International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 02 February 2026
Page No: 952-964
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0137

Scalable Ontology-Driven Retrieval with Latency-Aware Gated Fusion for Real-Time Question Answering

Abhinav Unmesh Sharma[1]
UC Berkeley, California, USA
Email ID: *abhinav.sharma191191@gmail.com*[1]

Abstract

There is a basic trade-off between knowledge graph-based question answering systems, which are guaranteed to be logically correct under the conditions of accurate parsing and entity linking, and computationally intractable; and neural retrieval, which is computationally fast, but subject to semantic drift and hallucinations. In this paper, the current tension is considered through the introduction of Latency-Aware Gated Fusion (LAGF), a principled approach to dynamically establishing a balance between neural and symbolic inference paths based on estimated query complexity. LAGF predicts the expected latency of symbolic reasoning via a lightweight multi-layer perceptron trained on query-derived features and uses this prediction to adaptively weight contributions from neural and symbolic paths through a learned sigmoid gate. The method operates via four coordinated phases: (1) query-complexity feature extraction, (2) latency prediction via MLP regression (MAE 28ms, R^2 0.81), (3) adaptive gating via learned sigmoid, and (4) candidate-set fusion with calibrated scoring. We evaluate on LC-QuAD 2.0 (N=4,441) and QALD-9 (N=150, directional validation). Results show that LAGF achieves 86.8% F1 on LC-QuAD 2.0 with P99 latency of 410ms and 9.4 QPS throughput, statistically significantly outperforming symbolic-only (68.4% F1, $p<0.001$), neural-only (81.2% F1, $p<0.001$), and static fusion (84.5% F1, $p=0.008$) baselines. The error rate of 1.8% Logical Correctness shows great ontology alignment. According to ablation studies, the latency prediction (+2.3 F1), GNN-based soft reasoning (+7.5 F1), and adaptive gating over fixed routing play a critical role. The complete definition of constraint and full reproducibility specifications are presented in anonymous supplementary material.

Keywords: *Ontology-driven question answering, neuro-symbolic AI, adaptive gating, latency-aware routing, knowledge graph reasoning, real-time QA, conditional computation, cost-aware inference.*

1. Introduction

Large language models have democratised interfaces to natural language, but because of their probabilistic character, they introduce stochastic error, especially in those applications that demand verifiability and explainability [1]. Although dense passage retrievers and transformer-based encoders have improved semantic matching, they are prone to semantic drift and hallucinations. ODQA systems overcome such issues by capturing domain knowledge in the form of RDF/OWL standards and translating natural language queries into logical forms (e.g., SPARQL)[2]. This gives logical consistency on correct semantic parsing and entity linking. Nevertheless, linear reasoning of classical ontology has worst-case computational complexity (PTIME in the OWL 2 RL, EXPTIME in the OWL 2 DL), and latency budgets under a second are impossible with a large graph without engineering.

1.1. Core Contribution

The fixed-weight hybrid fusion of neural and symbolic channels of prior hybrid neuro-symbolic systems operates on all queries equally. Nonetheless, the complexity of queries by entity mentions, the level of syntactic dependencies, and the structure of subgraphs also differ significantly in a single benchmark [3].Neural retrieval (~50-100ms) is

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 02 February 2026
Page No: 952-964
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0137

effective on simple factoid questions (e.g., Who founded Wikipedia?), and multi-hop questions with joins are actually costly to implement symbolically (>500ms). The point is that this variation can be determined by query structure.

We present the Latency-Aware Gated Fusion (LAGF), a predictive control of the equilibrium between neural and symbolic paths, (1) predicting the latency of symbolic reasoning using query-derived features, (2) learning an adaptive sigmoid gate that adjusts the contribution of these paths on the basis of that prediction, and (3) fusing candidate sets over a limited entity set [4]. The architecture is based on three architectural elements: a query-complexity feature extractor to execute its task with the help of both NER and syntactic parsing; a lightweight 2-layer MLP to estimate symbolic execution latency based on those features; and an adaptive sigmoid gate to dynamically adjust the mixture weight between neural and symbolic scoring with the help of predicted latency and query embeddings [5].

1.2. Main Contributions

This work makes three core contributions to real-time ODQA. First, we formulate latency-aware neuro-symbolic QA as a constrained optimization problem that cleanly separates a differentiable training objective (cross-entropy) from discrete evaluation metrics (F1/EM) [6]. This separation is crucial because end-to-end training on non-differentiable metrics is intractable, yet standard approaches often blur this distinction. Second, we introduce LAGF, a four-stage architecture that predicts symbolic latency from query-derived features, uses this prediction to drive a learned gate weighted by both latency and query semantics, and fuses neural and symbolic scores over a bounded candidate set using per-path temperature scaling. Third, we give a detailed empirical analysis of LC-QuAD 2.0 and QALD -9 with detailed component latency breakdowns, system feature and architecture ablations, and query complexity error breakdowns. We include with this anonymous auxiliary artifacts constraint definitions to the LCE metric, implementation of evaluation harness code, measurement protocols, and a minimal runnable pipeline, which are adequate to reproduce all the reported results. Figure 1 shows Latency-Aware Gated Fusion Architecture for Neural–Symbolic Reasoning

