Lecture 03 - Contents

An overview of the parts in the medical RAG systems lecture.

This outline is for guidance. Navigate the slides with the left/right arrow keys.

Lecture 3:

RAG for Healthcare

Evidence-Based AI

Ho-min Park

homin.park@ghent.ac.kr

powersimmani@gmail.com

RAG Architecture for Healthcare

Query Processing → Knowledge Retrieval → Evidence Generation

📚 Literature Search

Search 35M+ PubMed articles with semantic understanding

📋 Clinical Guidelines

Access WHO, CDC guidelines with real-time updates

💊 Drug Information

Query DrugBank, RxNorm for interactions and side effects

🔬 Diagnostic Support

Evidence-based differential diagnosis recommendations

🏥 Treatment Planning

Protocol recommendations based on latest research

⚠️ Safety Monitoring

Real-time adverse event detection and reporting

✅ Factual Accuracy

🔗 Source Citations

🔄 Always Up-to-date

🎯 Domain-Specific

Part 1:

Building Medical Knowledge Bases

Clinical Guidelines Ingestion

🌍 WHO
Global health standards and protocols

🏛 CDC
Disease prevention and control guidelines

🏥 Professional Societies
Specialty-specific recommendations

💊 FDA
Drug approval and safety information

🔬 NIH
Research-based clinical protocols

EU EMA
European medical guidelines

📋 Metadata Schema

Source: Organization name

Version: Publication date

Topic: Medical category

Evidence: Quality level

Updates: Revision history

Language: Multi-lingual support

Reserved Slot (L03_05)

추후 내용이 추가될 자리입니다. 강의 흐름의 연속성을 위해 번호를 보존합니다.

Drug Database Integration

DrugBank
13,000+
Comprehensive drug data with molecular structures

RxNorm
150,000+
Standardized medication nomenclature

FDA Labels
100,000+
Official prescribing information

Drug Knowledge Graph

Drug Entity ↔ Interactions ↔ Side Effects ↔ Indications

Interaction Matrix
Drug-drug, drug-food, drug-disease interaction checking

Pharmacokinetics
ADME properties, half-life, metabolism pathways

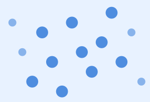
Adverse Events
FDA FAERS database with 10M+ reports

Pricing & Access
Cost information and formulary status

Vector Embedding Strategies

📷 Dense Embeddings

BERT, BioBERT, Sentence-BERT



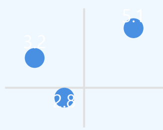
High-dimensional continuous space

✅ Semantic similarity
✅ Context understanding
❌ Computational cost

Best for: "chest pain" ≈ "cardiac discomfort"

📊 Sparse Embeddings

BM25, TF-IDF



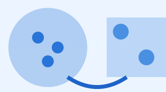
Most values = 0
Only keywords

✅ Fast retrieval
✅ Interpretable
❌ No semantics

Best for: Exact term "ICD-10 I21.0"

🔀 Hybrid Approach

Dense + Sparse fusion



✅ Best of both
✅ High accuracy (95%+)
⚠️ More complex

Recommended for medical applications

Retrieval Accuracy Comparison

📏 Dimension Selection

384d - Fast, general purpose

768d - BERT standard

1024d - High precision

Dense vs Sparse Retrieval

Aspect	Dense Retrieval	Sparse Retrieval
Similarity Type	Semantic meaning	Keyword matching
Speed	Medium (ANN search)	Fast (inverted index)
Accuracy	High for concepts	High for exact terms
Medical Terms	Understands synonyms	Exact match required

Dense best for:

