desarrollo de la tesis se presenta en los Capítulos 4 y 5. En el capítulo 4 se propone un sistema de recuperación de información que utiliza un algoritmo propuesto en este proyecto para codificar una imagen a texto y, mediante un algoritmo de cercanía, encuentra conjuntos de subimágenes relacionadas entre sí. En el capítulo 5, se implementa un sistema de recuperación de información que usa un modelo de detección de objetos para mapear una imagen a un conjunto de palabras. De esta forma se crea un algoritmo que convierte una imagen en una representación texto, y estas representa-

ciones textuales se utilizan para crear un corpus de texto que alimenta a un sistema de recuperación de información textual.

- En el Capítulo 6 se presentan los resultados obtenidos. Finalmente, en el Capítulo 7 se exponen las conclusiones de este trabajo de tesis.

Capítulo 2

Marco teórico

Así como el presente es una consecuencia, un resultado del pasado, el futuro es una prolongación del presente. Todo se sostiene; el pasado, el presente y el futuro no están separados. El futuro se edificará sobre los cimientos que coloquéis ahora.

“Reglas de oro para la vida cotidiana” (1957),

Omraam Mikhael Aivanhov

En un mundo donde la cantidad de datos parecen crecer exponencialmente, se vuelve necesario desarrollar herramientas tecnológicas capaces de encontrar información relevante en estos volúmenes masivos de datos, el campo de la recuperación de información se enfoca en desarrollar métodos y técnicas para recuperar documentos o datos relevantes que satisfagan la necesidad de información de los usuarios.

Este capítulo aborda los sistemas de recuperación de información, destacando los conceptos y técnicas utilizados en la recuperación de textos y multimedia, así como el uso de redes neuronales como base para sistemas de recuperación multimedia.

2.1. Sistemas de recuperación de información de texto

Manning en [Manning (2009)] describe a la recuperación de información (*IR* por sus siglas en inglés) como el proceso para encontrar material no estructurado (generalmente texto), que satisfaga una necesidad de información, dentro de bastos volúmenes de datos (normal-

mente documentos de textos) llamado *corpus*.

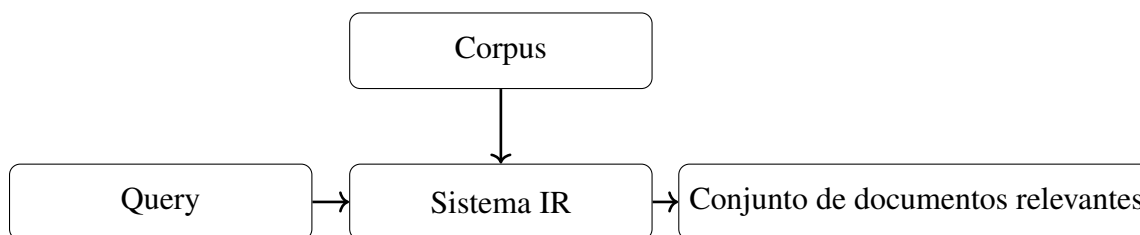


Figura 2.1: Sistema IR convencional.

En la Figura 2.1 se muestra el diagrama de un sistema de recuperación de información (*Sistema IR*) convencional. Este sistema tiene como propósito encontrar los documentos más pertinentes para una consulta (en este documento se utilizará el término *query* para referirse a una consulta) específica en un corpus. Surge de forma natural la interrogante: ¿Cómo opera un sistema IR? En los siguientes apartados se presentará la teoría esencial de un sistema IR y se incluirán mas elementos al diagrama mostrado en la Figura 2.1. La teoría que se muestra en las siguientes secciones se basa en los conceptos presentados en el libro Introduction to Information Retrieval [Manning (2009)].

2.1.1. Vocabulario

El conjunto de términos o palabras únicas que aparecen en una colección de documentos constituye el *vocabulario* de un corpus. Es importante destacar que en los campos de la recuperación de información y el procesamiento de lenguaje natural, el término comúnmente utilizado es "token", el cual suele equipararse con la noción de palabra. No obstante, el concepto de *token* abarca una gama más amplia, incluyendo palabras, números, signos de puntuación y otras secuencias de caracteres [Vasilev (2020)]. Con esta aclaración, en este documento se utilizará el término sin incurrir en ambigüedad. Para obtener el vocabulario de un documento, se suele dividir el texto en fragmentos más pequeños. Esta división se lleva a cabo frecuentemente mediante los espacios en blanco y saltos de línea, un proceso conocido como *tokenización*.

Ejemplo 2.1.1

Suponga que se tiene un documento de texto con el contenido siguiente:

Todos tenemos un papel que desempeñar en la protección de nuestro planeta. Desde reducir nuestro consumo de plástico hasta apoyar políticas ambientales sólidas, cada acción cuenta. Es hora de actuar con urgencia y trabajar juntos para preservar el medio ambiente y asegurar un futuro sostenible para todos.

El vocabulario del documento será el siguiente:

Todos, tenemos, un, papel, que, desempeñar, en, la, protección, de, nuestro, planeta., desde, reducir, consumo, plástico, hasta, apoyar, políticas, ambientales, sólidas,, cada, acción, cuenta., Es, hora, actuar, con, urgencia, y, trabajar, juntos, para, preservar, el, medio, ambiente, asegurar, un, futuro, sostenible, todos.

En el Ejemplo 2.1.1, se destaca la necesidad de considerar los detalles que surgen del proceso de tokenización. Por ejemplo, los tokens *Todos* y *todos*. se distinguen como cadenas diferentes debido a que la primera comienza con una letra mayúscula y la segunda comienza con minúscula, además de terminar con un punto. A pesar de estas diferencias superficiales, ambas cadenas representan la misma palabra y, por lo tanto, deben tratarse como un único token. Esto conduce típicamente a la inclusión de un proceso de normalización que convierte las letras mayúsculas a minúsculas y elimina los signos de puntuación. Dependiendo del contexto, puede ser conveniente también eliminar acentos y números. Si se aplica un proceso de normalización que solo convierta las letras mayúsculas a minúsculas y elimine los signos de puntuación en el texto del Ejemplo 2.1.1, se obtiene el siguiente vocabulario:

todos, tenemos, un, papel, que, desempeñar, en, la, protección, de, nuestro, planeta, desde, reducir, consumo, plástico, hasta, apoyar, políticas, ambientales,

sólidas, cada, acción, cuenta, es, hora, actuar,
 con, urgencia, y, trabajar, juntos, para, preservar,
 el, medio, ambiente, asegurar, un, futuro, sostenible

Generalmente, el proceso de normalización se ejecuta en una etapa llamada preprocesamiento, la cual desempeña un papel fundamental para lograr encontrar información relevante en un corpus de grandes dimensiones [Virmani and Taneja (2019)]. En la Figura 2.2 se agrega esta etapa a un sistema IRconvencional. Es importante mencionar que, de la misma manera que el corpus, la consulta debe pasar por un preprocesamiento

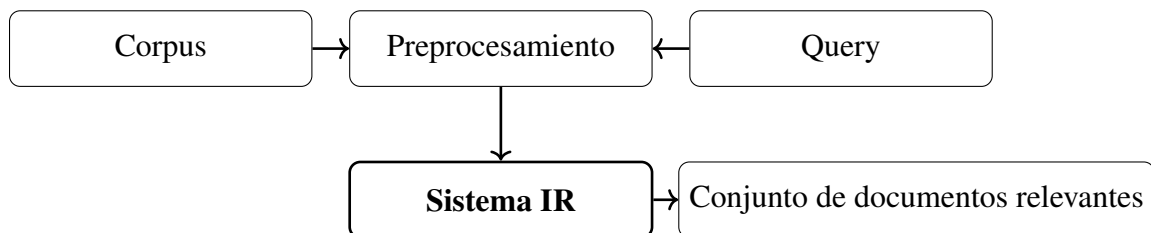


Figura 2.2: Preprocesamiento en un sistema IR.

2.1.2. Índice invertido

El índice invertido es una estructura de datos diseñada para agilizar y eficientar la búsqueda de documentos que contienen palabras clave o términos específicos de una consulta. Esta técnica opera asignando a cada término del corpus una lista de documentos que lo contienen. En ocasiones a cada lista de documentos se le asignan las posiciones donde el término aparece, esta estructura se conoce convencionalmente como *posting list*.

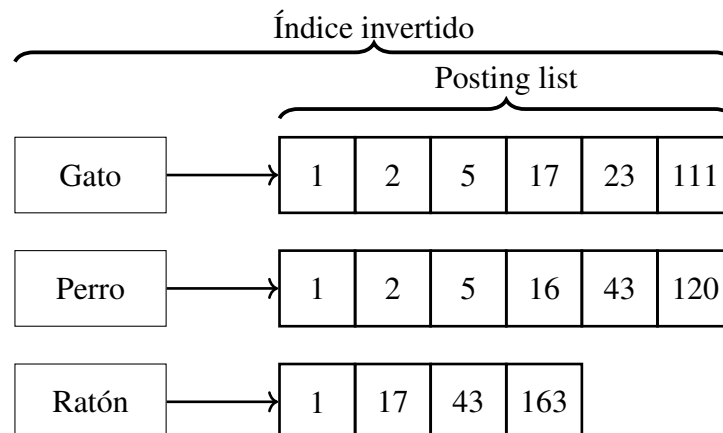


Figura 2.3: Estructura de un índice invertido.

En la Figura 2.3, se muestra un ejemplo de un índice invertido que incluye tres palabras: Gato, Perro y Ratón, cada palabra tiene asignada una lista de postings que contiene los documentos en los que aparece. Si se agrega la palabra *Perro* al documento 13, el índice invertido se debe modificar, esta modificación se aprecia en la Figura 2.4.

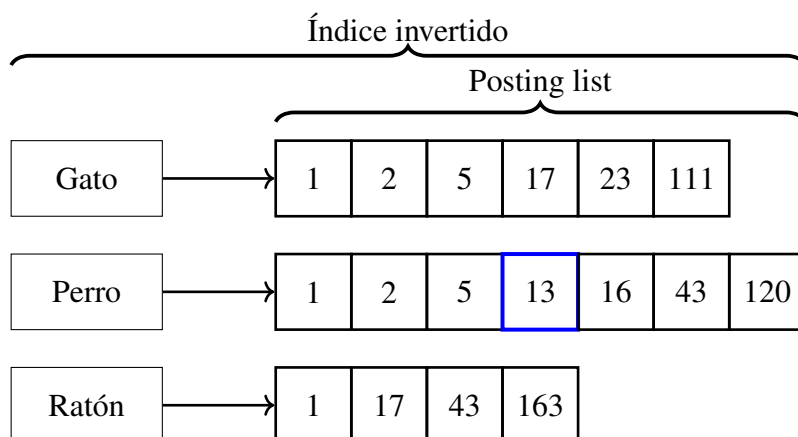


Figura 2.4: Índice invertido al agregar la palabra *Perro* al documento 13.

La preservación del orden en los documentos facilita operaciones lógicas al permitir una búsqueda eficiente a través de las listas de postings y la identificación de documentos que cumplen con las condiciones de la consulta. Esto es fundamental en sistemas de recuperación de información booleanos (de los cuales se hablará en el siguiente apartado) para ofrecer resultados precisos y relevantes a las consultas de los usuarios.

Un aspecto crucial en la construcción de índices invertidos es la frecuencia de términos en un documento, la cual se refiere al número de veces que una palabra aparece en un documento. Aunque este dato no resulta esencial para un sistema IR, sí permite optimizar la eficiencia de la consulta. Además, constituye una estadística utilizada para ponderar los resultados obtenidos, lo que mejora la relevancia y precisión de las respuestas proporcionadas por los sistema IR. En la Figura 2.5 se presenta un índice invertido que cuenta la frecuencia de documentos.

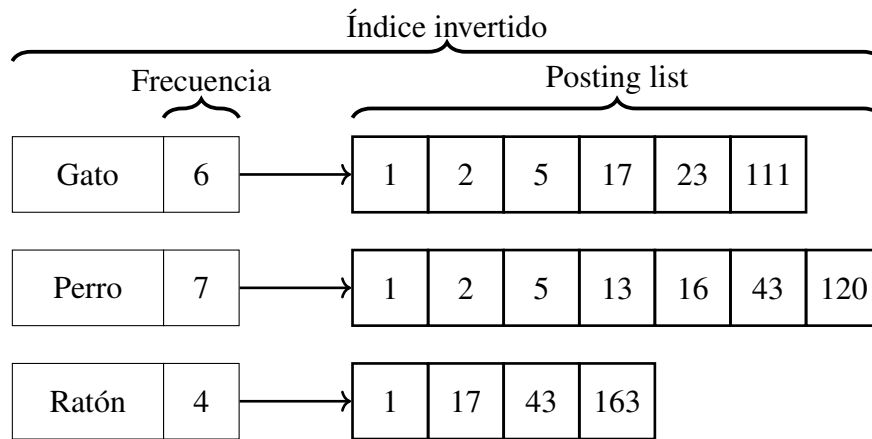


Figura 2.5: Índice invertido con frecuencia de documentos.

En la Figura 2.6 se resalta que el índice invertido es el componente principal de un sistema de recuperación de información convencional.

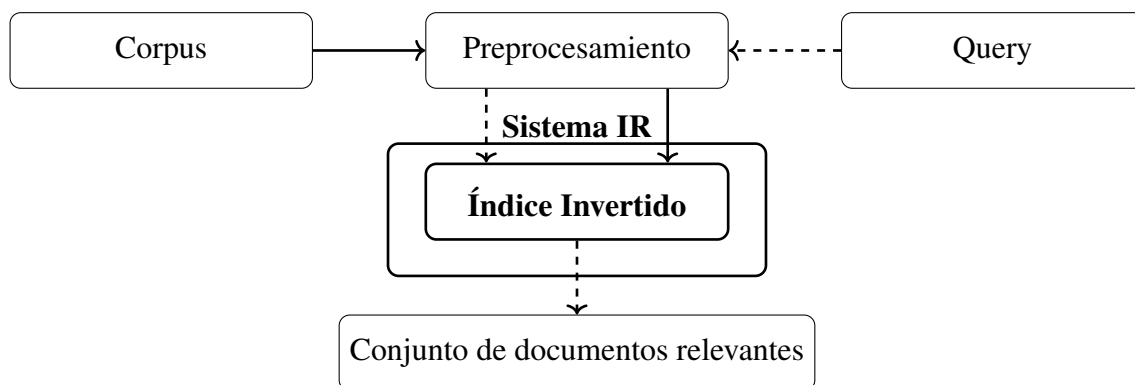


Figura 2.6: Índice invertido como componente principal de un sistema IR convencional.

Generalmente, los índices invertidos se implementan en diccionarios y árboles balanceados, este proceso de creación de un índice invertido se le llama *indexación* y se realiza después del preprocesamiento del corpus, ya que esta estructura es la base de los sistemas de recuperación de información más avanzados.

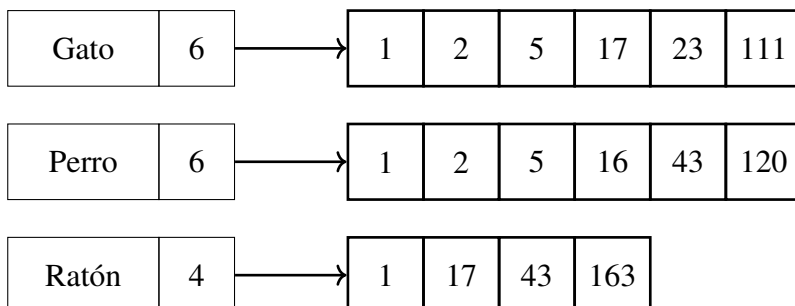
2.1.3. Recuperación de información booleana

En un sistema de recuperación booleana, los usuarios pueden formular consultas utilizando palabras clave y operadores booleanos para especificar la relación entre los términos de búsqueda. Por ejemplo, una consulta que tenga la siguiente estructura: *Gato AND Perro*, como resultado recupera documentos que contienen ambas palabras. Estos sistemas se basan en

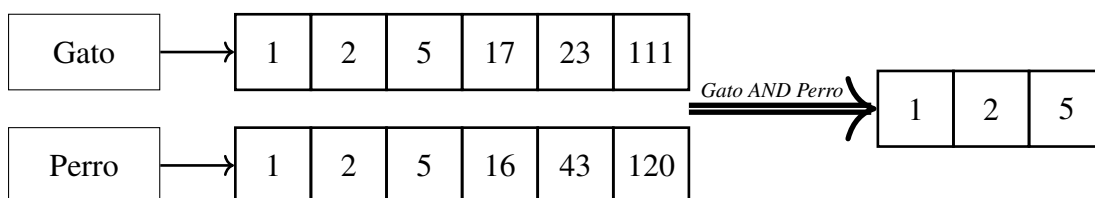
la lógica booleana para permitir a los usuarios realizar consultas complejas y específicas. Sin embargo, su capacidad para comprender el significado contextual de las consultas es limitada, lo que puede resultar en una recuperación de información menos relevante en comparación con otros enfoques más avanzados.

Ejemplo 2.1.2

Se considera el siguiente



Cuando un usuario busca documentos que contengan las palabras *Gato* y *Perro* juntas en un mismo documento, se lleva a cabo una búsqueda booleana. La consulta formulada es *Gato AND Perro*. En este contexto, el usuario utiliza el operador booleano *AND* para indicar que ambos términos deben aparecer simultáneamente en los documentos recuperados. Para lograr esto, el sistema IR se centra en las dos listas de postings a las que apuntan los términos buscados y debe devolver los documentos 1, 2 y 5.



El siguiente algoritmo presentado en el libro de Manning [Manning (2009)] ofrece una técnica simple y efectiva que aprovecha la estructura del índice invertido para realizar consultas *AND*, se puede adaptar este algoritmo para realizar consultas *OR* y *NOT*.

```

INTERSECT(p1, p2):
1  answer ← < >
2  while p1 != NIL and p2 != NIL

```

```

3 do if docID(p1) == docID(p2)
4   then ADD(answer, docID(p1))
5       p1 ← next(p1)
6       p2 ← next(p2)
7   else if docID(p1) < docID(p2)
8       then p1 ← next(p1)
9       else p2 ← next(p2)
10 return answer

```

este código opera de la siguiente manera:

1. Se inicializa una lista de respuestas *answer* como vacía.
2. Se ejecuta un bucle mientras no se halla llegado al final de ambas listas de postings *p1* y *p2*.
3. Dentro del bucle, se compara el *docID* (identificador de documento) de los nodos actuales **p1** y *p2*.
4. Si los *docID* son iguales, se agrega ese *docID* a la lista de respuestas *answer*.
5. Posteriormente, se mueve al siguiente nodo en ambas listas *p1* y *p2*.
6. Si los *docID* no son iguales, se avanza al siguiente nodo de la lista con el *docID* más pequeño.
7. Se repite este proceso hasta que se llegue al final de una de las listas.
8. Una vez que finalice el ciclo, se devuelve la lista de respuestas *answer* que contiene la intersección de los *docID* comunes entre los postings list.

En la Figura 2.7 se muestra un diagrama de un sistema de recuperación booleano. El proceso de indexación se realiza después del preprocesamiento del corpus y el índice invertido generado alimenta al sistema de recuperación booleano. En este ejemplo, el sistema se compone de los operadores *AND*, *OR* y *NOT*, que retornan los documento relevantes mediante algoritmos similares al planteado en esta sección.

