

Design and Implementation of a PostgreSQL-Milvus Data Pipeline with a PostgreSQL-Based Korean Hybrid Retrieval Module: An Evaluation of Retrieval Performance

Min-Ho Park*, Hyunchul Ahn**

*PhD Candidate, Graduate School of Business IT, Kookmin University, Seoul, Korea

**Professor, Graduate School of Business IT, Kookmin University, Seoul, Korea

[Abstract]

This paper presents the design and implementation of a PostgreSQL-Milvus data pipeline for Korean semantic search applications, including ontology-oriented services. PostgreSQL 18 is used as the source of truth for managing structured data and dense embeddings stored through the pgvector extension. INSERT, UPDATE, and DELETE events are captured by Debezium and delivered through Kafka to a dedicated Milvus consumer, allowing changes in PostgreSQL to be propagated to Milvus. For Korean information retrieval, the pipeline incorporates the BGE-M3 multilingual embedding model, which generates 1,024-dimensional dense vectors, and the textsearch_ko extension based on the mecab-ko morphological analyzer. A PostgreSQL-based hybrid retrieval function combines dense and lexical rankings using weighted reciprocal rank fusion within a single SQL statement. As a component-level evaluation, the retrieval effectiveness of the PostgreSQL-based hybrid search module was examined using an evaluation subset constructed from MIRACL-ko, consisting of 1,500 passages and 100 queries. Dense vector retrieval achieved Recall@10 of 0.9466, Recall@100 of 1.0000, MRR of 0.7898, and nDCG@10 of 0.8077. The weighted-RRF hybrid retrieval maintained comparable recall while improving MRR to 0.8219 and nDCG@10 to 0.8284. These results indicate that the proposed hybrid retrieval module improves ranking quality over dense retrieval alone under the given experimental setting. The end-to-end performance of the full PostgreSQL-Milvus pipeline, including propagation latency, failure recovery, cross-store consistency, and scalability, remains to be evaluated in future work.

► **Key words:** PostgreSQL, Milvus, Data Pipeline, Change Data Capture, Hybrid Retrieval, Korean Information Retrieval

-
- First Author: Min-Ho Park, Corresponding Author: Hyunchul Ahn
 - *Min-Ho Park (javatime@naver.com), Graduate School of Business IT, Kookmin University
 - **Hyunchul Ahn (hcahn@kookmin.ac.kr), Graduate School of Business IT, Kookmin University
 - Received: 2026. 07. 29, Revised: 2026. 08. 18, Accepted: 2026. 08. 20.

[요 약]

본 논문에서는 온톨로지 지향 서비스를 포함한 한국어 시맨틱 검색 응용을 지원하기 위한 PostgreSQL-Milvus 데이터 파이프라인을 설계하고 구현하였다. PostgreSQL 18을 정형 데이터와 pgvector 기반 밀집 임베딩을 관리하는 원본 데이터 저장소로 사용하고, INSERT, UPDATE 및 DELETE 이벤트를 Debezium과 Kafka를 통해 전용 Milvus 컨슈머에 전달함으로써 PostgreSQL의 변경분이 Milvus에 반영되도록 구성하였다. 한국어 정보 검색을 위해 BGE-M3 다국어 임베딩 모델의 1,024차원 밀집 벡터와 mecab-ko 형태소 분석기 기반의 textsearch_ko 전문검색을 적용하였다. 또한 밀집 벡터 검색과 어휘 검색의 순위를 가중 상호 순위 융합 방식으로 결합하는 PostgreSQL 기반 하이브리드 검색 함수를 단일 SQL 문장으로 구현하였다. 구성요소 수준의 성능평가로서, MIRACL-ko로부터 구성한 1,500개 문단과 100개 질의의 평가 데이터셋을 이용하여 PostgreSQL 기반 하이브리드 검색 모듈의 검색 효과를 분석하였다. 밀집 벡터 검색은 Recall@10 0.9466, Recall@100 1.0000, MRR 0.7898 및 nDCG@10 0.8077을 기록하였다. 가중 상호 순위 융합을 적용한 하이브리드 검색은 유사한 재현율을 유지하면서 MRR을 0.8219, nDCG@10을 0.8284로 향상시켰다. 이러한 결과는 본 실험 환경에서 제안한 하이브리드 검색 모듈이 밀집 벡터 검색만을 사용한 경우보다 상위 결과의 순위 품질을 개선할 수 있음을 보여준다. 다만 변경 전파 지연, 장애 복구, 저장소 간 정합성 및 확장성 등 전체 PostgreSQL-Milvus 파이프라인의 중단 간 성능평가는 향후 연구에서 수행할 필요가 있다.

▶ **주제어:** PostgreSQL, Milvus, 데이터 파이프라인, 변경 데이터 캡처, 하이브리드 검색, 한국어 정보 검색

I. Introduction

최근 대규모 언어모델과 텍스트 임베딩 기술의 발전으로 정보 검색은 키워드 일치 중심의 방식에서 문서와 질의의 의미적 관계를 반영하는 벡터 기반 검색으로 확장되고 있다[1][2][3]. 이러한 검색 방식은 검색 증강 생성, 지식관리, 질의응답 및 온톨로지 구축·검증과 같은 다양한 응용에 활용될 수 있다[2]. 그러나 실제 의미 검색 서비스를 구축하기 위해서는 임베딩 모델의 검색 성능뿐만 아니라 원본 데이터와 벡터의 관리, 데이터 변경의 검색 시스템 반영, 정형 질의와 비정형 검색의 결합 등을 함께 고려해야 한다. 또한 한국어는 조사와 어미 변화, 복합명사 및 고유명사와 같은 언어적 특성을 가지므로, 의미 기반 벡터 검색과 형태소 기반 어휘 검색을 상호 보완적으로 활용할 필요가 있다[3][4].

관계형 데이터베이스는 트랜잭션 처리와 정형 데이터 관리에 강점을 가지며, pgvector와 같은 확장을 통해 벡터 검색도 지원할 수 있다[5]. 반면 Milvus와 같은 전용 벡터 데이터베이스는 대규모 벡터 저장과 검색 계층의 독립적인 확장에 적합하다[6]. 관계형 데이터베이스와 벡터 데이터베이스를 함께 활용하면 각 시스템의 장점을 결합할 수 있지만, 원본 데이터의 변경을 벡터 저장소에 전달하기 위한 동기화 파이프라인과 서로 다른 검색 결과를 통합하기 위한 구현이 필요하다[7][8][9]. 검색 측면에서도

벡터 검색과 어휘 검색은 상호 보완적일 수 있으나, 하이브리드 검색의 효과는 임베딩 모델, 질의 유형 및 결과 융합 방식에 따라 달라질 수 있다[10][11][12][13]. 따라서 데이터 관리와 벡터 검색을 연계하는 시스템 구조와 한국어 하이브리드 검색을 함께 고려한 구현 연구가 필요하다.

