

SAHFER CONSULTORES, SA DE C.V. CAMPECHE

**"Recuperación de Información mediante Búsqueda Basada en Embeddings y
Búsqueda de Texto en Firebase"**

TESIS PARA TITULACIÓN INTEGRAL

INGENIERÍA EN SISTEMAS COMPUTACIONALES

**EGRESADO:
JESUS ANGEL BALAM SÁNCHEZ**

**DIRECTOR Y CO-DIRECTOR:
DR. Jose Luis Lira Turriza
DRA. Yaqueline Pech Huh**

Calkiní, Campeche, México 2025



Educación
Secretaría de Educación Pública



TECNOLÓGICO
NACIONAL DE MÉXICO



Instituto Tecnológico Superior de Calkiní
Subdirección de Planeación y Desarrollo

Instituto Tecnológico Superior de Calkiní
Departamento de Administración y
Apoyo al Estudiante

Calkiní, Campeche 07/03/2025
P/03/25/620

INGENIERÍA EN SISTEMAS COMPUTACIONALES

CONSTANCIA DE NO INCONVENIENCIA

MODALIDAD: TESIS

C. JESUS ANGEL BALAM SANCHEZ 7294
PASANTE DE LA CARRERA DE
INGENIERÍA EN SISTEMAS COMPUTACIONALES.
PRESENTE

Con relación a lo establecido en el Reglamento de Estudios de Licenciatura, capítulo VII, fracción I, vigente en este Instituto, y una vez emitido el fallo favorable por el director JOSÉ LUIS LIRA TURRIZA, y el co-director YAQUELINE PECH HUH, respectivamente, este Instituto consta de no tener inconveniencia para la realización del acto protocolario de titulación integral de su trabajo profesional titulado: **"RECUPERACIÓN DE INFORMACIÓN MEDIANTE BÚSQUEDA BASADA EN EMBEDDINGS Y BÚSQUEDA DE TEXTO EN FIREBASE."**

ATENTAMENTE

Excelencia en Educación Tecnológica®
Sabiduría. Fortaleza de Nuestra Cultura®



INSTITUTO TECNOLÓGICO SUPERIOR
DE CALKINÍ

DEPARTAMENTO DE ADMINISTRACIÓN
ESCOLAR Y APOYO A
ESTUDIANTES

CLAVE: 04MSU0013P
CALKINÍ, CAMPECHE, MÉXICO.

Dr. Jorge Alfonso Hau Puc.
Departamento de Administración y
Apoyo al Estudiante.

C.c.p. Archivo



2025
Año de
La Mujer
Indígena

Av. Ah Canul S/N por Carretera Federal, Calkiní, Campeche,
C.P. 24900 Tel. 996-8134870,
e-mail: se_dcalkini@tecnm.mx | itescam.edu.mx



Dedicatoria.

“Acepta las cosas que no puedas cambiar,
ten el valor de cambiar las cosas que puedes
y ten la sabiduría para conocer la diferencia.”

-The Flashpoint

Dedico este trabajo a una persona muy especial en mi vida a mi mamá, Irma Del Rosario Sánchez López. Eres una mujer fuerte y valiente, que ha enfrentado grandes desafíos, pero siempre estuvo a mi lado. Gracias, mamá, por tu apoyo incondicional, por tu amor y por todo lo que has hecho por mí. Este logro es gracias a ti, Te amo mamá

Agradecimientos.

"Porque yo sé los planes que tengo para ustedes, dice el Señor, planes de bienestar y no de calamidad, a fin de darles un futuro y una esperanza." Jeremías 29:11

Agradezco a Dios por permitirme llegar hasta aquí. Sin Él, no podría haberlo logrado. Sé que soy una persona diferente a los demás, pero siempre estuviste conmigo. Me diste muchas lecciones de vida, salud, y la bendición de tener a mi lado a mis familiares y amigos.

De igual forma, agradezco a mis padres, Irma y Olegario. Les doy las gracias por toda la ayuda que me han brindado. Gracias, papá Ole, por llegar a nuestras vidas y, sobre todo, por cuidar a mamá mientras yo estudiaba.

Agradezco a mis hermanas, Chelsy y Carmen, por estar a mi lado. Sé que nunca se los digo, pero de verdad las quiero mucho. Pasamos por muchas cosas juntos, pero a pesar de todo, las quiero y sé que ustedes también pueden superar cualquier obstáculo.

A mis abuelos: Adonay Balam, Carmen Medina, Lorenzo Sánchez, Carmen López, les agradezco todo el apoyo. Sé que soy un nieto algo difícil, pero valoro todas sus palabras de sabiduría y las experiencias de vida que compartieron conmigo. Muchas gracias.

Quiero agradecer especialmente a mis tías, Fátima Balam y Judith Sánchez. Son mujeres fuertes que siempre luchan por salir adelante. De ustedes aprendí que la familia nunca se abandona; pase lo que pase, siempre estarán ahí.

Agradezco también a los DRS. José Luis Lira Turriza, Yaqueline Pech Huh y José Manuel Lira Turriza, por haberme permitido trabajar con ustedes. Gracias por confiar en mí, por sus sabias palabras y por todo el asesoramiento durante las clases y esta etapa de residencia.

A mis hermanos adoptivos, Ángela y Nijel, los conocí en una de las mejores etapas de mi vida. Gracias por cuidarme. Sé que ustedes también están triunfando, y eso me hace muy feliz. Los quiero mucho.

A mi mejor amigo, Fernando Magdiel Chi Canul, gracias por todo el apoyo que me has dado desde que nos conocimos. Sé que hemos pasado por muchas cosas juntos y estoy seguro de que seguiremos enfrentándolas. Te admiro mucho y te agradezco por todo.

Dios puso a muchas personas en mi vida, y por eso estoy agradecido. A todos los amigos y familiares que me cuidaron y confiaron en mí: Cinthia, Candy, Martín, Pablo Che, Yair, Jonathan, Eduardo, Julian, Jordi, al padrino Pablo, a mamá Mechis, y a todos los integrantes de la familia Navarro Delgado: muchas gracias por su apoyo. A mis mamás adoptivas, mami Magui y mami Flor, gracias de corazón por todo el apoyo que me brindaron estaré eternamente agradecido con ustedes.

Finalmente, quiero agradecer a la Fundación José Ortiz Ávila por darme la oportunidad de ser uno de sus becarios. Gracias por contribuir a mi formación académica y por todo el apoyo brindado durante mi estudio.

Indice

Capítulo 1 Problema De Investigación.....	10
Planteamiento del problema.....	10
Objetivos.....	12
Objetivo general.....	12
Objetivos específicos	12
Capítulo 2 Marco Teórico.....	13
Antecedentes.....	13
Hipótesis.....	20
Hipótesis específicas:	20
Capítulo 3 Metodología.....	21
Diseño de investigación.....	21
Población y Muestra	21
Instrumentos de Recolección de Datos.....	22
Procedimiento.....	22
Análisis de Datos.....	23
Procedimiento.....	23
Capitulo 4 Resultados y Discusión.....	24
Resultados.....	24
Resultados de la Búsqueda de precisión simple.....	26
Tabla de Resultados de la Precisión en Búsqueda Simple.....	26
Desglose del Procedimiento Paso a Paso Simple.....	27
Resultados del Tiempo de Búsqueda Simple.....	29
Tabla de Resultados del Tiempo de Búsqueda Simple.....	29
Graficas comparativas de resultados simples.....	32
Resultados de la Precisión en Búsqueda Compleja	34
Resultados del Tiempo de Búsqueda en Consultas Complejas.....	36
Graficas comparativas de resultados Complejo	39
Conclusiones y recomendaciones.....	42
Conclusiones.....	42
Recomendaciones	44
Anexos.....	47

Índice De Tablas.

Tabla 1 Resultados de la precisión..... 26

Tabla 2 Resultados del Tiempo de Búsqueda Simple 29

Tabla 3 Resultados de la precisión de Búsqueda Compleja..... 34

Tabla 4 Resultados del Tiempo de Búsqueda 36

Resumen.

La recuperación de información es un componente esencial en el manejo de grandes volúmenes de datos. Esta investigación compara dos métodos de búsqueda en Firebase Firestore: la búsqueda basada en palabras clave y la búsqueda basada en embeddings. Mientras que el primer método se basa en la coincidencia exacta de términos, el segundo emplea representaciones vectoriales para identificar similitudes semánticas entre consultas y documentos.

El estudio se desarrolló mediante un diseño experimental cuantitativo y comparativo, evaluando ambos enfoques en términos de precisión y tiempo de respuesta. Se realizaron pruebas con consultas simples y complejas, midiendo su desempeño mediante métricas de similitud y pruebas estadísticas.