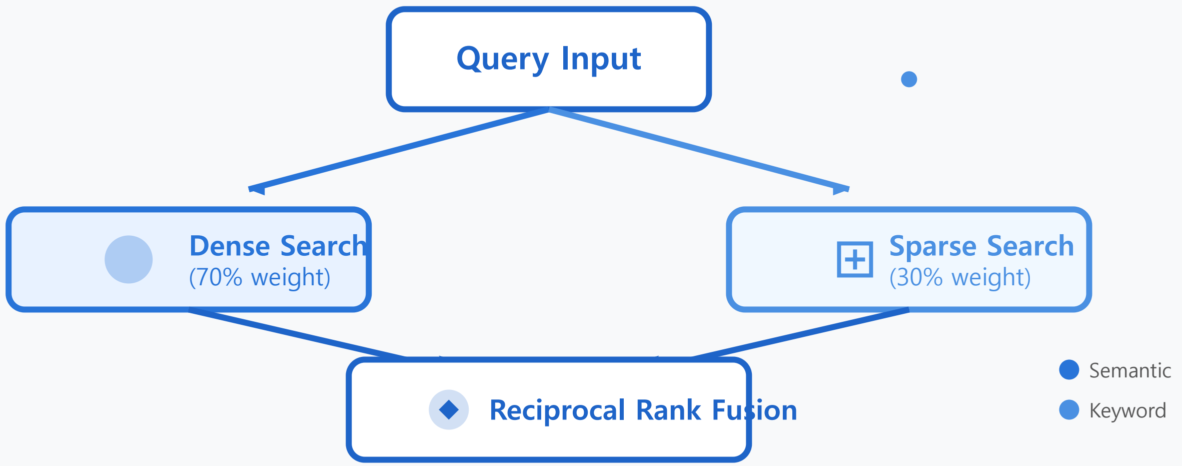
"Patient with chest pain and shortness of breath"

Sparse best for:

"ICD-10 code I21.0" or "Aspirin 81mg"

Hybrid Search Implementation

🔀 Hybrid Architecture



Weighted Sum	RRF (Recommended)	Ensemble
`score = α·dense + (1-α)·sparse`	`score = Σ 1/(k + rank)`	`Multiple models voting`

📈 Performance Improvement

Precision@10: **89% → 95%**

Recall@10: **76% → 92%**

Similarity Metrics for Medical Text

📐 Cosine Similarity

$\cos(\theta) = A \cdot B / (||A|| \ ||B||)$

Range: [-1, 1]

Best for: Dense embeddings

📏 Euclidean Distance

$d = \sqrt{\Sigma(ai - bi)^2}$

Range: [0, ∞]

Best for: Spatial similarity

🎯 Jaccard Index

$J = |A \cap B| / |A \cup B|$

Range: [0, 1]

Best for: Set overlap

🧠 Semantic Similarity

Based on medical ontology

Range: [0, 1]

Best for: Medical concepts

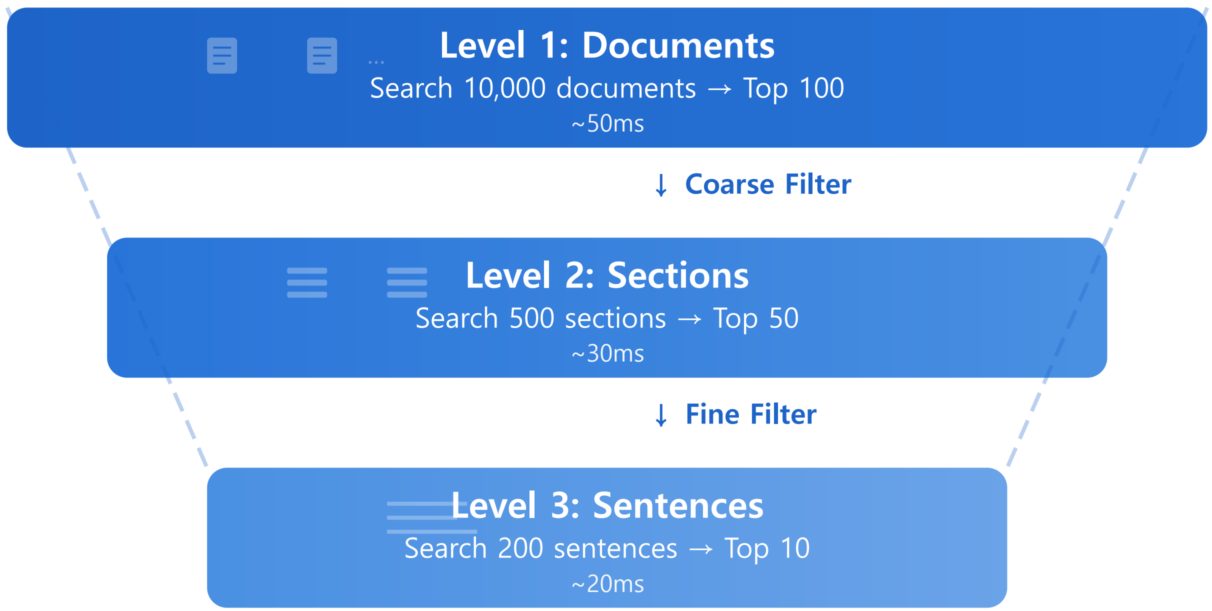
💡 Medical Text Example

Text 1: "Patient has myocardial infarction"