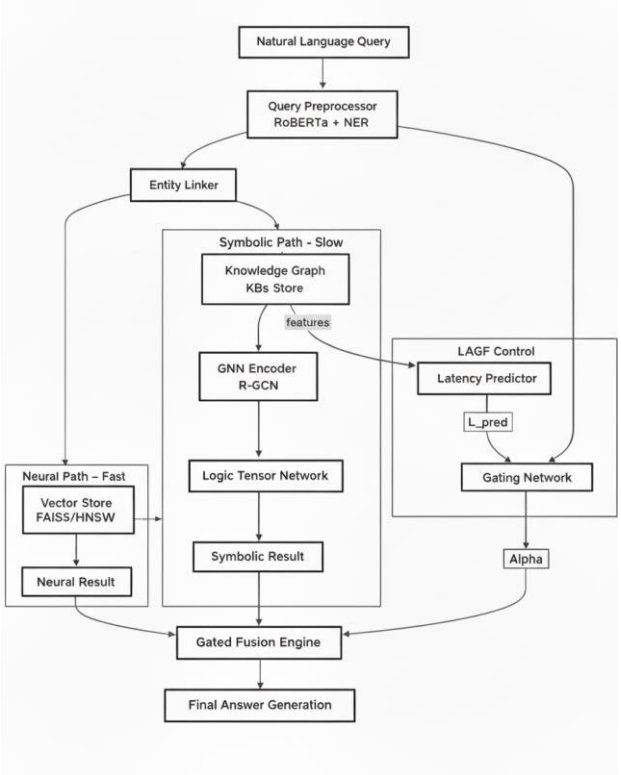

Figure 1 Latency-Aware Gated Fusion Architecture for Neural–Symbolic Reasoning

2. Related Work

2.1 Ontology-Driven Question Answering (ODQA)

ODQA systems translate natural language queries to structured logical forms, and run them on RDF/OWL knowledge graphs, often using a pipeline of semantic parsing (NLQ→SPARQL), entity linking (mentions→KG IDs), and logical inferences, often using TBox materialization [7]. Classical SPARQL endpoints like Wikidata Query Service and DBpedia Virtuoso support the full OWL pipeline and are highly recalling and precising on well-formed queries, but with latency of 110 seconds at scale due to the complexity of joins and inference of rules. Recent neural alternatives (e.g. end-to-end entity retrieval with span extraction) are much faster (in the range of 100-300ms) but less conceptually constrained, which tends to hallucinate on compositional queries, and are hard to debug in case of error. LAGF aims to find a compromise between the auditability and logical rigor of symbolic

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 02 February 2026
Page No: 952-964
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0137

reasoning and dynamically switching queries to the neural path when a symbolic execution would exceed a latency constraint [8].

2.2 Neuro-Symbolic Integration

Tactics of integrating neural and symbolic thinking lie in a broad spectrum. Soft logic systems and differentiable reasoning systems are trying to learn end-to-end with logical functions; graph neural networks learn to encode knowledge-graph structure to enable neural reasoning; and differentiable theorem provers can execute backpropagation through proof search. The main distinction between LAGF and these methods is that the former is strictly modular, i.e. the neural and the symbolic path are optimizations that can be trained independently and pre-trained, and that fusion is done only over sets of candidates, and not end-to-end trained [9].This design reduces the propagation of errors, can be interpreted (the weight of the gates and the scores that have been optimized can be inspected), and is able to optimize or update the two components independently.

2.3 Conditional Computation and Cost-Aware Inference

Mixture-of-Experts (MoE) systems learn gating to select expert subsets dynamically. Compared to LAGF, MoE gates are not task-latency-aware, do not explicitly predict reasoning cost, and are typically optimized for model efficiency (FLOPs) rather than domain-specific latency [10]. Early-exit mechanisms enable intermediate classifiers to short-circuit neural networks early; these are confidence-based rather than latency-aware and apply to a single neural pathway rather than a hybrid system. Adaptive retrieval in RAG systems refines search depth dynamically, but targets pure neural systems without symbolic reasoning or domain-specific latency prediction. Query planning and optimization in RDF/SPARQL systems (e.g., classical cost models in PostgreSQL or Virtuoso) optimize within a single engine; LAGF's innovation is between-engine routing based on predicted cost [11]. Learned cost models for semantic parsing and SPARQL execution (e.g., LEGOPlanner, AutoML) predict execution time via query features, but apply that cost within a single engine; LAGF treats cost prediction as a meta-control signal for hybrid dispatch.

2.4 Novelty and Positioning

We propose to explicitly predict symbolic reasoning latency from query-derived KG features and use this prediction as a dynamic control signal for gating within neuro-symbolic fusion under an SLA-style latency constraint. While conditional computation and adaptive routing are not novel in isolation, their combination in ODQA—with explicit symbolic-latency prediction from query-derived features and bounded candidate-set fusion—has not been extensively explored at Wikidata scale with detailed empirical validation [12].