본 연구에서는 PostgreSQL을 원본 기준 저장소로 사용하고, Debezium과 Kafka를 통해 변경 데이터를 Milvus에 전달하는 PostgreSQL-Milvus 데이터 파이프라인을 설계하고 구현하였다[5][6][7]. 또한 BGE-M3 기반 밀집 벡터 검색과 mecab-ko 기반 어휘 검색을 가중 RRF로 결합하고, 이를 단일 SQL 문장으로 호출할 수 있는 PostgreSQL 기반 한국어 하이브리드 검색 기능으로 구현하였다[1][4][10]. 본 연구의 실증분석은 전체 데이터 파이프라인의 중단 간 성능이 아니라, 파이프라인에 포함된 한국어 하이브리드 검색 모듈의 검색 효과를 평가하는 데 초점을 둔다. 이를 위해 MIRACL-ko로부터 구성한 100개 질의와 1,500개 문단의 평가 데이터셋을 이용하여 벡터 검색, 어휘 검색 및 하이브리드 검색의 성능을 비교하였다[14]. 분석 결과, 하이브리드 검색은 벡터 검색과 유사한 재현율을 유지하면서 MRR과 nDCG를 각각 4.07%와 2.56% 향상시켰다.

본 연구의 주요 기여는 다음과 같다. 첫째,

PostgreSQL, Debezium, Kafka 및 Milvus를 연계한 이기종 데이터 저장소 파이프라인의 설계와 구현을 제시한다. 둘째, BGE-M3 기반 의미 검색과 mecab-ko 기반 어휘 검색을 가중 RRF로 결합한 SQL 기반 한국어 하이브리드 검색 기능을 구현한다. 셋째, MIRACL-ko 기반 평가 데이터셋을 이용하여 제안한 하이브리드 검색 모듈이 벡터 검색만을 적용한 경우보다 상위 검색 결과의 순위 품질을 향상시킬 수 있음을 확인한다. 전체 PostgreSQL-Milvus 파이프라인의 변경 전파 성능, 장애 복구, 저장소 간 정합성 및 확장성에 대한 종단 간 평가는 향후 연구에서 수행하고자 한다.

II. Related Works

1. Relational and Vector Database Architectures

최근 임베딩 기반 정보 검색의 활용이 확대되면서 관계형 데이터와 고차원 벡터를 어떻게 저장하고 검색할 것이냐가 중요한 시스템 설계 문제로 부각되고 있다. 관계형 데이터베이스는 트랜잭션 처리, 데이터 무결성 및 SQL 기반 정형 질의에 강점을 가지며, pgvector는 PostgreSQL에 벡터 자료형과 유사도 검색 기능을 추가함으로써 관계형 데이터와 임베딩 벡터를 하나의 데이터베이스에서 관리할 수 있도록 지원한다[5]. 이러한 통합형 구조는 기존 SQL 응용과의 연계가 용이하고 별도의 벡터 저장소를 운영하지 않아도 된다는 장점이 있다.

반면 Milvus는 대규모 벡터 데이터의 저장과 근사 최근접 이웃 검색을 목적으로 설계된 전용 벡터 데이터베이스이다[6]. Milvus는 HNSW, IVF 계열 등 다양한 벡터 인덱스를 지원하며, 데이터 규모가 증가하는 환경에서 검색 계층을 별도로 확장할 수 있다는 장점을 가진다. 그러나 원본 데이터가 관계형 데이터베이스에 저장되고 벡터가 별도의 데이터베이스에서 관리되는 경우에는 두 저장소 간 데이터 동기화와 운영 관리가 추가로 필요하다.

최근에는 관계형 데이터 처리와 벡터 검색을 결합하기 위한 다양한 아키텍처가 제안되고 있다. SingleStore-V는 관계형 데이터베이스 엔진에 벡터 저장과 검색 기능을 통합하여 SQL 기반의 정형 질의와 벡터 검색을 동일한 시스템에서 수행하는 구조를 제시하였다[8]. 반면 Liu et al.[9]은 PostgreSQL의 트랜잭션 처리와 벡터 검색 실행을 분리하여 각각을 독립적으로 최적화하는 분리형 접근을 제안하였다. 이러한 연구들은 통합형과 분리형 아키텍처가 각각 운영 단순성, 검색 확장성 및 자원 분리 측면에서 서

로 다른 장점을 가질 수 있음을 보여준다.

본 연구는 PostgreSQL을 원본 기준 저장소로 유지하면서 Milvus를 벡터 검색 확장 계층으로 사용하는 이기종 저장소 구조를 채택한다. 이는 관계형 데이터베이스의 트랜잭션 관리 기능과 전용 벡터 데이터베이스의 검색 확장성을 함께 활용하기 위한 접근이다. 또한 PostgreSQL의 pgvector에도 동일한 임베딩을 저장함으로써 PostgreSQL 내부 검색과 Milvus 기반 검색을 모두 지원할 수 있도록 구성한다.

2. CDC-Based Data Synchronization

관계형 데이터베이스와 벡터 데이터베이스를 분리하여 운영하는 경우, 원본 데이터의 변경 내용을 벡터 저장소에 지속적으로 반영하는 데이터 동기화 체계가 필요하다. 애플리케이션이 두 저장소에 각각 데이터를 기록하는 이중 쓰기 방식은 구현이 단순할 수 있으나, 일부 쓰기 작업의 실패나 네트워크 장애가 발생하면 저장소 간 불일치가 발생할 가능성이 있다. 따라서 원본 데이터베이스의 변경 기록을 기반으로 다른 시스템에 변경 이벤트를 전달하는 변경 데이터 캡처(Change Data Capture, CDC) 방식이 활용될 수 있다.

