Rapporto Tecnico Strategico: Ottimizzazione Architetturale e Roadmap Operativa per SemEval 2026 Task 8 (MTRAGEval)

1. Introduzione e Analisi del Contesto Strategico

1.1 Definizione dell'Obiettivo e Ambito del Task

Il presente rapporto tecnico definisce il piano di esecuzione dettagliato per la partecipazione al **SemEval 2026 Task 8: MTRAGEval (Evaluating Multi-Turn RAG Conversations)**. A differenza delle iterazioni precedenti dei workshop SemEval, che si concentravano prevalentemente su task di classificazione o estrazione di entità isolate, il Task 8 del 2026 rappresenta un punto di svolta critico verso la valutazione di sistemi conversazionali complessi basati su Retrieval-Augmented Generation (RAG).[1] L'obiettivo primario non è la semplice risposta a una domanda (QA), ma la gestione di un dialogo multi-turno in cui il contesto evolve dinamicamente, richiedendo al sistema di mantenere una "memoria di stato" coerente e di discernere quando le informazioni recuperate sono insufficienti, optando per un rifiuto esplicito ("I don't know") piuttosto che per un'allucinazione.[2]

L'analisi dei documenti preliminari e dei benchmark di riferimento, in particolare il dataset **mtRAG**, evidenzia come le architetture RAG tradizionali (Naive RAG) falliscano sistematicamente nel gestire dipendenze contestuali a lungo termine e domande ambigue che richiedono chiarimenti o che non hanno risposta nel corpus fornito.[4] Di conseguenza, la strategia tecnica proposta in questo documento abbandona l'approccio lineare a favore di un'architettura **Agentica Ibrida** implementata tramite **LangGraph**, che integra meccanismi di auto-riflessione (**Self-RAG**) e correzione attiva (**Corrective RAG - CRAG**).[6]

1.2 Risoluzione delle Incongruenze nel Draft Iniziale

Dall'analisi del draft di progetto iniziale fornito e confrontandolo con le specifiche ufficiali di SemEval 2026, emergono e vengono risolte le seguenti incongruenze critiche:

- **Identificazione del Task:** Il draft faceva riferimento ambiguo a task di "Political Question Evasions" o "Everyday Knowledge". È imperativo chiarire che il **Task 8** è specificamente dedicato a **MTRAGEval**. I Task 6 e 7 sono tracce separate con organizzatori e obiettivi distinti.[1] Ogni sforzo ingegneristico deve essere focalizzato esclusivamente sulle metriche e sui domini del benchmark mtRAG.
- **Natura del Dataset:** Il benchmark non è un semplice QA dataset, ma include conversazioni umane estese (media 7.7 turni) su quattro domini specifici: Finanza (FiQA),

Governo (Govt), Documentazione Tecnica (Cloud) e Conoscenza Generale (ClapNQ).[2] Due di questi corpora (Govt e Cloud) sono inediti, il che invalida qualsiasi strategia basata sul pre-training del modello su dati pubblici simili. Il sistema deve fare affidamento su capacità di retrieval robusto piuttosto che sulla conoscenza parametrica del Large Language Model (LLM).[3]

- **Gestione delle Risposte "Non Rispondibili":** Una componente cruciale, spesso trascurata nei piani RAG standard, è la capacità di identificare query non risponbili. Il benchmark mtRAG penalizza severamente le risposte plausibili ma non supportate dai documenti (allucinazioni). La nostra architettura deve includere un nodo esplicito di "Hallucination Grading" calibrato per emettere token di rifiuto (IDK) quando la confidenza del retrieval è sotto soglia.[8]

1.3 Visione Architetturale di Alto Livello

La soluzione proposta si configura come un sistema **Graph-Based Neuro-Symbolic**. Utilizzando LangGraph, modelleremo il flusso conversazionale non come una catena deterministica (Chain), ma come un grafo a stati finiti (State Graph) che permette cicli di feedback. Questo approccio consente al sistema di:

1. **Valutare** la qualità dei documenti recuperati prima della generazione.
2. **Riformulare** la query se il retrieval iniziale è insufficiente.
3. **Correggere** la generazione se vengono rilevate incongruenze fattuali.
4. **Eseguire Fallback** su strategie alternative (es. ricerca web se permessa, o query decomposition) in tempo reale.[10]

---

2. Analisi Approfondita del Benchmark mtRAG e Strategia dei Dati

Il successo nel Task 8 dipende intrinsecamente dalla comprensione profonda dei dati sottostanti. Il benchmark mtRAG è progettato per "rompere" i sistemi RAG attuali esponendo le fragilità nella gestione del contesto e nel recupero di informazioni disperse.