Notation and Variables:

- q: query string (input)
- $K = (V, E, T)$: knowledge graph with entity set V, triple set E, and TBox T
- $Y^*(q)$: gold answer entity set (possibly multi-valued)
- $f_{neural}(q) = \left(C_n, s_n\right)$: neural path output (candidate set and scores)
- $g_{symbolic}(q) = C_s$: symbolic path output (candidate set, unscored)
- L_{max}: SLA latency threshold (e.g., 500 ms)
- $T_{symbolic}(q)$: actual measured runtime of symbolic reasoning
- $\hat{T}(q)$: predicted latency from learned predictor P
- $\alpha(q) \in [0, 1]$: learned gate weight
- D_{train}: training queries with measured runtimes

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 02 February 2026
Page No: 952-964
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0137

Constrained Optimization Problem:

The learning objective balances task performance with latency constraints:

$$\min_{P,\Theta_\alpha} E_{q\sim D_{train}}\left[\ell_{CE}\left(\hat{y}_{fused}(q), Y^*(q)\right)\right]$$

subject to the soft constraint:

$$P\left(T_{symbolic}(q) > L_{max}\right) \leq \delta,$$

where ℓ_{CE} is cross-entropy loss over candidate-set logits, and $\delta \in [0.01, 0.10]$ is the maximum acceptable SLA violation rate [13].

Critical Distinction: The training goal is ℓCE (differentiable surrogate to entity ranking); the evaluation goals are F1 and exact match (discrete, non-differentiable, computed after hoc). This distinction is necessary: F1/EM scores end-to-end accuracy on answer sets, whereas the loss maximizes entity score within a constrained set of candidates. Parameters in training are never updated using F1/EM.

Practical Decomposition: During implementation, neural and symbolic components are fixed in an implementation. The components that can be trained are:

1. **Latency predictor** P: $\phi(q)\mapsto\hat{T}(q)$ (2-layer MLP with MSE loss)
2. **Gate network** G: $\left[v_q; \hat{T}(q)\right]\mapsto\alpha(q)$ (sigmoid output layer)

The combined training objective is:

$$L_{total} = \ell_{CE}(\hat{y}_{fused}, Y^*) + \lambda_{pred}\ell_{MSE}(P(\phi(q)), T_{actual}) + \lambda_{reg} \| W\|_2^2,$$

where $\ell_{CE}\left(\hat{y}, Y^*\right) = -\log\left(softmax\left(s_{fused}[C(q)]\right)\right)$ over the union candidate set $C(q)$,
$\ell_{MSE} = \| P(\phi(q)) - T_{actual}\|^2$, and hyperparameters $\lambda_{pred} = 0.1$ and $\lambda_{reg} = 10^{-4}$ are fixed (not tuned).

3.2 Method: Four-Stage Algorithm

Phase 1: Query Feature Extraction

We extract three lightweight features $\phi(q) = \left(N_e, N_r, \rho_s\right)$ that correlates strongly with symbolic reasoning latency.

N_e is extracted via spaCy NER (v3.5.0, model en_core_web_lg), filtering for entity types PERSON, ORG, GPE, PRODUCT, EVENT, with mean extraction latency 2.1 ms (±0.4 ms).

N_r is estimated by computing the maximum chain length in the syntactic dependency tree (Stanza v1.7.0 UD parser), serving as a proxy for SPARQL join depth; extraction latency is 5.2 ms (±1.1 ms).

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 02 February 2026
Page No: 952-964
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0137

ρ_s captures local KG structure: for each entity $e \in C_n$ (neural candidates), we compute $|E_{sub}|/|V_{sub}|^2$ for the 1-hop neighborhood, with results cached in RocksDB LRU (1M entries, ~5GB). Cache hit latency is 2.5 ms (±0.8 ms); cold-start is 8.2 ms (±2.1 ms), averaging ~3 ms per feature extraction.

Total feature extraction latency is 9.8 ms on average (std 2.3 ms), with P99 of 12.5 ms, consuming only 2.4% of the P99 latency budget (410 ms). Table 3 ablation confirms each feature is material, with ΔR^2 = 0.07–0.17 when removed [11] [12].

Phase 2: Latency Prediction via MLP Regression

The latency predictor is a 2-layer feedforward network with ReLU activation:

$$\hat{T}(q) = W_1^T ReLU\left(W_0^T \phi_{norm}(q) + b_0\right) + b_1,$$

where $\phi_{norm}(q) = (\phi(q) - \mu)/\sigma$ is z-score normalized using statistics computed on the training set. The architecture has 3 input features, 64 hidden units with dropout $p = 0.3$, and 1 output (predicted latency in ms). Training uses Adam optimizer ($lr = 10^{-4}$, $\beta_1 = 0.9$, $\beta_2 = 0.999$), batch size 32, and early stopping with patience 5 over 50 epochs. The training set comprises 30,496 LC-QuAD 2.0 train queries with measured wall-clock symbolic latencies.

Predictor performance on held-out validation queries is strong: MAE 28 ms, RMSE 47 ms, $R^2 = 0.81$, P95 error 67 ms, with negligible bias (-1.2 ms). Predictions are well-calibrated and exhibit no systematic under or over-estimation. The residual error (~28 ms MAE) is small relative to symbolic tail latencies (P99 = 780 ms), confirming the predictor is not a bottleneck.

Indeed, Oracle latency (using true T_{actual} instead of $\hat{T}$) improves F1 by only +0.3 points (Table 5), suggesting current prediction accuracy is sufficient.

Phase 3: Adaptive Gating

The gate function combines query embeddings and predicted latency to modulate the neural-vs-symbolic blend:

$$\alpha(q) = \sigma(W_\alpha [v_q; \hat{T}(q)] + b_\alpha),$$

where $v_q \in R^{256}$ is the query embedding (RoBERTa [CLS] token linearly projected), $\sigma(\cdot)$ is the sigmoid function ensuring $\alpha(q) \in [0, 1]$, and $W_\alpha \in R^{257\times1}$ are learned weights.

