



ING. SISTEMAS

Trabajo de titulación previo a la obtención del título de
Ingeniero en Sistemas de Información.

Autor: Juan David Moromenacho Aguirre

Tutora: Fernanda Paulina Vizcaino Imacaña

Ajuste fino de un modelo de *embeddings* ligero para la recuperación semántica de jerga coloquial en comercio electrónico mediante bases de datos vectoriales

*Fine-tuning of a Lightweight Embedder for Colloquial Slang
Semantic Retrieval in E-commerce Using Vector Databases*

Quito-2026

Aprobación del Tutor del Trabajo de Integración Curricular

Yo Fernanda Paulina Vizcaino Imacaña, en calidad de tutora del Trabajo de Integración Curricular denominado: *Ajuste fino de un modelo de embeddings ligero para la recuperación semántica de jerga coloquial en comercio electrónico mediante bases de datos vectoriales*, realizado por Juan David Moromenacho Aguirre, ha sido orientado y revisado en todas sus partes y considero que cumple los requisitos establecidos por la institución. En virtud de ello doy mi aprobación a fin de continuar con el proceso académico correspondiente.

Fernanda Paulina Vizcaino Imacaña

Tutora del Trabajo de Integración Curricular

Declaración de autoría y cesión de derechos

Yo, Juan David Moromenacho Aguirre, declaro bajo juramento que el trabajo aquí descrito es de mi autoría; que no ha sido presentado anteriormente para ningún grado o calificación profesional y que se ha consultado la bibliografía detallada.

Cedo mis derechos de propiedad intelectual a la Universidad Internacional del Ecuador, para que sea publicado y divulgado en internet, según lo establecido en la Ley de Propiedad Intelectual, Reglamento y Leyes.

Autor: Juan David Moromenacho Aguirre

C.I.: 1724215957

Acuerdo de Confidencialidad

La Biblioteca de la Universidad Internacional del Ecuador se compromete a:

No divulgar, utilizar ni revelar a otros la información confidencial obtenida en el presente trabajo, ya sea intencionalmente o por falta de cuidado en su manejo, en forma personal o bien a través de sus empleados.

Manejar la información confidencial de la misma manera en que se maneja la información propia de carácter confidencial, la cual en ninguna circunstancia podrá estar por debajo de los estándares aceptables de debida diligencia y prudencia.

[Breve información del motivo por el que el trabajo de integración curricular es confidencial]

Fernanda Paulina Vizcaino Imacaña

Directora de la Escuela de Ciencias de la Computación

Fernández Gilbert María Gabriela

Directora de Biblioteca y Desarrollo Cultural

Dedicatoria

A mi padre, Alfredo Moromenacho, por haberme guiado hacia la Universidad Internacional del Ecuador y dar inicio a este camino que hoy culmina. A mi madre, Ximena Aguirre, por enseñarme a creer en mí y a enfrentar cada desafío sin miedo a intentarlo. A mi hermano, Diego Moromenacho, por su apoyo incondicional y por brindarme la oportunidad de dedicarme plenamente a mi formación universitaria. Gracias a mi familia.

Agradecimiento

Expreso mi más sincero agradecimiento a la Universidad Internacional del Ecuador por brindarme el espacio y las oportunidades para desarrollarme académica y profesionalmente. De manera especial, agradezco a la Dra. Fernanda Paulina Vizcaino Imacaña, Directora de la Escuela de Ciencias de la Computación, por su invaluable orientación, apoyo y confianza. Más allá de su asesoría académica, fue la primera investigadora que me abrió las puertas para colaborar en proyectos de investigación, permitiéndome descubrir mi vocación científica y fortalecer las competencias que hoy forman parte de mi perfil profesional. Mi profundo agradecimiento también al Mag. Iván Galo Reyes Chacón, Coordinador de la Carrera de Ingeniería en Sistemas de Información, por su constante respaldo, sus enseñanzas y la confianza depositada en mí durante mi formación. Tanto la Dra. Fernanda Paulina Vizcaino Imacaña como el Mag. Iván Galo Reyes Chacón han sido pilares fundamentales en mi crecimiento académico y profesional, inspirándome con su ejemplo de compromiso, liderazgo y excelencia. Deseo expresar un reconocimiento especial a los investigadores Santiago Guerrero, Lizbeth Peña, Ronie Martínez y Steven Vinuesa. Su influencia fue decisiva en mi formación como estudiante. Gracias a sus paciencia con la que guiaron mi proceso de aprendizaje, incluso en aquellos momentos en los que cometí errores. Finalmente, agradezco a toda la comunidad de la Escuela de Ciencias de la Computación, por la confianza y las oportunidades brindadas a lo largo de mi trayectoria universitaria. Constituye un orgullo formar parte de la primera promoción de graduados de la Carrera de Ingeniería en Sistemas de Información en la sede Quito.

Tabla de Contenido

Portada	I
Aprobación del Tutor	II
Declaración de Autoría y Cesión de Derechos	III
Acuerdo de Confidencialidad	IV
Dedicatoria	V
Agradecimiento	VI
Tabla de Contenido	VII
Índice de Tablas	X
Índice de Figuras	XII
Índice de Nomenclatura	XIII
Convenciones Terminológicas	XV
Resumen	1
Abstract	2
Capítulo I	3
Introducción	3
1.1 Definición del Problema de Investigación	3
1.2 Justificación e Importancia del Estudio	4
1.3 Alcance de la Investigación	4
1.4 Objetivos de Investigación	6
1.4.1 Objetivo General	6
1.4.2 Objetivos Específicos	6
Capítulo II	7
Revisión de Literatura	7
2.1 Recuperación de Información en el Comercio Electrónico	7
2.2 Discrepancia de Vocabulario y Lenguaje Coloquial	10
2.3 <i>Embeddings</i> y Modelos <i>Transformer</i>	12
2.4 <i>Fine-tuning</i> de Modelos de <i>Embeddings</i> Ligeros	14
2.5 Justificación Comparada de las Decisiones de Modelado	18

2.5.1	Primer Eje: Codificador Preentrenado para el Español Informal	18
2.5.2	Segundo Eje: Arquitectura de Recuperación Densa	20
2.5.3	Tercer Eje: Dimensionalidad de la Representación Almacenada	21
2.5.4	Síntesis del Compromiso Técnico	23
2.6	Bases de Datos Vectoriales y Recuperación Semántica	24
2.7	Procedencia, Confiabilidad y Margen de Error de los Datos	27
2.8	Construcción y Validación de Colecciones de Prueba	36
2.8.1	El Paradigma de Cranfield y sus Supuestos	36
2.8.2	Juicios de Relevancia: Quién Decide Qué es Relevante	37
2.8.3	Complejidad de los Juicios y Sesgo de Agrupamiento	38
2.8.4	Consultas Reales frente a Consultas Construidas	39
2.8.5	Generación Sintética de Pares Consulta–Documento	40
2.8.6	Tipos de Validez Aplicados a una Colección de Prueba	41
2.8.7	Documentación del Conjunto como Requisito Metodológico	41
2.9	Evaluación de Sistemas de Recuperación	42
2.10	Reducción de Dimensionalidad para Visualización: Análisis de Componentes Principales	45
	Capítulo III	47
	Desarrollo	47
3.1	Arquitectura del Sistema	47
3.1.1	Componentes Principales	47
3.1.2	Puntos de Acceso de la API	48
3.1.3	Pila Tecnológica	49
3.2	Construcción del Conjunto de Datos	50
3.2.1	Generación de los Pares Base	51
3.2.2	Partición y Aumentación	52
3.2.3	Procedencia y Validación del Vocabulario Coloquial	54
3.2.4	Distribución de Fenómenos Coloquiales	57
3.2.5	Relación con el Catálogo de la API	58
3.2.6	Taxonomía del Catálogo	59
3.2.7	Contraste del Protocolo con los Estándares de Construcción de Colecciones ..	60
3.3	Entrenamiento y <i>Fine-tuning</i>	63

3.3.1	Objetivo de Entrenamiento	63
3.3.2	Configuración de Entrenamiento	64
3.4	Implementación de la Búsqueda	66
3.4.1	Arquitectura Integrada vs. de Dos Pasos	67
3.5	Protocolo de Evaluación	68
3.5.1	Sistemas Comparados y Nomenclatura	68
3.5.2	Métricas	68
3.5.3	Significancia Estadística	69
3.5.4	Validación Externa con Datos Reales de Mercado Libre	70
3.5.5	Trazabilidad de las Cifras Reportadas	71
Capítulo IV		73
Análisis de Resultados		73
4.1	Configuración Experimental	73
4.1.1	Independencia del Conjunto de Prueba	73
4.2	Evaluación del Modelo Base y Ajustado	74
4.2.1	Convergencia del <i>Fine-tuning</i>	74
4.2.2	Calidad de Recuperación	75
4.2.3	Control por Solapamiento con el Entrenamiento	76
4.3	Comparación con BM25	79
4.3.1	Rendimiento por Fenómeno Lingüístico	80
4.3.2	Análisis de Casos	82
4.4	Validación Externa con Consultas Reales de Mercado Libre	84
4.5	Comparación Arquitectónica	88
4.6	Discusión de Hallazgos	91
4.6.1	Interpretación de los Resultados	91
4.6.2	Limitaciones	95
Capítulo V		98
Conclusiones y Recomendaciones		98
5.1	Conclusiones	98
5.2	Aporte de Ingeniería: Producto Entregado	100
5.3	Recomendaciones y Trabajo Futuro	102
Referencias Bibliográficas		104

Índice de Tablas

Tabla I	Primer eje de decisión: codificador de partida para el español informal	19
Tabla II	Segundo eje de decisión: bi-codificador de recuperación	21
Tabla III	Tercer eje de decisión: dimensionalidad y costo de almacenamiento	23
Tabla IV	Síntesis de las decisiones de modelado y su criterio de justificación	24
Tabla V	Grado de reutilización de cada subconjunto y su efecto sobre el valor reportado.....	33
Tabla VI	Fuentes de error identificadas en el conjunto de datos y su efecto esperado	35
Tabla VII	Puntos de acceso REST expuestos por la capa de presentación FastAPI ..	49
Tabla VIII	Componentes de la pila del sistema	49
Tabla IX	Composición del conjunto de datos base	51
Tabla X	Tipología de las variantes del vocabulario coloquial	55
Tabla XI	Contraste de las variantes léxicas geográficas con el DLE y el DAmE	56
Tabla XII	Presencia de fenómenos coloquiales en las anclas del conjunto base	58
Tabla XIII	Taxonomía de categorías del catálogo utilizado para el entrenamiento y la evaluación.....	59
Tabla XIV	Contraste del conjunto de datos con los estándares de construcción de colecciones de prueba	61
Tabla XV	Configuración e instrumentación del <i>fine-tuning</i>	65
Tabla XVI	Rendimiento del modelo base vs. el modelo ajustado	76
Tabla XVII	Rendimiento sobre el subconjunto de consultas cuya cadena no aparece en entrenamiento ($n = 139$)	77
Tabla XVIII	MRR según la relación de la consulta de prueba con el entrenamiento	78
Tabla XIX	BM25 vs. modelo ajustado	79
Tabla XX	MRR por fenómeno lingüístico presente en la consulta.....	81
Tabla XXI	Ejemplos de comportamiento del sistema sobre consultas de prueba	83
Tabla XXII	Resultados sobre 3 909 consultas reales del MeLi Data Challenge 2020 (Mercado Libre México).....	85

Tabla XXIII	Validación externa por familia de producto y por presencia de marcadores coloquiales.....	86
Tabla XXIV	Descomposición de la latencia por consulta: integrada (Qdrant) vs. dos pasos (FAISS+SQL)	89
Tabla XXV	Resumen de todos los experimentos comparativos	91

Índice de Figuras

Fig. 1	Arquitectura Limpia de cuatro capas del sistema	48
Fig. 2	Comparación de las arquitecturas de recuperación: integrada (Qdrant) vs. dos pasos (FAISS + SQL)	68
Fig. 3	Convergencia de las pérdidas de entrenamiento y validación (MNRL) durante las 20 épocas del <i>fine-tuning</i>	75
Fig. 4	Comparación de calidad de recuperación: BM25 vs. modelo ajustado	80
Fig. 5	Descomposición de la latencia por consulta: codificación frente a recuperación en cada arquitectura	90
Fig. 6	Proyección PCA (384 → 2 dimensiones) de los <i>embeddings</i> reales del catálogo, coloreados por categoría	94

Índice de Nomenclatura

A continuación se listan, en orden alfabético, las siglas y abreviaturas empleadas a lo largo de este documento.

Sigla	Significado
API	<i>Application Programming Interface</i> (interfaz de programación de aplicaciones)
APA	<i>American Psychological Association</i> (norma de citación bibliográfica)
BM25	<i>Best Matching 25</i> (función de puntuación léxica para recuperación de información)
CPU	<i>Central Processing Unit</i> (unidad central de procesamiento)
DAmer	<i>Diccionario de americanismos</i> (Asociación de Academias de la Lengua Española)
DCG	<i>Discounted Cumulative Gain</i> (ganancia acumulada descontada)
DLE	<i>Diccionario de la lengua española</i> (Real Academia Española)
DPR	<i>Dense Passage Retrieval</i> (recuperación densa de pasajes)
GPU	<i>Graphics Processing Unit</i> (unidad de procesamiento gráfico)
HNSW	<i>Hierarchical Navigable Small World</i> (estructura de grafo para búsqueda aproximada del vecino más cercano)
IA	Inteligencia Artificial
IDCG	<i>Ideal Discounted Cumulative Gain</i> (ganancia acumulada descontada ideal)
JSON	<i>JavaScript Object Notation</i> (formato de intercambio de datos)
MAP	<i>Mean Average Precision</i> (precisión promedio media)
MeLi	Mercado Libre (MeLi Data Challenge: concurso de ciencia de datos de la empresa)
MLM	Prefijo del sitio de Mercado Libre México en los datos del concurso
MNRL	<i>Multiple Negatives Ranking Loss</i> (pérdida de ordenamiento con múltiples negativos, para aprendizaje contrastivo)
MRR	<i>Mean Reciprocal Rank</i> (rango recíproco medio)
NDCG	<i>Normalized Discounted Cumulative Gain</i> (ganancia acumulada descontada normalizada)
PCA	<i>Principal Component Analysis</i> (análisis de componentes principales)
PLN	Procesamiento de Lenguaje Natural (equivalente en inglés: NLP)
RAG	<i>Retrieval-Augmented Generation</i> (generación aumentada por recuperación)
REST	<i>Representational State Transfer</i> (estilo arquitectónico para servicios web)
SQL	<i>Structured Query Language</i> (lenguaje de consulta estructurado)
TF-IDF	<i>Term Frequency – Inverse Document Frequency</i> (frecuencia de término – frecuencia inversa de documento)

Convenciones Terminológicas

Para mantener la coherencia entre español e inglés, el documento aplica tres reglas. Primera: los conceptos que tienen una traducción consolidada en la literatura técnica en español se nombran solo en español, según la tabla siguiente. Segunda: se conservan en inglés, siempre en cursiva, únicamente los términos sin equivalente asentado, de uso predominante en inglés en el área o que forman parte del nombre propio de un método o de una arquitectura, y se definen en su primera aparición. Tercera: las siglas de métricas, métodos y herramientas (MRR, NDCG@K, Recall@K, BM25, MNRL, FAISS, Qdrant) se mantienen en su forma original, que es la que emplea la literatura citada. El término *embedding* se conserva en inglés por formar parte del título de este trabajo y porque sus traducciones propuestas (*incrustación*, *representación vectorial*) no se han asentado como término técnico; designa siempre una representación vectorial densa de un texto. Por la misma razón se emplea siempre *fine-tuning*, y no su traducción.

Término en inglés	Forma empleada en este documento
<i>Se traducen</i>	
<i>baseline</i>	modelo base (MiniLM-L6-v2 original); BM25 se denomina referencia léxica
<i>two-step pipeline</i>	arquitectura de dos pasos
<i>payload</i>	carga útil (metadatos del producto almacenados junto al vector)
<i>endpoint</i>	punto de acceso (de la API)
<i>ranking</i>	ordenamiento
<i>benchmark</i>	banco de pruebas; medición comparativa
<i>checkpoint</i>	punto de control
<i>early stopping / best checkpoint</i>	selección del mejor punto de control
<i>loss, batch, warmup</i>	pérdida, lote, calentamiento
<i>overfitting</i>	sobreajuste
<i>anchor, positive</i>	ancla (consulta del par), positivo (producto relevante)
<i>offline / online evaluation</i>	evaluación fuera de línea / en línea
<i>pooling</i> (colecciones de prueba)	agrupamiento de resultados
<i>re-ranking</i>	reordenamiento
<i>stopwords</i>	palabras vacías
<i>recall</i>	exhaustividad (la métrica conserva su sigla, Recall@K)
<i>machine learning</i>	aprendizaje automático
<i>framework</i>	marco de trabajo
<i>code-switching</i>	alternancia de código
<i>Se conservan en inglés, en cursiva</i>	
<i>embedding</i>	representación vectorial densa de un texto
<i>fine-tuning</i>	reentrenamiento del modelo con los pares del conjunto; produce el <i>modelo ajustado</i>
<i>token</i>	unidad mínima de texto que procesa el modelo
<i>Transformer</i>	nombre propio de la arquitectura de red neuronal basada en atención
<i>Multiple Negatives Ranking Loss</i>	nombre propio de la función de pérdida (MNRL)
<i>bootstrap</i>	nombre del método de remuestreo empleado para los intervalos de confianza
<i>hardware, software</i>	extranjerismos de uso general, que el DLE registra como voces inglesas y en cursiva

Resumen

Este trabajo evalúa el *fine-tuning* de un modelo de *embeddings* ligero (MiniLM-L6-v2) para la recuperación semántica de productos en comercio electrónico cuando las consultas se formulan en español coloquial. Los buscadores basados en palabras clave no logran asociar consultas informales como “estuche pal celu xiami” con descripciones técnicas como “Carcasa protectora Xiaomi Redmi Note 12”. Todo el estudio se realizó con datos sintéticos y en condiciones experimentales controladas: se generó un conjunto base de 1 656 pares ancla–positivo, particionado en entrenamiento, validación y prueba de forma disjunta y ampliado por sustitución de variantes coloquiales a 9 684 pares de entrenamiento, y se ajustó el modelo con *Multiple Negatives Ranking Loss* durante 20 épocas, con selección del mejor punto de control por pérdida de validación. Sobre 165 consultas de prueba y un corpus de 833 productos, el modelo ajustado mejoró el MRR en un 15.8 % frente al modelo base ($p = 1.3 \times 10^{-5}$; $d_z = 0.383$, efecto pequeño) y en un 11.6 % frente a BM25, al que el modelo sin *fine-tuning* no superaba. Estas cifras reflejan sobre todo la generalización a consultas nuevas sobre un catálogo parcialmente conocido: 140 de las 165 consultas de prueba se refieren a productos presentes en el entrenamiento. En las 25 consultas sobre productos no vistos la ventaja frente a BM25 se mantuvo, y la evidencia específica sobre jerga coloquial (23 consultas con marcadores) es preliminar. La latencia por consulta la domina la codificación (13.42 de 15.91 ms en la mediana) y no el almacén vectorial, de modo que las dos arquitecturas comparadas — Qdrant con carga útil integrada y FAISS+SQL de dos pasos — operan por debajo de 100 ms en CPU, y una API REST en FastAPI resolvió 1 000 peticiones con un 100 % de éxito. Una validación externa sobre 3 909 búsquedas reales de Mercado Libre México (MeLi Data Challenge 2020) no reprodujo la ventaja del modelo ajustado: BM25 recuperó mejor el producto comprado (MRR de 0.044 frente a 0.037) y el modelo base identificó mejor el tipo de producto (74.0 % frente a 67.8 % de acierto en la primera posición). Los resultados aportan evidencia favorable al enfoque en el escenario controlado, pero muestran que el *fine-tuning* con datos sintéticos no basta para consultas reales, que deberán formar parte del *fine-tuning* en trabajos posteriores.

Palabras clave: búsqueda semántica, *fine-tuning*, *embeddings* de oraciones, lenguaje coloquial, comercio electrónico, base de datos vectorial, Qdrant, recuperación de información.

Abstract

This work evaluates the fine-tuning of a lightweight embedding model (MiniLM-L6-v2) for semantic product retrieval in e-commerce when queries are written in colloquial Spanish. Keyword-based search fails to map informal queries such as “estuche pal celu xiami” to technical descriptions such as “Carcasa protectora Xiaomi Redmi Note 12”. The whole study was carried out on synthetic data under controlled experimental conditions: a base set of 1,656 anchor–positive pairs was generated, split into disjoint training, validation and test partitions, and expanded by colloquial-variant substitution to 9,684 training pairs; the model was fine-tuned with Multiple Negatives Ranking Loss over 20 epochs, selecting the best checkpoint by validation loss. On 165 test queries over an 833-product corpus, the fine-tuned model improved MRR by 15.8 % over the base model ($p = 1.3 \times 10^{-5}$; $d_z = 0.383$, small effect) and by 11.6 % over BM25, which the untuned model did not outperform. These figures mainly reflect generalization to new queries over a partially known catalog: 140 of the 165 test queries refer to products seen during training. On the 25 queries about unseen products the advantage over BM25 held, and the evidence specific to colloquial slang (23 queries with colloquial markers) is preliminary. Per-query latency is dominated by query encoding (13.42 of 15.91 ms at the median) rather than by the vector store, so both compared architectures — Qdrant with an integrated payload and a two-step FAISS+SQL pipeline — run below 100 ms on CPU, and a FastAPI REST service answered 1,000 requests with a 100 % success rate. An external validation on 3,909 real Mercado Libre Mexico searches (MeLi Data Challenge 2020) did not reproduce the fine-tuned model's advantage: BM25 retrieved the purchased product better (MRR 0.044 vs. 0.037), and the base model better identified the product type (74.0 % vs. 67.8 % top-1 accuracy). The results provide favorable evidence for the approach in the controlled setting, but show that fine-tuning on synthetic data is not sufficient for real queries, which should be part of the training data in future work.

Keywords: semantic search, fine-tuning, sentence embeddings, colloquial language, e-commerce, vector database, Qdrant, information retrieval.

Capítulo I

Introducción

Los motores de búsqueda tradicionales en el comercio electrónico se apoyan predominantemente en la coincidencia de palabras clave y en algoritmos de recuperación léxica como TF-IDF o BM25 (Robertson & Zaragoza, 2009). Estos sistemas presentan limitaciones inherentes que afectan negativamente la experiencia del usuario, principalmente el *problema de discrepancia de vocabulario* (Furnas et al., 1987), en el cual la intención del usuario no coincide con los términos exactos utilizados en el catálogo. Este capítulo delimita el problema de investigación, justifica su relevancia técnica y aplicada, establece el alcance del estudio y formula los objetivos que guían el desarrollo del trabajo.

Definición del Problema de Investigación

Este proyecto se centra en la mejora de la recuperación semántica en el comercio electrónico cuando el usuario emplea jerga coloquial, regionalismos y abreviaturas que no coinciden con la terminología técnica del catálogo. El problema se origina en la discrepancia de vocabulario (Furnas et al., 1987): distintas personas describen un mismo concepto con palabras diferentes, y esta brecha afecta directamente a los sistemas tradicionales de búsqueda basados en coincidencia léxica como TF-IDF o BM25 (Robertson & Zaragoza, 2009), generando resultados poco precisos y degradando la experiencia del usuario.

El problema es especialmente agudo en el comercio electrónico latinoamericano, un mercado caracterizado por una alta variabilidad ortográfica y dialectal en las consultas de búsqueda, fenómeno documentado empíricamente en la literatura de recuperación de información y que se ha mostrado capaz de degradar el desempeño de los modelos de búsqueda cuando estos no fueron expuestos a dicha variación durante su entrenamiento (Chari et al., 2023; Ren et al., 2024). En este contexto, los usuarios formulan consultas informales como “estuche pal celu xiami”, “funda pal iphon” o “cauchos pal auto”, que un emparejador léxico no logra asociar con entradas del catálogo redactadas en terminología técnica, tales como “Carcasa protectora Xiaomi Redmi Note 12”, “Funda protectora iPhone 13” o “Neumáticos para vehículo”. Los modelos de *embeddings* genéricos, preentrenados sobre corpus formales, tampoco resuelven adecuadamente esta brecha, pues su desempeño se degrada de forma consistente al aplicarse fuera del dominio y del registro lingüístico sobre el que fueron entrenados, sin mediar una

etapa de adaptación específica (Gururangan et al., 2020; Thakur et al., 2021).

La investigación propone abordar esta limitación mediante el ***fine-tuning* supervisado** de un modelo de *embeddings* ligero (MiniLM-L6-v2) usando aprendizaje contrastivo con pares consulta–producto coloquiales, y su integración con una base de datos vectorial que almacena los metadatos del producto junto a cada vector. El estudio evalúa esa propuesta sobre un conjunto de datos sintético y en condiciones experimentales controladas, y la contrasta de forma complementaria con búsquedas reales de usuarios de Mercado Libre.

Justificación e Importancia del Estudio

La relevancia del trabajo radica en su aporte tanto técnico como aplicado. Desde el punto de vista técnico, el uso de *embeddings* y del aprendizaje contrastivo responde a la evolución actual del Procesamiento de Lenguaje Natural (PLN) hacia modelos capaces de capturar similitud semántica más allá de las coincidencias exactas de términos. La irrupción de las arquitecturas basadas en *Transformers* (Vaswani et al., 2017) y de las representaciones vectoriales densas (Mikolov et al., 2013) ha impulsado un cambio de paradigma en la recuperación de información, en el que el significado, y no la forma superficial de las palabras, determina la relevancia.

Desde el punto de vista práctico, la propuesta se orienta a mejorar la búsqueda de productos en plataformas de comercio electrónico latinoamericanas, donde expresiones coloquiales como “estuche pal celu xiami” deben mapearse correctamente a descripciones técnicas del catálogo. Una recuperación deficiente se traduce en pérdida de conversiones y en una experiencia de usuario degradada, tal como lo reportan estudios de recuperación semántica desplegados en plataformas de comercio electrónico a gran escala (J. Lin et al., 2024), por lo que resolver la discrepancia de vocabulario tiene un impacto comercial directo. Además, al privilegiar un modelo ligero apto para inferencia en CPU, la solución resulta accesible para comercios que operan con infraestructura modesta, sin depender de aceleradores de *hardware* costosos.

Alcance de la Investigación

La investigación se delimita a la comparación entre un motor de búsqueda léxico tradicional (BM25) y un modelo de *embeddings* ligero con *fine-tuning*, dentro del dominio del comercio electrónico y para consultas en español latinoamericano, con atención particular a formas de uso documentado en Ecuador, Colombia y Perú (la procedencia y el alcance regional del vocabulario empleado se detallan en la Sección 3.2.3). El propósito central del estudio

es determinar en qué medida el enfoque semántico mejora la recuperación frente al enfoque léxico tradicional en un escenario experimental controlado y con datos sintéticos, y valorar si constituye una alternativa o un complemento viable; establecer si puede reemplazarlo en condiciones de producción exigiría someter el modelo a *fine-tuning* y validarlo con consultas reales de la plataforma de destino, lo que queda fuera de este trabajo. Tampoco es objetivo explorar otras familias de arquitecturas de recuperación fuera de este contraste. Para ello, se evalúa el comportamiento de un modelo MiniLM-L6-v2 (W. Wang et al., 2020) con *fine-tuning* frente a tres puntos de comparación: el mismo modelo sin *fine-tuning* (modelo base), el buscador léxico BM25 (Robertson & Zaragoza, 2009) (comparación central del estudio) y, de forma complementaria, una arquitectura alternativa de dos pasos basada en FAISS+SQL, incluida únicamente para contrastar el costo operativo de dos estrategias de almacenamiento vectorial una vez adoptado el enfoque semántico.

El sistema se valida sobre un corpus de 833 descripciones de producto y una partición reservada de 165 consultas (el catálogo sintético de 500 productos se emplea únicamente para la validación funcional de la API), empleando métricas de calidad de recuperación (MRR, NDCG@10, Recall@K) y de rendimiento (latencia p50 y p95), con el objetivo de verificar, en el entorno de medición empleado, que la operación en CPU se mantiene por debajo de 100 ms. De forma complementaria, los tres sistemas se contrastan sobre 3,909 consultas reales de Mercado Libre México, procedentes del MeLi Data Challenge 2020 (**meli2020challenge**), como validación externa sobre datos que no intervienen en el entrenamiento (Sección 4.4).

Quedan fuera del alcance de este trabajo el *fine-tuning* del modelo con consultas reales, la evaluación con juicios de relevancia humanos, el escalamiento a catálogos masivos (más de 100,000 productos), la cobertura de otros idiomas y el despliegue sobre GPU. Tampoco se aborda la personalización de resultados a partir del historial o perfil de una cuenta de usuario, ni la integración del motor de búsqueda en interfaces de autoservicio (por ejemplo, quioscos de autopago). Estos aspectos se identifican como trabajo futuro: de confirmarse la viabilidad del enfoque semántico frente al tradicional, la misma arquitectura de *embeddings* podría, en investigaciones posteriores, extenderse hacia la personalización de productos por cuenta de usuario y hacia su aplicabilidad en interfaces de autopago, sin que ello forme parte de los objetivos del presente estudio.

Objetivos de Investigación

Objetivo General

Evaluar, sobre un conjunto de datos sintético y en condiciones experimentales controladas, la aplicabilidad de un modelo de *embeddings* ligero con *fine-tuning* supervisado (MiniLM-L6-v2) como motor de búsqueda semántica para comercio electrónico en mercados latinoamericanos, midiendo su capacidad de asociar consultas con jerga coloquial, abreviaturas y variaciones ortográficas con las descripciones técnicas del catálogo, con el fin de determinar si constituye una alternativa viable frente al motor de búsqueda léxico BM25 sobre *hardware* de CPU, con una latencia inferior a 100 ms.

Objetivos Específicos

- Construir un conjunto de datos sintético de 1,656 pares ancla–positivo, ampliable por aumentación por reglas (9,684 pares de entrenamiento tras la partición), que aproxime la escritura informal de los usuarios latinoamericanos (jerga, regionalismos, abreviaturas y variaciones ortográficas), a fin de entrenar y evaluar el modelo en un escenario controlado, sin pretender representatividad estadística del habla real.
- Aplicar *fine-tuning* al modelo MiniLM-L6-v2 (22M de parámetros) mediante aprendizaje contrastivo con *Multiple Negatives Ranking Loss* (MNRL) (Henderson et al., 2017) sobre el conjunto de datos construido, para especializarlo en la asociación semántica entre lenguaje coloquial y terminología técnica de catálogo.
- Comparar el desempeño del modelo ajustado frente al motor de búsqueda léxico tradicional (BM25) y frente al mismo modelo sin *fine-tuning*, mediante métricas de calidad de recuperación (MRR, NDCG@10, Recall@K) y pruebas de significancia estadística (prueba de rangos con signo de Wilcoxon (Wilcoxon, 1945) y tamaño del efecto con la d de Cohen para muestras pareadas, d_z), a fin de establecer si la mejora observada es distinguible del azar y de qué magnitud es.
- Evaluar la viabilidad operativa de la solución en un entorno de ejecución controlado, descomponiendo la latencia por consulta y comparando dos arquitecturas de almacenamiento vectorial (Qdrant integrado vs. FAISS+SQL de dos pasos) en términos de latencia p50 y p95, con el fin de comprobar si el costo de la búsqueda semántica en CPU es compatible con el umbral de 100 ms e identificar qué componente domina el presupuesto de tiempo.

Capítulo II

Revisión de Literatura

Este capítulo revisa los fundamentos teóricos y los trabajos previos que sustentan el sistema propuesto, siguiendo un orden jerárquico que va de lo general a lo específico. La recuperación de información y su aplicación al comercio electrónico constituyen el punto de partida. Profundizando en la materia, se aborda el problema de la discrepancia de vocabulario. Respecto a las soluciones semánticas, se examinan las representaciones densas basadas en *Transformers*. Igualmente, se detalla el *fine-tuning* de modelos ligeros de *embeddings*. Mediante la gestión de arquitectura, se incorporan las bases de datos vectoriales. Evaluando la procedencia y confiabilidad de los datos, se vela por la calidad del sistema. Resaltan también las métricas empleadas para evaluar los sistemas de recuperación. Otras comprobaciones formales incluyen las pruebas estadísticas aplicadas al modelo. Además, se introduce un apartado instrumental especializado. La reducción de dimensionalidad sirve de soporte técnico directo en esta etapa. Habilitando de este modo la exploración analítica de estructuras latentes. Un marco riguroso sostiene el tratamiento metodológico de la información. Esta sección facilita la inspección cualitativa del espacio vectorial. Vinculando estas herramientas con la arquitectura global, se consolida el marco teórico. Ofreciendo finalmente el sustento necesario para la evaluación realizada en el Capítulo IV.

Recuperación de Información en el Comercio Electrónico

La recuperación de información (RI) consiste en ordenar una colección de documentos según su relevancia respecto a una consulta. En el comercio electrónico, los documentos son registros de productos y las consultas son los términos que escriben los compradores. Los sistemas clásicos de RI ordenan los documentos mediante funciones de puntuación léxica como TF-IDF y BM25 (Manning et al., 2008; Robertson & Zaragoza, 2009), que ponderan los términos según su frecuencia en el documento y su rareza en la colección. Estos métodos siguen siendo relevantes por su eficiencia e interpretabilidad; sin embargo, operan sobre la coincidencia léxica superficial y no logran salvar la brecha semántica entre palabras diferentes que expresan la misma intención. El comercio electrónico, donde la consulta del usuario y el lenguaje del catálogo provienen de registros lingüísticos distintos, exige por tanto mecanismos de recuperación más robustos que la simple coincidencia de palabras. Trabajos recientes sobre

recuperación semántica de productos confirman que este problema persiste incluso en plataformas de comercio electrónico maduras, y que la incorporación de representaciones densas mejora de forma medible la experiencia de búsqueda del usuario final (Kothari et al., 2026).

Conviene precisar el mecanismo que hace posible esa eficiencia. Los sistemas léxicos se apoyan en el *índice invertido*, una estructura que asocia cada término del vocabulario con la lista de documentos en que aparece. Recuperar equivale entonces a intersecar unas pocas listas y acumular puntuaciones, operación cuyo costo depende del número de documentos que contienen los términos de la consulta y no del tamaño total del catálogo. Sobre esa estructura, BM25 puntúa un documento D frente a una consulta Q mediante

$$\text{BM25}(D, Q) = \sum_{t \in Q} \text{IDF}(t) \cdot \frac{f(t, D) (k_1 + 1)}{f(t, D) + k_1 \left(1 - b + b \frac{|D|}{\text{avgdl}} \right)}, \quad (2.1)$$

donde $f(t, D)$ es la frecuencia del término t en el documento, $|D|$ la longitud del documento medida en número de términos, avgdl (*average document length*) la longitud media de los documentos de la colección, e $\text{IDF}(t)$ el peso de rareza del término. El cociente $|D|/\text{avgdl}$ es, por tanto, la longitud del documento expresada en unidades de la longitud media: vale 1 para un documento de longitud típica, es mayor que 1 para los documentos largos (a los que penaliza) y menor que 1 para los cortos. Los parámetros k_1 y b controlan, respectivamente, la saturación de la frecuencia (la segunda aparición de un término aporta menos que el primero) y la normalización por longitud, que evita que los documentos extensos acumulen puntuación por el solo hecho de contener más palabras (Robertson & Zaragoza, 2009). Estas dos correcciones explican por qué BM25 sigue siendo una referencia exigente treinta años después de su formulación y por qué la literatura de recuperación densa continúa reportándola como referencia obligada (Thakur et al., 2021).