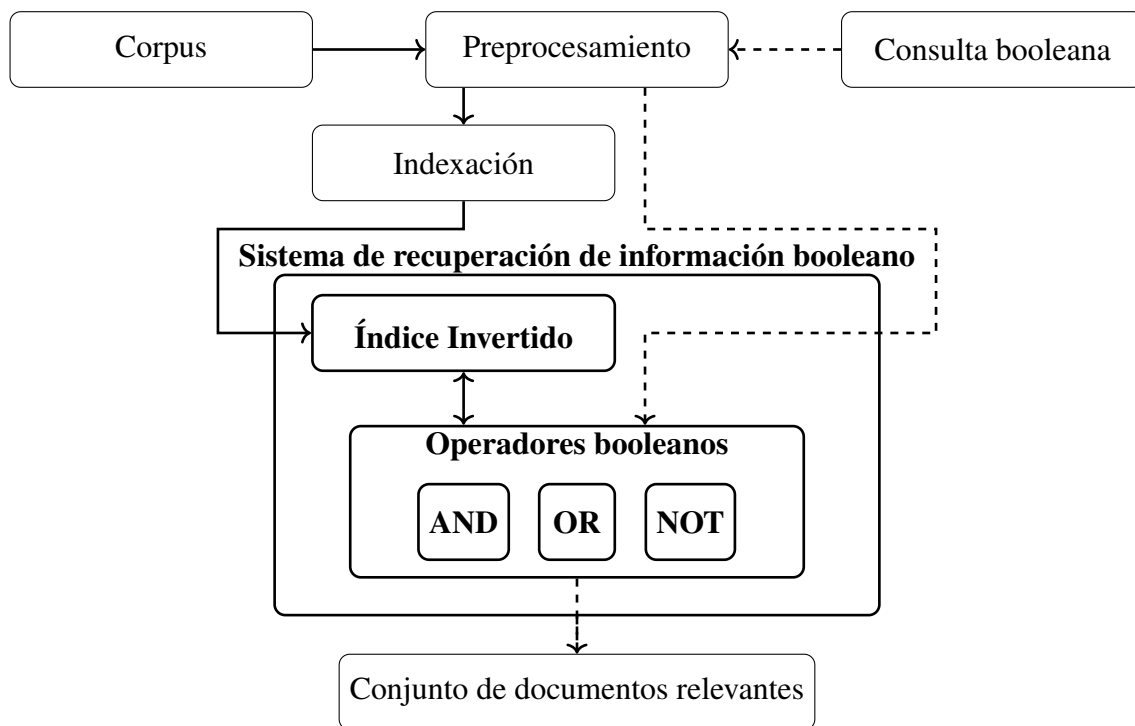


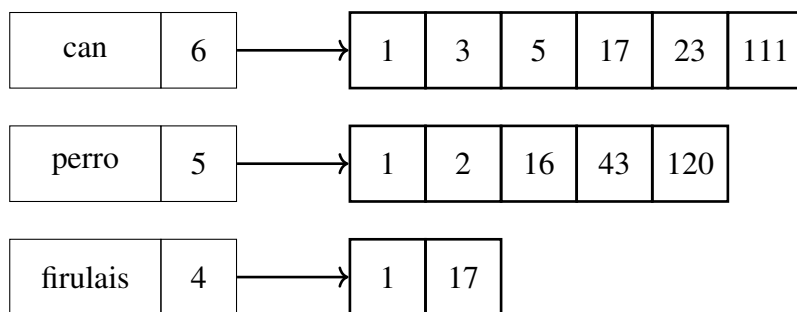
Figura 2.7: Sistema de recuperación de información booleano.

2.1.4. Recuperación de información tolerante a fallos

Cuando un usuario realiza una consulta en un motor de búsqueda, es posible que cometa errores al ingresar los términos de búsqueda, como errores de escritura, omisión de letras o adición de letras adicionales. La recuperación tolerante a fallos permite que el sistema de búsqueda encuentre resultados relevantes incluso cuando se cometen tales errores en la consulta. Esta técnica se basa en algoritmos que buscan términos similares a los términos de la consulta original y devuelve resultados que coinciden con esos términos similares. Esto puede implicar la corrección automática de errores ortográficos, la búsqueda de términos similares o la expansión de la consulta para incluir términos relacionados.

2.1.4.1. Búsqueda por sinónimos

En un ejemplo donde las palabras *can*, *perro* y *firulais* se encuentran en un índice invertido como se muestra a continuación:



Si el usuario realiza la consulta con la palabra *perro*, el sistema de recuperación de información solo regresará los documentos 1, 2, 16, 43 y 120. Tal vez, la información que necesita el usuario se encuentra en el documento 111, bajo la percepción del usuario, el sistema no arrojó los resultados deseados. Sin embargo, dado que las palabras *can* y *perro* son sinónimos, ampliar la consulta para incluir sinónimos del término ingresado permitiría al usuario recuperar más información relacionada con la consulta inicial, buscando en documentos donde se encuentren estos sinónimos del término ingresado, el sistema permitiría al usuario recuperar mayor información asociada a la consulta inicial [Vidal et al. (2015)], ya que la palabra *can* se encuentra en el documento 111, es decir, el documento que necesita el usuario.

Para realizar la expansión por sinónimos se debe contar con un diccionario de sinónimos, por ejemplo, el siguiente diccionario solo contiene los sinónimos de la palabra *perro*.

Can: Firulais, Perro.

Firulais: Can, Perro.

Perro: Can, Firulais.

Con ayuda de este diccionario, es posible buscar en el índice invertido las palabras *perro*, *can* y *firulais*, de esta forma el sistema recuperará los documentos 1, 2, 3, 5, 16, 17, 23, 111 y 120.

En la Figura 2.8 se aprecia el funcionamiento de la expansión por sinónimos en un sistema IR convencional.

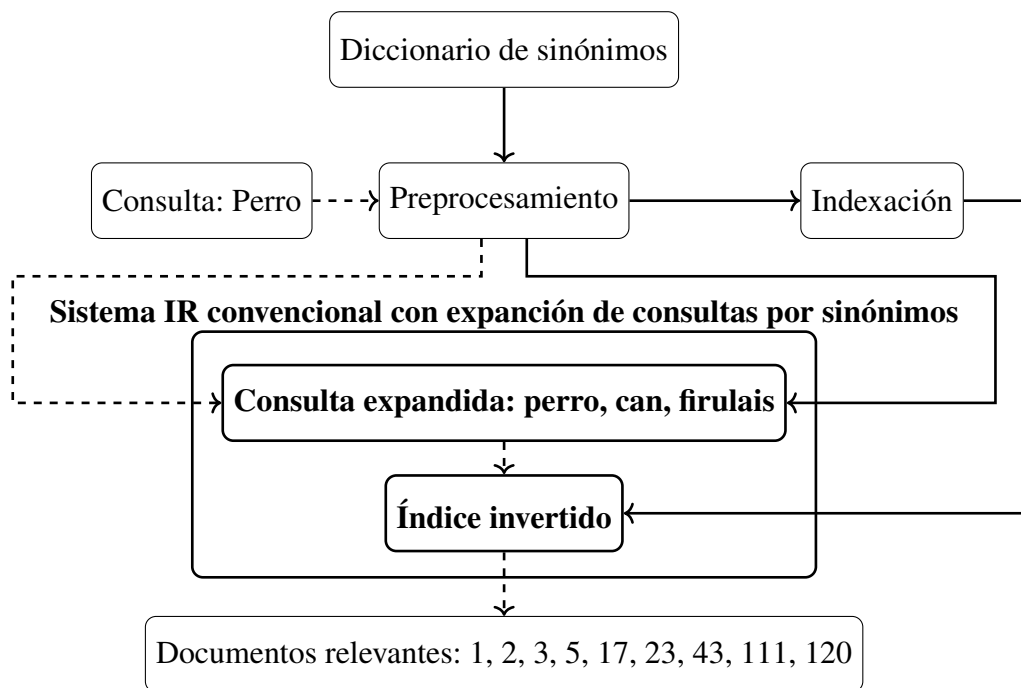


Figura 2.8: Sistema de recuperación de información con expansión por sinónimos.

2.1.4.2. Algoritmo de Levenshtein

En ocasiones, el usuario puede introducir términos incorrectamente, como por ejemplo escribir *perri* en lugar de *perro*. Este tipo de error es común y puede desafiar a los sistemas de recuperación de información convencionales, ya que no podrán localizar este token en su índice invertido, lo que resulta en la ausencia de respuestas. Por ello, es crucial que el sistema pueda sugerir al usuario palabras similares a la consulta realizada que estén presentes en su índice invertido.

La distancia de Levenshtein, es una técnica utilizada para calcular la distancia entre dos cadenas de caracteres, esta distancia se define como el número mínimo de operaciones requeridas para transformar una cadena en otra, donde las operaciones permitidas son la inserción, eliminación o sustitución de un solo carácter [Levenshtein et al. (1966)].

El proceso implica construir una matriz donde las filas representan los caracteres de una cadena y las columnas representan los caracteres de la otra cadena. Cada celda de la matriz contiene el costo mínimo de transformar una subcadena en otra subcadena. El algoritmo sigue estos pasos básicos:

1. Inicialización de la matriz: La primera fila y la primera columna de la matriz se inicia-

lizan con valores que representan el número de inserciones o eliminaciones necesarias para igualar una cadena vacía con la cadena correspondiente.

2. Cálculo de la distancia: A partir de la segunda fila y la segunda columna, se calculan los valores de cada celda de la matriz. Para cada celda (i, j) , el valor se calcula como el mínimo de tres posibles operaciones:

- La celda inmediatamente arriba más 1 (representando una operación de inserción).
- La celda inmediatamente a la izquierda más 1 (representando una operación de eliminación).
- La celda diagonalmente arriba a la izquierda más 1 si los caracteres en las posiciones (i, j) son diferentes, y la misma celda si son iguales (representando una operación de sustitución).

3. El valor en la esquina inferior derecha de la matriz representa la distancia de edición entre las dos cadenas originales.

Cuando la matriz se ha construido, se puede retroceder desde la esquina inferior derecha siguiendo los valores mínimos para determinar las operaciones necesarias para transformar una cadena en la otra.

Ejemplo 2.1.3

Se tienen las palabras *gato* y *patio*. Para calcular su distancia de Levenshtein se desarrollan los siguientes pasos:

1. Inicializar la matriz:

	—	<i>p</i>	<i>a</i>	<i>t</i>	<i>i</i>	<i>o</i>
—	0	1	2	3	4	5
<i>g</i>	1					
<i>a</i>	2					
<i>t</i>	3					
<i>o</i>	4					

2. Calcular la distancia mediante el mínimo de las tres operaciones posibles:

–	–	<i>p</i>	<i>a</i>	<i>t</i>	<i>i</i>	<i>o</i>
–	0	1	2	3	4	5
<i>g</i>	1	1	2	3	4	5
<i>a</i>	2	2	1	2	3	4
<i>t</i>	3	3	2	1	2	3
<i>o</i>	4	4	3	2	2	2

3. La distancia de Levenshtein entre *gato* y *patio* es 2, el valor de la esquina inferior derecha.

Si se calcula la distancia de Levenshtein de una palabra mal escrita, por ejemplo, *perri*, con todo el vocabulario de nuestro corpus, las palabras *perro* y *perra* tendrían una distancia de 1. Podemos entonces mostrar al usuario documentos donde se encuentren estas palabras, lo que ayuda a compensar errores tipográficos y mejorar la efectividad de la búsqueda de información.

En La figura 5.2 se presenta un sistema de recuperación de información booleano que incluye todos los conceptos vistos hasta el momento. En este sistema, como primer paso se le aplica un preprocesamiento al corpus, posteriormente se aplica el proceso de indexación para crear el índice inverso. Mediante este índice inverso los módulos de expansión por sinónimos y búsquedas booleanas realizan la recuperación de documentos. Por otra parte, la consulta del usuario también es preprocesada y pasa por un modulo de identificación para saber si es una consulta . Posteriormente se verifica si es necesario aplicar recuperación tolerante a fallos, y se realiza una expansión por sinónimos para proporcionar mejores resultados. Finalmente, el sistema devuelve los documentos recuperados.

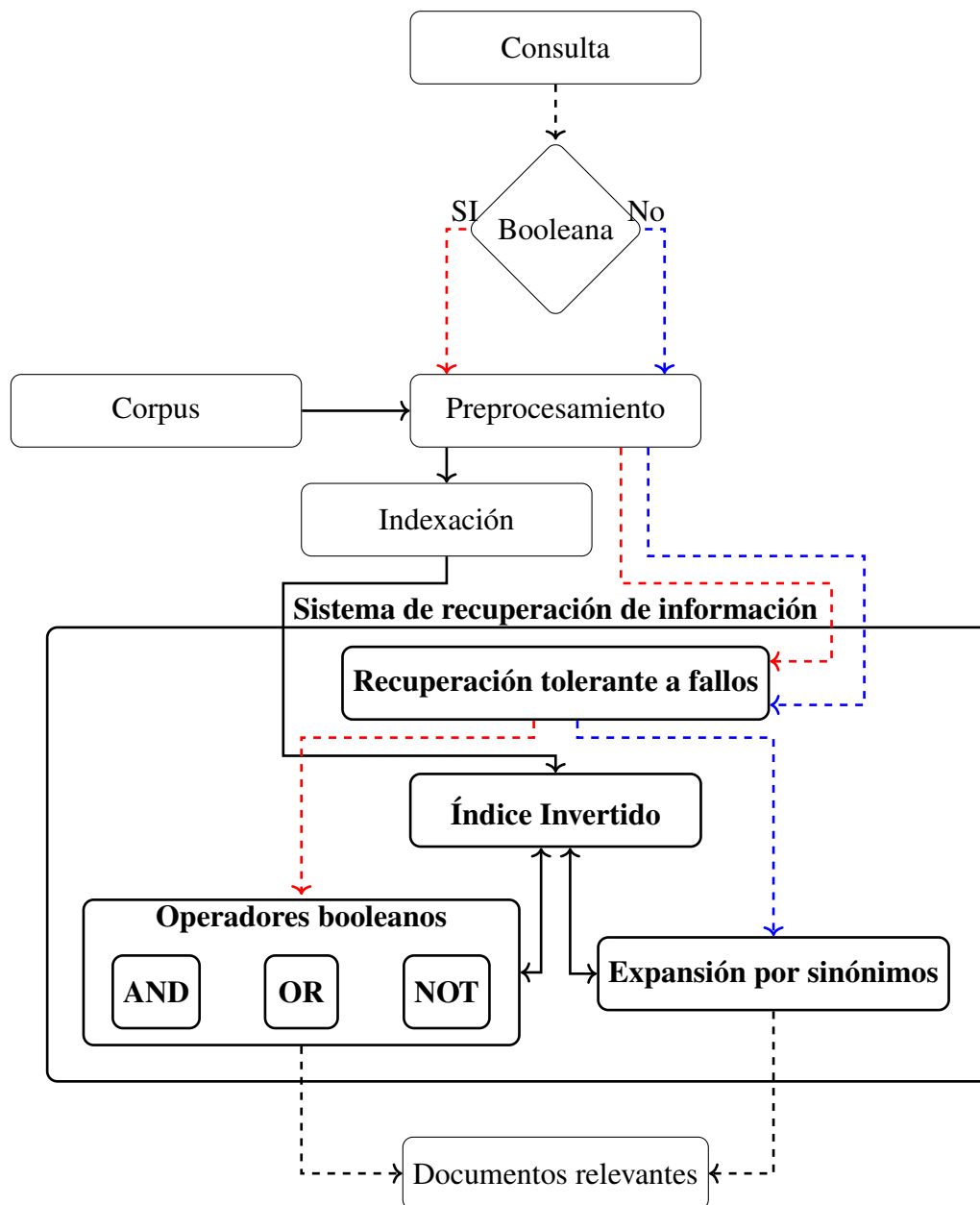


Figura 2.9: Sistema de recuperación de información booleano con expansión de consultas por sinónimos y corrección ortográfica.

Es importante notar que, al identificar una búsqueda booleana, el sistema no pasa por la expansión de sinónimos, ya que las búsquedas booleanas son especializadas y precisas. Generalmente, son realizadas por personas que están familiarizadas con el uso de operadores booleanos.

2.1.5. Modelo de espacio vectorial

El modelo de espacio vectorial (Vector Space Model, VSM por sus siglas en inglés) es un enfoque algebraico utilizado en los sistemas de recuperación de información para representar documentos y consultas como vectores en un espacio multidimensional. Este modelo es fundamental para la recuperación de información, ya que permite medir la similitud entre documentos y consultas de manera eficiente y efectiva. Toda la teoría presente en este apartado se toma del libro *Introduction to Retrieval Information* [Manning (2009)].

En el VSM, cada documento y consulta es representado por un vector de términos, donde cada dimensión corresponde a un término en el corpus. La ponderación de los términos en estos vectores puede ser determinada por diversas técnicas, como la frecuencia de término (TF), la frecuencia inversa de documento (IDF), o la combinación de ambas (TF-IDF).

La frecuencia de término $TF_{t,d}$ del término t en el documento d se define como el número de veces que t aparece en d . Definimos el IDF (frecuencia inversa de documentos) de t por

$$IDF_t = \log_{10} \left(\frac{N}{DF_t} \right) \quad (2.1)$$

Se usa $\log \left(\frac{N}{DF_t} \right)$ en lugar de $\frac{N}{DF_t}$ para amortiguar el efecto de IDF.

El peso TF-IDF de un término es el producto de su peso TF y su peso IDF.

$$W_{t,d} = (1 + \log TF_{t,d}) \times \log_{10} \left(\frac{N}{DF_t} \right) \quad (2.2)$$

Ahora se construye un espacio vectorial con las siguientes ideas:

- **Construcción de Vectores:** Representa cada documento como un vector en un espacio de dimensiones iguales al número de términos en el vocabulario V (todos los términos únicos en la colección de documentos). Cada dimensión corresponde a un término y su valor es el peso TF-IDF del término en ese documento.
- **Espacio Vectorial:** Los documentos se representan como vectores en este espacio de alta dimensión. La similitud entre documentos puede ser calculada usando medidas como la similitud del coseno.

Sin embargo, esto suele proporcionar vectores de alta dimensionalidad con muchas entradas

con valor cero. Otro punto a considerar es como relacionar las consultas con los documentos. Para ello, se considera a la consulta. Aunque se pueden utilizar medidas de distancia entre vectores, como la distancia euclidiana, esta medida no siempre es adecuada. Por ello, es común implementar la similitud del coseno como medida de similitud en lugar de la distancia euclidiana. Sin embargo, se ha mostrado que esta medida no es adecuada, por lo que es usual implementar la similitud del coseno como medida de similitud en lugar de la distancia euclidiana. La similitud del coseno entre dos documentos d_i y d_j se define como:

$$\cos(q, d) = \frac{\mathbf{q} \cdot \mathbf{d}}{\|\mathbf{q}\| \|\mathbf{d}\|} = \frac{\sum_{i=1}^{|V|} q_i d_i}{\sqrt{\sum_{i=1}^{|V|} q_i^2} \sqrt{\sum_{i=1}^{|V|} d_i^2}} \quad (2.3)$$

Donde:

- q_i es el peso TF-IDF del término i en la consulta.
- d_i es el peso TF-IDF del término i en el documento.

Para solucionar el problema de la dispersión entre vectores, es decir, la cantidad de ceros que existen en un vector, se implementa el algoritmo *CosineScore*.

CosineScore(q):

```

float Scores[N]=0
float Length[N]
for each termino t en la consulta q
do calcular el peso Wt,q y obtener posting list para t
    for each par (d,TFt,d) en el posting list
    do Scores[d]+= Wt,d X Wt,q
Leer el arreglo Length
for each d
do Scores[d] = Scores[d]/Length[d]
return Top K componentes de Scores

```

Este algoritmo hace uso de los posting list, para evitar cálculos innecesarios, de esta forma el algoritmo es rápido, eficaz y una herramienta útil para rankear documentos relevantes.

2.2. Recuperación de información multimedia

En la era digital actual, el volumen de datos multimedia, que incluye imágenes, audio y video, ha crecido exponencialmente. Recuperar información multimedia relevante para un usuario es sumamente complejo y se requiere de sistemas capaces de extraer características que puedan ser analizadas, comprendidas y manipuladas. Esta situación convierte a la recuperación de información multimedia en un área desafiante llena de oportunidades [Qiu (2022)].

