⚡ 쇼특허(Short-Cut) v3.0

아이디어의 입력만으로 선행특허 조사와 침해 리스크를 평가하는 AI 솔루션

Team: 뀨

SK networks x en·core

목차

시스템 개요

우리가 해결하려는 것

- ✓ 자연어 아이디어 입력만으로 유사 특허 후보(Top-K)를 빠르게 검색
- ✓ 청구항/구성요소 단위로 침해 위험을 자동 점검
- ✓ 선정 근거(인용 구간/랭킹/필터)와 함께 결과를 스트리밍으로 즉시 제공

쇼특허(Short-Cut) v3.0 정의

- ✓ 사용자 아이디어를 기준으로 선행특허를 검색·비교해 리스크를 빠르게 점검하는 AI 기반 시스템
- ✓ Self-RAG 기반 검색 고도화(멀티쿼리·HyDE·쿼리 재작성) + 하이브리드 검색(Dense+Sparse) 결합
- ✓ IPC 필터링 → 통합 랭킹(RRF) → Reranker → LLM 요약/분석까지 연결한 엔드투엔드 프로토타입

핵심 산출물

- ✓ 유사 특허 후보(Top-K) 및 선정 근거 요약
- ✓ 침해 리스크 평가: **위험 청구항/구성요소 매칭 포인트** 정리
- ✓ 구성요소 기반 회피·차별화 전략 + PDF 리포트/시각화(Guardian Map 등) 제공

목표와 가치

쇼특허(Short-Cut) v3.0는 출원/제품화 단계에서 사용자의 아이디어가 기존 특허와 얼마나 유사한지, 침해 리스크(청구항/구성요소)가 어디에 있는지, 그리고 어떤 방식으로 회피·차별화할지까지 빠르게 판단하도록 돕는 AI 기반 선행기술 분석 시스템입니다.

빠른 의사결정 지원	- 사용자 아이디어 입력만으로 유사 특허 후보(Top-K) 신속 제시 - 하이브리드 검색(Dense+Sparse) + IPC 필터링으로 관련 후보 압축 - 통합 랭킹(RRF) + Reranker로 핵심 후보 우선 도출
침해 리스크 평가 (Claim 관점)	- 청구항(Claim)·구성요소 단위로 매칭하여 침해 가능성 점검 - All Elements Rule 관점으로 충족/누락 요소를 명확히 구분 - 위험 청구항/구성요소를 근거(인용 구간/요약)와 함께 제시
회피/차별화 전략 제안	- 아이디어 vs 특허 구성요소 대비표 제공 - 충돌(위험) 요소 중심으로 설계 변경·대체안 제안 - 차별화 포인트 및 개발 방향을 리포트(PDF/시각화) 형태로 정리

※ 청구항: 특허가 법적으로 보호받는 권리 범위를 정의한 문장

사용자 플로우

입력 단계

- 사용자가 아이디어(기술 설명)를 자연어로 입력
- 필요 시 관심 기술 분야 IPC 필터 선택
- 검색 범위/관점(키워드, 목적, 비교 대상 특허 등) 설정

검색 단계

- 아이디어를 다양한 관점으로 확장(멀티쿼리 생성, HyDE/쿼리 재작성)
- 하이브리드 검색: Dense(의미 기반) + Sparse(BM25 키워드)
- 결과 통합(RRF)으로 후보군 통합 및 중복 제거

분석 단계

- Reranker(Cross-Encoder)로 상위 후보 정밀 재정렬
- 상위 특허에 대해 청구항(Claim)·구성요소 단위로 비교/매칭
- 위험 포인트(침해 가능성)와 근거(인용 구간/요약) 정리

출력 단계

- 결과를 스트리밍 방식으로 실시간 제공
- 유사 특허 후보 리스트(Top-K) + 선정 근거
- 침해 리스크 리포트 + 회피/차별화 가이드(PDF/시각화 포함)

데이터 수집

데이터 소스	Google Patents Public Dataset (BigQuery)
수집 기간	2018-01-01 ~ 2024-12-31
대상 국가	US, EP, WO, CN, JP, KR
수집량	10,000건(데모/프로토타입 목적)
도메인/IPC	AI/NLP 도메인 키워드 기반 선별 + IPC: G06F 16, G06F 40, G06N 3/5/20, H04L 12

데이터 전처리 파이프라인

데이터 추출

- BigQuery에서 특허 데이터 추출 (2018–2024)
- 대상: US / EP / WO / CN / JP / KR
- 텍스트 정규화(특수문자·공백·인코딩·HTML/ 개행 처리)

청구항 파싱

- 4-Level 청구항 파싱 적용(구성요소 단위 분해)
- 국가/형식 차이를 고려한 파싱 규칙 적용 (US/EP/KR 등)
- 괄호/대괄호/번호/서술 패턴 정리(전처리 규칙)

