BEYOND INDEPENDENT CANDIDATES: JOINT QUANTUM REASONING FOR AMBIGUOUS STRUCTURED GENERATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Structured generation often requires selecting among multiple plausible interpretations of an underspecified input. Existing ambiguity-aware approaches preserve multiple valid outputs during learning, but typically score candidate structures independently, ignoring how competing interpretations agree or conflict with one another. We introduce a hybrid quantum–classical framework that instead reasons jointly over the candidate set. A classical generator produces multiple structured candidates, which are represented as relational graphs and encoded into a shared candidate-indexed quantum state. Pairwise structural disagreements condition parameterized quantum interactions, allowing the probability assigned to each candidate to depend on its relationships with competing hypotheses. Because several interpretations may be simultaneously valid, we further introduce a quantum multi-valid objective that maximizes measurement probability over the valid-candidate subspace rather than a single reference. We additionally derive finite-shot selection-stability conditions based on probability margins between competing interpretations. Experiments on text-to-SQL and text-to-visualization benchmarks show that the proposed framework outperforms classical graph reasoning, parameter-matched joint reasoning, and candidate-wise quantum baselines, reaching 62.5 and 73.4 exact match on nvBench and Spider, respectively. The gains increase with ambiguity, while finite-shot, circuit-depth, and quantum-noise analyses characterize robustness under quantum execution constraints.

1 INTRODUCTION

Structured generation translates natural-language inputs into formal representations Dong et al. (2025); Yang et al. (2019); Zheng et al. (2024), including SQL queries Gao et al. (2024), visualization specifications Ge et al. (2025), programs Ma et al. (2025), and other structured outputs. Most existing methods assume that each input corresponds to a single correct output. However, natural language is often underspecified and can admit multiple structurally distinct yet semantically valid interpretations Vijayvargiya et al. (2026); D'Amour et al. (2022). Ambiguity-aware methods address this problem by generating and selecting among multiple candidate structures Sani et al. (2026); Uy et al. (2023); Zhang et al. (2026a). In most cases, each candidate is independently encoded and scored before ranking. Consequently, these methods do not explicitly capture how competing interpretations agree, conflict, or differ. Graph representations can express these relational distinctions Fey et al. (2024); Liu et al. (2025); Zheng et al. (2025), but existing graph-reasoning methods predominantly operate on individual candidates Shao et al. (2020); Huang et al. (2026); Ni et al. (2025). This limitation is particularly important under high ambiguity, where several candidates may appear individually plausible while differing in subtle structural choices, such as join paths, grouping variables, aggregation functions, or condition attachments Luo et al. (2026); Bhaskar et al. (2023); Widdowson & Kurlin (2022).

This motivates us to investigate whether quantum computation can support joint reasoning over competing interpretations. Our objective goes beyond replacing a classical graph neural network with a quantum model that evaluates each candidate independently. We propose a hybrid quantum-classical model in which competing structured hypotheses coexist in a joint quantum state. Given an input x, a classical generator first produces a candidate set $\mathcal{S}(x) = \{s_1, \ldots, s_k\}$, with each candidate

