JDCS
디지털콘텐츠학회논문지
Journal of Digital Contents Society
Vol. 27, No. 2, pp. 533-543, Feb. 2026

Check for updates

온톨로지 기반 지식 그래프(KG) RAG의 NL2SPARQL 병목 정량화를 위한 Oracle 상한 분석: M2/3 브래들리 전투차량 정비훈련교범을 중심으로

김 기 환*
국방기술진흥연구소 방위산업전략팀 연구원

Oracle-Based Quantification of the Natural Language-to-SPARQL Bottleneck in Ontology-Driven Knowledge Graph-Based Retrieval-Augmented Generation Systems: A Study Using Bradley Fighting Vehicle Soldier Training Procedures

Ki-Hwan Kim*

Researcher, Defense Industry Strategy Team, KRIT, Daejeon 35222, Korea

[요 약]

기존의 벡터 기반 검색 증강 생성(Vector RAG)의 한계를 보완하는 지식그래프 기반 검색 증강 생성(Knowledge Graph RAG, KG-RAG)은 계층적 문서 검색에 유리하나, 자연어 질의를 SPARQL 쿼리로 변환하는 과정(Natural Language to SPARQL, 이하 NL2SPARQL)에서의 오류가 주요 병목으로 작용한다. 본 연구는 미 육군 브래들리 전투차량 정비훈련교범(119개 Task, 13,542개 RDF Triple)을 대상으로, 이상적인 정답 쿼리를 가정한 오라클(Oracle) 기반 성능 상한 분석을 수행하여 NL2SPARQL 단계의 구조적 병목을 정량적으로 규명하는 것을 목적으로 한다. 실험 결과, GPT-4는 Oracle 대비 23.4%, Gemini는 48.5%의 성능 저하를 보였으며, 주요 원인은 속성명(33%)과 구문(22%), 복잡 질의(18%), 의미(16%) 오류였다. 이는 KG-RAG의 성능이 KG 자체보다 SPARQL 생성 품질에 의해 결정적으로 제한됨을 입증하며, 향후 온톨로지 제약 기반 생성 기술 및 난이도 적응형 Few-shot 학습 등 고도화 방안의 필요성을 시사한다.

[Abstract]

While Knowledge Graph-based Retrieval-Augmented Generation (KG-RAG) complements Vector RAG, errors in Natural Language-to-SPARQL (NL2SPARQL) conversion remain a critical bottleneck. This study quantitatively identifies these limitations through an Oracle-based upper-bound analysis of the U.S. Army Bradley maintenance manual (119 Tasks, 13,542 Triples). Experiments showed performance declines of 23.4% for GPT-4 and 48.5% for Gemini relative to the Oracle. The primary error sources were property names (33%), syntax (22%), complex queries (18%), and semantics (16%). Notably, the low Oracle performance (0.141) highlights the extreme structural difficulty of the domain. These findings demonstrate that KG-RAG performance is primarily constrained by SPARQL generation quality rather than the Knowledge Graph, indicating the need for advanced methods, such as ontology-constrained generation and difficulty-adaptive few-shot learning.

색인어 : 오라클 분석, 자연어-SPARQL 변환, 지식그래프 검색증강생성(KG-RAG), 거대언어모델, 병목 진단

Keyword : Oracle Analysis, NL2SPARQL, Knowledge Graph RAG, Large Language Model, Bottleneck Diagnosis

http://dx.doi.org/10.9728/dcs.2026.27.2.533

Received 08 December 2025; **Revised** 15 January 2026
Accepted 21 January 2026

***Corresponding Author; Ki-Hwan Kim**

Tel: +82-042-259-9834
E-mail: priest0228@naver.com

Ⅰ. 서 론

1-1 연구 배경 및 문제 정의

글로벌 항공우주 및 방산 산업에서는 전통적인 예방정비(Preventive Maintenance) 중심의 정비 체계가 AI 기반 예측정비(Predictive Maintenance) 및 상태기반정비+(Condition Based Maintenance Plus, CBM+)로 급속히 전환되고 있다[1]. 특히 McKinsey 보고서에 따르면 2033년까지 항공정비 기술자의 20%(약 70,000명)가 부족할 것으로 예상되어, AI 기술을 통한 정비 효율화가 산업 생존의 핵심 과제로 부상했다[2]. 이러한 흐름 속에서 AI, 디지털 트윈(Digital Twin) 기술의 융합이 차세대 지능형 정비 시스템의 핵심 요소로 주목받고 있다[3].

특히 온톨로지 기반 지식 그래프(Knowledge Graph, KG)는 복잡한 계층구조와 엄격한 절차 준수가 요구되는 군용 정비 매뉴얼에 적합한 구조적 특성을 갖추고 있다[4]. 국내에서도 방위사업청을 중심으로 한국형 무기체계에 CBM+ 도입을 위한 정책 연구가 활발히 진행되고 있으며, 온톨로지 기반 정비 지식체계 구축의 필요성이 대두되고 있다[5].

본 연구는 미 육군 브래들리 전투차량(Bradley M2/3 Fighting Vehicle) 정비훈련교범(STP 9-91M14-SM-TG)을 대상으로 한다. 해당 교범은 일반적인 서술형 문서와 달리, 각 임무(Task)가 조건(Condition), 기준(Standard), 그리고 세부 수행 절차(Performance Steps)의 엄격한 계층 구조로 정의된 절차적 데이터(Procedural Data)이다[6]. 이는 단순한 정보 검색을 넘어 조건과 절차 간의 논리적 연결을 이해해야 하므로, 온톨로지 설계 및 AI 적용 연구에 적합한 표본적 특성을 갖는다.

1-2 RAG 구조의 문제 정의와 연구 필요성

Retrieval-Augmented Generation(RAG)은 LLM에 외부 지식을 결합하여 사실 기반 응답을 생성하는 기술이다. 그러나 벡터 기반 RAG는 구현이 간단하지만, 문맥 누락·검색 노이즈·관계 정보 미반영 등의 한계를 보여 군 정비 문서처럼 계층적 구조와 복잡한 관계를 포함한 문서에서는 정확도가 낮아지는 문제가 반복적으로 보고되고 있다[7].

온톨로지 기반 KG-RAG는 SPARQL 질의를 통한 구조적 탐색으로 장비 정비 절차와 같은 규칙 기반 문서에서 높은 정확도를 제공할 수 있다. 그러나 실제 구현에서는 자연어를 SPARQL (NL2SPARQL)로 변환할 때의 오류가 전체 파이프라인(Pipeline)의 성능을 저하시키는 주된 병목으로 작용한다는 보고가 증가하고 있다[8]. 특히, 속성명 불일치, 경로 탐색 오류, 고급 문법 생성을 포함한 변환 실패는 SPARQL 실행 자체를 막거나 부정확한 결과를 반환하여 성능 저하를 유발한다[9].

그러나 기존 연구들은 대부분 end-to-end 성능 비교에 머무르며, NL2SPARQL 변환이 전체 성능을 얼마나 결정적으로 제한하는지를 정량적으로 분석한 연구는 부족하다. 따라서 기반 성능을 Oracle로 대체해 상한을 측정하는 평가 접근법이 필요하다.

1-3 연구 질문 및 논문 구성

이러한 문제 인식을 바탕으로 본 연구는 다음 네 가지 연구 질문을 설정하였다.