La vigencia de los métodos léxicos no es solo histórica, sino que responde a propiedades que los modelos neuronales no ofrecen de forma automática. En primer lugar, BM25 no requiere datos de entrenamiento: funciona desde el primer día sobre un catálogo nuevo, lo que resuelve el problema de arranque en frío que afecta a cualquier modelo supervisado. En segundo lugar, su puntuación es interpretable término a término, de modo que un operador puede explicar por qué un producto apareció en determinada posición, algo relevante para la depuración y para la rendición de cuentas comercial. En tercer lugar, la coincidencia exacta sigue siendo el comportamiento deseable para ciertas clases de consulta: códigos de producto,

números de parte, referencias SKU y nombres de marca, donde una aproximación semántica que devuelva “algo parecido” constituye en realidad un error. Estas ventajas explican que los sistemas industriales rara vez sustituyan por completo el componente léxico y opten más bien por combinarlo con el componente denso (Ren et al., 2024).

La búsqueda en comercio electrónico presenta además rasgos que la distinguen de la búsqueda web general y que condicionan el diseño del sistema. Las consultas son notablemente cortas (del orden de dos a cuatro términos) y expresan intención de compra más que intención informativa, lo que reduce la cantidad de evidencia léxica disponible por consulta. El catálogo, a diferencia de una colección de documentos libres, es un conjunto de registros estructurados con atributos explícitos (precio, marca, categoría, disponibilidad, calificación) que participan tanto en el filtrado como en el ordenamiento final. Existe asimismo una distribución de cola larga: un número reducido de consultas concentra la mayor parte del tráfico, mientras que una fracción sustancial del volumen corresponde a formulaciones poco frecuentes para las que no hay suficiente señal de comportamiento histórico. Por último, el criterio último de éxito no es la relevancia temática sino un objetivo de negocio (conversión, ingreso, satisfacción del comprador) que no siempre coincide con las métricas de ordenamiento usadas en la evaluación fuera de línea (Magnani et al., 2022; Ren et al., 2024).

Entre el extremo puramente léxico y el puramente denso se ha consolidado una tercera vía: la *recuperación dispersa aprendida* (*learned sparse retrieval*). Modelos como SPLADE (Formal et al., 2021) emplean un codificador *Transformer* para proyectar la consulta y el documento sobre el espacio del vocabulario, aprendiendo qué términos activar y con qué peso, incluidos términos que no aparecen literalmente en el texto. El resultado es una representación dispersa (la mayoría de las coordenadas valen cero) que puede indexarse en un índice invertido convencional y que, no obstante, incorpora expansión semántica aprendida. En paralelo, la práctica industrial ha adoptado esquemas *híbridos* que fusionan las listas de resultados léxicos y densos; la investigación reciente estudia incluso cómo ponderar dinámicamente ambos componentes según la consulta, en lugar de fijar un peso constante (Hsu & Tzeng, 2025). El presente trabajo se sitúa deliberadamente en el extremo denso, y emplea BM25 únicamente como referencia léxica de comparación (Sección 4.3), pero conviene dejar constancia de que la frontera entre ambos paradigmas es hoy menos nítida de lo que sugiere la contraposición habitual.

En síntesis, el problema no es que los métodos léxicos sean deficientes, sino que su

unidad de emparejamiento (el *token* exacto) resulta insuficiente cuando el comprador y el catálogo no comparten vocabulario. La siguiente sección caracteriza precisamente esa condición.

Discrepancia de Vocabulario y Lenguaje Coloquial

El problema de discrepancia de vocabulario (*vocabulary mismatch*) (Furnas et al., 1987) describe el hecho ampliamente documentado de que distintas personas describen un mismo concepto con palabras diferentes. El estudio original cuantificó el fenómeno: al pedir a distintos sujetos que nombraran un mismo objeto o función familiar, la probabilidad de que dos personas eligieran espontáneamente el mismo término resultó inferior al 20 %. Ese resultado, obtenido décadas antes de la búsqueda web moderna, sigue explicando por qué un sistema que exige coincidencia literal recupera solo una fracción de lo relevante: no se trata de un defecto de implementación, sino de una propiedad estadística del lenguaje natural.

En el comercio electrónico regional latinoamericano, este problema se amplifica por la jerga coloquial, los regionalismos y las abreviaturas, un fenómeno de alternancia y mezcla de registros lingüísticos (*code-switching*) que la literatura reciente de PLN documenta como un desafío abierto incluso para los modelos de lenguaje de gran escala (Sheth et al., 2025). Un emparejador léxico que puntúa la consulta “estuche pal celu xiami” contra la entrada del catálogo “Carcasa protectora Xiaomi Redmi Note 12” encuentra casi ninguna coincidencia de términos y, por tanto, no logra recuperar un producto relevante. Lo mismo ocurre con consultas como “funda pal iphon” frente a “Funda protectora iPhone 13” o “cauchos pal auto” frente a “Neumáticos para vehículo”. Esta brecha entre la intención del usuario y la representación del sistema degrada la precisión de la recuperación y motiva enfoques que operan sobre el significado en lugar de sobre los *tokens* exactos.

Resulta útil descomponer esa brecha en los mecanismos lingüísticos concretos que la producen, porque cada uno impone una exigencia distinta al modelo de recuperación:

- **Contracción y elisión.** Formas como “pal” (por *para el*) suprimen material fonológico que el catálogo conserva íntegro. El *token* resultante no existe en el vocabulario formal del corpus de entrenamiento genérico.
- **Apócope y acortamiento.** “celu” por *celular*, “refri” por *refrigeradora*, “compu” por *computadora*. El acortamiento es productivo y altamente frecuente en el habla informal, pero prácticamente ausente de la redacción comercial.
- **Variación léxica regional.** Un mismo referente recibe nombres distintos según el país: *cauchos* en Venezuela, *llantas* en México y Ecuador, *neumáticos* en la redacción técnica;

celular en América frente a *móvil* en España. La discrepancia no ocurre solo entre usuario y catálogo, sino entre usuarios de distintas regiones que consultan el mismo catálogo. El caso de *llanta* ilustra además una dificultad adicional y más traicionera: en el catálogo empleado en este trabajo, *llanta* y *neumático* no son dos nombres del mismo objeto sino dos categorías distintas — *llanta de aleación* designa el aro metálico y *neumático* el caucho que lo recubre (Sección 3.2.6)—, de modo que la consulta coloquial “llantas pal carro” de un usuario ecuatoriano es genuinamente ambigua: el término que él usa como sinónimo de *neumático* nombra, en el catálogo, un producto diferente. La variación regional no solo separa al usuario del catálogo; a veces lo enfrenta a una homonimia real.

- **Anglicismos y su adaptación.** “cover”, “mouse”, “laptop” conviven con sus equivalentes en español dentro del mismo mercado, y con frecuencia dentro de la misma consulta.
- **Variación ortográfica y de marca.** La ausencia de tildes en el tecleo rápido (“bateria”, “telefono”), y la escritura aproximada de nombres propios extranjeros (“xiami” por *Xiaomi*, “iphon” por *iPhone*), rompen la coincidencia exacta justo en los términos de mayor poder discriminativo.

Este último mecanismo merece atención particular. Chari et al. (Chari et al., 2023) mostraron experimentalmente que las convenciones ortográficas regionales degradan el desempeño de los modelos de recuperación (incluidos los neuronales) cuando el registro de la consulta se aparta del registro sobre el que fueron entrenados, y que el efecto no se corrige simplemente aumentando el tamaño del modelo. En la misma línea, el desarrollo de codificadores preentrenados específicos para el español confirma que el registro importa: BETO (Cañete et al., 2020) fue entrenado sobre un corpus general en español, mientras que RoBERTuito (Pérez et al., 2022) se preentrenó sobre más de 500 millones de tuits y supera a los modelos generales precisamente en texto informal generado por usuarios, además de mostrar competencia en tareas de alternancia de código. Ambos son, sin embargo, codificadores orientados a clasificación y no a recuperación: producen representaciones contextuales por *token* que no están calibradas para comparar la similitud entre dos textos completos, que es la operación que exige un buscador semántico.

Las respuestas clásicas a la discrepancia de vocabulario dentro del paradigma léxico han sido la expansión de consulta, la corrección ortográfica y los diccionarios de sinónimos. Las tres comparten una limitación estructural en este dominio: requieren mantenimiento manual y

suponen un vocabulario relativamente estable. Un diccionario de sinónimos debería enumerar explícitamente que “celu”, “cel”, “telefono” y “smartphone” designan la misma familia de productos, y hacerlo para cada país, cada categoría y cada marca nueva que entre al catálogo; la corrección ortográfica, por su parte, tiende a fallar justamente en los nombres propios, que no figuran en ningún diccionario de referencia y donde la “corrección” puede alterar el término que identifica el producto. La alternativa que explora este trabajo consiste en no enumerar equivalencias, sino aprender un espacio de representación en el que la proximidad semántica se induzca a partir de ejemplos. Esa alternativa se apoya en las representaciones densas que se describen a continuación.

Embeddings y Modelos Transformer

Los *embeddings* de palabras (Mikolov et al., 2013) representan los *tokens* como vectores densos en un espacio continuo donde los elementos semánticamente relacionados se ubican próximos entre sí. Su fundamento es la hipótesis distribucional: las palabras que aparecen en contextos similares tienden a tener significados similares, de modo que el significado puede aproximarse a partir de la coocurrencia observada en un corpus suficientemente grande, sin supervisión explícita. Los modelos de esta primera generación, sin embargo, asignaban un único vector a cada tipo léxico, lo que los hacía incapaces de distinguir los sentidos de una palabra polisémica (*cargador* de teléfono frente a *cargador* de arma) y de representar unidades mayores que la palabra sin recurrir a promedios que diluyen el significado.

La introducción de la arquitectura *Transformer* (Vaswani et al., 2017), basada en el mecanismo de atención, permitió representaciones contextuales que capturan el significado de una palabra en función de su contexto, mejorando notablemente las tareas posteriores de PLN. El componente central es la atención de producto escalado,

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

donde cada *token* proyecta una consulta Q , una clave K y un valor V , y el peso que un *token* asigna a otro se calcula como la afinidad entre su consulta y la clave del otro, normalizada por $\sqrt{d_k}$ para estabilizar los gradientes. A diferencia de las arquitecturas recurrentes, esta operación relaciona todas las posiciones de la secuencia en un solo paso y es completamente paralelizable, lo que hizo viable el preentrenamiento sobre corpus de gran escala.

Modelos preentrenados de gran escala como BERT (Devlin et al., 2019) consolidaron

este paradigma al aprender representaciones bidireccionales profundas a partir de grandes corpus no etiquetados, mediante el objetivo de modelado de lenguaje enmascarado, y sirven como arquitectura base de la que se derivan codificadores más ligeros mediante destilación, como el empleado en este trabajo. Ahora bien, un hallazgo importante para esta investigación es que BERT no produce, sin adaptación, buenos vectores de oración: promediar sus estados ocultos o tomar el *token* [CLS] genera representaciones cuya similitud del coseno correlaciona pobremente con el juicio humano de similitud semántica. Reimers y Gurevych (Reimers & Gurevych, 2019) resolvieron esta limitación con las *Sentence-BERT*: una arquitectura siamesa que codifica dos textos con pesos compartidos, aplica agrupamiento por promedio sobre los estados ocultos y se entrena con un objetivo que optimiza directamente la comparabilidad de los vectores resultantes. Sobre esta base, los codificadores a nivel de oración proyectan una consulta o documento completo en un único vector de longitud fija, de modo que la similitud semántica puede calcularse directamente como la similitud del coseno

$$\text{sim}(q, d) = \frac{\mathbf{v}_q \cdot \mathbf{v}_d}{\|\mathbf{v}_q\| \|\mathbf{v}_d\|}, \quad (2.3)$$

que mide únicamente el ángulo entre ambos vectores y es invariante a su magnitud. Este paradigma de recuperación densa hace posible recuperar productos aun cuando no exista coincidencia literal entre la consulta y el documento, y constituye la base de la búsqueda semántica moderna.

Dentro de la recuperación neuronal conviven tres arquitecturas con perfiles de costo y calidad muy distintos, y la elección entre ellas condiciona todo el diseño del sistema. El *bi-codificador* (el adoptado en este trabajo) codifica consulta y documento por separado, lo que permite precalcular e indexar los vectores del catálogo y reduce la búsqueda a una comparación vectorial en tiempo de consulta. El *codificador cruzado* procesa conjuntamente el par consulta–documento en una sola pasada del *Transformer*, alcanzando mayor precisión gracias a la atención cruzada entre ambos textos, pero obligando a una inferencia por cada candidato, lo que lo restringe al reordenamiento de una lista corta. La *interacción tardía*, ejemplificada por ColBERTv2 (Santhanam et al., 2022), ocupa una posición intermedia: conserva un vector por *token* y calcula la relevancia mediante una suma de máximos de similitud, con mejor calidad que el bi-codificador a costa de un índice considerablemente mayor. La arquitectura de bi-codificador se justifica en este trabajo por el objetivo explícito de mantener la latencia por

debajo de 100 ms sobre *hardware* de CPU.

El panorama de modelos de *embeddings* ha evolucionado con rapidez en los últimos años. Las revisiones sistemáticas del área (Cao, 2024) documentan que las ganancias recientes provienen menos de la arquitectura que de la calidad y escala de los datos de entrenamiento contrastivo y del uso de modelos de lenguaje de gran escala para generarlos. En paralelo, han aparecido técnicas que flexibilizan el uso del vector resultante: el aprendizaje de representaciones *Matryoshka* (Kusupati et al., 2022) entrena el codificador de modo que los primeros k componentes del vector conserven por sí solos la mayor parte de la información, permitiendo truncar la dimensión en tiempo de despliegue para reducir memoria y latencia sin reentrenar. La evaluación comparativa, por su parte, se ha ampliado desde MTEB (Muennighoff et al., 2023) hasta su versión masivamente multilingüe MMTEB (Enevoldsen et al., 2025), que abarca más de 500 tareas en más de 250 idiomas y permite valorar el desempeño en español de forma independiente del inglés. Estos recursos son los que hacen posible justificar empíricamente la elección del modelo base que se discute en la siguiente sección.

***Fine-tuning* de Modelos de *Embeddings* Ligeros**

Buscando precisión, los codificadores grandes resultan ser muy costosos computacionalmente. Un modelo como MiniLM (W. Wang et al., 2020) destaca por ser un modelo destilado que conserva la mayor parte de la calidad representacional utilizando muchos menos parámetros. Sustancialmente, la destilación de conocimiento en la que se apoya no consiste en reentrenar un modelo pequeño sobre las mismas etiquetas. Consiste, en cambio, en transferir el comportamiento interno de un modelo maestro a un modelo estudiante. Aplicando destilación profunda de autoatención, MiniLM logra que el estudiante reproduzca las distribuciones de atención. Es decir, imita las relaciones entre valores de la última capa del maestro en lugar de solo su salida final. Mediante esta forma de supervisión, más rica que la simple etiqueta, se potencia el aprendizaje. Igualmente, esta riqueza estructural permite reducir el número de capas. Logrando a la vez disminuir la dimensión oculta, el sistema se vuelve mucho más ligero. Incluso con estos cambios, todo se realiza con una pérdida de calidad apenas moderada. Ofreciendo así un balance óptimo, el diseño mantiene un alto nivel de representación. Cada patrón transferido refuerza la eficiencia del modelo resultante. Un marco de trabajo tan compacto facilita la inferencia rápida. Esta estrategia demuestra el valor de capturar la atención interna. No se limita a imitar predicciones superficiales. Consigue transmitir el razonamiento completo de la arquitectura compleja. Así, el estudiante alcanza un desempeño sólido. Jus-

to por este motivo, los modelos destilados son ampliamente adoptados. Una implementación eficiente permite desplegarlos en diversos entornos. Además, se reducen notablemente los requerimientos de *hardware*. Nada de esto compromete la capacidad de representación general. De esta manera, el modelo pequeño destaca en rendimiento. Asegurando rapidez sin perder precisión, el diseño responde bien. El resultado es una compresión efectiva y robusta. La destilación profunda se consolida así como un pilar fundamental. Por ello, MiniLM constituye un referente claro en la literatura. Resolviendo el dilema entre costo computacional y calidad. Implementarlo asegura un equilibrio sobresaliente. Manteniendo las capacidades clave del maestro, se logran grandes avances. Es un enfoque sumamente valioso para el área. Reduciendo la complejidad, se obtienen soluciones escalables. Optimizar sistemas complejos es finalmente posible.

En términos sencillos: MiniLM-L6-v2 es el “cerebro” de inteligencia artificial que usa este proyecto para entender el significado de una consulta o de un producto. Se eligió esta variante, y no un modelo más grande, precisamente por ser ultraligera (pesa menos de 90 MB, frente a varios cientos de megabytes o incluso gigabytes de los modelos más grandes), por lo que puede ejecutarse en un servidor convencional sin necesidad de una tarjeta gráfica (GPU).

La variante MiniLM-L6-v2 produce *embeddings* de 384 dimensiones con aproximadamente 22 millones de parámetros y un tamaño cercano a 90 MB, lo que la hace adecuada para la inferencia en CPU en entornos de producción sensibles a la latencia. Modelos más recientes y de mayor escala, como la familia multilingüe E5 (L. Wang et al., 2024), alcanzan una calidad representacional superior en bancos de pruebas estandarizados como MTEB (Muennighoff et al., 2023) (*Massive Text Embedding Benchmark*, la evaluación de referencia de la comunidad para comparar modelos de *embeddings* entre sí), pero a costa de decenas o cientos de millones de parámetros adicionales; la elección de MiniLM-L6-v2 en este trabajo prioriza deliberadamente la eficiencia de inferencia en CPU sobre la calidad representacional marginal adicional de esos modelos más pesados, en línea con el objetivo de mantener el sistema accesible para comercios con infraestructura modesta.

Esta decisión debe entenderse como un punto dentro de un compromiso continuo y no como una elección atemporal. La frontera de los modelos compactos se ha desplazado con rapidez: EmbeddingGemma (Schechter Vera et al., 2025), por ejemplo, alcanza con 300 mi-

llones de parámetros un desempeño competitivo con codificadores casi el doble de grandes, e incorpora representaciones *Matryoshka* para permitir truncar la dimensión del vector según el presupuesto de memoria disponible. La consecuencia metodológica es que la comparación relevante no es “modelo pequeño frente a modelo grande” en abstracto, sino calidad de recuperación por milisegundo de latencia y por megabyte de memoria en el *hardware* concreto de despliegue. Sobre esa base, un codificador de 22 millones de parámetros ajustado al dominio puede resultar preferible a uno genérico de 300 millones, que es precisamente la hipótesis que este trabajo somete a prueba.

No obstante, los modelos de *embeddings* de propósito general se entrenan sobre corpus formales y, por tanto, rinden por debajo de lo esperado en el lenguaje coloquial específico de un dominio. El *fine-tuning* supervisado adapta un modelo base a un dominio objetivo mediante pares etiquetados. Un objetivo ampliamente utilizado para este propósito, dentro del marco de las *Sentence-Transformers* (Reimers & Gurevych, 2019; Sentence-Transformers, 2026) y el aprendizaje contrastivo, es la *Multiple Negatives Ranking Loss* (MNRL, pérdida de ordenamiento con múltiples negativos) (Henderson et al., 2017), que, dado un lote de pares ancla–positivo, trata cada otro positivo del lote como un ejemplo negativo. Esto incentiva al modelo a acercar los *embeddings* de los pares consulta–producto coincidentes y a alejar los no coincidentes, sin necesidad de minar explícitamente negativos difíciles. Los estudios de adaptación de dominio reportan que este *fine-tuning* suele producir ganancias de entre el 10 % y el 25 % en tareas de recuperación especializadas (Gururangan et al., 2020; Pandey, 2026; Thakur et al., 2021).

Formalmente, para un lote de N pares (q_i, d_i) , MNRL minimiza la entropía cruzada de una distribución *softmax* construida sobre las similitudes de la ancla con todos los positivos del lote:

$$\mathcal{L} = -\frac{1}{N} \sum_{i=1}^N \log \frac{\exp(\text{sim}(q_i, d_i)/\tau)}{\sum_{j=1}^N \exp(\text{sim}(q_i, d_j)/\tau)}, \quad (2.4)$$

donde τ es una temperatura que escala las similitudes y controla cuán exigente resulta la separación entre el positivo y los negativos. Una consecuencia práctica de esta formulación, relevante para la configuración experimental, es que el tamaño del lote no es un mero parámetro de eficiencia: cada ejemplo adicional del lote aporta un negativo más al denominador,

de modo que lotes mayores plantean una tarea de discriminación más difícil y suelen producir representaciones mejores.

La contrapartida de la simplicidad de MNRL es que sus negativos son *aleatorios*: proceden de los demás pares del lote y, en un catálogo diverso, tienden a ser fáciles de descartar (una consulta de neumáticos frente a un cargador), con lo que el gradiente aporta poca información una vez que el modelo ha aprendido las distinciones gruesas. La literatura ha abordado esta limitación mediante la minería de *negativos difíciles*: ejemplos que el propio modelo considera cercanos pero que no son relevantes. ANCE (Xiong et al., 2020) propuso construir esos negativos consultando periódicamente un índice de vecinos aproximados actualizado con el modelo en curso, de modo que la dificultad de los negativos crezca a la par del modelo. Este refinamiento queda fuera del alcance del presente trabajo y se identifica en el Capítulo V como una línea de mejora directa.

Un segundo eje de la literatura aborda el caso en que no se dispone de pares etiquetados en el dominio objetivo, que es la situación habitual de un comercio pequeño. GPL (K. Wang et al., 2022) combina un generador de consultas con el pseudoetiquetado de un codificador cruzado para adaptar un recuperador denso sin supervisión humana, y reporta mejoras de hasta 9.3 puntos de NDCG@10 sobre dominios especializados. En la misma dirección, los enfoques de generación sintética con modelos de lenguaje han evolucionado desde las primeras propuestas hasta esquemas más elaborados de filtrado y control de calidad de los pares generados (Krastev et al., 2025). El conjunto de entrenamiento empleado en este trabajo es asimismo sintético (Sección 3.2), lo que sitúa la investigación en esa línea, pero también hereda su principal riesgo metodológico: las consultas generadas pueden no reproducir la distribución real de formulaciones de los usuarios, y el modelo resultante puede sobreajustarse a los patrones del generador antes que al habla efectiva de los compradores. Esta limitación se retoma explícitamente en la Sección 4.6.2.

Finalmente, el *fine-tuning* no tiene por qué actualizar todos los parámetros del modelo. Las técnicas de *fine-tuning* eficiente en parámetros, y en particular LoRA (Hu et al., 2021), congelan los pesos del modelo base e insertan matrices de adaptación de rango bajo, reduciendo en órdenes de magnitud el número de parámetros entrenables y el costo de almacenar una variante por dominio. En un codificador de 22 millones de parámetros el ahorro es menos determinante que en un modelo de miles de millones, razón por la cual este trabajo realiza un *fine-tuning* completo; la técnica resulta pertinente, en cambio, si en el futuro se desea mantener

variantes separadas por país o por vertical de catálogo sobre un mismo modelo base.

Justificación Comparada de las Decisiones de Modelado

Las secciones anteriores describen el estado del arte; esta sección lo emplea para un propósito distinto: argumentar por qué, entre las alternativas disponibles, se eligieron las que componen el sistema propuesto. La justificación se organiza en tres ejes de decisión que la literatura trata de forma independiente y que en este trabajo se resuelven de manera acoplada: el codificador de partida para el español informal, la arquitectura de recuperación densa, y la dimensionalidad de la representación almacenada en la base vectorial. En los tres casos la decisión no responde a una superioridad absoluta de una opción sobre otra, sino a un compromiso explícito entre calidad de recuperación y costo operativo bajo la restricción de inferencia en CPU que define el alcance de esta investigación.

Primer Eje: Codificador Preentrenado para el Español Informal

El primer eje enfrenta dos codificadores preentrenados para el español que difieren esencialmente en el registro lingüístico de su corpus de preentrenamiento. BETO (Cañete et al., 2020) es un modelo de arquitectura BERT-base entrenado sobre un corpus general en español y constituye la línea de referencia habitual para tareas de PLN en este idioma. RoBERTuito (Pérez et al., 2022), en cambio, se preentrenó sobre más de 500 millones de tuits y fue evaluado explícitamente frente a modelos monolingües tradicionales en tareas sobre contenido generado por usuarios; sus autores reportan un desempeño superior en ese dominio y competencia adicional en escenarios de alternancia de código, precisamente los fenómenos (abreviaturas, faltas ortográficas, sintaxis informal) que caracterizan el objeto de esta tesis.

La conclusión inmediata sería adoptar RoBERTuito. Sin embargo, esa conclusión pasa por alto una distinción arquitectónica que resulta determinante. Tanto BETO como RoBERTuito son codificadores entrenados con modelado de lenguaje enmascarado, cuyo producto natural es una representación contextual *por token*, adecuada para clasificación de secuencias, etiquetado o extracción de entidades. Ninguno de los dos es, sin adaptación, un modelo de similitud de oraciones. Como estableció el trabajo fundacional de las *Sentence-BERT* (Reimers & Gurevych, 2019), los vectores que un codificador de este tipo produce sin entrenamiento contrastivo correlacionan pobremente con el juicio humano de similitud semántica, de modo que emplearlos directamente en un bi-codificador de recuperación produce resultados inferiores incluso a métodos léxicos clásicos. Utilizar RoBERTuito como recuperador exigiría, por tanto, una etapa adicional de entrenamiento contrastivo sobre pares consulta–documento en

español, es decir, exactamente el mismo procedimiento que este trabajo aplica sobre MiniLM, pero partiendo de un modelo con cinco veces más parámetros.

La Tabla I resume el compromiso. La decisión adoptada consiste en sustituir la afinidad de registro obtenida por preentrenamiento (la vía de RoBERTuito) por afinidad de registro obtenida mediante *fine-tuning* supervisado sobre pares coloquiales, sobre un codificador ya preparado para similitud de oraciones. Es una sustitución de un costo de preentrenamiento, inasumible para este proyecto, por un costo de *fine-tuning* de dos órdenes de magnitud menor.

Tabla I
Primer eje de decisión: codificador de partida para el español informal

Opción	Parám.	Ventaja	Costo o limitación
BETO	110 M	Cobertura amplia del español formal; referencia consolidada	Registro de preentrenamiento distante de la jerga; requiere etapa contrastiva adicional
RoBERTuito	110 M	Afinidad demostrada con texto informal y alternancia de código	No es un modelo de similitud de oraciones; 5× el tamaño del modelo adoptado
MiniLM-L6-v2 (adoptado)	22 M	Bi-codificador listo para similitud; inferencia viable en CPU	Preentrenamiento mayoritariamente en inglés y en registro formal

Fuente: Elaboración propia, a partir de (Cañete et al., 2020; Pérez et al., 2022; Reimers & Gurevych, 2019; W. Wang et al., 2020).

Nota: El recuento de parámetros corresponde a la arquitectura base declarada por los autores de cada modelo.

Esta decisión es falsable, y el diseño experimental del Capítulo IV la somete a prueba de forma indirecta: si el *fine-tuning* sobre pares coloquiales cumple la función que cumpliría un preentrenamiento en registro informal, la mejora del modelo ajustado debe concentrarse en las consultas que presentan marcadores coloquiales y no distribuirse uniformemente. Queda fuera del alcance de este trabajo, y se identifica en el Capítulo V como línea de continuación, la comparación empírica directa contra un bi-codificador construido sobre RoBERTuito, que sería el experimento decisivo para cuantificar cuánto aporta el registro del corpus de preentrenamiento frente al registro de los pares de *fine-tuning*.

Segundo Eje: Arquitectura de Recuperación Densa

El segundo eje contrapone el codificador ligero adoptado con la familia E5, diseñada específicamente para recuperación. E5 (L. Wang et al., 2022) introdujo un preentrenamiento contrastivo débilmente supervisado sobre un corpus curado de pares textuales a gran escala, y sus autores reportan que fue el primer modelo de su clase en superar a BM25 en recuperación sin *fine-tuning* específico al dominio, un umbral que la literatura previa no había alcanzado de forma consistente. Su extensión multilingüe (L. Wang et al., 2024) amplió esa capacidad a más de cien idiomas, y la evaluación masivamente multilingüe MMTEB (Enevoldsen et al., 2025) sitúa a *multilingual-e5-large-instruct*, con 560 millones de parámetros, como el modelo público de mejor desempeño de su conjunto de tareas.

El rasgo de diseño de E5 más pertinente para esta tesis no es, sin embargo, su escala sino su tratamiento explícito de la **asimetría** de la tarea. Los modelos de esta familia anteponen prefijos distintos a cada lado del par (uno para la consulta y otro para el documento), de modo que el codificador aprende dos funciones de proyección relacionadas pero no idénticas. Esa asimetría es exactamente la que caracteriza el problema aquí estudiado: la consulta es un fragmento coloquial de entre tres y siete palabras, mientras que el documento es una descripción técnica de entre siete y veintidós palabras redactada en un registro distinto (Sección 3.2). Un bi-codificador simétrico como MiniLM-L6-v2 proyecta ambos lados con la misma función y debe, por tanto, aprender a salvar esa distancia de registro enteramente durante el *fine-tuning*.

Reconocer esta ventaja teórica de E5 obliga a justificar por qué no se adoptó, y conviene precisar el argumento porque no todas las variantes de la familia imponen el mismo costo. La Tabla II contrasta las magnitudes en juego. La variante más pequeña de la familia *multilingüe*, *multilingual-e5-small*, multiplica por unas cinco veces el número de parámetros del modelo adoptado, pero produce vectores de 384 dimensiones, exactamente las mismas que MiniLM-L6-v2: el costo adicional se concentra en el tiempo de inferencia en CPU y no en la memoria del índice vectorial. La duplicación de la dimensión (768) corresponde a *e5-base*, que es un modelo *monolingüe en inglés* y por tanto inaplicable a este problema.

La justificación honesta de la decisión es, en consecuencia, más estrecha de lo que sugeriría un argumento de memoria: se adoptó MiniLM-L6-v2 porque multiplicar por cinco el costo de codificación en CPU no es gratuito en el escenario objetivo, y porque la medición de la Sección 4.5 muestra que la codificación de la consulta es precisamente el componente que domina el presupuesto de latencia. El argumento de memoria del índice, en cambio, no

distingue entre ambos modelos.

Tabla II
Segundo eje de decisión: bi-codificador de recuperación

Modelo	Parám.	Dim.	Característica distintiva
MiniLM-L6-v2 (adoptado)	22 M	384	Destilación profunda de autoatención; emparejamiento simétrico
multilingual-e5-small	118 M	384	Variante multilingüe más pequeña; misma dimensión que el modelo adoptado
e5-base (solo inglés)	110 M	768	Preentrenamiento contrastivo débilmente supervisado; prefijos <i>query/passage</i>
multilingual-e5-large-instruct	560 M	1024	Mejor desempeño público en MMTEB; cobertura de más de 100 idiomas

Fuente: Elaboración propia, a partir de (Enevoldsen et al., 2025; L. Wang et al., 2022, 2024; W. Wang et al., 2020) y de las fichas públicas de cada modelo.

Nota: La dimensión determina de forma lineal la memoria del índice vectorial y el costo de cada cálculo de distancia; el número de parámetros determina el costo de codificar cada consulta. Nótese que ambos factores no van juntos: `multilingual-e5-small` multiplica los parámetros sin aumentar la dimensión, mientras que `e5-base` aumenta la dimensión pero no admite español.

La formulación correcta del criterio de selección no es, en consecuencia, cuál modelo obtiene la mejor métrica de recuperación en abstracto, sino cuál ofrece la mejor calidad por unidad de latencia y de memoria en el *hardware* de despliegue. Bajo ese criterio, un codificador de 22 millones de parámetros ajustado al dominio compite con uno genérico varias veces mayor, y los resultados del Capítulo IV muestran que la variante ajustada supera a BM25 en el escenario evaluado, con datos sintéticos. Esa evidencia respalda la decisión para la escala examinada, pero no la extiende: a medida que el catálogo y la diversidad de consultas crezcan, el margen de calidad de un modelo de mayor capacidad podría justificar su costo adicional, y la comparación directa contra `multilingual-e5-small` queda señalada como el siguiente experimento natural.

Tercer Eje: Dimensionalidad de la Representación Almacenada

El tercer eje concierne a una decisión que suele quedar implícita: cuántas dimensiones almacenar por producto en la base vectorial. En un índice denso, tanto la memoria residente

como el costo de cada cálculo de distancia crecen linealmente con la dimensión d del vector, de modo que para un catálogo de N productos el consumo de memoria del índice es del orden de $N \cdot d$ valores en coma flotante. Reducir d a la mitad reduce a la mitad ambos costos; el problema es que truncar sin más las componentes de un vector convencional destruye información, porque nada en el entrenamiento garantiza que las primeras coordenadas concentren la mayor parte del significado.

El aprendizaje de representaciones *Matryoshka* (MRL) (Kusupati et al., 2022) resuelve precisamente ese problema. Su aporte consiste en modificar el objetivo de entrenamiento para que el modelo optimice simultáneamente varios prefijos anidados del vector (por ejemplo las primeras 64, 128, 256 y 768 coordenadas), de manera que cada prefijo constituya por sí mismo una representación utilizable. El resultado es un vector cuyo truncamiento en tiempo de inferencia degrada la calidad de forma suave y controlada en lugar de abruptamente, lo que permite decidir el compromiso entre precisión y costo *después* del entrenamiento y sin reentrenar el modelo. La técnica se ha incorporado desde entonces a modelos de recuperación recientes: BGE-M3 (Chen et al., 2024) la combina con recuperación densa, dispersa y multivectorial en un mismo codificador, y EmbeddingGemma (Schechter Vera et al., 2025) la emplea para permitir que un mismo modelo de 300 millones de parámetros se despliegue con distintos presupuestos de memoria.

La consecuencia para este trabajo es doble y conviene enunciarla con precisión.

En primer lugar, el modelo adoptado no admite truncamiento. MiniLM-L6-v2 no fue entrenado con un objetivo *Matryoshka*, por lo que sus 384 dimensiones no están jerarquizadas: conservar únicamente las primeras 128 no produce una representación degradada de forma proporcional sino una representación arbitrariamente mutilada. Esta observación corrige una confusión frecuente en la literatura aplicada, donde la reducción de dimensionalidad por truncamiento se presenta a veces como una operación universalmente disponible cuando en realidad depende de cómo se entrenó el codificador. La reducción de dimensionalidad de un vector convencional exige métodos de proyección como el PCA descrito en la Sección 2.10, que requieren ajustar y almacenar la transformación y no preservan la métrica de similitud original.