2.1 Caratterizzazione dei Domini e Implicazioni di Retrieval

Il dataset comprende 110 conversazioni umane suddivise in 842 task di valutazione. La diversità dei domini impone strategie di indicizzazione differenziate.[2]

Dominio	Fonte	Caratteristiche Dati	Sfida Tecnica	Strategia di Mitigazione
Finanza	Forum/Repor	Alta densità di	I modelli di	Hybrid Search

(FiQA)	Finanziari	gergo, ragionamento numerico implicito.	embedding standard faticano con la terminologia di nicchia.	(Keyword + Dense) con pesatura BM25 elevata per catturare termini esatti.[2]
Governo (Govt)	Siti web governativi	Struttura gerarchica, linguaggio burocratico, dipendenze legali.	Il chunking "piatto" distrugge il contesto normativo (es. eccezioni alle regole).	Chunking gerarchico (Parent-Child) per preservare il contesto della sezione.[12]
Tecnico (Cloud)	Documentazione IBM	Snippet di codice, versioning, procedure step-by-step.	La formattazione è semantica (indentazione, blocchi di codice).	Pre-processing Markdown-aware che mantiene intatti i blocchi di codice.[13]
Generale (ClapNQ)	Wikipedia	Testo narrativo, enciclopedico, ampia varietà di argomenti.	Ambiguità delle entità e vastità dello spazio di ricerca.	Dense Retrieval con reranking aggressivo per filtrare il rumore.[14]

L'analisi evidenzia che una pipeline di ingestione monolitica è destinata a sottoperformare. La roadmap operativa (Sezione 6) prevede una fase specifica di **ingestione multi-pipeline**, dove ogni corpus viene trattato con una strategia di chunking ottimizzata per la sua struttura intrinseca.

2.2 Strategie Avanzate di Chunking

Per i domini complessi come Govt e Cloud, il semplice fixed-size chunking (es. 512 token con overlap) è deleterio. Interrompere una frase legale o uno script di codice a metà rende il segmento inutile per il retrieval semantico.
Adotteremo una strategia di Semantic Hierarchical Chunking 15:

1. **Segmentazione Logica:** Utilizzo di parser basati su NLP (es. Unstructured.io o librerie custom) per identificare confini naturali come intestazioni, paragrafi e cambi di

argomento.
2. **Parent-Child Indexing:** Creazione di "Parent Chunks" di grandi dimensioni (es. 1000-2000 token) che contengono il contesto completo. Questi vengono suddivisi in "Child Chunks" più piccoli (es. 256-512 token) per l'embedding e la ricerca.
3. **Retrieval Window:** Durante la ricerca, il sistema calcola la similarità sui *Child Chunks*, ma restituisce al LLM il *Parent Chunk* corrispondente. Questo garantisce precisione nel puntamento (grazie ai chunk piccoli) e ricchezza di contesto per la generazione (grazie ai chunk grandi).[12]

2.3 Gestione delle Domande "Non Rispondibili" (IDK)

Il benchmark mtRAG include domande progettate per essere parzialmente o totalmente prive di risposta nel contesto fornito. I modelli LLM attuali soffrono di "horror vacui" e tendono ad inventare risposte pur di compiacere l'utente.
Per mitigare questo rischio, implementeremo un meccanismo di Confidence Calibration 8:

- Analisi della distribuzione dei punteggi di similarità (cosine similarity) del *retriever* e dei punteggi di confidenza del *reranker*.
- Definizione di una soglia dinamica τ. Se il punteggio del miglior documento recuperato è $score < \tau$, il sistema bypassa il generatore e restituisce direttamente il token di rifiuto.
- Questa soglia sarà calibrata empiricamente sul subset di validazione fornito dagli organizzatori o generato sinteticamente.[9]

---

3. Architettura Tecnica: Implementazione LangGraph Ottimizzata

L'architettura proposta fonde i paradigmi di **Self-RAG** e **Corrective RAG (CRAG)** in un unico grafo coerente gestito da LangGraph. Questa struttura permette di superare i limiti delle catene sequenziali, introducendo nodi decisionali che agiscono come "agenti di controllo qualità" in tempo reale.

3.1 Definizione dello Stato del Grafo (GraphState)

Il cuore del sistema è l'oggetto GraphState, un dizionario tipizzato che viene passato e mutato attraverso i nodi. La sua definizione ottimizzata per MTRAGEval è la seguente:

Python

```
from typing import TypedDict, List
```

```
class GraphState(TypedDict):
    question: str               # La domanda corrente dell'utente
    chat_history: List[str]     # Storico della conversazione (fondamentale per mtRAG)
    generation: str             # La risposta generata dall'LLM
    documents: List[str]        # Lista dei contesti recuperati
    relevance_score: str        # Giudizio del Grader ('yes', 'no')
    hallucination_score: str    # Giudizio anti-allucinazione ('grounded', 'not_grounded')
    answer_score: str           # Giudizio di utilità della risposta ('useful', 'not_useful')
    retry_count: int            # Contatore per evitare loop infiniti
    search_needed: bool         # Flag per attivare fallback (Web Search o Rewrite)
```

Questa struttura dati permette a ogni nodo di avere visibilità completa sul processo, abilitando decisioni complesse basate non solo sull'ultimo input, ma sull'intera storia dell'interazione.[10]

3.2 Descrizione Dettagliata dei Nodi Funzionali

Nodo 1: Contextual Query Rewriter (Guardiano dell'Intento)

In un contesto multi-turno, le domande dell'utente sono spesso ellittiche (es. "E rispetto a quello precedente?"). Questo nodo utilizza l'LLM per analizzare la chat_history e l'ultima question e produrre una **Standalone Question** de-contestualizzata.

- **Ottimizzazione:** Implementeremo una logica di "Query Expansion".[17] Invece di generare una sola query, il nodo ne genererà 3 varianti (es. una specifica per keyword, una semantica, una ipotetica) per massimizzare il recall nella fase successiva.
- **Prompting:** Il prompt includerà esempi few-shot tratti dal dominio specifico (es. FiQA) per insegnare al modello come risolvere le co-referenze nel gergo finanziario.[5]

Nodo 2: Hybrid Retrieval Engine

Questo nodo esegue la ricerca effettiva sul Vector Store (es. ChromaDB o Elasticsearch).

- **Logica Ibrida:** Combina i risultati di una ricerca vettoriale densa (Embedding BGE-M3) con una ricerca sparsa (BM25). I risultati vengono fusi utilizzando l'algoritmo **Reciprocal Rank Fusion (RRF)**.
- **Filtro Metadati:** Utilizza i metadati dei chunk per restringere la ricerca al dominio pertinente se inferibile dalla domanda (es. se la domanda riguarda "tasse", priorità al corpus *Govt*).[2]

Nodo 3: Relevance Grader (Il Critico CRAG)

Ispirato al paper CRAG [7], questo nodo agisce come un filtro di qualità. Un LLM leggero (es. Granite-3-8B o un modello SLM quantizzato per velocità) valuta la rilevanza di ogni documento recuperato rispetto alla domanda.

- **Output:** Assegna un punteggio binario ("relevant" / "irrelevant") a ciascun documento.

- **Logica Decisionale:** Se la percentuale di documenti rilevanti è inferiore a una soglia critica (es. 20%), imposta il flag search_needed = True nello stato, attivando il ramo correttivo.[10]

Nodo 4: Fallback & Knowledge Refinement (Azione Correttiva)

Se attivato dal Grader, questo nodo tenta di recuperare informazioni migliori.