1) RQ1: Ontology 기반 KG-RAG의 이론적 성능 상한은 무엇인가?
2) RQ2: 현실적 구현(LLM 기반 SPARQL 생성)과 Oracle간 성능 격차는 어느 정도인가?
3) RQ3: NL2SPARQL 변환 과정에서 발생하는 주요 오류 유형은 무엇이며, 그 비중은 어떠한가?
4) RQ4: KG-RAG는 벡터(Vector) RAG에 비해 실질적 성능 우위를 확보할 수 있는가?

논문은 다음과 같이 구성된다. 2장에서는 관련 연구를, 3장에서는 온톨로지 설계 및 세 모델 구현을, 4장에서는 90개 Golden Query 구축 과정을 설명한다. 5-6장에서는 정량적·정성적 분석 결과를 제시하며, 7장에서는 연구 질문 답변과 시사점을 논의한다.

Ⅱ. 관련 연구

2-1 벡터 기반 RAG(Vector-Based RAG)

RAG는 LLM에 외부 지식을 결합하여 환각을 줄이는 기술로, Lewis 등에 의해 제안된 이후 핵심 기술로 자리잡았다[10]. 초기 RAG는 문서를 벡터 임베딩한 후 의미적 유사도로 검색하는 방식이 중심이었으며, DPR, ColBERT 등이 대표적이다[11],[12]. 그러나 벡터 기반 접근은 의미적 유사도만으로 검색이 이루어지므로, 계층 구조·속성 제약·규칙 기반 조건이 엄격한 정비 교범에서는 본질적 한계를 보인다[6]. Task 번호, Skill Level, 수행 주기(Frequency), 조건(Conditions) 등 정비 교범 특유의 구조적 메타정보를 정확히 반영하기 어렵다. 이러한 한계가 온톨로지 기반 접근이 등장한 핵심 배경이다.

2-2 온톨로지 기반 지식그래프 RAG(Ontology-Based KG-RAG)

이 한계를 보완하기 위해 온톨로지(Ontology) 기반 지식그래프(Knowledge Graph, KG)를 활용한 KG-RAG가 제안되었다. 온톨로지는 개념·관계·속성을 구조적으로 정의하며, SPARQL 기반 질의와 논리적 추론을 가능하게 한다[13].

대표적인 기술로는 Microsoft의 GraphRAG, LLM이 그래프를 단계적으로 탐색하는 Think-on-Graph, 그리고 다중 단계 추론(multi-hop)에 강점을 보이는 연구도 있다[14]-[16]. 그러나 KG-RAG의 전체 성능은 SPARQL 생성

품질에 강하게 의존한다. SPARQL은 문법적 제약이 엄격하고, 경로·속성·관계 방향을 정확히 지정해야 하기 때문에 NL2SPARQL 변환 오류가 전체 파이프라인의 주요 병목으로 작용한다[8]. 또한, 속성명 불일치, 필터 조건 누락, 논리경로 오류 등은 실행 실패 또는 비정확한 답변을 유발하며, 고난도 질의에서 이러한 현상은 더욱 두드러진다[9]. 요약하면, KG-RAG는 구조적 질의 처리라는 측면에서 벡터 기반 접근보다 본질적 우위를 갖지만, 실제 성능은 SPARQL 변환 품질에 의해 제한된다. 따라서 평가 과정에서 "정확한 SPARQL을 사용했을 때의 이론적 상한(Oracle)"과 "LLM이 생성한 SPARQL의 실제 성능" 간의 격차를 구분해 분석하는 것이 필수적이다.

2-3 NL2SPARQL 변환 기술

NL2SPARQL은 자연어 질의를 SPARQL로 변환하는 기술로, KG-RAG 구조의 핵심 단계다. 초기 연구는 템플릿 기반 규칙(Rule-based Template)을 사용했으나, 도메인 확장성과 문장 다양성에 취약했다[17]. 이후 신경망 기반 접근이 등장하며 seq2seq, BERT·T5, PICARD 등 문법 제약 디코딩 기법이 성능 향상에 기여했다[18],[19]. 최근에는 GPT-4 등 LLM을 활용한 few-shot prompting이 복잡한 온톨로지에서도 일정 수준의 SPARQL 생성을 가능하게 하고 있다[20].

그러나 변환 품질은 여전히 불안정하다. 대표 오류는 △속성명 오류(property) △문법 오류(syntax) △의미 오류(semantic) △다중 조건 등 복잡 질의 실패(complex)로 요약된다[8]. 이러한 오류는 SPARQL 실행 실패 또는 비정확한 답변을 유발하며, 정비 매뉴얼처럼 구조적 제약이 많은 문서에서 더욱 심각해진다. 기존 연구는 SPARQL 생성 품질 향상에 초점을 두었으나, 이 단계가 KG-RAG 전체 성능을 얼마나 제한하는지 상한(upper bound) 관점에서 분석한 연구는 부족하다. 본 연구는 Oracle SPARQL을 활용해 변환 단계가 성능을 제한하는 비중을 정량적으로 규명하고자 한다.

2-4 Oracle 연구 및 평가 방법론

Oracle 접근법은 시스템 구성요소를 이상적인 값으로 대체해 전체 성능의 상한(upper bound)을 측정하는 평가 방법론이다. 기계번역 분야에서는 Wiseman & Rush가 Oracle beam search를 통해 디코딩 상한을 연구하였고[21], Koehn & Knowles는 정렬 품질이 전체 번역 성능에 미치는 영향을 정량화했다[22]. 질의응답(Question Answering, QA) 연구에서도 Min 등은 Oracle passage 제공 실험을 통해 검색 단계가 end-to-end QA 성능의 주요 병목임을 규명하였다[23]. 본 연구는 이 개념을 KG-RAG 평가에 적용하였다. NL2SPARQL 단계를 전문가가 검증한 Golden SPARQL로 대체하여 Oracle 성능을 계산하고, 실제 LLM 기반 성능과 비교함으로써 NL2SPARQL 단계가 전체 성능에서 차지하는 병목 비중을 정량적으로 평가한다.

Ⅲ. 연구 방법론

3-1 전체 시스템 구조

본 연구는 NL2SPARQL 변환이 KG-RAG 성능에 미치는 영향을 정량화하기 위해, 동일한 자연어 질의에 대해 SPARQL 생성 방식만 다른 세 가지 모델을 비교하였다. 아래 표 1은 각 모델의 구성을 요약한다.

표 1. 각 모델별 특징
Table 1. Feature of each model

Model	Component	SPARQL Generation	Retrieval Mechanism
Model A	LangChain, FAISS, GPT4	N/A (Vector-based)	Vector-Similarity
Model B – LLM	PDFLib, OWL Ontology GPT-4	LLM Generation	Ontology-based KG
Model B – Oracle	RDFLib, Golden,GPT-4 SPARQL	Verified SPARQL	Oracle KG Upper Bound

본 연구의 전체 시스템 구조는 동일한 자연어 질의에 대해 Model A, Model B-LLM, Model B-Oracle이 서로 다른 방식으로 입력 질의(예: What is Task 091-91M-1001?)가 주어지면 검색(Retrieval), 생성(Generation), 출력(Output) 과정을 수행하도록 구성되어 있다. 아래 그림 1은 세 모델의 end-to-end 처리 절차를 요약한 것이다.