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

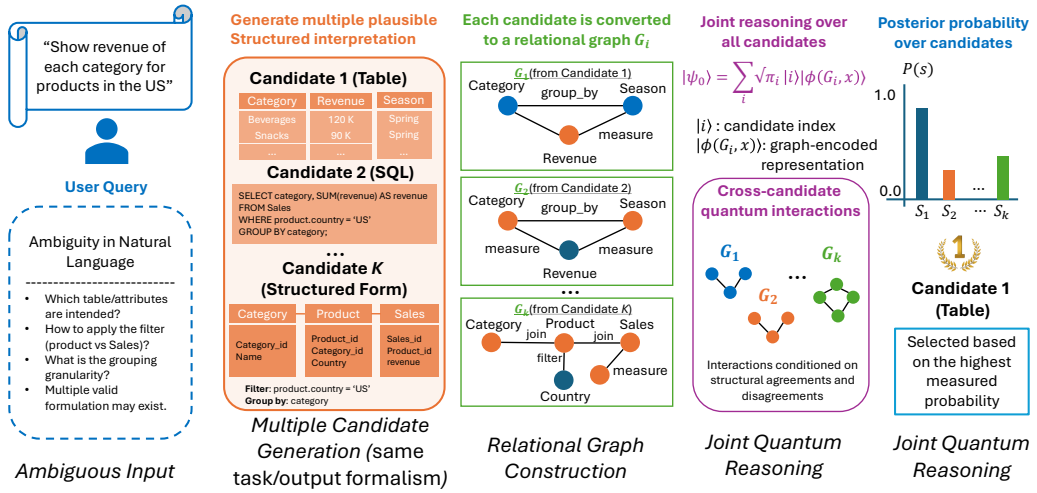


Figure 1: Overview of the proposed joint quantum reasoning framework for ambiguity-aware structured generation. Given an ambiguous natural-language input, a classical generator produces plausible structured candidates, which are converted into relational graphs. These graphs are jointly represented in a candidate-indexed quantum state, where disagreement-conditioned interactions capture structural agreements and conflicts among interpretations. Measuring the candidate register yields candidate probabilities, and the candidate with the highest measured probability is selected.

represented as a relational graph. These graphs are jointly encoded into a candidate-indexed quantum representation. Parameterized quantum interactions are conditioned on structural agreements and disagreements among the graphs, allowing the probability assigned to one interpretation to depend on its relationships with competing hypotheses. Candidate assessment thus considers both the candidate's representation and how its structural choices compare with those of other candidates. Ambiguity becomes a *set-level reasoning problem* rather than a sequence of independent scoring decisions.

Multiple candidates may be semantically valid, so optimizing toward a single reference can unfairly penalize valid alternatives Pourreza & Rafiei (2023); Luo et al. (2026). We therefore introduce a *quantum multi-valid objective* that maximizes the total measurement probability assigned to the valid-candidate subspace rather than concentrating it on one reference. We also analyze how finite-shot measurement affects candidate selection and derive stability conditions based on the probability margin between competing interpretations. As shown in Figure 1, candidate generation, graph construction, and relational feature extraction remain classical, while quantum computation supports joint reasoning among competing interpretations. We evaluate the framework on text-to-SQL and text-to-visualization tasks against deterministic generation, classical graph reasoning, a parameter-matched classical joint-reasoning model, and candidate-wise quantum baselines. We further examine performance across ambiguity levels, candidate-set sizes, measurement budgets, circuit depths, and quantum-noise levels. Our main contributions are:

- We formulate ambiguity-aware structured generation as a *joint hypothesis-set reasoning problem*, moving beyond independent evaluation of candidate structures.
- We introduce a hybrid quantum-classical model that jointly represents competing relational graphs and applies parameterized quantum interactions conditioned on their structural agreements and disagreements.
- We propose a quantum multi-valid objective that maximizes measurement probability over the subspace of semantically valid interpretations rather than a single reference output.
- We provide a finite-shot stability analysis and controlled comparisons with classical and quantum baselines, separating the effects of cross-candidate reasoning, disagreement conditioning, and multi-valid supervision across ambiguity levels, circuit depths, measurement budgets, and quantum-noise levels.

2 Related Work

Ambiguity-Aware Structured Generation. Structured generation tasks, including text-to-SQL Su et al. (2026); Cai et al. (2021), natural-language-to-visualization Lange et al. (2026); Ni et al. (2026), and program generation Mukherjee et al. (2021); Dong et al. (2025), typically assume a single target output, although ambiguous language can admit multiple structurally distinct yet semantically valid interpretations. Recent benchmarks increasingly expose this ambiguity; for example, nvBench 2.0 associates ambiguous visualization requests with multiple valid specifications Luo et al. (2026). Existing ambiguity-aware methods Kumari & Singh (2025); Sani et al. (2026); Li et al. (2026); Zhang et al. (2026a) generally retain multiple candidates but evaluate them independently using likelihood, semantic compatibility, or candidate-specific representations. Our work treats ambiguity as a property of the candidate set and explicitly models relationships among competing structured interpretations.