Debezium은 PostgreSQL을 포함한 여러 데이터베이스의 트랜잭션 로그를 읽어 INSERT, UPDATE 및 DELETE 이벤트를 행 단위로 포착하는 오픈소스 CDC 플랫폼이다[7]. PostgreSQL에서는 논리적 복제 기능을 이용하여 WAL(Write-Ahead Log)에 기록된 변경 내용을 디코딩하고, Debezium 커넥터가 이를 이벤트 스트림으로 변환한다. 생성된 변경 이벤트는 Kafka와 같은 메시지 시스템을 통해 후속 소비자에게 전달될 수 있다.

CDC 기반 구조에서는 애플리케이션이 원본 데이터베이스에만 쓰기 작업을 수행하고, 검색용 저장소에 대한 반영은 별도의 이벤트 처리 계층이 담당한다. 이를 통해 원본 데이터의 트랜잭션 처리와 벡터 저장소의 갱신 작업을 분리할 수 있으며, 검색 계층을 독립적으로 운영하고 확장하기도 용이하다. 다만 CDC 기반의 비동기 동기화에서는 변경 이벤트의 전달 지연, 중복 처리, 이벤트 순서, 소비자 장애 이후의 재처리 및 저장소 간 일관성 검증 등을 함께 고려해야 한다.

본 연구에서는 PostgreSQL을 원본 기준 저장소로 두고, Debezium이 포착한 변경 이벤트를 Kafka 토픽으로 전달한 후 전용 Milvus 컨슈머가 이를 구독하도록 구성한다. 이 구조는 PostgreSQL에 발생한 데이터 변경을 애플리케이션의 추가적인 이중 쓰기 로직 없이 Milvus에 전달

하기 위한 참조 파이프라인을 제시한다. 다만 변경 전파 지연, 장애 복구 및 데이터 규모에 따른 처리량 등 전체 CDC 파이프라인의 종단 간 성능평가는 본 연구의 실증분석 범위에 포함하지 않는다.

3. Korean Hybrid Information Retrieval

임베딩 기반 벡터 검색은 질의와 문서 사이의 의미적 유사성을 반영할 수 있어, 표현이 직접적으로 일치하지 않는 경우에도 관련 문서를 검색할 수 있다. BGE-M3는 100개 이상의 언어를 지원하며, 하나의 모델에서 밀집 벡터, 희소 벡터 및 ColBERT 기반 다중 벡터 표현을 생성할 수 있는 다국어 임베딩 모델이다[1]. 한국어를 포함한 다국어 검색에 적용할 수 있다는 점에서 RAG 및 의미 검색 시스템의 임베딩 모델로 활용될 수 있다.

국내에서도 생성형 AI를 위한 RAG 기술과 관련 응용에 관한 연구가 진행되고 있으며[2], 한국어 단어 임베딩의 품질과 학습 조건을 개선하려는 연구도 수행되어 왔다[3]. 그러나 한국어는 조사와 어미 변화, 복합명사, 동형이의어 및 고유명사와 같은 특성을 가지므로, 임베딩 기반 검색만으로 모든 유형의 질의를 처리하기는 어렵다. 정확한 개체명이나 특정 전문용어가 포함된 질의에서는 문자열 및 형태소 일치에 기반한 어휘 검색이 보완적인 역할을 할 수 있다. 한국어 형태소 분석 도구인 mecab-ko는 문장을 형태소 단위로 분석하여 한국어 전문검색에 필요한 토큰화와 형태소 분해에 활용될 수 있다[4].

어휘 검색과 의미 검색의 상호 보완성은 기존 정보 검색 연구에서도 다루어져 왔다. Gao et al.[13]은 어휘 검색 모델이 포착하기 어려운 의미적 정보를 잔차 임베딩으로 보완하는 방법을 제안하여, 어휘 신호와 의미 신호를 결합하는 접근의 가능성을 보여주었다. 서로 다른 검색기의 결과를 결합하는 대표적인 방법인 RRF(Reciprocal Rank Fusion)는 각 검색기의 원점수 대신 문서의 순위를 이용하여 복수의 검색 결과를 융합한다[10]. 이에 따라 코사인 거리와 어휘 일치 점수처럼 척도가 서로 다른 검색 결과도 별도의 복잡한 점수 정규화 없이 결합할 수 있다.

다만 하이브리드 검색이 항상 단독 검색보다 우수한 것은 아니다. 검색 결과의 융합 성능은 사용된 검색기, 융합 함수, 후보 문서 수 및 데이터 특성에 따라 달라질 수 있으며, 다양한 융합 함수 사이에도 성능 차이가 존재한다[11]. 최근의 실험 연구 역시 강한 임베딩 모델이 사용되는 환경에서는 어휘 검색의 추가적 기여가 질의 유형과 융합 조건에 따라 달라질 수 있음을 보여준다[12]. 따라서 한국어 환경에서 BGE-M3 기반 밀집 벡터 검색과 형태소 기반 어휘

검색을 결합했을 때 검색 결과의 순위 품질이 실제로 개선되는지를 확인할 필요가 있다.

본 연구에서는 BGE-M3가 생성한 1,024차원 밀집 벡터와 mecab-ko 기반의 textsearch_ko 전문검색을 결합한다. 두 검색 결과는 가중 RRF를 통해 융합하며, 하이브리드 검색 과정 전체를 PostgreSQL 내부의 단일 SQL 인터페이스로 구현한다. 검색 성능평가에는 18개 언어를 포함하는 다국어 정보 검색 데이터셋인 MIRACL의 한국어 부분 집합을 활용한다[14].

4. Research Positioning

선행연구를 종합하면, 기존 연구는 크게 세 가지 흐름으로 구분할 수 있다. 첫째, Milvus와 pgvector를 비롯한 벡터 저장 및 검색 시스템에 관한 연구가 수행되어 왔으며[5][6], 관계형 데이터베이스와 벡터 검색을 통합하거나 분리하여 최적화하는 아키텍처도 제안되었다[8][9]. 둘째, Debezium과 같은 CDC 플랫폼을 활용하면 관계형 데이터베이스의 변경 이벤트를 다른 저장소에 전달할 수 있다[7]. 셋째, 벡터 검색과 어휘 검색을 결합하는 하이브리드 검색 및 순위 융합 방법의 효과와 조건이 연구되어 왔다[10][11][12][13].

그러나 이러한 구성요소들은 대체로 개별 기술이나 알고리즘의 관점에서 다루어져 왔다. PostgreSQL을 원본 기준 저장소로 유지하면서 CDC를 통해 Milvus와 연계하고, BGE-M3 기반 의미 검색과 한국어 형태소 기반 어휘 검색을 단일 SQL 인터페이스로 통합하는 전체 구현 구조는 상대적으로 충분히 제시되지 않았다.