En segundo lugar, la dimensionalidad no es la restricción activa a la escala evaluada. El corpus de este trabajo consta de 833 productos representados en 384 dimensiones con precisión simple, lo que equivale a aproximadamente 1.3 MB de vectores: un tamaño para el

que cualquier elección de d resulta irrelevante. El compromiso que MRL resuelve solo se vuelve determinante varios órdenes de magnitud más arriba (un catálogo de un millón de productos ocuparía cerca de 1.5 GB únicamente en vectores densos, y uno de diez millones excedería la memoria de un servidor convencional), que es exactamente el régimen en el que los sistemas industriales revisados en la Sección 2.6 operan. Incorporar un objetivo Matryoshka al *fine-tuning* es, por tanto, una mejora cuyo beneficio aparece al escalar y no en el escenario aquí evaluado; se registra como línea de trabajo futuro con esa condición explícita.

Tabla III
Tercer eje de decisión: dimensionalidad y costo de almacenamiento

Catálogo	Dim.	Memoria	Régimen
833 productos (este trabajo)	384	≈ 1.3 MB	La dimensión no condiciona el diseño
10^5 productos	384	≈ 154 MB	Sigue siendo manejable en memoria
10^6 productos	384	≈ 1.5 GB	El truncamiento empieza a ser rentable
10^7 productos	384	≈ 15 GB	Exige cuantización, truncamiento o índice en disco

Fuente: Elaboración propia. Memoria estimada como $N \times d \times 4$ bytes (coma flotante de precisión simple), sin contar la estructura del índice HNSW ni la carga útil (*payload*) asociada a cada vector.

Nota: Las cifras ilustran el orden de magnitud en el que las técnicas de reducción (Matryoshka (Kusupati et al., 2022) o cuantización de producto (Jégou et al., 2011)) dejan de ser una optimización opcional.

Síntesis del Compromiso Técnico

Los tres ejes examinados comparten una estructura común: en cada uno existe una opción de mayor calidad representacional y un costo asociado que la restricción de despliegue vuelve prohibitivo. Presentar las decisiones de este trabajo como elecciones óptimas en sentido absoluto sería incorrecto; lo que puede sostenerse es que son óptimas condicionalmente al criterio declarado, que es la viabilidad sobre *hardware* de CPU accesible para un comercio de escala media en la región estudiada.

La Tabla IV consolida las tres decisiones, el criterio que gobernó cada una y la evidencia que permite evaluarla.

Tabla IV
Síntesis de las decisiones de modelado y su criterio de justificación

Eje	Decisión y criterio	Evidencia o verificación
Codificador	MiniLM-L6-v2 con <i>fine-tuning</i> coloquial, en lugar de RoBERTa: la afinidad de registro se obtiene por <i>fine-tuning</i> y no por preentrenamiento	Desglose por fenómeno lingüístico del Capítulo IV: la mejora debe concentrarse en las consultas con marcadores coloquiales
Arquitectura	Bi-codificador simétrico ligero, en lugar de la familia E5: mejor calidad por milisegundo de latencia en CPU	Comparación contra el modelo base y contra BM25 del Capítulo IV, sobre partición reservada
Dimensionalidad	384 dimensiones fijas, sin objetivo Matryoshka: a la escala evaluada dimensión no es la restricción activa	Estimación de memoria de la Tabla III; la decisión se revisa por encima de 10^6 productos

Fuente: Elaboración propia.

Una observación final sobre el método. Las tres decisiones se tomaron antes de disponer de los resultados experimentales y se justifican aquí por argumentos de diseño, no por su desempeño posterior; presentarlas al revés (seleccionar la justificación que concuerda con el resultado obtenido) constituiría una forma de razonamiento posterior a los hechos que la literatura de evaluación experimental identifica como fuente sistemática de conclusiones infundadas (Fuhr, 2017). Los criterios enunciados en esta sección son, por tanto, verificables de forma independiente del resultado: la restricción de *hardware*, el número de parámetros y la dimensión del vector son propiedades declaradas de cada alternativa, no mediciones de este trabajo.

Bases de Datos Vectoriales y Recuperación Semántica

Una vez que los documentos se codifican como vectores, la recuperación se convierte en un problema de búsqueda del vecino más cercano. La formulación exacta (comparar la consulta con los N vectores del catálogo y quedarse con los k más próximos) tiene costo lineal en N y, en dimensiones altas, no admite las estructuras de partición del espacio que resultan eficaces en dos o tres dimensiones: a medida que crece la dimensión, las distancias entre puntos aleatorios se concentran y el volumen que hay que examinar para garantizar el óptimo se aproxima al de la colección completa. De ahí que la práctica renuncie a la garantía de exactitud y adopte la búsqueda *aproximada*, que acepta omitir ocasionalmente algún

vecino verdadero a cambio de una reducción de varios órdenes de magnitud en el tiempo de respuesta. El parámetro de diseño deja de ser entonces la corrección y pasa a ser la curva que relaciona la exhaustividad (*recall*) alcanzada con el número de consultas atendidas por segundo.

Las bases de datos vectoriales indexan los *embeddings* mediante estructuras de vecino más cercano aproximado, como los grafos HNSW (Y. Lin et al., 2025; Malkov & Yashunin, 2018) (*Hierarchical Navigable Small World*), para devolver los vectores más similares en tiempo sublineal. Qdrant y FAISS destacan como tecnologías relevantes para la indexación y la búsqueda aproximada, esta última sustentada en los métodos de cuantización e indexación descritos por Johnson et al. (Johnson et al., 2019).

Las familias de índices disponibles responden a compromisos distintos. Los índices basados en *archivo invertido* (IVF) agrupan los vectores en celdas mediante un cuantizador grueso y restringen la búsqueda a las celdas más próximas a la consulta, con un costo de memoria bajo y una calidad que depende del número de celdas exploradas. La *cuantización de producto* (PQ) (Jégou et al., 2011) descompone cada vector en subvectores y sustituye cada uno por el identificador de su centroide en un pequeño diccionario, comprimiendo la representación en uno o dos órdenes de magnitud y permitiendo calcular distancias aproximadas mediante tablas precalculadas; su costo es una pérdida de precisión que debe compensarse con un reordenamiento posterior sobre los vectores completos. Los índices *basados en grafos*, como HNSW, construyen una jerarquía de grafos de proximidad y descienden desde una capa dispersa hasta una densa siguiendo en cada paso al vecino más cercano a la consulta; ofrecen la mejor curva exhaustividad–rendimiento entre los métodos en memoria, a costa de un consumo de memoria elevado y de un tiempo de construcción considerable. Cuando la colección excede la memoria disponible, DiskANN (Subramanya et al., 2019) demostró que es posible sostener grafos de miles de millones de puntos en un solo nodo manteniendo el grafo en disco y una versión comprimida de los vectores en memoria.

En el caso de HNSW, tres parámetros gobiernan ese compromiso y conviene documentarlos al reportar resultados: M , el número de aristas por nodo, que fija la densidad del grafo y con ella la memoria consumida; *efConstruction*, el tamaño de la lista de candidatos durante la construcción, que determina la calidad del grafo resultante; y *efSearch*, su equivalente en tiempo de consulta, que permite desplazarse sobre la curva exhaustividad–latencia sin reconstruir el índice. Esta última propiedad es operativamente valiosa, porque habilita ajustar la

calidad de la búsqueda en producción según la carga.

En términos sencillos: una base de datos vectorial como Qdrant (Qdrant, 2026) es un motor especializado en guardar y buscar “coordenadas matemáticas” (los vectores de 384 números que produce el modelo de *embeddings*) y encontrar, en milisegundos, cuáles son las más parecidas entre sí, aun cuando existan cientos de miles de ellas. FAISS (Johnson et al., 2019) es una biblioteca alternativa de búsqueda vectorial que opera directamente en memoria; en este trabajo se emplea únicamente como punto de comparación arquitectónico frente a Qdrant (Sección 3.4.1), no como componente de producción.

Un requisito propio del comercio electrónico, y que no aparece en los bancos de pruebas académicos de búsqueda vectorial pura, es la necesidad de combinar la similitud semántica con restricciones sobre atributos: precio máximo, categoría, marca, disponibilidad en inventario. Esta *búsqueda filtrada* es sustancialmente más difícil de lo que sugiere su enunciado. Filtrar después de recuperar puede devolver menos de k resultados si el filtro es selectivo, y filtrar antes obliga a recorrer el índice de un subconjunto que el grafo no fue construido para navegar, degradando la exhaustividad. La literatura reciente sistematiza las estrategias propuestas para este problema híbrido vector–escalar (Y. Lin et al., 2025), y su tratamiento nativo es una de las razones por las que este trabajo adopta un motor vectorial con soporte de carga útil en lugar de una biblioteca de indexación pura.

Dos patrones arquitectónicos son relevantes para este trabajo. En un patrón de *dos pasos*, un índice vectorial (por ejemplo, FAISS) devuelve identificadores candidatos y se consulta una base de datos relacional independiente para obtener los metadatos del producto. En un patrón *integrado*, la base de datos vectorial almacena la carga útil del producto junto a cada vector, de modo que una única consulta devuelve tanto el ordenamiento como los metadatos, eliminando un viaje de ida y vuelta por la red. La literatura reciente muestra además que los sistemas densos e híbridos pueden superar a BM25 en calidad de ordenamiento, aunque con costos distintos de latencia y de recursos (Thakur et al., 2021). El sistema de este trabajo adopta el patrón integrado usando Qdrant con la expectativa de reducir dicho costo de latencia, hipótesis que la medición del Capítulo IV no confirma (Sección 4.5).

La diferencia entre ambos patrones no es solo de latencia media, sino de propiedades operativas. El esquema de dos pasos introduce dos sistemas de almacenamiento que

deben mantenerse sincronizados: un producto dado de baja en la base relacional puede seguir figurando en el índice vectorial hasta la siguiente reindexación, produciendo resultados que apuntan a registros inexistentes. El esquema integrado traslada esa consistencia al propio motor vectorial, a cambio de acoplar el ciclo de vida de los metadatos al del índice. A escala de despliegue, ambas alternativas se insertan además en discusiones más amplias sobre la organización de servicios de aprendizaje automático en producción, donde la separación entre el servicio de inferencia del modelo y el servicio de búsqueda condiciona la capacidad de escalar cada componente de forma independiente (Karanam & Bhargav, 2026).

Es importante señalar que la combinación de un codificador preentrenado (o ajustado) con una base de datos vectorial para la recuperación, el mismo patrón adoptado en este trabajo y que en la práctica constituye el componente de recuperación de una arquitectura de tipo *Retrieval-Augmented Generation* (RAG) en su forma más elemental, no es una propuesta aislada ni exclusiva de esta investigación. El propio paradigma de recuperación densa mediante un codificador dual y un índice vectorial fue formalizado por Karpukhin et al. (Karpukhin et al., 2020) con *Dense Passage Retrieval*, y generalizado como componente de recuperación dentro de arquitecturas RAG por Lewis et al. (Lewis et al., 2020). Más allá del ámbito académico, este mismo patrón arquitectónico ha sido adoptado de forma independiente por múltiples sistemas de búsqueda de productos en producción a gran escala, cada uno con una pila tecnológica distinta: Walmart combina un modelo de *embeddings* propio con su propia infraestructura de recuperación vectorial (J. Lin et al., 2024; Magnani et al., 2022); Meta implementa un esquema equivalente para Facebook Marketplace (He et al., 2023); y Etsy adopta un modelo de *embeddings* unificado con su propio índice vectorial para la personalización (Jha et al., 2023). Que instituciones académicas y empresas de comercio electrónico de gran escala converjan de manera independiente hacia el mismo patrón arquitectónico (codificador preentrenado o ajustado más índice vectorial) con implementaciones tecnológicas distintas entre sí, indica que el diseño adoptado en este trabajo sigue un patrón establecido, más allá de las tecnologías puntuales (MiniLM, Qdrant) empleadas para instanciarlo; no constituye, por sí sola, evidencia de su desempeño en este dominio.

Procedencia, Confiabilidad y Margen de Error de los Datos

Toda conclusión empírica de este trabajo descansa sobre un conjunto de datos, de modo que la calidad de esa conclusión no puede exceder la calidad de su insumo. La ingeniería dispone desde hace décadas de marcos formales para razonar sobre esa dependencia (la

teoría del error en metrología y la ingeniería de confiabilidad en el diseño de sistemas), mientras que la práctica del aprendizaje automático ha tendido a documentar con mucho detalle el modelo y con muy poco el dato del que procede. Esta sección adopta explícitamente el primer marco para caracterizar el origen del conjunto de datos empleado, clasificar sus fuentes de error y delimitar hasta dónde alcanzan las afirmaciones que pueden sostenerse a partir de él.

La metrología distingue entre *error* e *incertidumbre*. El error es la diferencia entre el valor medido y el valor verdadero, magnitud que en general no puede conocerse; la incertidumbre es el intervalo dentro del cual se estima que se encuentra ese valor verdadero, y sí puede cuantificarse. La *Guía para la expresión de la incertidumbre de medida* (GUM) (Joint Committee for Guides in Metrology, 2008) estructura esa cuantificación en dos vías complementarias. La **evaluación de Tipo A** obtiene la incertidumbre por análisis estadístico de observaciones repetidas: su expresión natural es la desviación estándar de la media sobre una muestra. La **evaluación de Tipo B** la obtiene por cualquier otro medio (especificaciones del fabricante, certificados de calibración, datos publicados, juicio experto fundamentado) y resulta indispensable precisamente cuando no es posible repetir la medición. La distinción es relevante aquí porque los componentes más importantes de la incertidumbre de este trabajo son de Tipo B: no proceden de la dispersión de mediciones repetidas, sino de decisiones de diseño sobre cómo se construyó el conjunto de datos.

Siempre es pertinente recordar la distinción clásica entre error *aleatorio* y error *sistemático* (Taylor, 1997). Observando el caso aleatorio, este se manifiesta como dispersión en torno a un valor central y disminuye al aumentar el número de observaciones. Frente a esto, el error sistemático desplaza el valor central y *no* se reduce midiendo más veces, sino únicamente identificando y corrigiendo su causa, categoría a la que pertenece un conjunto de datos con sesgo de construcción. Incrementar el número de pares generados bajo el mismo procedimiento solo aumenta la precisión aparente de la estimación sin acercarla al valor verdadero. Aceptar esta asimetría es lo que impide interpretar el tamaño del conjunto de entrenamiento como una garantía de validez.

La ingeniería de confiabilidad aporta una segunda herramienta conceptual. La confiabilidad se define como la probabilidad de que un elemento desempeñe la función requerida, bajo condiciones declaradas y durante un intervalo declarado (O'Connor & Kleyner, 2012); su estimación exige, por tanto, enunciar las condiciones bajo las cuales el sistema fue caracterizado. Dentro de esta disciplina, el *Análisis de Modos de Fallo y sus Efectos* (FMEA), normalizado

en la IEC 60812 (International Electrotechnical Commission, 2018), propone un procedimiento sistemático: enumerar los modos en que cada elemento puede fallar, identificar sus causas y sus efectos sobre el sistema, y priorizarlos según su severidad, ocurrencia y detectabilidad. Trasladado a un conducto de datos, el FMEA invita a inventariar por anticipado las formas en que la generación, el etiquetado y el muestreo pueden degradar el conjunto, en lugar de descubrirlas al interpretar los resultados finales. Este trabajo aplica ese razonamiento de forma cualitativa en la Tabla VI.

En términos sencillos: en ingeniería nadie reporta la medida de una pieza sin declarar la tolerancia del instrumento con que la midió. La sección presente hace lo equivalente con los datos: antes de afirmar que el modelo mejoró en un determinado porcentaje, deja constancia de con qué “instrumento” se generó el conjunto sobre el que se midió esa mejora y qué imprecisiones arrastra ese instrumento.

Desde el lado de los sistemas de información, la calidad del dato se ha estudiado como un constructo multidimensional. Wang y Strong (R. Y. Wang & Strong, 1996) establecieron empíricamente, a partir de la percepción de los consumidores de datos, que la exactitud es solo una de varias dimensiones relevantes, junto con la completitud, la representatividad, la credibilidad, la pertinencia y la oportunidad; un conjunto exacto pero no representativo del fenómeno que pretende describir es, a efectos prácticos, un conjunto defectuoso. La investigación reciente en aprendizaje automático ha llegado a conclusiones convergentes por vía empírica. Sambasivan et al. (Sambasivan et al., 2021) documentaron el fenómeno de las *cascadas de datos* (*data cascades*): problemas originados aguas arriba, en la recolección o el etiquetado, que permanecen invisibles durante el desarrollo y se manifiestan aguas abajo de forma ampliada y difícil de diagnosticar. Northcutt et al. (Northcutt et al., 2021), por su parte, midieron la tasa de etiquetas erróneas en los conjuntos de prueba de referencia más utilizados del área y mostraron que es lo bastante alta como para alterar el ordenamiento entre modelos competidores, lo que implica que incluso las comparaciones publicadas sobre colecciones canónicas heredan una incertidumbre de etiqueta rara vez reportada.

La respuesta metodológica a este problema ha sido la documentación estructurada de la procedencia. Las *fichas técnicas de conjuntos de datos* (*datasheets for datasets*) (Geburu et al., 2021) proponen, por analogía con las hojas de datos de los componentes electrónicos, que todo conjunto declare su motivación, composición, proceso de recolección, preprocesamiento,

usos recomendados y usos desaconsejados. Las *declaraciones de datos* (*data statements*) para procesamiento de lenguaje natural (Bender & Friedman, 2018) añaden a ese esquema los aspectos propiamente lingüísticos: variedad de la lengua, características demográficas de los hablantes, situación comunicativa y procedencia del texto. Ambas propuestas son directamente aplicables a esta investigación, dado que su objeto es una variedad regional específica del español y no el idioma en abstracto.

Un componente de incertidumbre habitualmente omitido es el de la propia etiqueta. Cuando la relevancia de un producto frente a una consulta la determinan anotadores humanos, esa determinación es a su vez una medición sujeta a variabilidad, y la literatura de lingüística computacional dispone de coeficientes específicos (el κ de Cohen, el α de Krippendorff) para cuantificar el acuerdo entre anotadores descontando el acuerdo esperable por azar (Artstein & Poesio, 2008). Un conjunto de evaluación que no reporta acuerdo entre anotadores tiene, en rigor, una incertidumbre de etiqueta desconocida y no acotada.

Finalmente, el caso particular de los conjuntos generados sintéticamente introduce un riesgo propio. Shumailov et al. (Shumailov et al., 2024) demostraron, tanto analítica como experimentalmente, que el entrenamiento recursivo de modelos generativos sobre datos producidos por modelos anteriores conduce al *colapso del modelo*: las colas de la distribución (precisamente los casos infrecuentes) desaparecen primero, y la varianza se contrae hacia la región de mayor densidad. Aunque el conjunto de este trabajo se genera en una sola pasada y no de forma recursiva, el mecanismo subyacente es el mismo que motiva la cautela: lo que un generador no produce, el modelo entrenado sobre su salida no puede aprender. Los esquemas recientes de generación sintética para recuperación de información incorporan por ello etapas explícitas de filtrado y control de calidad de los pares producidos (Krastev et al., 2025).

Queda por examinar una dimensión que la ingeniería de confiabilidad trata de forma explícita y que la práctica del aprendizaje automático suele dejar implícita: el **grado de reutilización** del dato. La IEEE Reliability Society ha sostenido durante décadas que la evidencia sobre la confiabilidad de un elemento se *enriquece* con su reutilización. La práctica recomendada IEEE 1633 (IEEE Reliability Society, 2016) instruye precisamente a incorporar los datos históricos de versiones anteriores y de proyectos similares en la estimación de la confiabilidad de un sistema nuevo, y el cuerpo doctrinal del área sistematiza cómo acumular esa evidencia a lo largo de sucesivos ciclos de medición (Lyu, 1996). Bajo esta lógica, un conjunto de datos consultado muchas veces debería producir estimaciones cada vez mejor caracterizadas: más

observaciones, menor incertidumbre de Tipo A.

Esa conclusión, sin embargo, es válida únicamente mientras las reutilizaciones sean independientes entre sí, y la propia literatura de la Reliability Society es la que delimita esa condición. Malaiya (Malaiya, 1982), en *IEEE Transactions on Reliability*, mostró que cuando los elementos de un sistema redundante comparten una causa común (una fluctuación de alimentación, una variación térmica, un factor ambiental), las fallas dejan de ser estadísticamente independientes, y tratarlas como si lo fueran *sobrestima* de forma sistemática la confiabilidad calculada. El resultado es conceptualmente trasladable al dato: cada evaluación adicional realizada sobre el mismo conjunto no constituye una observación nueva e independiente, sino una repetición acoplada a través del conjunto compartido. La precisión aparente del valor reportado crece con el número de mediciones mientras que la información real disponible no lo hace, que es exactamente la forma que adopta un error sistemático.

La investigación en aprendizaje automático ha llegado por su cuenta al mismo resultado y lo ha formalizado. Dwork et al. (Dwork et al., 2015) demostraron que un conjunto de retención consultado de forma *adaptativa* (es decir, cuando las decisiones posteriores del investigador dependen de los resultados obtenidos sobre él) pierde sus garantías estadísticas, y propusieron mecanismos para acotar esa pérdida. Recht et al. (Recht et al., 2019) midieron la consecuencia empírica sobre un caso real: al construir un conjunto de prueba nuevo siguiendo el mismo procedimiento del original, el desempeño de los modelos publicados cayó de forma generalizada, evidenciando que una fracción de la mejora acumulada durante años correspondía al ajuste progresivo de la comunidad al conjunto de prueba reutilizado y no a una capacidad de generalización superior.

En términos sencillos: medir diez veces una pieza con el mismo calibrador descalibrado no mejora la medida; solo produce diez números muy parecidos entre sí y todos igualmente desplazados. Reutilizar el mismo conjunto de datos para tomar decisiones sucesivas tiene el mismo efecto: aumenta la consistencia aparente del resultado sin aumentar su veracidad.

Conviene por tanto distinguir grados de reutilización, ordenados de menor a mayor efecto sobre el valor reportado. En la **reutilización nula** la medición se realiza sobre datos nunca antes consultados, y conserva íntegro su valor estadístico. En la **reutilización no adaptativa** el mismo conjunto se emplea más de una vez, pero ningún resultado intermedio condiciona

las decisiones siguientes; la degradación existe pero es acotada. En la **reutilización adaptativa** los resultados obtenidos sobre el conjunto guían decisiones posteriores (seleccionar un punto de control, fijar un hiperparámetro, elegir entre arquitecturas), y el valor final queda sesgado de forma optimista en una magnitud que no puede estimarse desde el propio conjunto. Existe, por último, un cuarto grado que la analogía con la falla de causa común hace visible: la **reutilización de la fuente**, en la que dos conjuntos formalmente separados proceden del mismo procedimiento generador. Aunque ningún registro concreto se repita entre ellos, ambos comparten el sesgo del generador, exactamente como dos módulos redundantes comparten la fuente de alimentación que puede hacerlos fallar a la vez.

La Tabla V aplica esta clasificación a los subconjuntos empleados en la presente investigación. El caso más relevante es el del conjunto de validación: fue consultado una vez por época durante veinte épocas y su resultado determinó la selección del punto de control de la época 14, lo que constituye reutilización adaptativa en sentido estricto. En consecuencia, la pérdida de validación de ese punto de control es, por construcción, el mínimo de veinte valores observados, y por tanto un estimador optimista del desempeño sobre datos no vistos.

Precisamente por eso el diseño experimental reserva una *tercera* partición. Las 165 consultas sobre las que el Capítulo IV reporta el desempeño son distintas de las 165 que gobernaron la selección del mejor punto de control: no participan en ninguna decisión y se consultan una sola vez, al final, que es la condición bajo la cual la prescripción de esta sección — declarar el número de consultas por conjunto y reservar una partición que no intervenga en el diseño — queda efectivamente satisfecha. La Sección 4.1.1 cuantifica, comparando ambos diseños sobre las mismas consultas, el sesgo que esta separación elimina.

Tabla V
Grado de reutilización de cada subconjunto y su efecto sobre el valor reportado

Subconjunto	Consultas	Grado	Efecto sobre el valor reportado
Entrenamiento (9 684 pares)	20 épocas	Por diseño	Reutilización prevista por el algoritmo; no afecta a la validez del valor de prueba
Validación (1 262 pares)	20	Adaptativa	La pérdida del mejor punto de control es el mínimo de 20 observaciones: estimador optimista
Consultas de prueba (165)	1	Nula	Partición disjunta de la de validación; no interviene en ninguna decisión de diseño y se consulta una sola vez al final
Corpus (833 productos)	Todas las medidas	No adaptativa	Las comparaciones comparten catálogo: sus errores están correlacionados, no son independientes
Generador sintético (fuente común)	Todos los conjuntos	De la fuente	Falla de causa común: el sesgo del generador afecta simultáneamente a entrenamiento y evaluación

Fuente: Elaboración propia, aplicando el razonamiento sobre fallas correlacionadas de la IEEE Reliability Society (IEEE Reliability Society, 2016; Malaiya, 1982) al ciclo de vida del conjunto de datos.

Nota: El conjunto de evaluación del Capítulo IV es una tercera partición, disjunta de la que gobernó la selección del mejor punto de control, de modo que las métricas reportadas no heredan el sesgo de selección. Los acoplamientos que subsisten — catálogo compartido entre particiones y generador sintético común — se declaran en la Sección 4.6.2.

De este análisis se desprende la vía por la que la reutilización sí enriquece los valores en lugar de erosionarlos, que es la que prescribe la práctica de confiabilidad: declarar el número de consultas realizadas sobre cada conjunto, mantener la trazabilidad de qué decisión

se tomó a partir de qué medición, y reservar una partición que no participe en ninguna decisión de diseño y se consulte una sola vez al final. Bajo esas condiciones, cada reutilización documentada agrega evidencia acumulada (tal como IEEE 1633 prevé para las sucesivas versiones de un producto (IEEE Reliability Society, 2016)) en lugar de erosionar silenciosamente las garantías estadísticas del conjunto.

A partir de este marco, la Tabla VI inventaría las fuentes de error del conjunto empleado en esta investigación, clasificadas según su naturaleza (aleatoria o sistemática) y según la vía por la que podría evaluarse su incertidumbre. El inventario es deliberadamente cualitativo: no se atribuye un valor numérico a componentes que no fueron medidos, ya que hacerlo produciría una falsa apariencia de rigor.

Tabla VI
Fuentes de error identificadas en el conjunto de datos y su efecto esperado

Fuente	Naturaleza	Evaluación	Efecto esperado
Sesgo del generador: el vocabulario coloquial procede de una lista controlada y no de registros reales de consulta	Sistemático	Tipo B	La distribución de formulaciones es una hipótesis del diseñador; el desempeño puede sobrestimarse frente al habla real
Cobertura regional declarada (Ecuador, Colombia, Perú) sin ponderación por frecuencia real	Sistemático	Tipo B	Desempeño desigual entre variedades regionales, no observable en el agregado
Positivo único por ancla: se asume un solo producto correcto	Sistemático	Tipo B	Recall@K y MRR se subestiman cuando varios productos serían igualmente válidos
Ausencia de anotación humana y, por tanto, de acuerdo entre anotadores	Sistemático	No evaluada	Incertidumbre de etiqueta desconocida; impide acotar el error del criterio de relevancia
Variabilidad de muestreo sobre el conjunto de consultas de prueba	Aleatorio	Tipo A	Diferencias menores que la dispersión muestral no son distinguibles del azar
Condiciones de medición de la latencia (una sola máquina, sin carga concurrente)	Sistemático	Tipo B	p50 y p95 son válidos condicionalmente a ese entorno y no extrapolables a producción
Reutilización adaptativa del conjunto de validación para seleccionar el punto de control	Sistemático	Tipo B	Sesgo optimista sobre la pérdida de validación; no se propaga al conjunto de prueba, que es una partición disjunta (Tabla V)
Catálogo compartido: 140 de las 165 consultas de prueba tienen un producto que también aparece en entrenamiento, 136 con las mismas palabras en otro orden	Sistemático	Tipo B	Las métricas estiman la generalización a consultas nuevas sobre un catálogo parcialmente conocido, no a productos no vistos
Artefactos del generador: casi-duplicados del catálogo e inconsistencia en la plantilla de cargadores (Sección 3.2.1)	Sistemático	Tipo B	Los casi-duplicados sesgan las métricas a la baja; las regularidades de plantilla pueden favorecer al modelo ajustado frente a BM25

Fuente: Elaboración propia, siguiendo el esquema de evaluación Tipo A/Tipo B de la GUM (Joint Committee for Guides in Metrology, 2008) y el razonamiento por modos de fallo de la IEC

La lectura conjunta de la tabla arroja una conclusión metodológica precisa: el componente dominante de la incertidumbre de este trabajo es sistemático y de Tipo B, no aleatorio. Esto significa que las pruebas estadísticas aplicadas en el Capítulo IV (que controlan exclusivamente la variabilidad de muestreo) acotan una sola de las fuentes de error inventariadas, y que un valor p pequeño no constituye evidencia contra el sesgo de construcción del conjunto. La respuesta apropiada a un error sistemático no es una prueba de significancia sino un cambio en el procedimiento de obtención del dato, motivo por el cual el Capítulo IV incluye una validación externa con registros reales de consulta y el Capítulo V identifica el *fine-tuning* con esos registros como la línea de trabajo futuro de mayor prioridad. Con estas delimitaciones establecidas, la siguiente sección expone las métricas con las que se cuantifica el desempeño y las pruebas con las que se acota su componente aleatorio.

Construcción y Validación de Colecciones de Prueba

Las secciones anteriores han tratado el dato como un insumo: algo que se tiene y cuya confiabilidad conviene declarar. Esta sección invierte el orden de la pregunta y examina el dato como un *artefacto construido*, con un procedimiento, unos supuestos y unos modos de fallo propios. La razón para hacerlo es directa: en recuperación de información, la colección de prueba no es el escenario donde se mide el sistema, sino el instrumento con el que se mide. Un instrumento mal caracterizado produce números precisos y equivocados, y la literatura del área lleva sesenta años documentando exactamente cómo ocurre eso.

El Paradigma de Cranfield y sus Supuestos

La forma en que hoy se evalúa casi cualquier sistema de recuperación procede de los experimentos que Cleverdon dirigió en el Cranfield College of Aeronautics entre 1957 y 1966 (Cleverdon, 1967). Su aporte no fue un algoritmo sino un *diseño experimental*: separar el sistema del juicio humano mediante tres componentes fijos — un conjunto de documentos, un conjunto de consultas y un conjunto de juicios de relevancia que indican qué documentos responden a qué consultas — de modo que dos sistemas puedan compararse sin volver a consultar a ningún usuario. Esa separación es lo que convirtió la evaluación de buscadores en una actividad acumulativa y reproducible, y es la que sostiene tanto las campañas TREC como los bancos de pruebas contemporáneos de recuperación densa que el Capítulo IV emplea como referencia (Thakur et al., 2021).

El paradigma, sin embargo, descansa sobre cuatro supuestos que rara vez se enuncian y que conviene explicitar porque determinan qué puede y qué no puede concluirse de una

medición. Sanderson los sistematiza en su revisión del área (Sanderson, 2010).

El primero es que la relevancia es binaria y objetiva: un documento responde o no responde a una consulta, y esa condición no depende de quién pregunte ni de por qué. El segundo es que la relevancia de un documento es independiente de los demás: que un documento sea relevante no cambia porque otro igualmente relevante ya se haya mostrado, lo cual desatiende la redundancia y la novedad. El tercero es que los **juicios son completos**: se supone conocida la relevancia de todo documento de la colección frente a toda consulta. El cuarto es que las **consultas representan necesidades de información reales**, y no formulaciones inventadas por quien construye la colección.

Ninguno de los cuatro se cumple estrictamente en la práctica, y cada uno ha generado su propia línea de investigación. Lo relevante para un trabajo aplicado no es cumplirlos — es imposible— sino saber cuáles se violan, en qué dirección sesgan el resultado y con qué magnitud. Las subsecciones siguientes recorren los tres primeros; el cuarto, por ser el que más directamente afecta a esta investigación, recibe tratamiento propio en la Sección 2.8.4.

Juicios de Relevancia: Quién Decide Qué es Relevante

El supuesto de objetividad de la relevancia fue el primero en ser sometido a prueba empírica. Voorhees aprovechó que en TREC-4 y TREC-6 algunos temas habían sido juzgados de forma independiente por evaluadores distintos y midió qué ocurría al intercambiar unos juicios por otros (Voorhees, 2000). El resultado tiene dos caras y ambas importan. Por un lado, el acuerdo entre evaluadores sobre documentos individuales resultó bajo: dos personas competentes discrepan con frecuencia sobre si un documento concreto responde a una consulta. Por otro, y de forma menos intuitiva, el *ordenamiento relativo de los sistemas* se mantuvo notablemente estable al cambiar el conjunto de juicios. La conclusión práctica es que una colección de prueba sirve para decidir cuál de dos sistemas es mejor mucho mejor de lo que sirve para estimar el valor absoluto de una métrica.

Ese hallazgo delimita con precisión lo que puede afirmarse en un trabajo como el presente. La comparación entre el modelo ajustado y el modelo base, o entre el modelo ajustado y BM25, es el tipo de afirmación que el paradigma sostiene con solidez. En cambio, la afirmación de que el sistema alcanza un MRR de 0.887 *en términos absolutos* es mucho más frágil: ese número depende del criterio de relevancia empleado, y con otro criterio razonable sería distinto.

La estabilidad que Voorhees documentó no es, además, incondicional. Bailey y sus

coautores mostraron que los evaluadores no son intercambiables cuando difieren en experticia sobre el dominio: los juicios de quienes formularon originalmente la necesidad de información, los de especialistas del área y los de evaluadores genéricos producen colecciones que no son equivalentes entre sí, y las diferencias se concentran precisamente en los documentos ambiguos (Bailey et al., 2008). Alonso y Mizzaro extendieron la pregunta al la colaboración abierta distribuida (*crowdsourcing*) y encontraron que un conjunto suficiente de evaluadores no expertos puede aproximar los juicios de los expertos de TREC, siempre que el diseño de la tarea controle la calidad de las respuestas (Alonso & Mizzaro, 2012). De ambos trabajos se sigue una misma prescripción metodológica: declarar quién juzgó, con qué criterio y con qué grado de acuerdo, porque son propiedades de la colección tanto como su tamaño. El marco estadístico para cuantificar ese acuerdo es el de los coeficientes de concordancia entre anotadores, cuya interpretación y límites Artstein y Poesio revisan en detalle para tareas de lingüística computacional (Artstein & Poesio, 2008).