Text 2: "Heart attack diagnosed"

Cosine: **0.89** ✅ Jaccard: **0.12** ❌ Semantic: **0.95** ✅

Hierarchical Retrieval



⚡ Efficiency

100ms total vs 500ms flat

5x faster

🎯 Accuracy

Precision@10: 94%

2% drop acceptable

💾 Memory

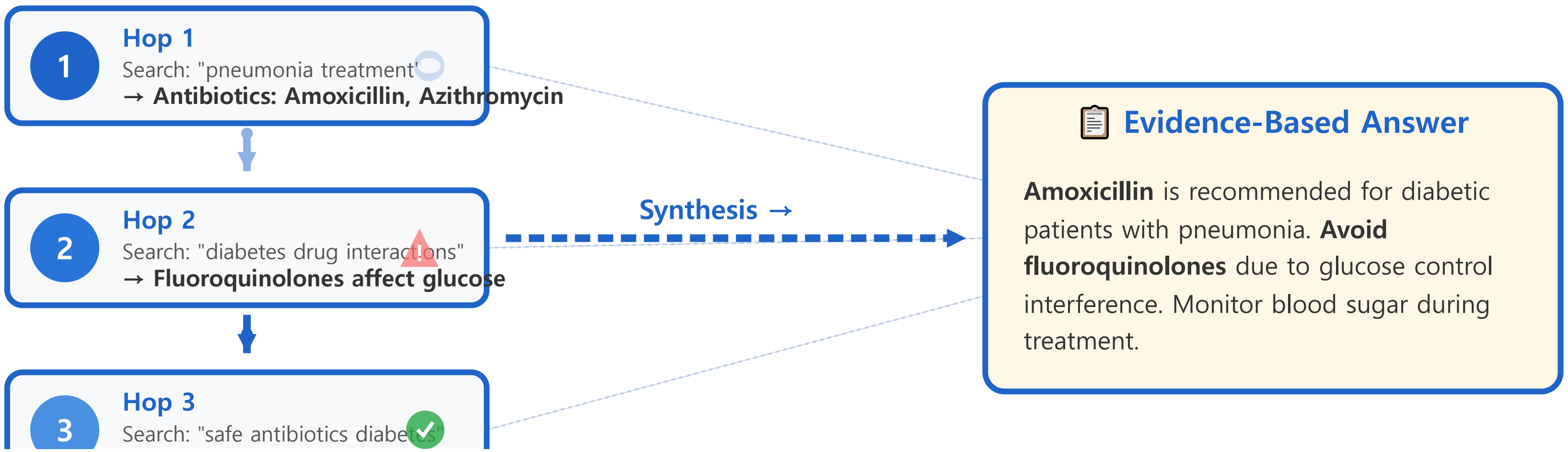
Incremental loading

Lower RAM usage

Multi-hop Reasoning

🔍 Multi-hop Query Example

Q: "Treatment for pneumonia in diabetic patients?"



Citation Generation

📝 Citation Formats

APA Style

Smith, J. (2024). Title. Journal, 12(3), 45-67.

MLA Style

Smith, John. "Title." Journal 12.3 (2024): 45-67.

Vancouver

Smith J. Title. Journal. 2024;12(3):45-67.

📊 Evidence Strength Indicators

⭐⭐⭐ High: Systematic reviews, RCTs

⭐⭐ Medium: Cohort studies

⭐ Low: Case reports, expert opinion

💡 Inline Citation Example

> Aspirin reduces cardiovascular events by 25% [Smith et al., 2024 ⭐⭐⭐] in high-risk patients [Johnson, 2023 ⭐⭐].