이에 본 연구는 두 가지 측면에서 기존 연구를 확장한다. 첫째, PostgreSQL, Debezium, Kafka 및 Milvus를 연결하여 관계형 데이터 관리와 벡터 검색 계층을 분리한 데이터 파이프라인의 설계와 구현을 제시한다. 둘째, BGE-M3 기반 밀집 벡터 검색과 mecab-ko 기반 어휘 검색을 가중 RRF로 결합하는 PostgreSQL 기반 한국어 하이브리드 검색 모듈을 구현하고, MIRACL-ko로부터 구성된 평가 데이터셋을 이용해 해당 모듈의 검색 효과를 분석한다. 다만 본 연구의 정량적 평가는 하이브리드 검색 모듈에 한정되며, 전체 PostgreSQL-Milvus 파이프라인의 전파 지연, 장애 복구, 저장소 간 정합성 및 확장성에 대한 평가는 향후 연구 범위로 남긴다.

III. The Proposed System

1. Overall Architecture

제안 파이프라인은 PostgreSQL을 원본 기준 저장소로 두고 Milvus를 벡터 검색 확장 계층으로 사용하는 이기종 이중 저장소 구조를 채택한다. 8개 컨테이너(PostgreSQL, etcd, MinIO, Milvus, Kafka, Debezium, milvus-init, milvus-consumer)가 Docker Compose를 통해 통합 관리되며, 사설 vectordb 네트워크(172.31.0.0/24) 위에서 동작한다. 각 컨테이너는 서비스별 헬스체크(pg_isready, etcdctl, MinIO health, Milvus healthz, kafka-broker-api-versions, Kafka Connect REST)를 갖추고 있으며, 시작 순서와 의존성이 명시적으로 관리된다.

데이터의 저장 흐름은 다음과 같다. 애플리케이션이 PostgreSQL에 문서를 삽입하면 pgvector 컬럼에 저장된 임베딩과 함께 WAL(Write-Ahead Log)에 기록된다. WAL은 PostgreSQL이 트랜잭션의 변경분을 먼저 순차 파일에 남긴 뒤 데이터 페이지에 반영하는 로깅 방식으로, 장애 복구와 복제의 기본 매체이다. 특히 wal_level=logical로 설정하면 physical 로그(바이트 단위)에 더해 논리 복제 슬롯을 통해 INSERT·UPDATE·DELETE 이벤트를 행 단위로 디코딩할 수 있게 되고, Debezium 커넥터가 이 슬롯을 구독해 변경분을 Kafka 토픽으로 발행한다. Milvus 컨슈머는 이 토픽을 구독해 CDC 이벤트를 Milvus 컬렉션에 반영한다. 이 구조는 원본 데이터의 트랜잭션 관리와 벡터 검색을 서로 다른 계층으로 분리함으로써, 두 계층을 독립적으로 운영하고 확장할 수 있도록 설계되었다.

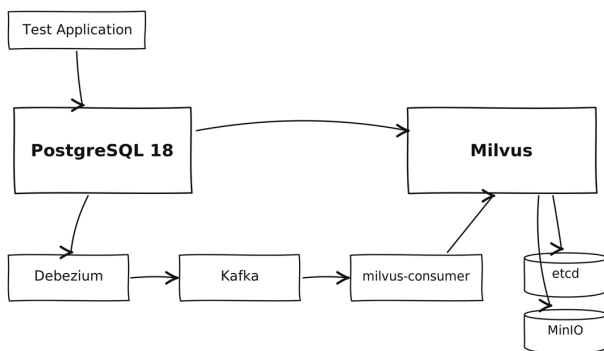


Fig. 1. Overall architecture of the proposed PostgreSQL-Milvus data pipeline

2. Embedding Generation and Korean Full-Text Search

임베딩 모델로는 BAAI가 공개한 BGE-M3를 사용하였

대[1]. BGE-M3는 100개 이상의 언어를 지원하며 밀집 벡터, 희소 벡터 및 ColBERT 기반 다중 벡터 표현을 생성할 수 있다. 본 연구에서는 이 가운데 1,024차원의 밀집 벡터 표현을 사용하였다. 임베딩은 sentence-transformers 라이브러리를 통해 생성되며, 생성된 벡터는 PostgreSQL의 pgvector 컬럼에 원문 및 메타데이터와 함께 저장된다 [5]. 저장된 벡터는 앞 절에서 설명한 CDC 파이프라인을 통해 Milvus에도 전달된다.

한국어 어휘 검색에는 mecab-ko 형태소 분석기와 세종 프로젝트 기반 사전을 사용하는 textsearch_ko 확장을 적용하였다[4]. 형태소 분석 결과로 생성되는 tsvector 컬럼은 GENERATED ALWAYS 방식으로 정의하여 원문이 변경될 때 자동으로 갱신되도록 하였으며, GIN 인덱스를 적용하여 형태소 기반 전문검색을 수행하도록 구성하였다.

3. Hybrid Search via pg_hybrid Extension

pg_hybrid 확장의 hybrid_search() 함수는 벡터 검색과 어휘 검색 및 결과 융합을 하나의 SQL 문장 안에서 수행한다. 먼저 PostgreSQL의 pgvector 컬럼에 대해 코사인 거리를 계산하여 상위 c 개의 벡터 검색 후보를 생성한다[5]. 다음으로 입력 질의를 형태소 단위로 분석하여 OR 방식의 tsquery를 구성하고, ts_rank를 이용하여 상위 c 개의 어휘 검색 후보를 생성한다. 마지막으로 두 순위 목록을 가중 RRF(Reciprocal Rank Fusion)로 결합하여 최종 상위 top- k 결과를 반환한다[10].

RRF는 서로 다른 척도로 산출되는 검색 점수를 직접 더하지 않고, 각 문서가 개별 검색 결과에서 차지한 순위를 이용하여 최종 점수를 계산한다. 본 연구에서 사용한 가중 RRF 점수($score(d)$)는 다음의 수식 (1)과 같다.

$$score(d) = \frac{w_v}{k + r_d^{vec}} + \frac{w_{kw}}{k + r_d^{kw}} \quad (1)$$

여기서 r_d^{vec} 과 r_d^{kw} 는 각각 벡터 검색과 어휘 검색에서 문서 d 의 순위이고, w_v 와 w_{kw} 는 두 검색기의 가중치이며, k 는 상위 순위 간 점수 차이를 완화하기 위한 평활 상수이다. 본 연구에서는 $k = 60$ 을 사용하였다. 하이브리드 검색 함수는 두 검색기의 가중치를 매개변수로 받도록 구현하였다. 본 연구의 실험은 두 가중치를 모두 1.0으로 둔 동일 가중치 구성에서 수행하였으므로, 가중 RRF는 가중치를 매개변수화한 순위 융합 방식을 가리킨다.