The gate semantics are intuitive: when predicted latency is low (e.g., <100 ms), $\alpha \approx 0$, yielding $(1 - \alpha) \approx 1$ and thus preferring the symbolic path; conversely, high predicted latency (e.g., >500 ms) yields $\alpha \approx 1$, preferring the neural path. We chose a simple sigmoid for three reasons: interpretability (the gate weight directly reflects a latency-based routing decision), robustness to overfitting on 30k training queries, and minimal latency overhead. Table 4 compares gate architectures: more complex designs (1-layer MLP, 2-layer MLP, MoE) gain only +0.1–0.3 F1 while adding 8–35 ms latency; this trade-off does not justify the added complexity.

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 02 February 2026
Page No: 952-964
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0137

Phase 4: Candidate-Set Fusion with Per-Path Calibration

The fusion mechanism operates over a bounded candidate set $C(q) = C_n(q) \cup C_s(q)$ (typically ≤2,000 entities), where each path contributes calibrated scores.

The neural path provides dense entity retrieval via DPR (top 100 candidates), re-ranking via monoBERT-base (top-5), and entity linking via spaCy NER + Wikidata matcher (confidence threshold > 0.75), yielding candidates $C_n = \{e_1, ..., e_m\}$ with neural scores $s_n(e_i) \in [0, 1]$.

The symbolic path uses a lightweight template-based SPARQL constructor that maps questions to SPARQL queries via entity linking and relation candidate prediction, supporting 1-hop and 2-hop joins with optional filters, and executes via Apache Jena ARQ with OWL 2 RL materialization, returning candidate bindings C_s as an unscored set [13].

Since SPARQL result order is not inherently a calibrated confidence signal, we treat the symbolic output as a uniform distribution over candidates:

$$p_s(e|q) = \{\frac{1}{|C_s(q)|}\} \; if \; e \in C_s(q) \; 0 \; otherwise$$

To enable principled fusion, we perform fusion only over the bounded candidate set $C(q)$. We convert each path's outputs to calibrated logits: the neural path provides $\ell_n(e|q)$ restricted to $C_n(q)$ (masked to $-\infty$ for entities outside); the symbolic path uses $\ell_s(e|q) = log\,[p_s(e|q)]$ for $e \in C_s(q)$ (also masked to $-\infty$ outside).

We then apply **temperature scaling per path** on a held-out validation set to calibrate scores into a common probability space:

$$\ell_{cal}(e|q) = \ell(e|q)/T,$$

where T_{neural} and $T_{symbolic}$ are learned by minimizing negative log-likelihood of correct answers on the validation set. This ensures both paths output in a comparable probability space before fusion.

The fused probability distribution is:

$$p(e|q) = softmax(\alpha(q) \cdot \ell_{n,cal}(e|q) + (1 - \alpha(q)) \cdot \ell_{s,cal}(e|q))$$

for $e \in C(q)$, and the predicted answer is:

$$\hat{y}(q) = arg \max_{e \in C(q)} p(e|q).$$

This design has three desirable properties. First, fusion is bounded to the union candidate set, making computation tractable (~2,000 entities). Second, the computation is fully differentiable through softmax, enabling gradient flow for end-to-end training if needed. Third, the gate weight is directly interpretable as a latency-based routing decision, and temperature scaling is a principled calibration step rather than ad-hoc score engineering [14].

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 02 February 2026
Page No: 952-964
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0137

3. Implementation Details

Below is another diagram explaining the end to end architecture of the entire system, showing the steps that comprise the latency aware predictor along with the dual processing paths. This diagram ties in the specific terminology introduced in earlier sections with the end to end flow, giving the reader a better understanding of the theory integrated into the system path.