Evidence Scoring System

🔺 Evidence Pyramid

Meta-analyses & Systematic Reviews	Score: 9-10
Randomized Controlled Trials (RCTs)	Score: 7-8
Cohort Studies	Score: 5-6
Case-Control Studies	Score: 3-4
Case Reports & Expert Opinion	Score: 1-2

📋 GRADE System Factors

- ✅ Study design quality
- ✅ Consistency of results
- ✅ Directness of evidence
- ✅ Precision (CI, p-value)
- ✅ Publication bias check

Confidence Calibration

🎯 What is Calibration?

If model says 80% confidence, it should be correct 80% of the time

Temperature Scaling

Adjust logits with temperature T
p' = softmax(logits / T)

Platt Scaling

Logistic regression on outputs
p' = 1 / (1 + exp(A·p + B))

Isotonic Regression

Non-parametric calibration
Monotonic function fitting

📊 Calibration Metrics

ECE (Expected Calibration Error): |confidence - accuracy|

MCE (Maximum Calibration Error): max|confidence - accuracy|

Brier Score: Mean squared error of probabilities

Query Decomposition

🔍 **Complex Query**

"What are the contraindications for prescribing metformin in elderly patients with chronic kidney disease?"

↓ Decompose ↓

1. Metformin contraindications
2. Elderly patients drug considerations
3. Chronic kidney disease drug safety
4. Metformin + CKD interactions

↓ Integrate Results ↓

📋 **Synthesized Answer**

Metformin is contraindicated in CKD stage 4-5 (eGFR <30) due to lactic acidosis risk. In elderly CKD stage 3, dose reduction to 500mg BID with careful monitoring is recommended.

Vector Database Selection

🎯 Pinecone
- ✅ Fully managed
- ✅ Excellent scalability
- ❌ Proprietary, costly

🌊 Weaviate
- ✅ Open source
- ✅ Built-in vectorization
- ⚠️ Self-hosting required

⚡ Milvus
- ✅ High performance
- ✅ Trillion-scale
- ⚠️ Complex setup

🚀 Qdrant
- ✅ Rust-based speed
- ✅ Easy deployment
- ✅ Good for medical

🔍 Selection Criteria

- Data volume: **>10M vectors**
- QPS: **1000+**
- Latency: **<100ms**
- HIPAA compliance: **Required**

Reserved Slot (L03_20)

추후 내용이 추가될 자리입니다. 강의 흐름의 연속성을 위해 번호를 보존합니다.

Real-time Literature Updates

🔄 Real-time Update Pipeline

PubMed API Polling	→	Quality Filtering	→	Auto Indexing	→	Vector Update
Every 1 hour		Impact factor > 2.0		~5 min/article		Hot-swap