본 시스템은 벡터 검색을 수행하는 위치에 따라 두 가지 하이브리드 검색 경로를 지원한다. 첫 번째는 PostgreSQL의 pgvector와 textsearch_ko를 이용해 벡터 검색과 어휘 검색을 모두 PostgreSQL 내부에서 수행

하고, pg_hybrid 함수가 두 결과를 SQL 내부에서 융합하는 방식이다. 이 방식에서는 애플리케이션이 하나의 PostgreSQL 연결과 단일 SQL 호출을 통해 하이브리드 검색 결과를 얻을 수 있다.

두 번째는 pg_milvus 확장을 이용하여 벡터 검색을 Milvus에서 수행하고, PostgreSQL의 textsearch_ko에서 산출한 어휘 검색 결과와 결합하는 방식이다. 이때 pg_milvus는 Milvus 검색을 PL/Python 함수로 감싸 PostgreSQL의 SQL 인터페이스에서 Milvus 벡터 검색을 호출할 수 있도록 한다. 이 방식은 Milvus를 벡터 검색 확장 계층으로 활용할 수 있지만, PostgreSQL과 Milvus 간 데이터 동기화가 전제되어야 한다. 다음의 Table 1은 두 구현 경로의 주요 특성을 비교한다.

Table 1에서 보듯이 PostgreSQL 내부 방식은 벡터 검색과 어휘 검색을 동일한 데이터베이스 안에서 수행하므로, 하이브리드 검색을 위해 별도의 저장소 간 동기화가 필요하지 않다. 반면 PostgreSQL-Milvus 방식은 Milvus를 별도의 벡터 검색 계층으로 활용할 수 있으나, 검색 결과의 일관성을 유지하기 위해 PostgreSQL의 변경 내용을 CDC 파이프라인을 통해 Milvus에 전달해야 한다. 두 방식 모두 애플리케이션에는 PostgreSQL 기반의 단일 SQL 인터페이스를 제공하도록 설계되었지만, 내부적인 벡터 검색 위치와 동기화 요구사항에서 차이가 있다.

Table 1. Comparison of hybrid retrieval implementation patterns

Characteristic	PostgreSQL-local hybrid retrieval	PostgreSQL-Milvus hybrid retrieval
Vector retrieval engine	pgvector	Milvus
Lexical retrieval engine	textsearch_ko	
Result fusion	pg_hybrid in PostgreSQL	PostgreSQL SQL via pg_milvus
Application interface	Single PostgreSQL interface	
Vector data synchronization	Not required for retrieval	Required through the CDC pipeline
Independent scaling of vector search	Limited to PostgreSQL deployment	Supported through Milvus deployment
Quantitatively evaluated in this study	Yes	No

본 연구의 정량적 검색 성능평가는 이 가운데 PostgreSQL의 pgvector 기반 벡터 검색과 textsearch_ko 기반 어휘 검색을 결합한 PostgreSQL 내부 하이브리드 검색 경로를 대상으로 수행하였다.

PostgreSQL-Milvus 검색 경로는 제안 시스템이 지원하는 확장 구조로 구현하였으나, 해당 경로의 검색 지연, 처리량 및 종단 간 성능에 대한 평가는 본 연구의 범위에 포함하지 않았다.

IV. Experimental Evaluation of the Hybrid Retrieval Module

1. Experimental Dataset and Environment

본 장에서는 제안한 전체 PostgreSQL-Milvus 데이터 파이프라인의 종단 간 성능이 아니라, 파이프라인에 포함된 PostgreSQL 기반 한국어 하이브리드 검색 모듈의 검색 효과를 평가한다. 평가에는 MIRACL(Multilingual Information Retrieval Across a Continuum of Languages)의 한국어 부분 집합인 MIRACL-ko를 활용하였다[14]. MIRACL은 18개 언어의 위키피디아 문단을 검색 코퍼스로 사용하며, 각 언어의 원어민이 작성한 자연어 질의와 사람이 검토한 관련도 판단 정보를 제공하는 다국어 정보 검색 데이터셋이다.

MIRACL-ko는 한국어 위키피디아에서 추출한 약 150만 개의 문단과 213개의 개발용 질의 및 868개의 학습용 질의로 구성된다[14]. 본 연구에서는 MIRACL-ko의 개발용 질의 중 100개를 무작위로 선택하고, 해당 질의의 관련 문단과 무작위로 추가한 비관련 후보 문단을 결합하여 총 1,500개 문단으로 이루어진 평가 데이터셋을 구성하였다. 질의별 정답 집합은 MIRACL-ko가 제공하는 관련도 판단 정보를 사용하였으며, 각 질의는 1개에서 9개의 관련 문단을 포함한다. 선택한 100개 질의의 관련 문단은 모두 평가 코퍼스에 포함하였다. 나머지는 이들 질의의 관련 문단을 제외한 MIRACL-ko 문단에서 무작위로 추출해 전체 문단 수를 1,500개로 맞추었다. 따라서 본 연구의 실험 결과는 MIRACL-ko 전체 검색 코퍼스에 대한 평가가 아니라, MIRACL-ko로부터 구성한 축소 평가 환경에서의 결과로 해석해야 한다.

실험은 Windows 11 환경에서 WSL2와 Docker Desktop을 사용하여 수행하였다. PostgreSQL, etcd, MinIO, Milvus, Kafka, Debezium, milvus-init 및 milvus-consumer의 8개 컨테이너가 vectordb 사설 네트워크에서 실행되었다. 컨테이너의 기본 이미지는 Ubuntu 22.04.5 LTS이며, 호스트 및 가상화 환경의 세부 구성은 Table 2에 제시하였다.