Graph and Set-Based Reasoning. Graph representations are widely used to capture relational structure in formal outputs Ying et al. (2021); Cotta et al. (2021); Dwivedi et al. (2026). In text-to-SQL, prior work models database-schema connectivity and relationships among query tokens, tables, columns, and schema elements Bogin et al. (2019); Wang et al. (2020); Cai & Wan (2020); Chen et al. (2021). Set-learning methods, such as Deep Sets Zaheer et al. (2017) and Set Transformer Lee et al. (2019), support permutation-invariant Tabaghi & Wang (2024) or interaction-aware processing of unordered elements Han et al. (2026). However, these architectures are not specifically designed to reason over competing structured interpretations through task-specific disagreement relations. We therefore include a classical joint-set baseline to distinguish generic cross-candidate interaction from the proposed disagreement-conditioned quantum framework.

Quantum Graph Learning and Hypothesis Reasoning. Quantum machine-learning methods have explored graph-structured data through parameterized and hybrid quantum-classical models. GraphQNTK Tang & Yan (2022), GQGLA Ye et al. (2025), and the Quantum Graph Transformer Aktar et al. (2025) investigate quantum processing of graph structures, while quantum hypothesis-selection methods evaluate competing quantum states or objectives Fawzi et al. (2023); Qiu et al. (2026). Our hypotheses are classical structured interpretations of a natural-language input. We jointly represent multiple candidate graphs, condition their interactions on typed structural disagreements, and maximize total probability within the verified valid-candidate subspace. Quantum computation thus serves as a trainable interaction mechanism among competing interpretations rather than replacing the classical graph encoder.

Research Gap and Positioning. Existing ambiguity-aware structured generation methods typically score candidates independently, while graph- and set-based methods represent structure or interactions without explicitly modeling competing interpretations through task-specific disagreement relations. Quantum graph-learning methods predominantly process individual graphs, whereas quantum hypothesis-selection methods generally operate on quantum states or objectives rather than classical structured candidates. We address this gap by jointly encoding candidate graphs in a shared candidate-indexed quantum state, conditioning cross-candidate interactions on structural disagreements, and training with a multi-valid objective over the verified valid-candidate subspace. This combination distinguishes our framework from independent candidate scoring, generic set interaction, single-graph quantum processing, and single-reference supervision.

3 Methodology

Overview. We formulate ambiguity-aware structured generation as joint reasoning among competing interpretations. Given a natural-language input x, a classical generator produces plausible structured outputs represented as relational graphs. We encode these candidates jointly in a shared quantum state and apply disagreement-conditioned interactions. The candidate-register marginal then defines a probability distribution over interpretations.

3.1 Problem Formulation and Candidate Graphs

Let $\mathcal{Y}(x)$ denote the set of semantically valid structured outputs for input x, where $|\mathcal{Y}(x)|$ may exceed one. The generator produces up to $k_{\max}$ candidates. After parsing, canonicalization, and

duplicate removal, the retained set is

$$\mathcal{S}(x) = \{s_1, \ldots, s_{K(x)}\}, \qquad K(x) = |\mathcal{S}(x)| \leq k_{\max}. \tag{1}$$

We write K for $K(x)$ when the input is clear from context. The valid retained candidates are

$$\mathcal{S}_{\text{valid}}(x) = \mathcal{S}(x) \cap \mathcal{Y}(x). \tag{2}$$

Successful candidate selection requires at least one valid interpretation in $\mathcal{S}(x)$. Each retained candidate s_i is converted into a typed relational graph $G_i = (V_i, E_i, \tau_i)$, yielding

$$\mathcal{G}(x) = \{G_1, \ldots, G_K\}. \tag{3}$$

For text-to-SQL, these graphs represent tables, columns, joins, aggregation, conditions, and values; for text-to-visualization, they represent attributes, encodings, aggregation, grouping, and filtering. A relational encoder produces

$$\mathbf{g}_i = \text{RelEnc}(G_i, x), \qquad \tilde{\mathbf{g}}_i = W_p \mathbf{g}_i + \mathbf{b}_p, \tag{4}$$

where $\tilde{\mathbf{g}}_i$ is the projected representation used for quantum encoding. To support joint candidate evaluation, we model

$$p_\theta\left(s_i \mid x, G_1, \ldots, G_K\right), \tag{5}$$

so that each candidate's probability can depend on the full set of competing interpretations.