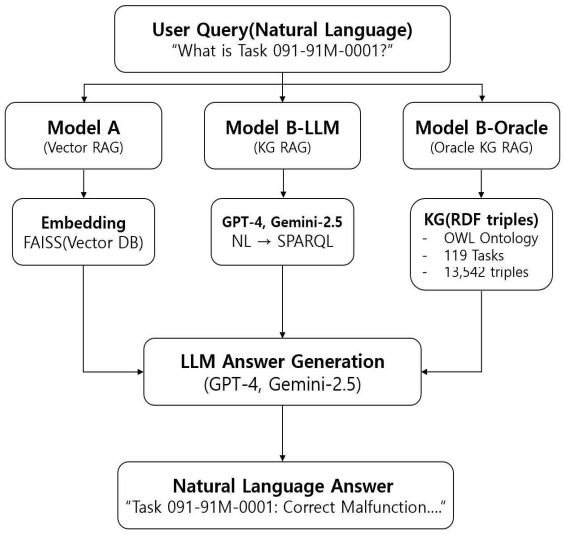

그림 1. 3가지 RAG모델 시스템 아키텍처 구조
Fig. 1. Three-model RAG system architecture

1) Model A (Vector RAG)

텍스트 청크를 임베딩하여 FAISS에서 유사도 기반으로 검색

하고, 검색 결과를 GPT-4가 그대로 활용해 답변을 생성한다.

2) Model B-LLM (LLM 기반 KG-RAG)

GPT-4와 Gemini-2.5가 질의를 SPARQL로 변환하고, 지식그래프에서 실행한 결과를 이용해 답변을 생성한다.

3) Model B-Oracle (Oracle KG-RAG)

SPARQL 생성 대신 전문가 검증 Golden SPARQL을 사용해 검색 오류를 제거한 상한 성능을 측정한다.

세 모델의 차이는 SPARQL 생성 방식의 정확성에만 있으며, 온톨로지·데이터셋·후처리 방식은 동일하게 유지된다. 따라서 Oracle 대비 LLM 기반 SPARQL 성능 저하는 순수하게 NL2SPARQL 변환 품질의 한계를 정량적으로 반영한다. 평가 기준은 아래 표 2와 같이 세 가지 층위로 구성된다.

표 2. 연구 평가 기준
Table 2. Research evaluation and analysis methodology

Category	Description
Quantitative Evaluation	Computation of F1-Score, Precision, Recall, and Cosine Similarity metrics
Bottleneck Quantification	Quantifying performance gap using bottleneck ratio
Qualitative Analysis	Classification of SPARQL generation errors by type and analysis of their root causes.

이 구조를 통해 Vector RAG와 KG-RAG의 구조적 차이를 명확히 비교하고, NL2SPARQL 단계가 전체 성능에서 차지하는 제한 요인을 독립적으로 측정할 수 있다.

3-2 온톨로지 설계

본 연구의 온톨로지는 미 육군 브래들리 전투차량 정비훈련교범(STP 9-91M14-SM-TG)의 계층 구조를 기반으로 설계하였다. 온톨로지는 OWL 2 DL 형식으로 구현되었으며, 주요 클래스와 인스턴스로 구성된다. 표 3은 각 Task별 세밀하게 정의된 속성 구조를 요약한다. 온톨로지 모델은 Protege(5.6.7)에서 작성한 후 RDFLib을 사용하여 Colab 환경에서 triple 형태로 변환하였다.

표 3. 클래스, 데이터타입 및 속성 예시
Table 3. Example of class, datatype, and object properties

Class	Datatype Properties	Object Properties
Task	title	requiresSkillLevel
SkillLevel	taskNumber	hasConditions
SubjectArea	locationCode	trainedAtLocation
Conditions	conditionsText	frequencyCode
References	referenceType	standardsText

최종 온톨로지는 총 13,542개의 RDF triple로 구성되며, 각 triple은 Task–속성–값의 구조적 관계를 명시적으로 표현하여 SPARQL 기반 질의를 정밀하게 지원한다. 이를 바탕으로 기존 벡터 기반 접근에서는 불가능했던 계층적·조건적 탐색을 가능하게 했다. 그림 2는 Task 091-91M-1001을 중심으로 한 Triple 구성과 지식 그래프 구조를 시각적으로 보여준다.

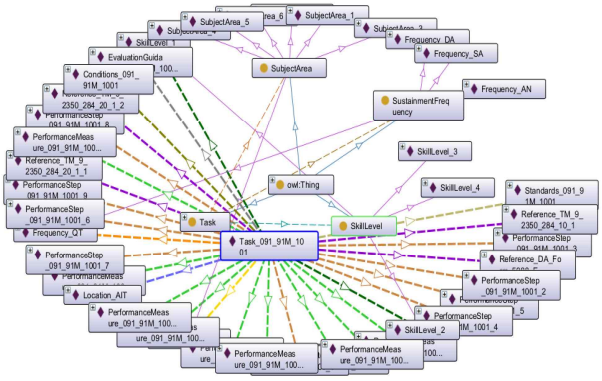

그림 2. 091-91M-1001 과업 온톨로지 지식 그래프 구현 예
Fig. 2. KG Sample Focused on Task 091-91M-1001

Ⅳ. 실험 설계

4-1 데이터셋 구축 및 실험 환경(Dataset & Experimental Setup)

본 연구는 미 육군 브래들리 장갑차 정비훈련교범의 119개 Task를 대상으로 한다. 온톨로지는 Protege 5.6.7을 통해 13,542개의 RDF Triple로 구축되었으며, PDF 원문은 555개의 텍스트 청크로 가공되었다.

실험 모델로는 GPT-4와 Gemini 2.5 Flash를 사용하였다. 결과의 재현성(Reproducibility) 확보를 위해 Temperature 파라미터를 0.0으로 설정하여 무작위성을 배제한 결정론적 디코딩(Deterministic Decoding)을 수행 하였다. 또한, 시스템 프롬프트에 스키마 정보와 구문 제약을 명시하는 'Schema-Aware Prompting' 전략을 적용하여 환각을 억제하였다. 평가 질의셋은 단일 사실 검색부터 다중 조건을 포함한 복합 추론 유형까지 포괄하며, 전문가 검증을 거친 90개의 Golden Query로 구성된다.

4-2 평가 지표

본 연구의 평가는 정량 지표와 정성 분석으로 구성된다. 정량 평가는 Oracle 상한과 NL2SPARQL 병목 산정을 위해 F1-Score, Precision, Recall, Cosine Similarity를 사용하였다. 정성 평가는 오류 패턴을 네 가지 유형으로 분류하였다.

1) 문법 오류(Syntax Error): SPARQL 문법 규칙 위반
2) 속성 오류(Property Error): 온톨로지 스키마 불일치
3) 의미 오류(Semantic Error): 관계 방향 및 타입 혼동
4) 복합 조건 오류(Complex Question Error): 다중 조건 등 고급 문법 생성 실패

4-3 최소 샘플 크기 산정

본 연구는 119개 Task의 Subject Area별 분포를 대표하는 평가 질의셋을 구축하기 위해, 층화 추출(Stratified Sampling) 이론에 기반한 최소 샘플 크기를 산정하였다. 층화 추출에서 소규모 층의 신뢰성을 확보하기 위해서는 층별 최소 3~5개의 표본이 필요하다는 고전 이론은 현대 표본설계 방법론에서도 지속적으로 적용되고 있다[24],[25]. 가장 작은 범주인 SA1(Task 4개)에 최소 3개 이상의 질의가 할당되도록 전체 샘플 크기 N을 결정하였다. 기대빈도는 식 (1)과 같다.

$$E_{SA1} = N \times \frac{4}{119} \geqq 3 \quad \rightarrow \quad N \geqq 89.25 \qquad (1)$$

N을 30부터 105까지 변화시키며 SA1의 기대빈도를 계산한 결과, N=90일 때 $E_{SA1} = 3.03$으로 처음으로 최소 기준($\geqq 3$)을 충족하였다(표 4). 따라서 본 연구는 90개 질의를 최적 표본 크기로 채택하였다.