Los resultados muestran que la búsqueda basada en embeddings mejora significativamente la precisión de los resultados en comparación con la búsqueda por palabras clave. Sin embargo, este beneficio conlleva un mayor tiempo de procesamiento. Por el contrario, la búsqueda por palabras clave es más rápida, pero menos eficiente en la recuperación semántica de la información.

En conclusión, el uso de embeddings representa una alternativa viable para mejorar la relevancia de los resultados en sistemas de recuperación de información. No obstante, su aplicación debe considerarse en función de los requerimientos de tiempo de respuesta y disponibilidad de recursos computacionales.

Abstract.

Information retrieval is an essential component in handling large volumes of data. This research compares two search methods in Firebase Firestore: keyword-based search and embedding-based search. While the first method is based on exact term matching, the second uses vector representations to identify semantic similarities between queries and documents.

The study was developed using a quantitative and comparative experimental design, evaluating both approaches in terms of accuracy and response time. Tests were performed with simple and complex queries, measuring their performance using similarity metrics and statistical tests.

The results show that embedding-based search significantly improves the accuracy of results compared to keyword search. However, this benefit entails a longer processing time. In contrast, keyword search is faster, but less efficient in the semantic retrieval of information.

In conclusion, the use of embeddings represents a viable alternative to improve the relevance of results in information retrieval systems. However, its application must be considered in terms of response time requirements and availability of computing resources.

Introducción.

En la era digital, el acceso rápido y preciso a la información es fundamental para el desarrollo de sistemas eficientes en diversas aplicaciones. La recuperación de información se ha convertido en un área de interés creciente debido a la explosión de datos y la necesidad de obtener resultados relevantes en tiempos óptimos. Los motores de búsqueda y las bases de datos juegan un papel esencial en este proceso, pero los métodos tradicionales basados en búsqueda por palabras clave presentan limitaciones significativas, especialmente cuando los usuarios formulan consultas con sinónimos o frases con significados similares pero diferentes términos.

Ante esta problemática, han surgido nuevas técnicas para mejorar la recuperación de información, entre ellas la búsqueda basada en embeddings. Esta metodología, fundamentada en el uso de representaciones vectoriales de palabras y documentos, permite capturar relaciones semánticas entre términos, ofreciendo resultados más precisos y contextualizados. A pesar de sus ventajas en precisión, este enfoque suele requerir un mayor consumo computacional y tiempos de respuesta más prolongados en comparación con las búsquedas tradicionales por palabras clave.

En este contexto, Firebase Firestore es una base de datos ampliamente utilizada por su rapidez y escalabilidad en aplicaciones web y móviles. Sin embargo, su búsqueda nativa se basa en coincidencias exactas de texto, lo que puede generar limitaciones en la recuperación de información. La integración de embeddings en Firestore podría representar una alternativa prometedora para mejorar la precisión de las búsquedas, pero su impacto en el tiempo de respuesta y el rendimiento del sistema no ha sido explorado a profundidad.

El objetivo de esta investigación es comparar el rendimiento y la precisión de la recuperación de información en Firestore mediante búsqueda basada en embeddings y búsqueda

por palabras clave. Se evaluará la eficiencia de ambos métodos considerando métricas de velocidad y relevancia de los resultados, con el propósito de determinar cuál de ellos es más adecuado para diferentes escenarios de uso.

Los resultados de este estudio proporcionarán información clave para el diseño y optimización de sistemas de recuperación de información, contribuyendo al desarrollo de soluciones más eficientes y precisas. Asimismo, permitirán identificar las condiciones en las que cada enfoque es más ventajoso, facilitando la toma de decisiones en la implementación de estrategias de búsqueda en bases de datos modernas como Firestore.

Capítulo 1 Problema De Investigación.

Planteamiento del problema.

En la actualidad, la recuperación eficiente de información es un factor clave en el desempeño de sistemas que manejan grandes volúmenes de datos. Firestore, como base de datos, permite almacenar y consultar información de manera rápida, pero su método de búsqueda tradicional basado en palabras clave puede presentar limitaciones en términos de precisión, ya que solo recupera información cuando hay coincidencias exactas en los términos de búsqueda. Esto puede generar resultados irrelevantes o incompletos cuando los usuarios utilizan sinónimos o frases con un significado similar pero diferentes palabras. (Apilyca, 2023)

Para abordar esta problemática, ha surgido la búsqueda basada en embeddings, un método que permite encontrar información a partir de la similitud semántica en lugar de depender exclusivamente de coincidencias de texto. Este enfoque puede mejorar la precisión de la recuperación de información al identificar términos conceptualmente relacionados en lugar de solo palabras exactas. Sin embargo, el uso de embeddings puede incrementar el tiempo de procesamiento y el consumo de recursos computacionales, lo que podría afectar la eficiencia del sistema en aplicaciones con grandes volúmenes de datos.

A pesar del potencial de los embeddings en la recuperación de información, no está claro qué método es más eficiente en términos de precisión y tiempo de respuesta dentro de Firestore. Mientras que la búsqueda por palabras clave es generalmente más rápida, puede comprometer la relevancia de los resultados. Por otro lado, la búsqueda con embeddings podría ofrecer resultados más precisos, pero a costa de un mayor tiempo de procesamiento. (MYSCALE, 2024)

Ante esta incertidumbre, se vuelve necesario realizar una comparación entre estos dos métodos para determinar cuál proporciona una mejor recuperación de información en Firestore. Esta investigación tiene como objetivo evaluar el rendimiento y la precisión de la búsqueda basada en embeddings en comparación con la búsqueda por palabras clave. Para ello, se analizarán tiempos de respuesta, relevancia de los resultados.

A partir de este estudio, se busca responder las siguientes preguntas de investigación:

¿La búsqueda basada en embeddings en Firestore proporciona resultados más relevantes que la búsqueda por palabras clave?

¿La búsqueda con embeddings en firestore disminuye el tiempo de respuesta en comparación con la búsqueda por palabras clave?

Los resultados obtenidos permitirán comprender cuál de los dos métodos es más adecuado para la recuperación de información en diferentes escenarios, proporcionando información clave para el diseño y optimización de sistemas que dependen de consultas eficientes en bases de datos como firestore.

Objetivos.

Objetivo general

- Comparar el rendimiento y la precisión en la recuperación de información mediante búsqueda basada en embeddings y búsqueda de texto en firestore, evaluando su eficiencia en términos de velocidad y relevancia de los resultados.

Objetivos específicos

- Medir y comparar el tiempo de respuesta entre la búsqueda basada en embeddings y la búsqueda por palabras clave en firestore.
- Evaluar la precisión de ambos métodos mediante pruebas de relevancia y similitud de los resultados obtenidos.
- Proponer mejoras en la búsqueda de información con base en los resultados obtenidos de la comparación.

Capítulo 2 Marco Teórico.

Antecedentes.

La recuperación de información ha evolucionado significativamente en los últimos años debido al crecimiento exponencial de datos y la necesidad de acceder a información relevante de manera eficiente. Tradicionalmente, los sistemas de búsqueda han dependido de la coincidencia de palabras clave, lo cual, aunque efectivo en ciertos contextos, presenta limitaciones significativas en términos de precisión semántica. Este enfoque se basa en la coincidencia exacta de términos, lo que puede llevar a resultados irrelevantes cuando los usuarios utilizan sinónimos o frases con significados similares, pero palabras diferentes.

Tradicionalmente, los sistemas de recuperación de información empleaban modelos vectoriales, representando documentos y consultas como vectores en un espacio multidimensional. Este enfoque permitía medir la similitud entre documentos y consultas mediante cálculos de proximidad. Sin embargo, la representación efectiva de consultas, especialmente debido a su limitada capacidad descriptiva, ha sido un desafío constante.

Para abordar esta limitación, (Alfredo Silva, 2021) propusieron una estrategia basada en datos para combinar embeddings de palabras. Utilizando combinaciones ponderadas por frecuencia inversa de documentos (Idf), lograron enriquecer la representación de las consultas, mejorando la precisión en la recuperación de información. Los resultados experimentales en conjuntos de datos de referencia demostraron que esta metodología supera a las representaciones promedio de embeddings previamente propuestas en la literatura.

Los embeddings son representaciones numéricas de palabras o frases en un espacio vectorial de alta dimensión, capturando relaciones semánticas entre términos. Esta técnica ha sido

esencial para mejorar tareas como la búsqueda de información, la agrupación de textos y las recomendaciones.