2.2.1. Recuperación de información multimedia de imágenes

Los sistemas de recuperación de información multimedia (MSRI por sus siglas en inglés) están ligados fuertemente con el avance de las tecnologías. Existen diversas técnicas que se aplican a estos sistemas, por ejemplo, el método de bolsa de palabras visuales consiste en poder encontrar descriptores globales de las imágenes a partir de la creación de histogramas de alta dimensionalidad. Esta técnica facilita la creación de índices invertidos y espacios vectoriales para mediar su distancia con otras imágenes, pero suele ser menos precisa al realizar las búsquedas [Zhou et al. (2016)]. Se ha mostrado que los descriptores globales no capturan características relevantes y que los descriptores locales tienden a tener mejores resultados [Deselaers et al. (2008)]. En la década pasada, se introdujo el concepto de Vector de Descriptores Localmente Agregados (VLAD, por sus siglas en inglés). Este método extrae regiones utilizando detectores invariantes (algoritmo para extraer características de una imagen, robusto frente a transformaciones geométricas) para después caracterizarlos por descriptores SIFT (Scale-Invariant Feature Transform). Los resultados se agrupan en clusters, los cuales, para la recuperación de información se asocian a un vocabulario [Jégou et al. (2010)]. A estos tipos de sistemas se les conoce como Sistemas de Recuperación de Imágenes Basadas en Contenido (CBIR, por sus siglas en inglés) y son ocupados en distintas áreas, por ejemplo en aplicaciones médicas [Coelho and Ribeiro (2010)]. En los últimos años se han mejorado los métodos para obtener los descriptores de imágenes. Por ejemplo, Alsmadi en [Alsmadi (2020)] obtiene información sobre color, forma y textura, y combinando esta información presenta resultados con buena precisión. Algunos métodos adicionalmente crean

códigos Hash (cadenas de bits. Estos códigos se utilizan para indexar las imágenes y aplicar los algoritmos IR a partir de estos índices, como se describe en [Öztürk (2020)].

2.2.2. Sistemas de recuperación multimedia de imágenes basados en redes neuronales.

La propuesta de las redes neuronales como modelo de aprendizaje computacional se atribuye a Warren McCulloch y Walter Pitts en 1943 [McCulloch and Pitts (1943)]. Sin embargo, el concepto presentado por McCulloch y Pitts se ha ido refinando con el tiempo hasta llegar a las redes neuronales actuales. Un avance significativo en este desarrollo fue realizado por Frank Rosenblatt en 1958 con su trabajo sobre el perceptrón [Rosenblatt (1958)]. En su propuesta, el perceptrón consiste en una serie de unidades de procesamiento simples, denominadas neuronas, organizadas en capas. Los componentes de la red neuronal presentada por Rosenblatt son los siguientes:

- **Neurona:** Una neurona recibe varias entradas, aplica una ponderación a cada una de ellas, luego suma las entradas ponderadas y pasa el resultado a través de una función de activación para producir una salida. Matemáticamente, una neurona puede ser representada como:

$$y = f\left(\sum_{i=1}^n w_i x_i + b\right) \quad (2.4)$$

donde y es la salida de la neurona, x_i son las entradas, w_i son los pesos asociados a las entradas, b es el sesgo, y f es la función de activación.

- **Capas:** Las neuronas están organizadas en capas. Una red neuronal típica tiene una capa de entrada, una o más capas ocultas y una capa de salida. Cada capa oculta puede ser descrita como:

$$\mathbf{h} = f(\mathbf{W}\mathbf{x} + \mathbf{b}) \quad (2.5)$$

donde $\mathbf{h}$ es el vector de salidas de la capa oculta, $\mathbf{W}$ es la matriz de pesos, $\mathbf{x}$ es el vector de entradas, y $\mathbf{b}$ es el vector de sesgos.

- **Función de activación:** Las funciones de activación introducen no linealidades en el modelo, permitiendo que la red neuronal aprenda relaciones complejas. Ejemplos co-

munes incluyen la función sigmoide, ReLU (Rectified Linear Unit), y la función tangente hiperbólica.

En la Figura 2.10 se ilustra una red neuronal simple con una capa de entrada (conformado por las neuronas $I1$, $I2$, $I3$.), una capa oculta (conformada por $H1$, $H2$ y $H3$.) y una capa de salida con una sola neurona O . Cada neurona en cada capa está conectada a todas las neuronas de la capa siguiente, lo que es típico en una red neuronal completamente conectada.

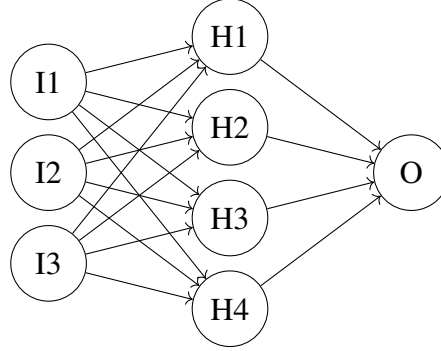


Figura 2.10: Esquema de una red neuronal con una capa de entrada, una capa oculta y una capa de salida.

No fue si no hasta 1980, cuando David E. Rumelhart *et al* propusieron la retropropagación (backpropagation, en inglés), en su trabajo proporcionan un algoritmo es clave para el entrenamiento de redes neuronales multicapa [Rumelhart et al. (1986)]. Su propósito principal es minimizar el error de predicción ajustando los pesos de la red de manera eficiente durante el aprendizaje supervisado y se sigue bajo los siguientes pasos:

1. Inicialización de los Parámetros:

- Los pesos $\mathbf{W}$ y los sesgos $\mathbf{b}$ de la red se inicializan aleatoriamente:

$$\mathbf{W}^{(l)} \sim \mathcal{N}(0, \sigma^2), \quad \mathbf{b}^{(l)} = 0 \quad (2.6)$$

donde l indica la capa de la red.

2. Propagación Hacia Adelante (Forward Propagation):

- Para una entrada $\mathbf{x}$, se calcula la salida de cada capa l aplicando una transformación lineal seguida de una función de activación σ :

$$\mathbf{z}^{(l)} = \mathbf{W}^{(l)} \mathbf{a}^{(l-1)} + \mathbf{b}^{(l)} \quad (2.7)$$

$$\mathbf{a}^{(l)} = \sigma(\mathbf{z}^{(l)}) \quad (2.8)$$

donde:

- $\mathbf{a}^{(0)} = \mathbf{x}$ es la entrada de la red.
 - $\mathbf{z}^{(l)}$ son las activaciones lineales de la capa l .
 - $\mathbf{a}^{(l)}$ son las activaciones de la capa l .
- La salida de la red $\hat{\mathbf{y}}$ se obtiene en la última capa L :

$$\hat{\mathbf{y}} = \mathbf{a}^{(L)} \quad (2.9)$$

3. Evaluación de la Función de Pérdida:

- La función de pérdida $\mathcal{L}$ mide la discrepancia entre la salida de la red $\hat{\mathbf{y}}$ y la etiqueta verdadera $\mathbf{y}$:

$$\mathcal{L}(\mathbf{y}, \hat{\mathbf{y}}) = -\frac{1}{N} \sum_{i=1}^N [y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i)] \quad (2.10)$$

para clasificación binaria (cross-entropy loss), o

$$\mathcal{L}(\mathbf{y}, \hat{\mathbf{y}}) = \frac{1}{2N} \sum_{i=1}^N \|\mathbf{y}_i - \hat{\mathbf{y}}_i\|^2 \quad (2.11)$$

para regresión (mean squared error).

4. Retropropagación del Error (Backpropagation):

- Se calcula el gradiente de la pérdida respecto a los parámetros de la red usando la regla de la cadena. Para cada capa l , los gradientes se calculan como:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{z}^{(L)}} = \hat{\mathbf{y}} - \mathbf{y} \quad (2.12)$$

$$\frac{\partial \mathcal{L}}{\partial \mathbf{a}^{(l)}} = \left(\mathbf{W}^{(l+1)} \right)^T \frac{\partial \mathcal{L}}{\partial \mathbf{z}^{(l+1)}} \quad (2.13)$$

$$\frac{\partial \mathcal{L}}{\partial \mathbf{z}^{(l)}} = \frac{\partial \mathcal{L}}{\partial \mathbf{a}^{(l)}} \odot \boldsymbol{\sigma}'(\mathbf{z}^{(l)}) \quad (2.14)$$

$$\frac{\partial \mathcal{L}}{\partial \mathbf{W}^{(l)}} = \frac{\partial \mathcal{L}}{\partial \mathbf{z}^{(l)}} \left(\mathbf{a}^{(l-1)} \right)^T \quad (2.15)$$

$$\frac{\partial \mathcal{L}}{\partial \mathbf{b}^{(l)}} = \frac{\partial \mathcal{L}}{\partial \mathbf{z}^{(l)}} \quad (2.16)$$

donde $\odot$ denota el producto elemento a elemento (Hadamard).

5. Actualización de los Parámetros:

- Los parámetros se actualizan usando un algoritmo de optimización, como el descenso de gradiente estocástico (SGD):

$$\mathbf{W}^{(l)} \leftarrow \mathbf{W}^{(l)} - \eta \frac{\partial \mathcal{L}}{\partial \mathbf{W}^{(l)}} \quad (2.17)$$

$$\mathbf{b}^{(l)} \leftarrow \mathbf{b}^{(l)} - \eta \frac{\partial \mathcal{L}}{\partial \mathbf{b}^{(l)}} \quad (2.18)$$

donde η es la tasa de aprendizaje.

6. Iteración:

- Los pasos de propagación hacia adelante, evaluación de la función de pérdida, retropropagación del error y actualización de los parámetros se repiten para cada mini-lote de datos durante múltiples épocas hasta que la función de pérdida converja o se alcance el número máximo de épocas.

El auge de la inteligencia artificial, en específico el uso de las redes neuronales convolucionales (CNN, por su acrónimo en inglés) ha mostrado que los descriptores pueden ser encontrados casi automáticamente [Yue-Hei Ng et al. (2015)]. Los códigos Hash son usados para simplificar y acelerar las búsquedas, actualmente se entrenan CNN para obtener estos códigos [Han et al. (2019)] y se empiezan a realizar estudios con otras arquitecturas como los transformers para generar estos códigos [Dubey et al. (2022)]. Cabe mencionar que el uso de CNN necesita de un dataset que empareje imágenes con texto, las CNN han mostrado poder

asociar imagen a texto y texto a imagen, esto lo hacen proyectando el texto y la imagen a un solo subespacio de características [Wu et al. (2021a); Gordo et al. (2017); Iscen et al. (2023)].

Existen diversos estudios enfocados en el área médica que usan CNN para crear un MSIR especializado. Shamna y Aziz entrenan un modelo de clasificación de resonancias magnéticas mediante CNN, y utilizando una comparación de similitud, recuperan documentos relacionados que ayudan al médico en el diagnóstico [Shamna and Aziz Musthafa (2023)]. En la misma línea, Zhang *et al* utilizan redes neuronales para clasificar resonancias y emparejarlas con texto relacionado con el diagnóstico médico. De esta manera, además de recuperar imágenes, el especialista también recibe posibles diagnósticos [Zhang et al. (2022)].

Capítulo 3

Estado del arte

Sólo hay un bien: el conocimiento. Sólo hay un mal: la ignorancia.

Sócrates

Existen diversos sistemas de recuperación de información multimedia. Por ejemplo, en el área de la moda, Whu y GAO presentan el primer conjunto de datos diseñado para apoyar al avance de sistemas de recuperación de imágenes en este campo [Wu et al. (2021b)]. A partir de esta base de datos, surgieron los primeros modelos de recuperación de imágenes que la utilizan como referencia [Dodds et al. (2022); Saito et al. (2023)].

Por otra parte, la geolocalización ha tomado gran relevancia desde la automatización de vehículos no tripulados hasta el uso turístico. Tang *et al* presentan un modelo de recuperación de información que no depende de un sistema satelital de navegación global. Utilizan CNN para predecir imágenes del entorno, RIM, identifican las posibles zonas geográficas en las que se encuentran [Tang et al. (2022)]. GeoWINE es un sistema de recuperación multimodal basado en geolocalización diseñado para apoyar a la verificación de hechos, detección de noticias falsas o verificación de imágenes [Tahmasebzadeh et al. (2021)].

Con el propósito de simplificar las tareas relacionadas con la videovigilancia, los sistemas de seguridad y monitoreo están experimentando una constante evolución. Esto se evidencia en el estudio exhaustivo realizado por Elharrouss [Elharrouss et al. (2021)], donde se analizan los principales componentes, arquitecturas y metodologías utilizadas en estos sistemas. Además, se lleva a cabo una comparación entre las funcionalidades de diversos sistemas.

Prathiba [Prathiba and Kumari (2021)] muestra un sistema de recuperación multimodal usando un prometedor algoritmo de clustering para la interacción humano-computadora. Este algoritmo extrae características de los fotogramas en el video, realiza una segmentación, aplica un método basado en vecinos cercanos y, en función de esto, calcula la distancia entre fotogramas para recuperar las imágenes más similares

Castañón [Castanon et al. (2015)] se enfoca en la recuperación de segmentos de videos en videos de vigilancia de larga duración. Extrae características de objetos y movimientos de rutas, almacenándolas en un árbol basado en tablas hash para realizar consultas de manera eficiente. También desarrolla un sistema de recuperación de actividades raras, anormales y recurrentes mediante programación dinámica.

Karbalaie [Karbalaie et al. (2022)] realiza una revisión de varios sistemas de detección de eventos en cámaras de seguridad. Destaca la importancia de una detección rápida y precisa, así como la implementanci3n de modelos de clasificaci3n efectivos para obtener resultados en sistemas autom3ticos de detecci3n de eventos. Tambi3n se3ala que la mayoría de los clasificadores suelen trabajar con videos cortos, lo cual no es práctico en situaciones del mundo real.

Murugesan [Murugesan and Thilagamani (2020)] emplea una red neuronal recurrente de percepci3n multicapa para la detecci3n de anomalías en sistemas de videovigilancia, sus resultados indican un alto desempe3o en las métricas de precisi3n, sensibilidad y especificidad.

Por otro lado, Ingle [Ingle and Kim (2022)] desarrolla un clasificador de escenas MSD-CNN (Multiclass Subclass Detection Convolutional Neural Network) que distingue entre cuadros normales, como caminar, y cuadros anormales, como escenas de personas con armas o cuchillos. Sus resultados muestran que, incluso con recursos limitados, se puede lograr una precisi3n del 85.5 % en la detecci3n de pistolas o cuchillos.

P3rez-Hernandez [P3rez-Hernández et al. (2020)] propone el uso de técnicas de binarizaci3n en redes neuronales convolucionales para mejorar la detecci3n de objetos peque3os que pueden ser manipulados con las manos y confundirse f3cilmente con una pistola o cuchillo. Su experimento demuestra que es posible reducir los falsos positivos hasta en un 56.50 %.

En la última década, You Only Look Once (YOLO) fue presentado por Redmon *et al.* en [Redmon et al. (2016)]. Esta arquitectura tiene la capacidad de predecir cajas delimitadas y la probabilidad de que estas cajas pertenezcan a ciertas clases. YOLO se ha utilizado

ampliamente en una variedad de tareas. Una [Uma et al. (2023)] realiza una revisión de las diversas arquitecturas YOLO y sus aplicaciones, resaltando su potencial para integrarse con la realidad aumentada.

Por otro lado, Abdusalomov [Abdusalomov et al. (2021)] mejora el rendimiento de YOLOv3 en la detección de incendios en videos de vigilancia mediante la creación de un dataset especializado y un ajuste fino. Su experimento demuestra que la red neuronal puede detectar con alta precisión las regiones de incendio en tiempo real.

Además de la identificación y segmentación de objetos, YOLO es una herramienta muy versátil que puede ser utilizada para la recuperación de información multimedia. Xin [Xin et al. (2023)] utiliza un dataset especializado para entrenar YOLOv5 y mediante las características obtenidas, puede recuperar imágenes similares a una imagen de entrada.

Nath [Nath and Behzadan (2019)] genera datos mediante minería web para entrenar YOLO y mejorar su precisión en diversos conjuntos de pruebas. Por su parte, Kumar utiliza las etiquetas proporcionadas por YOLO para recuperar y mostrar información de la web [Kumar et al. (2019)].

Tseng [Tseng et al. (2021)] propone un sistema de búsqueda y recuperación de personas en sistemas de videovigilancia multicámara. Utiliza el modelo YOLACT++ para segmentar imágenes y una red neuronal convolucional multicapa para extraer atributos de apariencia de las personas. Este sistema implementa un motor de coincidencias de atributos y genera resúmenes asociados a los elementos recuperados.

Una rápida revisión a la literatura pone en evidencia que las redes neuronales llevan la batuta en las tareas relacionadas a la recuperación de información multimedia [Yue-Hei Ng et al. (2015); Han et al. (2019); Dubey et al. (2022); Wu et al. (2021a); Gordo et al. (2017); Shamna and Aziz Musthafa (2023); Zhang et al. (2022); Wu et al. (2021b); Tang et al. (2022); Tahmasebzadeh et al. (2021); Prathiba and Kumari (2021); Iscen et al. (2023); Medina Cortes et al. (2021); Chambi et al. (2022); Butt et al. (2021)].

Artículos	Técnicas ocupadas						Año
	Descriptores	Código Hash	Pair Text-Image	Similitud	Indexación	CNN	
Zhou et al. (2016)	✓			✓			2016
Gordo et al. (2017)	✓					✓	2017
Shi et al. (2018)	✓	✓				✓	2018
Li et al. (2019)	✓				✓		2019
Han et al. (2019)	✓	✓		✓		✓	2019
Vo et al. (2019)	✓		✓			✓	2019
Luo et al. (2020)	✓	✓			✓	✓	2020
Öztürk (2020)	✓	✓		✓		✓	2020
Wang et al. (2020)	✓		✓	✓		✓	2020
Fang et al. (2021b)	✓	✓			✓	✓	2020
Shen et al. (2020)	✓	✓	✓		✓	✓	2020
Alsmadi (2020)	✓				✓		2020
Öztürk (2021)	✓	✓		✓		✓	2021
Wu et al. (2021a), Bain et al. (2021), Fang et al. (2021a)	✓		✓			✓	2021
Dubey et al. (2022)	✓		✓			✓	2022
Iscen et al. (2023), Tschannen et al. (2023)	✓		✓			✓	2023

Tabla 3.1: Revisión del estado del arte.

En la tabla 3.1 se describen los artículos presentes en el estado del arte, donde se identifican las técnicas utilizadas en los últimos años en los MSRI. Se han catalogado las técnicas en 6 grupos ordenados en la tabla de izquierda a derecha como sigue: aquellos que obtienen descriptores (Descriptores); los que generan códigos Hash (Código Hash); los modelos que utilizan conjuntos de datos en los que a cada imagen le corresponde un texto o palabras (Pair textImage); los sistemas que ocupan técnicas de similitud sin indexación (Similitud), es decir, los que transforman sus representaciones en vectores; los sistemas que indexan sus datos (Indexación) y a partir de ellos generan la similitud y los sistemas que se basan en redes neuronales (CNN). Se puede observar que los estudios recientes tienden a inclinarse por ocupar redes neuronales convolucionales, ya que estas redes se utilizan intrínsecamente para encontrar características en un conjunto de datos, trabajando así con descriptores. La gran mayoría de los artículos revisados omite la explicación de cómo obtienen el rank de los documentos, pero se infiere que al usar CNN utilizar CNN, es posible emplear los espacios vectoriales generados para calcular medidas de similitud.. No obstante, al no mencionarlo explícitamente, no se ha especificado qué técnica de similitud se emplea.

Aunque diversos autores utilizan los descriptores para indexar los datos multimedia, raramente se especifica en los artículos el método de indexar, además los descriptores transformados en códigos Hash o cadenas de bits, no suelen ser de gran utilidad cuando se requiere que un usuario no especializado genere consultas en lenguaje natural. Por otra parte, los sistemas que utilizan redes neuronales, son muy costosos computacionalmente, es por ello que en este trabajo de tesis se presentan dos sistemas. El primero genera cadenas de texto mediante un algoritmo de codificación imagen-texto, lo que permite realizar consultas en un lenguaje no necesariamente natural. El segundo sistema se destaca por ocupar una red neuronal convolucional para crear el corpus de texto, pero no como la base del sistema de recuperación multimedia, de esta forma se reduce en gran medida el uso de recursos computacionales.

Capítulo 4

Algoritmo de codificación imagen-texto

Ninguna cantidad de experimentación puede darme la razón; un solo experimento puede probar que estoy equivocado.

Albert Einstein

En esta sección se presenta un enfoque para la recuperación de escenas de interés en videos de larga duración mediante la codificación de imágenes a texto. A diferencia de los enfoques convencionales que suelen implicar el uso de redes neuronales, este método propone una técnica que evita el uso de estas estructuras complejas con el objetivo de reducir el consumo de recursos computacionales. A través de experimentos, se demuestra la viabilidad y efectividad de esta técnica, concluyendo que es factible emplearla para la recuperación de información multimedia, ofreciendo una alternativa eficiente y económica para esta tarea.