표 4. 질문 수(N)에 따른 SA1 기대빈도(E) 산출
Table 4. Min. expected frequency(E) by number of query(N)

Number of Query	Formula for E_{SA_1}	E_{SA_1}	$E_{SA_1} \geqq 3$
30	30 x (4/119)	1.01	x
45	45 x (4/119)	1.51	x
60	60 x (4/119)	2.02	x
75	75 x (4/119)	2.52	x
90	90 x (4/119)	3.03	√
105	105 x (4/119)	3.53	√

4-4 비례 층화 설계를 통한 평가 질의 배분

N=90개의 질의를 119개 Task의 Subject Area별 분포에 비례하여 배분하기 위해 비례 층화 설계(Proportional Stratified Design)를 채택하였다. 각 Subject Area(SA_i)에 배정될 질의 수(n_i)는 다음 식 (2)를 통해 산정하였다.

$$E_i = 90 \times \frac{Task_{SA_i}}{119}, \qquad n_i = \partial\,(E_i) \qquad (2)$$

최종 배분 결과는 다음 표 5에 제시하였으며, 이론적 기대빈도와 실제 배분 간 평균 편차는 0.28개로 매우 작았다. 모든 Subject Area의 배분값(Planned Allocation)은 3 이상으로, 층화 추출의 통계적 안정성을 확보하였다. 이러한 비례 배분 방식을 통해 90개 질의가 119개 Task의 Subject Area별 분포를 정확히 반영하도록 설계되었으며, 특정 영역에 편향되지 않은 균형 잡힌 평가 데이터셋을 구축하였다.

표 5. 비례 층화 설계 기반 질의 배분
Table 5. Query Allocation Based on Proportional Stratified

Subject Area	Total Task	Ratio (%)	Expected Freq.(E_i)	Planned Allocation (n_i)	$E_i - n_i$
SA1	4	3.4	3.03	3	0.03
SA2	36	30.3	27.23	28	0.77
SA3	23	19.3	17.39	17	0.39
SA4	16	13.4	12.10	12	0.10
SA5	9	7.6	6.81	7	0.19
SA6	19	16.0	14.37	14	0.37
SA7	12	10.1	9.08	9	0.08
TOTAL	**119**	**100**	**90.00**	**90**	**Avg. 0.28**

4-5 난이도 기반 Golden Query 선정

앞서 비례 층화 설계로 대표성을 확보한 90개의 평가 질의는 SPARQL 질의 복잡도를 반영하여 Easy, Medium, Hard 세 단계로 분류하였다[26],[27]. 난이도 기준은 아래 표 6과 같이 정의하였다.

표 6. 질의 난이도 분류 기준
Table 6. Query difficulty classification criteria

Difficulty Level	Description
Easy	Simple query based on a single triple property
Medium	Query utilizing 2–3 triples, including simple operations
Hard	Query involving multiple triples, conditional filters, aggregation, and relational inference.

· Easy: 단일 triple 기반 속성(Property) 단순 질의
· Medium: 2~3 triple 사용 + 단순 연산 포함 질의
· Hard: 다중 triple + 조건 필터 + 집계 및 관계 추론

난이도 분류 결과는 쉬움 37개(37.8%), 보통 40개(40.0%), 어려움 22개(22.2%)로 구성된다. 자연어 질의는 GPT-4 및 Gemini 2.5로 생성 후 정비 전문가 검증을 거쳤으며, SPARQL은 Protege에서 실행 검증 후 Golden Ground Truth(GT)로 확정하였다. 난이도별 구조적 특징은 아래 표 7을 통해 확인할 수 있다.

첫째, Easy 난이도는 단일 triple을 활용한 기본 속성 조회로, "What are the required references for task 091-91M-1001?"과 같은 형태가 대표적이다. 이는 특정 Task의 단일 속성값(reference)을 직접 반환하는 가장 단순한 SPARQL 패턴이며, 약 4개의 triple만을 사용한다.

둘째, Medium 난이도는 두세 개의 triple을 연속적으로 탐색하는 경로 기반 질의로 구성된다. 예를 들어 "What are the standards for task 091-109-0001?" 질의는 (Task → Standards → standardsText)와 같은 구조적 경로 추적과 더불어 단일 속성 조회보다 한 단계 높은 복잡도를 요구한다.

표 7. 난이도 기준 Golden SPARQL 예시
Table 7. Example of Golden SPARQL queries by difficulty

Difficulty	Natural Language	Golden SPARQL	Expected Result

Easy	"What are the required references for task 091-91M-1001?"	PREFIX bradley: <http://army.mil/bradley/ontology#>
		SELECT ?refCode ?refType
		WHERE {
		?task bradley:taskNumber "091-91M-1001" .
		?task bradley:hasRequiredReference ?ref .
		?ref bradley:referenceCode ?refCode .
		?ref bradley:referenceType ?refType .
		}
			DA Form 5988-E, TM 9-2350-284-10-1, TM 9-2350-284-20-1-1, TM 9-2350-284-20-1-2

Medium	"What are the standards for task 091-109-0001 (Maintain TMDE)?"	PREFIX bradley: <http://army.mil/bradley/ontology#>
		SELECT ?standardsText
		WHERE {
		?task bradley:taskNumber "091-109-0001" .
		?task bradley:hasStandards ?std .
		?std bradley:standardsText ?standardsText .
		}
			Maintain the TMDE in accordance with the applicable technical publications, procedures, and specifications. When this task was completed, the equipment was fully mission-capable, calibration date was current, and maintenance actions had been identified.

Hard	"Which tasks are trained at ALC (Advanced Leaders Course) and require Skill Level 3?"	PREFIX bradley: <http://army.mil/bradley/ontology#>
		SELECT ?taskNumber ?title
		WHERE {
		?task a bradley:Task .
		?task bradley:taskNumber ?taskNumber .
		?task bradley:title ?title .
		?task bradley:trainedAtLocation ?loc .
		?loc bradley:locationCode "ALC" .
		?task bradley:requiresSkillLevel ?sl .
		?sl bradley:skillLevelNumber 3 .
		}
		ORDER BY ?taskNumber
			Subject Area 5 (COMMON LOGISTIC TASKS):
			091-CLT-3001 through 091-CLT-3012 (9 tasks)
			Subject Area 6 (TECHNICAL TASKS):
			091-91M-3004 through 091-91M-3046 (10 tasks)
			Total: 19 tasks

셋째, Hard 난이도는 다중 triple, 조건 필터링, 관계 제약이 복합적으로 결합된 고난도 질의로 구성된다. 대표적으로 trainedAtLocation="ALC" AND Skill Level=3 조건을 동시에 만족하는 Task 목록을 조회하는 질의는 최소 6개 이상의 triple, 조건문, ORDER BY 절을 포함한다. 이 유형은 여러 엔티티 관계를 동시에 추론해야 하므로 LLM의 구조적·논리적 처리 능력을 평가하는 데 가장 중요한 역할을 한다.