En términos sencillos: preguntar “¿cuán bueno es este buscador?” es como preguntar “¿cuánto mide esta mesa?” sin decir con qué regla. La respuesta cambia con la regla. Lo que sí se conserva al cambiar de regla es cuál de dos mesas es más larga, y por eso las comparaciones son más confiables que los valores absolutos.

Complejidad de los Juicios y Sesgo de Agrupamiento

El supuesto de completitud es inviable a cualquier escala interesante: juzgar un millón de documentos contra doscientas consultas exigiría doscientos millones de decisiones humanas. La solución que TREC adoptó desde su inicio es el agrupamiento de resultados (*pooling*): se reúnen los primeros resultados devueltos por un conjunto de sistemas participantes, se juzga únicamente esa unión y todo documento no juzgado se supone irrelevante.

Zobel analizó las consecuencias de esa aproximación y encontró que, si bien el agrupamiento produce colecciones utilizables, introduce un sesgo sistemático contra los sistemas que no contribuyeron al conjunto juzgado: un buscador nuevo que recupere documentos relevantes que ningún participante original encontró será penalizado, porque esos documentos constan como irrelevantes (Zobel, 1998). Buckley y Voorhees llevaron el diagnóstico más lejos al mostrar que, cuando la fracción de documentos juzgados disminuye, las métricas tradicionales dejan de ordenar correctamente a los sistemas; propusieron por ello la medida *bpref*, definida solo sobre documentos efectivamente juzgados y por tanto mucho más robusta ante

juicios incompletos (Buckley & Voorhees, 2004).

Esta línea de trabajo tiene una consecuencia que conviene registrar para el diseño empleado en esta memoria. Cuando una colección asigna a cada consulta un único documento correcto, el problema del agrupamiento desaparece por construcción —no hay nada que agrupar— pero reaparece transformado en un supuesto más fuerte: el de que no existe ningún otro documento de la colección que responda igualmente bien a esa consulta. Si ese supuesto falla, las métricas *subestiman* el desempeño real, porque penalizan al sistema por devolver en primer lugar un documento que un evaluador humano habría aceptado. El sentido del sesgo es, en este caso, conservador: las cifras reportadas son un piso, no un techo.

Consultas Reales frente a Consultas Construidas

El cuarto supuesto de Cranfield —que las consultas representan necesidades reales— es el que separa con más nitidez a las colecciones modernas de las clásicas. El giro lo marcó MS MARCO, construida a partir de consultas anónimas efectivamente escritas por usuarios del buscador Bing, con pasajes juzgados por anotadores humanos (Nguyen et al., 2016). Su influencia excede el volumen: al proceder de registros de uso, la distribución de formulaciones, errores tipográficos, ambigüedades y expresiones truncadas es la del habla real y no la que un diseñador imagina.

En comercio electrónico, el equivalente es el *Shopping Queries Dataset* publicado por Amazon, que reúne consultas reales de compradores con juicios humanos en cuatro grados —*Exact*, *Substitute*, *Complement*, *Irrelevant*, de donde toma su nombre ESCI— y cobertura multilingüe en inglés, japonés y español (Reddy et al., 2022). Dos de sus rasgos son directamente pertinentes aquí. Primero, la relevancia no es binaria: un producto sustituto no es ni correcto ni irrelevante, y tratarlo como una u otra cosa distorsiona la medición. Segundo, una misma consulta admite varios productos aceptables, de modo que la suposición de positivo único queda descartada desde el diseño.

Construir una colección con consultas reales exige, no obstante, condiciones que un trabajo de titulación rara vez reúne: acceso a registros de un buscador en operación, tratamiento de datos personales y presupuesto de anotación. Cuando esas condiciones no se dan, la alternativa practicable es construir las consultas, y entonces la pregunta pertinente ya no es si la colección es real sino *en qué se aparta de una real y con qué efecto*. La respuesta honesta suele ser que una colección construida sobrestima el desempeño, porque el diseñador genera las consultas que el sistema puede resolver y no las que un usuario realmente escribiría. Este

es, precisamente, el punto donde el conjunto de datos de la presente investigación es más vulnerable, y la Sección 3.2.7 lo examina con detalle.

Generación Sintética de Pares Consulta–Documento

Entre la colección con consultas reales y la colección escrita a mano existe una tercera vía que ha crecido con rapidez: generar las consultas automáticamente a partir de los documentos. La idea aparece con *doc2query*, que entrena un modelo secuencia a secuencia para predecir, dado un pasaje, las consultas que ese pasaje respondería, y usa esas predicciones para expandir el documento antes de indexarlo (Nogueira et al., 2019). El giro conceptual es notable: en lugar de salvar la discrepancia de vocabulario desde el lado de la consulta, la salva desde el lado del documento, anticipando cómo lo buscará el usuario.

La llegada de los modelos de lenguaje de gran escala convirtió esa técnica en un método general de construcción de datos de entrenamiento. *InPars* genera pares consulta–documento mediante indicaciones de pocos ejemplos sobre un modelo de lenguaje, filtra los pares resultantes por su calidad estimada y entrena con ellos un reordenador, obteniendo mejoras sobre modelos supervisados con datos reales en varios dominios (Bonifacio et al., 2022). *Promptagator* lleva el planteamiento al extremo al mostrar que ocho ejemplos anotados bastan para generar, por dominio, un conjunto de entrenamiento competitivo, lo que sugiere que el cuello de botella de la recuperación densa especializada no es la cantidad de datos sino el costo de arrancar (Dai et al., 2023). Trabajos posteriores han refinado las etapas de filtrado y control de calidad de los pares producidos, que es donde se concentra el margen de mejora de estos esquemas (Krastev et al., 2025).

Esta literatura legitima metodológicamente el uso de datos sintéticos en recuperación, pero conviene leer con precisión qué legitima y qué no. Los tres trabajos citados generan consultas *a partir de documentos reales* usando modelos de lenguaje entrenados sobre texto humano; la variabilidad léxica de las consultas producidas procede, en última instancia, de la variabilidad del corpus de preentrenamiento. Un esquema que en cambio genere las consultas aplicando *reglas de sustitución sobre una lista cerrada de términos* hereda la cobertura de esa lista y nada más: puede producir muchos pares, pero no puede producir un fenómeno lingüístico que la lista no contemple. La distinción es la diferencia entre ampliar una muestra y replicarla.

El riesgo de fondo de entrenar sobre datos generados ha sido caracterizado en términos generales por Shumailov y sus coautores, quienes muestran que el reentrenamiento

recursivo sobre salidas de modelos previos degrada progresivamente la cobertura de las colas de la distribución hasta colapsarla (Shumailov et al., 2024). El mecanismo es directamente trasladable a la construcción de colecciones: si las consultas se generan desde un vocabulario controlado, los fenómenos raros —justamente los que un buscador debe resolver y un emparejamiento léxico no resuelve— quedan sub-representados o ausentes, y la evaluación no puede detectar esa carencia porque el conjunto de prueba comparte el mismo origen.

Tipos de Validez Aplicados a una Colección de Prueba

Las observaciones anteriores pueden ordenarse con la tipología de validez que Shadish, Cook y Campbell consolidaron para el diseño experimental en ciencias sociales (Shadish et al., 2002). Su traslado a la evaluación de sistemas de recuperación no es forzado: se trata en ambos casos de inferir, a partir de una medición sobre una muestra, una afirmación sobre una población.

La **validez de conclusión estadística** pregunta si la relación observada entre variables es real o atribuible al azar. Es la que atienden las pruebas de significancia y los intervalos de confianza del Capítulo IV, y sobre la que Fuhr y Sakai han discutido cuáles son las prácticas correctas en el área (Fuhr, 2017; Sakai, 2020).

La **validez interna** pregunta si la diferencia observada se debe al factor que se cree haber manipulado y no a otro. En un experimento de recuperación, las amenazas típicas son la fuga de información entre particiones y la reutilización adaptativa del conjunto de retención, tratadas en la Sección 2.7.

La **validez de constructo** pregunta si lo que se mide es lo que se dice medir. Aquí se ubica la pregunta más incómoda para cualquier colección: ¿mide el MRR la utilidad para el usuario, o mide la coincidencia con un criterio de relevancia que alguien fijó? Cuando la relevancia es binaria y de positivo único, el constructo medido es estrecho, y conviene nombrarlo con precisión en lugar de dejarlo implícito.

La **validez externa** pregunta hasta dónde se generaliza el resultado. Es la dimensión que una colección construida compromete de forma más severa, porque la población de consultas sobre la que se mide no es una muestra de la población de consultas reales sino una construcción del diseñador.

Documentación del Conjunto como Requisito Metodológico

De todo lo anterior se sigue que las propiedades de una colección son parte del resultado, no un preámbulo administrativo. Esa es la tesis que Gebu y sus coautores formalizaron

con las fichas técnicas de conjuntos de datos: todo conjunto debería acompañarse de una ficha que declare su motivación, composición, proceso de recolección, usos previstos y limitaciones conocidas (Gebru et al., 2021). Bender y Friedman propusieron el instrumento equivalente para datos lingüísticos —las declaraciones de datos—, con énfasis en las variedades de lengua representadas y en las poblaciones de hablantes, precisamente la información que determina si un sistema funcionará para quien no habla la variedad dominante (Bender & Friedman, 2018).

La justificación de estas prácticas no es formal sino empírica. Sambasivan y sus coautores documentaron, entrevistando a equipos que despliegan sistemas de aprendizaje automático en producción, cómo los problemas de calidad del dato se propagan aguas abajo y se manifiestan tarde, cuando corregirlos es costoso; llamaron cascadas de datos a ese fenómeno (Sambasivan et al., 2021). Northcutt y sus coautores mostraron, por su parte, que incluso los conjuntos de prueba canónicos del área contienen errores de etiqueta en proporción suficiente para alterar el orden relativo de los modelos evaluados sobre ellos (Northcutt et al., 2021). Uno y otro hallazgo apuntan a la misma conclusión: la calidad del instrumento de medida acota la calidad de cualquier conclusión que se derive de él, y esa cota no se estrecha acumulando más mediciones sobre el mismo instrumento.

Sobre esta base conceptual, la Sección 3.2.7 del capítulo siguiente contrasta punto por punto el procedimiento de construcción del conjunto empleado en esta investigación con los estándares aquí revisados, declarando en cada caso qué se cumple, qué no y en qué dirección sesga el resultado lo que no se cumple.

Evaluación de Sistemas de Recuperación

La calidad de la recuperación se mide con métricas sensibles al orden, es decir, que no solo verifican si el sistema encontró un producto relevante, sino en qué posición de la lista de resultados lo colocó. Esta distinción es esencial en comercio electrónico: un comprador rara vez se desplaza más allá de los primeros resultados, de modo que un producto relevante situado en la posición veinte equivale, en la práctica, a un producto no recuperado.

- El **rango recíproco medio** (MRR, *Mean Reciprocal Rank*) promedia el recíproco de la posición del primer resultado relevante. *En términos sencillos*: mide si el producto correcto apareció en primer lugar de la lista de resultados. Un valor cercano a 1 significa que el sistema casi siempre acierta en la primera posición; un valor cercano a 0 significa que el usuario debe desplazarse muy abajo en los resultados (o que nunca aparece) para

encontrar lo que buscaba.

- La **ganancia acumulada descontada normalizada** (NDCG@K, *Normalized Discounted Cumulative Gain*) admite relevancia graduada y premia colocar los elementos relevantes cerca de la cima de la lista. *En términos sencillos*: mide la calidad del orden de los primeros K resultados, otorgando más peso a que lo más relevante aparezca arriba y penalizando que quede enterrado entre resultados menos útiles.
- La **exhaustividad en K** (Recall@K) mide la fracción de elementos relevantes recuperados dentro de los primeros K resultados. *En términos sencillos*: responde a la pregunta “de todos los productos que realmente le servían al usuario, ¿qué porcentaje logró encontrar el buscador entre los primeros K resultados mostrados?”.
- La **latencia** se reporta mediante percentiles (p50, p95) para capturar los tiempos de respuesta típicos y de cola: p50 es el tiempo que tarda la mitad de las consultas más rápidas (el caso “típico”), mientras que p95 es el tiempo por debajo del cual responde el 95 % de las consultas, capturando así los casos más lentos que un promedio simple podría esconder.

Formalmente, para un conjunto de $|Q|$ consultas, el MRR se define como el promedio de los recíprocos de la posición rank_i del primer documento relevante,

$$\text{MRR} = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{\text{rank}_i}, \quad (2.5)$$

mientras que el NDCG@K normaliza la ganancia acumulada descontada frente a su valor ideal:

$$\text{DCG@K} = \sum_{i=1}^K \frac{2^{\text{rel}_i} - 1}{\log_2(i + 1)}, \quad \text{NDCG@K} = \frac{\text{DCG@K}}{\text{IDCG@K}}, \quad (2.6)$$

donde rel_i es el grado de relevancia del documento en la posición i e IDCG@K es el DCG del ordenamiento perfecto. El descuento logarítmico es lo que formaliza la intuición de que las posiciones altas valen más, y la normalización es lo que hace comparables consultas con distinto número de documentos relevantes. Estas son las definiciones generales; en este trabajo la relevancia es binaria y hay un único producto relevante por consulta, de modo que NDCG@K y Recall@K adoptan la forma simplificada que se precisa en la Sección 3.5.2, y así deben leerse todas las cifras del Capítulo IV.

Estas métricas pertenecen a la evaluación fuera de línea, que reproduce el comportamiento del sistema sobre un conjunto de consultas y juicios de relevancia fijos. Su virtud es

la reproducibilidad (cualquiera puede repetir el experimento sobre la misma colección), y su límite, que mide relevancia y no el objetivo comercial último. La evaluación en línea, mediante experimentos controlados sobre tráfico real, captura ese objetivo pero exige una plataforma en operación, un volumen de usuarios suficiente y semanas de recolección, condiciones que quedan fuera del alcance de este trabajo. La literatura documenta que la correlación entre ambos tipos de evaluación es positiva pero imperfecta, de modo que las ganancias reportadas aquí deben leerse como evidencia de mejora en la calidad de recuperación y no como una predicción directa de incremento en conversiones. En cuanto a las colecciones de referencia, BEIR (Thakur et al., 2021) estandarizó la evaluación de recuperación en cero disparos sobre dominios heterogéneos, y MTEB junto con su extensión multilingüe MMTEB (Enevoldsen et al., 2025; Muennighoff et al., 2023) hicieron lo propio para los modelos de *embeddings*; ninguna de ellas cubre, sin embargo, el registro coloquial hispanoamericano que motiva esta investigación, lo que justifica la construcción de un conjunto de evaluación propio.

Para determinar si las diferencias entre sistemas son significativas y no producto del azar, se aplica la prueba no paramétrica de rangos con signo de Wilcoxon (Wilcoxon, 1945) sobre mediciones pareadas, complementada con la d de Cohen para muestras pareadas (d_z , Sección 3.5.3) para cuantificar el tamaño del efecto práctico. La elección de una prueba no paramétrica responde a dos características de los datos: las métricas de ordenamiento están acotadas en $[0, 1]$ y distan de ser normales, y las mediciones son pareadas por consulta, ya que ambos sistemas se evalúan sobre exactamente el mismo conjunto de consultas, lo que permite controlar la variabilidad debida a la dificultad intrínseca de cada una.

Es preciso, no obstante, reconocer que el uso de estas herramientas está sujeto a debate metodológico dentro de la propia comunidad de recuperación de información. Fuhr (Fuhr, 2017) sistematizó una lista de errores frecuentes en la evaluación experimental (entre ellos, el uso del MRR por tratarse de una métrica sobre una escala ordinal cuyo promedio aritmético es cuestionable, la omisión de pruebas de significancia, y la comparación contra sistemas de referencia débiles) que afectan a buena parte de la literatura publicada. Sakai (Sakai, 2020) respondió matizando varias de esas objeciones y defendiendo el uso informado de métricas basadas en el recíproco del rango. El presente trabajo reporta el MRR por su amplia adopción y su interpretabilidad para un público no especializado, pero lo hace acompañado de NDCG@10 y Recall@K, de la prueba de Wilcoxon y del tamaño del efecto, siguiendo la recomendación común a ambos autores de no sustentar una conclusión en una sola métrica ni en una dife-

rencia de medias sin respaldo estadístico. Esta combinación de métricas de ordenamiento, percentiles de latencia y validación estadística otorga rigor metodológico tanto al estado del arte como al análisis experimental del presente trabajo.

Reducción de Dimensionalidad para Visualización: Análisis de Componentes

Principales

Los *embeddings* que produce MiniLM-L6-v2 son vectores de 384 dimensiones: números que, por sí solos, no pueden dibujarse ni inspeccionarse visualmente. El **Análisis de Componentes Principales** (*Principal Component Analysis*, PCA) es un método matemático de reducción de dimensionalidad que encuentra las combinaciones lineales de esas 384 dimensiones que explican la mayor parte de la varianza de los datos, y proyecta cada vector sobre las dos o tres direcciones (componentes principales) que retienen la mayor cantidad de información posible. *En términos sencillos*: PCA permite “aplanar” un espacio de 384 dimensiones a solo 2 o 3, de modo que los *embeddings* de los productos puedan graficarse en un plano o en un cubo y observarse visualmente si productos semánticamente similares, por ejemplo, de la misma categoría, quedan agrupados entre sí (evidencia cualitativa de que el modelo capturó relaciones de significado), o si se dispersan de forma aparentemente aleatoria. Esta técnica se emplea en el Capítulo IV para visualizar el espacio de *embeddings* del catálogo evaluado.

Desde el punto de vista formal, PCA obtiene las direcciones principales como los vectores propios de la matriz de covarianza de los datos centrados, ordenados de mayor a menor valor propio; equivalentemente, y de forma numéricamente más estable, mediante la descomposición en valores singulares de la matriz de datos. Cada valor propio indica la varianza capturada por su dirección, de modo que la *proporción de varianza explicada* por las dos primeras componentes cuantifica cuánta información conserva la proyección bidimensional. Jolliffe y Cadima (Jolliffe & Cadima, 2016) ofrecen una revisión actualizada del método y de sus extensiones, y subrayan una advertencia pertinente para esta aplicación: cuando la varianza está repartida entre muchas direcciones, como ocurre característicamente en los espacios de *embeddings* de alta dimensión, las dos primeras componentes explican una fracción pequeña del total y la proyección debe interpretarse con cautela.

Esa advertencia delimita el alcance de la evidencia obtenida por esta vía. Que dos categorías aparezcan solapadas en el plano no demuestra que el modelo no las distinga, ya que pueden estar separadas en direcciones que la proyección descarta; a la inversa, una separación visible sí constituye evidencia positiva, porque sobrevive a la pérdida de información de la

proyección. Existen alternativas no lineales de uso extendido para este propósito (t-SNE (van der Maaten & Hinton, 2008) y UMAP (McInnes et al., 2018)) que suelen producir agrupamientos visualmente más nítidos al preservar la estructura local de vecindad. Su contrapartida es que no conservan las distancias globales ni, en general, las densidades relativas, que dependen de hiperparámetros cuya variación altera de forma apreciable la figura resultante y que introducen una componente estocástica en el resultado. Por esas razones este trabajo emplea PCA: es determinista, no tiene hiperparámetros que ajustar, y reporta de forma explícita (mediante la varianza explicada) cuánta información se perdió al proyectar, lo que permite al lector calibrar por sí mismo la fuerza de la evidencia presentada.

Capítulo III

Desarrollo

Este capítulo describe la solución implementada: la arquitectura del sistema bajo el patrón de Arquitectura Limpia, la construcción del conjunto de datos sintético, el proceso de entrenamiento y *fine-tuning* del modelo de *embeddings*, y la implementación de la búsqueda semántica con su comparación frente a la arquitectura de dos pasos.

Arquitectura del Sistema

El sistema propuesto sigue un patrón de **Arquitectura Limpia** (*Clean Architecture*) implementado en Python, organizado en cuatro capas independientes: Dominio, Aplicación, Infraestructura y Presentación. Esta separación garantiza la mantenibilidad, la capacidad de prueba y la adaptabilidad ante cambios tecnológicos, cualidades importantes para un sistema experimental que además pretende ser reproducible. Este patrón de capas desacopladas coincide, además, con las recomendaciones actuales para sistemas de aprendizaje automático llevados a producción, donde separar el modelo, la lógica de negocio y la capa de exposición facilita el mantenimiento y la escalabilidad independiente de cada componente (Karanam & Bhargav, 2026).

En su núcleo, la solución combina un modelo de *embeddings* ajustado con una base de datos vectorial para realizar la recuperación semántica, que es en esencia el componente de recuperación de una arquitectura RAG en su forma más elemental (Lewis et al., 2020). Esta decisión de diseño no se plantea de forma aislada: como se discutió en el Capítulo II, el mismo patrón ha sido adoptado de forma independiente en sistemas de búsqueda de producción a gran escala como Walmart (J. Lin et al., 2024; Magnani et al., 2022), Facebook Marketplace (He et al., 2023) y Etsy (Jha et al., 2023), cada uno con su propia pila tecnológica. La contribución de este trabajo no reside, por tanto, en proponer una arquitectura novedosa, sino en instanciar y validar empíricamente dicho patrón, ya establecido en la industria, con una combinación específica de tecnologías ligeras (MiniLM-L6-v2 y Qdrant) orientada a mercados de comercio electrónico latinoamericanos con recursos de cómputo limitados.

Componentes Principales

- **Capa de Dominio:** define las entidades de negocio (Product, Query, TrainingPair) y las interfaces de repositorio, sin dependencias externas.

- **Capa de Aplicación:** implementa los casos de uso, incluyendo el entrenamiento del modelo, la evaluación, la carga de productos y la comparación de búsquedas (BM25 vs. semántica).
- **Capa de Infraestructura:** provee las implementaciones concretas: `SentenceTransformerAdapter` para la generación de *embeddings*, `QdrantAdapter` para el almacenamiento vectorial, `BM25Adapter` para la búsqueda léxica y `SQLiteAdapter` para la persistencia de métricas.
- **Capa de Presentación:** expone los puntos de acceso (*endpoints*) de la API REST mediante FastAPI, gestionando las solicitudes HTTP de búsqueda, evaluación y recuperación de métricas.

FastAPI, en términos sencillos. FastAPI es la plataforma que conecta el modelo de inteligencia artificial con una aplicación web de forma rápida: recibe una solicitud HTTP (por ejemplo, el texto de una búsqueda), la entrega al modelo de *embeddings* y a Qdrant para que hagan su trabajo, y devuelve la respuesta en un formato estándar (JSON) que cualquier sitio web o aplicación móvil puede consumir. Al estar escrito en Python, permite reutilizar directamente el mismo lenguaje que el modelo de aprendizaje automático, y su validación automática de datos reduce errores de integración entre el motor de búsqueda y las aplicaciones que lo consumen.

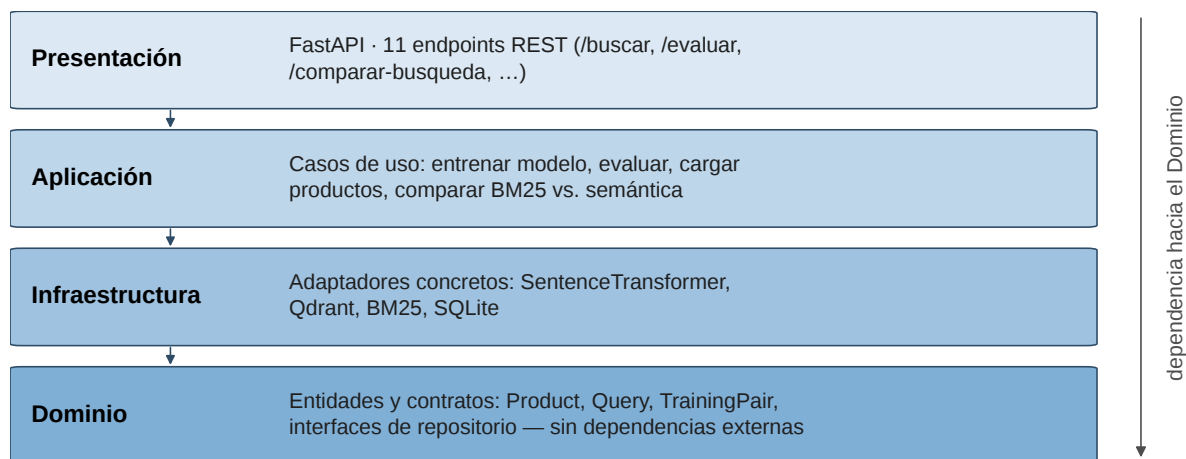


Fig. 1. Arquitectura Limpia de cuatro capas del sistema

Fuente: Elaboración propia.

Puntos de Acceso de la API

La Tabla VII resume los once puntos de acceso REST validados por la suite de pruebas del proyecto (`test_api.py`).

Tabla VII
Puntos de acceso REST expuestos por la capa de presentación FastAPI

Ruta	Método	Descripción
/cargar-datos	POST	Carga 500 productos en Qdrant
/buscar	POST	Búsqueda semántica (<i>embeddings</i>)
/buscar/lote	POST	Búsqueda semántica por lotes
/evaluar	POST	Evalúa modelo base vs. ajustado
/comparar/latencia	POST	Medición comparativa de latencia
/modelo/info	GET	Metadatos del modelo (dim, nombre)
/health	GET	Verificación de salud del servicio
/bm25/indexar	POST	Indexa documentos para BM25
/bm25/buscar	POST	Búsqueda léxica (BM25)
/comparar-busqueda	POST	Comparación BM25 vs. semántica
/metricas/resumen	GET	Resumen agregado de métricas

Fuente: Elaboración propia.

Pila Tecnológica

La Tabla VIII lista los componentes principales de la pila tecnológica.

Tabla VIII
Componentes de la pila del sistema

Componente	Tecnología
Arquitectura	Arquitectura Limpia (4 capas)
Marco de trabajo de la API	FastAPI + Uvicorn (Python 3.13)
Pruebas	Pytest 9.0.3 + FastAPI TestClient
Base de datos vectorial	Qdrant (con almacenamiento de carga útil)
Modelo de <i>embeddings</i>	MiniLM-L6-v2 ajustado (384 dim.)
Búsqueda léxica	rank-bm25
Almacenamiento de métricas	SQLite
Aprendizaje profundo	PyTorch / Sentence-Transformers

Fuente: Elaboración propia.

Construcción del Conjunto de Datos

El conjunto de datos empleado en este trabajo es sintético y está compuesto por 1,656 pares (ancla, positivo) almacenados en `Data/training_data.json`. Las anclas son consultas en registro coloquial y los positivos son descripciones de producto en registro técnico. La construcción de este conjunto es la decisión que más condiciona la calidad del *fine-tuning*, porque define exactamente qué tipo de correspondencia coloquial-técnica puede aprender el modelo. Cada par lleva asociadas dos anotaciones adicionales: la categoría fina del producto (`product_category`) y una calificación simulada (`star_rating`).

Conviene precisar el papel de cada una. La categoría fina se emplea en el análisis desagregado del Capítulo IV y en la proyección PCA. La calificación simulada, en cambio, **no interviene en ningún punto del experimento**: la función de pérdida MNRL solo consume el par (ancla, positivo), no pondera ni filtra ejemplos, y las métricas de recuperación tratan la relevancia como binaria. Se conserva en el archivo porque habilita una línea de trabajo futuro concreta — usarla como relevancia graduada para un NDCG que distinga entre un acierto sobre un producto bien valorado y uno sobre un producto mediocre — pero declararla como una variable sin uso actual evita atribuirle una influencia que no tiene. Cada par contiene:

- **Ancla (coloquial):** “estuche pal celu xiami”, “funda pal iphon”, “cauchos pal auto”.
- **Positivo (técnico):** “Carcasa protectora Xiaomi Redmi Note 12”, “Funda protectora iPhone 13”, “Neumáticos para vehículo”.

La Tabla IX resume la composición exacta del conjunto, medida directamente sobre el archivo de datos.

Tabla IX
Composición del conjunto de datos base

Propiedad	Valor
Pares (ancla, positivo)	1,656
Anclas únicas	1,534
Productos únicos (textos <i>positive</i>)	833
Categorías finas de producto	43
Longitud de la ancla (palabras): media / mediana / rango	3.65 / 4 / 3–7
Longitud del positivo (palabras): media / mediana / rango	13.93 / 14 / 7–22
Distribución de <i>star_rating</i> (3 / 4 / 5)	574 / 552 / 530

Fuente: Elaboración propia, a partir de *Data/training_data.json*.

Nota: Las anclas únicas (1,534) son menos que los pares (1,656) porque una misma formulación coloquial puede asociarse a más de un producto. Los 833 productos únicos son la población sobre la que se realiza la proyección PCA del Capítulo IV.

Generación de los Pares Base

Los 1,656 pares proceden de dos fuentes, ambas elaboradas por el autor. La primera, que aporta 1,600 pares, es un generador por plantillas (*Scripts/generate_training_data.py*) que combina un tipo de producto (una de las 43 categorías finas), una marca tomada de una lista cerrada por familia, un modelo o una especificación técnica y una de nueve frases promocionales genéricas (por ejemplo, “Envío rápido y garantía oficial.”). Cada producto así formado da lugar a *dos* anclas que contienen las mismas palabras en distinto orden: una en el orden tipo–marca–modelo (“webcam hp brio 4k”) y otra en el orden modelo–marca–tipo (“brio 4k hp webcam”). La segunda fuente son 56 pares redactados manualmente con formas coloquiales explícitas (“cargador pa’l celu”, “cauchos pal auto”, “frenos pal auto”), cada uno asociado a un producto propio. Las variantes coloquiales que introduce la aumentación (Sección 3.2.2) se añaden después, sobre la partición de entrenamiento, a partir del vocabulario descrito en la Sección 3.2.3.

Ningún par procede de un sitio de comercio electrónico, de un conjunto de datos público ni de registros de usuarios: las marcas, los modelos, las especificaciones, las frases promocionales y los 56 pares manuales están definidos como listas cerradas dentro del propio script. Su procedencia es, por tanto, verificable: ejecutar *Scripts/generate_training_data.py* con

su semilla fija (`random.seed(42)`) reproduce `Data/training_data.json` byte a byte (huella SHA-256 que comienza por `fe783d65623f453a`). La consecuencia es la que ya anticipa la Sección 2.8.4: el corpus de 833 productos es tan sintético como las consultas, y combinaciones como “Smartphone Lenovo Galaxy S25” o “Espejo retrovisor Pirelli” muestran que las plantillas cruzan marcas y modelos sin restricción de realismo comercial. Los únicos datos reales del trabajo, procedentes de Mercado Libre, no se emplean en el entrenamiento sino en la validación externa descrita en la Sección 3.5.4.

Cuatro propiedades del generador condicionan la lectura de los resultados del Capítulo IV y conviene declararlas desde aquí:

- **Casi-duplicados.** 157 de los 833 productos tienen al menos otro producto cuya descripción solo difiere en la frase promocional final (por ejemplo, “Radiador Dayco de aluminio. Enfriamiento eficiente.” seguido de “Certificación de calidad internacional.” o de “Envío rápido y garantía oficial.”). Cuando la consulta no contiene ningún término que los distinga, el criterio de positivo único convierte la elección entre ambos en un acierto o un error fortuito.
- **Términos de relleno.** 1,122 de las 1,656 anclas incluyen un término como *pro*, *ultra*, *max* o *elite*, que en 868 de ellas no aparece en la descripción del producto y, por tanto, no aporta información para identificarlo.
- **Inconsistencia en la plantilla de cargadores.** La potencia que menciona la consulta y la que figura en la descripción se sortean por separado, de modo que 42 de los 46 pares de cargador generados por plantilla asocian una consulta con un producto de otra potencia (por ejemplo, “cargador lg inalámbrico 15w” con “Cargador LG USB-C 65W GaN”). La etiqueta de esos pares es, en rigor, discutible.
- **Especificación repetida.** En las categorías sin especificación propia, la plantilla inserta el modelo también como especificación, de modo que 38 descripciones lo repiten (“Laptop JBL M4 Pro 18GB con procesador M4 Pro 18GB”). La repetición aumenta la puntuación léxica de esas descripciones frente a las de otros productos del mismo modelo.

Estas propiedades se cuantifican en `evaluacion/analisis_complementario.json` y su efecto sobre las métricas se examina en las Secciones 4.2.3 y 4.3.2.

Partición y Aumentación

La partición se realiza *antes* de la aumentación, de forma deliberada, y separa tres subconjuntos y no dos. El procedimiento baraja los 1,656 pares base con una semilla fija

(`random.Random(42)`) y toma bloques consecutivos de $\min(200, n/10) = 165$ pares: el primero se reserva como conjunto de prueba, el segundo como conjunto de validación y los 1,326 restantes forman el conjunto de entrenamiento. Solo después se aplica la aumentación a cada bloque por separado.

La razón de separar tres y no dos subconjuntos es la que desarrolla la Sección 2.7: el conjunto que gobierna la selección del mejor punto de control se consulta una vez por época y su resultado decide qué punto de control se conserva, lo que constituye reutilización adaptativa y sesga de forma optimista cualquier métrica que se reporte sobre él (Dwork et al., 2015). Con la partición triple, la validación absorbe ese sesgo y el conjunto de prueba se consulta una sola vez, al final, sin haber influido en ninguna decisión de diseño.

El orden entre partición y aumentación es, a su vez, una salvaguarda contra la fuga de información. La aumentación sustituye palabras formales por variantes coloquiales tomadas de `Data/colloquial_words.json`, de modo que varias paráfrasis comparten el mismo texto *positive*. Si se aumentara primero y se particionara después, variantes coloquiales de un mismo par base podrían quedar a ambos lados de la división, y el modelo se evaluaría sobre contenido casi idéntico al que vio durante el entrenamiento. Aplicada tras la partición, la aumentación produce 9,684 pares de entrenamiento y 1,262 de validación.

Sobre el archivo `training_data_augmented.json`. Ese archivo contiene 12,040 pares y es el resultado de aumentar los 1,656 pares base *de una sola vez*, sin particionar. No es el conjunto de entrenamiento y no debe leerse como tal: sirve únicamente como muestra del volumen que la aumentación genera sobre el conjunto completo. Aumentar cada partición por separado produce $9,684 + 1,262 + 1,308 = 12,254$ pares, es decir 214 más, porque el deduplicador de variantes opera dentro de cada llamada y no elimina las colisiones que se producen entre particiones distintas.

Persisten, no obstante, dos acoplamientos que conviene declarar y que una partición aleatoria por pares no elimina. Primero, de las 165 anclas de prueba, 140 (84.8 %) tienen un producto asociado que también aparece como positivo de algún par de entrenamiento. Como el generador produce dos anclas por producto con las mismas palabras en distinto orden (Sección 3.2.1), en 136 de esos 140 casos la consulta de entrenamiento del mismo producto contiene exactamente las mismas palabras que la de prueba, reordenadas. El modelo no vio

la cadena de prueba, pero sí ese producto y, casi siempre, ese mismo conjunto de palabras; las métricas del Capítulo IV miden, por tanto, principalmente la generalización a consultas nuevas sobre un catálogo parcialmente conocido, y no la generalización a productos nuevos. Solo 25 consultas de prueba se refieren a productos ausentes del entrenamiento. Segundo, 26 de las 165 formulaciones de prueba aparecen también en entrenamiento emparejadas con otro producto, porque el conjunto base contiene 1,656 pares sobre solo 1,534 anclas únicas (Tabla IX). La Sección 4.2.3 desagrega los resultados según ambos acoplamientos.