Table 2. Experimental environment

Layer	Component	Specification
Host	OS	Windows 11 Home (10.0.26200, Build 26200)
Host	CPU	13th Gen Intel Core i7-1360P (12C / 16T)
Host	Memory	31.69 GB (32 GB installed)
Virt.	WSL	WSL 2.7.10.0 · kernel 6.18.33.2-2 · WSLg 1.0.73.2
Virt.	.wslconfig	memory=24GB, processors=12, swap=16GB (see below)
Container	Docker Desktop	4.57.0 (215387) · Engine 29.1.3 · API 1.52
Container	Runtime	containerd v2.2.1 · runc 1.3.4
Container	Base image	Ubuntu 22.04.5 LTS (PostgreSQL container)
Container	Network	vectordb bridge, 172.31.0.0/24, 8 services

본 실험에서는 벡터 검색과 키워드 검색에서 산출된 두 순위에 각각 1.0의 동일한 가중치를 부여하여 융합하였으며, 순위 융합에는 RRF(Reciprocal Rank Fusion) 방식의 융합 상수인 rrf_k 만을 사용하였다. 임베딩 모델로는 sentence-transformers 5.6.0 환경에서 제공되는 BAAI/bge-m3를 사용하였으며, 임베딩 벡터의 차원은 1,024로 설정하였다. 또한 `normalize_embeddings = True`를 적용하여 임베딩 벡터를 정규화하였다. 임베딩 생성 시 배치 크기인 `EMB_BATCH`는 64로, 데이터베이스 처리 배치 크기인 `DB_BATCH`는 500으로 설정하였다.

벡터 인덱스는 PostgreSQL과 Milvus에 각각 별도로 구성하였다. 정량 평가 대상인 PostgreSQL의 pgvector 컬럼에는 $M=16$, $efConstruction=256$ 의 HNSW 인덱스를 생성하였고 거리 측정에는 코사인 거리를 사용하였다. CDC 파이프라인으로 동기화되는 Milvus 컬렉션에도 같은 파라미터의 HNSW 인덱스를 `metric_type = COSINE`으로 구축하였다. 벡터 검색 시 탐색 파라미터는 $ef = 100$ 으로 두었다. Milvus 기반 검색 경로는 본 연구의 정량 평가 범위에 포함하지 않았다. 키워드 검색을 위해서는 문서의 content 필드를 한국어 텍스트 검색 설정에 따라 변환한 content_tsvector를 저장형 컬럼으로 구성하고, GIN 인덱스를 구축하였다. 이 과정에는 mecab-ko와 textsearch_ko를 사용하였다. RRF 기반의 검색 결과 융합에서는 한국어 텍스트 검색 설정(`Nts_config = 'korean'`)을 적용하고, rrf_k 를 60으로 설정하였다. 벡터 검색과 키워드 검색에서 각각 최대 100개의 후보를 추출하였으며(`candidates = 100`, `NCAND = 100`), 최종적으로 상위 100개의 결과를 반환하도록 `top_n = 100`으로 설정하였다.

2. Retrieval Methods and Evaluation Metrics

본 연구에서는 다음 세 가지 검색 방법을 비교하였다.

첫째, Vector retrieval은 BGE-M3가 생성한 1,024차원 밀집 벡터와 PostgreSQL의 pgvector를 이용한다 [1][5]. 문서 벡터는 HNSW 인덱스로 구성하였으며, 인덱스 생성 시 $M=16$, $efConstruction=256$ 을 적용하였다. 질의와 문서 사이의 유사도는 코사인 거리를 이용하여 계산하였다.

둘째, Keyword retrieval은 mecab-ko 형태소 분석기에 기반한 textsearch_ko 전문검색을 이용한다[4]. 질의를 형태소 단위로 분석한 후 OR 방식의 tsquery를 구성하고, ts_rank를 이용하여 검색 결과의 순위를 산출하였다. 문서의 tsvector 컬럼에는 GIN 인덱스를 적용하였다.

셋째, Hybrid retrieval은 벡터 검색과 어휘 검색에서 생성된 두 순위 목록을 가중 RRF 방식으로 결합한다[10]. RRF 평할 상수는 $k=60$ 으로 설정하였으며, 최종 점수는 앞서 수식 (1)에 제시된 $score(d)$ 로 산출한다.

Table 3. Metrics for retrieval performance

Metric	Definition	Interpretation	Range
Recall@10	Relevant documents retrieved within top 10	Recall in top-ranked results	0-1
Recall@100	Relevant documents retrieved within top 100	Candidate-set coverage	0-1
MRR	Mean reciprocal rank of the first relevant document	Position of the first relevant result	0-1
nDCG@10	Normalized discounted cumulative gain	Overall ranking quality	0-1

검색 성능은 Table 3에 제시된 것과 같이 Recall@10, Recall@100, MRR 및 nDCG를 이용하여 평가하였다. Recall@k는 질의별 관련 문서 가운데 상위 k개의 검색 결과에 포함된 관련 문서의 비율을 측정한다. MRR(Mean Reciprocal Rank)은 첫 번째 관련 문서가 등장한 순위의 역수를 질의 전체에 대해 평균한 값으로, 관련 문서가 검색 결과 상위에 얼마나 빨리 나타나는지를 나타낸다. 끝으로 nDCG(normalized Discounted Cumulative Gain)는 관련 문서가 상위 순위에 배치될수록 더 높은 가중치를 부여하여 검색 결과의 전반적인 순위 품질을 평가한다.

3. Results and Discussion

Table 4는 MIRACL-ko로부터 구성한 1,500개 문단과 100개 질의의 평가 데이터셋에서 벡터 검색, 어휘 검색 및 가중 RRF 기반 하이브리드 검색의 성능을 비교한 결과다.

BGE-M3 기반 벡터 검색은 Recall@10 0.9466, Recall@100 1.0000, MRR 0.7898 및 nDCG@10 0.8077 을 기록하였다. 이는 본 연구에서 구성한 평가 데이터셋에서 밀집 벡터 검색만으로도 대부분의 관련 문서를 상위 검색 결과에 포함할 수 있음을 보여준다.

반면 mecab-ko 기반 어휘 검색은 Recall@10 0.4059, Recall@100 0.4070, MRR 0.3764 및 nDCG@10 0.3707 로 벡터 검색보다 낮은 성능을 보였다. 이러한 결과에는 자연어 질의와 관련 문단 사이의 어휘 불일치, 조사와 어미 변화, 복합명사 처리 및 형태소 분석 사전의 적용 범위가 영향을 미쳤을 가능성이 있다. 다만 본 연구에서는 질의 유형이나 형태소 분석 오류에 따른 세부 분석을 수행하지 않았으므로, 어휘 검색의 낮은 성능을 특정 원인만으로 단정하기는 어렵다.