V. 정량적 성능 분석 및 결과

본 장에서는 90개 Golden Query를 기반으로 세 모델의 성능을 F1, Precision, Recall, Similarity로 비교한다. NL2SPARQL이 KG-RAG 성능에 미치는 영향을 분리하기 위해 Oracle 접근법을 사용하였다. Oracle은 Golden SPARQL을 통해 검색 오류를 제거한 이론적 상한선으로, LLM 기반 모델과의 차이는 NL2SPARQL 단계의 순수한 성능 손실을 나타낸다. 병목률(Bottleneck Ratio)은 식 (3)과 같이 정의된다.

$$Bottleneck\ \ Ratio(R_{LLM}) = 1 - \frac{F1_{B-Oracle} - F1_{B-LLM}}{F1_{B-Oracle}} \quad (3)$$

이 지표는 LLM 기반 SPARQL 생성이 이론적 상한 대비 얼마나 성능을 제한하는지를 백분율로 표현한다. 본 장에서는 먼저 전체 성능 비교를 통해 RQ1~RQ4에 답하고(5-1절), 이어서 난이도별 성능 분석(5-2절)과 LLM별 비교 분석(5-3절)을 수행하여 NL2SPARQL 병목의 특성을 다각도로 규명한다. 전체 분석 프레임워크는 그림 3에 제시되어 있다.

5-1 전체 성능 비교: 이론적 상한과 병목 정량화(RQ1~RQ4)

다음 표 8은 세 모델(Model A, Model B-LLM, Model B-Oracle)의 전체 성능을 요약한 것이다. Model A는 baseline이며, LLM 기반 Model B-LLM과 Oracle Model B-Oracle은 SPARQL 생성 방식만 다르고 나머지 시스템 구성은 동일하다.

표 8. 전체 정량적 성능 지표 비교
Table 8. Overall performance comparison

LLM	Model	F1 Score	Precision	Recall	Similarity
GPT-4	A-Vector	0.046	0.028	0.294	0.025
	B-LLM	0.108	0.086	0.208	0.064
	B-Oracle	0.141	0.109	0.288	0.088
Gemini 2.5 Flash	A-Vector	0.031	0.018	0.238	0.016
	B-LLM	0.085	0.060	0.187	0.048
	B-Oracle	0.165	0.149	0.273	0.105

표 8의 결과는 본 연구의 핵심 질문인 구조적 병목과 모델

별 특성을 규명하는 네 가지 중요한 패턴을 시사한다.

첫째, 모든 모델에서 F1-Score의 절대 수치가 낮게 측정되었다(최고 0.165). 이는 군 정비 매뉴얼의 구조적 난이도를 시사하기도 하지만, 근본적으로는 본 연구의 병목 진단 중심 실험 설계에 기인한다. 본 연구는 NL2SPARQL 변환 단계의 정확성을 독립적으로 평가하기 위해, 검색된 데이터(Entity)를 자연어 문장으로 변환하는 답변 생성(Answer Generation) 단계를 배제하였다. 이로 인해 모델이 출력한 정제된 데이터 값과 정답셋(Golden Answer)의 자연어 문장 간에 형식 불일치(Modality Mismatch)가 발생하여, 핵심 정보를 정확히 추출했음에도 토큰 중첩(Token Overlap) 부족으로 점수가 낮게 산출되었다. 따라서 본 결과의 해석은 절대 점수가 아닌, Oracle 대비 상대적 성능 격차(병목률)에 집중되어야 함을 시사한다.

둘째, Oracle 모델의 Precision이 다른 지표 대비 유의미하게 높다(GPT-4: 0.109, Gemini-2.5: 0.149). 이는 Golden SPARQL을 통해 정확한 Triple 경로를 탐색할 경우 검색 정밀도가 크게 향상됨을 보여주며, KG-RAG가 정확한 질의 생성만 보장된다면 Vector RAG 대비 우수한 신뢰성을 가질 수 있음을 입증한다.

셋째, Vector RAG(Model A)는 Recall은 상대적으로 높지만 Precision이 매우 낮다(GPT-4 기준 Recall 0.294, Precision 0.028). 이는 구조적 제약 없이 임베딩 유사도 기반으로 폭넓게 검색하면서 발생하는 전형적인 고재현-저정밀(High-Recall, Low-Precision) 문제를 반영하며, 정확한 정답이 필요한 정비 도메인에서의 한계를 보여준다.

넷째, 모든 모델에서 Cosine Similarity가 F1보다 낮다. 이는 생성된 답변이 정답(Golden Answer)과 단어·서술 수준에서도 상당한 차이를 보이며, 앞서 언급한 형식 불일치와 더불어 의미적 모호성이 구조적 질의 실패와 함께 나타나고 있음을 시사한다.

이러한 결과를 종합하면, 각 모델은 다음과 같은 뚜렷한 특성을 보인다.

1) Vector RAG: 과다 검색 중심의 유사도 모델
2) LLM 기반 KG-RAG: SPARQL 생성 품질에 민감한 구조 중심 모델
3) Oracle KG-RAG: 전 지표에서 우수한 상한 모델

특히 주목할 점은 Gemini-2.5의 역설적 성능 차이이다. Gemini-2.5는 이상적인 조건인 Oracle 모드에서는 GPT-4를 앞서지만(B-Oracle F1: 0.165 vs 0.141), 실제 생성 모드(B-LLM)에서는 오히려 뒤처지는 결과(B-LLM F1: 0.085 vs 0.108)를 보였다. 이는 Gemini-2.5가 잠재적인 성능 상한(Potential Capability)은 우수하나, 복잡한 제약 조건하에서 정확한 SPARQL 코드를 생성하는 구현 능력(Implementation)은 GPT-4보다 취약함을 시사한다. 이러한 미세한 특성은 본 연구가 제안한 Oracle 기반 비교 분석을 통해서만 분리 및 해석이 가능하다.

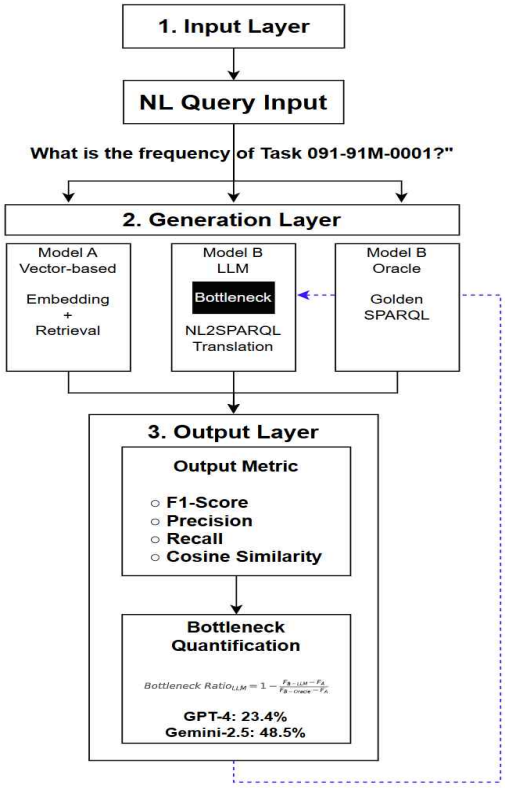

그림 3. 병목현상 분석 프레임워크
Fig. 3. Bottleneck analysis framework

아래 그림 4는 F1-Score를 기준으로 세 모델의 상대 성능을 시각화한 것이다. Vector RAG(Model A)를 기준(1.0배)으로 했을 때 GPT-4 시스템에서는 Model B-LLM이 2.35배, Model B-Oracle이 3.07배 성능 향상을 보였다. Gemini-2.5 시스템에서는 각각 2.74배, 5.32배로 더 큰 차이를 나타냈다. 이 배율 차이는 두 가지 중요한 통찰을 제공한다.