- **Scenario A (Web Search Permessa):** Esegue una ricerca web tramite API (es. Tavily) per trovare informazioni aggiornate non presenti nel corpus statico.
- **Scenario B (Web Search Vietata - Probabile per Task 8):** Esegue una "Query Decomposition". La domanda complessa viene spezzata in sotto-domande più semplici che vengono rieseguite sul Vector Store. Questo approccio è spesso sufficiente a sbloccare documenti che una query complessa aveva mancato.[18]

Nodo 5: Generator (Sintesi della Risposta)

Utilizza i documenti filtrati (e potenzialmente integrati dal fallback) per generare la risposta finale.

- **Prompt Engineering:** Il system prompt deve imporre rigorosamente la citazione delle fonti (se richiesto) e l'adozione di un tono neutro. Cruciale è l'istruzione negativa: "Se le informazioni non sono presenti nei documenti forniti, rispondi esclusivamente con 'Non ho informazioni sufficienti per rispondere'".[8]

Nodo 6: Hallucination & Answer Grader (Il Critico Self-RAG)

Questo nodo chiude il ciclo di feedback.[19] Valuta la generazione su due assi:

1. **Faithfulness (Fedeltà):** La risposta contiene informazioni non presenti nei documenti di contesto? (Sì/No). Se "No" (Allucinazione), il sistema incrementa retry_count e torna al nodo *Generator* con istruzioni di correzione.
2. **Answerability (Pertinenza):** La risposta risolve effettivamente la domanda dell'utente? Se "No", il sistema potrebbe tentare una nuova *Query Rewrite*.

3.3 Flusso del Grafo e Archi Condizionali

La potenza di LangGraph risiede nella logica degli archi (Edges) che connettono questi nodi.

- **Start** $\rightarrow$ rewrite_query
- rewrite_query $\rightarrow$ retrieve
- retrieve $\rightarrow$ grade_documents
- **Conditional Edge (Decisione CRAG):**
  - Se relevance è alta $\rightarrow$ generate
  - Se relevance è bassa $\rightarrow$ transform_query (Fallback)
- transform_query $\rightarrow$ retrieve (Loop di miglioramento)
- generate $\rightarrow$ grade_generation

- **Conditional Edge (Decisione Self-RAG):**
  - Se grounded AND useful $\rightarrow$ **End**
  - Se not grounded (Allucinazione) $\rightarrow$ generate (Retry)
  - Se not useful (Fuori tema) $\rightarrow$ rewrite_query (Ricomincia il ciclo)
  - **Safety Break:** Se retry_count > 3, forza l'uscita con una risposta di fallback ("I don't know") per evitare loop infiniti e latenza eccessiva.[20]

---

4. Specifiche Tecniche dei Componenti di Retrieval

Per competere ai massimi livelli nel Task 8, la scelta dei modelli di embedding e reranking è determinante. I benchmark recenti mostrano che la semplice similarità vettoriale non è sufficiente per cogliere le sfumature di query complesse.

4.1 Embedding Model: BGE-M3

Selezioniamo il modello **BAAI/bge-m3** come spina dorsale del sistema di retrieval.

- **Motivazione:** BGE-M3 supporta contesti lunghi (fino a 8192 token), è multilingua (importante se il task ha sfumature linguistiche, anche se il corpus è prevalentemente inglese), e offre una rappresentazione densa, sparsa e multi-vettoriale (ColBERT-style) simultaneamente.[14]
- **Implementazione:** Utilizzeremo la modalità ibrida nativa di BGE-M3, che calcola sia lo score denso che quello sparso (lessicale) in un unico passaggio, eliminando la necessità di gestire un indice BM25 separato e semplificando l'infrastruttura.

4.2 Reranker: BGE-Reranker-v2-M3 vs. Qwen

Dopo la fase di retrieval iniziale (che potrebbe restituire 50-100 documenti candidati), è essenziale applicare un modello di *Cross-Encoder* per riordinare i risultati con maggiore precisione.