Procedencia y Validación del Vocabulario Coloquial

Procedencia. El archivo `Data/colloquial_words.json` es una **selección propia del autor**. No se extrajo de registros de consultas, de un corpus ni de una obra lexicográfica: se elaboró durante el diseño del generador como hipótesis de trabajo sobre las formas en que un comprador podría nombrar los productos y atributos del catálogo, y en ese momento no se documentó un protocolo de recopilación. El archivo asocia 62 términos formales (claves) con 301 variantes (278 distintas), y la aumentación sustituye cada clave por cada una de sus variantes allí donde aparece (Sección 3.2.2). Como el procedimiento de recopilación no puede reconstruirse, su pertinencia lingüística y regional se evaluó *a posteriori*, por tres vías: la clasificación de cada variante según el fenómeno que representa, el contraste de las variantes léxicas con distribución geográfica con dos obras de referencia —el *Diccionario de la lengua española* (DLE) (**rae2014dle**) y el *Diccionario de americanismos* (DAmer) (**asale2010damer**), que registra el uso de cada acepción por país— y la medición de qué claves llegan efectivamente a los datos de entrenamiento.

Composición. La Tabla X clasifica los 301 pares clave–variante según la forma de la variante. Solo 31 (10.3 %) son variantes léxicas con distribución geográfica, es decir, regionalismos en sentido estricto; el resto corresponde a fenómenos que atraviesan las variedades del español (acortamientos, contracciones, variación ortográfica, anglicismos) o a sinónimos y paráfrasis de uso general.

Tabla X
Tipología de las variantes del vocabulario coloquial

Tipo de variante	Pares	Ejemplos (clave → variante)
Sinónimo o paráfrasis general	126	bluetooth → sin cables; resistente al agua → a prueba de agua
Anglicismo, sigla o marca	69	funda → <i>case</i> , <i>cover</i> ; memoria → <i>pendrive</i>
Acortamiento	31	celular → <i>celu</i> , <i>cel</i> ; teclado → <i>tecl</i>
Variante léxica geográfica	31	neumático → <i>llanta</i> , <i>caucho</i> ; automóvil → <i>carro</i>
Variación ortográfica	20	batería → <i>bateria</i> ; vehículo → <i>vehiculo</i>
Variante idéntica a la clave	11	garantía → <i>garantía</i>
Diminutivo	8	celular → <i>telefonito</i> ; funda → <i>fundita</i>
Contracción	5	para → <i>pa</i> , <i>pal</i> , <i>pa'l</i>
Total	301	

Fuente: Elaboración propia, a partir de `Data/colloquial_words.json`; clasificación en `evaluacion/analisis_complementario.py`.

Nota: Clasificación del autor por la forma de la variante. Cada par se asigna a una sola clase aplicando los criterios en este orden: idéntica a la clave, contracción, variación ortográfica, diminutivo, acortamiento, variante geográfica, anglicismo y, por exclusión, sinónimo. Las 11 variantes idénticas a su clave no producen ninguna sustitución.

Contraste con obras de referencia. La Tabla XI recoge, para las variantes léxicas con distribución geográfica y para una omisión relevante, el registro que ofrecen el DLE y el DAmér, si ese registro incluye a Ecuador, Colombia o Perú, y cuántas anclas de la partición de entrenamiento contienen la variante tras la aumentación.

Tres resultados se desprenden del contraste. Primero, de las variantes que llegan a las consultas de entrenamiento, *celu*, *cel*, *fono*, *audífonos*, *llanta* y *caucho* tienen uso registrado en al menos uno de los tres países; *cascos*, *goma*, *cubierta* y *pila* son de uso general; *móvil* es, según el DLE, de uso mayoritario en España, y *diademas* no está registrada con el sentido de auriculares en ninguna de las dos obras. Segundo, las formas propias de otras variedades (*chido* y *padre*, de México; *pura vida*, de Costa Rica) y las que carecen de registro regional pertenecen a claves de vocabulario valorativo o de opinión (*bueno*, *barato*, *funciona*) que nunca aparecen en las anclas, que son consultas de producto, y por tanto no intervienen en ningún resultado de este trabajo. Tercero, el contraste detecta un error de pertinencia —*caserito* como

Tabla XI
Contraste de las variantes léxicas geográficas con el DLE y el DAmer

Variante (clave)	Registro en DLE o DAmer	Ec/Co/Pe	Anclas
celu (celular)	DAmer: Ec, Pe, Bo, Ch, Ho, PR; uso juvenil	Ec, Pe	41
cel (celular)	DAmer: Pe, Mx, Ho, Bo; uso juvenil	Pe	41
fono (celular)	DAmer: Ec, Pe, Ch, Ur (auricular); Pe, Ch (aparato)	Ec, Pe	41
móvil (celular)	DLE: “usado más en España”	No	41
audífonos (auriculares)	DLE: Am.	Sí	38
casco (auriculares)	DLE: sin marca geográfica	General	38
diademas (auriculares)	Sin registro con este sentido en ninguna de las dos obras	No verif.	38
llanta (neumático)	DLE: Am.; DAmer: Co, Ec, Pe y otros 15 países	Sí	35
caucho (neumático)	DAmer: Co, Ve	Co	35
goma, cubierta (neumático)	DLE: sin marca geográfica	General	35
pila (batería)	DLE: sin marca geográfica	General	56
carro (automóvil)	DLE: Am.; DAmer: Co, Pe (automóvil); Ec, Co, Pe (vehículo)	Sí	0 ^a
auto, coche (automóvil)	DLE: sin marca geográfica	General	0 ^a
movilidad (automóvil)	DAmer: Pe, Bo, Py, Ar	Pe	0 ^a
jala (funciona)	DAmer: Mx, Ec	Ec	0 ^b
chido, padre, pura vida (bueno)	DAmer: Mx (<i>chido, padre</i>); CR y Mx (<i>pura vida</i>)	No	0 ^b
cojonudo, re güeno (bueno); de chollo (barato)	DLE: <i>cojonudo</i> y <i>chollo</i> sin marca geográfica; <i>re güeno</i> sin registro en el DAmer	No verif.	0 ^b
caserito (hogar)	DAmer: <i>casero</i> , “cliente o vendedor habitual” (Ec, Pe y otros); sin sentido de hogar	Sentido erróneo	0 ^b
parlante (ausente)	DLE: América Meridional, CR, ES, Ni; DAmer: Co, Ec, Pe y otros 11 países	Sí	—

Fuente: Elaboración propia, a partir del DLE (**rae2014dle**) y del DAmer (**asale2010damer**), consultados el 8 de octubre de 2026, y de `evaluacion/analisis_complementario.json`.

Nota: Abreviaturas de país del DAmer: Ar Argentina, Bo Bolivia, Ch Chile, Co Colombia, CR Costa Rica, Ec Ecuador, ES El Salvador, Ho Honduras, Mx México, Ni Nicaragua, Pe Perú, PR Puerto Rico, Py Paraguay, Ur Uruguay, Ve Venezuela. “Anclas”: anclas de la partición de entrenamiento que contienen la variante tras la aumentación. ^a La clave solo aparece en descripciones de producto, de modo que la variante se introduce en los positivos (entre 18 y 27 veces) y no en las consultas. ^b La clave nunca aparece en los pares base, por lo que la variante no interviene en el entrenamiento ni en la evaluación.

variante de *hogar*, cuando en Ecuador y Perú *casero* designa al cliente o vendedor habitual— y una omisión: *parlante*, nombre del altavoz con uso registrado en los tres países, no figura en el vocabulario, lo que explica uno de los fallos analizados en la Sección 4.3.2.

Relación con las categorías comerciales. La cobertura del vocabulario es desigual entre las tres familias del catálogo (Sección 3.2.6). Las claves se concentran en la electrónica de consumo (celular, auriculares, cargador, batería, teclado, mouse, memoria, funda, disco, monitor, laptop, tablet) y, en autopartes, en el neumático y el vehículo; ninguna nombra un producto de la familia de accesorios personales. Medido sobre la partición de entrenamiento, de las 62 claves solo 17 se activan en alguna consulta, 25 lo hacen únicamente en las descripciones de producto y 20 no se activan nunca. En consecuencia, las variantes coloquiales llegan a las consultas de 21 de las 43 categorías finas; las 22 restantes —entre ellas todas las de accesorios personales y la mayoría de las autopartes, como bujía, radiador o embrague— no reciben ninguna. A ello se suma un defecto de la regla de sustitución, que opera por subcadena y sin respetar los límites de palabra: 408 de las 8,358 variantes generadas (4.9 %) son espurias, como *pabrisas* a partir de *parabrisas* al sustituir la clave *para*. La sustitución de *neumático* por *llanta*, por último, coincide con el uso americano registrado, pero refuerza la homonimia con la categoría *llanta* (aro) del catálogo descrita en la Sección 3.2.6.

Alcance que se desprende. La cobertura de Ecuador, Colombia y Perú declarada en el Capítulo I debe entenderse en sentido acotado: el vocabulario contiene un núcleo de formas con uso registrado en esos países, mezclado con formas de uso general y con algunas de otras variedades, y no permite atribuir ningún resultado a una variedad nacional concreta. La validación de las variantes con hablantes de los tres países, o su obtención a partir de consultas reales, se registra como trabajo futuro en el Capítulo V.

Distribución de Fenómenos Coloquiales

Dado que la hipótesis del trabajo se refiere específicamente al lenguaje coloquial, es necesario cuantificar cuánto de ese lenguaje contiene efectivamente el conjunto. Clasificando cada ancla mediante las listas de `colloquial_words.json`, se obtiene la distribución de la Tabla XII.

Tabla XII
Presencia de fenómenos coloquiales en las anclas del conjunto base

Fenómeno	Anclas	% del total	De pares manuales
Anglicismo (<i>usb, mouse, laptop, case</i>)	152	9.2 %	6
Contracción (<i>pal, pa</i>)	56	3.4 %	56
Regionalismo (<i>llanta, cauchos, carro</i>)	51	3.1 %	17
Apócope (<i>celu</i>)	18	1.1 %	18
Variación ortográfica (<i>bateria, iphon</i>)	9	0.5 %	9
Al menos un fenómeno detectado	236	14.2 %	56
Sin fenómeno detectado	1,420	85.8 %	0

Fuente: Elaboración propia; clasificación por reglas léxicas derivadas de
 Data/colloquial_words.json.

Nota: Las categorías no son excluyentes: una misma ancla puede presentar varios fenómenos (por ejemplo “parlante pa'l celu” combina contracción y apócope), por lo que la suma de las cinco primeras filas excede el total de anclas con fenómeno. La última columna indica cuántas de esas anclas pertenecen a los 56 pares redactados manualmente (Sección 3.2.1); en las anclas de plantilla, el término detectado es el propio nombre de la categoría o de la especificación (*usb*, 62; *mouse*, 46; *laptop*, 38; *llanta*, 34).

Esta distribución tiene una consecuencia metodológica directa que se retoma en el Capítulo IV: el 85.8 % de las consultas del conjunto no presenta ningún marcador coloquial detectable, de modo que cualquier métrica agregada sobre el conjunto completo está dominada por consultas que no ejercitan la capacidad que el *fine-tuning* pretende mejorar. La clasificación por reglas, además, sobrestima el contenido coloquial: de las 236 anclas con algún fenómeno detectado, 180 son anclas de plantilla en las que el término marcado es el nombre de la categoría o de la especificación (*mouse, laptop, memoria usb, usb-c, llanta*), y solo los 56 pares redactados a mano (3.4 % del conjunto) contienen formas coloquiales escritas como tales. El análisis desagregado por fenómeno es, por tanto, la evidencia pertinente para la hipótesis, y no el promedio global, pero debe leerse con esa salvedad.

Relación con el Catálogo de la API

Es necesario distinguir dos poblaciones de productos que aparecen en este trabajo y que no deben confundirse. Los 833 productos únicos descritos arriba constituyen el corpus

documental del conjunto de datos y son la población sobre la que se entrena, se evalúa la recuperación y se calcula la proyección PCA del Capítulo IV. Por separado, el caso de uso `LoadProductsUseCase` genera un catálogo de demostración de 500 productos por combinación aleatoria de marca, tipo y atributo de calidad, destinado a ejercitar los puntos de acceso de la API. Ambas poblaciones son disjuntas: los 500 productos generados no proceden del conjunto de entrenamiento y, al no fijarse una semilla en su generación, difieren entre ejecuciones. Las cifras de recuperación reportadas en el Capítulo IV corresponden al corpus de 833 productos; el catálogo de 500 se emplea únicamente para la validación funcional de la API.

Este enfoque de generación sintética de pares consulta-documento sigue una línea de trabajo reciente en recuperación de información, donde la generación asistida de datos de entrenamiento, cuando no se dispone de suficientes consultas reales etiquetadas por usuarios, ha demostrado mejorar de forma consistente el desempeño de modelos de recuperación afinados en dominios específicos (Krastev et al., 2025).

Taxonomía del Catálogo

El catálogo se organiza en tres familias de productos, cada una compuesta por varias categorías finas. La Tabla XIII resume esta taxonomía con ejemplos representativos de categorías reales del conjunto de datos.

Tabla XIII
Taxonomía de categorías del catálogo utilizado para el entrenamiento y la evaluación

Familia	Categorías finas representativas
Electrónica y tecnología	cargador, celular, funda, tablet, laptop, auriculares, memoria USB, teclado, mouse, monitor, altavoz, micrófono, webcam, router, hub USB, disco duro, cable, batería portátil
Autopartes	aceite de motor, alternador, amortiguador, batería de auto, bomba de agua, bujía, correa de distribución, embrague, espejo, faros, filtro de aceite, filtro de aire, llanta, neumático, parabrisas, pastilla de freno, radiador, silenciador
Accesorios personales	cartera, cinturón, gorra, lentes, mochila, pulsera, reloj

Fuente: Elaboración propia, a partir de las 43 categorías finas de `Data/training_data.json`.

Dos de esas categorías finas merecen una advertencia, porque su relación con el lenguaje coloquial no es la que el Capítulo II describe para el resto. El catálogo contiene *llanta* (36 pares, correspondientes a *llantas de aleación*, es decir aros) y *neumático* (46 pares, el caucho)

como categorías *separadas*. En el habla de Ecuador, México y buena parte de la región, sin embargo, “llanta” se usa habitualmente como sinónimo de *neumático*. La consecuencia es una fuente de ruido de etiqueta que conviene declarar: para una consulta coloquial con la palabra “llanta”, el único positivo que el conjunto reconoce como correcto puede ser el aro cuando el usuario real quería el caucho, o al revés, y la métrica penalizará al sistema por una respuesta que un evaluador humano aceptaría. El mismo razonamiento vale, en menor medida, para *caucho*. Este efecto no se corrigió — hacerlo exigiría reetiquetar el conjunto con juicios de relevancia múltiples — y se suma al inventario de fuentes de error de la Tabla VI.

Esta segmentación en tres familias amplias – electrónica y tecnología, autopartes, y accesorios personales – refleja dominios de vocabulario técnico marcadamente distintos entre sí (por ejemplo, “bujía” y “correa de distribución” no comparten prácticamente ningún término con “memoria USB” o “auriculares”), lo cual pone a prueba la capacidad del modelo ajustado para generalizar el aprendizaje de la relación entre lenguaje coloquial y descripción técnica más allá de una sola familia de productos. Cada categoría fina aporta entre 18 y 58 pares ancla–positivo al conjunto de datos (38.5 de media), de modo que ninguna categoría domina el proceso de *fine-tuning*. La cobertura del vocabulario coloquial, en cambio, no es uniforme entre familias, como se detalla en la Sección 3.2.3.

Contraste del Protocolo con los Estándares de Construcción de Colecciones

La Sección 2.8 revisó los supuestos sobre los que descansa el paradigma de Cranfield y las prácticas que la literatura ha decantado para construir colecciones de prueba defendibles. Esta subsección aplica ese marco al conjunto efectivamente empleado en esta investigación. El propósito no es acreditar el procedimiento sino caracterizarlo: para cada estándar se declara si se cumple, y cuando no se cumple, en qué dirección sesga el resultado. Un sesgo declarado y con signo conocido es información utilizable; uno no declarado es un error sistemático.

La Tabla XIV resume el contraste.

Tabla XIV
Contraste del conjunto de datos con los estándares de construcción de colecciones de prueba

Estándar	Cumple	Dirección del sesgo
Consultas procedentes de necesidades de información reales (Sección 2.8.4)	No	Optimista. Las anclas se construyen por reglas sobre una lista cerrada; el sistema se evalúa sobre el lenguaje que el diseñador previó
Relevancia graduada en lugar de binaria (Reddy et al., 2022)	No	Conservador. Un producto sustituto cuenta como error, no como acierto parcial
Varios documentos relevantes por consulta	No	Conservador. Con 833 productos y 43 categorías, otros productos de la misma categoría serían aceptables y se cuentan como fallo
Juicios de relevancia emitidos por evaluadores humanos (Sección 2.8.2)	No	Desconocido. La etiqueta es el par generador; no hay criterio externo contra el cual contrastarla
Acuerdo entre anotadores declarado (Artstein & Poesio, 2008)	No	No evaluable. Un único criterio automático no admite medida de concordancia
Compleitud de los juicios / control del sesgo de agrupamiento (Sección 2.8.3)	No aplica	Con positivo único no hay conjunto que agrupar; el supuesto se traslada al punto anterior
Partición disjunta de entrenamiento, validación y prueba	Sí	—
Conjunto de prueba consultado una sola vez, sin intervenir en decisiones de diseño (Dwork et al., 2015)	Sí	—
Documentación de composición y procedencia (Gebu et al., 2021)	Parcial	Secciones 3.2.1 y 3.2.3 y Tablas IX a XIII cubren generación, vocabulario y composición; no hay ficha formal
Declaración de variedades de lengua representadas (Bender & Friedman, 2018)	Parcial	Desconocido. Se documenta el registro lexicográfico de las variantes (Sección 3.2.3), pero no su frecuencia real de uso en cada país

Fuente: Elaboración propia, aplicando los estándares revisados en la Sección 2.8 al conjunto descrito en esta sección.

Lectura del contraste. Tres lecturas se siguen de la tabla, y conviene ordenarlas por su peso sobre las conclusiones del trabajo.

Los incumplimientos no apuntan todos en la misma dirección. Es el punto que más se pierde cuando una limitación se enuncia en bloque. El origen construido de las consultas sesga el resultado hacia arriba: el conjunto contiene los fenómenos que la lista de `colloquial_words.json` contempla y ninguno más, de modo que el sistema se examina sobre un temario que conoce. En cambio, la relevancia binaria con positivo único sesga hacia abajo: cuando el modelo devuelve en primer lugar un producto distinto del anotado pero de la misma categoría fina, la métrica lo cuenta como fallo aunque un comprador lo hubiera aceptado. Con 833 productos repartidos en 43 categorías, esa situación no es marginal. Las dos fuerzas no se cancelan —no hay razón para suponer que tengan la misma magnitud— pero su coexistencia impide afirmar que las cifras reportadas sean simplemente generosas.

El caso de la homonimia interna mide el efecto del segundo sesgo. La Sección 3.2.6 documentó que *llanta* (36 pares, el aro de aleación) y *neumático* (46 pares, el caucho) conviven como categorías distintas mientras que en el habla de la región “llanta” nombra habitualmente el segundo. Para las consultas afectadas, el único positivo reconocido puede no ser el que el usuario quería, y el sistema es penalizado por una respuesta defendible. Es un ejemplo concreto, cuantificable y presente en el propio conjunto de lo que la literatura describe en abstracto como límite del supuesto de positivo único (Buckley & Voorhees, 2004).

La validez más comprometida es la externa. Siguiendo la tipología de la Sección 2.8.6: la validez interna del experimento es razonable en lo que respecta a la selección del modelo, porque la partición triple impide que el conjunto de prueba influya en ninguna decisión; queda, sin embargo, acotada por el solapamiento de productos y de palabras entre particiones (Sección 3.2.2), cuyo efecto se desagrega en la Sección 4.2.3. La validez de conclusión estadística está atendida mediante contrastes pareados e intervalos por remuestreo. Lo que el diseño no puede sostener es la validez externa. La población de consultas sobre la que se mide no es una muestra de las consultas que escriben los compradores latinoamericanos: es una construcción que las imita según la hipótesis del diseñador sobre cómo escriben. Por eso las conclusiones del Capítulo V se enuncian como válidas *a la escala y en el escenario evaluados*, y no como una estimación del desempeño en producción.

Qué exigiría cerrar la brecha. La literatura revisada ofrece dos caminos, de costo muy distinto, y conviene precisarlos porque son el contenido sustantivo de la primera recomendación

del Capítulo V.

El camino completo es el de MS MARCO y ESCI: partir de registros reales de consulta de una plataforma en operación y someterlos a anotación humana con relevancia graduada (Nguyen et al., 2016; Reddy et al., 2022). Resolvería de un golpe el primer, el segundo y el tercer incumplimiento de la tabla, y permitiría medir el acuerdo entre anotadores. Exige, a cambio, acceso a esos registros, tratamiento de datos personales y un presupuesto de anotación que excede el alcance de un trabajo de titulación.

El camino intermedio es el de la generación asistida por modelos de lenguaje sobre documentos reales (Bonifacio et al., 2022; Dai et al., 2023). No resuelve el problema de la procedencia —las consultas siguen siendo generadas— pero sí ataca el defecto específico que la Sección 2.8.5 identificó en la generación por reglas: la variabilidad léxica dejaría de estar acotada por una lista cerrada y pasaría a proceder del corpus de preentrenamiento del generador. Una etapa de filtrado por calidad de los pares producidos, como la que la literatura reciente incorpora (Krastev et al., 2025), y una anotación humana de verificación sobre una muestra permitirían además estimar por primera vez la tasa de error de etiqueta del conjunto.

Un tercer elemento, de costo marginal frente a los anteriores y aplicable de inmediato, es permitir varios positivos por consulta mediante expansión por categoría fina: aceptar como correcto cualquier producto de la misma categoría que el anotado. No requiere datos nuevos —la anotación `product_category` ya existe en el conjunto— y convertiría el sesgo conservador de la segunda y tercera filas de la tabla en una medición acotada, al permitir reportar las métricas bajo ambos criterios y observar la diferencia. Se registra como la extensión más inmediata del protocolo actual.

Entrenamiento y *Fine-tuning*

Se seleccionó MiniLM-L6-v2 (W. Wang et al., 2020) como modelo base: 22 millones de parámetros, *embeddings* de 384 dimensiones, aproximadamente 90 MB, adecuado para el despliegue en CPU.

Objetivo de Entrenamiento

El *fine-tuning* utilizó la **Multiple Negatives Ranking Loss (MNRL)** (Henderson et al., 2017), que trata todos los demás elementos del lote como negativos:

$$\mathcal{L} = -\log \frac{e^{\text{sim}(q, p^+)/\tau}}{\sum_{i=1}^N e^{\text{sim}(q, p_i^-)/\tau}} \quad (3.1)$$

donde q es el *embedding* de la consulta, p^+ es el producto positivo, p_i^- son los negativos del lote, $\tau = 0.05$ es la temperatura y sim es la similitud del coseno. Esta lógica de aprendizaje contrastivo acerca en el espacio vectorial las consultas coloquiales a sus productos correspondientes y aleja las combinaciones incorrectas.

MNRL, en términos sencillos. Durante el entrenamiento, el modelo observa una consulta coloquial (por ejemplo, “estuche pal celu xiami”) junto con su producto correcto y con todos los demás productos del mismo lote, que actúan como ejemplos incorrectos. La función de pérdida *premia* al modelo cuando acerca, en el espacio vectorial, la consulta a su producto correcto, y lo *castiga*, ajustando sus pesos internos, cuando la ubica más cerca de un producto incorrecto que del correcto. Repetido miles de veces sobre el conjunto de entrenamiento, este mecanismo de premio y castigo es lo que permite que el modelo aprenda a asociar el lenguaje informal con la descripción técnica del producto que el usuario realmente busca.

Configuración de Entrenamiento

El *fine-tuning* se ejecutó con la configuración de la Tabla XV. Todos los valores proceden del registro que deja el propio entrenador (`evaluacion/historial_entrenamiento_triple.json`), no de una declaración de diseño.

Tabla XV
Configuración e instrumentación del fine-tuning

Parámetro	Valor
Modelo base	all-MiniLM-L6-v2
Función de pérdida	<i>Multiple Negatives Ranking Loss</i>
Épocas	20
Tamaño de lote / acumulación de gradiente	32 / 2 (lote efectivo 64)
Tasa de aprendizaje (AdamW)	2×10^{-5}
Pasos de calentamiento (<i>warmup</i>)	302
Pasos por época / pasos totales	152 / 3,040
Longitud máxima de secuencia	64 <i>tokens</i>
Semilla	42
Precisión	FP32 (CPU)
Estrategia de evaluación y guardado	al final de cada época
Criterio del mejor punto de control	menor <code>eval_loss</code>
Pares de entrenamiento (tras aumentación)	9,684
Pares de validación (tras aumentación)	1,262
Época del mejor punto de control	14 (<code>eval_loss</code> = 0.2713)

Fuente: Elaboración propia, a partir de `evaluacion/historial_entrenamiento_triple.json`.

Nota: La longitud máxima de secuencia se redujo de 256 a 64 *tokens* porque el ejemplo más largo del conjunto tiene 22 palabras: ningún texto se trunca y el costo por paso en CPU disminuye sustancialmente.

Se configuró un límite de 20 épocas de entrenamiento para dar al modelo tiempo suficiente de aprender los modismos coloquiales del conjunto de datos sin restringir artificialmente el proceso de optimización. Sin embargo, entrenar durante las 20 épocas completas no garantiza el mejor modelo posible: pasado cierto punto, un modelo puede empezar a memorizar particularidades del conjunto de entrenamiento en lugar de aprender patrones generalizables, un fenómeno conocido como sobreajuste. Por ello se aplicó una **selección del mejor punto de control**: tras cada época se evaluó la pérdida sobre el conjunto de validación (datos que el modelo no usa para ajustar sus pesos), el entrenamiento completó las 20 épocas y al final se

conservó el punto de control correspondiente a la época con menor pérdida de validación. Es la variante retrospectiva de la parada temprana: cumple la misma función, pero no interrumpe el entrenamiento. Ese conjunto de validación es, por construcción de la Sección 3.2.2, distinto del de prueba: la selección no consulta en ningún momento las 165 consultas sobre las que el Capítulo IV reporta el desempeño.

La Figura 3, presentada en el Capítulo IV, muestra la evolución de ambas pérdidas a lo largo de las 20 épocas. El comportamiento observado es el característico de un *fine-tuning* sobre un conjunto pequeño: la pérdida de entrenamiento cae abruptamente de 0.3170 a 0.1010 entre la primera y la segunda época y después se estabiliza en torno a 0.055, mientras que la pérdida de validación desciende de 0.3821 a un mínimo de 0.2713 en la época 14 y a partir de ahí oscila sin mejorar, en el rango 0.2725–0.2736. La separación persistente entre ambas curvas (entrenamiento en 0.055 frente a validación en 0.27) indica que el modelo ajusta el conjunto de entrenamiento con holgura y que la capacidad adicional no se traduce en generalización; es precisamente la condición que la selección del mejor punto de control está diseñada para detectar. El punto de control de la época 14 fue, por tanto, el seleccionado como modelo final para las evaluaciones del Capítulo IV.

Conviene registrar una diferencia respecto del entorno de cómputo. El entrenamiento se ejecutó en CPU y requirió aproximadamente 1 h 15 min para los 3,040 pasos; no se empleó aceleración por GPU ni precisión mixta. Esta circunstancia refuerza el argumento de accesibilidad que motiva la elección de un codificador ligero: el *fine-tuning* completo del modelo es alcanzable sin *hardware* especializado.

Implementación de la Búsqueda

La recuperación semántica se implementó sobre Qdrant y se expuso mediante la API REST construida con FastAPI descrita en la Sección 3.1. El Algoritmo 1 describe la recuperación integrada de un solo paso utilizada por el punto de acceso `/buscar`.

Algoritmo 1 Búsqueda semántica integrada vía carga útil de Qdrant

Entrada: Consulta coloquial q (p. ej., “estuche pal celu xiami”), entero top_k

Salida: Lista ordenada de productos con metadatos y puntuación de similitud

```

1: procedure Buscar( $q, top\_k$ )
2:    $v \leftarrow$  GenerarEmbedding( $q$ )                                ▷ vector de 384 dim.
3:    $resultados \leftarrow$  BusquedaQdrant( $v, top\_k$ )
4:   return  $resultados$                                              ▷ título, precio, image_url, score
5: end procedure
6: function GenerarEmbedding( $q$ )
7:   return SentenceTransformer.encode( $q$ )
8: end function
9: function BusquedaQdrant( $v, top\_k$ )
10:  Consultar Qdrant con similitud del coseno
11:  Recuperar puntos con su carga útil (sin llamada secundaria a BD)
12:  return lista ordenada de productos
13: end function
  
```

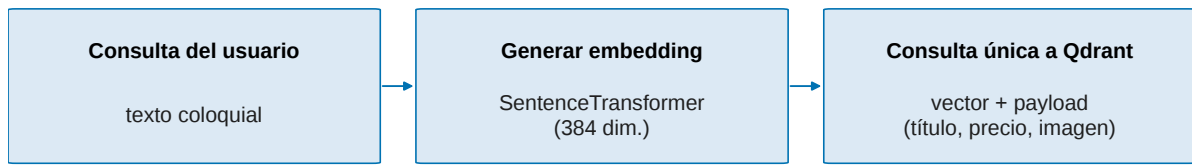
Arquitectura Integrada vs. de Dos Pasos

Para justificar el enfoque vectorial integrado se implementaron y compararon dos variantes:

Dos pasos (FAISS + SQL). (1) Generar el *embedding* de la consulta. (2) Buscar en FAISS los identificadores de producto. (3) Ejecutar una consulta SQL para obtener los metadatos (precio, título, URL de imagen).

Integrada (Qdrant con carga útil). (1) Generar el *embedding* de la consulta. (2) Una única consulta a Qdrant devuelve los *embeddings* y la carga útil completa del producto, sin un segundo viaje de ida y vuelta.

(a) Integrada — 1 viaje de ida y vuelta



(b) Dos pasos — 2 viajes de ida y vuelta

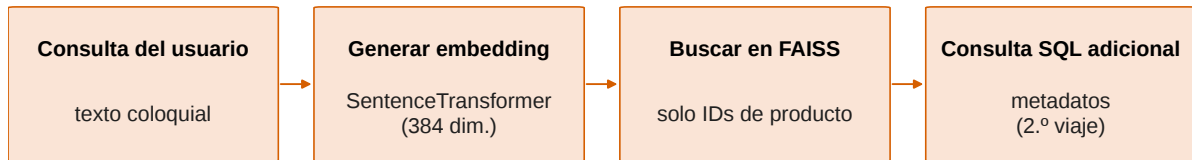


Fig. 2. Comparación de las arquitecturas de recuperación: integrada (Qdrant) vs. dos pasos (FAISS + SQL)

Fuente: Elaboración propia.

Protocolo de Evaluación

Sistemas Comparados y Nomenclatura

Para evitar ambigüedades, en los Capítulos III a V se emplean de forma fija tres denominaciones. **BM25** es la referencia léxica (BM25 Okapi con $k_1 = 1.5$ y $b = 0.75$). El **modelo base** es MiniLM-L6-v2 preentrenado y sin *fine-tuning*, que actúa como referencia semántica. El **modelo ajustado** es el mismo codificador tras el *fine-tuning* de la Sección 3.3. Cuando el texto habla de *búsqueda semántica* sin más precisión, se refiere al procedimiento común a los dos últimos (codificar la consulta y ordenar el corpus por similitud del coseno); las comparaciones de resultados nombran siempre el sistema concreto, porque el modelo base y el modelo ajustado no se comportan igual frente a BM25. Asimismo, *ancla* y *consulta* designan el mismo texto visto desde el entrenamiento y desde la evaluación, respectivamente, y *positivo* y *producto relevante*, la descripción de producto asociada a él.

Métricas

El criterio de relevancia de este trabajo es binario y de positivo único: para cada consulta, un solo producto del corpus cuenta como relevante (el positivo del par) y cualquier otro cuenta como no relevante (Sección 2.8.3). Cada sistema devuelve los 20 primeros productos del corpus, y rank_i denota la posición del producto relevante para la consulta i en esa lista. Bajo ese criterio, las definiciones generales de la Sección 2.9 —Ecuaciones 2.5 y 2.6— adoptan la forma siguiente:

- **MRR**: $\frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{\text{rank}_i}$, donde una consulta cuyo producto relevante no aparece entre los

20 primeros aporta 0.

- **NDCG@10**: con un único relevante, $IDCG@10 = 1$ y la métrica vale $1/\log_2(\text{rank}_i + 1)$ si $\text{rank}_i \leq 10$ y 0 en otro caso. No incorpora relevancia graduada; es un recíproco del rango con descuento logarítmico, más tolerante que el MRR con los aciertos en segunda o tercera posición.
- **Recall@K** ($K = 5$ y $K = 10$): con un único relevante, vale 1 si el producto aparece entre los K primeros y 0 si no, de modo que su promedio es la proporción de consultas resueltas dentro de los K primeros resultados (tasa de acierto en K).
- **Latencia p50, p95 y p99**: percentiles del tiempo por consulta, en milisegundos; p50 es la mediana.

Las comparaciones entre dos sistemas A y B se expresan de dos maneras que no deben confundirse. La **ganancia relativa**, $(B - A)/A$ en porcentaje, es la que reportan las tablas de resultados globales (por ejemplo, +15.8 % de MRR). La **diferencia absoluta**, $B - A$ en unidades de la métrica (por ejemplo, +0.195 de MRR), es la que se emplea en los desgloses por subgrupo, donde los valores de partida son muy distintos entre sí y un porcentaje resultaría engañoso.

Significancia Estadística

Se aplicó la prueba de rangos con signo de Wilcoxon (Wilcoxon, 1945) (no paramétrica, pareada). El diseño es pareado por construcción: cada consulta se evalúa con ambos sistemas, de modo que las observaciones vienen de dos en dos y no de dos grupos independientes. El tamaño del efecto correspondiente a ese diseño es la d de Cohen para muestras pareadas, denotada d_z (Cohen, 1988; Lakens, 2013), que divide la diferencia media entre la desviación estándar de las diferencias y no entre una desviación combinada de dos grupos:

$$d_z = \frac{|\bar{d}|}{s_d}, \quad \bar{d} = \frac{1}{n} \sum_{i=1}^n d_i, \quad s_d = \sqrt{\frac{1}{n-1} \sum_{i=1}^n (d_i - \bar{d})^2} \quad (3.2)$$

donde $d_i = m_i^{(B)} - m_i^{(A)}$ es la diferencia, en la consulta i , entre la métrica obtenida por el sistema B y la obtenida por el sistema A , y n es el número de consultas de prueba. Los umbrales de interpretación son los mismos que Cohen propone para d : $d_z < 0.2$ insignificante; $0.2 \leq d_z < 0.5$ pequeño; $0.5 \leq d_z < 0.8$ mediano; $d_z \geq 0.8$ grande.