청킹 & 임베딩

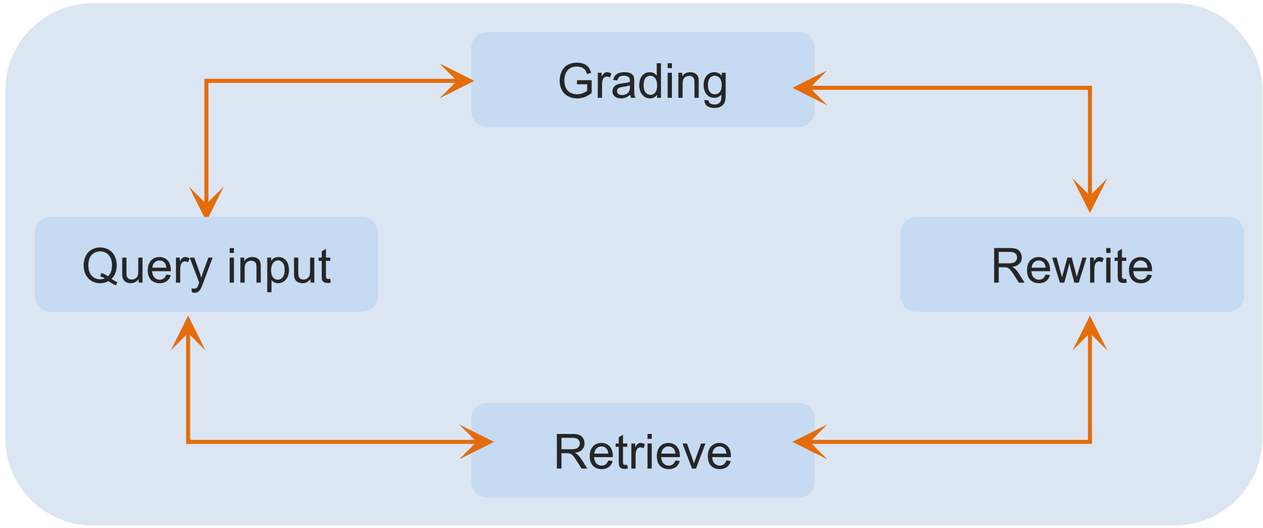
- 최대 1024 토큰 단위로 청킹
- 오버랩 128 토큰 적용(문맥 유지)
- 임베딩: text-embedding-3-small (1536차원)

인덱싱

- Dense: Pinecone(Vector DB) 인덱스
- Sparse: BM25 키워드 인덱스 구성
- Hybrid 검색 기반 구성(Dense + Sparse)
- 약 20,664 청크/벡터 생성

데이터 전처리는 검색 품질을 좌우하는 핵심 단계입니다.
BigQuery 데이터는 청구항 4-Level 파싱 → 청킹/임베딩을 거쳐 의미 기반 검색이 가능해지고,
Hybrid(Dense+Sparse/BM25) 인덱싱으로 하이브리드 검색 기반을 마련합니다.

핵심 기술 구성 (1)



Self-RAG + Grading/Rewrite Loop

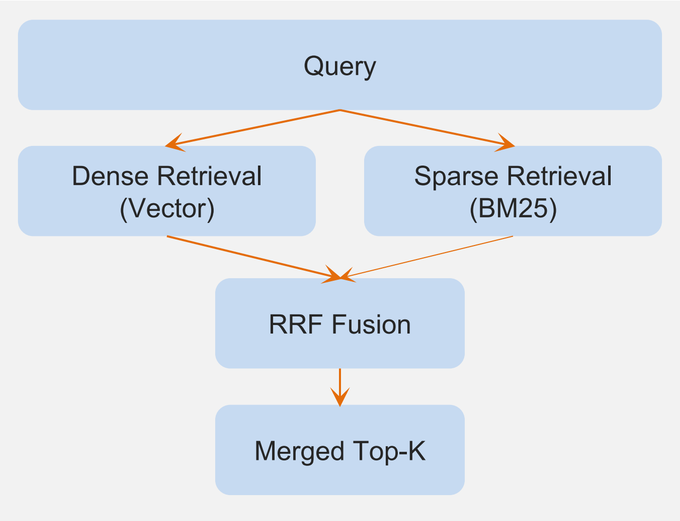
- 검색 결과에 대해 **0~1 관련성 점수(Grading)**를 부여
- 평균 점수가 임계값(예: 0.6) 미만이면 쿼리 재작성(Rewrite) → 재검색(Retrieve) 1회 수행
- 저품질 검색 결과를 자동 걸러내어 분석 품질을 안정화하는 루프

HyDE & Multi-Query RAG

- HyDE: 아이디어로부터 '가상 특허 설명/청구항' 형태의 문장을 생성해 검색 쿼리를 강화
- Multi-Query: 기술/청구항/문제해결 관점 등 다중 쿼리로 확장해 검색 커버리지 개선
- 짧거나 모호한 입력에서도 검색 재현율(Recall)과 후보 품질을 높이는 목적

핵심 기술 구성 (2)