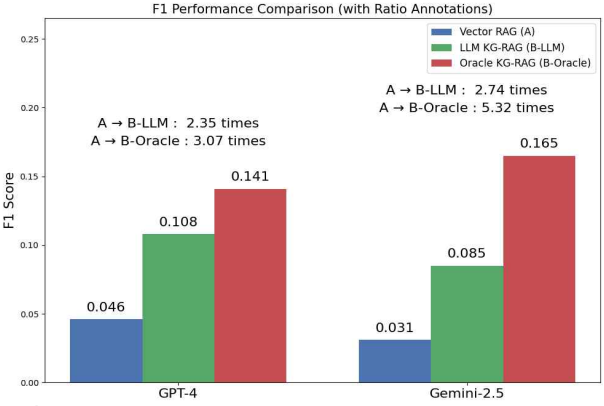

그림 4. 모델별 F1-Score 비교 및 성능 배율
Fig. 4. F1-Score comparision across models with ratio

첫째, Oracle의 일관된 우수성이다. GPT-4와 Gemini-2.5 모두에서 Oracle 모델은 Vector RAG 대비 3배 이상 성능을 향상시키며, 구조 기반 탐색의 이론적 잠재력이 LLM 종류와 무관하게 검증되었다(RQ1).

둘째, NL2SPARQL 병목의 정량화이다. B-LLM과 B-Oracle의 성능 격차를 Bottleneck Ratio로 계산하면 GPT-4는 23.4%, Gemini-2.5는 48.5%를 기록하였다(RQ2). 아래 수식들은 두 LLM의 병목률 계산 과정을 보여준다.

$$R_{LLM} = 1-\frac{0.108}{0.141} = 0.234(23.4\%)\ (GPT-4) \quad (4)$$

$$R_{LLM} = 1-\frac{0.085}{0.165} = 0.485(48.5\%)\ (Gemini-2.5) \quad (5)$$

이는 NL2SPARQL 변환이 전체 KG-RAG 성능의 약 25~50%를 제한하며, 특히 Gemini-2.5 Flash는 GPT-4 대비 병목률이 약 2배 더 심각함을 의미한다.

5-2 LLM별 비교 분석

두 LLM(GPT-4, Gemini-2.5)의 성능 차이는 전적으로 NL2SPARQL 변환 품질에서 기인한다. Model B-LLM 기준으로 GPT-4는 0.108 F1-Score를 기록하여 Gemini-2.5(0.085) 대비 27.1% 우수한 성능을 보였다. 반면, Oracle(성능 상한선)에서는 Gemini-2.5 대비하여 14.5% 낮았다(0.141 vs 0.165). 이는 Gemini-2.5가 답변 생성 능력은 우수하지만, NL2SPARQL 생성에서는 GPT-4보다 더 잦은 오류를 발생시킴을 의미한다. 결과적으로 병목률 분석 시, GPT-4는 23.4%로 비교적 안정적인 변환 능력을 보인 반면, Gemini-2.5 48.5%의 성능 격차를 보였다.

Ⅵ. 정성적 분석 및 논의

전 장에서 NL2SPARQL 병목(23.4~48.5%)을 정량화하였다. 본 장에서는 이러한 성능 저하를 유발하는 오류 패턴을 정성적으로 분석한다.

6-1 SPARQL 오류 유형 분류

GPT-4와 Gemini-2.5가 생성한 SPARQL을 Golden SPARQL과 비교한 결과, 전체의 89%가 하나 이상의 오류를 포함하고 있었다. 오류는 발생 양상에 따라 속성명 오류(Property Name Error), 구문 오류(Syntax Error), 의미 오류(Semantic Error), 복잡 질의 오류(Complex Query Error)의 네 유형으로 분류되었다. 표 9는 각 오류 유형의 빈도, 발생 메커니즘, 평균 F1-Score를 요약한다.

표 9에서 확인할 수 있듯이, 정상 생성된 SPARQL은 전체의 11%에 불과하며, 오류를 포함하는 쿼리는 평균 F1-Score가 0.02~0.12로 매우 낮다. 특히 복잡 질의 오류(F1=0.02)와 속성명 오류(F1=0.05)가 가장 치명적이다. 각 오류 유형의 특성은 다음과 같다.

표 9. SPARQL 오류 유형별 분석
Table 9. Analysis of SPARQL Error Type

Error Type	Freq.	Mechanism	F1 Avg	Example
Property Name	33%	Schema Mismatch	0.05	hasFreq → hasSustainmentFrequency
Syntax	22%	Grammar Rule Violation	0.12	Missing punctuation Bracket mismatch variable typo
Semantic	16%	Relationship Confusion	0.02	ObjectProperty string comparision error
Complex Query	18%	Grammar Not Supported	0.08	Missing GROUP BY or COUNT()
Correct	11%	No Error	0.85	Matches Golden SPARQL

1) 속성명 오류(Property Name Error, 33%)

가장 빈번한 오류로, LLM이 온톨로지 스키마를 정확히 참조하지 못하고 hasFreq, measureNum 등 축약형 또는 추측형 속성명을 생성하면서 발생한다. 실제 온톨로지는 measureNumber, hasSustainmentFrequency 처럼 명시적 표현을 요구하며, 불일치 시 SPARQL 실행 자체가 불가능해진다. 이는 스키마 기반 제약을 반영하지 못한 전형적 환각(hallucination) 현상이다.

2) 구문 오류(Syntax Error, 22%)

구문 오류는 전체의 22%에서 나타났으며, triple pattern 간 마침표 누락, 괄호 불일치, 변수명 재정의 등 문법적 결함으로 발생하며, 특히 다중 triple 생성을 시도할 때 구조적 일관성이 유지되지 않아 오류가 빈번하게 나타난다.

3) 의미 오류(Semantic Error, 16%)

문법적으로는 유효하지만 관계 방향, 경로 구조, 클래스 타입 선택이 부정확해 의미적 오류를 유발한다. 정방향/역방향 혼동, ObjectProperty 문자열 비교, 중간 클래스 생략 등이 대표 사례이며, 실행 결과가 부정확해 실제 시스템 적용에 큰 위험성을 내포한다.

4) 복잡 질의 오류(Complex Query Error, 18%)

Hard 난이도에서 주로 발생하며, COUNT, GROUP BY, ORDER BY 등 집계 구문 누락이나 다중 조건 처리 실패 등이 주요 원인이다. 예를 들어 "SA2의 분기별 Task 수"와 같은 질의에 대해 단순 SELECT로 회귀하는 오류가 반복되며, Gemini-2.5 Flash는 집계 구조 생성을 거의 수행하지 못했다.