Por qué d_z y no la d de muestras independientes. Usar la desviación combinada de dos grupos supondría que las dos muestras son independientes, lo que aquí es

falso: las mismas 165 consultas se resuelven con ambos sistemas sobre el mismo corpus, de modo que buena parte de la variabilidad entre consultas (unas son intrínsecamente más difíciles que otras) es común a ambas mediciones y se cancela al restar. Tomar la desviación de las diferencias aprovecha ese emparejamiento; tomar la combinada lo desperdicia y subestima el efecto. Es la misma razón por la que se aplica una prueba de Wilcoxon pareada y no su versión para muestras independientes. Todos los valores de d_z del Capítulo IV se calculan con la Ecuación 3.2 en `evaluacion/evaluacion_triple.py`, función `wilcoxon_y_efecto`.

Validación Externa con Datos Reales de Mercado Libre

Para contrastar los resultados obtenidos con datos sintéticos con el comportamiento de los sistemas ante consultas escritas por compradores reales, se realizó una validación externa con los datos del **MeLi Data Challenge 2020**, publicados por Mercado Libre para su concurso anual de ciencia de datos (**meli2020challenge**). Estos datos *no* intervienen en el entrenamiento: el modelo ajustado es el mismo de la Sección 3.3, entrenado únicamente con el conjunto sintético, y se evalúa sobre ellos sin ninguna adaptación adicional.

Fuente y acceso. Los archivos se obtuvieron de la copia de los datos originales del concurso publicada en Kaggle (**meli2020kaggle**) (<https://www.kaggle.com/datasets/marlesson/meli-data-challenge-2020>), que se descarga como un único archivo comprimido (`archive.zip`, 307 480 455 bytes, huella SHA-256 42d49392584dcf96...). Contiene el catálogo de productos (`item_data.jl`: identificador, título, dominio, categoría y precio de 2 102 277 publicaciones de Mercado Libre Brasil y México) y los historiales de navegación (`train_dataset.jl`: 413 163 sesiones, cada una con las búsquedas y las vistas de producto de un usuario durante la semana previa a una compra y el producto finalmente comprado). La huella completa del archivo y la de cada archivo interno quedan registradas en `evaluacion/validacion_meli2020.json`, de modo que cualquier lector puede verificar que trabaja con los mismos datos.

Construcción del conjunto de validación. Las decisiones se fijaron antes de calcular ninguna métrica y están codificadas en el script `evaluacion/validacion_meli2020.py`:

- **Sitio.** Solo Mercado Libre México (prefijo MLM), único sitio en español del concurso; el otro, Brasil, está en portugués.
- **Tipos de producto.** Se seleccionaron los dominios de la taxonomía de Mercado Libre equivalentes a las 43 categorías finas del catálogo de la tesis (por ejem-

plo, *celular* → MLM-CELLPHONES, *funda* → MLM-CELLPHONE_COVERS, *neumático* → MLM-AUTOMOTIVE_TIRES); la correspondencia completa figura en el script. El corpus de recuperación lo forman las 54 305 publicaciones de esos dominios, deduplicadas por título normalizado: **51 061 títulos reales**, redactados por los vendedores.

- **Consulta y producto relevante.** De cada sesión cuyo producto comprado pertenece a esos dominios se toma la última búsqueda realizada *antes* de la primera vista del producto comprado, es decir, la búsqueda que plausiblemente condujo a él. Las sesiones sin una búsqueda previa a esa vista se descartan. Tras eliminar pares repetidos, quedan **3 909 consultas reales**, cada una asociada al título del producto que el usuario terminó comprando.

Criterios de relevancia y métricas. La relevancia es implícita —una compra posterior, no un juicio humano— y se mide con dos criterios. El *estricto* considera relevante solo el título del producto comprado entre los 51 061 del corpus y emplea las métricas de la Sección 3.5.2 sobre los 100 primeros resultados (MRR, NDCG@10, Recall@10 y Recall@100). El criterio *por dominio* considera correcto cualquier resultado del mismo tipo de producto que el comprado (el mismo dominio de Mercado Libre) y reporta el acierto de dominio en la primera posición y la precisión de dominio en los diez primeros resultados. El primero mide si el sistema encuentra el producto exacto; el segundo, si entiende qué tipo de producto se busca, que es la capacidad que el análisis de casos de la Sección 4.3.2 identifica como la ventaja semántica más clara. Los intervalos y contrastes se calculan con las mismas funciones que el resto del Capítulo IV.

Trazabilidad de las Cifras Reportadas

Toda cifra experimental del Capítulo IV procede de uno de los siete artefactos que el proyecto deja escritos en disco, generados por los scripts indicados y versionados junto a la memoria:

- `evaluacion/historial_entrenamiento_triple.json` — configuración efectiva, partición y pérdida por época; lo escribe `evaluacion/reentrenar_triple.py`.
- `evaluacion/resultados_evaluacion_triple.json` — métricas de recuperación, intervalos *bootstrap*, contrastes y desgloses; lo escribe `evaluacion/evaluacion_triple.py`, que además deja las consultas empleadas en `evaluacion/test_queries_triple.json`.
- `evaluacion/benchmark_latencia.json` — descomposición de la latencia por componente sobre 1,000 consultas; lo escribe `evaluacion/benchmark_latencia.py`.
- `evaluacion/validacion_api.json` — tasa de éxito y latencia de 1,000 peticiones a la

API; lo escribe `evaluacion/validacion_api.py`.

- `evaluacion/analisis_complementario.json` — desgloses por solapamiento con el entrenamiento, intervalos por fenómeno, análisis de casos y de errores, y caracterización del vocabulario coloquial; lo escribe `evaluacion/analisis_complementario.py` a partir de los artefactos anteriores, sin reentrenar ni volver a codificar.
- `evaluacion/validacion_meli2020.json` — validación externa sobre consultas reales del MeLi Data Challenge 2020, con la huella SHA-256 del archivo de datos empleado; lo escribe `evaluacion/validacion_meli2020.py` (Sección 3.5.4).
- `evaluacion/validacion_meli2020_errores.json` — análisis de errores de esa validación (tipo de producto devuelto en la primera posición, consultas con marca y combinaciones de marca y modelo del conjunto sintético); lo escribe `evaluacion/validacion_meli2020_errores.py` con los mismos datos y modelos.

Cada archivo registra además la versión de Python y de las bibliotecas empleadas, la fecha de ejecución y las semillas utilizadas, de modo que cualquier cifra del capítulo siguiente puede rastrearse hasta la corrida que la produjo. Ninguna cifra reportada procede de una estimación de diseño o de una ejecución no registrada.

Capítulo IV

Análisis de Resultados

Este capítulo reporta y discute los resultados experimentales del sistema. Primero se describe la configuración experimental y la convergencia del *fine-tuning*; a continuación se evalúa el modelo ajustado frente al modelo base y frente a BM25, se analizan casos concretos de acierto y de error, se contrastan los tres sistemas sobre consultas reales de Mercado Libre y se compara la arquitectura integrada con la de dos pasos; finalmente se discuten los hallazgos en relación con el problema de investigación y la literatura revisada. Salvo la validación externa de la Sección 4.4, todos los resultados se obtuvieron con el conjunto de datos sintético descrito en el Capítulo III y en condiciones experimentales controladas; las denominaciones de los sistemas y la forma de las métricas son las fijadas en las Secciones 3.5.1 y 3.5.2.

Configuración Experimental

Los experimentos se realizaron bajo las siguientes condiciones:

- **Hardware:** CPU; el *fine-tuning* y la evaluación se ejecutaron sin aceleración por GPU.
- **Corpus de recuperación:** 833 descripciones de producto únicas del conjunto de datos (Sección 3.2). El catálogo de 500 productos generados se emplea solo para la validación funcional de la API.
- **Consultas de evaluación:** las 165 anclas de la partición de *prueba*, disjunta tanto de las 1,326 de entrenamiento como de las 165 de validación y fijada por la semilla 42 (Sección 3.2.2).
- **Protocolo:** recuperación de los 20 primeros resultados por coseno sobre el corpus completo; intervalos de confianza del 95 % por *bootstrap* con 10,000 remuestreos.
- **Artefactos:** `evaluacion/test_queries_triple.json`, `evaluacion/resultados_evaluacion_triple.json`, `evaluacion/evaluacion_triple.py` y `evaluacion/reentrenar_triple.py`.

Independencia del Conjunto de Prueba

Conviene declarar de entrada la propiedad de la que depende la validez de todo lo que sigue. El conjunto de 165 consultas sobre el que se reportan las métricas **no participó en ninguna decisión de diseño**: la selección del mejor punto de control la gobernó una partición de validación distinta, también de 165 pares, disjunta de la de prueba. Se consulta, por tanto, una sola vez y al final, que es la condición bajo la cual un valor de retención conserva sus

garantías estadísticas (Dwork et al., 2015).

Esta separación corrige una limitación del diseño anterior, en el que un único conjunto reservado gobernaba la selección del punto de control y reportaba el resultado final, situación que la Sección 2.7 clasifica como reutilización adaptativa y que sesga el valor reportado de forma optimista en una magnitud no estimable desde el propio conjunto. Disponer de las dos mediciones permite ahora acotar empíricamente ese sesgo: sobre las mismas 165 consultas, el modelo seleccionado con el conjunto de prueba alcanzaba un MRR de 0.889 y un NDCG@10 de 0.916, frente a 0.887 y 0.915 del modelo seleccionado con una validación independiente. La diferencia — dos milésimas en MRR y una en NDCG@10 — es dos órdenes de magnitud menor que el ancho del intervalo de confianza, de modo que el sesgo de selección existía pero resultó despreciable en este montaje. El dato es informativo en sí mismo: confirma que la corrección era necesaria por rigor metodológico y no porque las conclusiones dependieran de ella.

La independencia es, sin embargo, una independencia de *decisiones*, no de contenido. Persisten dos acoplamientos, declarados en la Sección 3.2.2 y recogidos como limitaciones en la Sección 4.6.2: el catálogo es compartido entre particiones (140 de las 165 consultas de prueba se refieren a un producto que también aparece en entrenamiento, y en 136 de ellas una consulta de entrenamiento del mismo producto contiene las mismas palabras en otro orden) y 26 de las 165 formulaciones de prueba aparecen también en entrenamiento, emparejadas con otro producto. La Sección 4.2.3 desagrega los resultados según ambos acoplamientos.

Evaluación del Modelo Base y Ajustado

Convergencia del Fine-tuning

La pérdida de entrenamiento descendió de 0.3170 en la primera época a 0.1010 en la segunda y se estabilizó en torno a 0.055 durante el resto del proceso. La pérdida de validación, que es la que gobierna la selección del modelo, descendió de 0.3821 hasta un mínimo de **0.2713 en la época 14** y a partir de ahí osciló entre 0.2725 y 0.2736 sin volver a mejorar. La Figura 3 presenta ambas series completas, con un punto por época efectivamente registrado.

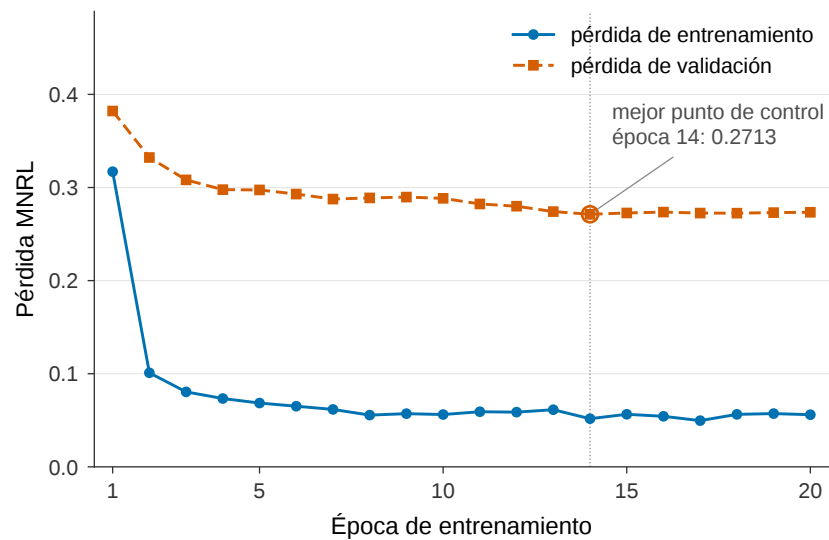


Fig. 3. Convergencia de las pérdidas de entrenamiento y validación (MNRL) durante las 20 épocas del *fine-tuning*

Fuente: Elaboración propia, a partir de `figuras/curva_perdida.csv`, derivado de `evaluacion/historial_entrenamiento_triple.json`.

La distancia persistente entre ambas curvas (0.055 en entrenamiento frente a 0.27 en validación) indica que el modelo ajusta con holgura el conjunto de entrenamiento sin que ello se traduzca en generalización, que es exactamente la condición que la selección del mejor punto de control detecta. El punto de control de la época 14 fue el seleccionado para todas las evaluaciones siguientes.

Calidad de Recuperación

La Tabla XVI compara el modelo base con el ajustado sobre las 165 consultas de prueba.

Tabla XVI
Rendimiento del modelo base vs. el modelo ajustado

Métrica	Modelo base	Ajustado	Ganancia
MRR	0.766 [0.712–0.816]	0.887 [0.852–0.920]	+15.8 %
NDCG@10	0.814 [0.770–0.855]	0.915 [0.887–0.940]	+12.4 %
Recall@5	0.903 [0.855–0.945]	0.988 [0.970–1.000]	+9.4 %
Recall@10	0.964 [0.933–0.988]	0.994 [0.982–1.000]	+3.1 %

Fuente: Elaboración propia, a partir de `evaluacion/resultados_evaluacion_triple.json`.

Nota: Entre corchetes, intervalo de confianza del 95 % por *bootstrap* (10,000 remuestreos). Ganancia: ganancia relativa, $(B - A)/A$. Prueba de Wilcoxon sobre el recíproco del rango por consulta:

$p = 1.3 \times 10^{-5}$; $d_z = 0.383$ (efecto pequeño); 61 de 165 consultas cambiaron de posición.

El modelo ajustado supera al modelo base en las cuatro métricas, con la mayor ganancia en MRR (+15.8 %) y la menor en Recall@10 (+3.1 %). Este patrón es informativo: el *fine-tuning* no amplía sustancialmente el conjunto de productos que el sistema logra recuperar (el modelo base ya alcanzaba 0.964 de Recall@10) sino que *reordena* los resultados para situar el producto correcto en posiciones más altas. La prueba de Wilcoxon confirma que la diferencia no es atribuible al azar ($p = 1.3 \times 10^{-5}$), y el tamaño del efecto pareado ($d_z = 0.383$) la sitúa en el rango pequeño, consistente con las ganancias de adaptación de dominio reportadas en la literatura (Gururangan et al., 2020; Thakur et al., 2021). Reportar ambas cifras es deliberado: un valor p pequeño indica detectabilidad, no magnitud (Fuhr, 2017). Ninguna de las dos controla, además, el sesgo de construcción del conjunto (Sección 2.7), cuyo efecto examina la sección siguiente.

Control por Solapamiento con el Entrenamiento

Como se anticipó en la Sección 4.1.1, la separación por pares no impide que una consulta de prueba comparta palabras o producto con el entrenamiento. Esta sección desagrega los resultados en dos niveles, del más estrecho al más amplio.

Formulaciones repetidas literalmente. 26 de las 165 consultas de prueba emplean una formulación que también aparece en el conjunto de entrenamiento asociada a otro producto. La Tabla XVII repite la comparación sobre las 139 consultas cuya cadena exacta el modelo nunca vio.

Tabla XVII

Rendimiento sobre el subconjunto de consultas cuya cadena no aparece en entrenamiento ($n = 139$)

Métrica	BM25	Modelo base	Ajustado
MRR	0.828	0.773	0.916
NDCG@10	0.867	0.820	0.936
Recall@5	0.950	0.899	0.986
Recall@10	0.986	0.971	0.993

Fuente: Elaboración propia, a partir de `evaluacion/resultados_evaluacion_triple.json`, clave `subconjunto_ancla_no_vista`.

Nota: Wilcoxon, ajustado vs. modelo base: $p = 4.7 \times 10^{-7}$, $d_z = 0.497$; ajustado vs. BM25:

$$p = 7.6 \times 10^{-4}, d_z = 0.323.$$

Las ganancias se mantienen y aumentan: el MRR del modelo ajustado pasa de 0.887 sobre el conjunto completo a 0.916 sobre este subconjunto. El control descarta, por tanto, que la mejora dependa de la repetición *literal* de consultas, pero no de su repetición con otro orden: 111 de esas 139 consultas contienen exactamente las mismas palabras que una consulta de entrenamiento asociada al mismo producto, porque el generador produce dos anclas por producto con las palabras reordenadas (Sección 3.2.1).

Productos y palabras vistos en entrenamiento. La Tabla XVIII agrupa las 165 consultas según lo que el modelo vio de ellas durante el entrenamiento.

Tabla XVIII
MRR según la relación de la consulta de prueba con el entrenamiento

Relación con el entrenamiento	n	BM25	Base	Ajustado	Ajust. – BM25
Mismas palabras que una consulta del mismo producto, en otro orden	136	0.834	0.806	0.909	+0.075 [0.030, 0.121]
Mismo producto, otra formulación	4	0.625	0.479	0.750	+0.125 [0.000, 0.375]
Producto ausente del entrenamiento	25	0.605	0.590	0.786	+0.181 [0.066, 0.298]
Total	165	0.795	0.766	0.887	+0.092 [0.051, 0.135]

Fuente: Elaboración propia, a partir de `evaluacion/analisis_complementario.json`, clave `solapamiento`.

Nota: MRR por sistema. La última columna es la diferencia absoluta de MRR con su intervalo de confianza del 95 % por *bootstrap* (10,000 remuestreos). Wilcoxon, ajustado vs. BM25: $p = 4.4 \times 10^{-3}$, $d_z = 0.277$ ($n = 136$); $p = 0.012$, $d_z = 0.583$ ($n = 25$). Ajustado vs. modelo base: $p = 4.6 \times 10^{-4}$ ($n = 136$); $p = 4.6 \times 10^{-3}$ ($n = 25$). El grupo de cuatro consultas no admite inferencia.

Tres lecturas se siguen de la tabla. Primera: el 82.4 % de las consultas de prueba (136 de 165) son reordenamientos de una consulta de entrenamiento del mismo producto. Sobre ellas el modelo ajustado supera a BM25, pero ha visto durante el entrenamiento ese mismo conjunto de palabras asociado a ese mismo producto, de modo que parte de su ventaja puede deberse a haber memorizado la asociación; BM25, que no aprende del entrenamiento, no puede beneficiarse de ello. Segunda: en las 25 consultas sobre productos ausentes del entrenamiento —siete de ellas, las consultas coloquiales redactadas a mano— los tres sistemas rinden peor, lo que indica que son consultas intrínsecamente más difíciles, y aun así la ventaja del modelo ajustado sobre BM25 es mayor (+0.181 de MRR, con un intervalo que excluye el cero) y también lo es frente al modelo base. Con 25 consultas, el dato es preliminar, pero es lo más parecido a una medición de generalización a productos no vistos que permite el conjunto, y apunta en la misma dirección que el resultado global. Tercera: las cifras de las Tablas XVI y XIX deben leerse, en consecuencia, como estimaciones de la generalización a consultas nuevas sobre un catálogo parcialmente conocido, y no como una medida del desempeño sobre productos o formulaciones nunca vistos.

Comparación con BM25

La Tabla XIX compara el modelo ajustado con BM25, evaluados sobre el mismo corpus y las mismas consultas.

Tabla XIX
BM25 vs. modelo ajustado

Métrica	BM25	Ajustado	Ganancia
MRR	0.795 [0.750–0.839]	0.887 [0.852–0.920]	+11.6 %
NDCG@10	0.843 [0.808–0.878]	0.915 [0.887–0.940]	+8.5 %
Recall@5	0.945 [0.909–0.976]	0.988 [0.970–1.000]	+4.5 %
Recall@10	0.988 [0.970–1.000]	0.994 [0.982–1.000]	+0.6 %

Fuente: Elaboración propia, a partir de `evaluacion/resultados_evaluacion_triple.json`.

Nota: BM25 Okapi con $k_1 = 1.5$ y $b = 0.75$, tokenización en minúsculas sin eliminación de palabras vacías ni lematización. Ganancia: ganancia relativa, $(B - A)/A$. Prueba de Wilcoxon: $p = 1.5 \times 10^{-4}$; $d_z = 0.332$ (efecto pequeño); 47 de 165 consultas cambiaron de posición. Una variante de BM25 con normalización de tildes produjo resultados idénticos ($p = 1$).

El modelo ajustado supera a BM25 en calidad de ordenamiento (+11.6 % en MRR, +8.5 % en NDCG@10) con significancia estadística ($p = 1.5 \times 10^{-4}$) y un tamaño del efecto pequeño ($d_z = 0.332$). La ventaja se concentra, de nuevo, en el ordenamiento y no en la cobertura: en Recall@10 ambos sistemas son prácticamente equivalentes (0.988 frente a 0.994), lo que indica que BM25 también encuentra el producto correcto entre los diez primeros, pero con menor frecuencia en la primera posición.

Dos precisiones acotan este resultado. La primera es que la ventaja procede del *fine-tuning* y no de la representación densa por sí misma: sin *fine-tuning*, el modelo base no supera a BM25 en este conjunto (MRR de 0.766 frente a 0.795; Tabla XVI). La segunda es que BM25 es una referencia exigente aquí, porque las anclas comparten vocabulario literal con las descripciones de producto, y que la mayor parte de las consultas de prueba reordena palabras ya vistas en entrenamiento (Sección 4.2.3). El margen obtenido indica que, en el escenario evaluado, el modelo ajustado ordena mejor que BM25; no basta para concluir que pueda reemplazarlo de forma general, ni es una medida del desempeño esperable frente a consultas de usuarios reales.

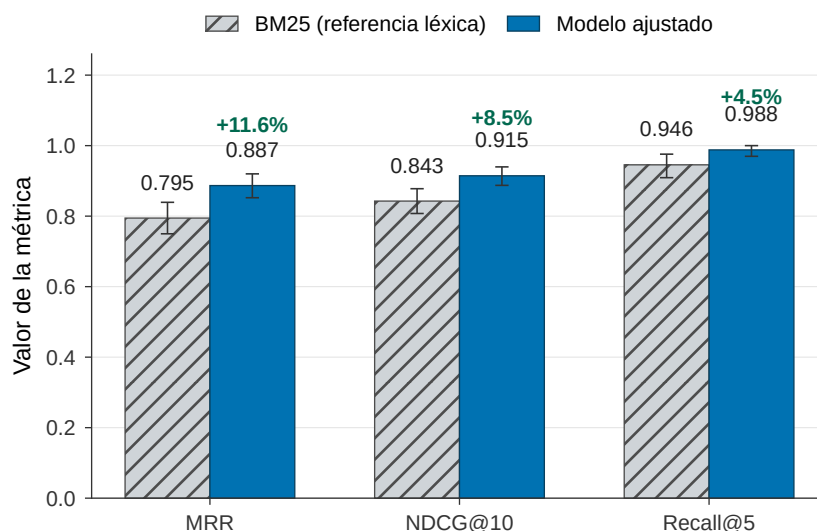


Fig. 4. Comparación de calidad de recuperación: BM25 vs. modelo ajustado

Fuente: Elaboración propia; las barras de error son el intervalo de confianza del 95 % obtenido por *bootstrap* con 10,000 remuestreos.

Rendimiento por Fenómeno Lingüístico

Dado que la hipótesis del trabajo se refiere específicamente al lenguaje coloquial, el promedio sobre el conjunto completo no es la evidencia pertinente: como se estableció en la Sección 3.2.4, el 85.8 % de las anclas no presenta ningún marcador coloquial detectable. La Tabla XX desagrega el MRR por tipo de fenómeno y añade, para cada subgrupo, el intervalo de confianza de la diferencia entre el modelo ajustado y BM25, que es lo que permite juzgar cuánta evidencia aporta cada fila.

Tabla XX
MRR por fenómeno lingüístico presente en la consulta

Fenómeno	n	BM25	Base	Ajustado	Ajust. – BM25
Anglicismo (<i>mouse, usb, laptop</i>)	15	0.671	0.737	0.867	+0.195 [0.056, 0.362]
Contracción (<i>pal</i>): pares manuales	7	0.481	0.524	0.663	+0.182 [−0.027, 0.435]
Apócope (<i>celu</i>)	3	0.222	0.278	0.381	+0.159 [−0.167, 0.500]
Regionalismo (<i>llanta, carro</i>)	3	1.000	0.381	1.000	0.000 [0.000, 0.000]
Sin fenómeno detectado	142	0.822	0.788	0.899	+0.078 [0.034, 0.124]

Fuente: Elaboración propia, a partir de `evaluacion/resultados_evaluacion_triple.json` y `evaluacion/analisis_complementario.json`, clave `por_fenomeno`.

Nota: Clasificación por reglas léxicas derivadas de `Data/colloquial_words.json`; las categorías no son excluyentes. Última columna: diferencia absoluta de MRR con su intervalo de confianza del 95 % por *bootstrap*. 14 de las 15 consultas con anglicismo y 2 de las 3 con regionalismo lo presentan solo porque el término forma parte del nombre de la categoría o de la especificación (*mouse, memoria usb, laptop, usb-c, llanta*); las siete consultas con contracción son exactamente las siete consultas de prueba procedentes de los pares redactados a mano. Wilcoxon, ajustado vs. BM25: $p = 0.027$ (anglicismo), $p = 0.375$ (contracción), $p = 2.5 \times 10^{-3}$ (sin fenómeno).

La lectura de la tabla exige separar la dirección de los resultados de su solidez. En *dirección*, el modelo ajustado obtiene el mejor MRR, o lo iguala, en las cinco categorías, y las diferencias más amplias frente a BM25 aparecen en los subgrupos con marcadores (+0.195, +0.182 y +0.159, frente a +0.078 en las consultas sin fenómeno). En *solidez*, la evidencia específica sobre jerga coloquial es preliminar, por dos motivos. Primero, el tamaño: los subgrupos de contracción, apócope y regionalismo tienen entre 3 y 7 consultas, sus intervalos incluyen el cero o son degenerados, y no admiten inferencia estadística. Segundo, la composición: el único subgrupo con un intervalo que excluye el cero, el de anglicismos, no está formado por jerga sino, salvo una consulta, por nombres de producto o especificaciones (“mouse acer mx master 3s”, “cargador belkin usb-c 65w”), y parte de su diferencia procede de consultas de cargador afectadas por la inconsistencia de la plantilla descrita en la Sección 3.2.1. Las consultas que sí contienen jerga escrita como tal son las siete de contracción, redactadas a mano; en ellas el modelo ajustado supera a BM25 en MRR (0.663 frente a 0.481), pero la diferencia no es estadísticamente distinguible del azar ($p = 0.375$). El resultado es compatible con la hipótesis

central, sin llegar a confirmarla, y la Sección 4.3.2 examina esas consultas una por una.

La fila de regionalismos merece comentario aparte, porque es la única que no aporta evidencia a favor y omitirla sería reporte selectivo. En esos tres casos BM25 y el modelo ajustado empatan en un MRR perfecto de 1.000: el emparejamiento léxico ya resolvía la consulta y el *fine-tuning* no tenía nada que añadir. Dos de las tres consultas, además, emplean *llanta* como nombre de la propia categoría del catálogo (el aro de aleación), no como regionalismo, y el término es homónimo en el catálogo (Sección 3.2.6); lo que la fila sí muestra es que el modelo base sin *fine-tuning* era el único de los tres sistemas que fallaba en ellas (0.381). Con $n = 3$ ninguna lectura es concluyente; se reporta por transparencia.

Ampliar el conjunto de evaluación con consultas coloquiales reales en número suficiente para la inferencia es, por todo ello, la primera recomendación metodológica del Capítulo V.

Análisis de Casos

Las métricas agregadas no muestran qué tipo de consulta resuelve cada sistema ni por qué falla. Comparando consulta a consulta la posición del producto relevante, el modelo ajustado lo coloca más arriba que BM25 en 38 de las 165 consultas, BM25 lo coloca más arriba en 9, y en las 118 restantes empatan (en 100 de ellas, ambos en primera posición). Ninguno de los dos sistemas lo sitúa en primer lugar en 27 consultas. La Tabla XXI recoge ejemplos representativos de cada situación, seleccionados para ilustrar los mecanismos y no por ser los más favorables.

Los ejemplos permiten precisar tres mecanismos. El primero es la ventaja semántica más clara del modelo ajustado: distinguir el *tipo* de producto que se busca cuando la consulta comparte palabras con descripciones de otro tipo (un freno frente a un neumático *para auto*, un teléfono frente a su funda), que es justamente la limitación del emparejamiento léxico descrita en el Capítulo II. No es, sin embargo, el más frecuente: de las 38 consultas en que el modelo ajustado supera a BM25, el primer resultado de BM25 es un producto de otro tipo en 6, otro producto del mismo tipo en 21 y un casi-duplicado del relevante en 11. En 10 de esas 21, la descripción que BM25 antepone repite la especificación pedida porque la plantilla la inserta dos veces (“Laptop JBL M4 Pro 18GB con procesador M4 Pro 18GB”), lo que eleva su puntuación por frecuencia de término; se trata, de nuevo, de una particularidad del generador. En “frenos pal auto” el modelo base ya acertaba, de modo que ese caso muestra una virtud de la representación densa más que del *fine-tuning*; en “cable pa’l celu”, en cambio, solo el modelo ajustado resuelve la forma coloquial.

Tabla XXI
Ejemplos de comportamiento del sistema sobre consultas de prueba

Consulta	BM25	Base	Ajust.	Primer resultado y lectura
<i>(a) El modelo ajustado supera claramente a BM25</i>				
"frenos pal auto"	5	1	1	BM25 antepone "Neumáticos Continental 205/55R16 para auto" por la coincidencia literal de <i>auto</i> ; los dos modelos semánticos recuperan "Pastillas de freno Brembo para auto". Producto ausente del entrenamiento.
"celular lenovo galaxy s25"	3	2	1	BM25 antepone una funda cuya descripción contiene <i>celular</i> y <i>Galaxy S25</i> ; el modelo ajustado distingue el teléfono ("Smartphone Lenovo Galaxy S25") del accesorio. La consulta reordena una de entrenamiento.
"cable pa'l celu"	2	2	1	BM25 antepone "Cable USB-C a Lightning 1m para iPhone"; el ajustado asocia <i>celu</i> con <i>celular</i> y recupera "Cable USB-C a USB-C 1m para celular". Producto ausente del entrenamiento.
<i>(b) BM25 supera al modelo ajustado</i>				
"funda pa'l celu"	6	>20	>20	El ajustado antepone "Cover para celular con diseño personalizado": otra funda, aceptable para un comprador, pero contada como error por el criterio de positivo único.
"lentes seiko ultra"	1	4	3	El ajustado antepone un casi-duplicado: el mismo producto con otra frase promocional final. Nada en la consulta distingue ambos textos; ocurre en 8 de los 9 casos de este grupo.
<i>(c) Ambos sistemas fallan</i>				
"memoria pal pc"	2	3	2	Ambos anteponen "Teclado mecánico para PC": <i>pc</i> solo aparece literalmente en esa descripción, y la clave <i>computadora</i> del vocabulario nunca se activó en el entrenamiento.
"parlante pa'l celu"	>20	3	7	<i>Parlante</i> no figura en el catálogo ni en el vocabulario (Tabla XI); BM25 no encuentra el producto y el ajustado se guía por <i>celu</i> y antepone una batería externa para celular. El <i>fine-tuning</i> empeora aquí al modelo base.
"radiador dayco elite"	2	4	2	Ambos anteponen un casi-duplicado; <i>elite</i> no aparece en la descripción del producto y la consulta no permite distinguir el producto anotado.
<i>(d) Resultado condicionado por el generador</i>				
"cargador razer usb-c 20w"	8	8	2	La etiqueta es "Cargador Razer 15W inalámbrico". BM25 antepone "Cargador USB-C 20W para celular", que coincide con la especificación pedida; el ajustado tiende a priorizar la marca, la regularidad que el generador mantiene constante.

Fuente: Elaboración propia, a partir de `evaluacion/analisis_complementario.json`, clave casos.

Nota: Columnas BM25, Base y Ajust.: posición del producto relevante en la lista de cada sistema; ">20" indica que no aparece entre los 20 primeros resultados.

El segundo mecanismo explica la mayor parte de los errores: el criterio de positivo único sobre un catálogo con casi-duplicados. En 27 de las 35 consultas en que el modelo ajustado no coloca el producto relevante en primer lugar, lo que antepone es un casi-duplicado de ese producto, y lo mismo ocurre en 24 de los 57 errores de BM25. Esos errores no reflejan una confusión semántica sino la imposibilidad de distinguir, a partir de la consulta, dos descripciones que solo difieren en la frase promocional; el sesgo es conservador para ambos sistemas (Sección 3.2.7). El caso “funda pa’l celu” es una variante del mismo problema: la respuesta del modelo ajustado sería aceptable para un comprador.

El tercer mecanismo obliga a matizar parte de los aciertos: el modelo aprende las regularidades del generador, incluidas las que no corresponden a una búsqueda real. En la plantilla de cargadores, donde la potencia de la consulta no coincide con la del producto etiquetado en 42 de 46 pares, el modelo ajustado tiende a priorizar la marca, y BM25 resulta penalizado por devolver el cargador que coincide con la especificación pedida. Excluir la categoría de cargadores (7 consultas) no altera la conclusión de la Sección 4.3: la ventaja del modelo ajustado sobre BM25 pasa a ser del 9.6 % en MRR (0.891 frente a 0.813; $p = 6.0 \times 10^{-4}$, $d_z = 0.306$, $n = 158$). Por último, los fallos compartidos en consultas genuinamente coloquiales (“memoria pal pc”, “parlante pa’l celu”) coinciden con términos que el vocabulario de aumentación no cubre, lo que es consistente con el límite que la Sección 2.8.5 atribuye a la generación por lista cerrada: el modelo no aprende lo que el generador no produce.

Validación Externa con Consultas Reales de Mercado Libre

Todos los resultados anteriores proceden del conjunto sintético. Esta sección responde a la pregunta que esos resultados no pueden contestar: cómo se comportan los tres sistemas cuando la consulta la escribe un comprador real y el catálogo lo redactan vendedores reales. Para ello se emplean los datos del MeLi Data Challenge 2020 (**meli2020challenge**; **meli2020kaggle**) con el protocolo de la Sección 3.5.4: 3 909 búsquedas reales de Mercado Libre México, cada una asociada al producto que el usuario compró después, sobre un corpus de 51 061 títulos de publicaciones reales de los mismos tipos de producto que el catálogo de la tesis. Ningún sistema se entrenó ni se ajustó con estos datos.