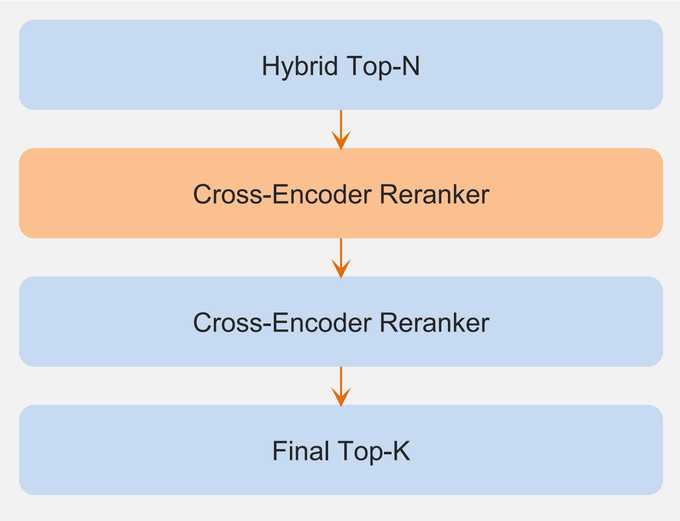
Hybrid Search + RRF



k=60, dense:sparse=0.5:0.5

- Dense(의미 기반) + Sparse(BM25 키워드) 검색을 결합해 후보를 폭넓게 확보
- Dense: Pinecone 벡터 검색 / Sparse: BM25 키워드 검색
- RRF(Reciprocal Rank Fusion)로 두 결과를 통합해 상위 후보를 안정적으로 선정
- 파라미터: k=60, dense:sparse 가중치 0.5:0.5

IPC Filtering & Reranker



ms-marco-MiniLM

- 사용자가 선택한 IPC(기술 분류)를 기준으로 검색 결과를 1차 필터링
- Cross-Encoder 기반 Reranker로 상위 후보를 정밀 재정렬
- 모델: ms-marco-MiniLM(Cross-Encoder)
- 최종적으로 분석 가치가 높은 Top-K를 선별

핵심 기술 구성 (3)

Claim-Level Analysis

- All Elements Rule 기반으로 청구항(Claim) 침해 가능성 평가
- 문서 유사도(abstract) 중심이 아닌, 구성요소 단위로 세부 매칭/비교
- 위험 청구항 및 침해 가능 포인트 도출(충족/누락 요소 구분)
- 아이디어 ↔ 청구항 구성요소 매칭/차이점 정리
- 구성요소 대비표 자동 생성 + 회피/차별화 방향 제안

피드백 & 시각화

- 사용자 피드백을 로그로 수집하여 품질 개선에 활용
- 피드백 데이터/분석 결과를 이력(SQLite)으로 관리(재조회·비교 가능)
- LLM 결과를 스트리밍 방식으로 즉시 출력(대기시간 감소)
- Guardian Map(성/침입자) + 특허 지형도 시각화로 후보군을 직관적으로 제공
- PDF 리포트 자동 생성(Top-K, 위험 청구항/구성요소, 회피·차별화 전략 포함)

정리 및 다음 단계로

시스템 아키텍처 및 모듈 구성

Streamlit(app.py) – UI 레이어

- 아이디어 입력/IPC 선택/실행 버튼 등 사용자 입력 처리
- 검색·분석 결과 스트리밍 출력 및 화면 구성
- 후보 리스트/리포트/시각화 화면 렌더링

- UI 레이어

patent_agent.py, vector_db.py, reranker.py

- Multi-Query/HyDE 기반 쿼리 생성 및 오케스트레이션
- Hybrid 검색(Dense + Sparse/BM25) + IPC 필터링 + RRF 결과 통합
- Cross-Encoder 기반 정밀 재정렬(Rerank) 및 Top-K 선별

- 핵심모듈

analysis_logic.py, session_manager.py, preprocessor.py, embedder.py, feedback_logger.py, history_manager.py, ui/*

- 청구항 파싱/청킹 등 전처리 파이프라인 및 임베딩 생성
- 인덱싱/검색 설정 지원 및 결과 후처리(근거 정리/요약 입력 구성)
- 사용자 피드백 로깅 및 분석 이력(SQLite) 관리(재조회·비교)
- UI 공통 컴포넌트 분리 및 상태(Session) 관리

- 데이터 처리 & 사용자 경험

정리 및 다음 단계로

한계 및 향후 개선 방향

현재 시스템은 데모용 10K 특허 데이터 기반으로 구축되어, 전체 특허 커버리지 및 도메인 확장에 한계가 있습니다.

- QA 현황: pytest 32개 테스트 100% Pass(DeepEval 포함)
- 데이터 확장: 전체 특허 DB/도메인 확장으로 검색 커버리지 개선
- 비용/안정성: OpenAI API Mock + 캐싱으로 비용 절감 및 재현성 강화
- 검색 고도화: IPC 필터 정교화 + Hybrid(Dense+BM25) 파라미터 튜닝 + RRF 개선
- 피드백 기반 고도화: 사용자 피드백 로그를 활용한 Reranker/검색 전략 개선
- 산출물 고도화: PDF 리포트/시각화(Guardian Map 등) 품질 및 자동화 범위 확대