4,000+
New articles/day

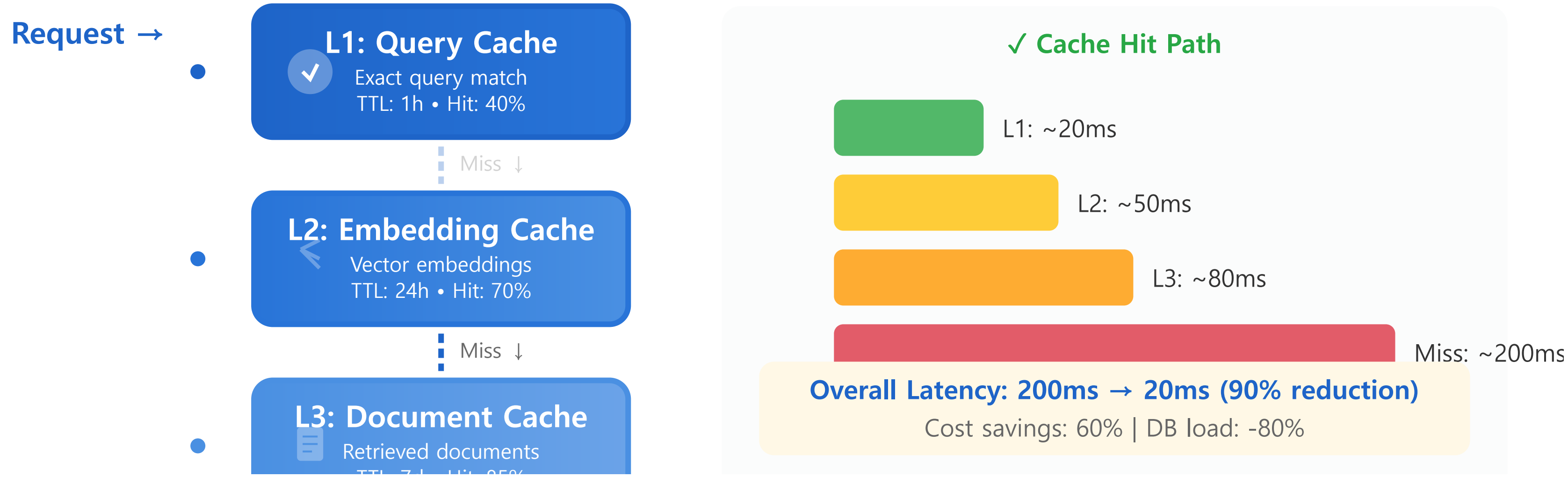
<30min
Detection latency

99.5%
Indexing success rate

🔍 Quality Filters

- ✓ Peer-reviewed journals only
- ✓ English language preferred
- ✓ Full-text availability
- ✓ Duplicate detection

Caching Optimization



⚙️ Redis Configuration

Memory: 16GB with LRU eviction

Persistence: RDB + AOF for durability

Cluster: 3 nodes with replication

📊 Performance Impact

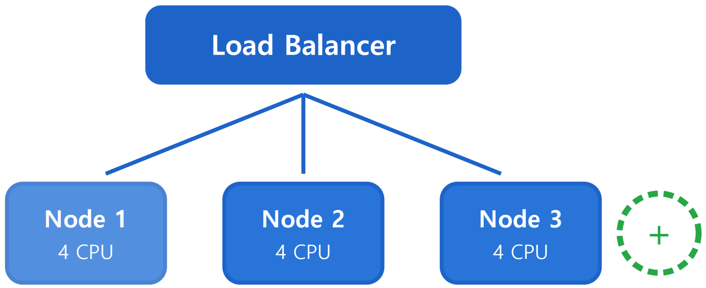
Latency reduction: **200ms → 20ms**

Cost savings: **60% reduction**

DB load: **-80%**

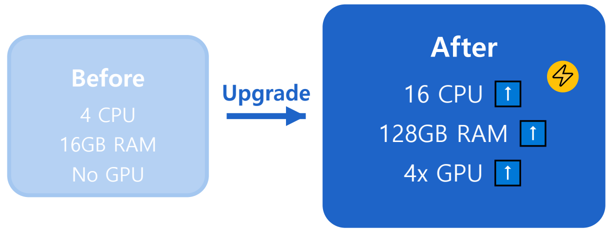
Scalability Patterns

↔ Horizontal Scaling



Capacity: Linear Growth ↑

↕ Vertical Scaling



Capacity: Hardware Limited 🔒

Faster per-node but can't scale infinitely

🔀 Sharding Strategies

Hash-based
Uniform distribution

Range-based
Date/category sharding

Geo-based
Regional data locality

🛡 High Availability

- ✓ 3x replication factor
- ✓ Auto-failover in <5s

✓ 99.99% uptime SLA

Case Study: UpToDate Integration

📚 UpToDate RAG Integration

World's Leading Clinical Decision Support

6,000+	12,000+	130+	40+
Clinical Topics	Expert Authors	Countries	Updates/day

🔍 Search Layer

Hybrid search with medical synonym expansion

🎯 Ranking

Evidence-based + Usage frequency + Recency

📝 Generation

Graded recommendations with source citations

📊 Clinical Impact

6% reduction in mortality (NEJM 2012)

19% reduction in length of stay

92% of users change clinical decision

Performance Benchmarks

🎯
95.2%
Retrieval Accuracy
Baseline: 87%

⚡
82ms
P95 Latency
Target: <100ms

📈
1,200
Queries/sec
Peak load

✅
98.5%
Citation Accuracy
Manual verification

⚖️ System Comparison

Factual Accuracy	**95%**	78%	91%
Latency (P50)	**45ms**	2000ms	120ms
Source Citations	**Yes**	No	Yes
Up-to-date Info	**Real-time**	Training cutoff	Index lag