- **Selezione: BAAI/bge-reranker-v2-m3** è attualmente lo stato dell'arte (SOTA) per il reranking leggero.[22] Tuttavia, monitoreremo anche le prestazioni di **Qwen-Reranker**, che ha mostrato risultati promettenti in contesti conversazionali.
- **Configurazione:** Il reranker prenderà in input la coppia (Query Espansa, Documento) e restituirà uno score di rilevanza. Solo i top-K documenti (dove K=5 o 10, da ottimizzare) saranno passati al generatore. Questo passaggio è computazionalmente costoso ma fondamentale per aumentare la metrica NDCG@10 e ridurre il rumore nel contesto del LLM.[24]

4.3 Database Vettoriale e Infrastruttura

Per supportare le operazioni di LangGraph e Hybrid Search, l'infrastruttura dati sarà basata su:

- **Vector Store: ChromaDB** o **Qdrant**. Qdrant è preferibile per la sua gestione nativa del filtraggio dei metadati e per le prestazioni scalabili con HNSW index.
- **Orchestrazione:** L'intero stack sarà containerizzato via Docker per garantire la riproducibilità, requisito essenziale per la sottomissione a SemEval.

---

5. Strategie di Generazione e Mitigazione delle Allucinazioni

La fase di generazione non è un processo passivo. L'LLM deve agire come un analista critico delle informazioni fornite.

5.1 Chain of Verification (CoVe)

Per ridurre le allucinazioni, implementeremo una variante semplificata della **Chain of Verification (CoVe)** [25] integrata nel nodo di generazione.

1. **Generazione Baseline:** L'LLM genera una risposta preliminare basata sui documenti.
2. **Verifica dei Fatti:** L'LLM estrae le affermazioni fattuali dalla risposta e verifica se ciascuna è supportata da una specifica frase nei documenti recuperati (Citation Span).
3. **Raffinamento:** Se un'affermazione non è supportata, viene rimossa o corretta. Questo processo avviene all'interno di una singola chiamata LLM (usando tecniche di prompting avanzato come *Chain-of-Thought*) per minimizzare la latenza, oppure in un nodo separato se la precisione è prioritaria sulla velocità.

5.2 Rilevamento e Gestione dell'Incertezza (IDK)

Come anticipato, la capacità di dire "non lo so" è critica. Oltre alla calibrazione della soglia di retrieval, implementeremo un controllo semantico post-generazione.

- **Metodo:** Se l'LLM genera frasi evasive ("non è chiaro dal testo", "il documento non specifica"), un classificatore basato su regole o un piccolo modello BERT etichetterà la risposta come "Unanswerable".
- **Ottimizzazione:** Questo segnale sarà usato per uniformare l'output nel formato standard richiesto dal task (es. restituire stringa vuota o token specifico ``).[8]

---

6. Piano Operativo e Roadmap: 30 Giorni / 5 Persone

La seguente roadmap è progettata per un team di 5 specialisti:

- **Lead Architect (LA):** Orchestrazione LangGraph, gestione dello stato, integrazione.
- **Data Engineer (DE):** Ingestione dati, pipeline di chunking, gestione Vector DB.
- **Retrieval Specialist (RS):** Ottimizzazione embedding, reranking, hybrid search.
- **GenAI Engineer (GE):** Prompt engineering, CoVe, gestione IDK.

- **QA & Eval Engineer (QE):** Pipeline di valutazione (Ragas), dati sintetici, analisi errori.

Settimana 1: Fondamenta e Baseline (Giorni 1-7)

L'obiettivo è avere una pipeline "End-to-End" funzionante, anche se con prestazioni non ottimali.

Ruolo	Attività Principali	Output Atteso
LA	Setup repo Git, Docker, scaffolding LangGraph (nodi vuoti).	Repo funzionante con Hello World LangGraph.
DE	Download mtRAG, analisi esplorativa, setup Qdrant. Chunking base.	Dati indicizzati (Naive Chunking).
RS	Implementazione pipeline BM25 + BGE-M3 base.	Funzione retrieve(query) funzionante.
GE	Prompt template base per generazione e rewrite.	Prompt testati su Playground.
QE	Setup Ragas/DeepEval. Selezione "Golden Set" di validazione.	Pipeline di eval configurata.