3.2 Joint Quantum Representation

We use a candidate register and a feature register. The candidate register requires

$$q_c = \lceil \log_2 k_{\max} \rceil \tag{6}$$

qubits to accommodate the configured maximum candidate set, while a q_f-qubit feature register encodes relational information. For $K \leq k_{\max}$ retained candidates, only the first K candidate basis states are active, with candidate s_i associated with $|i-1\rangle_c$. Its projected graph representation is encoded as

$$|\phi_i\rangle_f = U_{\text{enc}}(\tilde{\mathbf{g}}_i)|0\rangle_f^{\otimes q_f}. \tag{7}$$

The K active candidates are jointly represented as

$$|\Psi_0(x)\rangle = \sum_{i=1}^{K} \sqrt{\pi_i}\, |i-1\rangle_c |\phi_i\rangle_f, \tag{8}$$

with a *uniform prior*

$$\pi_i = \frac{1}{K}, \qquad i \in \{1, \ldots, K\}, \tag{9}$$

so that the reasoning stage does not inherit a preference from the generator. Unused candidate-register basis states have zero amplitude under ideal state preparation. Equation equation 8 represents all active hypotheses in a shared quantum state rather than processing them independently. We use explicit controlled state preparation without assuming quantum random-access memory; circuit details are provided in Appendix B.

3.3 Cross-Candidate Quantum Reasoning

To capture structural differences among candidates, we compute a pairwise disagreement vector

$$\mathbf{d}_{ij} = D(G_i, G_j, x) = [d_{ij}^{(1)}, \ldots, d_{ij}^{(R)}], \tag{10}$$

where $d_{ij}^{(r)}$ measures disagreement for relation type r, such as join structure, grouping, aggregation, condition attachment, or value selection. Let

$$T_{ij} = |i-1\rangle\langle j-1| + |j-1\rangle\langle i-1| \tag{11}$$

denote the transition operator connecting candidates i and j. At reasoning layer ℓ, the ambiguity Hamiltonian is

$$H_{\text{amb}}^{(\ell)} = \sum_{1 \leq i < j \leq K} \sum_{r=1}^{R} \theta_r^{(\ell)} d_{ij}^{(r)} T_{ij} \otimes P_r, \tag{12}$$

where $\theta_r^{(\ell)}$ is trainable and P_r is a Hermitian Pauli operator or Pauli string acting on the feature register. Structural disagreement thus controls the interaction strength: small $d_{ij}^{(r)}$ yields a weak interaction, while greater disagreement increases the magnitude of the effective angle $\theta_r^{(\ell)} d_{ij}^{(r)}$. The corresponding evolution is

$$U_{\mathrm{amb}}^{(\ell)} = \exp\left(-iH_{\mathrm{amb}}^{(\ell)}\right). \tag{13}$$

After L reasoning layers,

$$|\Psi_L(x)\rangle = U_{\mathrm{amb}}^{(L)} \cdots U_{\mathrm{amb}}^{(1)} |\Psi_0(x)\rangle. \tag{14}$$

The exact marginal probability assigned to candidate s_i is

$$p_\theta(s_i \mid x, \mathcal{G}(x)) = \langle \Psi_L | \left(|i-1\rangle\langle i-1|_c \otimes I_f \right) |\Psi_L\rangle. \tag{15}$$

Thus, each candidate's probability depends on its relational representation and structural relationships with competing interpretations.