4.1. Metodología

La metodología propuesta en este documento se presenta en la Figura 4.1, esta metodología consta de dos componentes principales: primero, se implementa un módulo para generar el corpus de texto, este proceso se lleva a cabo mediante un algoritmo de conversión de imagen a texto; segundo, se implementa un sistema de recuperación de información de texto que permite realizar consultas y obtener documentos relevantes asociados a la consulta. Es importante mencionar que debido a las limitaciones del tiempo, las consultas que se realizan en

este sistema no se realizan en lenguaje natural.

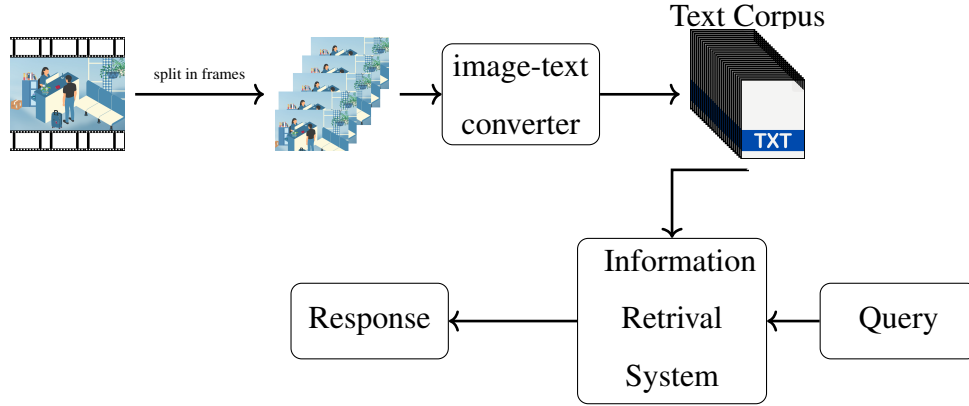


Figura 4.1: Metodología propuesta para la identificación de escenas de interés en videos de larga duración mediante un módulo de codificación de imagen a texto.

El flujo de la metodología se desarrolla de la siguiente manera: primero se obtienen imágenes de un video de larga duración, cada imagen es procesada por el algoritmo de conversión de imagen a texto propuesto. A partir de las representaciones textuales se crea un corpus de texto que alimenta a un sistema de recuperación de información de texto, a este sistema se le puede realizar una consulta en lenguaje no natural, recuperando así una serie de documentos asociados a la consulta.

4.2. Algoritmo de conversión de imagen a texto

Para la conversión de una imagen a una representación textual, se tomó como base el algoritmo que presenta Luo [et al Luo et al. (2020)], este algoritmo es usado para encriptar información en imágenes, sin embargo, mediante una modificación, utilizamos este algoritmo para generar representaciones textuales. Para esto se ejecutan los siguientes pasos:

1. Del vídeo se extraen imágenes cada determinado tiempo t , cada imagen es nombrada como I_t .
2. Cada I_t se divide en $m \times n$ bloques, es decir, cada imagen será dividido en una matriz de subimágenes, de esta forma la imagen se describe como un conjunto de bloques, $I_t = \{B_{1,1}, B_{1,2}, \dots, B_{m,n}\}$.

3. A cada elemento $B_{i,j} \in I_t$ se le asigna una cadena de caracteres α mediante un mapeo f , de modo que $f(B_{i,j}) = \alpha$.
4. Las cadenas resultantes formarán un documento de texto con n filas y m cadenas en cada fila.
5. Los documentos de texto conformarán un corpus que alimentará a un sistema de recuperación de información.

Es importante destacar que la parte fundamental del algoritmo propuesto depende en gran medida del mapeo f , en este capítulo se presentan dos mapeos: en primer lugar, se obtienen cadenas de caracteres mediante el promedio de grises de los elementos $B_{i,j}$ presentes en una imagen; en segundo lugar, se transforman los segmentos $B_{i,j}$ de una imagen en código Hash. Debido a la naturaleza de estos mapeos, no es posible utilizar el lenguaje natural para realizar consultas.

Para los experimentos presentados en este documento, se descargó de internet un vídeo de 5 horas de duración que contiene grabaciones de una oficina de impuestos en Estados Unidos [[Corners-News \(2023\)](#)].

4.3. Primer experimento: Conversión a caracteres mediante el promedio de grises

Siguiendo el algoritmo para obtener representaciones textuales presentada en la sección anterior, se lleva a cabo el siguiente proceso para generar un corpus de texto:

1. Se obtiene un frame en escala de grises cada 1.5 segundos.
2. Se obtiene una partición de cada frame de la siguiente forma:
 - a) Cada imagen, I_t , es dividida en N filas y M Columnas, formando una matriz de subimágenes, $I_t = \{B_{1,1}, B_{1,2}, \dots, B_{m,n}\}$.
 - b) Cada subimagen, $B_{i,j}$, es dividida en P elementos, así $B_{i,j} = \{b_1, b_2, \dots, b_p\}$.
3. Para obtener una representación textual, es necesario aplicar un mapeo f a cada subimagen $B_{i,j}$. Para esto se siguen los siguientes pasos:

- a) A cada elemento $b \in B_{i,j}$ se le aplica la función l para obtener una letra del alfabeto inglés.

Dado que el elemento b está compuesto por valores entre 0 y 255 (valores de los pixeles en escalas de grises), se define la función l como sigue.

$$l(b) = letra \left(\left[\frac{\sum_{k \in b} k}{|b|} \right] \bmod 26 \right)$$

donde k es un pixel de b , $|b|$ es el número de pixeles que conforman a b y $[\cdot]$ es la función parte entera. La función letra se define como sigue:

$$letra(x) = \begin{cases} a & \text{si } x = 0 \\ b & \text{si } x = 1 \\ \vdots & \\ z & \text{si } x = 25 \end{cases}$$

- b) El mapeo f se define de la siguiente manera::

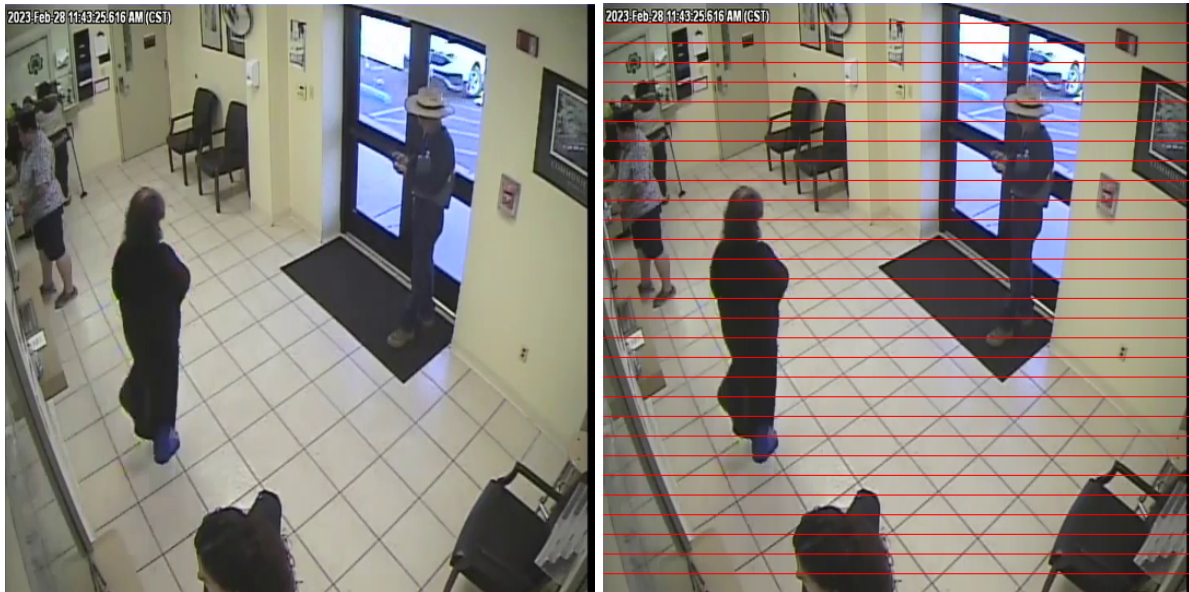
$$f(B_{i,j}) = \bigcup_{b \in B_{i,j}} l(b)$$

donde $\bigcup$ denota la concatenación entre caracteres.

De esta forma se ha obtenido una cadena de caracteres por cada subimagen.

4. Por cada imagen I_t , se genera un documento de texto siguiendo estos pasos:
 - a) Se unen todas las cadenas obtenidas al aplicar el mapeo f a las subimágenes que pertenecen a la misma fila, separándolas con un espacio en blanco.
 - b) Las filas resultantes del paso anterior se unirán mediante un salto de línea, creando así un documento de texto con M filas y N cadenas de caracteres en cada fila.
5. Finalmente, cada documento obtenido conformará parte de un corpus de texto, este

corpus servirá como base para el sistema de recuperación de información de texto.



(a) Frame 13 del video.

(b) División del frame 13 por 30 líneas.



(c) División del frame en 10 columnas.

(d) Cada elemento obtenido se divide en 9 secciones.

Figura 4.2: Segmentación del frame 13 obtenido del video.

En este experimento, se utilizan los siguientes parámetros: $N = 30$, $M = 10$, y $P = 9$. En la Figura 4.2, se ilustra el proceso para particionar el frame 13 (Figura 4.2a) obtenido del vídeo. Para ello, el frame se divide en 30 filas (Figura 4.2b) y 10 columnas (Figura 4.2c). Posteriormente, cada subimagen se subdivide en 9 elementos (Figura 4.2d). Esta división

puede interpretarse como un documento de texto que consta de 30 líneas, donde cada línea contiene 10 “palabrascompuestas por 9 “letras”.

Una vez finalizado el proceso de división, se aplica el mapeo f descrito en esta sección, generando así una representación textual para cada subimagen y un documento de texto al unir todas las representaciones siguiendo el paso 4 presentado en este apartado. El documento generado al aplicar este proceso se muestra en la Figura 4.3.

```

ikkjklillm kmjllmlll jmojklknn opnmkklil nkprpppq qrrqqpjfg lnnmlkmnn oppppppoo oooonnnmm mllkkkia
ijklllllno nkhhpppmm hlmkjlmooppmmknjl nrrsppppo mrrrsrkfg syyywsh ggnmkjkmn ppooooiik lmmlllja
jkkklmnq oiedpnkmm ikmkkmmooppqoopkk mssppppp qsssrkfg rvvxyxqh gjpsswxur ooppoljk knnmmlja
klmljlmml mkefnool lmmmmmlloo pppqqqqqr rrsqqqr qsssrkef pssrrrlg gkompqsws tqppppoo oghiklma
klmkleefk kkgfkppok lmmmmmmop ppijnqqq qrrsqqqr prsssrkef qrrrrrli lmmmmmmnr rqqqqpppp mddiihgga
jffefggjk kknoppppm mmmmmmmmn nkeeehnff gsssqqr sssrqjde rssrskk jllqrsrr qqqqqqpp jefikjja
fdeegkkii hlopppppm mmmmmmlhk igddfihee erssqqr sssrpohee egjlnpshf fgglrrrr rrrqqqqq gfihikiga
dcegikmhd imooppppn lmmmmmmhi giffgfee eqrsrqrr rssrqphdg spkhggff ggffkrrr rrrrrrqo fiiihhfa
hghjhjlid hnmoosooo nooooooim hkghfggi gorsrqrr rssrqhdh uuuuuplff fffffinpr rrrqqqqn ifgggiha
giigfhjll hmkmoooop pnkkmppp nolmlhkjk jmrqqqr rrrsqfdj sttttnff fffgheem rrroknqq qqljhgfa
jigffghgg knoopppq meeeefnqq pppplnnn noqrrppq qrrrqofdg msssskef eeffottq rrrollqq qqqqppnka
jiheecddf nooppqqq hdddiqqq qrrqqqqq qqqqqqqq pqqonlge edeilogee eejssssr rrrrrrrr qqqqppna
ikjkkfdde moppqqqph ddddegka rrrrrrrr rrrrrrrr pmkhfeffh iihfddde eemrrssr rrrrrrrr qqqqppna
jklmlkhjl joppqrrid dddddddee iqsrrssr rrrrrnhf ffffffff fffhiigfe eegknrrr rrrrrrrr qqqqppna
nillligjl hkqqqrkd ddddddde gssssssss ssssssssp lhffffff fffffeghe eedddmq rrrrrrrr qqqqppna
llhkmllk mprrrqrrl pidddddl rssssssss sssssssss sqmiffff fffffee eegjihqr rrrrrrrr qqqqppna
hmjiknmno opprrsrq fddddddei sssssssss ssssstss srrrrnjf ffffffee efelqqq rrrrrrrr qqppppma
gimopommo oopqrrrq ofddddd k ssssssrst tttttrss sssssrrr njffffee eefilpqq qqrrqqpo qqppppma
iginmklkl opopqqqq pidddddl rssssssss sssssssss rrrkgffg kossrppp qqqqqqqq qqpppoma
jigiokjik mooopqqn fdddddde srrssssss sssssssss srrssss srrqropr srrqppp pppppqqq ppppooa
kihginppo ppppoopp iedddddd sssrrssss sssrrssss sssrrrrr rsrrrqqr qqqqqqpp pppppppp pppoonla
lkjginpp poooooopp pnddddeg srrssrrs rrrssrrs sssrrrrr rsrrrrrq qrrrqqqp ooppoooo oooooomia
klllkhio pppppooo onliefgl qrrrrrrr sssssssr qrrssssr rrrrrrrr rrrqqqqp qppppopp oonnmllka
klkklhln ppppoooo nmnmkkon oqrrssss sssrrrrs srrrrssss rrrrrrrr rrrqqqqq qppppkijm onnmjka
jkkklhln mpooppo nnoonmno poprrssss srqnprss sssrrrrr sssrrqqq rqqqqqqp plhfglonk iiiihgga
kkkklmli gkppppoo nmnnoooo pppppnnm ngffqss sssrrrrr sssrrqqq qqqqqqql gfeeeegil lkiefiha
lkkklmml jgoonno ooonnoooo oolfeeee eefffpqr rssssssr rrrrrrrr rrrppqohf eeeeeedd efgkllja
llkjjklkl lgghlmmn nooonnnn oofeeddd eeeefhpr qssssssr rqqqrrr rqqqmhfe eeeeeedd eillkkja
llkjjkk kkkjhlmm mnnnnnoo oplfddddd ddddefiq rrrrrrrr rrrrrqqq qpppofee eeeeeedd iihghija
kllkjjk kkkkihlm mnnnooon opoiffedd ddddddefm qrrqqqr rrrrrrrq ppppjknig edddddd degkhiffa

```

Figura 4.3: Representación textual generado del frame 13 al implementar el mapeo de conversión a caracteres mediante el promedio de grises.

Es importante recopilar datos con respecto al uso de recursos computacionales. De este experimento se puede destacar lo siguiente:

- Se obtuvieron 1,237 archivos de texto.
- El proceso de obtención del corpus tardó aproximadamente 15 minutos 40 segundos, es decir, procesar cada imagen tarda 0.75 segundos.
- El peso total de los archivos de texto es de aproximadamente 3.7 MB.

Esto señala que el algoritmo procesa el video rápidamente, además, genera un corpus de tamaño adecuado para aplicar técnicas de recuperación de información de manera eficiente, ya que indexar 3.7 MB es una tarea fácil para cualquier equipo de cómputo.

Después de fijar el frame 13 como escena de interés. Se realizó una búsqueda manual en los frames cercanos, se obtuvo que las escenas presentes desde el frame 11 hasta el frame 20 son muy similares, al observar la Figura 4.4 se infiere que el único cambio significativo que sucede desde el frame 11 hasta el 20 es el movimiento de una persona.



(a) Frame 11 del video.

(b) Frame 20 del video.

Figura 4.4: Frames cercanos al frame 13

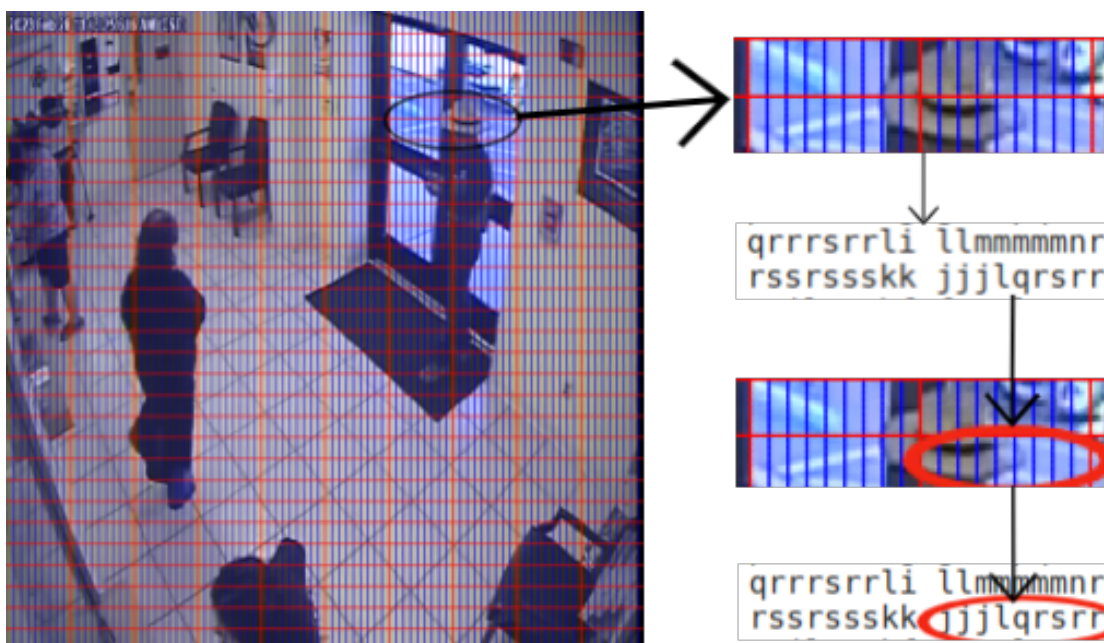


Figura 4.5: Aplicación del mapeo basado en promedio de grises sobre la franja del sombrero

De todos los objetos presentes en el frame 13, se realiza un enfoque en el sombrero de la persona cercana a la puerta, específicamente en la sección donde se encuentra la franja del sombrero. La Figura 4.5 muestra este enfoque, y la cadena asociada a este elemento es **jjjlqrsrr**.

Al realizar una búsqueda en el sistema de recuperación de información, se encuentra que el único archivo que contiene la cadena **jjjlqrsrr** es el frame 13 y esta cadena se encuentra en la línea 5 y columna 7. Al revisar los frames cercanos manualmente, se observa que esta franja aparece en la misma posición en los frames del 11 al 20, las cadenas en esta posición de los frames 11 y 20 son:

- Frame 11 -> **jjjlqrsr**.
- Frame 20 -> **jkkkoqsr**.

Se observa que la única diferencia entre la cadena obtenida del frame 11 es la letra s (resaltada en color rojo) mientras que en el frame 20 existe mayor diferencia.

Archivo	Posición	Distancia
frame_13.txt	(5, 7)	0
frame_11.txt	(5, 7)	1
frame_12.txt	(5, 7)	
frame_967.txt	(17, 1)	3
frame_1106.txt	(20, 0)	
frame_1114.txt	(20, 0)	
frame_1113.txt	(20, 0)	
frame_1109.txt	(20, 0)	
frame_1111.txt	(20, 0)	
frame_1115.txt	(20, 0)	
frame_1110.txt	(20, 0)	
frame_1112.txt	(20, 0)	
frame_1107.txt	(20, 0)	
frame_1108.txt	(20, 0)	
frame_501.txt	(18, 4)	
frame_14.txt	(5, 7)	
frame_862.txt	(19, 4)	
frame_500.txt	(20, 4)	
frame_503.txt	(25, 4)	

Tabla 4.1: Cadenas con una distancia de Levenshtein menor o igual a 3.

Si se considera que la cadena **jjjlqrsrr** obtenida del frame 13 es una palabra y que esta

palabra está escrita correctamente, correcta, y que las cadenas obtenidas en los frames 11 y 20 son versiones incorrectas de esta palabra, cuya corrección debería ser la cadena **jjjlqrsrr**, entonces podríamos aplicar un algoritmo presente en la recuperación de información tolerante a fallos, este algoritmo calcula la distancia de Levenshtein [Levenshtein et al. \(1966\)](#). Por ejemplo, la distancia de Levenshtein entre las cadenas **jjjlqrsr** y **jjjlqrsrr** es de 1, ya que solo se requiere cambiar de la primera cadena la letra s (resaltada en color rojo) por la r para obtener la segunda cadena.