Settimana 2: Architettura Avanzata e Specializzazione (Giorni 8-14)

Implementazione della logica "Self-CRAG" e ottimizzazione dei dati.

Ruolo	Attività Principali	Output Atteso
LA	Collegamento nodi LangGraph, implementazione logica condizionale (Loop).	Grafo Self-CRAG funzionante.
DE	Implementazione **Parent-Child Chunking** e	Indice vettoriale v2

	metadati di dominio.	(Ottimizzato).
RS	Integrazione **Reranker** (BGE-Reranker-v2-M3). Tuning Top-K.	Miglioramento NDCG@10 rilevabile.
GE	Sviluppo nodi **Grader** (Relevance & Hallucination).	Nodi di controllo qualità attivi.
QE	Generazione dati sintetici per edge cases (Allucinazioni).	Dataset di test arricchito.

Settimana 3: Ottimizzazione e Calibrazione (Giorni 15-21)

Il focus si sposta sulla precisione e sulla gestione dei casi limite (IDK).

Ruolo	**Attività Principali**	**Output Atteso**
LA	Tuning dei timeout e dei retry loops. Ottimizzazione latenza.	Sistema stabile e reattivo.
DE	Pulizia dati specifica per domini Govt/Cloud (regex custom).	Indice v3 (Cleaned).
RS	Sperimentazione con **Query Decomposition** per domande complesse.	Strategia di fallback avanzata.
GE	Calibrazione soglie IDK. Tuning prompt "Chain of Verification".	Riduzione allucinazioni sul dev set.
QE	Esecuzione benchmark completi. Analisi errori per	Report dettagliato sulle debolezze.

	dominio.	

Settimana 4: Finalizzazione e Sottomissione (Giorni 22-30)

Freeze del codice, stress test e preparazione deliverable.

Ruolo	Attività Principali	Output Atteso
LA	Codice finale, documentazione tecnica, API wrapper.	Submission package pronto.
DE	Backup indici, script di riproduzione ingestione.	Data pipeline riproducibile.
RS	Grid search finale sui parametri (K, alpha ibrido, soglie).	Configurazione ottimale "frozen".
GE	Validazione finale output generation (formattazione, citazioni).	Output conforme alle regole Task 8.
QE	Validazione finale su set di test nascosto (se disp) o holdout.	Score finale stimato.

---

7. Valutazione e Quality Assurance (QA)

Per garantire la competitività della soluzione, non possiamo affidarci solo alla valutazione finale degli organizzatori. È necessario un framework di valutazione interna continua.

7.1 Framework di Valutazione Automatizzata

Utilizzeremo DeepEval in combinazione con metriche Ragas per monitorare i progressi giornalieri.27
Le metriche chiave da monitorare sono:

1. **Faithfulness (Ragas):** Misura quanto la risposta è aderente ai documenti recuperati. Essenziale per il controllo allucinazioni.

2. **Context Precision:** Capacità del retriever di posizionare i documenti rilevanti nei top-K.
3. **Answer Relevancy:** Quanto la risposta è pertinente alla domanda originale.
4. Refusal Accuracy (Custom): Una metrica personalizzata definita come:

   $$\text{Refusal Accuracy} = \frac{\text{True Positives (IDK)}}{\text{Total Unanswerable Queries}}$$

   Questa metrica è vitale per il Task 8.9

7.2 Generazione Dati Sintetici

Dato che 110 conversazioni potrebbero non essere sufficienti per un tuning robusto, utilizzeremo modelli come GPT-4o per generare **conversazioni sintetiche multi-turno** a partire dai documenti dei corpora.

- **Tecnica:** "Evol-Instruct". Si parte da un documento, si genera una domanda semplice, poi si chiede al modello di renderla progressivamente più complessa o dipendente da un contesto precedente immaginario.
- **Scopo:** Creare un dataset di "Hard Negatives" (domande che sembrano pertinenti ma non hanno risposta nel chunk specifico) per addestrare i nodi Grader a essere più discriminativi.[29]

---

8. Conclusioni e Raccomandazioni Strategiche

La partecipazione al SemEval 2026 Task 8 richiede un cambio di paradigma rispetto ai sistemi QA tradizionali. L'architettura presentata in questo rapporto non si limita a "cercare e rispondere", ma implementa un comportamento cognitivo di **auto-critica e correzione attiva**.