3.4 Multi-Valid Learning and Inference

Because multiple candidates may be semantically valid, single-reference training can penalize valid alternatives. Let $\mathcal{I}_{\mathrm{valid}}(x)$ denote the indices of valid candidates. Their total probability mass is

$$P_{\mathrm{valid}}(x) = \sum_{i \in \mathcal{I}_{\mathrm{valid}}(x)} p_\theta(s_i \mid x, \mathcal{G}(x)), \tag{16}$$

and we optimize

$$\mathcal{L}_{\mathrm{QMulti}} = -\log\left(P_{\mathrm{valid}}(x) + \epsilon\right). \tag{17}$$

This objective rewards probability assigned to the *valid-candidate subspace* and penalizes probability outside it. Joint Quantum+Single uses the same joint architecture but optimizes toward a single reference candidate, enabling a controlled comparison of multi-valid supervision.

Validity annotations are not required during inference. Under exact state-vector inference, we select

$$\hat{s}_{\mathrm{exact}} = \arg\max_{s_i \in \mathcal{S}(x)} p_\theta(s_i \mid x, \mathcal{G}(x)). \tag{18}$$

With M measurement shots, we estimate

$$\hat{p}_i = \frac{N_i}{M}, \tag{19}$$

where N_i counts outcomes corresponding to candidate state $|i-1\rangle_c$, and select

$$\hat{s}_{\mathrm{shot}} = \arg\max_{s_i \in \mathcal{S}(x)} \hat{p}_i. \tag{20}$$

The initial experiments use exact state-vector inference, while finite-shot execution is evaluated separately for robustness. Details of state preparation, circuit decomposition, optimization, quantum-resource requirements, and pseudocode are provided in Appendices B–C.

4 Theoretical Analysis

We analyze the proposed ambiguity-aware quantum reasoning framework under the candidate-space formulation introduced in Section 3. The analysis separates three roles of the method: multi-valid supervision preserves alternative valid interpretations, disagreement-conditioned interactions provide stable cross-candidate reasoning, and finite-shot measurement recovers the exact or valid candidate when probability margins are sufficiently large. All proofs are deferred to Appendix A.

A. Multi-valid candidate subspace: For an input x, let $\mathcal{V}(x) = \mathcal{I}_{\mathrm{valid}}(x)$ denote the valid candidate indices. For the final state $|\Psi_L\rangle$, let $\Pi_i = |i-1\rangle\langle i-1|_c \otimes I_f$ and $p_i = \langle \Psi_L | \Pi_i | \Psi_L \rangle$. The valid-subspace projector is $\Pi_{\mathcal{V}} = \sum_{i \in \mathcal{V}(x)} \Pi_i$, giving total valid probability $P_{\mathcal{V}} = \langle \Psi_L | \Pi_{\mathcal{V}} | \Psi_L \rangle = \sum_{i \in \mathcal{V}(x)} p_i$ and objective $\mathcal{L}_{\mathrm{QMulti}} = -\log(P_{\mathcal{V}} + \epsilon)$.

Lemma 4.1 (Valid-subspace invariance)**.** *For any two candidate distributions* $\mathbf{p}$ *and* $\mathbf{p}'$ *satisfying* $\sum_{i \in \mathcal{V}} p_i = \sum_{i \in \mathcal{V}} p'_i$*, we have* $\mathcal{L}_{\mathrm{QMulti}}(\mathbf{p}) = \mathcal{L}_{\mathrm{QMulti}}(\mathbf{p}')$*. Thus, redistributing probability within the valid-candidate subspace while preserving its total mass leaves the objective unchanged.*

The proof is provided in Appendix A.1. Lemma 4.1 formalizes the intended ambiguity-aware behavior: verified valid interpretations do not compete at the objective level when their aggregate probability is unchanged.