Se obtuvieron todos los documentos que contienen tokens con una distancia de Levenshtein menor o igual a 3 en comparación con el token **jjjlqrsrr**, recuperando 19 tokens. En la Tabla 4.1 se presentan estos resultados. De los 19 resultados solo 4 documentos se encuentran en el rango buscado.

4.4. Segundo experimento: Conversión a cadenas de códigos Hash

Nuevamente, de acuerdo con los pasos generales para la obtención de representaciones textuales, se presenta un segundo experimento que contempla los pasos descritos a continuación:

1. Se obtiene un frame del video cada 1.5 segundos.
2. Se obtiene una partición de cada frame de la siguiente forma: Cada imagen, I_t , es dividida en N filas y M Columnas, formando una matriz de subimágenes, de esta forma la imagen I_t queda expresada como:

$$I_t = \{B_{1,1}, B_{1,2}, \dots, B_{n,m}\}.$$

3. A cada subimagen, $B_{i,j}$, se le aplica un mapeo f el cual retorna un código Hash, es decir,

$$f(B_{i,j}) = Hash(B_{i,j})$$

4. Por cada imagen I_t , se genera un documento de texto mediante los siguientes pasos:

- a) Se unen todas las cadenas obtenidas al aplicar el mapeo f a las subimágenes que pertenecen a la misma fila, separándolas con un espacio en blanco.
 - b) Las filas resultantes del paso anterior se unirán mediante un salto de línea, creando así un documento de texto con M filas y N cadenas de caracteres en cada fila.
5. Finalmente, cada documento obtenido conformará parte de un corpus de texto, este corpus servirá como base para el sistema de recuperación de información de texto.

En la figura 4.6 se muestra la partición del frame 13 en 30 filas y 10 columnas.



Figura 4.6: Segmentación del frame 13 al aplicar el algoritmo basado en códigos Hash.

En la figura 4.7 se muestra el documento de texto asociado al frame 13.

```
070f00f3554d7f00 fcf001cf27f300 0363016d7b6ff703 e1e0e0e4e4e4e4 3f3f3f3f3f3f3c3c f8f8f8f8f8f8f8f7c ff7f1f0f07c3e0f0 f0f0f0f8f8f8f8f8 fcfcfcfcfcfcfcf
0f0f1f1f3b33347 fefcf8f838f8f8f 677f7f7f37274747 e5e1e1f1f8f8f9fc 343434323272673 78787878f8f8f8f8 787c7e7e7e7e7e 3f1f0f0703637170 ffff9f8f8f8f8f8 fcfcfcfcfcfcf7c7c
0f0f1f1f27272727 0e8f8f8f8989898b 07473f3f3f3f3b3b c0e0f0cfcfcfcfcf 737272707270f0f f8f8f8f8f8f8f8f8 fefefef7c3c1c0c 3c3c3e3f17070f07 7838383cfefeffff 7c7c7cfcfcfcfcf
2726667777777d 0e8f8f8f8f8f898b 23030303c3434343 1f1f1f3f1f1f1f2f 12f1f1f1f13133f3 f8f8f8f8f8f8f8f8 ccf4f0fcfcfc7c7c 0f7f3f1f03230301 c0c0c0a0a0e0e0f0 fefefe9e9e8e86b6
f9f1f1f1f1f2c0c0 ae8e8e8e8f0f0f1f 0303030303030303 ffffcfcf7c78786 f39393b372737373 f8f8f8f8f8f8f8f8 dcdccfc7c3cdcd c0b0f070301b09cdf a0a0206f070f8f8f8 82b298989c9c9e9e
e1a181838f0f0f07 0e1e1e3e3e3e3e3e e3e3e3e3c3c2c0c4 c0c0c0c08404d4c4 7373737377777373 f8f8f8f8f8f8f8f8 ece4ecf74e4fcfc 7f0f0f1f1f0f1f1f 6070f0f0f8f8f8f8 9e9e9e9e8e8e8e
0f0f070e0e1c1c0d 3e3e3e3e3e3e3e3f fdfdfdfdfcfcf8f8f ccc8c00040c0c0c 7f7f7f7f7f7f7f7f f8f8f8f8f8f8f8f8 7e7e3c1c0c0c0604 1f0f0f0f0f0f0707 f0f0f0f8f8f8f8f8 bebe3e3e2c3c3e3c
0e0e1e0e1e1e1e3e 3f3f3f3f3f3f3e3e f8f0f4ecdcdbd7d7d 0c0c1830e0c0c0c0 7f7f7f7f7f7f7f7f f8f8f8f8f8f8f8f8 0808c0c0e0e4f0f0 0707070707070707 f8f8f8f8f8f8f8f8 3e7e7e7268341058
3e3e3e3e2626a2e 3f3f3f3f3f3f3f3f 1c3c64fcfcfcf7f7f c0c04040c1e7f7f7 7f7f7f3f3f3f3f3f3f f8f8f8f8f8f8f8f8 f8f8f8f8f8f8f8f8 fefefefefefef7e3 682c34361a2aac9c
0e2307c7c3436767 1f1f1f1f0f0f1f1f ffffcfcf8f878783 5452d0d0d0c0f1f3 3f3f3f3f3f3f3f3f f8f8f8f8f8f8f8f8 fcfcfcfcfcfcfcfcf 090909191191392d e3e3e3e3e3e3e3 c0c0e0e0f0f0f8f8
c7c7cc6c480c0c0 0f0f0f0f0f0f1f3f 8787878787878787 f0f0f0f0f0f0f1f 3f3f3b393939701 f8f8f8f8f8f8f8f8 fcfcf8f8f8f8f8f8 0d0d0f0f1f0f0f1f e3e3e3e3e3e3e37f f8f8f8f8f8f8f8f8
f0f1e1e0e0c0d0d 17171711f1f1f1f 0707070707070707 c0e4e4e7f2f2f7a7c 0b09099c2e6e4f5 f8f8f8f8f8f8f8f8 3c3c1c1c0c0c0000 0f0f1f1f1f1f1f1f e0f0f8f8f8f8f8f8 fcfcfcfcfcfcfcf
90d0f0f0f8f8f8f8 3f3f3e3e3e3e3e3c 0707030303010101 607830b2a6c9f9f9 f0f3f3ebc95c3d3e f3f1e1c0c0808000 0808c0c0c0e06070 1f1f3f3f3f3f3f3f f8f87878f8f8f8f8 fcfcfcfcfcfcfcfcf
78787878a3ab3bb0 7e7c7c7c7c7c7c7c 0707030303030101 f7f7f7f7f7f3f373f fefcf8f8f8f8f8f8f8 78f8f8f8f8f8f8f8f8 303018181c0c0e06 3f3f1f1f0f0f0707 f8f8f8f8f8f8f8f8 fcfcfcfcfcfcfcfcf
bb9b9b1b1b9baba 7c7c7c3c3c3e3e3e 03030303070f0f0f 3f3f3f3f7f7f7e7e ecd696b33b7bdfdc 000000008080c0c0 0606030300000000 0703030703070707 f8f8f8f8f8f8f8f8 fcfcfcfcfcfcfcfcf
a888cccccfcfcf4f 3e3e3e3a3e3e3e3f 0001010101818181 7c393bb3a78f9f9f fcfcf87a73a7a78f c0c0e0e0e0f0f0f0 fefefefefcf8f8f8 2727171717171f0f07 fcfcfcfcfcfcfcfcf
474747474344f1f 1f1f1f1f1d1d1d19 0101010181818181 8f2f2f6f67e6f4f4 9f9f9f0f6f6f6f7f f8f8f8f8f8f8f8f8 fcfcf8f8f8f8f8f8 a0a0a0a0c4ccdf0f 0f07070707070707 fcfcfcfcfcfcfcfcfcf
1f3f3f3b33313931 1f1f2f0e1e1f1f1f 818181818181c1c1 f19f3f3e3e9c9dd f6f0f8f9f9f8f1f5 e2f3f3f2f8f8f8f9 0c0c8c8c0c0c0e0e 07070f0f1f1f1f3f ffffdfdf8f8f8f8f8 fcfcfcfcfcfcfcf
3f3f3f3f17131210 1f1f1f1f1d0b0b03 c1c1c181818181 9c9c3e7e7e7b7b7e e4c4c4e9ebc3e7e7e f8f2f2f667674f1f f0e0e0f0f0f0f8f9 3a3b7b797cfcfcf8 197c7e3e3e8e9e1e fcfcfcfcfcfcfcfcf
181818181808089 6f6f4f4f0e0e2e26 8181010101010101 3ebc9cb99b9397c7 3f3f3b7f9deddccc 9f9f3f1f5f5cfcf7f f9e9cf9cfdfcfcfaf7 b0f0f0f0e0e0e0e0 5f4f6f2727131303 fcfcfcfcfcfcfcfcf
0b0f0f0f8f8f8787 66e6e6f7f5f1f3 0101010181818181 cfcfcf8fa7276767 cde9e3e3e7e3c3c 7e7f7f7f2f2f8f8f8f9 a47072f2f8f8f8f8f8 e2d2d2dab8b83878 ebe3edcd4c2d298 fcfcfcfcfcfcfcfcf
07076747e7a3a323 3f3e3e3c3bd99939 81c1c1c1c1c1c1c1 f7f7f3f39b190818 d39b9939397d7cfc f9f9f0f4e4e4e4e4 f4fcf8f8f8f8f8f8f8 70f0f8f8f8787838ba bcb3c3c7c7c3c3c fcfcfcfcfcfcfcfcf
6373737373535151 3979f9f8f8f8f8f8 e1f1f0f0f0f1f1f1 7d3d393a3636262f fefefefef7f7be7e de9e3e3f7f7f3f3f 7b73b3b78787cfcf be9cdcdcc9e9e2 9cdccce8e8e0f0 fcfcfcfcfcfcfcfcf
f9f9f9f95959587c fcfcf8f8f2f6e6ee f1f1d39b1f17f7f 0f1f1f1f3f3f3f3f c7c7c7f9b0b3b7a7 3f9f9f1f4fcfcfee c18f8f8f27777373 12f2f2f0e0e0e0c ffff9f9f9f8f8f8f8 fefcf8f6f67e2e0b
7c7c7c7c7c7c7e7e cede5f7f7f7f7e7e 3475f1f9b3b3a7a7 3f1f1f1f1f0f1f0f efcdfcf79fa4f4f7 eee6e4f5f1f1f3f3 f8f9f9f8f8f8f8f8f8 ccdcd9ebcb3878 e4c6c6c8f8f8f878 f78787832b296c6
0e0e1e1e1e3e3f3f 7f7f3f3f3f3f3f3e 8cf9f9f9f1f1f4f4f ffffff7f8f8f8f8f8 c7c7c783838303 1f1f5ecccccd9e fcfcfcfcfcfcfcf8f9 7f7f7f7e7e7e7e7e 8787830301010000 c4e4e4e4e4e4e4e4
0f07078787878787 3f3f3f3f1b171f1f c3c3e3e3e7e7e7e7 e0e0e0e0c0c0c0c 0707070707070707 9e3e3e7e7e7e7e7e 797b733337a7a78f fefcfcfcfcfcfcfcf8f8 f0f0f0f0f0f0f0f0 c4e4e1e3e3e3e3e
c7c7c3c3c3c3c0c 9f9f9f8f8f8f8f8f8 e7f7f2f2f475717b c0c0c0c0c0c0c0c 0707030303030303 7e7f7f7f3f3f3f3f cfcfcf8f8f8f8f8f8f8 f8f8f8f8f8f8f8f8 f0f0f0f0f0f0f0f0 3e3e3e7e7e7e7efa
e0e0e1e1f1e1e1e 8f0f0f2767676727 737372f0d1d1d3f c0c0e0e0e0e0e0e 0303030303030301 bf9f9e9ecccc9c3 7773f3f3b9f8f8f8f8 f0f0f0f0f0f0f0f0f0 f0f0f1d08e08e060 fefefe86840c2c2c
e1e1e0e0f0f0f0f0 e7e7e3e3e36361e 3d3f1e1e1e1f1f1f e0c0e0f0f0e0e0e0e 0303030301010101 e9e36377f7e3e3eb fcfcf8f8f8f8f8f8f8 f0f4f4f4f4e4e4e4 7c7e3a18080801c0 3c3c3c3c7878583a
```

Figura 4.7: Texto asociado al frame 13 al aplicar el mapeo basado en códigos Hash.

Se recopilaron los siguientes datos sobre el uso de recursos computacionales:

- Se generaron 1,237 archivos de texto.
- El proceso tuvo una duración aproximada de 14 minutos 32 segundos, aproximadamente 0.70 segundos por frame.
- El tamaño total de los archivos de texto es de aproximadamente 6.3MB.

Lo que indica (nuevamente) que el tiempo de procesamiento es pequeño y que es viable aplicar técnicas de recuperación de información al texto generado, ya que indexar un corpus de texto de 6.3 MB es factible para casi cualquier computador actual.

Nuevamente se analizará la sección del frame 13 donde se encuentra una persona con sombrero, en particular se enfocará en el código Hash que representa la franja del sombrero, en la Figura 4.8 se puede apreciar este segmento de imagen a analizar.

Al realizar una búsqueda de la cadena **7f0f0f1fc70f1f1f** en el sistema de recuperación de información, se identificaron dos documentos: el frame 13 y el frame 12. Similar al experimento anterior, la franja del sombrero aparece en esa posición desde el frame 11 hasta el frame 20, los códigos Hash correspondientes a esta franja en los frames son los siguientes:

- Frame 11 -> **5f0f0f1fc70f1f1f**.
- Frame 20 -> **7e0f0f0fc7670f1f**.

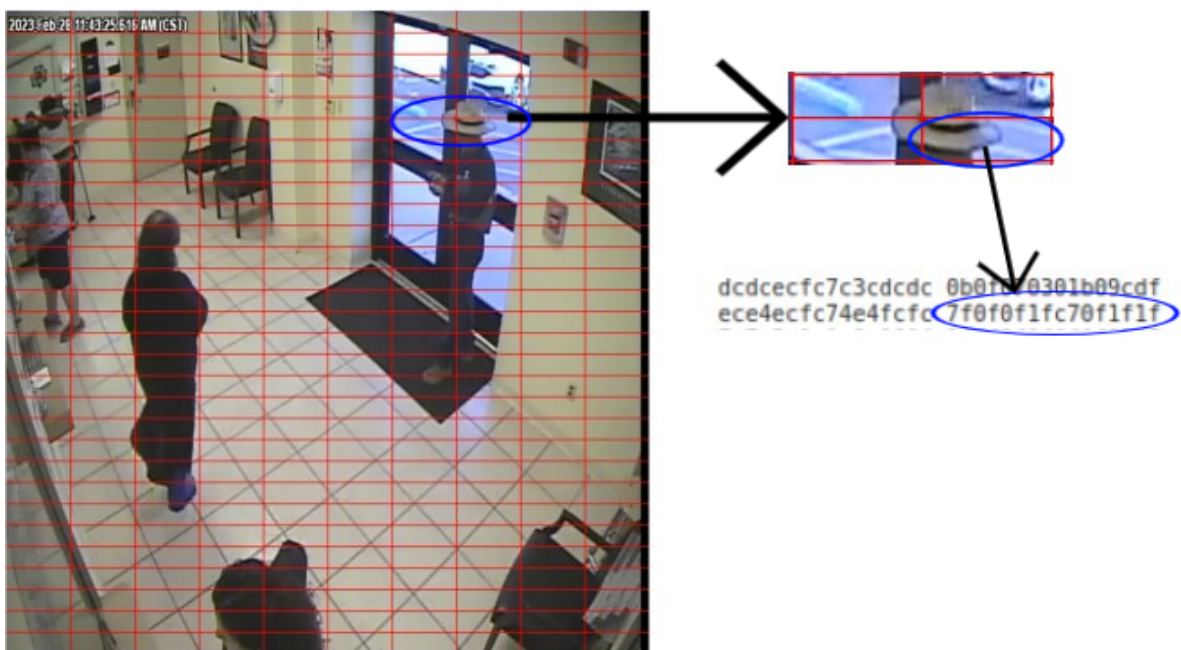


Figura 4.8: Analizando frame 13 con el código Hash.

Si se considera que las cadenas presentes en el frame 11 y 20 están mal escritas, es posible aplicar el algoritmo de Levenshtein. Al buscar tokens con una distancia menor o igual a 3 al token **7f0f0f1fc70f1f1f** se obtienen 11 resultados los cuales se presentan en la Tabla 4.2. De estos 11 resultados, 9 se encuentran dentro del rango previamente identificado manualmente.

Archivo	Posición	Distancia
frame_13.txt	(5, 7)	0
frame_12.txt	(5, 7)	
frame_14.txt	(5, 7)	1
frame_11.txt	(5, 7)	
frame_15.txt	(5, 7)	2
frame_16.txt	(5, 7)	
frame_1176.txt	(10, 1)	3
frame_19.txt	(5, 7)	
frame_1223.txt	(10, 1)	
frame_18.txt	(5, 7)	
frame_17.txt	(5, 7)	

Tabla 4.2: Documentos recuperados al aplicar recuperación de información tolerante a fallos.

4.5. Discusión sobre los experimentos

Los datos obtenidos sobre el uso de recursos computacionales revelan que la técnica descrita en este capítulo presenta un bajo costo en comparación con el empleo de redes neuronales. Esto se debe a que no es necesario almacenar grandes volúmenes de imágenes ni utilizar una GPU para el entrenamiento de modelos neuronales. Además, el tiempo de ejecución es reducido, ya que se evita no solo la construcción de un dataset especializado para la tarea, sino también el tiempo de entrenamiento y ajuste de los modelos de deep learning.

Si se analizan los datos de la Tabla 4.1 que presentan los resultados del primer experimento, se puede observar que se recuperaron 4 de las 19 escenas donde aparece la franja del sombrero en la misma posición. Cabe destacar que, para este experimento, se aplicó el mapeo basado en promedio de grises.

Por otra parte, en la Tabla 4.2 se presentan los resultados del experimento basado en códigos Hash, donde se recuperaron 9 de las 10 escenas en las que la franja del sombrero aparece en la misma posición.

Si aplicamos la métrica de precisión a estos resultados obtenemos los siguientes datos, para el primer experimento tenemos una precisión de:

$$\text{Precisión} = \frac{4}{19} = 0,21$$

y para el segundo experimento se tiene una precisión de:

$$\text{Precisión} = \frac{9}{11} = 0,81$$

Esto sugiere que el uso de código Hash ofrece mejores resultados en comparación con el mapeo basado en el promedio de colores de grises, haciendo que sea más prometedor continuar la investigación utilizando algoritmos similares al código Hash

4.6. Algoritmo de cercanía entre tokens

La estructura de los archivos de texto es de gran ayuda, ya que cada archivo puede ser visto como una matriz de m filas y n columnas, al enfocarse en una cadena de caracteres, se compara esta cadena con las cadenas vecinas, de esta forma cadenas “similares” indicarían una posible forma, es decir, se logra identificar algún objeto al encontrar todas las cadenas “similares” cercanas presentes en una imagen.

Para explicar esta idea se observa la Figura 4.9, que muestra una matriz de 6 filas y 6 columnas. En la posición (3,3) de esta matriz se encuentra la cadena “**AX00F**”. Al observar las cadenas cercanas, notamos que las cadenas en las posiciones (2,3), (2,4) y (3,4) difieren solo en un carácter a la cadena de interés. Esto puede indicar que el conjunto de cadenas $\{AX00F, AX01F, AX02F, AX03F\}$ podría corresponder a algún objeto en la imagen original. Siguiendo esta idea, se crea un módulo que aprovecha la estructura de matriz de los archivos de textos. Mediante el algoritmo de Levenshtein se obtienen todas las cadenas cercanas, finalmente obtuvimos los fragmentos de imagen que representan estas cadenas en la imagen original.

	0	1	2	3	4	5	6
0							
1							
2			BY34X	AX02F	AX03F		
3			BY56X	AX00F	AX01F		
4			BY11X	RZZ1R	RZZZ2		
5							
6							

Figura 4.9: Estructura de un archivo de texto cuya estructura se asemeja a una matriz de 6 filas y 6 columnas.