L'adozione di **LangGraph** come orchestratore permette di gestire la complessità dello stato conversazionale, trasformando le debolezze tipiche dei LLM (allucinazioni, perdita di contesto) in punti decisionali gestibili. La combinazione di **Self-RAG** per la qualità della generazione e **CRAG** per la robustezza del retrieval, supportata da una strategia di dati gerarchica e da un reranking allo stato dell'arte, posiziona il team in modo competitivo per la sfida.

La roadmap di 30 giorni è aggressiva ma strutturata per mitigare i rischi tecnici nelle prime fasi, lasciando spazio sufficiente per il tuning fine delle soglie decisionali, che sarà, in ultima analisi, il fattore discriminante per la vittoria nel benchmark mtRAG.

Fonti Utilizzate

Bibliografia

1. SemEval-2026 Tasks | SemEval-2026 - GitHub Pages, accesso eseguito il giorno dicembre 2, 2025, https://semeval.github.io/SemEval2026/tasks.html
2. MTRAG: Multi-Turn RAG Benchmark - GitHub, accesso eseguito il giorno dicembre 2, 2025, https://github.com/IBM/mt-rag-benchmark
3. mtRAG: A Multi-Turn Conversational Benchmark for Evaluating Retrieval-Augmented Generation Systems | Transactions of the Association for Computational Linguistics - MIT Press Direct, accesso eseguito il giorno dicembre 2, 2025, https://direct.mit.edu/tacl/article/doi/10.1162/TACL.a.19/132114/mtRAG-A-Multi-Turn-Conversational-Benchmark-for
4. A benchmark for evaluating conversational RAG - IBM Research, accesso eseguito il giorno dicembre 2, 2025, https://research.ibm.com/blog/conversational-RAG-benchmark
5. Learning When to Retrieve, What to Rewrite, and How to Respond in Conversational QA, accesso eseguito il giorno dicembre 2, 2025, https://arxiv.org/html/2409.15515v1
6. Self-reflective RAG with LangGraph: Self-RAG and CRAG - YouTube, accesso eseguito il giorno dicembre 2, 2025, https://www.youtube.com/watch?v=pbAd8O1Lvm4
7. Self-Reflective RAG with LangGraph - LangChain Blog, accesso eseguito il giorno dicembre 2, 2025, https://blog.langchain.com/agentic-rag-with-langgraph/
8. Detect hallucinations for RAG-based systems | Artificial Intelligence - AWS, accesso eseguito il giorno dicembre 2, 2025, https://aws.amazon.com/blogs/machine-learning/detect-hallucinations-for-rag-based-systems/
9. A Library of LLM Intrinsics for Retrieval-Augmented Generation - arXiv, accesso eseguito il giorno dicembre 2, 2025, https://arxiv.org/html/2504.11704v1
10. Build a Corrective RAG Chatbot with LangGraph | by Shwet Prakash ..., accesso eseguito il giorno dicembre 2, 2025, https://medium.com/@shwet.prakash97/build-a-corrective-rag-chatbot-with-langgraph-ee4cd4cf7144
11. Building an Effective RAG Pipeline: A Guide to Integrating Self-RAG, Corrective RAG, and Adaptive RAG | by kirouane Ayoub | GoPenAI, accesso eseguito il giorno dicembre 2, 2025, https://blog.gopenai.com/building-an-effective-rag-pipeline-a-guide-to-integrating-self-rag-corrective-rag-and-adaptive-ab7767f8ead1
12. Chunking Strategies for AI and RAG Applications - DataCamp, accesso eseguito il giorno dicembre 2, 2025, https://www.datacamp.com/blog/chunking-strategies
13. The Evolution of RAG Text Chunking: Why Precision Still Matters | by Tao An - Medium, accesso eseguito il giorno dicembre 2, 2025, https://tao-hpu.medium.com/the-evolution-of-rag-text-chunking-why-precision-