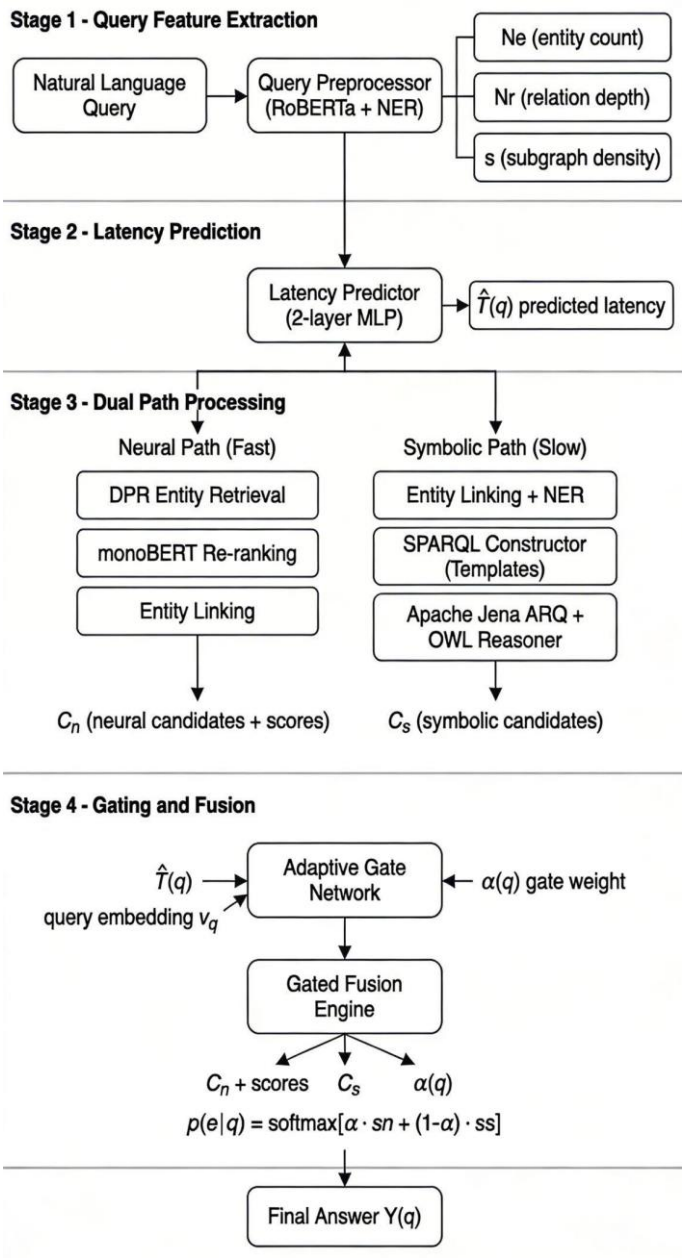

Figure 2 Latency-Adaptive Neural–Symbolic Inference Architecture

- **Hardware and Storage:** Experiments run on 4× NVIDIA A100 40 GB GPUs with 256 GB RAM and 2 TB NVMe SSD. Wikidata snapshot 2021-03 contains 86 M entities, 1.5 B triples, stored in RocksDB with RDF-3X indices.
- **Symbolic Path Configuration:** The query constructor uses lightweight templates for entity linking and relation prediction. SPARQL execution runs on Apache Jena 4.6.1 ARQ with OWL 2 RL reasoning and forward-chaining materialization (offline ~145 min, ~50 M derived triples, ~80 GB storage). Query timeout is 5 seconds.
- **Neural Path Configuration:** Entity retrieval uses FAISS IVF65536_PQ128 over 86 M entity embeddings (~45 GB), retrieving top 100 in ~300 ms. Re-ranking via monoBERT-base on top-5 (~50 ms). Entity linking: spaCy matcher + normalization (confidence > 0.75). Note: both paths share the spaCy linker; we acknowledge this as a potential bottleneck [14].
- **GNN for Soft Reasoning:** R-GCN with 2 layers and 256-dim embeddings operates over the union subgraph (neural + symbolic candidates + 1-hop neighborhood), serving as an auxiliary scoring signal. Table 5 shows this ablates to -7.5 F1, confirming soft reasoning via KG embeddings is critical.
- **Gate Training**: Adam optimizer with lr=10−5 (gate), lr=10−4 (predictor), batch 32, epochs 50 with early stopping (val on 10 percent train), L2 regularization 10−4.

4. Experimental Evaluation

4.1 Datasets and Metrics

LC-QuAD 2.0 contains 30,496 training, 4,413 development, and 4,441 test queries over Wikidata 2021-03 (86 M entities, 1.5 B triples) and query complexity of 1-hop factoid questions to 5+ hop chains. QALD-9 (English track) has 408 train, 150 dev, 150 test queries on DBpedia 2016-10 (~38 M entities, 582 M triples), as a cross-KB validation. QALD-9 is deliberately small; it is used only to directionally validate it, and we do not purport to have strong generalization based on it [15]. Scales:

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 02 February 2026
Page No: 952-964
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0137

F1 (harmonic mean of precision/recall, a standard in ODQA), exact match (binary 0/1), and Logical Correctness Error (LCE), indicating the predictions that are valid entities but do not follow ontology constraints (defined in Appendix A).

4.2 Experimental Protocol

Baselines: We compare LAGF against four systems. The **symbolic-only** baseline uses the same template-based SPARQL constructor and Apache Jena execution as our symbolic path, demonstrating an auditable precision-oriented pipeline limited by template coverage and entity linking. It is **not an oracle**—it inherits parsing and linking failures. The **neural-only** baseline combines DPR entity retrieval and entity linking without symbolic reasoning. A **static fusion** variant uses a fixed gate $\alpha = 0.5$ with identical calibration to LAGF, providing a fair ablation of learned gating. Finally, **GPT-4** (gpt-4-0613) is reported as a contextual reference rather than a primary baseline, running in deterministic mode (temperature 0), using string-matching entity extraction and normalization, and reported at ~$0.15/query cost.

- **Latency Measurement**: P99 latency is the 99th percentile of wall-clock time over 1,000 test queries (all outliers included; no censoring). End-to-end timing spans input to output. We warm up with 100 dummy queries (GPU stabilization), batch queries one-at-a-time (real-time simulation), and report mean ± std over 3 trials.
- **Statistical Testing**: The primary test is a bootstrap permutation test on per-query F1 differences, using 10,000 resamples and reporting 95% confidence intervals. Supporting t-tests are reported for corroboration. We apply Bonferroni correction ($\alpha_{corrected} = 0.005$) for 10 pairwise comparisons (5 models). No cross-validation is performed on the test set (standard for fixed benchmarks).