(Tolgahan Cakaloglu, 2019) exploraron cómo las representaciones de texto en espacios vectoriales pueden utilizarse para la recuperación eficiente de información en grandes bases de conocimiento. Entrenando modelos neuronales profundos específicamente para tareas de recuperación, demostraron mejoras significativas en la precisión al utilizar embeddings textuales

La aplicación de embeddings ha transformado significativamente los sistemas de recuperación de información. Estos embeddings son representaciones numéricas de datos que capturan el significado y las relaciones entre palabras, frases o documentos completos, permitiendo a las máquinas procesar información compleja de manera más efectiva (Intersistemas, 2024)

En el contexto de la recuperación de información, los embeddings permiten mapear consultas y documentos a un espacio vectorial donde las similitudes semánticas se reflejan en proximidades espaciales. Esto facilita la identificación de documentos relevantes incluso cuando no comparten palabras clave exactas con la consulta del usuario (Ojeda, s.f.)

Un enfoque destacado que utiliza embeddings es la Generación Mejorada por Recuperación (Retrieval-Augmented Generation, RAG). Este método combina modelos de generación de lenguaje con mecanismos de recuperación de información, almacenando documentos como embeddings en una base de datos vectorial. Cuando se realiza una consulta, el sistema recupera los embeddings más relevantes y los utiliza para generar respuestas más precisas y contextuales (google, cloud.google.com, s.f.)

La integración de embeddings en sistemas de recuperación de información no solo mejora la precisión y relevancia de los resultados, sino que también permite manejar consultas más

complejas y entender mejor el contexto y la intención del usuario. Esta evolución representa un avance significativo respecto a los métodos tradicionales basados únicamente en coincidencias de palabras clave.

Embedding

La búsqueda por embeddings se utiliza para representaciones numéricas de datos, compara conceptos y encuentra similitudes. Los embeddings son representaciones matemáticas de datos, comúnmente utilizados en este caso en el procesamiento de lenguaje natural (PLN). Un Embedding convierte una palabra, frase o incluso un párrafo en vectores. La ventaja grande de trabajar con ellos es que las máquinas entienden relaciones y similitudes entre diferentes términos de manera más eficiente que usando representaciones tradicionales.

Los textos, originalmente en formato no estructurado, se convierten en vectores que codifican su significado semántico, ubicándolos en un espacio multidimensional donde las distancias reflejan similitudes (por ejemplo, "gato" y "perro" estarán más cerca que "gato" y "computadora"), y estos embeddings se generan entrenando modelos en grandes conjuntos de datos para capturar patrones y relaciones contextuales entre palabras. (IBM, [ibm.com](https://www.ibm.com), 2024)

Aplicaciones de los embeddings

Búsqueda semántica: Encuentran información relevante al comparar similitudes entre consultas de usuario y una base de datos, independientemente de las palabras exactas utilizadas.

Clasificación de texto: Ayudan a agrupar documentos o textos similares en categorías.

Recomendaciones: Se emplean en sistemas de recomendación para sugerir contenido relacionado según patrones de texto.

Firestore

Para entender Firestore, es importante saber que se trata de una plataforma de desarrollo de aplicaciones que ofrece una amplia variedad de herramientas y servicios. Entre ellos se incluyen el desarrollo y la gestión de aplicaciones móviles y web, así como el almacenamiento de datos en tiempo real. Firestore es una base de datos moderna y flexible, organizada en colecciones y documentos, lo que facilita su uso y la hace más potente en términos de consultas y escalabilidad.

Se trata de una base de datos NoSQL, escalable y flexible, diseñada para almacenar y sincronizar datos en tiempo real en aplicaciones móviles, web y servidores. Una de sus principales características es su estructura de datos jerárquica, lo que significa que organiza la información en colecciones y documentos.

Las colecciones son contenedores de documentos y pueden compararse con las tablas en una base de datos relacional, aunque Firestore no es relacional. Por otro lado, los documentos son los elementos dentro de las colecciones, que contienen datos, generalmente en formato JSON. Un documento puede tener campos como texto, números, valores booleanos, arrays, mapas, entre otros. (google, firestore google , 2024)

Algoritmos de Búsqueda

Un algoritmo de búsqueda es un procedimiento diseñado para localizar un elemento específico dentro de un conjunto de datos. Este proceso es esencial en informática, ya que permite acceder y procesar información de manera eficiente, independientemente de la naturaleza de los datos, que pueden ser números, texto o incluso imágenes. Según el objetivo y la estructura de los

datos, los algoritmos de búsqueda pueden variar desde métodos básicos hasta técnicas avanzadas que emplean modelos de inteligencia artificial.

Los métodos básicos incluyen la búsqueda secuencial, que examina cada elemento de la colección uno a uno hasta encontrar el deseado. Este enfoque es simple y no requiere que los datos estén ordenados, pero puede ser poco eficiente para grandes volúmenes de información, ya que su complejidad es proporcional al tamaño del conjunto de datos (Box, 2024)

Por otro lado, los algoritmos más avanzados, como aquellos basados en modelos de aprendizaje profundo, utilizan redes neuronales para interpretar patrones complejos en los datos. Estas técnicas son capaces de realizar búsquedas semánticas, comprendiendo el significado contextual detrás de una consulta en lugar de simplemente buscar coincidencias exactas. Esto es particularmente útil en aplicaciones como motores de búsqueda, sistemas de recomendación y reconocimiento de voz, donde la precisión y la relevancia de los resultados son cruciales (IBM, IBM, 2024)

Mecanismos de Búsqueda

Los mecanismos de búsqueda más comunes incluyen:

- Búsqueda Secuencial: Examina cada elemento de la colección hasta encontrar el resultado deseado.
- Búsqueda Binaria: Utiliza la división iterativa para localizar elementos en estructuras de datos ordenadas.
- Búsqueda Basada en Similitud: Evalúa la proximidad entre representaciones de datos, como vectores numéricos.

- Búsqueda Semántica: Un método avanzado que interpreta el significado detrás de una consulta, utilizando representaciones vectoriales (embeddings).

Búsqueda Semántica con Embeddings

En el sistema desarrollado, se implementa una búsqueda semántica utilizando embeddings generados por el modelo text-embedding-ada-002 de OpenAI, en combinación con la base de datos Firestore de Firebase. Este enfoque permite buscar documentos no solo basándose en palabras exactas, sino también en el significado del texto.

El proceso comienza generando un embedding, que es una representación vectorial del texto de la consulta. Para ello, se emplea el modelo de OpenAI, que transforma las palabras en un vector de alta dimensión que captura su significado semántico. Posteriormente, el sistema accede a Firestore, donde cada documento almacenado contiene su propio embedding, generado previamente.

El núcleo del algoritmo consiste en comparar el embedding de la consulta con los embeddings de los documentos mediante la métrica de similitud coseno. Esta métrica evalúa qué tan cercanos son dos vectores, devolviendo un valor entre -1 y 1, donde un valor más alto indica mayor similitud. Matemáticamente, se calcula dividiendo el producto escalar de los vectores entre el producto de sus normas. Así, el sistema identifica los documentos más relacionados semánticamente con la consulta.

Finalmente, los documentos se ordenan en función de su similitud con la consulta y se devuelve un conjunto de resultados limitado. Este enfoque permite encontrar documentos relevantes incluso cuando las palabras usadas en la consulta y en los documentos no coinciden exactamente, priorizando la relevancia semántica sobre la literalidad.

Hipótesis.

"El uso de embeddings en firestore mejora la recuperación de información en términos de precisión y tiempo de respuesta en comparación con la búsqueda por palabras clave."

Hipótesis específicas:

Hipótesis sobre la precisión:

- H_0 (hipótesis nula): No hay diferencia significativa en la precisión entre la búsqueda basada en embeddings y la búsqueda por palabras clave.
- H_A (hipótesis alternativa): La búsqueda basada en embeddings tiene una precisión significativamente mayor.

Hipótesis sobre el tiempo de respuesta:

- H_0 : No hay diferencia significativa en el tiempo de respuesta entre ambos métodos.
- H_A : El método basado en embeddings reduce significativamente el tiempo de respuesta en comparación con la búsqueda por palabras clave.

Capítulo 3 Metodología.

Diseño de investigación.

Este estudio adopta un diseño experimental Cuantitativo y Comparativo, con el propósito de evaluar la eficacia de dos métodos de recuperación de información en firestore: búsqueda basada en palabras clave y búsqueda basada en embeddings. La investigación se centrará en la comparación de precisión de los resultados y tiempo de respuesta.

El diseño experimental permite medir el desempeño de cada método bajo condiciones controladas, utilizando un conjunto de consultas predeterminadas en firestore. Se analizarán los tiempos de respuesta y la precisión en los resultados utilizando métricas de similitud.

Para determinar diferencias significativas entre los métodos, se aplicará la prueba Cuantitativa y Comparativo para muestras independientes, con el fin de establecer si existen variaciones relevantes en la precisión y el tiempo de respuesta.

Población y Muestra

- Población: Consultas realizadas en Firebase para la recuperación de información en firestore.
- Muestra: Se aplicó la regla del 1%, basada en un promedio de 18,000 consultas diarias en firestore. De este total, se seleccionó una muestra representativa de 180 consultas.