El cambio de escenario es profundo y conviene tenerlo presente al leer las cifras. Las consultas reales son cortas (2.8 palabras de media) y suelen escribirse en mayúsculas y sin tildes (“FUNDA RUDO NOTE 7 XIAOMI”, “AUDIFONOS XIAOMI”, “MEMORIAS USB MAYOREO”), mientras que los títulos de los vendedores son más largos (8.6 palabras) y acumulan

modelos compatibles y reclamos comerciales (“Funda Samsung Galaxy A10/ A20/ A30/ A50/ A70/ A80 Uso Rudo”). El corpus es además 61 veces mayor que el sintético, y el criterio estricto exige encontrar, entre 51 061 títulos, exactamente la publicación que el usuario compró, aunque existan decenas de publicaciones equivalentes de otros vendedores. Por eso los valores absolutos del criterio estricto no son comparables con los de las secciones anteriores; lo que interesa es el ordenamiento relativo de los tres sistemas y el criterio por dominio.

Tabla XXII
Resultados sobre 3 909 consultas reales del MeLi Data Challenge 2020 (Mercado Libre México)

Métrica	BM25	Modelo base	Ajustado
<i>Relevancia estricta: el producto comprado</i>			
MRR (100 primeros)	0.044 [0.039, 0.049]	0.033 [0.029, 0.037]	0.037 [0.033, 0.042]
NDCG@10	0.051 [0.045, 0.056]	0.036 [0.031, 0.040]	0.043 [0.038, 0.048]
Recall@10	0.096 [0.087, 0.106]	0.067 [0.060, 0.076]	0.085 [0.077, 0.094]
Recall@100	0.320 [0.305, 0.335]	0.276 [0.262, 0.290]	0.312 [0.298, 0.327]
<i>Relevancia por dominio: el mismo tipo de producto</i>			
Acierto de dominio en la 1. ^a posición	0.729 [0.715, 0.742]	0.740 [0.726, 0.753]	0.678 [0.664, 0.693]
Precisión de dominio en los 10 primeros	0.690 [0.678, 0.701]	0.713 [0.701, 0.725]	0.671 [0.659, 0.683]

Fuente: Elaboración propia, a partir de `evaluacion/validacion_meli2020.json`; datos del MeLi Data Challenge 2020 (**meli2020kaggle**).

Nota: En negrita, el mejor valor de cada fila. Entre corchetes, intervalo de confianza del 95 % por *bootstrap* (10,000 remuestreos). Diferencias medias por consulta, con su intervalo, en el recíproco del rango: ajustado – BM25 = -0.007 [-0.011 , -0.003], $d_z = 0.06$; ajustado – base = $+0.005$ [0.002 , 0.008], $d_z = 0.05$; base – BM25 = -0.011 [-0.015 , -0.008], $d_z = 0.09$. En la precisión de dominio en los 10 primeros: ajustado – BM25 = -0.019 [-0.026 , -0.012], $d_z = 0.08$; ajustado – base = -0.042 [-0.048 , -0.036], $d_z = 0.22$; base – BM25 = $+0.023$ [0.016 , 0.031], $d_z = 0.10$. Wilcoxon pareado: $p < 0.001$ en los seis contrastes.

Tres resultados destacan. El primero es que ningún sistema recupera con frecuencia la publicación exacta que se compró: aparece entre los 10 primeros resultados en menos del 10 % de las consultas y entre los 100 primeros en el 28–32 %. Esto refleja sobre todo la dureza del criterio estricto en un catálogo lleno de publicaciones casi idénticas: ante “DISCO DURO EXTERNO 2TB TOSHIBA”, los tres sistemas devuelven en primera posición un disco externo

Toshiba, pero solo BM25 coloca primero la publicación concreta que el usuario compró (el modelo base la sitúa en la posición 8 y el ajustado en la 13).

El segundo es que el orden de los sistemas se invierte respecto del conjunto sintético. BM25 obtiene el mejor MRR (0.044), seguido del modelo ajustado (0.037) y del modelo base (0.033). El *fine-tuning* mejora al modelo base en la recuperación del producto exacto, pero no alcanza a BM25. Muchas consultas reales se reducen a marca y modelo (“KZ ZST PRO”, “I12 TWS”, “HUAWEI Y6”, “XIAOMI MI 9T ROJO”), identificadores que el título del vendedor repite literalmente: es el caso en que la coincidencia léxica es más fuerte y la ventaja semántica, menos necesaria.

El tercero es que, en el criterio por dominio, que mide si el sistema entiende qué tipo de producto se pide, el mejor sistema es el modelo base (74.0 % de acierto en la primera posición), seguido de BM25 (72.9 %), y el modelo ajustado queda último (67.8 %). El *fine-tuning* reduce en 4.2 puntos la precisión de dominio en los 10 primeros respecto del modelo del que parte, y es la única diferencia del conjunto completo cuyo tamaño de efecto llega a pequeño ($d_z = 0.22$). Todas las diferencias son estadísticamente significativas por el tamaño de la muestra, pero sus efectos son insignificantes o pequeños: en la práctica, los tres sistemas rinden de forma parecida sobre estas consultas reales, y ninguno reproduce la ventaja que el modelo ajustado mostraba en el conjunto sintético.

Tabla XXIII

Validación externa por familia de producto y por presencia de marcadores coloquiales

Subgrupo	n	MRR (producto comprado)			Precisión de dominio@10		
		BM25	Base	Ajust.	BM25	Base	Ajust.
Electrónica y tecnología	2 884	0.043	0.032	0.033	0.689	0.719	0.654
Accesorios personales	959	0.044	0.035	0.048	0.684	0.687	0.710
Autopartes	66	0.096	0.035	0.051	0.803	0.818	0.839
Con marcador coloquial	747	0.054	0.040	0.038	0.855	0.873	0.850
Sin marcador coloquial	3 162	0.042	0.031	0.037	0.651	0.675	0.629

Fuente: Elaboración propia, a partir de `evaluacion/validacion_meli2020.json`.

Nota: En negrita, el mejor valor de cada bloque de la fila; los intervalos de confianza y los contrastes de cada subgrupo están en el archivo fuente. Marcadores coloquiales según la misma clasificación por reglas del Capítulo III; en estas consultas reales el más frecuente es *audífonos*, que la clasificación cuenta como regionalismo.

La Tabla XXIII localiza la pérdida de precisión de dominio del modelo ajustado en la electrónica, la familia más numerosa: 6.5 puntos por debajo del modelo base (IC 95 % de la diferencia $[-7.3, -5.8]$; $d_z = 0.32$). La única familia en la que el modelo ajustado supera a los

otros dos sistemas es la de accesorios personales (relojes, mochilas, lentes, carteras, gorras, cinturones y pulseras): mejora la precisión de dominio frente a BM25 (+2.5 puntos [1.4, 3.7]) y frente al modelo base (+2.3 [1.4, 3.1]), y el MRR frente al modelo base (+0.014 [0.007, 0.021]), aunque no frente a BM25 (+0.004 [−0.004, 0.012]). En autopartes, BM25 obtiene el MRR más alto (0.096), pero con solo 66 consultas los intervalos son amplios y el criterio por dominio no separa a los sistemas: todos los intervalos de esas diferencias incluyen el cero o lo rozan.

Las consultas con marcadores coloquiales tampoco favorecen al modelo ajustado. En ellas BM25 conserva la ventaja en MRR (el ajustado queda 0.015 por debajo [−0.025, −0.006]) y el modelo base la de dominio (el ajustado queda 2.3 puntos por debajo [−3.4, −1.3]). Sus valores absolutos son más altos que los de las consultas sin marcador porque el marcador más frecuente, *audífonos*, nombra de forma explícita el tipo de producto. Conviene subrayar qué detecta aquí la clasificación por reglas: sobre todo regionalismos de uso general (412 consultas) y anglicismos (269), con solo 25 apócope; no la jerga andina que motiva el trabajo.

El análisis de errores (`evaluacion/validacion_meli2020_errores.json`) identifica el mecanismo de la pérdida de precisión de dominio. De las 347 consultas en que el modelo base acierta el tipo de producto en la primera posición y el ajustado no, 247 son búsquedas de fundas para las que el modelo ajustado devuelve el propio teléfono. Ante “FUNDA SAMSUNG A50”, el modelo base antepone “Funda Compatible Samsung Galaxy A50” y el ajustado, “Samsung A50”; ante “RUDO HUAWEI P30 LITE”, “Funda Uso Rudo Huawei P30 Lite Nova 3...” frente a “Huawei P30 Lite”; ante “REDMI NOTE 7 FUNDA”, “Funda Xiaomi Redmi Note 7 Redmi 7” frente a “Redmi Note 7”. El modelo ajustado da más peso a la marca y al modelo que a la palabra que nombra el tipo de producto. En las 1 089 consultas que mencionan alguna de las 65 marcas del generador sintético, su primer resultado contiene esa marca en el 99.3 % de los casos, pero pertenece a otro tipo de producto en el 31.1 %, frente al 19.0 % del modelo base, y su acierto de dominio cae del 80.1 % al 68.6 %; en las consultas sin esas marcas la caída es menor (del 71.6 % al 67.6 %).

Es la misma tendencia a priorizar la marca que el análisis de casos encontró en el conjunto sintético (Sección 4.3.2), y el generador explica por qué el *fine-tuning* la aprende: de las 28 combinaciones de marca y modelo de sus fundas y las 26 de sus celulares, solo 2 coinciden, de modo que la marca y el modelo bastan casi siempre para separar una funda de un teléfono y la palabra *funda* apenas aporta información para el ordenamiento. En el catálogo real ocurre lo contrario: “Samsung A50” nombra el teléfono y decenas de fundas de distintos vendedores.

El modelo ajustado aprendió, por tanto, una regularidad de su conjunto de entrenamiento que no se cumple en los datos reales.

La relevancia implícita añade un ruido que afecta por igual a los tres sistemas: en 779 consultas (19.9 %) ninguno devuelve en la primera posición un producto del tipo comprado, y muchas de ellas no guardan relación con la compra (“HUMIDIFICADOR ULTRASONICO” antes de comprar una funda, “TWENTY ONE PILOTS” antes de comprar un cargador para laptop). Junto a esos fallos compartidos, algunos casos ilustran el comportamiento propio de cada sistema. Los modelos semánticos aciertan donde BM25 falla cuando la consulta es una referencia abreviada o reordenada: con “I12 TWS” el modelo ajustado sitúa el producto comprado en la posición 24, frente a la 52 del modelo base y la 55 de BM25, y con “XIAOMI BATERIA EXTERNA” el modelo base y el ajustado lo sitúan en las posiciones 4 y 6, mientras que BM25 lo relega a la 29 y antepone una batería externa genérica. BM25 acierta donde los modelos fallan cuando la consulta reproduce el título del vendedor: ante “AUDIFONOS BLUETOOTH MANOS LIBRES50 EASY TOUCH” devuelve en primera posición la publicación comprada, que ninguno de los dos modelos sitúa entre sus 100 primeros resultados.

La validación tiene límites propios que acotan lo que puede concluirse de ella. La relevancia es implícita: que un usuario comprara un producto después de una búsqueda no garantiza que esa búsqueda lo describiera (en algunas sesiones la consulta previa no guarda relación con el producto comprado), y tampoco excluye que otros productos del corpus fueran igualmente adecuados, lo que hace del criterio estricto una cota inferior. Las consultas proceden de México y no de Ecuador, Colombia o Perú, de modo que esta validación contrasta el comportamiento ante español coloquial real, pero no ante las variedades andinas que motivan el trabajo. Y la distribución por familias está dominada por la electrónica (el 73.8 % de las consultas), con solo 66 consultas de autopartes, de modo que los resultados de esa familia no admiten inferencia.

Comparación Arquitectónica

La decisión de almacenar la carga útil del producto junto a cada vector de Qdrant, en lugar de recuperar los metadatos con una consulta SQL posterior, se tomó para eliminar un viaje de ida y vuelta por búsqueda. La Tabla XXIV mide qué cuesta realmente ese viaje.

Tabla XXIV
Descomposición de la latencia por consulta: integrada (Qdrant) vs. dos pasos (FAISS+SQL)

Componente (mediana, ms)	Integrada	Dos pasos
1. Codificación de la consulta	13.42	13.42
2. Búsqueda vectorial	2.45	0.18
3. Consulta de metadatos	—	0.16
Extremo a extremo, p50	15.91	14.04
Extremo a extremo, p95	18.81	16.16
Extremo a extremo, p99	24.35	22.41

Fuente: Elaboración propia, a partir de `evaluacion/benchmark_latencia.json`; script reproducible en `evaluacion/benchmark_latencia.py`.

Nota: 1,000 consultas sobre el corpus de 833 productos, $top_k = 10$, tras descartar 50 consultas de calentamiento; CPU, sin carga concurrente. El guion en la fila 3 indica que la arquitectura integrada no realiza ese acceso: Qdrant devuelve la carga útil junto con el vector, que es precisamente la decisión de diseño evaluada. Las medianas por componente no suman la mediana del total, porque la mediana de una suma no es la suma de las medianas. Ambas arquitecturas se ejecutan en el mismo proceso:

Qdrant en modo embebido (la implementación de referencia del cliente, no el servicio en Rust) y SQLite en memoria, porque el entorno de medición no dispone de Docker; la comparación aísla el costo del segundo acceso a metadatos y no incluye latencia de red, que afectaría a ambas. El script se ejecutó tres veces consecutivas: la tabla reporta la primera y el archivo de resultados registra el rango de las tres, porque la latencia absoluta depende de la máquina mientras que la proporción entre componentes se mantiene estable. Wilcoxon pareado sobre el tiempo extremo a extremo:

$p = 9.9 \times 10^{-135}$, $d_z = 0.845$, diferencia media de 1.92 ms a favor del esquema de dos pasos.

El resultado no confirma la hipótesis de partida y conviene enunciarlo sin rodeos: a la escala evaluada, la arquitectura integrada no reduce la latencia; es 1.9 ms más lenta en la mediana. La razón la da la propia descomposición, y ahora puede atribuirse a cada componente por separado. El acceso a metadatos que el diseño integrado elimina — la fila 3, las diez consultas SQLite por búsqueda — cuesta 0.16 ms. La búsqueda vectorial, en cambio, cuesta 2.45 ms con Qdrant en modo embebido frente a 0.18 ms con FAISS, que es una biblioteca compilada y especializada: el diseño integrado ahorra 0.16 ms en el paso que elimina y paga 2.27 ms de más en el paso que conserva. Ambas magnitudes son, de todos modos, irrelevantes al lado del componente que domina el presupuesto: los **13.42 ms de codificación de la consulta**,

idénticos en las dos arquitecturas porque emplean el mismo codificador, representan el 84 % del tiempo total en el esquema integrado y el 96 % en el de dos pasos.

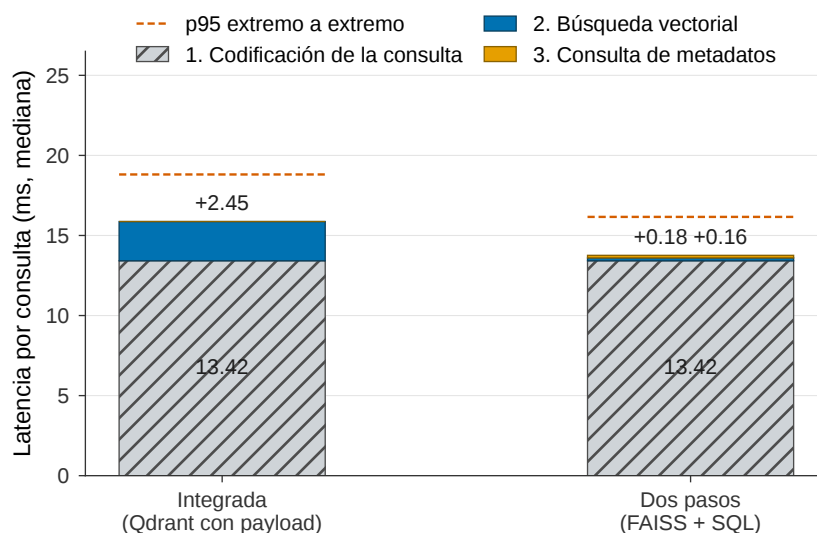


Fig. 5. Descomposición de la latencia por consulta: codificación frente a recuperación en cada arquitectura

Fuente: Elaboración propia, a partir de `evaluacion/benchmark_latencia.json` (1,000 consultas, CPU). La línea discontinua marca el percentil 95 extremo a extremo.

De aquí se siguen tres lecturas, ordenadas de más a menos cierta. Primera, y la única que las mediciones sostienen con firmeza: **el cuello de botella de un buscador semántico ligero sobre CPU es el codificador, no el almacén vectorial**. Optimizar el almacén a esta escala no mejora la experiencia del usuario de forma perceptible; cuantizar, destilar o poner en caché el codificador sí. Segunda: con 15.91 ms de p50 y 18.81 ms de p95, *ambas* arquitecturas quedan muy por debajo del techo de 100 ms exigido para la búsqueda en tiempo real, de modo que la decisión entre ellas no es una decisión de rendimiento. Tercera, y la que justifica mantener el diseño integrado: su ventaja es de *simplicidad operativa* — un solo almacén que mantener, respaldar y mantener sincronizado, en lugar de un índice y una base de metadatos que pueden divergir — y esa ventaja no se mide en milisegundos.

Quedan dos salvedades que acotan el alcance de la medición y que la Sección 4.6.2 recoge como limitaciones. La primera es la más obvia: el esquema de dos pasos se midió con SQLite en memoria y en el mismo proceso, que es su mejor caso posible; en un despliegue donde los metadatos vivan en un servicio separado, sus diez consultas por búsqueda costarían diez viajes de red, y el balance podría invertirse. No se reporta ninguna cifra para ese escenario porque no se midió. La segunda afecta a la lectura de los valores absolutos: repetir el mismo

script tres veces en la misma máquina produjo medianas extremo a extremo de entre 15.91 y 17.68 ms para la arquitectura integrada, de modo que la cifra concreta depende de la carga del equipo. Lo que se mantiene estable entre corridas es la proporción — la codificación ocupa entre el 82.9 % y el 85.1 % del total en el esquema integrado y entre el 95.6 % y el 95.8 % en el de dos pasos —, y es sobre esa proporción, y no sobre los milisegundos absolutos, que descansa la conclusión de este apartado.

La Tabla XXV consolida las tres comparaciones.

Tabla XXV
Resumen de todos los experimentos comparativos

Comparación	Ganador	Métrica clave	Valor p
Ajustado vs. modelo base	Ajustado	NDCG@10 +12.4 %	1.3×10^{-5}
Ajustado vs. BM25	Ajustado	MRR +11.6 %	1.5×10^{-4}
Integrada vs. dos pasos	Sin ganador	p50 +1.9 ms	9.9×10^{-135}

Fuente: Elaboración propia, a partir de `evaluacion/resultados_evaluacion_triple.json` y `evaluacion/benchmark_latencia.json`.

Nota: Resultados obtenidos con datos sintéticos y en condiciones controladas; las dos primeras filas se refieren a un catálogo parcialmente conocido (Sección 4.2.3). “Sin ganador” indica que la diferencia, aunque estadísticamente detectable por el gran número de mediciones pareadas, es de 1.9 ms sobre un presupuesto de 100 ms y carece de consecuencia práctica; el criterio de decisión entre ambas arquitecturas no es la latencia (Sección 4.5).

Discusión de Hallazgos

Interpretación de los Resultados

La mejora consistente en todas las métricas de recuperación aporta evidencia experimental y estadística compatible con la hipótesis central: en el escenario evaluado —datos sintéticos, condiciones controladas y un catálogo parcialmente conocido por el modelo— el *fine-tuning* supervisado de un modelo de *embeddings* ligero sobre pares consulta–producto mejora la recuperación de productos. Las ganancias del 15.8 % en MRR y 12.4 % en NDCG@10 se sitúan en el rango del 10–25 % que la literatura reporta para el *fine-tuning* supervisado en dominios especializados (Gururangan et al., 2020; Thakur et al., 2021); Recall@5 y Recall@10 ganan menos (9.4 % y 3.1 %) precisamente porque el modelo base ya recuperaba el producto correcto entre los diez primeros en el 96 % de las consultas, de modo que el margen disponible

estaba en el ordenamiento y no en la cobertura. Una explicación compatible con estos resultados es que el aprendizaje contrastivo expone al modelo al vocabulario y a las regularidades del catálogo que el preentrenamiento sobre corpus formales no cubre; el análisis de casos (Sección 4.3.2) indica que entre esas regularidades figuran tanto la distinción entre tipos de producto como particularidades del generador sintético, y la evidencia de que el efecto alcance específicamente a la jerga coloquial es, por ahora, preliminar (Tabla XX).

La ventaja del modelo ajustado sobre BM25 (+11.6 % en MRR, +8.5 % en NDCG@10) es consistente con la literatura, aunque de magnitud moderada, y depende del *fine-tuning*: el modelo base, sin *fine-tuning*, no superaba a BM25. Los métodos léxicos no resuelven la discrepancia de vocabulario, mientras que los *embeddings* pueden capturar relaciones semánticas sin coincidencia léxica, como ilustran los casos de la Tabla XXI. El margen debe leerse, no obstante, en el contexto de un conjunto donde las anclas comparten buena parte de su vocabulario con las descripciones de producto, lo que convierte a BM25 en una referencia especialmente fuerte, y donde la mayoría de las consultas de prueba reordena palabras vistas en entrenamiento. Estos resultados no permiten afirmar que el modelo ajustado pueda reemplazar a BM25 en producción: BM25 conserva su utilidad por su interpretabilidad término a término, su nula necesidad de entrenamiento y su comportamiento ante códigos y especificaciones exactas, lo que orienta el trabajo futuro hacia esquemas híbridos antes que hacia una sustitución.

La validación externa (Sección 4.4) acota de forma decisiva el alcance de los dos párrafos anteriores. Sobre 3,909 búsquedas reales de Mercado Libre México, la ventaja del modelo ajustado no se reproduce: BM25 recupera mejor el producto exacto, el modelo base identifica mejor el tipo de producto, y el *fine-tuning* solo mejora al modelo base en el primero de esos criterios, a costa del segundo. La explicación más directa es la distancia entre los datos con que se ajustó el modelo y los reales: 833 productos generados con plantillas, de descripción regular y con anclas que comparten vocabulario con ellas, frente a consultas más cortas, con marcas y modelos que en buena parte el generador no contempla, sobre un catálogo 61 veces mayor redactado por vendedores reales. El análisis de errores precisa el mecanismo: el modelo ajustado aprendió a apoyarse en la marca y el modelo, que en el conjunto sintético bastan para separar, por ejemplo, una funda de un teléfono, y que en el catálogo real no bastan. El resultado no invalida lo observado en el escenario controlado, pero delimita qué describe: el comportamiento del modelo sobre datos como los que lo entrenaron. Extrapolarlo a consultas

reales exige aplicar *fine-tuning* al modelo con datos de ese tipo.

Como evidencia cualitativa adicional de la calidad representacional del modelo ajustado, la Figura 6 presenta una proyección por Análisis de Componentes Principales (PCA, véase la explicación conceptual de la Sección 2.10) de los *embeddings* reales de 833 productos del catálogo, reducidos de 384 a 2 dimensiones. La figura colorea las diez categorías más frecuentes del conjunto de datos (de un total de 43 categorías finas) y agrupa las 552 observaciones restantes en gris. Se observan agrupamientos parciales por categoría (*cargador* y *funda*, por ejemplo, ocupan regiones con centroides claramente distintos), junto con un solapamiento apreciable entre categorías de accesorios electrónicos pequeños, lo cual es consistente con que estos productos comparten vocabulario técnico y contextos de uso similares en las consultas coloquiales de prueba. Cabe aclarar que esta proyección usa las categorías finas reales del conjunto `Data/training_data.json` (43 categorías de producto), y no una taxonomía de cuatro categorías amplias tipo “Hogar, Deportes, Automotriz y Electrónicos”; dado que las dos primeras componentes principales explican solo el 4.2 % y el 3.8 % de la varianza total, respectivamente, la separación observada en el plano 2D es parcial y debe interpretarse como evidencia cualitativa complementaria, no como prueba definitiva de separabilidad total del espacio de *embeddings*.

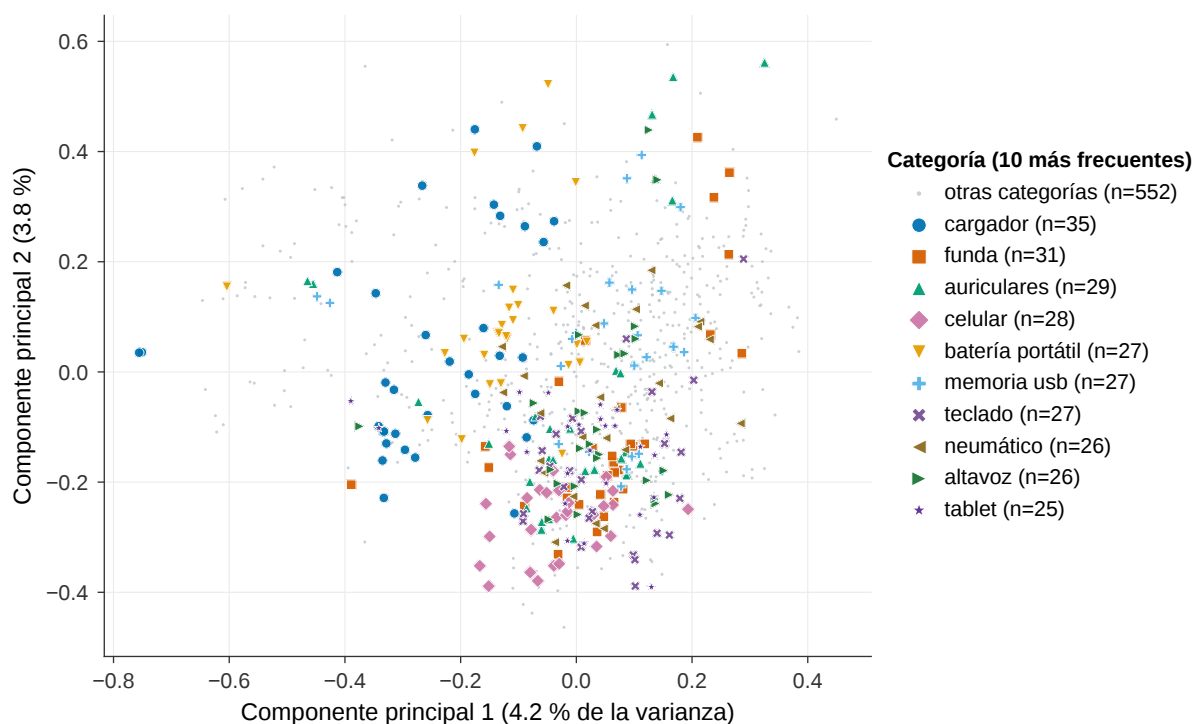


Fig. 6. Proyección PCA (384 → 2 dimensiones) de los *embeddings* reales del catálogo, coloreados por categoría

Fuente: Elaboración propia, a partir de `Data/training_data.json` y el modelo ajustado; script reproducible en `figuras/fig_pca_embeddings.py`.

En el plano de la ingeniería, el hallazgo con más consecuencias prácticas no es el que se esperaba encontrar. La comparación arquitectónica muestra que, a la escala evaluada, la elección del almacén vectorial es indiferente para la latencia y que el presupuesto de tiempo por consulta lo consume casi íntegramente la codificación de la consulta (13.42 de 15.91 ms). Con un p95 de 18.81 ms, el sistema opera holgadamente por debajo del techo de 100 ms sobre *hardware* de CPU en el entorno de medición, y la validación funcional de la API lo confirma a nivel de punto de acceso: 1,000 peticiones `POST /buscar` sobre el catálogo de demostración de 500 productos — distinto del corpus de 833 empleado para las métricas de recuperación, como se precisó en la Sección 3.2.5 — se resolvieron con un 100 % de éxito (código 200 y diez resultados) y un tiempo medio de respuesta de 18.95 ms (p95 de 21.81 ms), medido en proceso mediante `TestClient` y registrado en `evaluacion/validacion_api.json`. Estas cifras aportan evidencia preliminar de viabilidad operativa en el escenario experimental evaluado, con la salvedad de que no incluyen ni la pila TCP ni la concurrencia de un despliegue real. Si se confirman con consultas reales, estos hallazgos serían especialmente pertinentes para el comercio electrónico latinoamericano, donde la brecha entre lenguaje coloquial y terminología

de catálogo es pronunciada y la infraestructura disponible suele limitarse a CPU.

Limitaciones

Las limitaciones se ordenan de mayor a menor peso sobre las conclusiones del Capítulo V.

- **Datos sintéticos y condiciones controladas:** el entrenamiento y todas las cifras de este capítulo, salvo las de la Sección 4.4, se obtuvieron sobre pares generados por plantillas y reglas elaboradas por el autor (Sección 3.2.1), no sobre consultas de usuarios, y en un entorno controlado (una sola máquina, sin carga concurrente, mediciones en proceso). Los resultados muestran evidencia favorable al enfoque en ese escenario, pero no constituyen una validación en un entorno de producción. La validación externa con datos reales de Mercado Libre indica, además, que no se trasladan a consultas reales: en ella el modelo ajustado no supera a BM25 ni al modelo base. Esa validación tiene a su vez límites propios —consultas de México, relevancia implícita por la compra posterior y ausencia de juicios de relevancia humanos—, de modo que mide la dirección del efecto con más fiabilidad que su magnitud.
- **Catálogo parcialmente conocido:** 140 de las 165 consultas de prueba (84.8 %) se refieren a un producto que también aparece en el entrenamiento, y en 136 de ellas una consulta de entrenamiento del mismo producto contiene las mismas palabras en otro orden, porque el generador produce dos anclas por producto. La separación garantiza que el modelo no vio esas *cadenas*, pero sí esos *productos* y, casi siempre, esas *palabras*; las métricas estiman, por tanto, la generalización a consultas nuevas sobre un catálogo parcialmente conocido. La generalización a productos no vistos solo puede examinarse sobre 25 consultas (Tabla XVIII); el resultado es favorable pero preliminar. Medirla con solidez exigiría una partición disjunta por producto, que a la escala del conjunto dejaría subgrupos demasiado pequeños para la inferencia.
- **Solapamiento literal de formulaciones:** 26 de las 165 anclas de prueba aparecen también en entrenamiento asociadas a otro producto, porque el conjunto base contiene 1,656 pares sobre 1,534 anclas únicas. Las ganancias se mantienen al excluirlas (Tabla XVII), pero la cifra agregada de la Tabla XVI incluye ese solapamiento y así se declara.
- **Evidencia preliminar sobre jerga coloquial:** el 85.8 % de las anclas no presenta ningún marcador coloquial, y de las 23 consultas de prueba que sí lo presentan, 16 lo hacen porque el nombre de la categoría o la especificación es un anglicismo o *llanta*. Las con-

sultas con jerga escrita como tal son siete; en ellas el modelo ajustado supera a BM25 en MRR, pero la diferencia no es estadísticamente distinguible del azar (Tabla XX). Los resultados sobre lenguaje coloquial deben tomarse como indicativos de una dirección, no como una conclusión.

- **Artefactos del generador:** 157 de los 833 productos tienen un casi-duplicado que solo difiere en la frase promocional, y en 27 de los 35 errores del modelo ajustado el producto antepuesto es uno de ellos, lo que sesga las métricas a la baja para todos los sistemas. En sentido contrario, la plantilla de cargadores asocia en 42 de 46 pares una consulta con un producto de otra potencia, una regularidad que el modelo ajustado puede aprender y BM25 no, y 38 descripciones repiten la especificación del modelo, lo que lleva a BM25 a anteponerlas en 10 de las consultas que pierde. Al excluir la categoría de cargadores la ventaja sobre BM25 se mantiene (9.6 % en MRR), pero el efecto conjunto de estas regularidades no se cuantificó por completo.
- **Vocabulario coloquial de selección propia:** `Data/colloquial_words.json` se elaboró sin un protocolo de recopilación documentado y su pertinencia se verificó *a posteriori* contra el DLE y el DAmer (Sección 3.2.3). Mezcla formas con uso registrado en Ecuador, Colombia y Perú con formas de uso general y de otras variedades; 20 de sus 62 claves nunca se activan, 22 de las 43 categorías no reciben variantes en las consultas, el 4.9 % de las variantes generadas son espurias por un defecto de la regla de sustitución y omite términos de uso regional frecuente como *parlante*.
- **Condiciones de la medición de latencia:** las cifras de la Tabla XXIV se obtuvieron con Qdrant en modo embebido y SQLite en memoria, ambos en el mismo proceso que el codificador, por no disponer de Docker en el entorno de medición. No son, por tanto, una medida del rendimiento de Qdrant como servicio ni del costo real de un almacén de metadatos remoto; miden el costo relativo de los componentes bajo el mejor caso posible para el esquema de dos pasos. Los valores absolutos dependen además de la máquina: tres corridas consecutivas del mismo script dieron medianas extremo a extremo entre 15.91 y 17.68 ms. La conclusión que sí soportan — que la codificación domina el presupuesto por consulta, con una proporción estable entre el 83 % y el 96 % según la arquitectura — es robusta frente a ambas salvedades, porque cualquier despliegue realista solo puede aumentar el costo del almacén, nunca el de la codificación.
- **Homonimia no resuelta en el catálogo:** las categorías *llanta* (aro) y *neumático* (caucho)

coexisten como productos distintos, mientras que en el habla de la región “llanta” se usa habitualmente para nombrar el segundo (Sección 3.2.6), y la propia aumentación introduce *llanta* como variante de *neumático*. El conjunto asigna un único positivo correcto por consulta, de modo que las consultas afectadas introducen ruido de etiqueta cuyo signo y magnitud no se cuantificaron.

- **Escala del catálogo:** el corpus de recuperación consta de 833 productos y el catálogo auxiliar de la API de 500. Escalar a más de 100,000 entradas requeriría una evaluación adicional de latencia y gestión del índice.
- **Alcance lingüístico:** el estudio presta atención particular a formas de uso registrado en Ecuador, Colombia y Perú, pero el diseño no permite atribuir ningún resultado a una variedad nacional concreta ni compara variedades entre sí; tampoco aborda otros idiomas.
- **Inferencia solo en CPU:** las cifras de latencia reflejan inferencia en CPU; el despliegue en GPU las reduciría aún más, pero queda fuera del alcance del entorno objetivo actual.