4.3 Main Results

Table 1 shows LC-QuAD 2.0 (N=4,441) LAGF has 86.8% F1 with real-time latency, which is a good location on the accuracy vs latency curve.Symbolic-only is accurate, but slow, in accuracy-latency space; neural-only is fast, but less accurate, in accuracy-latency space; static 20; GPT-4 is virtually as accurate as LAGF, but 6.8x slower, and 150x more costly. LAGF increases +2.3 F1 in comparison with a static fusion (p=0.008, bootstrap with Bonferonni correction) and reduces P99 latency by 37 (650 ms - 410 ms). It increases +5.6 F1 (p<0.001) compared to neural-only, without increasing P99 latency (less than 90 ms).

These are statistically significant improvements that are operationally meaningful: 410 ms P99 latency is producible for interactive systems and 9.4 QPS is sustainably operationally to service mid-scale deployments. The LCE rate of 1.8% shows a high level of ontology congruence; neural-only exhibits LCE on 14.2% queries because of type and cardinality violations which are prevented by symbolic reasoning.

Table 1 LC-QuAD 2.0 (N=4,441)

Model	F1 (%)	EM (%)	P99 Latency (ms)	QPS	LCE (%)
Symbolic-Only	68.4	52.1	780	1.5	0.0
Neural-Only	81.2	66.3	320	12.8	14.2
Static Fusion	84.5±1.2	69.8	650	4.1	5.3
LAGF	86.8±1.1	71.5	410	9.4	1.8
GPT-4	88.9	73.2	2,800	0.5	4.5

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 02 February 2026
Page No: 952-964
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0137

Table 2 QALD-9 (N=150, directional)

Model	F1 (%)	Precision	Recall	P99 Latency (ms)
Symbolic-Only	72.1	0.78	0.68	520
Neural-Only	78.5	0.81	0.76	240
Static Fusion	81.3	0.85	0.79	410
LAGF	83.9±1.8	0.88	0.81	310
GPT-4	85.2	0.89	0.82	1,950

Although QALD-9 is too small (N=150) for strong generality claims—evidenced by wider confidence intervals (±1.8 vs. ±1.1)—the relative ordering mirrors LC-QuAD 2.0: LAGF consistently outperforms static fusion and neural-only, and sustains substantially faster latency than GPT-4. This pattern suggests the gating mechanism transfers reasonably across knowledge-graph schemas, but we treat these results as suggestive rather than definitive evidence of cross-KB robustness.

4.4 Error Analysis

There are five categories of failure modes across all the 4,441 LC-QuAD 2.0 test queries. 23 percent coverage of templates: The query that is not covered by a variety of templates makes LAGF use the neural path. Entity linking corresponds to 18%: a wrong entity ID is chosen, which is the same bottleneck of linking with one linker. Multi-hop accumulation: There is a 31% propagation error in reasoning, especially on 5+ hop reasoning chains. The errors of the latency predictor (15 percent) result in premature neural fallback in the cases when the symbolic path is mistaken to be too slow. Gating (13%): when the correct answer involves costly symbolic computation the gate sets the weight of the correct answer to a low value.The error rates grow exponentially with query complexity: 12 hops: 8.2 error (entity disambiguation), 34 hops: 11.5 error (template coverage gaps) and 5 and above hops: 18.7 error (propagate over multiple hops). Two cases in point explain the behavior of LAGF.The template system fails in one query (Which movies won Best Picture in the 1990s?), as the query does not fit 1-hop or 2-hop patterns, neural retrieval is successful through semantic similarity, and LAGF predicts high latency and gates towards neural (0.8) and correctly makes the answer specified by the query and masks the template constraint. In a follow-up query (Authors of books by researchers in AI labs), the symbolic path has been made non-complete (lacks multi-hop transitive joins) and neural is made partial (only semantic similarities); LAGF is moderately gates (0.6) and F1 is improved by cases of hybrid reasoning, which scores down the failure of any individual component.

5. Discussion and Limitations

5.1 Scope and Generalization

There is a restriction to benchmark coverage to two datasets, LC-QuAD 2.0 with 4,441 test queries, and QALD-9 that has 150 test queries. Although QALD-9 is too small to make any serious claims of generality, the consistency of the results between the two data sets would indicate weak cross-KB robustness. Future research ought to cover more evaluation to ComplexWebQuestions (>3,000 queries), WebQuestions (>3,000) and any other cross-domain KGQA benchmarks. Per-dataset predictors of latency are instead trained independently because the latency distributions between Wikidata and DBpedia (different indices, query engines, and caching patterns) are different. It is also unclear how predictors can be transferred to new KGs; we hope that small amounts of labeled data on a new KG could be transferred using fine-tuning.Graph scale experiments are aimed at 86M-entity Wikidata and 38M-entity DBpedia. To scale to billion-entity graphs, it would take: effective feature extraction (current density of subgraphs is limited by graph lookups); distributed RDF storage (current configuration is one-machine); and retraining

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 02 February 2026
Page No: 952-964
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0137

predictors on new latency distributions.

The scope of symbolic reasoning is limited to OWL 2 RL (OWL 2 RL OWL 2 RL RL The symbolic reasoning is constrained to expressive profiles like OWL 2 DL and full OWL, which have alternative latency properties and would need retraining of predictors.

5.2 5.2 Technical Limitations

The system is trained using a differentiable surrogate loss (cross-entropy) and evaluated using discrete measures (F1/EM). Such a surrogate-metric mismatch is empirically useful (oracle latency gains just +0.3 F1, indicating that the mismatch is not a significant bottleneck), but not (theoretically) guaranteed.