Distribución de la muestra:

- 90 consultas simples: Consultas que contienen un solo término.

- 90 consultas complejas: Consultas con múltiples términos o filtros.

Esta selección permite un balance entre representatividad y viabilidad, asegurando una comparación eficiente sin la necesidad de analizar la totalidad de las consultas diarias.

Instrumentos de Recolección de Datos.

Para medir el desempeño de cada método, se emplearán los siguientes instrumentos:

- Registro de tiempos de respuesta: Se capturarán los tiempos en milisegundos de cada consulta en firestore.
- Cálculo de precisión mediante similitud coseno: Se empleará el modelo text-embedding-ada-002 de OpenAI para evaluar el grado de similitud entre la consulta y los resultados obtenidos.
- Base de datos estructurada para almacenamiento de resultados: Para registrar y comparar los tiempos de respuesta y la precisión de cada método.

Procedimiento.

El estudio se llevará a cabo en tres diferentes fases: en la primera fase, preparación de los datos, se seleccionará un conjunto de documentos de prueba en firestore y se diseñarán 180 consultas, divididas en 90 simples y 90 complejas. En la segunda fase, ejecución de las búsquedas, se realizarán las 180 consultas utilizando búsqueda por palabras. Se repetirán las mismas 180 consultas con búsqueda basada en embeddings, registrando los tiempos de respuesta para cada método. Finalmente, en la fase etapa, medición y análisis, se calculará la precisión de los resultados mediante la métrica de similitud coseno, se compararán los tiempos de respuesta entre ambos

métodos y se aplicará la prueba t de Student para determinar si las diferencias en tiempo y precisión son estadísticamente significativas. Este diseño permite una comparación rigurosa y objetiva entre los dos métodos de búsqueda, evaluando su eficiencia y precisión en diferentes tipos de consultas.

Análisis de Datos.

- Cálculo del tiempo medio de respuesta y análisis de su variabilidad en cada método.
- Comparación de la precisión mediante la métrica de similitud coseno.
- Las Pruebas aplicadas fueron de estadísticas para comparar dos métodos de búsqueda (Búsqueda de Texto vs. Embeddings) en términos de precisión (similitud) y tiempo de respuesta.

Procedimiento.

El estudio se llevará a cabo en tres diferentes fases: en la primera fase, preparación de los datos, se seleccionará un conjunto de documentos de prueba en firestore y se diseñarán 180 consultas, divididas en 90 simples y 90 complejas. En la segunda fase, ejecución de las búsquedas, se realizarán las 180 consultas utilizando búsqueda por palabras. Se repetirán las mismas 180 consultas con búsqueda basada en embeddings, registrando los tiempos de respuesta para cada método. Finalmente, en la fase etapa, medición y análisis, se calculará la precisión de los resultados mediante la métrica de similitud coseno, se compararán los tiempos de respuesta entre ambos métodos y se aplicará la prueba t de Student para determinar si las diferencias en tiempo y precisión son estadísticamente significativas. Este diseño permite una comparación rigurosa y objetiva entre los dos métodos de búsqueda, evaluando su eficiencia y precisión en diferentes tipos de consultas.

Capítulo 4 Resultados y Discusión.

Resultados.

I. Prueba Z para Dos Muestras Grandes

La prueba Z para dos muestras independientes es un procedimiento estadístico utilizado para comparar las medias de dos poblaciones cuando el tamaño muestral es suficientemente grande ($n > 30$) y se desconocen las varianzas poblacionales (xlstat, n.d.). Esta prueba permitió evaluar si las diferencias observadas en la precisión y el tiempo de búsqueda entre la Búsqueda basada en Embeddings y la Búsqueda de Texto en Firebase eran estadísticamente significativas o si pudieran ser atribuidas al azar.

Fórmula aplicada.

La ecuación utilizada para el cálculo de la estadística Z es:

$$z = \frac{(\bar{x}_1 - \bar{x}_2)}{\sqrt{\frac{\sigma_1^2}{n_1} + \frac{\sigma_2^2}{n_2}}}$$

Donde:

- $\bar{x}_1$ y $\bar{x}_2$ representan las medias de cada método de búsqueda.
- σ_1 y σ_2 son las desviaciones estándar de cada método.
- n_1 y n_2 son los tamaños de muestra.

II. Cálculo del p-Valor.

El p-valor es una métrica fundamental en estadística inferencial, ya que permite evaluar la significancia estadística de una diferencia entre dos muestras. Representa la probabilidad de obtener los resultados observados si la hipótesis nula (H_0) es verdadera (Blog, 2019)

En este estudio, el p-valor se calculó a partir de la función de distribución normal estándar $p = 1 - P(Z)$, donde $P(Z)$ es la función de distribución acumulada de la normal estándar.

Interpretación del p-Valor

- Si $p < 0.05 \rightarrow$ Se rechaza la hipótesis nula H_0 , lo que indica que la diferencia entre los métodos es significativa.
- Si $p > 0.05 \rightarrow$ No se rechaza H_0 , lo que sugiere que no hay evidencia suficiente para afirmar que un método es mejor que el otro.

III. Prueba de Normalidad (Aproximación por el Teorema del Límite Central)

Dado que el tamaño de la muestra era $n = 90$ en cada grupo, no fue necesario realizar una prueba de normalidad explícita. En estadística inferencial, cuando el tamaño muestral es grande ($n > 30$), el Teorema del Límite Central establece que la distribución de la media muestral se aproxima a una distribución normal, independientemente de la forma original de los datos. Por esta razón, la prueba Z fue válida sin necesidad de realizar pruebas adicionales de normalidad como Kolmogorov-Smirnov o Shapiro-Wilk. (openstax, n.d.).

Resultados de la Búsqueda de precisión simple

En esta sección se presentan los resultados obtenidos al evaluar la recuperación de información en consultas simples mediante búsqueda basada en Embeddings y Búsqueda de Texto en firestore. Se analizaron dos métricas clave: precisión de búsqueda (similitud) y tiempo de respuesta, aplicando pruebas estadísticas para determinar si las diferencias observadas entre ambos métodos son significativas.

Tabla de Resultados de la Precisión en Búsqueda Simple.

Tabla 1 Resultados de la precisión

Símbolo	Métrica	Fórmula en Excel
X1	Media de la Búsqueda de Texto en firestore	28.28433333
X2	Media de la Búsqueda basada en Embeddings	0.85111111
σ_1	Desviación estándar de la Búsqueda de Texto en firestore	18.72182903
σ_2	Desviación estándar de la Búsqueda basada en Embeddings	0.047126526
n1	Tamaño de la muestra	90
n2	Tamaño de la muestra	90

X1-X2	Diferencia de Medias	27.43322222
σ_1^2	Varianza de la Búsqueda de Texto en firestore	350.5068823
σ_2^2	varianza de la Búsqueda basada en Embeddings	0.002220909
$\frac{\sigma_1^2}{n_1}$	Varianza de la Búsqueda de Texto en firestore dividida entre el tamaño de la muestra	3.894520915
$\frac{\sigma_2^2}{n_2}$	Varianza de la Búsqueda basada en Embeddings dividida entre el tamaño de la muestra	2.46768E-05
$\sqrt{\frac{\sigma_1^2}{n_1} + \frac{\sigma_2^2}{n_2}}$	Raíz cuadrada de la suma de varianzas divididas	1.973460309
Z	Estadístico de prueba Z	13.90107624
p	p-valor obtenido	0

Desglose del Procedimiento Paso a Paso Simple

Cálculo de la Diferencia de Medias:

$$x_1 - x_2 = 28.28 - 0.85 = 27.43$$

Cálculo de la Varianza:

$$\sigma_1^2 = (18.72)^2 = 350.51, \sigma_2^2 = (0.047)^2 = 0.0022$$

Dividir cada Varianza entre el Tamaño de la Muestra:

$$\frac{\sigma_1^2}{n_1} = \frac{350.51}{90} = 3.89, \frac{\sigma_2^2}{n_2} = \frac{0.0022}{90} = 2.46E - 05$$

Suma de las Varianzas Divididas y Raíz Cuadrada:

$$\sqrt{3.89 + 2.46E - 05} = 1.9734$$

Cálculo de la Estadística Z:

$$Z = \frac{27.43}{1.9734} = 13.90$$

Cálculo del p-valor:

Dado que $Z = 13.90$, se obtiene $p = 0.00$, lo que indica una diferencia estadísticamente significativa.

Interpretación de los Resultados

Se observa que la media de similitud obtenida en la Búsqueda basada en Embeddings es 0.85, mientras que la media en Búsqueda de Texto en firestore es 28.28, lo que indica que la similitud en la primera es considerablemente superior. La prueba de hipótesis $Z = 13.90$ con un p-valor de 0.00 confirma que esta diferencia es estadísticamente significativa, lo que permite rechazar la hipótesis nula y concluir que Embeddings proporciona una mayor precisión en la búsqueda de información en firestore.