5) 정상 생성(Correct, 11%)

전체의 11%로 대부분 Easy 난이도 질의에 해당한다. 평균 F1-Score는 0.85로 높으며, 4개 이하 triple을 사용하는 단순 경로 질의에서는 정확한 구조가 재현되지만, 질의 복잡도가 높아지면 성능이 급격히 하락하는 경향을 보였다.

Ⅷ. 논의 및 결론

7-1 연구 질문 답변

본 연구는 Oracle 방법론을 통해 KG-RAG의 NL2SPARQL 병목을 정량화하였다. 1장에서 제시한 네 가지 연구 질문에 대한 답변은 다음과 같다.

1) Ontology 기반 KG-RAG의 이론적 성능 상한은 무엇인가? (RQ1)

Oracle KG-RAG는 Vector RAG 대비 GPT-4에서 3.07배, Gemini-2.5에서 5.32배 높은 F1-Score를 기록하여 구조화된 SPARQL 질의의 우수성과 온톨로지 기반 추론의 안정성을 입증하였다.

2) 현실적 구현(LLM 기반 SPARQL 생성)과 Oracle간 성능 격차는 어느 정도인가? (RQ2)

NL2SPARQL 오류로 인해 GPT-4는 23.4%, Gemini-2.5는 48.5%의 성능 손실을 보였다. LLM 기반 KG-RAG는 Oracle 대비 최대 76.6% 수준의 성능만 실현하고 있다.

3) NL2SPARQL 변환 과정에서 발생하는 주요 오류 유형은 무엇이며, 그 비중은 어떠한가? (RQ3)

생성된 SPARQL의 89%에서 오류가 확인되었으며, 주요 유형은 속성명(33%), 구문(22%), 의미(16%), 복잡 질의(18%) 오류로 나타났다. 특히 스키마 불일치와 고급 문법 처리 실패가 성능 저하의 핵심 원인이다.

4) KG-RAG는 벡터 RAG에 비해 실질적 성능 우위를 확보할 수 있는가? (RQ4)

LLM 기반 KG-RAG는 Vector RAG보다 GPT-4 기준 2.35배, Gemini-2.5 기준 2.74배 높은 성능을 보이며, 구조적 접근의 강점을 입증하였다. 그러나 NL2SPARQL의 품질 한계로 인해 실용적 성능 우위 확보에는 제약이 따른다.

7-2 연구의 시사점

본 연구는 KG-RAG의 병목 구조를 정량·정성적으로 규명함으로써, 학술적으로는 평가 프레임워크의 확장, 실무적으로는 질의 난이도 기반 하이브리드 운용 전략, 정책적으로는 NL2SPARQL 품질 관리의 중요성이라는 세 가지 핵심 시사점을 제시한다.

1) 학술적 기여

Oracle 기법을 활용해 구성요소별 병목을 정량화하고, 네 가지 SPARQL 오류 체계를 정립하였다. 각 오류의 발생 원인과 성능 영향도 실증 분석하였다.

2) 실무적 시사점

방산 AI 설계 시 KG-RAG의 잠재성과 NL2SPARQL 병목(23.4~48.5%)을 모두 고려한 성능 목표 설정이 필요하다. Easy 질의에서는 하이브리드 운용 전략이 효과적일 수 있다.

3) 정책적 시사점

CBM+ 도입 시 지식그래프 구축만으로는 충분하지 않으며, NL2SPARQL 품질 관리 체계 마련이 병행되어야 한다. 본 연구의 평가 프레임워크는 국방 AI뿐 아니라 공공영역 전반으로 확장 가능하다.

7-3 연구의 한계 및 향후 연구 방향

본 연구는 Oracle 상한 분석을 통해 KG-RAG 시스템의 구조적 병목을 정량적으로 규명하였으나, 다음과 같은 한계점을 가지며 이를 해결하기 위한 후속 연구가 요구된다.

첫째, 단일 도메인 및 제한된 데이터 규모의 한계이다. 본 연구는 브래들리 전투차량 정비교범(119개 Task, 13,542개 Triple)과 90개의 Golden Query만을 대상으로 평가를 수행하였다. 따라서 수백만 Triple 이상의 대규모 지식그래프나, 여러 교범 간의 상호 참조가 필요한 복잡한 Multi-hop 추론 환경에서의 병목 양상은 검증되지 않았다. 향후 연구에서는 의료·법률·제조 등 이종 도메인으로 대상을 확장하여, 본 연구에서 식별된 병목 패턴의 보편성과 특수성을 검증할 필요가 있다.

둘째, 평가 모델 및 지표의 제한이다. 본 실험은 GPT-4와 Gemini 두 가지 상용 모델에 국한되었으며, 성능 지표 또한 F1-Score에 집중되었다. 향후에는 Llama-3와 같은 고성능 오픈소스 모델이나 Fine-tuned 모델을 비교군에 포함하고, 의미적 유사성(Semantic Similarity), 응답 속도, 비용 효율성 등 다차원적인 성능 평가를 수행해야 한다. 또한, 동일 질의 반복 실행을 통해 생성 결과의 일관성(Consistency)을 검증하는 연구도 병행되어야 할 것이다.

셋째, 병목 진단에 집중하여 개선 방안의 실증 검증이 부족하다. 본 연구는 Oracle 분석을 통해 NL2SPARQL 변환이 23.4~48.5%의 주요 병목임을 정량적으로 규명하였으나, 이를 해소하기 위한 구체적 기술의 실증적 효과 검증은 수행하지 않았다. 병목 원인을 규명하는 진단 단계에 초점을 두었으며, 실제 성능 개선을 위한 기술적 해결 방안의 개발 및 검증은 다음과 같은 후속 연구 과제로 남겨두었다.

첫째, 온톨로지 스키마 정합성을 보장하는 제약 기반 SPARQL 생성 기법이 요구된다. 본 연구에서 가장 빈번했던 속성명 오류가 33%를 해결하기 위해, LLM이 SPARQL을 생성할 때, 스키마 제약을 디코딩 과정에 반영하여 오류를 감소

시킬 필요가 있다.

둘째, 질의 복잡도에 따른 적응적 생성 전략이 필요하다.

본 연구에서 복잡 질의 오류가 18%를 차지했으므로, 다중 조건 처리 및 집계 구문 생성에 특화된 SPARQL 생성 기법의 개발이 요구된다.

셋째, 질의 복잡도 기반 하이브리드 RAG 구조를 고려할 수 있다. Easy 난이도에서는 Vector RAG와 성능 차이가 크지 않으므로, 복잡도 예측 기반 동적 라우팅을 통해 전체 시스템 효율을 최적화하는 방안을 검토할 수 있다. 이러한 접근을 통해 NL2SPARQL 병목을 대폭 감소시키고, Oracle 성능에 근접하는 것을 목표로 한다. 또한, 본 연구는 단일 도메인(군 정비)에 국한되었으므로, 향후 의료·법률·제조 등 이종 도메인으로 확장하여 병목 패턴의 보편성과 특수성을 검증할 필요가 있다.

이러한 후속 연구를 통해 KG-RAG 시스템의 실용화 가능성을 높이고, 고신뢰성이 요구되는 국방·공공 분야에서의 적용 기반을 마련할 수 있을 것으로 기대된다.