still-matters-c3e35ef79c50
14. bge-reranker-v2-m3 | AI Model Details - AIModels.fyi, accesso eseguito il giorno dicembre 2, 2025, https://www.aimodels.fyi/models/replicate/bge-reranker-v2-m3-yxzwayne
15. Advanced RAG Techniques | StackAI, accesso eseguito il giorno dicembre 2, 2025, https://www.stack-ai.com/blog/advanced-rag-techniques
16. 8 Types of Chunking for RAG Systems - Analytics Vidhya, accesso eseguito il giorno dicembre 2, 2025, https://www.analyticsvidhya.com/blog/2025/02/types-of-chunking-for-rag-systems/
17. Multi-turn Retrieval-Augmented Generation - Emergent Mind, accesso eseguito il giorno dicembre 2, 2025, https://www.emergentmind.com/topics/multi-turn-retrieval-augmented-generation-rag
18. Build a corrective RAG agent by using IBM Granite and Tavily, accesso eseguito il giorno dicembre 2, 2025, https://www.ibm.com/think/tutorials/build-corrective-rag-agent-granite-tavily
19. Built with LangGraph! #21: Self-RAG | by Okan Yenigün | Towards Dev - Medium, accesso eseguito il giorno dicembre 2, 2025, https://medium.com/@okanyenigun/built-with-langgraph-21-self-rag-381ab952da6b
20. Self-Rag: A Guide With LangGraph Implementation | DataCamp, accesso eseguito il giorno dicembre 2, 2025, https://www.datacamp.com/tutorial/self-rag
21. vbarda/pandas-rag-langgraph - GitHub, accesso eseguito il giorno dicembre 2, 2025, https://github.com/vbarda/pandas-rag-langgraph
22. QuantFactory/Qwen3-Reranker-0.6B-GGUF - Hugging Face, accesso eseguito il giorno dicembre 2, 2025, https://huggingface.co/QuantFactory/Qwen3-Reranker-0.6B-GGUF
23. Ultimate Guide - The Most Accurate Reranker for Real-Time Search in 2025 - SiliconFlow, accesso eseguito il giorno dicembre 2, 2025, https://www.siliconflow.com/articles/en/most-accurate-reranker-for-real-time-search
24. MAIR: A Massive Benchmark for Evaluating Instructed Retrieval - ACL Anthology, accesso eseguito il giorno dicembre 2, 2025, https://aclanthology.org/2024.emnlp-main.778.pdf
25. Chain-of-Verification (CoVe): Reduce LLM Hallucinations - Learn Prompting, accesso eseguito il giorno dicembre 2, 2025, https://learnprompting.org/docs/advanced/self_criticism/chain_of_verification
26. CodeHalu: Investigating Code Hallucinations in LLMs via Execution-based Verification, accesso eseguito il giorno dicembre 2, 2025, https://arxiv.org/html/2405.00253v3
27. DeepEval vs Ragas | DeepEval - The Open-Source LLM Evaluation Framework, accesso eseguito il giorno dicembre 2, 2025, https://deepeval.com/blog/deepeval-vs-ragas
28. The 5 best RAG evaluation tools in 2025 - Articles - Braintrust, accesso eseguito il

giorno dicembre 2, 2025, https://www.braintrust.dev/articles/best-rag-evaluation-tools
29. How we Moved from LLM Scorers to Agentic Evals? - Research AIMultiple, accesso eseguito il giorno dicembre 2, 2025, https://research.aimultiple.com/agentic-evals/
30. Chunking Strategies - Salesforce Help, accesso eseguito il giorno dicembre 2, 2025, https://help.salesforce.com/s/articleView?id=data.c360_a_search_index_supported_chunking_strategies.htm&language=en_US&type=5
31. mtRAG: A Multi-Turn Conversational Benchmark for Evaluating Retrieval-Augmented Generation Systems - arXiv, accesso eseguito il giorno dicembre 2, 2025, https://arxiv.org/html/2501.03468v1