Dado que la varianza en la Búsqueda de Texto en firestore es mayor (350.51 frente a 0.0022 en Embeddings), también se observa una mayor dispersión en los valores obtenidos para la

similitud en firestore, lo que puede deberse a una menor capacidad de este método para representar relaciones semánticas en los datos de entrada.

Resultados del Tiempo de Búsqueda Simple.

En esta sección se presentan los resultados obtenidos al evaluar el tiempo de respuesta en la recuperación de información mediante Embeddings y Búsqueda de Texto en firestore. El tiempo de respuesta se mide en segundos y representa el tiempo transcurrido desde que se realiza la consulta hasta que se obtiene el resultado.

Tabla de Resultados del Tiempo de Búsqueda Simple.

Tabla 2 Resultados del Tiempo de Búsqueda Simple

Símbolo	Métrica	Fórmula en Excel
X1	Media del tiempo de búsqueda de Texto	0.000838889
X2	Media del tiempo de búsqueda en Embeddings	3.311444444
σ_1	Desviación estándar del tiempo de búsqueda de Texto	0.000547863
σ_2	Desviación estándar del tiempo en Embeddings	0.414715266
N1	Tamaño de la muestra de búsqueda de Texto	90

N2	Tamaño de la muestra Embeddings	90
$X_1 - X_2$	Calcular la Diferencia de Medias	-3.310605556
σ_1^2	Varianza de la búsqueda de Texto	3.00154E-07
σ_2^2	Varianza de Embeddings	0.171988752
$\frac{\sigma_1^2}{n_1}$	Dividir Cada Varianza Entre el Tamaño de la Muestra	3.33505E-09
$\frac{\sigma_2^2}{n_2}$	Varianza del tiempo en Embeddings dividida entre el tamaño de la muestra	0.001910986
$\sqrt{\frac{\sigma_1^2}{n_1} + \frac{\sigma_2^2}{n_2}}$	Estadístico de prueba Z	0.043714865
Z	Calcular la Raíz Cuadrada	-75.7318024
P	p-valor obtenido	1

Cálculo de la Diferencia de Medias:

$$x_1 - x_2 = 0.0008 - 3.3114 = -3.3106$$

Cálculo de la Varianza:

$$\sigma_1^2 = (0.0005)^2 = 3.00154E - 07, \quad \sigma_2^2 = (0.4147)^2 = 0.1719$$

Dividir cada Varianza entre el Tamaño de la Muestra:

$$\frac{\sigma_1^2}{n_1} = \frac{3.00154E - 07}{90} = 3.33505E - 09, \frac{\sigma_2^2}{n_2} = \frac{0.1719}{90} = 0.0019$$

Suma de las Varianzas Divididas y Raíz Cuadrada:

$$\sqrt{3.33505E - 09 + 0.0019} = 0.0437$$

Cálculo de la Estadística Z:

$$z = \frac{-3.3106}{0.0437} = -75.73$$

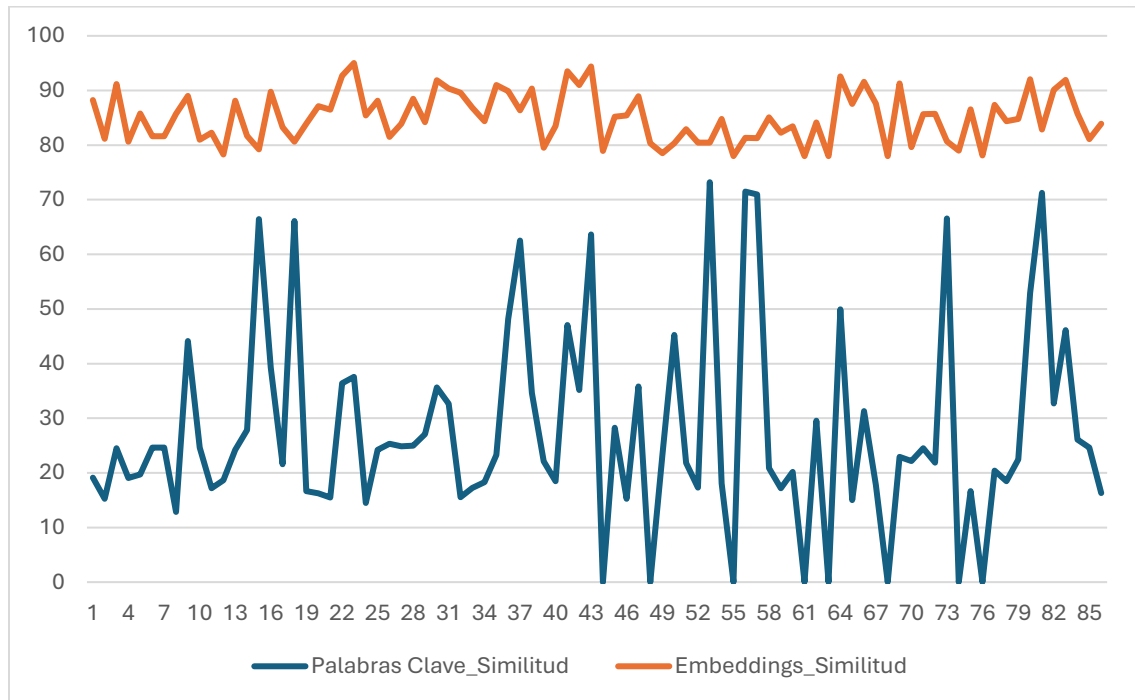
Interpretación de los Resultados

Se observa que la Búsqueda de Texto en firestore tiene un tiempo de respuesta promedio de 0.0008 segundos, mientras que el método basado en Embeddings presenta un tiempo de 3.3114 segundos en promedio. La prueba de hipótesis $Z = -75.73$ con un p-valor de 1.00 confirma que esta diferencia es estadísticamente significativa, lo que permite concluir que la Búsqueda de Texto en firestore es considerablemente más rápida que la búsqueda basada en Embeddings.

La varianza del tiempo en Embeddings es significativamente mayor (0.1719 frente a 3.00154E-07 en palabras clave), lo que sugiere que el método de Embeddings no solo es más lento, sino también más inconsistente en términos de tiempo de respuesta.

Graficas comparativas de resultados simples.

Figura 1 Comparación de Similitud Simple.



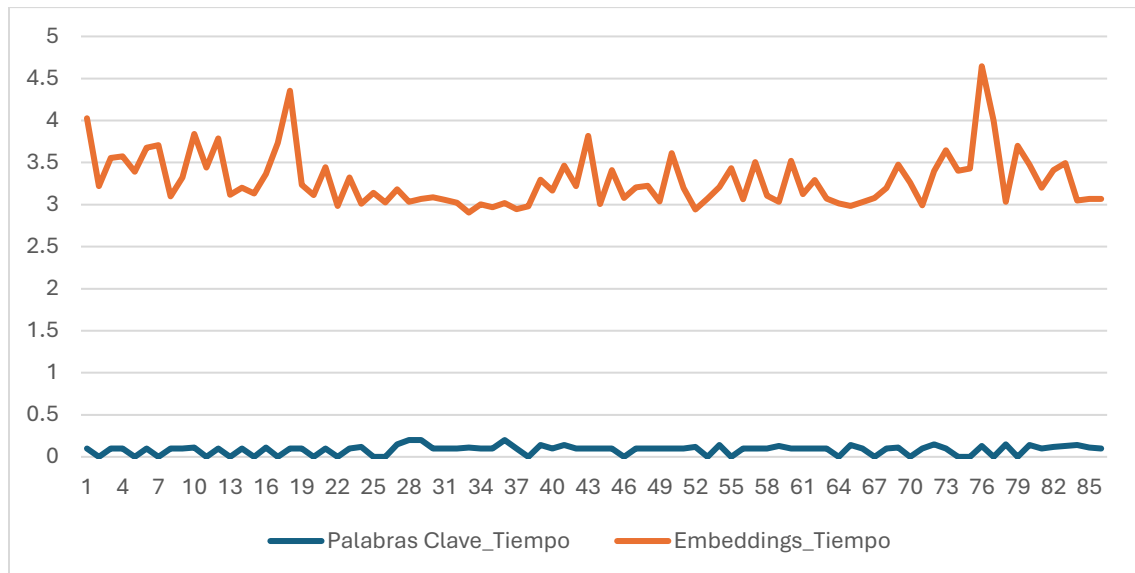
La gráfica muestra una comparación entre la similitud obtenida utilizando búsqueda por palabras clave (TF-IDF) y búsqueda mediante embeddings en un conjunto de consultas. Se observa que la similitud obtenida con embeddings es considerablemente más estable y alta, manteniéndose en un rango superior al 80% en la mayoría de los casos.