가중 RRF 기반 하이브리드 검색은 Recall@10 0.9471, Recall@100 1.0000, MRR 0.8219 및 nDCG@10 0.8284 를 기록하였다. 벡터 검색과 비교하면 Recall@10은 0.0005 증가하였고 Recall@100은 동일하게 유지되었다. 반면 MRR은 0.0321, nDCG@10은 0.0207 증가하였다.

Table 4. Retrieval performance on the MIRACL-ko evaluation subset

Method	Recall@10	Recall@100	MRR	nDCG@10
Vector (BGE-M3)	0.9466	1.0000	0.7898	0.8077
Keyword (textsearch_ko)	0.4059	0.4070	0.3764	0.3707
Hybrid (weighted RRF)	0.9471	1.0000	0.8219	0.8284

Table 5는 하이브리드 검색이 벡터 단독 대비 얼마나 개선되었는지를 나타낸다. Recall@10과 Recall@100은 벡터가 이미 상한에 근접하여 개선폭이 미미하지만(각각 +0.0005, ± 0), 랭킹 품질을 측정하는 MRR과 nDCG@10은 각각 +0.0321(+4.07%), +0.0207(+2.56%) 향상되었다. 즉, BGE-M3처럼 강한 다국어 임베딩 환경에서도 형태소 기반 어휘 검색을 가중 RRF로 결합함으로써 상위 검색 결과의 순위 품질을 개선하는 결과를 보였다.

Table 5. Improvement of hybrid retrieval over vector-only retrieval

Metric	Vector	Hybrid	Δ (abs.)	Δ (%)
Recall@10	0.9466	0.9471	+0.0005	+0.05%
Recall@100	1.0000	1.0000	± 0.0000	$\pm 0.00\%$
MRR	0.7898	0.8219	+0.0321	+4.07%
nDCG@10	0.8077	0.8284	+0.0207	+2.56%

이 결과는 형태소 기반 어휘 검색의 독립적인 성능은 낮았지만, 일부 질의에서는 어휘 정보가 벡터 검색 결과의 순서를 보완하여 관련 문서를 더 높은 순위로 이동시켰을 가능성을 보여준다. 즉, 가중 RRF를 적용한 하이브리드 검색은 벡터 검색의 높은 재현율을 대체로 유지하면서 첫 번째 관련 문서의 위치와 전반적인 순위 품질을 개선하였다.

본 장의 결과는 100개 질의와 1,500개 문단으로 축소 구성한 평가 데이터셋에서 얻은 것이다. 후보 문단 수가 작으면 인덱스 탐색 부담이 낮고 상위 순위의 차별성도 두드러지기 쉬워, 여기서 확인한 개선폭은 대규모 검색 환경에서 줄어든 수 있다. 규모가 커질 때의 비용도 고려해야 한다. 색인과 질의 양쪽에서 형태소 분석 연산이 병목이 될 수 있고, 재현율을 유지하려고 후보 문단 수를 늘리면 두 검색기의 후보 목록을 결합하는 가중 RRF 계산 비용이 그만큼 증가한다. 향후 연구에서는 대규모 코퍼스에서의 검색 효과와 이러한 연산 오버헤드를 정량적으로 분석하고자 한다.

V. Conclusions

본 연구에서는 한국어 시맨틱 검색 응용을 지원하기 위한 PostgreSQL-Milvus 데이터 파이프라인을 설계하고 구현하였다. 제안 시스템은 PostgreSQL을 정형 데이터와 임베딩을 관리하는 원본 기준 저장소로 사용하고, PostgreSQL에서 발생하는 데이터 변경을 Debezium과 Kafka를 통해 Milvus에 전달하도록 구성하였다. 이를 통해 원본 데이터의 트랜잭션 관리 계층과 벡터 검색 계층을 분리하고, PostgreSQL의 변경 내용을 Milvus에 이벤트 기반으로 전달할 수 있는 이기종 데이터 저장소 참조 아키텍처를 제시하였다.

또한 BGE-M3 기반 밀집 벡터 검색과 mecab-ko 기반 어휘 검색을 가중 RRF 방식으로 결합하는 PostgreSQL 기반 한국어 하이브리드 검색 모듈을 구현하였다. 이 모듈은 pgvector와 textsearch_ko에서 생성한 검색 결과를 PostgreSQL 내부에서 융합하고, 애플리케이션이 단일 SQL 문장을 통해 하이브리드 검색 결과를 호출할 수 있도록 설계되었다. 이와 함께 pg_milvus를 통해 PostgreSQL의 SQL 인터페이스에서 Milvus의 벡터 검색을 호출할 수 있는 확장 경로도 구현하였다.

제안 시스템의 구성요소 수준 평가로서, MIRACL-ko로부터 구성한 100개 질의와 1,500개 문단의 평가 데이터셋을 이용하여 PostgreSQL 기반 하이브리드 검색 모듈의 검

색 효과를 분석하였다. BGE-M3 기반 벡터 검색은 Recall@10 0.9466, Recall@100 1.0000, MRR 0.7898 및 nDCG@10 0.8077을 기록하였다. 가중 RRF 기반 하이브리드 검색은 Recall@10 0.9471과 Recall@100 1.0000으로 유사한 재현율을 유지하면서, MRR과 nDCG@10을 각각 0.8219와 0.8284로 향상시켰다. 이는 본 연구에서 구성된 평가 환경에서 형태소 기반 어휘 정보가 밀집 벡터 검색의 상위 결과 순위를 보완하여, MRR과 nDCG@10이 각각 4.07%와 2.56% 높아지는 결과를 보였음을 의미한다.

본 연구의 주요 기여는 다음과 같이 정리할 수 있다. 첫째, PostgreSQL, Debezium, Kafka 및 Milvus를 연계하여 관계형 데이터 관리와 벡터 검색 계층을 분리한 데이터 파이프라인을 설계하고 컨테이너 기반으로 구현하였다. 둘째, BGE-M3 기반 밀집 벡터 검색과 mecab-ko 기반 어휘 검색을 가중 RRF로 결합하고, 이를 단일 SQL 인터페이스로 제공하는 한국어 하이브리드 검색 기능을 구현하였다. 셋째, MIRACL-ko 기반 평가 데이터셋을 이용하여 하이브리드 검색 모듈을 정량적으로 평가하고, 벡터 검색의 높은 재현율을 유지하면서 상위 검색 결과의 순위 품질을 개선할 수 있음을 확인하였다.