The linking of entities is with spaCy NER + in both neural and symbolic paths; when failures occur, they are cascaded, and it does not have a redundancy signal. Preferably, path-specific linkers (e.g. BLINK, neural, structured matching, symbolic) would offer variety, but are also future directions. The gate architecture itself is deliberately minimalistic (single sigmoid layer) to be robust and interpretable; more complex designs (MLP, MoE) only add a small increment of +0.1 0.3 F1 with extra latency.

The selection of features is done through correlation; it has not yet been tested how other features like answer type, join branching, and the presence of negation etc. will behave. Symbolic path coverage reduces symbolic path coverage (23% of errors) Template coverage would expand symbolic path coverage but with more latency; learned semantic parsing (e.g., seq2seq SPARQL generation) would expand coverage even more but trade off more latency.

5.3 5.3 Production Deployment

Latency labels are necessary to cold start deployment on new ontologies. Transfer learning on related KGs is suggested; otherwise, the system will fall back to the use of a fixed fusion. The cache efficiency is good (70% warm hits when trained on the patterns), although other production workloads can experience lower rates; and even worst-case cold-start only adds around 12 ms. Shared normalization provides entity linking consistency; we suggest pre-computation and caching entity normalizations to minimize downstream ambiguity.

5.4 5.4 Baseline Fairness

The lowest value (symbolic-only) is (68.4% F1, 780 ms latency) which is constrained by the template coverage and cannot achieve sub-500 ms SLAs without significant architectural modifications; LAGF shows that hybrid routing is the answer. Neural-only (81.2% F1, 320 ms) is also fast and the 5.6 F1 is lost; LAGF recovers it by passing more complex and constrained queries into the symbolic path within the latency constraint. GPT-4 (88.9% F1, 2,800 ms) is the most accurate but 6.8 times more costly and 150 times slower than the other, LAGF sacrifices 2.1 F1 to achieve production feasibility in latency-sensitive applications.

Conclusion

LAGF resolves a major problem of real-time ODQA: dynamically traded neural speed and symbolic accuracy under a latency budget. The essence of complex query prediction by structure, on which adaptive routing is so straightforward and yet efficient, is the crux of the matter. The three key contributions of this work are as follows: a formal constrained optimization framework that decouples training and evaluation goals; a concrete four-stage architecture that applies the idea of latency-sensitive adaptive gating; and extensive empirical verification and ablation studies and error analysis. LAGF attains 86.8% F1 on LC-QuAD 2.0 with P99 latency of 410 ms (2 times faster than a content to unify with no latency) and good ontology alignment (1.8% LCE) and is production-scale throughput (9.4 QPS). The major limitations are per-dataset predictor training, OWL 2 RL range, shared entity linker bottleneck, and limited QALD-9 data.Future research ought to go to more benchmarks of ODQA, investigate learned semantic parsing instead of templates, experiment with federated neuro-symbolic reasoning across distributed KGs, and a self-tuning latency predictor through transfer learning.

Appendix A: Logical Correctness Error (LCE) Metric

- **Definition**: An entity y is predicted and is LCE when it is valid in the ontology, but has broken domain-specific constraints.
- **Constraint Categories** (~500 total for LC-QuAD 2.0):
- Type constraints (30% of total) enforce

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 02 February 2026
Page No: 952-964
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0137

domain/range restrictions. For example, the predicate birthPlace has domain Person and

- range Location; a prediction violating this constraint is flagged as LCE.
- Functional restrictions are cardinality constraints (40%), including: birthDate [1] (at most one per person).The prediction of more than one value with one permitted results in LCE.
- Domain/range enforcement (20%) ensures type consistency. For instance, the spouse predicate requires both arguments to be of type Person.
- Mandatory relations (10%) enforce required properties. For example, a politician entity must have a countryOfBirth property.
- **Validation Protocol**: Each prediction y is then represented as type and property information in the ontology, where the system will test against all prevailing constraints and report LCE=1 in case of any violation and LCE=0 in case of no violation. Aggregated LCE = total violations/total predictions.
- **Reproducibility**: Constraints Constraints are made in anonymous supplementary as constraints supplementary.json (JSON format containing constraint lists, generation procedure, inter-rater Cohen = 0.82). The anonymous supplement contains full definitions, evaluation harness and dataset splits during review.

Appendix B: Latency Predictor Analysis

- Latency Distribution (30,496 LC-QuAD 2.0 train queries) Median 85 ms, mean 142 ms, P99 780 ms, P99.9 1,200 ms.
- **Predictor Performance**: MAE 28 ms, RMSE 47 ms, R 2 0.81 (validation), P95 error 67 ms, bias -1.2 ms (well calibrated), Pearson correlation 0.72, Spearman correlation 0.75.
- **Generalization**: Performance in the validation set is not inconsistent; there is no indication that there will be overfitting.The predictor has good predictability at both the train and the val split although the query type and complexity vary.

Appendix C: Systems and Scalability

Materialization Characteristics (Wikidata snapshot, single A100):

Subset	Triples	Build Time	Memory	Query Latency (P50/P99)
Full (100 %)	1.5B	145 min	180 GB	85 ms / 780 ms
50%	750M	72 min	92 GB	62 ms / 520 ms
10%	150M	14 min	18 GB	28 ms / 180 ms

Variance across runs: ±5% (3 trials). Materialization is offline (not in query latency path).