Recapitulando, en el apartado 4.4 se presentó el algoritmo basado en código Hash el cual demostró ser más efectivo para la recuperación de documentos, por este motivo, para esta sección se utiliza el corpus de texto generado por este mapeo. Cada archivo de este corpus tiene 30 filas y 10 cadenas de texto en cada fila. El algoritmo implementado para obtener conjuntos de cadenas similares a una cadena dada es el siguiente:

```
vecinos(documento, i,j, umbral):
```

```

vecinos = []
indices = []
filas, columnas = size(documento)
for x in (max(0,i-r), min(i+(r+1),filas))
    for y in range(max(0, j-r), min(j+(r+1), columnas))
        if distance(documento(i,j),documento(x,y))<umbral
            do vecinos.append(documento(x,y))
            indices.append(x,y)
return vecinos, indices

```

Donde:

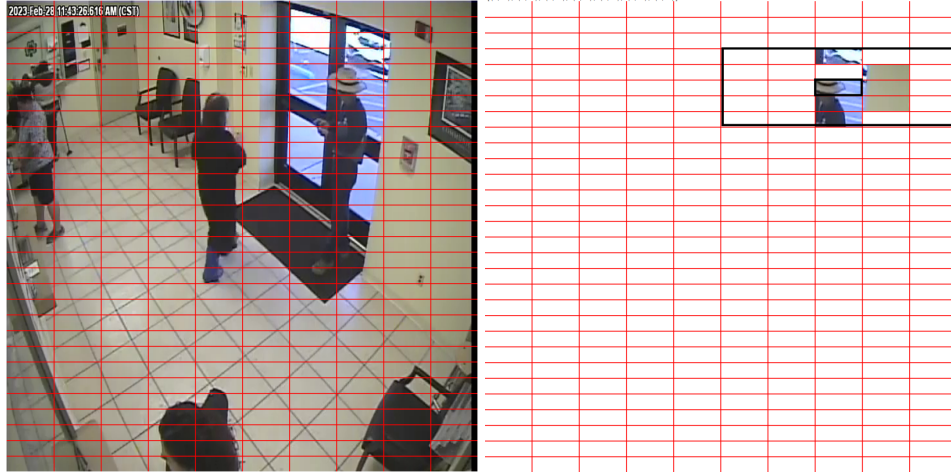
- **documento**: documento de texto a analizar.
- **i**: fila donde se encuentra la cadena a analizar.
- **j**: columna donde se encuentra la cadena a analizar.
- **filas**: número de filas del documento de texto.
- **columnas** número de columnas(cadenas) del documento de texto.
- **r**: rango que se desea analizar, por ejemplo, si $r=1$ el algoritmo buscara en las cadenas más cercanas (véase Figura 4.9 en donde se resalta en azul las cadenas a analizar con el valor $r=1$), $r > 0$.

Este algoritmo toma como entrada el documento de texto a analizar, así como la fila y columna en la que se ubica la cadena de interés y un umbral que determina que tan cercanas deben ser las cadenas para ser aceptadas como similar. Como salida, el algoritmo genera dos listas: en una se almacenan todas las cadenas similares y en la otra se encuentran las posiciones de estas cadenas.

Para el primer análisis nos enfocamos en el segmento de imagen (ver Figura 4.10a) que muestra la franja del sombrero, ubicada en la posición (5, 7) del documento de texto asociado a la imagen, la cadena que representa la franja del sombrero es **7e0f0f0fc7670f1f**, al aplicar el algoritmo con el parámetro *umbral* = 10 se obtiene el siguiente conjunto de cadenas:

```
{1c0f3f2f07070100, f0f0f8787878f8fc, 6060e0f0f0f0f8f8,
 1f0f0f0f0f070707, f0f0f0f0f0f0f0f0, 0707070707070707,
 7e0f0f0fc7670f1f }
```

En la Figura 4.10 se presenta la imagen analizada (ver Figura 4.10a) y los segmentos de imágenes recuperados (ver Figura 4.10b) que están asociados al conjunto de cadenas obtenidas mediante el algoritmo descrito.



(a) Imagen original dividida en 30 filas y 10 columnas (b) Segmentos de imágenes recuperadas al aplicar el algoritmo enfocándose en la franja del sombrero.

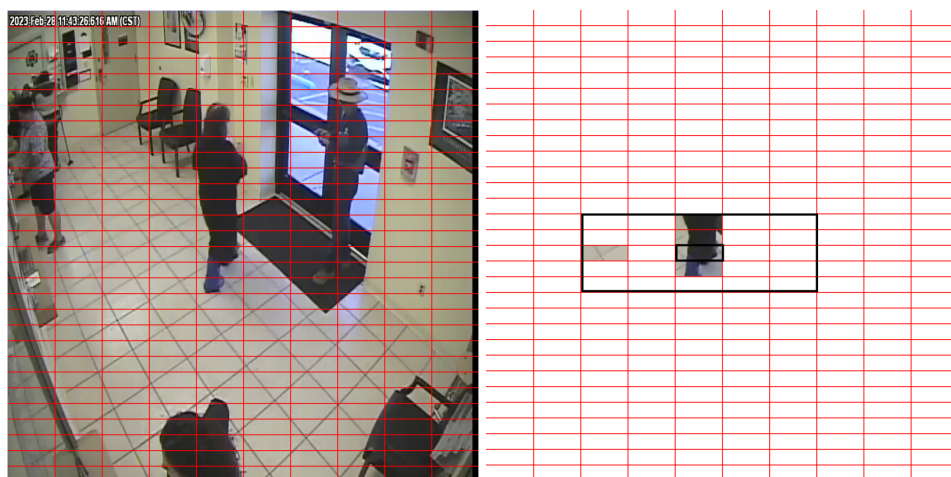
Figura 4.10: Experimento: Analizar la cadena asociada a la franja del sombrero.

Después realizar varios experimentos modificando los parámetros *umbral* y *r* se determinó que los valores de $r = 2$ y $umbral = 10$ permiten recuperar un mayor número de segmentos de imágenes similares.

En las siguientes figuras se presentan varios resultados de conjuntos de segmentos que se encuentran en la misma imagen, para ello se toma los valores de $r = 2$ y $umbral = 10$, a la izquierda se muestra la imagen base, mientras que a la derecha se ilustra el segmento de imagen analizado, la vecindad en donde se buscó cadenas similares y los segmentos de imágenes recuperados.

En la Figura 4.11 se muestra el resultado de analizar una parte de una prenda de una persona, se puede ver que al fondo de este segmento se ve un par de azulejos lo que influye

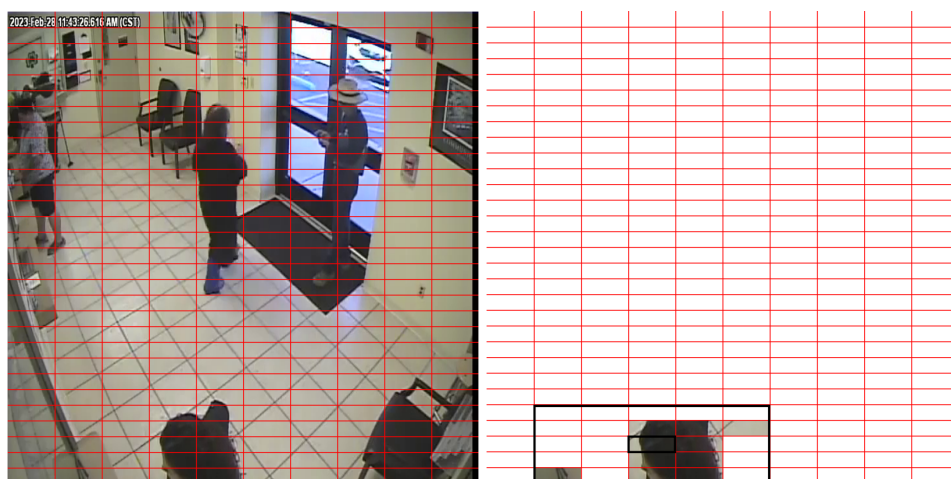
en el resultado al obtener un azulejo como parte del conjunto de segmentos similares.



(a) Imagen original dividida en 30 filas y 10 columnas
(b) Segmentos de imágenes recuperadas al aplicar el algoritmo en la parte inferior de una persona.

Figura 4.11: Experimento: Analizar la cadena asociada a las piernas de una persona.

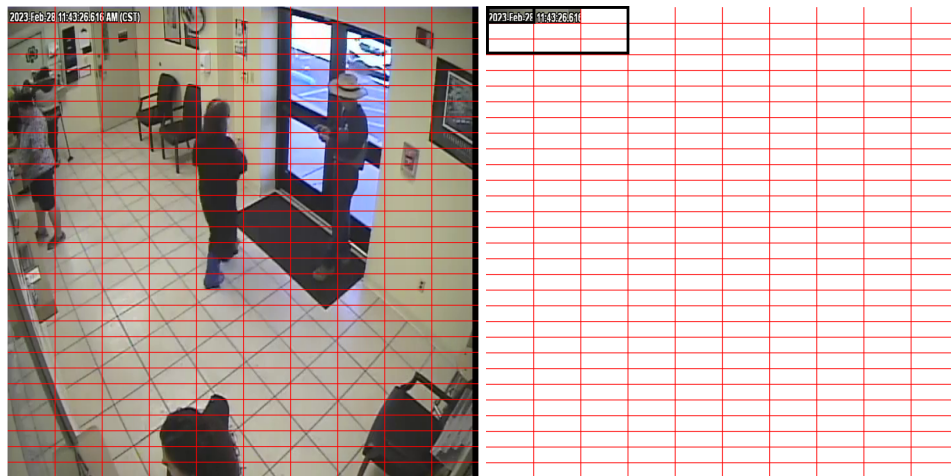
En la Figura 4.12 se presenta el resultado obtenido al analizar parte de la cabeza de una persona, nuevamente se aprecia que parte del segmento de imagen es un azulejo lo que influye en los segmentos similares obtenidos.



(a) Imagen original dividida en 30 filas y 10 columnas
(b) Segmentos de imágenes recuperadas al aplicar el algoritmo en el cabello de una persona.

Figura 4.12: Experimento: Analizar la cadena asociada al cabello de una persona.

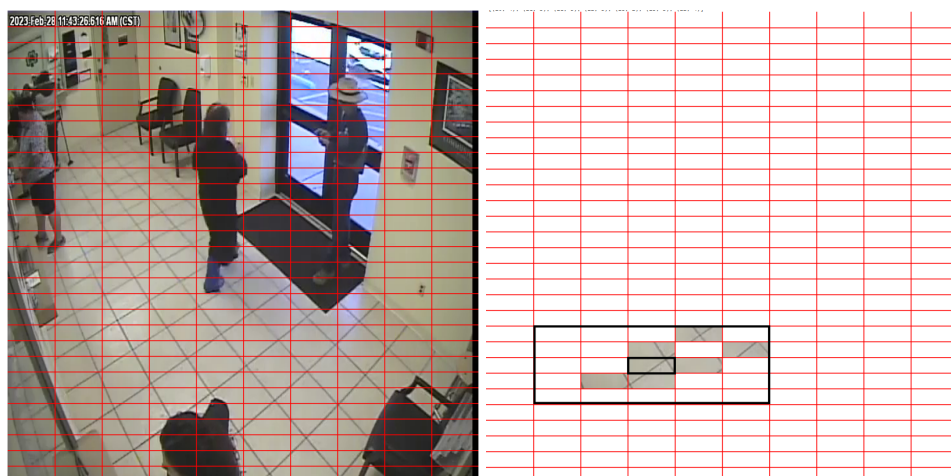
El resultado de analizar el segmento de la fecha y hora se presenta en la Figura 4.13, se destaca que el algoritmo no recuperó un segmento similar presente en la imagen.



(a) Imagen original dividida en 30 filas y 10 columnas
(b) Segmentos de imágenes recuperadas al aplicar el algoritmo en la fecha de la imagen

Figura 4.13: Experimento: Analizar la cadena asociada a la fecha, en particular al segmento de imagen que muestra los caracteres 2023-Feb.

Finalmente, en la Figura 4.14 se presenta el resultado al analizar un segmento que contiene azulejos, se observa que el algoritmo no recupera todos los segmentos similares.



(a) Imagen original dividida en 30 filas y 10 columnas
(b) Segmentos de imágenes recuperadas al aplicar el algoritmo en un azulejo del piso.

Figura 4.14: Experimento: Analizar la cadena asociada a un azulejo en el piso.

Al realizar el análisis con los resultados obtenidos de los experimentos de las figuras

4.10, 4.11, 4.12, 4.13 y 4.14 se concluye que es posible obtener representaciones textuales (conjuntos de caracteres) que identifiquen a objetos en las imágenes. Sin embargo, de estos resultados se desprenden diversas preguntas, por ejemplo:

1. ¿De que forma se pueden definir los parámetros r y $umbral$ para obtener mejores resultados?
2. ¿Cuántas divisiones se deben aplicar a la imagen para obtener mejores resultados?
3. ¿Cómo resolver los problemas de cambios de luz y escalamiento?

Es importante señalar que estas preguntas presentan un desafío considerable, ya que la metodología aplicada en este proyecto de tesis es innovadora y, hasta el momento no es posible encontrar en el estado del arte investigaciones que ayuden a solucionar las preguntas 1 y 2. Esto convierte el tema de esta tesis en una gran área de oportunidad para la investigación, sin embargo, debido a limitaciones de tiempo, el proyecto de tesis no se centró en dar respuesta a las preguntas. Durante las discusiones sobre el rumbo de la tesis, se concluyó que profundizar en esta rama es adentrarse en un proyecto de mayor envergadura. No obstante, se han sentado las bases para futuras investigaciones y se ha cumplido con los objetivos planteados para este proyecto de tesis.

4.7. Conclusiones.

En la actualidad el desarrollo de la inteligencia artificial impulsa significativamente el avance en las tecnologías, pero es crucial recordar que no todos los organismos y personas tienen acceso a los recursos computacionales requeridos para estos algoritmos, ni el tiempo que se requiere para desarrollarlos. Por esta razón, es vital continuar investigando técnicas alternativas. Esta tesis demuestra que es posible encontrar partes de un objeto de interés en una imagen mediante la recuperación de información, sin embargo, los resultados obtenidos sugieren que es posible encontrar objetos en su totalidad y no solo una sección de ellos, sin la necesidad del uso de aprendizaje profundo.

Con los hallazgos presentados en esta investigación, se abre el camino hacia el desarrollo de un método eficiente para la recuperación de escenas de interés en videos de larga duración sin requerir recursos computacionales elevados.

Capítulo 5

Sistema de recuperación de escenas de interés en videos de larga duración

Las ideas poderosas están en tu bolsillo.

“1984”, George Orwell

En esta sección, se presenta un innovador sistema de recuperación de información que identifica de manera rápida y a bajo costo los objetos presentes en videos de seguridad de larga duración. Para lograr esta tarea, el video se procesa utilizando un detector de objetos (modelo preentrenado de redes neuronales), lo que permite generar un corpus de texto a partir de las etiquetas obtenidas. Este corpus alimenta un sistema de recuperación de información que utiliza técnicas altamente eficientes. Finalmente, a través de una aplicación web, los usuarios pueden interactuar con el sistema ingresando sus consultas en lenguaje natural y obtener en cuestión de milisegundos la localización en espacio y tiempo de los objetos buscados en todo el video.

5.1. Metodología

En la la Figura 5.1 se ilustra la metodología que conforma el sistema de recuperación de información descrito en este capítulo, este sistema se compone de tres elementos principales: en primer lugar, un módulo que genera un corpus de texto a partir de un video de larga duración; en segundo lugar, un sistema de recuperación de información textual que utiliza el

corpus generado. Finalmente, la interacción entre el usuario y el sistema se realiza a través de una aplicación web

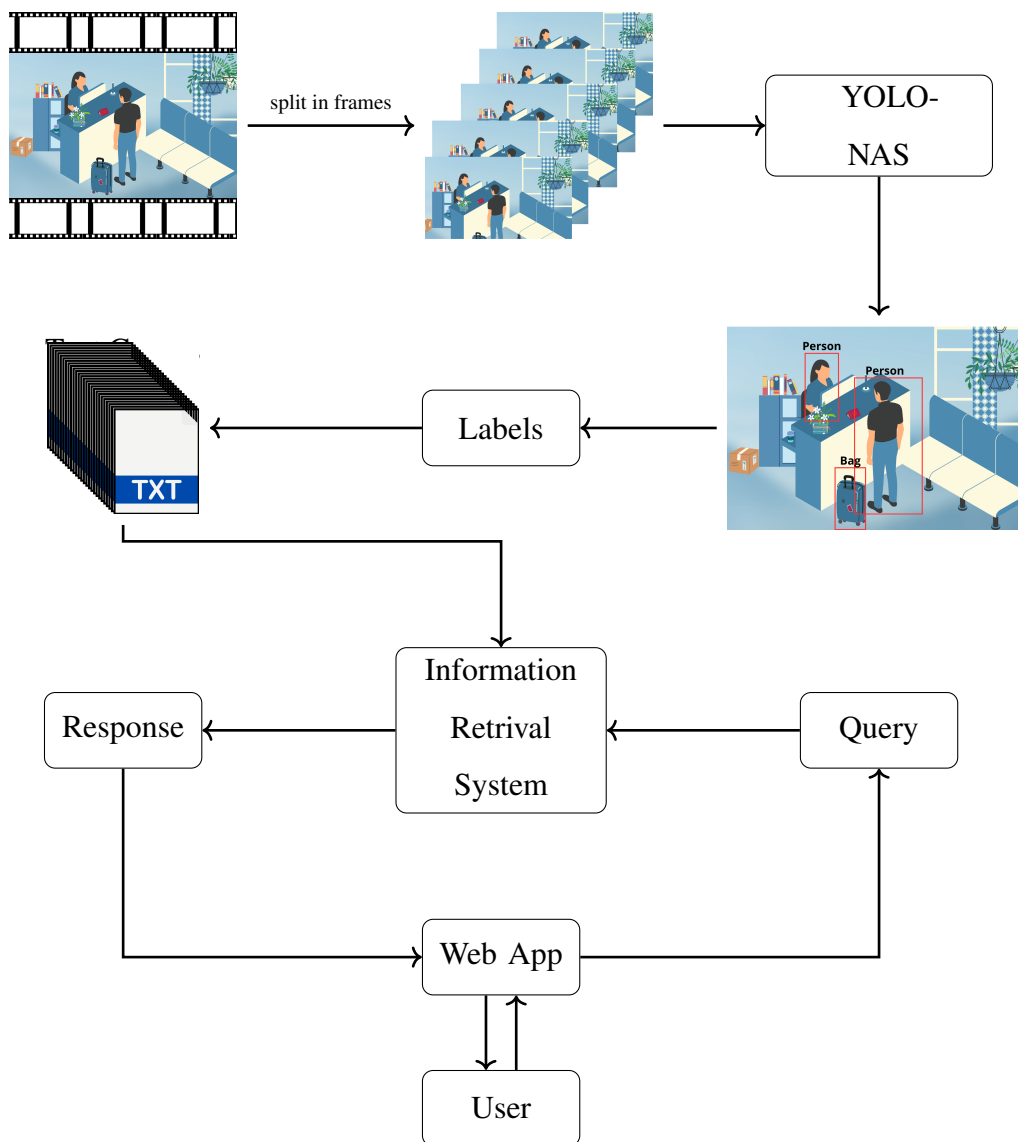


Figura 5.1: Metodología del sistema de recuperación de objetos para una rápida identificación de objetos en videos de larga duración.

El flujo del sistema es el siguiente: primero, se extraen frames de un video de larga duración. Cada imagen se procesa mediante el modelo de detección de objetos YOLO-NAS, se recuperan las etiquetas de los objetos que identificó el modelo YOLO-NAS. Estas etiquetas se almacenan en un archivo de texto, y el conjunto de todos los archivos de texto conforman un corpus de texto que alimenta a un sistema de recuperación de información de texto. A través de una aplicación web el usuario escribe una consulta en lenguaje natural, esta consulta

es enviada al sistema de recuperación de información, que procesa la solicitud y como respuesta se obtienen los documentos relevantes a esta consulta. Los resultados son presentados al usuario mediante la aplicación web.