Hallucination Mitigation

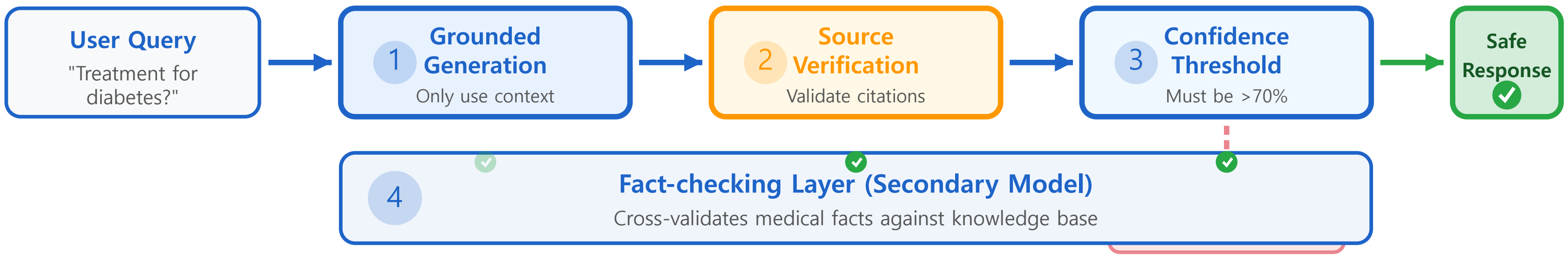
⚠ Types of Hallucinations

Factual Errors
Wrong dosage, incorrect diagnosis

Fabricated Citations
Made-up study references

Outdated Info
Using obsolete guidelines

🛡 Mitigation Pipeline



📊 **Results: Hallucination rate reduced from 12% → 2% | Citation accuracy: 98.5%**

Evaluation Metrics

🔍 Retrieval Metrics

Precision@K
relevant in top K / K

Recall@K
relevant in top K / total relevant

NDCG
Normalized Discounted Cumulative Gain

MRR
Mean Reciprocal Rank

📝 Generation Metrics

ROUGE-L
Longest common subsequence

BLEU
N-gram overlap with reference

BERTScore
Semantic similarity

🏥 Medical-Specific

Clinical Relevance
Expert judgment (1-5 scale)

Safety Score
Harm potential assessment

Citation Accuracy
Correct source attribution %

🎯 Recommended Thresholds

Precision@10: **> 90%**	NDCG@10: **> 0.85**
Clinical Relevance: **> 4.0/5.0**	Safety Score: **100%**

Hands-on: RAG Pipeline

💻 LangChain RAG Implementation

```
from langchain.vectorstores import Qdrant
from langchain.embeddings import OpenAIEmbeddings
from langchain.llms import OpenAI
from langchain.chains import RetrievalQA

# 1. Setup Vector Store
vectorstore = Qdrant(
    embeddings=OpenAIEmbeddings(),
    collection_name="medical_kb"
)

# 2. Create Retrieval Chain
qa_chain = RetrievalQA.from_chain_type(
    llm=OpenAI(temperature=0),
    retriever=vectorstore.as_retriever(
        search_kwargs={"k": 5}
    ),
    return_source_documents=True
)

# 3. Query with Citations
result = qa_chain({
    "query": "Treatment for Type 2 Diabetes?"
})

print(result['answer'])
print(result['source_documents'])
```

📋 Implementation Steps

1. Load medical documents (PDFs, text)
2. Chunk documents (512 tokens with 50 overlap)
3. Generate embeddings (BioBERT recommended)
4. Index in vector database (Qdrant/Weaviate)
5. Configure retrieval (hybrid search, top-k)
6. Test with medical queries and evaluate

Deployment Strategies

🚀 CI/CD Pipeline

1. Build	→	2. Test	→	3. Stage	→	4. Deploy
Docker container		Unit + Integration		Pre-prod validation		Canary rollout

📊 Monitoring Dashboard

Latency	Error Rate	QPS	CPU Usage
P95 < 100ms	< 0.1%	1000+	< 70%

🔄 Rollback Strategy

- ✓ Blue-green deployment for instant rollback
- ✓ Automated health checks every 30s
- ✓ Alert on accuracy drop > 5%

✓ Canary: 5% → 25% → 100% traffic

Thank You

🎓 Key Takeaways

✅

✅

✅

✅

💡

📚

Vector DBs: Pinecone, Weaviate, Qdrant

Research: arXiv.org (cs.CL, cs.IR)

Questions & Answers