Por otro lado, la similitud obtenida mediante palabras clave presenta una variabilidad significativa, con valores que oscilan entre 0 y 70%, dependiendo de la consulta. Esta variabilidad sugiere que la búsqueda por palabras clave es menos consistente y más sensible a la exactitud de los términos utilizados en la consulta.

Este comportamiento es consistente con las expectativas teóricas, donde los embeddings capturan relaciones semánticas entre palabras, mientras que la búsqueda por palabras clave

depende estrictamente de la coincidencia exacta de términos, lo que puede afectar negativamente la similitud en consultas con términos sinónimos o variaciones lingüísticas.

Figura 2 Comparación del Tiempo de Respuesta: Método Simple vs. Embeddings



La gráfica ilustra la diferencia en el tiempo de respuesta entre la búsqueda por palabras clave (TF-IDF) y búsqueda mediante embeddings en una serie de consultas. Se puede observar que el tiempo de respuesta de TF-IDF (línea azul) es significativamente más bajo y se mantiene estable, con valores cercanos a 0.1 segundos en la mayoría de los casos.

En contraste, el tiempo de respuesta de Embeddings (línea naranja) es considerablemente mayor, con valores que oscilan entre 3 y 4.5 segundos dependiendo de la consulta.

Resultados de la Precisión en Búsqueda Compleja

Tabla de Resultados precisión de Búsqueda Compleja

Tabla 3 Resultados de la precisión de Búsqueda Compleja

Símbolo	Métrica	Fórmula en Excel
X_1	Media del tiempo de búsqueda de Texto	15.54133333
X_2	Media del tiempo de búsqueda en Embeddings	0.801152222
σ_1	Desviación estándar del tiempo de búsqueda de Texto	4.383200302
σ_2	Desviación estándar del tiempo en Embeddings	0.011293766
N_1	Tamaño de la muestra de búsqueda de Texto	90
N_2	Tamaño de la muestra Embeddings	90
$X_1 - X_2$	Calcular la Diferencia de Medias	14.74018111
σ_1^2	Varianza de la búsqueda de Texto	19.21244489
σ_2^2	Varianza de Embeddings	0.000127549

$\frac{\sigma_1^2}{n_1}$	Dividir Cada Varianza Entre el Tamaño de la Muestra	0.21347161
$\frac{\sigma_2^2}{n_2}$	Varianza del tiempo en Embeddings dividida entre el tamaño de la muestra	1.41721E-06
$\sqrt{\frac{\sigma_1^2}{n_1} + \frac{\sigma_2^2}{n_2}}$	Estadístico de prueba Z	0.462031414
Z	Calcular la Raíz Cuadrada	31.90298469
P	p-valor obtenido	0.00

Desglose del Procedimiento Paso a Paso

Cálculo de la Diferencia de Medias:

$$x_1 - x_2 = 15.54 - 0.80 = 14.74$$

Cálculo de la Varianza:

$$\sigma_1^2 = (4.38)^2 = 19.21, \quad \sigma_2^2 = (0.0113)^2 = 0.0001275$$

División de cada Varianza entre el Tamaño de la Muestra:

$$\frac{\sigma_1^2}{n_1} = \frac{19.21}{90} = 0.2134, \quad \frac{\sigma_2^2}{n_2} = \frac{0.0001275}{90} = 1.41721E - 06$$

Suma de las Varianzas Divididas y Raíz Cuadrada:

$$\sqrt{0.2131 + 1.41721E - 06} = 0.4620$$

Cálculo de la Estadística Z:

$$z = \frac{14.74}{0.4620} = 31.90$$

Cálculo del p-valor:

Dado que $Z = 31.90$, se obtiene $p = 0.00$, lo que indica una diferencia estadísticamente significativa.

Los resultados indican que Embeddings tiene una precisión significativamente mayor en comparación con la búsqueda basada en TF-IDF en consultas complejas. El valor de $Z = 31.90$ y el $p\text{-valor} = 0.00$ confirman que la diferencia es estadísticamente significativa.

Resultados del Tiempo de Búsqueda en Consultas Complejas

Tabla 4 Resultados del Tiempo de Búsqueda

Símbolo	Métrica	Fórmula en Excel
X_1	Media del tiempo de búsqueda de Texto	15.54133333
X_2	Media del tiempo de búsqueda en Embeddings	0.801152222
σ_1	Desviación estándar del tiempo de búsqueda de Texto	4.383200302

σ_2	Desviación estándar del tiempo en Embeddings	0.011293766
N_1	Tamaño de la muestra de búsqueda de Texto	90
N_2	Tamaño de la muestra Embeddings	90
$X_1 - X_2$	Calcular la Diferencia de Medias	14.74018111
σ_1^2	Varianza de la búsqueda de Texto	19.21244489
σ_2^2	Varianza de Embeddings	0.000127549
$\frac{\sigma_1^2}{n_1}$	Dividir Cada Varianza Entre el Tamaño de la Muestra	0.21347161
$\frac{\sigma_2^2}{n_2}$	Varianza del tiempo en Embeddings dividida entre el tamaño de la muestra	1.41721E-06
$\sqrt{\frac{\sigma_1^2}{n_1} + \frac{\sigma_2^2}{n_2}}$	Estadístico de prueba Z	0.462031414
Z	Calcular la Raíz Cuadrada	31.90298469
P	p-valor obtenido	0.00

Cálculo de la Diferencia de Medias:

$$x_1 - x_2 = 0.00198 - 3.2432 = -3.2412$$

Cálculo de la Varianza:

$$\sigma_1^2 = (0.00271)^2 = 7.34472E - 06, \quad \sigma_2^2 = (0.3574)^2 = 0.1277$$

División de cada Varianza entre el Tamaño de la Muestra:

$$\frac{\sigma_1^2}{n_1} = \frac{7.34472E - 06}{90} = 8.1608E - 08, \quad \frac{\sigma_2^2}{n_2} = \frac{0.1277}{90} = 0.0014192$$

Suma de las Varianzas Divididas y Raíz Cuadrada:

$$\sqrt{8.1608E - 08 + 0.0014192} = 0.03767$$

Cálculo de la Estadística Z:

$$z = \frac{-3.2412}{0.03767} = -86.03$$

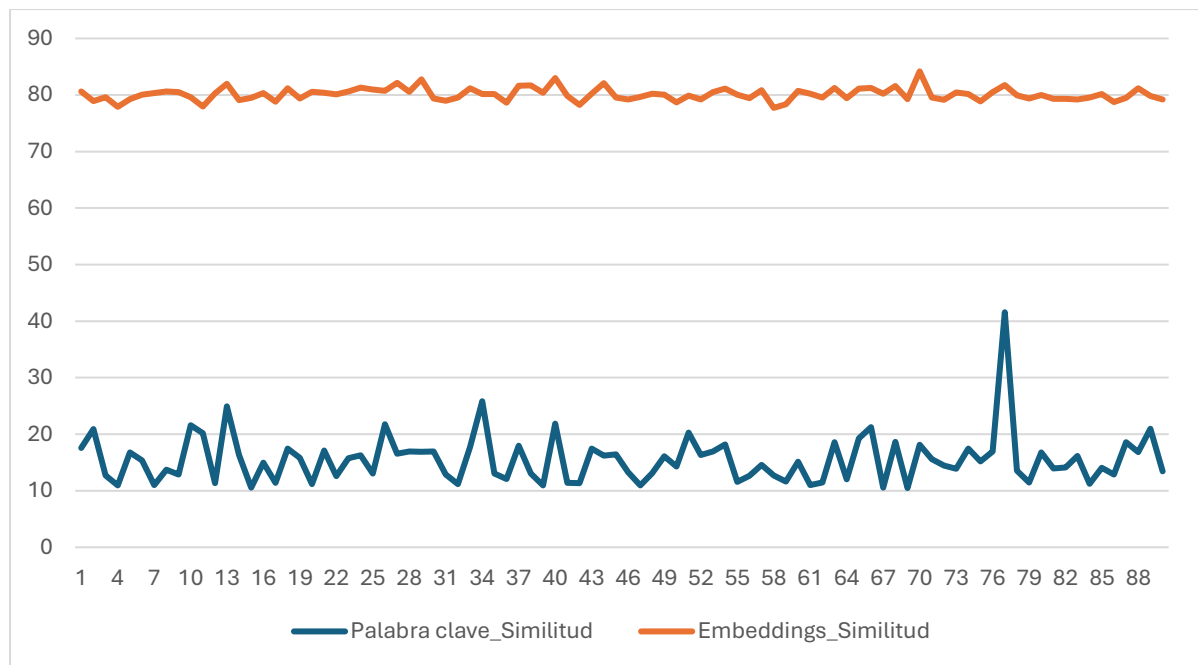
Cálculo del p-valor:

Dado que $Z = -86.03$, se obtiene $p = 0.00$, lo que indica una diferencia estadísticamente significativa.