Capítulo V

Conclusiones y Recomendaciones

Conclusiones

Este trabajo evaluó el *fine-tuning* de un modelo de *embeddings* ligero (MiniLM-L6-v2) para la recuperación semántica de productos en comercio electrónico, orientado a la jerga coloquial en español que emplean con frecuencia los usuarios latinoamericanos. Las conclusiones responden a los objetivos formulados en el Capítulo I y tienen un alcance preciso: se obtuvieron con un conjunto de datos sintético, generado por plantillas y reglas, en condiciones experimentales controladas y sobre un catálogo que el modelo conocía en buena parte, y deben leerse dentro de ese marco. La única excepción es la validación externa con búsquedas reales de Mercado Libre, que se resume en el cuarto punto y que acota el alcance de los tres primeros.

- **Fine-tuning frente al modelo base.** El *fine-tuning* con *Multiple Negatives Ranking Loss* sobre 9,684 pares coloquial-técnico mejoró el MRR en un 15.8 % (de 0.766 a 0.887) y el NDCG@10 en un 12.4 % (de 0.814 a 0.915) sobre una partición de prueba de 165 consultas que no intervino en la selección del punto de control. La diferencia es estadísticamente significativa según la prueba de rangos con signo de Wilcoxon ($p = 1.3 \times 10^{-5}$), con un tamaño del efecto pareado pequeño ($d_z = 0.383$): la mejora es sistemática y no atribuible al azar, pero su magnitud es moderada. Como 136 de las 165 consultas de prueba reordenan palabras de una consulta de entrenamiento del mismo producto, la cifra mide sobre todo la generalización a consultas nuevas sobre un catálogo parcialmente conocido; en las 25 consultas sobre productos no vistos la mejora se mantuvo, con una muestra que solo permite considerarla preliminar.
- **Modelo ajustado frente a BM25.** El modelo ajustado superó a BM25 con un incremento del 11.6 % en MRR (0.887 frente a 0.795), con un tamaño del efecto pequeño ($d_z = 0.332$, $p = 1.5 \times 10^{-4}$), y la ventaja se mantuvo al excluir la categoría afectada por la inconsistencia del generador (9.6 %). Sin *fine-tuning*, el modelo base no superaba a BM25, de modo que la ventaja procede del *fine-tuning*. El análisis de casos muestra que el modelo ajustado distingue el tipo de producto cuando la consulta comparte palabras con productos de otro tipo, que parte de su ventaja procede de particularidades del generador sintético

y que la mayoría de sus errores se debe a casi-duplicados del catálogo. Estos resultados aportan evidencia favorable a la búsqueda semántica ajustada en el escenario evaluado, pero no bastan para sostener que pueda reemplazar a BM25 de forma general; su complementariedad, mediante un esquema híbrido, es la lectura más prudente.

- **Jerga coloquial.** La evidencia específica sobre lenguaje coloquial es preliminar. En las siete consultas de prueba con jerga escrita como tal, el modelo ajustado obtuvo un MRR mayor que BM25 (0.663 frente a 0.481), pero la diferencia no es estadísticamente distinguible del azar con ese tamaño de muestra, y los fallos compartidos coinciden con términos que el vocabulario de aumentación no cubre. La dirección del resultado es compatible con la hipótesis central, pero no se observó en las búsquedas reales de Mercado Libre con marcadores coloquiales, donde el modelo ajustado no superó a BM25 ni al modelo base; esos marcadores son, sobre todo, regionalismos de uso general y no jerga andina. Confirmar o descartar la hipótesis exige un conjunto de evaluación coloquial más amplio y procedente de usuarios de la región.
- **Validación externa con consultas reales.** Sobre 3,909 búsquedas reales de Mercado Libre México, tomadas del MeLi Data Challenge 2020 y asociadas al producto que cada usuario compró después, la ventaja obtenida con datos sintéticos no se reprodujo. BM25 recuperó mejor el producto comprado (MRR de 0.044, frente a 0.037 del modelo ajustado y 0.033 del modelo base), y el modelo base identificó mejor el tipo de producto (74.0 % de acierto de dominio en la primera posición, frente a 72.9 % de BM25 y 67.8 % del modelo ajustado). El *fine-tuning* mejoró al modelo base en la recuperación del producto exacto, pero empeoró la identificación del tipo de producto, sobre todo en electrónica; solo en accesorios personales superó a los otros dos sistemas. Los tamaños de efecto son insignificantes o pequeños y la relevancia es implícita, pero el resultado es claro en su dirección: el modelo ajustado únicamente con datos sintéticos no generaliza por sí solo a consultas reales, y su uso en un entorno real exigiría aplicarle *fine-tuning* con consultas reales (Sección 4.4).
- **Arquitectura y latencia.** La comparación arquitectónica no confirmó la hipótesis de partida y ese es, en sí mismo, un resultado de ingeniería útil: la arquitectura integrada basada en Qdrant con almacenamiento de carga útil no redujo la latencia frente a la arquitectura de dos pasos FAISS+SQL (15.91 frente a 14.04 ms de mediana). La descomposición por componente explica por qué: la codificación de la consulta consume 13.42 ms — entre

el 83 % y el 96 % del presupuesto según la arquitectura — mientras que el acceso a metadatos que el diseño integrado elimina cuesta 0.16 ms y la búsqueda vectorial que conserva cuesta 2.45 ms con Qdrant embebido frente a 0.18 ms con FAISS. **En el entorno medido, el cuello de botella de un buscador semántico ligero sobre CPU es el codificador, no el almacén vectorial.** Ambas arquitecturas se mantienen holgadamente por debajo del techo de 100 ms, de modo que la elección entre ellas debe decidirse por simplicidad operativa y no por rendimiento.

- **Viabilidad operativa.** El sistema basado en FastAPI alcanzó una tasa de éxito del 100 % sobre 1,000 consultas de prueba, con un tiempo de respuesta promedio de 18.95 ms (p95 de 21.81 ms), lo que aporta evidencia preliminar de viabilidad de operación en CPU con latencia inferior a 100 ms a la escala evaluada. Conviene no confundir las dos poblaciones de productos que intervienen: las métricas de recuperación se calculan sobre el corpus de 833 productos del conjunto de datos, mientras que esta validación funcional se ejecuta sobre el catálogo auxiliar de 500 productos generados. La medición se realizó en proceso, sin pila TCP ni concurrencia, y así se declara en la Sección 4.6.2.

En conjunto, la evidencia indica que, en el escenario evaluado, un modelo de *embeddings* ligero ajustado estrecha la brecha entre las consultas y la terminología técnica del catálogo, y lo hace dentro de un presupuesto de latencia compatible con la búsqueda interactiva en CPU. La validación con búsquedas reales de Mercado Libre muestra, sin embargo, que esa mejora depende de que los datos de *fine-tuning* se parezcan a las consultas que el sistema recibirá: ajustado solo con datos sintéticos, el modelo no supera a BM25 ni al modelo base sobre consultas reales. El sistema funciona de extremo a extremo y el protocolo de evaluación es reproducible; la ventaja de recuperación en un entorno real queda por demostrar mediante un *fine-tuning* con consultas reales y una evaluación en producción, que son el objeto de las recomendaciones de la Sección 5.3. El margen de mejora en rendimiento, según muestra la descomposición de la latencia, está en el codificador y no en la capa de almacenamiento.

Aporte de Ingeniería: Producto Entregado

Más allá de los resultados experimentales, es necesario responder con precisión a la pregunta que orienta a un trabajo de titulación en Ingeniería: ¿qué construyó el estudiante para graduarse? El producto entregado por esta investigación no es únicamente un modelo de aprendizaje automático, sino un sistema completo de ingeniería de *software*, diseñado, implementado y validado experimentalmente de extremo a extremo.

Producto entregado. Un microservicio de búsqueda semántica, expuesto como una API REST construida con FastAPI (Sección 3.1), estructurado bajo el patrón de Arquitectura Limpia en cuatro capas independientes (Dominio, Aplicación, Infraestructura, Presentación; Sección 3.1.1), que integra un modelo de *embeddings* ligero (MiniLM-L6-v2, 22M de parámetros, aproximadamente 90 MB) ajustado mediante aprendizaje contrastivo sobre un conjunto sintético de consultas coloquiales, junto con una base de datos vectorial (Qdrant) que almacena la carga útil del producto junto a cada vector. El sistema está preparado para integrarse mediante llamadas HTTP con una plataforma de comercio electrónico sin requerir infraestructura de GPU; su adopción en producción debería ir precedida de una validación con el catálogo y las consultas reales de esa plataforma.

Este aporte se manifiesta en tres planos concretos:

- **Aporte algorítmico.** La adaptación de dominio de un modelo de *embeddings* preentrenado mediante *Multiple Negatives Ranking Loss* sobre un corpus sintético construido para aproximar la jerga coloquial latinoamericana, con una mejora medible y estadísticamente significativa frente al modelo base sobre la partición de prueba (Tabla XVI).
- **Aporte arquitectónico.** El diseño e implementación de un sistema modular bajo Arquitectura Limpia, verificado mediante una suite de pruebas automatizadas (Tabla VII) y comparado empíricamente frente a una arquitectura alternativa de dos pasos. La medición acotó dónde está realmente el costo por consulta (Tabla XXIV) y, al no confirmar la ventaja de latencia que se presuponía, reorientó el criterio de decisión entre ambas arquitecturas hacia la simplicidad operativa.
- **Aporte metodológico.** Un protocolo de evaluación reproducible (métricas de ordenamiento, prueba de significancia estadística, desgloses por solapamiento con el entrenamiento, percentiles de latencia y una validación externa con datos públicos de Mercado Libre que cualquiera puede repetir con el mismo archivo) que puede aplicarse a la validación de sistemas de recuperación semántica similares, más allá del caso de estudio específico de este trabajo.

En conjunto, este producto demuestra la capacidad del estudiante para llevar una idea de investigación desde el planteamiento del problema hasta un sistema de *software* funcional y medible, articulando conocimientos de aprendizaje automático, ingeniería de *software* y validación experimental, el perfil de egreso esperado de un Ingeniero en Sistemas de Información.

Recomendaciones y Trabajo Futuro

A partir de los hallazgos y de las limitaciones identificadas durante esta investigación, se recomiendan las siguientes líneas de trabajo futuro y de mejora del sistema:

- **Optimización del codificador.** Dado que la codificación de la consulta consume entre el 83 % y el 96 % del presupuesto de latencia (Sección 4.5), las vías con mayor rendimiento esperado son la cuantización del codificador a enteros de 8 bits, su destilación a un modelo de menos capas y una caché de *embeddings* para las consultas más frecuentes. Cualquiera de ellas tendría más impacto sobre la latencia percibida que sustituir el almacén vectorial.
- **Medición de la arquitectura en despliegue real.** Repetir la comparación arquitectónica con Qdrant como servicio y con el almacén de metadatos en un proceso separado, condiciones que la medición actual no cubre (Sección 4.6.2) y bajo las cuales el balance entre ambas arquitecturas podría invertirse.
- **Motor de búsqueda híbrido.** Combinar BM25 y búsqueda semántica en un único flujo de recuperación: BM25 para consultas que incluyen códigos de producto, SKU o números de modelo exactos, donde la coincidencia léxica es por diseño más confiable (Sección 4.3) y donde, en las búsquedas reales de Mercado Libre, resultó la más eficaz (Sección 4.4), y búsqueda semántica para el lenguaje coloquial e informal, donde esta investigación encontró evidencia preliminar a su favor en el escenario sintético. Un mecanismo de enrutamiento o de combinación de puntuaciones con reordenamiento permitiría aprovechar ambos enfoques sin sacrificar la latencia en el caso léxico simple; propuestas recientes de ajuste dinámico del peso relativo entre el componente léxico y el semántico según las características de cada consulta (Hsu & Tzeng, 2025) ofrecen un punto de partida concreto para implementar este enrutamiento sin necesidad de una regla fija predefinida.
- **Fine-tuning con consultas reales y conjunto coloquial auténtico.** Es la recomendación prioritaria a la luz de la validación externa: aplicar *fine-tuning* al modelo con consultas reales y no solo evaluarlo con ellas. Los historiales del MeLi Data Challenge 2020 permiten construir pares consulta–producto comprado a partir de búsquedas reales (**meli2020kaggle**); separando sesiones de entrenamiento y de prueba, bastaría repetir el protocolo de la Sección 3.5.4 para medir si el *fine-tuning* con datos reales invierte el resultado. El paso siguiente sería reunir al menos 5,000 consultas reales de usuarios de la región andina, anotadas con juicios de relevancia expertos y con varios productos aceptables por

consulta, con el fin de aumentar la validez ecológica de la evaluación y de disponer de subgrupos coloquiales con tamaño suficiente para la inferencia.

- **Validación del vocabulario coloquial.** Contrastar y ampliar el vocabulario de aumentación con hablantes de Ecuador, Colombia y Perú o con un corpus de consultas reales, depurar las variantes sin registro o de otras variedades, incorporar términos ausentes de uso frecuente (como *parlante*), extender la cobertura a las categorías que hoy no reciben variantes y corregir la regla de sustitución para que respete los límites de palabra (Sección 3.2.3).
- **Corrección del generador y partición por producto.** Eliminar los casi-duplicados del catálogo, corregir la plantilla de cargadores y repetir la evaluación con una partición disjunta por producto, de modo que la generalización a productos no vistos pueda medirse con una muestra mayor que las 25 consultas disponibles (Sección 4.2.3).
- **Escalamiento del catálogo.** Validar la calidad de recuperación y la latencia en catálogos de 100,000 productos o más, incluyendo el ajuste de los parámetros del índice HNSW de Qdrant para preservar tiempos de respuesta inferiores a 100 ms a escala.
- **Extensión multilingüe.** Adaptar el enfoque a la recuperación translingüe (español–inglés–portugués) para atender los mercados regionales que operan en varios idiomas.
- **Comparación con modelos más grandes.** Comparar el modelo de 22M de parámetros frente a un MiniLM-L12-v2 multilingüe (118M de parámetros) para cuantificar el compromiso entre calidad y latencia sobre *hardware* de CPU.
- **Pruebas A/B en producción.** Desplegar el sistema en una plataforma de comercio electrónico en vivo y medir su impacto sobre la tasa de conversión bajo condiciones de tráfico real, como validación industrial de los resultados experimentales.

Referencias Bibliográficas

- Alonso, O., & Mizzaro, S. (2012). Using Crowdsourcing for TREC Relevance Assessment. *Information Processing and Management*, 48(6), 1053-1066. <https://doi.org/10.1016/j.ipm.2012.01.004>
- Artstein, R., & Poesio, M. (2008). Survey Article: Inter-Coder Agreement for Computational Linguistics. *Computational Linguistics*, 34(4), 555-596. <https://doi.org/10.1162/coli.07-034-R2>
- Bailey, P., Craswell, N., Soboroff, I., Thomas, P., de Vries, A. P., & Yilmaz, E. (2008). Relevance Assessment: Are Judges Exchangeable and Does It Matter? *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 667-674. <https://doi.org/10.1145/1390334.1390447>
- Bender, E. M., & Friedman, B. (2018). Data Statements for Natural Language Processing: Toward Mitigating System Bias and Enabling Better Science. *Transactions of the Association for Computational Linguistics*, 6, 587-604. https://doi.org/10.1162/tacl_a_00041
- Bonifacio, L., Abonizio, H., Fadaee, M., & Nogueira, R. (2022). InPars: Data Augmentation for Information Retrieval using Large Language Models. *arXiv preprint arXiv:2202.05144*. <https://doi.org/10.48550/arXiv.2202.05144>
- Buckley, C., & Voorhees, E. M. (2004). Retrieval Evaluation with Incomplete Information. *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 25-32. <https://doi.org/10.1145/1008992.1009000>
- Cañete, J., Chaperon, G., Fuentes, R., Ho, J.-H., Kang, H., & Pérez, J. (2020). Spanish Pre-trained BERT Model and Evaluation Data. *PML4DC Workshop at the International Conference on Learning Representations (ICLR)*. <https://doi.org/10.48550/arXiv.2308.02976>
- Cao, H. (2024). Recent Advances in Text Embedding: A Comprehensive Review of Top-Performing Methods on the MTEB Benchmark. *arXiv preprint arXiv:2406.01607*. <https://doi.org/10.48550/arXiv.2406.01607>
- Chari, A., MacAvaney, S., & Ounis, I. (2023). On the Effects of Regional Spelling Conventions in Retrieval Models. *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. <https://doi.org/10.1145/3539618.3592030>
- Chen, J., Xiao, S., Zhang, P., Luo, K., Lian, D., & Liu, Z. (2024). M3-Embedding: Multi-Linguality, Multi-Functionality, Multi-Granularity Text Embeddings Through Self-Knowledge Distillation. *arXiv preprint arXiv:2402.03216*. <https://doi.org/10.48550/arXiv.2402.03216>
- Cleverdon, C. W. (1967). The Cranfield Tests on Index Language Devices. *Aslib Proceedings*, 19(6), 173-194. <https://doi.org/10.1108/eb050097>
- Cohen, J. (1988). *Statistical Power Analysis for the Behavioral Sciences* (2.^a ed.). Lawrence Erlbaum Associates.
- Dai, Z., Zhao, V. Y., Ma, J., Luan, Y., Ni, J., Lu, J., Bakalov, A., Guu, K., Hall, K. B., & Chang, M.-W. (2023). Promptagator: Few-shot Dense Retrieval From 8 Examples. *Proceedings of the 11th International Conference on Learning Representations (ICLR)*. <https://doi.org/10.48550/arXiv.2209.11755>
- Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT)*, 4171-4186. <https://doi.org/10.18653/v1/N19-1423>
- Dwork, C., Feldman, V., Hardt, M., Pitassi, T., Reingold, O., & Roth, A. (2015). The Reusable Holdout: Preserving Validity in Adaptive Data Analysis. *Science*, 349(6248), 636-638. <https://doi.org/10.1126/science.aaa9375>
- Enevoldsen, K., Chung, I., Kerboua, I., Kardos, M., Mathur, A., Stap, D., et al. (2025). MMTEB: Massive Multilingual Text Embedding Benchmark. *International Conference on Learning Representations (ICLR)*. <https://doi.org/10.48550/arXiv.2502.13595>

- Formal, T., Piwowarski, B., & Clinchant, S. (2021). SPLADE: Sparse Lexical and Expansion Model for First Stage Ranking. *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. <https://doi.org/10.1145/3404835.3463098>
- Fuhr, N. (2017). Some Common Mistakes In IR Evaluation, And How They Can Be Avoided. *ACM SIGIR Forum*, 51(3), 32-41. <https://doi.org/10.1145/3190580.3190586>
- Furnas, G. W., Landauer, T. K., Gomez, L. M., & Dumais, S. T. (1987). The Vocabulary Problem in Human-System Communication. *Communications of the ACM*, 30(11), 964-971. <https://doi.org/10.1145/32206.32212>
- Gebru, T., Morgenstern, J., Vecchione, B., Wortman Vaughan, J., Wallach, H., Daumé III, H., & Crawford, K. (2021). Datasheets for Datasets. *Communications of the ACM*, 64(12), 86-92. <https://doi.org/10.1145/3458723>
- Gururangan, S., Marasović, A., Swayamdipta, S., Lo, K., Beltagy, I., Downey, D., & Smith, N. A. (2020). Don't Stop Pretraining: Adapt Language Models to Domains and Tasks. *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, 8342-8360. <https://doi.org/10.18653/v1/2020.acl-main.740>
- He, Y., Tian, Y., Wang, M., Chen, F., Yu, L., Tang, M., Chen, C., Zhang, N., Kuang, B., & Prakash, A. (2023). Que2Engage: Embedding-based Retrieval for Relevant and Engaging Products at Facebook Marketplace. *arXiv preprint arXiv:2302.11052*. <https://doi.org/10.48550/arXiv.2302.11052>
- Henderson, M., Al-Rfou, R., Strobe, B., Sung, Y.-H., Lukacs, L., Guo, R., Kumar, S., Miklos, B., & Kurzweil, R. (2017). Efficient Natural Language Response Suggestion for Smart Reply. *arXiv preprint arXiv:1705.00652*. <https://doi.org/10.48550/arXiv.1705.00652>
- Hsu, H.-L., & Tzeng, J. (2025). DAT: Dynamic Alpha Tuning for Hybrid Retrieval in Retrieval-Augmented Generation. *arXiv preprint arXiv:2503.23013*. <https://doi.org/10.48550/arXiv.2503.23013>
- Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., & Chen, W. (2021). LoRA: Low-Rank Adaptation of Large Language Models. *arXiv preprint arXiv:2106.09685*. <https://doi.org/10.48550/arXiv.2106.09685>
- IEEE Reliability Society. (2016). *IEEE Recommended Practice on Software Reliability* (inf. téc. N.º IEEE Std 1633-2016). Institute of Electrical y Electronics Engineers. <https://doi.org/10.1109/IEEESTD.2017.7827907>
- International Electrotechnical Commission. (2018). *Failure Modes and Effects Analysis (FMEA and FMECA)* (3.^a ed., inf. téc. N.º IEC 60812:2018). International Electrotechnical Commission.
- Jégou, H., Douze, M., & Schmid, C. (2011). Product Quantization for Nearest Neighbor Search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(1), 117-128. <https://doi.org/10.1109/TPAMI.2010.57>
- Jha, R., Subramaniyam, S., Benjamin, E., & Taula, T. (2023). Unified Embedding Based Personalized Retrieval in Etsy Search. *arXiv preprint arXiv:2306.04833*. <https://doi.org/10.48550/arXiv.2306.04833>
- Johnson, J., Douze, M., & Jégou, H. (2019). Billion-scale Similarity Search with GPUs. *IEEE Transactions on Big Data*, 7(3), 535-547. <https://doi.org/10.1109/TBDATA.2019.2921572>
- Joint Committee for Guides in Metrology. (2008). *Evaluation of Measurement Data — Guide to the Expression of Uncertainty in Measurement (GUM)* (inf. téc. N.º JCGM 100:2008) (Equivalente a ISO/IEC Guide 98-3). Bureau International des Poids et Mesures (BIPM).
- Jolliffe, I. T., & Cadima, J. (2016). Principal Component Analysis: A Review and Recent Developments. *Philosophical Transactions of the Royal Society A*, 374(2065), 20150202. <https://doi.org/10.1098/rsta.2015.0202>
- Karanam, S., & Bhargav, J. (2026). Microservice Architecture Patterns for Scalable Machine Learning Systems. *arXiv preprint arXiv:2603.13672*. <https://doi.org/10.48550/arXiv.2603.13672>
- Karpukhin, V., Oğuz, B., Min, S., Lewis, P., Wu, L., Edunov, S., Chen, D., & Yih, W.-t. (2020). Dense Passage Retrieval for Open-Domain Question Answering. *Proceedings of the 2020 Conference*

- on *Empirical Methods in Natural Language Processing (EMNLP)*, 6769-6781. <https://doi.org/10.18653/v1/2020.emnlp-main.550>
- Kothari, N., Samdani, S., Mallick, R., Gupta, P., Vijay, A., & Kumar, S. (2026). Semantic Retrieval for Product Search in E-Commerce. *arXiv preprint arXiv:2606.01504*. <https://doi.org/10.48550/arXiv.2606.01504>
- Krastev, M., Hamar, M., Toapanta, D., Brouwers, J., & Lei, Y. (2025). InPars+: Supercharging Synthetic Data Generation for Information Retrieval Systems. *arXiv preprint arXiv:2508.13930*. <https://doi.org/10.48550/arXiv.2508.13930>
- Kusupati, A., Bhatt, G., Rege, A., Wallingford, M., Sinha, A., Ramanujan, V., Howard-Snyder, W., Chen, K., Kakade, S., Jain, P., & Farhadi, A. (2022). Matryoshka Representation Learning. *Advances in Neural Information Processing Systems (NeurIPS)*, 35. <https://doi.org/10.48550/arXiv.2205.13147>
- Lakens, D. (2013). Calculating and Reporting Effect Sizes to Facilitate Cumulative Science: A Practical Primer for *t*-Tests and ANOVAs. *Frontiers in Psychology*, 4, 863. <https://doi.org/10.3389/fpsyg.2013.00863>
- Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-t., Rocktäschel, T., Riedel, S., & Kiela, D. (2020). Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. *Advances in Neural Information Processing Systems (NeurIPS)*, 33, 9459-9474. <https://doi.org/10.48550/arXiv.2005.11401>
- Lin, J., Yadav, S., Liu, F., Rossi, N., Suram, P. R., Chembolu, S., Chandran, P., Mohapatra, H., Lee, T., Magnani, A., & Liao, C. (2024). Enhancing Relevance of Embedding-based Retrieval at Walmart. *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management (CIKM)*, 4694-4701. <https://doi.org/10.1145/3627673.3680047>
- Lin, Y., Zhang, K., He, Z., Jing, Y., & Wang, X. S. (2025). Survey of Filtered Approximate Nearest Neighbor Search over the Vector-Scalar Hybrid Data. *arXiv preprint arXiv:2505.06501*. <https://doi.org/10.48550/arXiv.2505.06501>
- Lyu, M. R. (Ed.). (1996). *Handbook of Software Reliability Engineering*. McGraw-Hill; IEEE Computer Society Press.
- Magnani, A., Liu, F., Chaidaroon, S., Yadav, S., Suram, P. R., Puthenpuhussery, A., Chen, S., Xie, M., Kashi, A., Lee, T., & Liao, C. (2022). Semantic Retrieval at Walmart. *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, 3495-3503. <https://doi.org/10.1145/3534678.3539164>
- Malaiya, Y. K. (1982). Linearly Correlated Intermittent Failures. *IEEE Transactions on Reliability*, R-31(2), 211-215. <https://doi.org/10.1109/TR.1982.5221310>
- Malkov, Y. A., & Yashunin, D. A. (2018). Efficient and Robust Approximate Nearest Neighbor Search Using Hierarchical Navigable Small World Graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(4), 824-836. <https://doi.org/10.1109/TPAMI.2018.2889473>
- Manning, C. D., Raghavan, P., & Schütze, H. (2008). *Introduction to Information Retrieval*. Cambridge University Press.
- McInnes, L., Healy, J., & Melville, J. (2018). UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. *arXiv preprint arXiv:1802.03426*. <https://doi.org/10.48550/arXiv.1802.03426>
- Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient Estimation of Word Representations in Vector Space. *arXiv preprint arXiv:1301.3781*. <https://doi.org/10.48550/arXiv.1301.3781>
- Muennighoff, N., Tazi, N., Magne, L., & Reimers, N. (2023). MTEB: Massive Text Embedding Benchmark. *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*. <https://doi.org/10.48550/arXiv.2210.07316>
- Nguyen, T., Rosenberg, M., Song, X., Gao, J., Tiwary, S., Majumder, R., & Deng, L. (2016). MS MARCO: A Human Generated Machine Reading Comprehension Dataset. *Proceedings of the Workshop on Cognitive Computation (CoCo) at NIPS 2016*. <https://doi.org/10.48550/arXiv.1611.09268>

- Nogueira, R., Yang, W., Lin, J., & Cho, K. (2019). Document Expansion by Query Prediction. *arXiv preprint arXiv:1904.08375*. <https://doi.org/10.48550/arXiv.1904.08375>
- Northcutt, C. G., Athalye, A., & Mueller, J. (2021). Pervasive Label Errors in Test Sets Destabilize Machine Learning Benchmarks. *Advances in Neural Information Processing Systems (NeurIPS), Track on Datasets and Benchmarks*. <https://doi.org/10.48550/arXiv.2103.14749>
- O'Connor, P. D. T., & Kleyner, A. (2012). *Practical Reliability Engineering* (5.^a ed.). John Wiley & Sons.
- Pandey, M. (2026). Domain-Adaptive and Scalable Dense Retrieval for Content-Based Recommendation. *arXiv preprint arXiv:2602.00899*. <https://doi.org/10.48550/arXiv.2602.00899>
- Pérez, J. M., Furman, D. A., Alonso Alemany, L., & Luque, F. M. (2022). RoBERTuito: A Pre-trained Language Model for Social Media Text in Spanish. *Proceedings of the Thirteenth Language Resources and Evaluation Conference (LREC)*, 7235-7243. <https://doi.org/10.48550/arXiv.2111.09453>
- Qdrant. (2026). Qdrant Vector Database Documentation. <https://qdrant.tech/documentation/>
- Recht, B., Roelofs, R., Schmidt, L., & Shankar, V. (2019). Do ImageNet Classifiers Generalize to ImageNet? *Proceedings of the 36th International Conference on Machine Learning (ICML)*, PMLR 97. <https://doi.org/10.48550/arXiv.1902.10811>
- Reddy, C. K., Márquez, L., Valero, F., Rao, N., Zaragoza, H., Bandyopadhyay, S., Biswas, A., Xing, A., & Subbian, K. (2022). Shopping Queries Dataset: A Large-Scale ESCI Benchmark for Improving Product Search. *arXiv preprint arXiv:2206.06588*. <https://doi.org/10.48550/arXiv.2206.06588>
- Reimers, N., & Gurevych, I. (2019). Sentence-BERT: Sentence Embeddings Using Siamese BERT-Networks. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 3982-3992. <https://doi.org/10.18653/v1/D19-1410>
- Ren, Z., He, X., Yin, D., & de Rijke, M. (2024). Information Discovery in e-Commerce. *arXiv preprint arXiv:2410.05763*. <https://doi.org/10.48550/arXiv.2410.05763>
- Robertson, S., & Zaragoza, H. (2009). The Probabilistic Relevance Framework: BM25 and Beyond. *Foundations and Trends in Information Retrieval*, 3(4), 333-389. <https://doi.org/10.1561/15000000019>
- Sakai, T. (2020). On Fuhr's Guideline for IR Evaluation. *ACM SIGIR Forum*, 54(1). <https://doi.org/10.1145/3451964.3451976>
- Sambasivan, N., Kapania, S., Highfill, H., Akrong, D., Paritosh, P., & Aroyo, L. (2021). "Everyone Wants to Do the Model Work, Not the Data Work": Data Cascades in High-Stakes AI. *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems (CHI)*. <https://doi.org/10.1145/3411764.3445518>
- Sanderson, M. (2010). Test Collection Based Evaluation of Information Retrieval Systems. *Foundations and Trends in Information Retrieval*, 4(4), 247-375. <https://doi.org/10.1561/15000000009>
- Santhanam, K., Khattab, O., Saad-Falcon, J., Potts, C., & Zaharia, M. (2022). ColBERTv2: Effective and Efficient Retrieval via Lightweight Late Interaction. *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*. <https://doi.org/10.48550/arXiv.2112.01488>
- Schechter Vera, H., Dua, S., Zhang, B., Salz, D., Mullins, R., et al. (2025). EmbeddingGemma: Powerful and Lightweight Text Representations. *arXiv preprint arXiv:2509.20354*. <https://doi.org/10.48550/arXiv.2509.20354>
- Sentence-Transformers. (2026). Sentence-Transformers Documentation. <https://www.sbert.net/>
- Shadish, W. R., Cook, T. D., & Campbell, D. T. (2002). *Experimental and Quasi-Experimental Designs for Generalized Causal Inference*. Houghton Mifflin.
- Sheth, R., Sinha, S. R., Patil, M., Beniwal, H., & Singh, M. (2025). Beyond Monolingual Assumptions: A Survey of Code-Switched NLP in the Era of Large Language Models. *arXiv preprint arXiv:2510.07037*. <https://doi.org/10.48550/arXiv.2510.07037>

- Shumailov, I., Shumaylov, Z., Zhao, Y., Papernot, N., Anderson, R., & Gal, Y. (2024). AI Models Collapse When Trained on Recursively Generated Data. *Nature*, 631, 755-759. <https://doi.org/10.1038/s41586-024-07566-y>
- Subramanya, S. J., Devvrit, Kadekodi, R., Simhadri, H. V., & Krishnaswamy, R. (2019). DiskANN: Fast Accurate Billion-point Nearest Neighbor Search on a Single Node. *Advances in Neural Information Processing Systems (NeurIPS)*. https://proceedings.neurips.cc/paper_files/paper/2019/hash/09853c7fb1d3f8ee67a61b6bf4a7f8e6-Abstract.html
- Taylor, J. R. (1997). *An Introduction to Error Analysis: The Study of Uncertainties in Physical Measurements* (2.^a ed.). University Science Books.
- Thakur, N., Reimers, N., Rücklé, A., Srivastava, A., & Gurevych, I. (2021). BEIR: A Heterogeneous Benchmark for Zero-shot Evaluation of Information Retrieval Models. *Advances in Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks Track*. <https://doi.org/10.48550/arXiv.2104.08663>
- van der Maaten, L., & Hinton, G. (2008). Visualizing Data using t-SNE. *Journal of Machine Learning Research*, 9, 2579-2605. <https://jmlr.org/papers/v9/vandermaaten08a.html>
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention Is All You Need. *Advances in Neural Information Processing Systems (NeurIPS)*, 30. <https://doi.org/10.48550/arXiv.1706.03762>
- Voorhees, E. M. (2000). Variations in Relevance Judgments and the Measurement of Retrieval Effectiveness. *Information Processing and Management*, 36(5). [https://doi.org/10.1016/S0306-4573\(00\)00010-8](https://doi.org/10.1016/S0306-4573(00)00010-8)
- Wang, K., Thakur, N., Reimers, N., & Gurevych, I. (2022). GPL: Generative Pseudo Labeling for Un-supervised Domain Adaptation of Dense Retrieval. *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*. <https://doi.org/10.48550/arXiv.2112.07577>
- Wang, L., Yang, N., Huang, X., Jiao, B., Yang, L., Jiang, D., Majumder, R., & Wei, F. (2022). Text Embeddings by Weakly-Supervised Contrastive Pre-training. *arXiv preprint arXiv:2212.03533*. <https://doi.org/10.48550/arXiv.2212.03533>
- Wang, L., Yang, N., Huang, X., Yang, L., Majumder, R., & Wei, F. (2024). Multilingual E5 Text Embeddings: A Technical Report. *arXiv preprint arXiv:2402.05672*. <https://doi.org/10.48550/arXiv.2402.05672>
- Wang, R. Y., & Strong, D. M. (1996). Beyond Accuracy: What Data Quality Means to Data Consumers. *Journal of Management Information Systems*, 12(4), 5-33. <https://doi.org/10.1080/07421222.1996.11518099>
- Wang, W., Wei, F., Dong, L., Bao, H., Yang, N., & Zhou, M. (2020). MiniLM: Deep Self-Attention Distillation for Task-Agnostic Compression of Pre-Trained Transformers. *Advances in Neural Information Processing Systems (NeurIPS)*, 33, 5776-5788. <https://doi.org/10.48550/arXiv.2002.10957>
- Wilcoxon, F. (1945). Individual Comparisons by Ranking Methods. *Biometrics Bulletin*, 1(6), 80-83. <https://doi.org/10.2307/3001968>
- Xiong, L., Xiong, C., Li, Y., Tang, K.-F., Liu, J., Bennett, P., Ahmed, J., & Overwijk, A. (2020). Approximate Nearest Neighbor Negative Contrastive Learning for Dense Text Retrieval. *arXiv preprint arXiv:2007.00808*. <https://doi.org/10.48550/arXiv.2007.00808>
- Zobel, J. (1998). How Reliable Are the Results of Large-Scale Information Retrieval Experiments? *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. <https://doi.org/10.1145/290941.291014>