5.2. Desarrollo del sistema de recuperación de información

5.2.1. Creación del corpus de texto

Para crear el corpus de texto se realiza el siguiente proceso:

1. Se extraen imágenes del vídeo a intervalos de tiempo definidos t , cada imagen es nombrada como I_t .
2. Cada imagen se procesa con el modelo de detección de objetos YOLO-NAS y se obtienen las etiquetas de los objetos identificados por el modelo. Con base en estas etiquetas, se define un mapeo f como sigue:

$$f(I_t) = etiquetas(YOLO - NAS(I_t)).$$

3. Las etiquetas obtenidas mediante el mapeo f se unen con un espacio en blanco y se guardan en un documento de texto
4. El compendio de todos los archivos de textos generados se utiliza para formar un corpus texto que posteriormente se indexa para alimentar al sistema de recuperación de información.

En cada uno de los pasos descritos, es crucial definir diversos parámetros, por ejemplo, el intervalo para obtener un frame y el preprocesamiento realizado a la imagen antes de ser procesada por el modelo de detección de objetos. Es fundamental investigar cómo varían de estos parámetros y sus efectos en el sistemas, en una sección posterior se detalla las particularidades utilizadas para realizar del análisis presentado en este documento.

5.2.2. Sistema de recuperación de información booleano tolerante a fallos con búsqueda por sinónimos

En la Figura 5.2 se describe el funcionamiento del sistema de recuperación textual implementado. Inicialmente, el corpus creado pasa por un módulo de preprocesamiento de texto antes de generar un índice invertido durante la etapa de indexación. Este índice invertido es esencial para la rapidez del sistema, ya que permite ejecutar consultas con una complejidad $O(1)$.

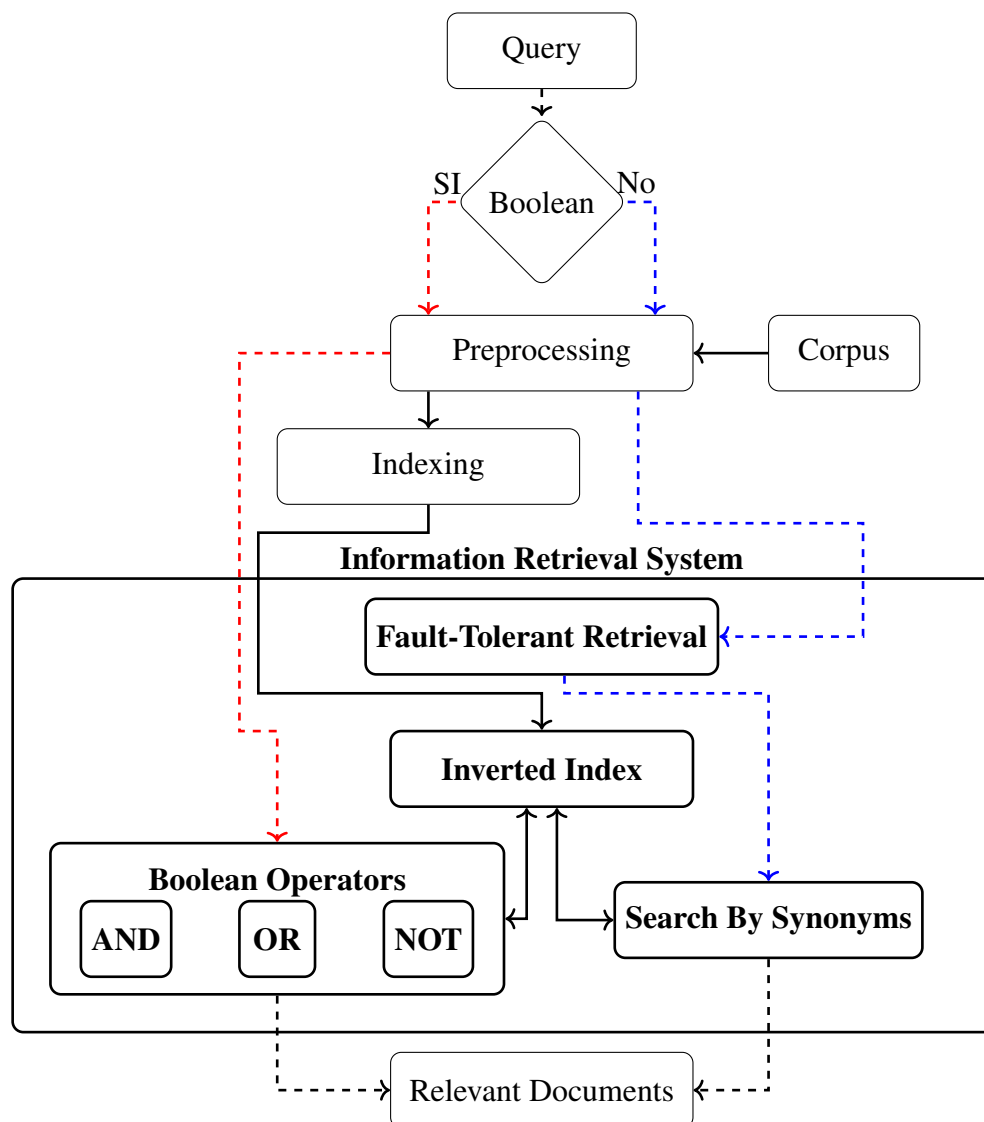


Figura 5.2: Sistema de recuperación de información booleano tolerante a fallos con búsqueda por sinónimos implementado

Cuando un usuario ingresa una consulta en formato de texto a través de la aplicación web, el sistema determina si la consulta es booleana o no, enseguida la consulta se somete al mismo preprocesamiento utilizado para el corpus de texto con el objetivo de homogeneizar la consulta y el texto presente en el corpus. En caso de que se identifique una consulta booleana, se aplica el operador booleano correspondiente. Por otro lado, si la consulta no es booleana se buscan coincidencias en el índice invertido.

Para mejorar la robustez del sistema, se implementan dos módulos adicionales. El primero es la recuperación tolerante a fallos, que intenta encontrar coincidencias en el índice invertido utilizando la distancia de Levenshtein [[Levenshtein et al. \(1966\)](#)]. De esta manera, si la consulta ingresada no se encuentra en el índice invertido (debido a errores tipográficos), se recuperan elementos que se encuentren dentro de una distancia determinada, es decir, el sistema encontrará la palabra más cercana mediante el algoritmo de Levenshtein. El segundo modulo implementa búsqueda por sinónimos, si se introduce un sinónimo de un término en el índice invertido, el sistema devuelve todos los elementos asociados al término en el índice. Así, se crea un sistema de recuperación de información booleano, tolerante a fallos y con expansión de búsqueda por sinónimos.

La aplicación web se desarrolló utilizando Django, un framework de desarrollo web en Python reconocido por su capacidad para simplificar y agilizar la creación de aplicaciones web. Se eligió este framework debido a su estructura robusta, la disponibilidad de componentes reutilizables y su escalabilidad, además de contar con sistemas de plantillas prácticas y optimizadas para la visualización de resultados. Esto permitió la creación de una aplicación web que, aunque sencilla e intuitiva, resulta muy potente.

5.3. Particularidades del hardware y del sistema de recuperación de información

Para el análisis que se presenta en este capítulo se descargó un video de internet de 5 horas 6 minutos y 4 segundos. El video muestra el lobby de una oficina [[Corners-News \(2023\)](#)], y el objetivo es identificar los siguientes objetos comunes en una oficina: personas, mochilas, teléfonos y computadoras. Además, es importante evaluar el funcionamiento del sistema

en un equipo sin GPU, por lo que el análisis se realiza en un hardware con las siguientes características:

- **Modelo de hardware:** Apple Inc. MacBookPro9,2. (2012).
- **Sistema operativo:** Ubuntu 22.04.4 LTS 64 bits.
- **Procesador:** Intel® Core™ i5-3210M CPU @ 2.50GHz × 4.
- **Memoria:** 8GB
- **Tarjeta gráfica:** Mesa Intel® HD Graphics 4000 (IVB GT2).
- **Almacenamiento:** 500.1 GB.

El experimento presentado en este artículo cuenta con las siguientes características: cada 5 segundos se captura un frame del video, esta imagen es convertida a escala de grises y redimensionada a 680×680 . Posteriormente este frame es procesado con el modelo YOLO-NAS preentrenado en el conjunto de datos COCO con una precisión de 0.7, el modelo obtiene los cuadros delimitadores y las clases a las que pertenece, los cuadros delimitadores son pintados en el frame para visualizar los objetos en la imagen y las etiquetas se almacenan en un archivo de texto con un nombre que asocia al tiempo en que se obtuvo el frame, de esta forma los objetos son localizados en tiempo y espacio dentro del video.

Con el corpus de entrenamiento generado, se crea un índice inverso, este índice tarda aproximadamente 124 milisegundos en ser creado y se almacena en un archivo JSON, facilitando su uso en la aplicación web. El archivo JSON tiene un tamaño de 162.2 KB, lo que permite su carga en memoria aún en equipos con menor capacidad de memoria RAM. El posting list creado es la base del sistema de recuperación de información previamente descrito.

En la siguiente sección se presenta una discusión de las respuestas del sistema a diversas consultas, la ejecución de los módulos implementados y el tiempo que tarda en presentar los resultados.

5.4. Resultados obtenidos al implementar el sistema de recuperación de información

Es natural preguntarse cuánto tiempo tarda el sistema en construir el corpus de texto, este tiempo depende de dos factores principales: el intervalo para la captura de frames del video y el modelo de detección de objetos utilizado, en el sistema se ocupó el modelo YOLO-NAS [Aharon et al. (2021)]. la ventaja de este modelo radica en su arquitectura ya que emplea bloques sensibles a la cuantificación, esto optimiza el rendimiento y aumenta la precisión en comparación con otros modelos YOLO, además es un modelo que ocupa bajos recursos computacionales.

En la Tabla 5.1 se presentan distintos intervalos de tiempo seleccionados para obtener un frame del video, el tiempo aproximado en minutos que tarda el sistema en crear el corpus de texto, la cantidad de archivos de texto generados (cantidad de frames procesados) y la cantidad de memoria en disco que ocupa el corpus de texto generado.

Intervalo	Tiempo en generar el corpus	Archivos de texto	Memoria de disco
5	53 minutos	3672	77.1 KB
10	31 minutos	1837	38.2 KB
15	18 minutos	1224	25.8 KB

Tabla 5.1: Datos obtenidos al generar el corpus de texto tomando diferentes intervalos de tiempo para obtener un frame del video.

Como era de esperarse el intervalo de tiempo es un factor crucial en la duración del proceso para generar el corpus, ya que a menor intervalo de tiempo mayor es el número de imágenes que deben ser procesadas. Sin embargo, el tamaño del corpus resultante es relativamente pequeño, lo que permite que la generación del índice inverso sea rápida y que cargar este índice en memoria tenga un costo muy bajo.

Al dividir el tiempo total que tarda el sistema en generar el corpus de texto por el número de documentos de texto obtenidos, se calcula el tiempo requerido para crear cada archivo de texto, que resulta ser aproximadamente 1 segundo, este dato es importante ya que nos indica que el sistema propuesto puede ser implementado en tiempo real y que se puede obtener un frame cada 1.5 segundos de un sistema de video vigilancia y antes de obtener el siguiente

frame, el sistema ya habrá procesado, guardado y actualizado el índice inverso con la nueva información.

Con los parámetros definidos en la sección 5.3 se realiza la búsqueda utilizando la consulta *person person* en la Figura 5.3 se observa que el sistema devuelve 2913 imágenes que contienen a una persona reconocida por el modelo de identificación de objetos. Es importante destacar que, al presentar los resultados se le notifica al usuario el momento exacto en el que puede localizar la imagen presentada mediante una leyenda.

Web App create for:
Edgar Abidán Padilla Luis

[Home](#)



BUAP

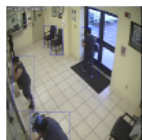
person

Search

Home

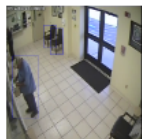
Results (2913):

[Go to video](#)



The image is located at time 03:08:15

[Go to video](#)



The image is located at time 03:31:50

Figura 5.3: Interfaz web desarrollada, en ella se presenta un cuadro de búsqueda en donde se escribe la consulta *person*. En la parte inferior del cuadro de búsqueda la aplicación web posee un de búsqueda y un botón que redirige al inicio. En la parte inferior se presenta los resultados de la consulta.

Al hacer click en **Go to video** se muestra la imagen correspondiente al tiempo 03:08:15 (indicado en la leyenda debajo de la imagen). En la imagen se observan diversos cuadros

delimitadores, tres de ellos corresponden a personas y los otros dos corresponden a otros objetos identificados. Esto se debe a que los cuadros se pintan al momento de procesar la imagen por el modelo de detección de objetos y no durante la consulta. Esta metodología permite que el sistema recupere los elementos en cuestión de milisegundos.



Figura 5.4: Primer imagen recuperada de la consulta *person*.

Es importante notar que el modelo de detección de objetos no logra detectar a una cuarta persona. Este resultado podría variar al ajustar el nivel de precisión del modelo. Además, se debe considerar que la calidad del video es baja lo que puede afectar al modelo de detección de objetos.

Si se realiza una consulta con la palabra *perosn* simulando que se ha escrito incorrectamente la palabra *person*. En la Figura 5.5 se observa que el sistema de recuperación muestra la misma cantidad de resultados (2913) y el mismo tiempo (03:08:15) en el que se encuentra

ubicado el primer resultado ofrecido por la consulta *person*. Al hacer click en **Go to video** el sistema devuelve la imagen presentada en la Figura 5.4, además el sistema muestra abajo del número de resultados obtenidos una leyenda (Potentially interesting: person) que sugiere que esta imagen podría estar relacionada con la consulta realizada.

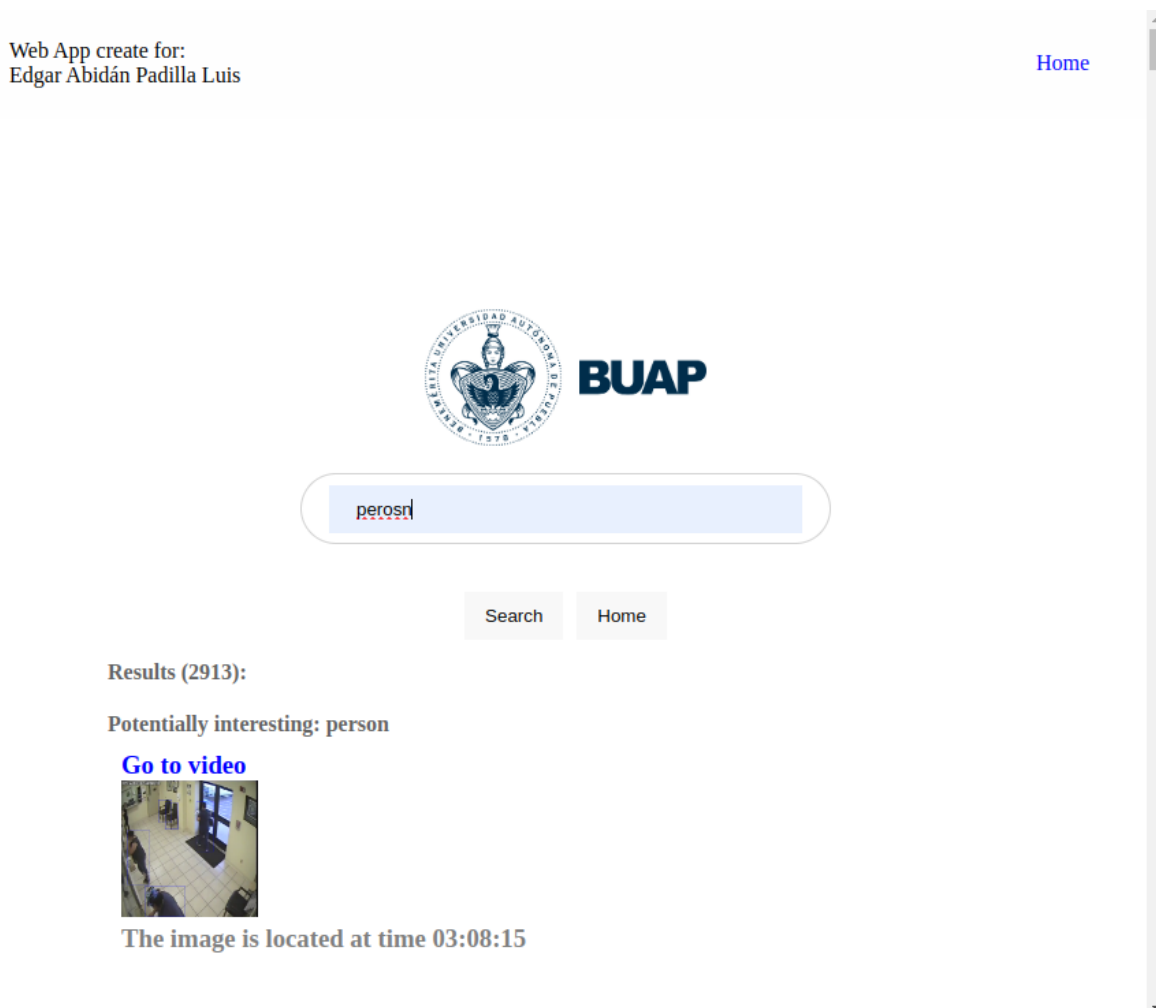


Figura 5.5: Respuesta del sistema a la consulta *perosn*. En la interfaz se observa el funcionamiento del modulo tolerante a fallos al devolver resultados que corresponden a la palabra *person*.

Generalmente, las consultas booleanas son ingresadas por personas que poseen conocimiento sobre operadores lógicos y suelen estar implementadas en los motores de búsqueda en la parte de herramientas avanzadas. Sin embargo, es muy común que las personas utilicen frases conectores que implican operaciones lógicas. Por esta razón, el sistema está diseñado para identificar ciertos conectores gramaticales que suelen ser operaciones booleanas, esto lo hace mediante el uso del siguiente diccionario

```
{ "and": "AND", "or": "OR", "not": "NOT", "with": "AND",
  "including": "AND", "also": "AND", "plus": "AND",
  "besides": "AND", "moreover": "AND", "furthermore": "AND",
  "alternatively": "OR", "instead": "OR", "otherwise": "OR",
  "else": "OR", "either": "OR", "nor": "OR", "elsewhere": "OR",
  "except": "NOT", "excluding": "NOT", "without": "NOT",
  "minus": "NOT" }
```

De esta forma no es necesario conocer como funcionan los operadores lógicos, solo basta con que el sistema identifique conectores lógicos presentes en el diccionario en consultas como *person with phone* para hacer la búsqueda booleana *person AND phone*. En Fig 5.6 se muestra el resultado de la consulta *person with phone*.

Web App create for:
Edgar Abidán Padilla Luis

[Home](#)



BUAP

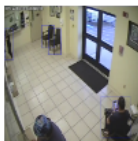
person with phone

Search

Home

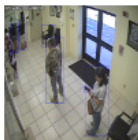
Results (4):

[Go to video](#)



La imagen se encuentra en el tiempo 03:07:40

[Go to video](#)



La imagen se encuentra en el tiempo 02:08:25

Figura 5.6: Respuesta del sistema a la consulta *person with phone*. Esta consulta posee el conector *with* que hace referencia al operador lógico *AND*.

En la Figura 5.7 se muestra el resultado para la consulta booleana *person AND phone*, esta consulta muestra los mismos resultados que los que se observa en la Figura 5.6.

Web App create for:
Edgar Abidán Padilla Luis

[Home](#)



BUAP

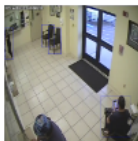
person AND phone

Search

Home

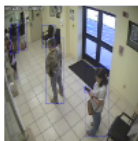
Results (4):

[Go to video](#)



La imagen se encuentra en el tiempo 03:07:40

[Go to video](#)



La imagen se encuentra en el tiempo 02:08:25

Figura 5.7: Respuesta del sistema a la consulta booleana especializada *person AND phone*

Las consultas booleanas son fundamentales en este sistema, ya que es posible rastrear momentos en los que una persona porta un celular, una laptop o una mochila, estas consultas pueden ser de interés en las tareas de video vigilancia.