참고문헌

[1] Deloitte. 2026 Aerospace and Defense Industry Outlook: Navigating the Intersection of Defense Spending and Commercial Recovery [Internet]. Available: https://www.deloitte.com/us/en/insights/industry/aerospace-defense/aerospace-and-defense-industry-outlook.html.

[2] McKinsey & Company. Aircraft MRO 2.0: The Digital Revolution [Internet]. Available: https://www.mckinsey.com/industries/travel/our-insights/aircraft-mro-2-point-0-the-digital-revolution.

[3] Digital Twin Consortium. Microfactory: Maintenance and Reliability Using Digital Twin Technology Digital Twin Consortium Technology Showcase [Internet]. Available: https://www.digitaltwinconsortium.org/initiatives/technology-showcase/micro-factory/.

[4] M. C. Suárez-Figueroa, A. Gómez-Pérez, and M. Fernández-López, The NeOn Methodology for Ontology Engineering, in *Ontology Engineering in a Networked World*, Berlin, Germany: Springer, pp. 9-34, 2012.

[5] H. S. Jung, "A Study on the Development Direction of Condition-Based Maintenance for Application and Expansion of CBM to Korean Weapon Systems," *Journal of the Korea Academia-Industrial Cooperation Society,* Vol. 22, No. 8, pp. 631-638, August 2021. https://doi.org/10.5762/KAIS.2021.22.8.631

[6] U.S. Army, STP9-91M14-SM-TG: Soldier's Manual and Trainer's Guide for M2/3 Bradley Fighting Vehicle System Maintainer [Internet]. Available: https://asktop.net/army-downloads/references/stp/stp9_91m14/#.

[7] S. Chen, Q. Zhang, Y. Bei, Z. Yuan, H. Zhou, Z. Hong, ... and X. Huang, "A Survey of Graph Retrieval-Augmented Generation for Customized Large Language Models," arXiv:2501.13958, January 2025. https://arxiv.org/abs/2501.13958

[8] H. M. Zahera, M. Ali, M. A. Sherif, D. Moussallem, and A. C. N. Ngomo, "Generating SPARQL from Natural Language Using Chain-of-Thoughts Prompting," in *Proceedings of the 20th International Conference on Semantic Systems SEMANTiCS*, Amsterdam, Netherlands, pp. 353-368, 2024.

[9] S. Min, V. Zhong, R. Socher, and C. Xiong, "Efficient and Robust Question Answering from Minimal Context over Documents," in *Proceedings of the. 56th Annual Meeting of the Association for Computational Linguistics (ACL),* Melbourne, Australia, pp. 1725-1735, July 2018.

[10] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, ... and D. Kiela, "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks," in *Proceedings of the 34th International Conference on Neural Information Processing Systems (NeurIPS)*, Vancouver, Canada, pp. 9459-9474, December 2020.

[11] V. Karpukhin, B. Oğuz, S. Min, P. Lewis, L. Wu, S. Edunov, ... and W. Yih, "Dense Passage Retrieval for Open-Domain Question Answering," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 6769-6781, November 2020.

[12] O. Khattab and M. Zaharia, "ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT," in *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, Virtual Event China, pp. 39-48, July 2020.

[13] W3C SPARQL Working Group. SPARQL 1.1 Query Language [Internet]. Available: https://www.w3.org/TR/sparql11-query/.

[14] D. Edge, H. Trinh, N. Cheng, J. Bradley, A. Chao, A. Mody, ... and J. Larson, "From Local to Global: A Graph RAG Approach to Query-Focused Summarization," arXiv:2404.16130, April 2024. https://arxiv.org/abs/2404.16130

[15] J. Sun, C. Xu, L. Tang, S. Wang, C. Lin, Y. Gong, ... and J. Guo, "Think-on-Graph: Deep and Responsible Reasoning of Large Language Model on Knowledge Graph," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2024.

[16] J. Baek, A. F. Aji, and A. Saffari, "Knowledge-Augmented Language Model Prompting for Zero-Shot Knowledge

Graph Question Answering," in *Proceedings of the 1st Workshop on Natural Language Reasoning and Structured Explanations (NLRSE)*, Toronto, Canada, pp. 78-106, June 2023.

[17] M. Dubey, S. Dasgupta, A. Sharma, K. Höffner, and J. Lehmann, "AskNow: A Framework for Natural Language Query Formalization in SPARQL," in *Proceedings of the 13th International Conference on the Semantic Web. Latest Advances and New Domains (ESWC 2016)*, Heraklion, Crete, Greece, pp. 300-316, May-June 2016.

[18] T. Soru, E. Marx, D. Moussallem, G. Publlo, A. Valdestilhas, D. Esteves, and C. Baron Neto, "SPARQL as a Foreign Language," in *Proceedings of the SEMANTiCS 2017; 13th International Conference on Semantic Systems*, 2017. http://w3id.org/neural-sparql-machines/soru-marx-semantics2017.html

[19] T. Scholak, N. Schucher, and D. Bahdanau, "PICARD: Parsing Incrementally for Constrained Auto-Regressive Decoding from Language Models," in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Online and Punta Cana, Dominican Republic, pp. 9895-9901, November 2021

[20] H. M. Zahera, M. Ali, M. A. Sherif, D. Moussallem, and A. C. N. Ngomo, "Generating SPARQL from Natural Language Using Chain-of-Thoughts Prompting," in *Proceedings of the 20th International Conference on Semantic Systems (SEMANTiCS)*, Amsterdam, Netherlands, pp. 353-368, 2024.

[21] S. Wiseman and A. M. Rush, "Sequence-to-Sequence Learning as Beam-Search Optimization," in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Austin, Texas, pp. 1296-1306, November 2016.

[22] P. Koehn and R. Knowles, "Six Challenges for Neural Machine Translation," in *Proceedings of the First Workshop on Neural Machine Translation*, Vancouver, Canada, pp. 28-39, August 2017.

[23] S. Min, V. Zhong, R. Socher, and C. Xiong, "Efficient and Robust Question Answering from Minimal Context over Documents," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL)*, Melbourne, Australia, pp. 1725-1735, July 2018.

[24] W. G. Cochran, *Sampling Techniques,* 3rd ed. New York, NY: John Wiley & Sons, 1977.

[25] S. L. Lohr, *Sampling: Design and Analysis*, 3rd ed. Boca Raton, FL: CRC Press, 2022.

[26] J. Pérez, M. Arenas, and C. Gutierrez, "Semantics and Complexity of SPARQL," *ACM Transactions on Database Systems*, Vol. 34, No. 3, 16, August 2009.

[27] C. Kosten, P. Cudré-Mauroux, and K. Stockinger, "Spider4SPARQL: A Complex Benchmark for Evaluating Knowledge Graph Question Answering Systems," arXiv:2309.16248, December 2023. https://arxiv.org/abs/2309.16248

김기환(Ki-Hwan Kim)

2011년 : 울산대학교(영문학사)
2024년 : 경상국립대학교 일반대학원
(기술경영공학석사-기술경영학과)

2022년 ~ 현 재: 국방기술진흥연구소 방위산업전략팀 연구원
※관심분야 : 인공지능(AI), 자연어처리(NLP), 온톨로지(Ontology), 지식그래프(KG), 검색증강생성(RAG) 등