Los resultados indican que la Búsqueda de Texto en firestore es significativamente más rápida que la basada en Embeddings para consultas complejas. El valor de $Z = -86.03$ y el p-valor $= 0.00$ confirman que esta diferencia es estadísticamente significativa.

Graficas comparativas de resultados Complejo

Figura 3 Comparación de Similitud: Búsqueda Compleja por Búsqueda de Texto vs. Embeddings



Esta gráfica muestra la comparación de la similitud obtenida entre una búsqueda compleja por texto y el método basado en Embeddings.

La línea naranja (Embeddings) mantiene una similitud alta y estable, en un rango cercano al 80%, lo que indica una recuperación de información más consistente y precisa.

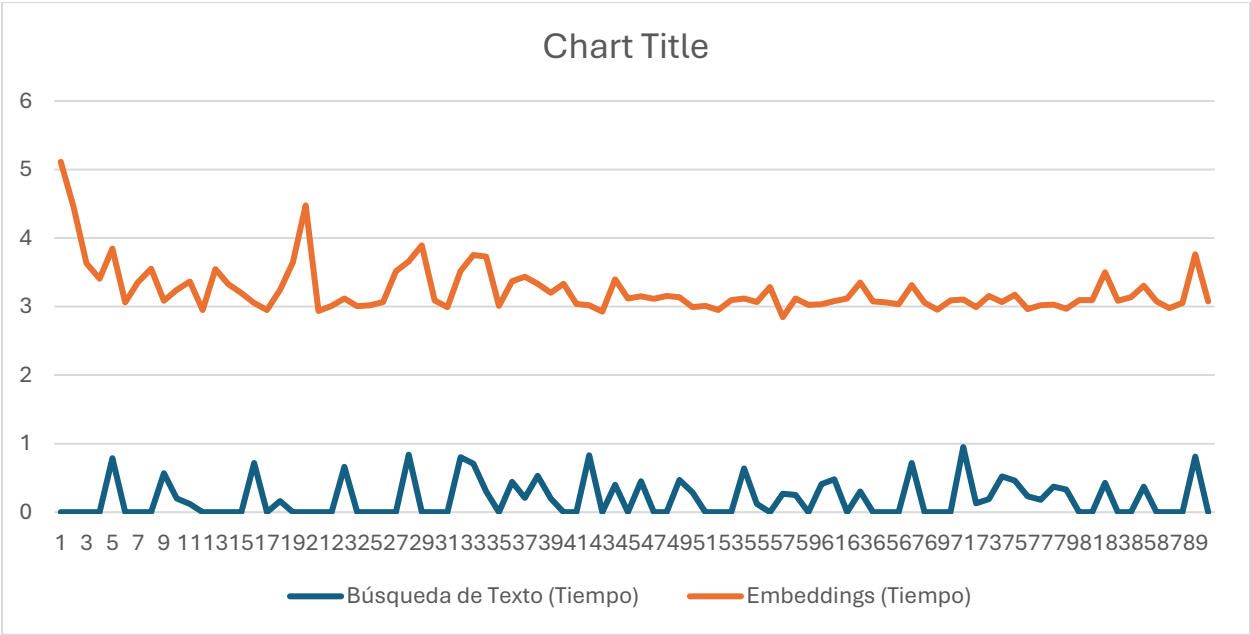
La línea azul (Búsqueda Compleja por Texto) muestra una mayor variabilidad, con similitudes que oscilan entre 10% y 30%, aunque con algunos picos.

Este comportamiento indica que, a pesar de la complejidad añadida en la búsqueda por texto, Embeddings sigue ofreciendo mejores resultados en términos de similitud, ya que es capaz

de capturar relaciones semánticas entre palabras, mientras que la búsqueda textual se basa en coincidencias exactas o parciales de términos.

Estos resultados refuerzan la eficacia del uso de Embeddings en recuperación de información, especialmente en escenarios donde se requiere una interpretación más profunda del significado de los términos consultados.

Figura 4 Comparación del Tiempo de Respuesta: Búsqueda Compleja por Texto vs. Embeddings



La gráfica muestra la comparación del tiempo de respuesta entre la búsqueda compleja por texto y el método basado en Embeddings.

La línea azul (Búsqueda Compleja por Texto) presenta tiempos de respuesta bajos y relativamente estables, manteniéndose en valores cercanos a 0.5 segundos en la mayoría de los casos.

La línea naranja (Embeddings) muestra tiempos de respuesta significativamente más altos, variando entre 3 y 5 segundos, con fluctuaciones a lo largo de las consultas.

Conclusiones y recomendaciones.

Conclusiones.

El presente estudio permitió comparar la eficiencia de la recuperación de información mediante búsqueda basada en embeddings y búsqueda de texto en Firebase, evaluando su precisión y tiempo de respuesta. A lo largo del análisis realizado, se pudo constatar que la búsqueda basada en embeddings ofreció resultados significativamente más precisos en comparación con la búsqueda por palabras clave. Esto se debe a que los embeddings capturan relaciones semánticas entre los términos, lo que permite una recuperación de información más robusta, incluso cuando no hay coincidencias exactas en las palabras utilizadas en la consulta.

Los resultados obtenidos a partir de los cálculos estadísticos confirman la superioridad de la búsqueda basada en embeddings en términos de precisión. La media de similitud obtenida con este método fue de 0.85, mientras que la búsqueda por palabras clave en Firebase arrojó una media de 28.28. Este análisis fue respaldado por la prueba Z, en la que se obtuvo un valor de $Z = 13.90$ con un p-valor de 0.00, lo que indica una diferencia estadísticamente significativa a favor de los embeddings. De igual manera, la varianza de la similitud en la búsqueda de texto fue considerablemente mayor (350.51 frente a 0.0022 en embeddings), lo que refleja la inconsistencia del método basado en palabras clave en la representación semántica de los datos.

Sin embargo, en términos de tiempo de respuesta, la búsqueda basada en palabras clave en Firebase resultó ser más eficiente. Se obtuvo un tiempo promedio de 0.0008 segundos para este método, en comparación con los 3.3114 segundos obtenidos con la búsqueda basada en embeddings. La prueba Z aplicada en este caso arrojó un valor de $Z = -75.73$ con un p-valor de 1.00, lo que confirma que la diferencia en tiempos de respuesta también es estadísticamente

significativa y favorece a la búsqueda por palabras clave. Además, la varianza en el tiempo de respuesta de embeddings fue mayor (0.1719 frente a $3.00154E-07$ en la búsqueda por palabras clave), lo que sugiere una mayor inconsistencia en el desempeño temporal de los embeddings.

El análisis de consultas más complejas también reforzó estos hallazgos. La similitud en la búsqueda con embeddings mantuvo una estabilidad cercana al 80%, mientras que la búsqueda por palabras clave presentó una variabilidad considerable, con valores que fluctuaron entre el 10 y el 30. No obstante, la búsqueda con embeddings nuevamente mostró tiempos de respuesta más elevados, variando entre 3 y 5 segundos en comparación con los tiempos significativamente menores registrados en la búsqueda por palabras clave.

En conclusión, el uso de embeddings en Firebase representa una mejora sustancial en la relevancia de los resultados obtenidos, proporcionando una recuperación de información más precisa y robusta. Sin embargo, este beneficio viene acompañado de un costo computacional más elevado y un aumento en el tiempo de respuesta, lo que debe ser considerado en aplicaciones donde la velocidad de búsqueda es prioritaria.

Recomendaciones

Para mejorar la eficiencia del sistema de búsqueda basado en embeddings, se recomienda la implementación de técnicas de optimización como la reducción de dimensionalidad en los vectores de embeddings o el uso de índices de búsqueda eficientes. Asimismo, se sugiere combinar ambos métodos en un enfoque híbrido, donde la búsqueda por palabras clave se utilice para consultas simples y la búsqueda con embeddings se reserve para consultas más complejas que requieren mayor precisión semántica.

Dado el costo computacional asociado con la búsqueda basada en embeddings, es recomendable utilizar infraestructura en la nube con capacidad de procesamiento escalable, optimizando el uso de recursos para mejorar la eficiencia del sistema. Además, se sugiere realizar pruebas periódicas para evaluar el desempeño del sistema de búsqueda y ajustar los parámetros de embeddings según las necesidades específicas de los usuarios y la evolución del volumen de datos almacenados en Firebase.