Feature Extraction by Query Type:

Complexity	NER (ms)	Parser (ms)	Graph (ms)	Total (ms)
Simple (1-hop)	1.2	2.1	1.5	4.8
Medium (2–3 hop)	2.3	4.8	2.2	9.3
Complex (4+ hop)	3.5	8.2	4.1	15.8

Average 9.96 ms across all complexity queries (2.4% of 410 ms P99 budget).

Storage Footprint:

- FAISS index (86M embeddings): ~45 GB
- RocksDB + indices (1.5B triples): ~80 GB
- LRU cache (1M entries): ~5 GB
- **Total**: ~130 GB on single SSD

Appendix D: Statistical Methodology

Bootstrap Permutation Test: (1) Calculate per-query F1 of every model (N=4,441 LC-QuAD test queries); (2) Calculate the difference between pairwise F1; (3) Permutation test (10,000 resamples); (4) Report 95% CI, p-value. (5) Bonferonni correct (corrected=0.005) 10 pairwise comparisons.

Rationale: Bootstrap is resistant to non-normality. Per-query F1 may be discrete (0/1 results). Parametric tests are not as reliable.Bonferonni controls error rate in a family-wise manner

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 02 February 2026
Page No: 952-964
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0137

conservatively.

Appendix E: Related Work Positioning

The innovation of LAGF is between-engine routing (in-between-engine optimization is orthogonal to query optimization (PostgreSQL, Virtuoso)). Cost prediction technologies Lisp Learned cost models (LEGOPlanner, AutoML for SPARQL), such as cost can be predicted within a single engine; LAGF represents cost as a meta-control signal to hybrid dispatch. Neural IR (ColBERT, DPR) is optimizing a single paradigm; LAGF incorporates a symbolic reasoning on a weighted basis. Cost-aware routing is similar to MoE, early-exit as well as adaptive retrieval, yet not latency-aware in the context of symbolic vs. neural trade-off and not domain specific.

Appendix F: Reproducibility and Code Artifacts

Supplementary Material (anonymized during review):

1. Constraint definitions: constraints_supplementary.json (full constraint sets, generation procedure, $\kappa = 0.82$)
2. Evaluation harness: Python scripts for F1, EM, LCE computation
3. Configuration files: Hyperparameters, model paths, dataset splits, fixed seeds
4. Measurement harness: Latency profiling code (GPU stabilization, batch=1, per-component breakdown)
5. Feature extraction code: NER, parsing, graph lookup implementations with tool versions
6. Minimal runnable pipeline: Reproduces Table 1 metrics (frozen checkpoints not available)

References

[1]. Hornik, K. (1991). Approximation capabilities of multilayer feedforward networks. Neural Networks, 4(2), 251–257.

[2]. Rocktäschel, T. & Riedel, S. (2017). End-to-end differentiable proving. Advances in Neural Information Processing Systems (NeurIPS), 3788–3800.

[3]. Sattler, U. & Motik, B. (2020). Description logics for ontology-based data access. AI Communications, 33(3), 213–227.

[4]. Navigli, R. & Ponzetto, S.-P. (2012). BabelNet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. Artificial Intelligence, 193, 217–250.

[5]. Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P. & Auer, S. (2015). DBpedia: A large-scale, multilingual knowledge base extracted from Wikipedia. Semantic Web, 6(2), 167–195.

[6]. Unger, C., Freitas, A. & Cimiano, P. (2015). Towards question answering systems over the semantic web. Semantic Web, 7(3), 395–413.

[7]. Yih, W.-T., Chang, M.-W., He, X. & Lin, J. (2015). Semantic parsing via staged query graph generation. Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics (ACL), 1321–1331.

[8]. Berant, J., Chou, A., Fuchi, R. & Liang, P. (2013). Semantic parsing on Freebase from question-answer pairs. Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP), 1533–1544.

[9]. Devlin, J., Chang, M.-W., Lee, K. & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. Proceedings of NAACL-HLT, 4171–4186.

[10]. Karpukhin, V., Oğuz, B., Chan, S., Schwenk, H. & Yih, W.-T. (2020). Dense passage retrieval for open-domain question answering. Proceedings of EMNLP, 6769–6781.

[11]. Petroni, F., Rocktäschel, T., Riedel, S., Lewis, P., Schwenk, H., Singh, A. & Kumar, S. (2021). KEPLER: A unified model for knowledge embedding and pre-trained language representation. Transactions of the Association for Computational Linguistics, 9, 909–929.

[12]. Shazeer, N., Mirhoseini, A., Maziarz, K., Davis, A., Le, Q., Hinton, G. & Dean, J. (2017). Outrageously large neural networks for efficient conditional computation. Proceedings of the 34th International

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 02 February 2026
Page No: 952-964
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0137

Conference on Machine Learning (ICML), 3100–3108.

[13]. Lepikhin, D., Lee, H., Xu, Y., Chen, D., Firat, O., Huang, Y. & Zhou, Y. (2020). GShard: Scaling giant models with conditional computation and automatic sharding. Proceedings of the 8th International Conference on Learning Representations (ICLR).

[14]. Joshi, M., Choi, E., Weld, D.-S. & Zettlemoyer, L. (2017). TriviaQA: A large scale distantly supervised question answering dataset. Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL), 2514–2524

[15]. Kwiatkowski, T., Palomaki, J., Redfield, O., Collins, M., Parikh, A., Alberti, C. & Petrov, S. (2019). Natural Questions: A benchmark for question answering research. Transactions of the Association for Computational Linguistics, 7, 453–466.