다만 본 연구는 몇 가지 한계를 가진다. 첫째, 검색 성능평가는 MIRACL-ko 전체 코퍼스가 아니라 100개 질의와 1,500개 문단으로 구성된 축소 평가 데이터셋을 대상으로 수행되었으므로, 대규모 검색 환경에 결과를 일반화하는 데 주의가 필요하다. 둘째, 정량적 평가는 PostgreSQL 내부에서 동작하는 하이브리드 검색 모듈에 한정되었으며, Milvus를 벡터 검색 계층으로 사용하는 검색 경로와 전체 PostgreSQL-Milvus 파이프라인의 종단 간 성능은 평가하지 않았다. 셋째, 단일 호스트의 WSL2 및 Docker Desktop 환경에서 시스템을 구현하였기 때문에 다중 노드 또는 클러스터 환경에서의 처리량과 확장성을 확인하지 못하였다. 넷째, 제안 파이프라인을 온톨로지 자동 구축 및 검증 서비스에 실제로 적용한 종단 간 성능 평가와 사용자 기반 평가는 연구 범위에 포함하지 않았다.

향후 연구에서는 MIRACL-ko 전체 코퍼스와 실제 업무 도메인의 대규모 문서 집합을 이용하여 검색 성능을 추가로 검증할 필요가 있다. 또한 PostgreSQL에서 Milvus로의 변경 전파 지연, INSERT·UPDATE·DELETE 처리량, 장애 이후의 복구 성능, 저장소 간 데이터 일치 및 데이터 규모 증가에 따른 확장성을 체계적으로 평가하고자 한다. 이와 함께 PostgreSQL 내부 검색 경로와 PostgreSQL-Milvus 검색 경로의 검색 지연 및 처리량을 비교하고, 제안 파이프라인을 온톨로지 자동 구축·검증 용

과 연계하여 실제 활용 가능성을 검토할 필요가 있다.

REFERENCES

- [1] J. Chen, S. Xiao, P. Zhang, K. Luo, D. Lian, and Z. Liu, "BGE M3-Embedding: Multi-Lingual, Multi-Functionality, Multi-Granularity Text Embeddings Through Self-Knowledge Distillation," arXiv preprint arXiv:2402.03216, 2024.
- [2] Y. Yoon and S. Kim, "Trends and Prospects of Retrieval-Augmented Generation (RAG) for Generative AI," Journal of the Korean Association of Computer Education, Vol. 28, No. 2, pp. 69-80, 2025. DOI: 10.32431/kace.2025.28.2.007
- [3] H. Kang and J. H. Yang, "Optimization of Word2vec Models for Korean Word Embeddings," Journal of Digital Contents Society, Vol. 20, No. 4, pp. 825-833, Apr. 2019. DOI: 10.9728/dcs.2019.20.4.825
- [4] Eunjeon Project, "mecab-ko / mecab-ko-dic: Korean Morphological Analyzer," <https://bitbucket.org/eunjeon>.
- [5] Pgvector, "pgvector: Open-source Vector Similarity Search for Postgres," <https://github.com/pgvector/pgvector>.
- [6] J. Wang, X. Yi, R. Guo, et al., "Milvus: A Purpose-Built Vector Data Management System," Proc. ACM SIGMOD International Conference on Management of Data, pp. 2614-2627, 2021. DOI: 10.1145/3448016.3457550
- [7] Debezium, "Debezium: An Open-source Distributed Platform for Change Data Capture," <https://debezium.io>.
- [8] C. Chen, C. Jin, Y. Zhang, S. Podolsky, C. Wu, S.-P. Wang, E. Hanson, Z. Sun, R. Walzer, and J. Wang, "SingleStore-V: An Integrated Vector Database System in SingleStore," Proceedings of the VLDB Endowment, Vol. 17, No. 12, pp. 3772-3785, 2024. DOI: 10.14778/3685800.3685805
- [9] J. Liu, Y. Zhang, C. Jin, A. Gupta, S. Liu, and J. Wang, "Fast Vector Search in PostgreSQL: A Decoupled Approach," 16th Conference on Innovative Data Systems Research (CIDR), Chaminade, USA, Jan. 2026.
- [10] G. V. Cormack, C. L. A. Clarke, and S. Büttcher, "Reciprocal Rank Fusion Outperforms Condorcet and Individual Rank Learning Methods," Proc. ACM SIGIR, pp. 758-759, July 2009. DOI: 10.1145/1571941.1572114
- [11] S. Bruch, S. Gai, and A. Ingber, "An Analysis of Fusion Functions for Hybrid Retrieval," ACM Transactions on Information Systems, Vol. 42, No. 1, Article 20, pp. 1-35, Jan. 2024. DOI: 10.1145/3596512
- [12] M. Wang, B. Tan, Y. Gao, H. Jin, Y. Zhang, X. Ke, X. Xu, and Y. Zhu, "Balancing the Blend: An Experimental Analysis of Trade-offs in Hybrid Search," arXiv preprint arXiv:2508.01405, 2025.

- [13] L. Gao, Z. Dai, T. Chen, Z. Fan, B. Van Durme, and J. Callan, "Complement Lexical Retrieval Model with Semantic Residual Embeddings," Proc. 43rd European Conference on Information Retrieval (ECIR), LNCS 12656, pp. 146-160, 2021. DOI: 10.1007/978-3-030-72113-8_10
- [14] X. Zhang, N. Thakur, O. Ogundepo, E. Kamalloo, D. Alfonso-Hermelo, X. Li, Q. Liu, M. Rezagholizadeh, and J. Lin, "MIRACL: A Multilingual Retrieval Dataset Covering 18 Diverse Languages," Transactions of the Association for Computational Linguistics (TACL), Vol. 11, pp. 1114-1131, 2023. DOI: 10.1162/tacl_a_00595

Authors



Min-Ho Park received his Master's degree from the Graduate School of Global Human Resource Development at Chung-ang University, Korea, in 2018. He is currently pursuing a Ph.D. at the Graduate School of

Business IT at Kookmin University. His primary research interests include LLM services of RAG and AI agent as well as AI-based business innovation strategies.



Hyunchul Ahn received his B.S. degree in Industrial Management and his M.E. and Ph.D. degrees from the KAIST Graduate School of Management, South Korea. He is currently a Professor at the Graduate School

of Business IT, Kookmin University. His research interests include AI applications in finance and marketing, as well as information systems adoption.