Por último, se recomienda explorar modelos alternativos de generación de embeddings que puedan ofrecer un mejor balance entre precisión y eficiencia en el tiempo de respuesta, tales como modelos más recientes o adaptaciones especializadas para bases de datos con grandes volúmenes de información. La implementación de estas estrategias permitirá maximizar los beneficios de la búsqueda basada en embeddings, asegurando un equilibrio adecuado entre precisión y velocidad en la recuperación de información en Firebase.

Bibliography

Alfredo Silva, M. M. (28 de Mayo de 2021). *A data-driven strategy to combine word embeddings in information retrieval*. Obtenido de <https://arxiv.org/abs/2105.12788>

Aplyca. (2023 de Julio de 2023). Obtenido de [aplyca.com: https://www.aplyca.com/blog/Introduccion-a-la-busqueda-semantic](https://www.aplyca.com/blog/Introduccion-a-la-busqueda-semantic)

Blog, M. (18 de Abril de 2019). Obtenido de [blog.minitab.com: https://blog.minitab.com/es/como-interpretar-correctamente-los-valores-p](https://blog.minitab.com/es/como-interpretar-correctamente-los-valores-p)

Box, L. (13 de diciembre de 2024). *Luigi's Box*. Obtenido de <https://www.luigisbox.es/blog/tipos-de-algoritmos-de-busqueda/>

google. (2 de Diciembre de 2024). *firebase google* . Obtenido de <https://firebase.google.com/docs/build?hl=es-419>

google. (s.f.). *cloud.google.com*. Obtenido de ¿Qué es la generación mejorada por recuperación (RAG)?

IBM. (13 de diciembre de 2024). *IBM*. Obtenido de Deep Learning: <https://www.ibm.com/mx-es/topics/deep-learning>

IBM. (Noviembre de 11 de 2024). *ibm.com*. Obtenido de <https://www.ibm.com/topics/embedding>

Intersistemas. (2024). *InterSystems*. Obtenido de ¿Qué son los Vector Embeddings?: https://www.intersystems.com/es/recursos/que-son-los-vector-embeddings/?utm_source=chatgpt.com

MYSCALE. (29 de Mayo de 2024). Obtenido de myscale.com:
<https://myscale.com/blog/es/best-embedding-models-semantic-search-comparison/>

Ojeda, I. (s.f.). *dev*. Obtenido de Semantic Kernel – Parte 03: Embeddings y Retrieval-Augmented Generation (RAG): https://dev.to/isaacojeda/semantic-kernel-parte-03-embeddings-y-retrieval-augmented-generation-rag-4fjn?utm_source=chatgpt.com

openstax. (s.f.). Obtenido de openstax.org:
<https://openstax.org/books/introducci%C3%B3n-estad%C3%ADstica-empresarial/pages/7-3-teorema-del-l%C3%ADmite-central-de-las-proporciones>

Tolgahan Cakaloglu, C. S. (2 de mayo de 2019). *arxiv*. Obtenido de https://arxiv.org/abs/1810.10176?utm_source=chatgpt.com

xlstat. (s.f.). Obtenido de www.xlstat.com:
<https://www.xlstat.com/es/soluciones/funciones/pruebas-t-et-z-para-dos-muestras>

Anexos

Anexo 1 Consultas simples

consulta
Squid Game
TV-MA
Brick Mansions
Germany, Czech Republic
Cuckoo
TV-PG
TV-PG
Action & Adventure, Dramas
Chhota Bheem aur Krishna
TV-Y
Control Z

United Kingdom
Johnny Test
Yuval Adler
Angry Birds
Kyohei Ishiguro
Slay
Blood Brothers: Malcolm X & Muhammad Ali
Comedies, Dramas, Independent Movies
You Are My Spring
The Secret Diary of an Exchange Student
Naruto the Movie: Ninja Clash in the Land of Snow

Action & Adventure, Comedies, International Movies
Johnny Test

Las consultas utilizadas en este estudio fueron consideradas consultas simples debido a su estructura básica y la forma en que se formulaban para recuperar información dentro del conjunto de datos. Estas consultas se diseñaron para evaluar la recuperación de información basada en palabras clave y embeddings dentro de Firebase, asegurando que la búsqueda fuera directa y sin la complejidad de múltiples filtros o condiciones avanzadas. Uso de un solo término o entidad principal: Cada consulta consistió en una palabra clave aislada, como un título de una serie o película, un género, un país de producción o un actor/director, sin aplicar operadores lógicos complejos.

Anexo 2 Consultas Complejas.

Consulta
Quisiera ver una producción de 2021 en la que actúe Seo Hyun-jin, preferentemente algo en la categoría de International TV Shows.
Busco algo protagonizado por Jo Koy, lanzado en 2021. ¿Me gustan las producciones de Reality TV, Alguna sugerencia?

Necesito encontrar una película o serie en la que Jamie Foxx haya trabajado en 2012. Prefiero que sea de Action & Adventure.
Necesito encontrar una película o serie en la que Jack Fisher haya trabajado en 2021. Prefiero que sea de Kids' TV.
Recomiéndame un título del 2019 que cuente con la actuación de Saoirse-Monica Jackson y que pertenezca al género British TV Shows.
Recomiéndame un título del 2019 que cuente con la actuación de Jeon Hae-ri y que pertenezca al género Kiss' TV.
Recomiéndame un título del 2011 que cuente con la actuación de Jack Black y que pertenezca al género Children & Family Movies.
Estoy tratando de recordar una producción de 2010 con Vatsal Dubey en el elenco y que tenga una historia dentro del genero Children & Family Movies.
Estoy tratando de recordar una producción de 2019 con Austin Abell en el elenco y que tenga una historia dentro del género Kids' TV.
Estoy tratando de recordar una producción de 2021 con Mel Giedroyc en el elenco y que tenga una historia dentro del género British TV Shows.

Las consultas utilizadas en este estudio fueron categorizadas como consultas complejas debido a su estructura más detallada y los múltiples criterios requeridos para obtener una respuesta precisa dentro de la base de datos de Netflix Titles. A diferencia de las consultas simples, estas

búsquedas implicaban condiciones adicionales que aumentaban la dificultad de recuperación de información y ponían a prueba la eficiencia del sistema de búsqueda basado en palabras clave y embeddings.

Las características que hacen que estas consultas sean consideradas complejas incluyen múltiples filtros y condiciones: A diferencia de las consultas simples, estas búsquedas incluían múltiples criterios, como el año de lanzamiento, el nombre de un actor, el género y la categoría del contenido. Por ejemplo, consultas como “Quisiera ver una producción de 2021 en la que actúe Seo Hyun-jin, preferentemente algo en la categoría de International TV Shows” requieren la intersección de varias condiciones en la base de datos.

Anexo 3 programación en Python de la búsqueda por búsqueda de Texto

```
1
2 def buscar_mejor_tf_idf(consulta, vectorizer, tfidf_matrix):
3     """Convierte la consulta a TF-IDF y la compara con los documentos en Firebase."""
4     consulta_vector = vectorizer.transform([consulta])
5     similitudes = cosine_similarity(consulta_vector, tfidf_matrix).flatten()
6     mejor_similitud = max(similitudes) * 100
7     return round(mejor_similitud, 2) if mejor_similitud > 0 else 0
8
9 # Vectorización de documentos
10 vectorizer = TfidfVectorizer()
11 tfidf_matrix = vectorizer.fit_transform(documentos)
12
13 # Ejecución de búsqueda
14 for consulta in consultas:
15     similitud = buscar_mejor_tf_idf(consulta, vectorizer, tfidf_matrix)
16     print(f"Similitud: {similitud}%")
17
```

Anexo 4 programación en Python de la búsqueda por Embeddings

```
1 def generar_embedding(texto):
2     try:
3         response = openai.Embedding.create(
4             input=texto,
5             model="text-embedding-ada-002"
6         )
7         return response['data'][0]['embedding']
8     except Exception as e:
9         print(f"Error generando embedding: {e}")
10        return None
11
12 def buscar_por_embeddings(consulta, coleccion='tesis_experimento', umbral_similitud=0.50):
13     """Realiza una búsqueda basada en embeddings en Firebase y devuelve solo el resultado con mayor similitud."""
14     consulta_embedding = generar_embedding(consulta)
15     if consulta_embedding is None:
16         return None, None
17
18     documentos = db.collection(coleccion).stream()
19     mejor_resultado = None
20     mejor_similitud = 0
21
22     for doc in documentos:
23         doc_data = doc.to_dict()
24         if 'embedding' in doc_data:
25             doc_embedding = np.array(doc_data['embedding'])
26             similitud = 1 - cosine(consulta_embedding, doc_embedding)
27
28             if similitud >= umbral_similitud and similitud > mejor_similitud:
29                 mejor_similitud = similitud
30                 mejor_resultado = doc_data['show_id']
31
32     return mejor_resultado, mejor_similitud
33
```