La incorporación del módulo de expansión de consultas por sinónimos ha sido de suma importancia, dado que los modelos de redes neuronales generalmente suelen asociar una única etiqueta a cada clase. De esta forma si un usuario no conoce las clases del modelo con el que se detectaron los objetos es muy probable que no encuentre los objetos que busca. Al utilizar un diccionario de sinónimos es posible asociar a las etiquetas de las clases del modelo de detección de objetos sinónimos o palabras similares. De esta manera, el usuario puede

realizar las búsquedas con términos como *individual* o *human* y obtener las mismas respuestas que la consulta *person*. En la Figura 5.8 se muestra que los resultados de la consulta con *human* son idénticos a los de la consulta con *person* presentada en la Figura 5.3.

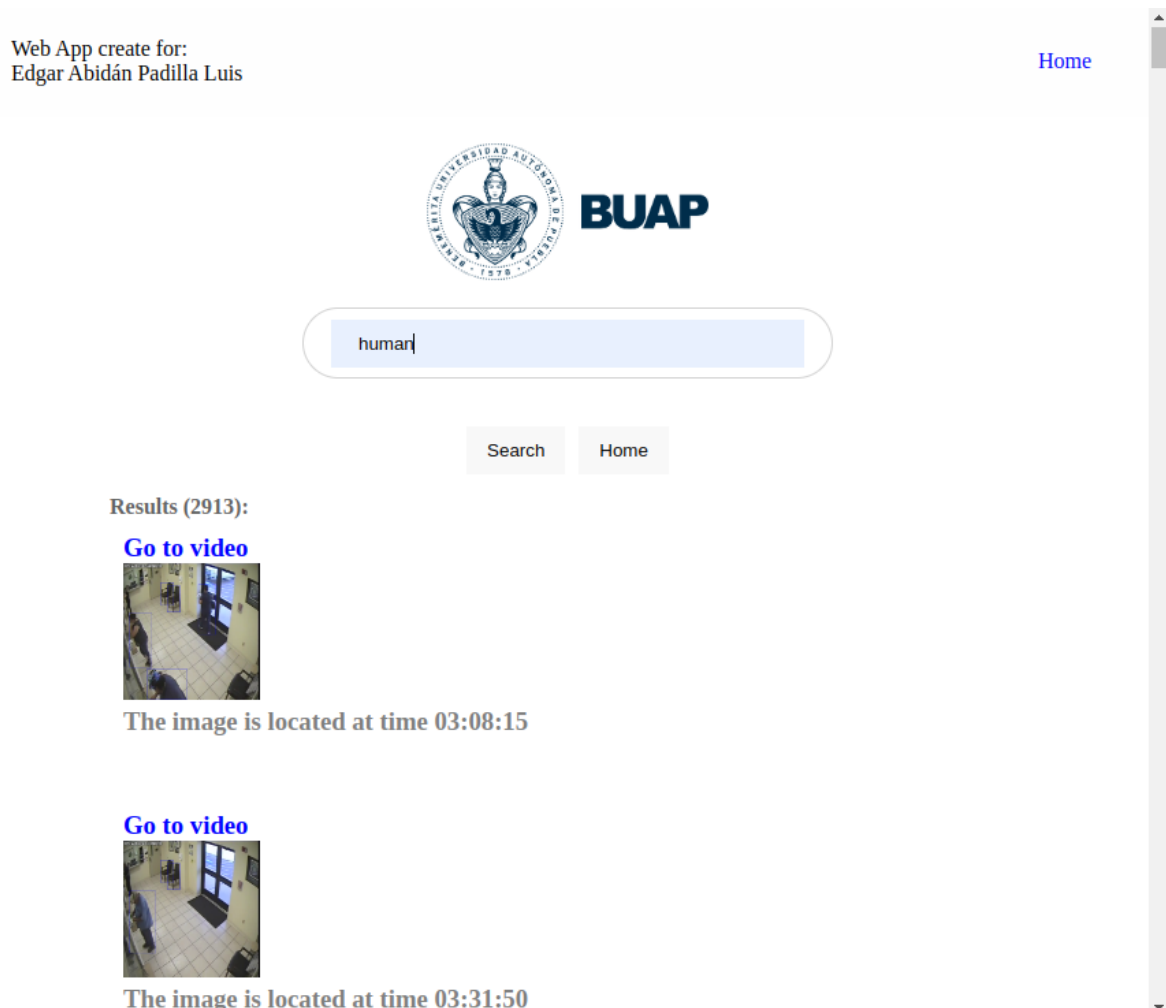


Figura 5.8: Respuesta del sistema a la consulta *human*. Se observan los mismos resultados a la consulta *person*, mostrando el funcionamiento del módulo de expansión de consulta por sinónimos.

5.5. Conclusiones

Este capítulo presenta un sistema de recuperación de información para identificación rápida de objetos en videos de vigilancia de larga duración, para ello se conforma un corpus de texto de las etiquetas de las clases de un modelo de detección de objetos, con este corpus se construye un sistema de recuperación de información basado en técnicas de recuperación de

texto, esto hace que el sistema sea eficiente y arroje consultas en milisegundo que el usuario puede visualizar en una aplicación web amigable. Se ha mostrado que es viable adaptar el sistema para ejecutarse en tiempo real en equipos que no cuenten con grandes recursos computacionales. Además, se ha mostrado que mediante el uso de diccionarios un usuario no especializado puede hacer consultas especializadas, esto hace que el sistema sea muy intuitivo y amigable con el usuario final. Finalmente, es importante notar que en este artículo se explica como el módulo de búsqueda booleana puede ser ocupado para tareas relacionadas a la video vigilancia.

Es importante hacer notar que la metodología ocupada puede ser adaptada a diversos modelos ampliamente ocupados, que ocupan bajos recursos y que se emplean para diversas tareas como el reconocimiento facial, lo cual hace que este artículo sea una base para una amplia gama de aplicaciones.

Capítulo 6

Conclusiones

Nadie se baña en el río dos veces, porque todo cambia en el río y en el que se baña.

Heráclito de Éfeso

Esta tesis propone una metodología novedosa para generar representaciones textuales de escenas de interés en un video. Cabe destacar que, según la revisión del estado del arte, no se ha encontrado un trabajo similar al respecto. Esta metodología permite obtener representaciones textuales que optimizan el proceso de recuperación de información, haciéndolo menos costoso en comparación con los métodos basados en redes neuronales comúnmente utilizados en sistemas de recuperación de información.

En el capítulo 4, se presentaron dos contribuciones importantes. En primer lugar, se demostró que es posible codificar una imagen en texto utilizando diversas estrategias. Tras realizar una serie de experimentos, se concluyó que el algoritmo basado en la conversión de imágenes a códigos Hash, combinado con técnicas de recuperación de información tolerantes a fallos, logra recuperar más documentos relevantes que el algoritmo basado en el promedio de píxeles en escala de grises. Además, se aprovechó la estructura de los documentos de texto generados para proponer un algoritmo de proximidad entre tokens con el objetivo de identificar objetos presentes en una imagen. Los resultados experimentales indicaron que es posible encontrar segmentos de imágenes que corresponden a un mismo objeto. Debido a las limitaciones de tiempo, no fue posible seguir perfeccionando estas técnicas; sin embargo, se deja un precedente para futuras investigaciones.

En el capítulo 5, se desarrolló un sistema de recuperación de información para la identificación de escenas de interés en videos de larga duración. En este caso, se utilizó el modelo de detección de objetos YOLO-NAS para obtener las representaciones textuales, es decir, el mapeo f se basó esencialmente en este modelo. También se construyó una aplicación web que permite a usuarios no especializados realizar búsquedas complejas, como búsquedas booleanas, utilizando diccionarios para ofrecer un sistema intuitivo y amigable para el usuario final. Además, se demostró que esta metodología es fácilmente aplicable a aplicaciones en tiempo real y puede utilizarse en diversas tareas relacionadas con la videovigilancia.

La hipótesis planteada al inicio de esta tesis fue comprobada: las representaciones textuales de imágenes, obtenidas mediante las técnicas desarrolladas en los capítulos 4 y 5, son eficientes para realizar búsquedas de escenas de interés en videos de larga duración, en comparación con los métodos tradicionales basados en coincidencia de imágenes. No obstante, aún es necesario refinar los métodos para alcanzar niveles de precisión similares a los de las técnicas que utilizan redes neuronales.

Es relevante señalar que los resultados obtenidos en los capítulos 4 y 5 fueron presentados en dos pósters durante el *9th International Symposium on Language & Knowledge Engineering (LKE 2024)*, celebrado en Dublín, Irlanda.

El proyecto de tesis propone una metodología innovadora para la recuperación de información multimedia, lo cual representa una gran área de oportunidad que seguirá siendo explorada. Además, la variedad de tareas en las que puede ser aplicado es amplia, lo que convierte este tema en un punto de interés tanto para organismos públicos como privados.

Bibliografía

- Abdusalomov, A., Baratov, N., Kutlimuratov, A., and Whangbo, T. K. (2021). An improvement of the fire detection and classification method using yolov3 for surveillance systems. *Sensors*, 21(19):6519.
- Aharon, S., Louis-Dupont, Ofri Masad, Yurkova, K., Lotem Fridman, Lkdci, Khvedchenya, E., Rubin, R., Bagrov, N., Tymchenko, B., Keren, T., Zhilko, A., and Eran-Deci (2021). Super-gradients.
- Alsmadi, M. K. (2020). Content-based image retrieval using color, shape and texture descriptors and features. *Arabian Journal for Science and Engineering*, 45(4):3317–3330.
- Bain, M., Nagrani, A., Varol, G., and Zisserman, A. (2021). Frozen in time: A joint video and image encoder for end-to-end retrieval. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1728–1738.
- Butt, H., Raza, M. R., Ramzan, M. J., Ali, M. J., and Haris, M. (2021). Attention-based cnn-rnn arabic text recognition from natural scene images.
- Castanon, G., Elgharib, M., Saligrama, V., and Jodoin, P.-M. (2015). Retrieval in long-surveillance videos using user-described motion and object attributes. *IEEE Transactions on Circuits and Systems for Video Technology*, 26(12):2313–2327.
- Chambi, J. L. S., Cáceres, J. C. G., and Castañón, C. A. B. (2022). Densenet 3d for violent action recognition in surveillance video sequences. In *2022 41st International Conference of the Chilean Computer Science Society (SCCC)*, pages 1–8.
- Coelho, F. and Ribeiro, C. (2010). Evaluation of global descriptors for multimedia retrieval in

- medical applications. In *2010 Workshops on Database and Expert Systems Applications*, pages 127–131. IEEE.
- Corners-News (2023). Uisd tax office lobby video footage. <https://www.youtube.com/watch?v=-gV67VpzP5M>. June, 2023.
- Deselaers, T., Keyzers, D., and Ney, H. (2008). Features for image retrieval: an experimental comparison. *Information retrieval*, 11:77–107.
- Dodds, E., Culpepper, J., and Srivastava, G. (2022). Training and challenging models for text-guided fashion image retrieval. *arXiv preprint arXiv:2204.11004*.
- Dubey, S. R. (2021). A decade survey of content based image retrieval using deep learning. *IEEE Transactions on Circuits and Systems for Video Technology*, 32(5):2687–2704.
- Dubey, S. R., Singh, S. K., and Chu, W.-T. (2022). Vision transformer hashing for image retrieval. In *2022 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6. IEEE.
- Elharrouss, O., Almaadeed, N., and Al-Maadeed, S. (2021). A review of video surveillance systems. *Journal of Visual Communication and Image Representation*, 77:103116.
- Fang, H., Xiong, P., Xu, L., and Chen, Y. (2021a). Clip2video: Mastering video-text retrieval via image clip. *arXiv preprint arXiv:2106.11097*.
- Fang, J., Fu, H., and Liu, J. (2021b). Deep triplet hashing network for case-based medical image retrieval. *Medical image analysis*, 69:101981.
- Gordo, A., Almazan, J., Revaud, J., and Larlus, D. (2017). End-to-end learning of deep visual representations for image retrieval. *International Journal of Computer Vision*, 124(2):237–254.
- Haering, N., Venetianer, P. L., and Lipton, A. (2008). The evolution of video surveillance: an overview. *Machine Vision and Applications*, 19(5-6):279–290.
- Han, L., Li, P., Bai, X., Grecos, C., Zhang, X., and Ren, P. (2019). Cohesion intensive deep hashing for remote sensing image retrieval. *Remote Sensing*, 12(1):101.

- Ingle, P. Y. and Kim, Y.-G. (2022). Real-time abnormal object detection for video surveillance in smart cities. *Sensors*, 22(10):3862.
- Iscen, A., Caron, M., Fathi, A., and Schmid, C. (2023). Retrieval-enhanced contrastive vision-text models. *arXiv preprint arXiv:2306.07196*.
- Jégou, H., Douze, M., Schmid, C., and Pérez, P. (2010). Aggregating local descriptors into a compact image representation. In *2010 IEEE computer society conference on computer vision and pattern recognition*, pages 3304–3311. IEEE.
- Karbalaie, A., Abtahi, F., and Sjöström, M. (2022). Event detection in surveillance videos: a review. *Multimedia tools and applications*, 81(24):35463–35501.
- Kumar, B. V., Abirami, S., Lakshmi, R. B., Lohitha, R., and Udhaya, R. (2019). Detection and content retrieval of object in an image using yolo. In *IOP conference series: materials science and engineering*, volume 590, page 012062. IOP Publishing.
- Levenshtein, V. I. et al. (1966). Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, volume 10, pages 707–710. Soviet Union.
- Li, S., Yuan, F., Ata-UI-Karim, S. T., Zheng, H., Cheng, T., Liu, X., Tian, Y., Zhu, Y., Cao, W., and Cao, Q. (2019). Combining color indices and textures of uav-based digital imagery for rice lai estimation. *Remote Sensing*, 11(15):1763.
- Luo, Y., Qin, J., Xiang, X., Tan, Y., Liu, Q., and Xiang, L. (2020). Coverless real-time image information hiding based on image block matching and dense convolutional network. *Journal of Real-Time Image Processing*, 17:125–135.
- Manning, C. D. (2009). *An introduction to information retrieval*. Cambridge university press.
- McCulloch, W. S. and Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5:115–133.
- Medina Cortes, A., Saldaña Pérez, M., Sossa Azuela, H., Torres Ruiz, M., and Moreno Ibarra, M. (2021). An application of deep neural network for robbery evidence using face recognition approach. In Visvizi, A., Lytras, M. D., and Aljohani, N. R., editors, *Research and Innovation Forum 2020*, pages 23–36, Cham. Springer International Publishing.

- Murugesan, M. and Thilagamani, S. (2020). Efficient anomaly detection in surveillance videos based on multi layer perception recurrent neural network. *Microprocessors and Microsystems*, 79:103303.
- Nath, N. D. and Behzadan, A. H. (2019). Deep learning models for content-based retrieval of construction visual data. In *ASCE International Conference on Computing in Civil Engineering 2019*, pages 66–73. American Society of Civil Engineers Reston, VA.
- Öztürk, Ş. (2020). Stacked auto-encoder based tagging with deep features for content-based medical image retrieval. *Expert Systems with Applications*, 161:113693.
- Öztürk, Ş. (2021). Hash code generation using deep feature selection guided siamese network for content-based medical image retrieval. *Gazi University Journal of Science*, pages 1–1.
- Pérez-Hernández, F., Tabik, S., Lamas, A., Olmos, R., Fujita, H., and Herrera, F. (2020). Object detection binary classifiers methodology based on deep learning to identify small objects handled similarly: Application in video surveillance. *Knowledge-Based Systems*, 194:105590.
- Prathiba, T. and Kumari, R. S. S. (2021). Content based video retrieval system based on multimodal feature grouping by kfc clustering algorithm to promote human–computer interaction. *Journal of Ambient Intelligence and Humanized Computing*, 12, 6215-6229.
- Qiu, G. (2022). Challenges and opportunities of image and video retrieval. *Frontiers in Imaging*, 1:951934.
- Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. (2016). You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788.
- Rosenblatt, F. (1958). The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386.
- Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning representations by back-propagating errors. *nature*, 323(6088):533–536.

- Saito, K., Sohn, K., Zhang, X., Li, C. L., Lee, C. Y., Saenko, K., and Pfister, T. (2023). Pic2word: Mapping pictures to words for zero-shot composed image retrieval. *n Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 19305-19314)*.
- Shamna, N. and Aziz Musthafa, B. (2023). Feature extraction method using hog with ltp for content-based medical image retrieval. *International journal of electrical and computer engineering systems*, 14(3):267–275.
- Shen, Y., Feng, Y., Fang, B., Zhou, M., Kwong, S., and Qiang, B.-h. (2020). Dsrph: Deep semantic-aware ranking preserving hashing for efficient multi-label image retrieval. *Information Sciences*, 539:145–156.
- Shi, X., Sapkota, M., Xing, F., Liu, F., Cui, L., and Yang, L. (2018). Pairwise based deep ranking hashing for histopathology image classification and retrieval. *Pattern Recognition*, 81:14–22.
- Tahmasebzadeh, G., Kacupaj, E., Müller-Budack, E., Hakimov, S., Lehmann, J., and Ewerth, R. (2021). Geowine: Geolocation based wiki, image, news and event retrieval. *In Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (pp. 2565-2569)*.
- Tang, J., Gong, C., Guo, F., Yang, Z., and Wu, Z. (2022). Automatic geo-localization framework without gnss data. *IET Image Processing*, 16(8), 2180-2195.
- Tschannen, M., Kumar, M., Steiner, A., Zhai, X., Houlsby, N., and Beyer, L. (2023). Image captioners are scalable vision learners too. *arXiv preprint arXiv:2306.07915*.
- Tseng, C.-H., Hsieh, C.-C., Jwo, D.-J., Wu, J.-H., Sheu, R.-K., and Chen, L.-C. (2021). Person retrieval in video surveillance using deep learning–based instance segmentation. *Journal of Sensors*, 2021:1–12.
- Uma, M., Abirami, S., Ambika, M., Kavitha, M., Sureshkumar, S., and Kaviyaraj, R. (2023). A review on augmented reality and yolo. In *2023 4th International Conference on Smart Electronics and Communication (ICOSEC)*, pages 1025–1030. IEEE.

- Vasiliev, Y. (2020). *Natural language processing with Python and spaCy: A practical introduction*. No Starch Press.
- Vidal, M. T., Sánchez, A. L. L., Ayala, D. V., Beltrán, B., and Cardona, M. C. (2015). Primera aproximación de un sistema de recuperación de información booleano con expansión semántica de consultas. *Res. Comput. Sci.*, 99:55–63.
- Virmani, D. and Taneja, S. (2019). A text preprocessing approach for efficacious information retrieval. In *Smart Innovations in Communication and Computational Sciences: Proceedings of ICSICCS 2017, Volume 1*, pages 13–22. Springer.
- Vo, N., Jiang, L., Sun, C., Murphy, K., Li, L.-J., Fei-Fei, L., and Hays, J. (2019). Composing text and image for image retrieval-an empirical odyssey. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6439–6448.
- Wang, S., Wang, R., Yao, Z., Shan, S., and Chen, X. (2020). Cross-modal scene graph matching for relationship-aware image-text retrieval. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 1508–1517.
- Wu, H., Gao, Y., Guo, X., Al-Halah, Z., Rennie, S., Grauman, K., and Feris, R. (2021a). Fashion iq: A new dataset towards retrieving images by natural language feedback. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11307–11317.
- Wu, H., Gao, Y., Guo, X., Al-Halah, Z., Rennie, S., Grauman, K., and Feris, R. (2021b). Fashion iq: A new dataset towards retrieving images by natural language feedback. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11307–11317.
- Xin, J., Ye, F., Xia, Y., Luo, Y., and Chen, X. (2023). A new remote sensing image retrieval method based on cnn and yolo. *Journal of Internet Technology*, 24(2):233–242.
- Yue-Hei Ng, J., Yang, F., and Davis, L. S. (2015). Exploiting local features from deep networks for image retrieval. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 53–61.

- Zhang, Y., Jiang, H., Miura, Y., Manning, C. D., and Langlotz, C. P. (2022). Contrastive learning of medical visual representations from paired images and text. In *Machine Learning for Healthcare Conference*, pages 2–25. PMLR.
- Zhou, Q., Wang, C., Liu, P., Li, Q., Wang, Y., and Chen, S. (2016). Distribution entropy boosted vlad for image retrieval. *Entropy*, 18(8):311.