Proposition 4.2 (Aggregate valid supervision)**.** *The multi-valid gradient is* $\nabla_\theta \mathcal{L}_{\mathrm{QMulti}} = -(P_\mathcal{V} + \epsilon)^{-1} \sum_{i \in \mathcal{V}(x)} \nabla_\theta p_i$*. For any designated valid reference* $r \in \mathcal{V}(x)$ *with* $\mathcal{L}_{\mathrm{QSingle}} = -\log(p_r + \epsilon)$*,*

$$\mathcal{L}_{\mathrm{QMulti}} \leq \mathcal{L}_{\mathrm{QSingle}}, \qquad \mathcal{L}_{\mathrm{QSingle}} - \mathcal{L}_{\mathrm{QMulti}} = \log\left(\frac{P_\mathcal{V} + \epsilon}{p_r + \epsilon}\right) \geq 0. \tag{21}$$

See Appendix A.2. Proposition 4.2 shows that QMULTI optimizes aggregate valid probability rather than forcing probability toward one designated interpretation. The result is an objective-level property and does not itself imply improved optimization or generalization.

B. Disagreement-conditioned quantum stability: For candidate pair (i, j) and relation type r, let $A_{ijr} = T_{ij} \otimes P_r$, where $\|A_{ijr}\|_2$=1. The implemented interaction is $U_{ijr}^{(\ell)}(d) = \exp[-i\theta_r^{(\ell)} d_{ij}^{(r)} A_{ijr}]$. Consider perturbed disagreements $\widetilde{d}_{ij}^{(r)} = d_{ij}^{(r)} + \xi_{ij}^{(r)}$, while keeping the candidate set, initial state, learned parameters, and product ordering fixed. Let $|\Psi_L\rangle$ and $|\widetilde{\Psi}_L\rangle$ denote the corresponding final states, with candidate distributions $\mathbf{p}$ and $\widetilde{\mathbf{p}}$.

Theorem 4.3 (Stability of disagreement-conditioned reasoning)**.** *Let us define* $E_D = \sum_{\ell=1}^{L} \sum_{1 \leq i < j \leq K} \sum_{r=1}^{R} |\theta_r^{(\ell)}||\xi_{ij}^{(r)}|$*. Then*

$$\left\| |\Psi_L\rangle - |\widetilde{\Psi}_L\rangle \right\|_2 \leq E_D, \;\; \mathrm{TV}(\mathbf{p}, \widetilde{\mathbf{p}}) \leq \min\{1, E_D\}. \tag{22}$$

Consequently, $|p_i - \widetilde{p}_i| \leq E_D$ *for every active candidate and* $|P_\mathcal{V} - \widetilde{P}_\mathcal{V}| \leq E_D$*. If* $P_\mathcal{V}, \widetilde{P}_\mathcal{V} \geq \mu$*, then* $|\mathcal{L}_{\mathrm{QMulti}} - \widetilde{\mathcal{L}}_{\mathrm{QMulti}}| \leq E_D/(\mu + \epsilon)$*. Moreover, if* $i^\star = \arg\max_i p_i$ *has margin* $\Delta = p_{i^\star} - \max_{j \neq i^\star} p_j > 2E_D$*, then* $\arg\max_i \widetilde{p}_i = i^\star$*.*

See Appendix A.3. Theorem 4.3 shows that errors in the structural disagreement representation induce bounded changes in the final candidate distribution. The bound weights each perturbation by its learned interaction magnitude $|\theta_r^{(\ell)}|$, while the decision remains unchanged whenever the perturbation is smaller than half of the candidate probability margin.

C. Finite-shot candidate recovery: Let $i^\star = \arg\max_i p_i$ denote the unique candidate selected under exact state-vector inference, let $\Delta_j = p_{i^\star} - p_j$ for each $j \neq i^\star$, and define the minimum margin $\Delta = \min_{j \neq i^\star} \Delta_j > 0$. Under M independent measurements, the empirical probability is $\widehat{p}_i = N_i/M$ and the finite-shot prediction is $\hat{s}_{\mathrm{shot}} = \arg\max_i \widehat{p}_i$.

Theorem 4.4 (Finite-shot selection stability)**.** *For* M *independent measurements of the candidate register,*

$$\Pr[\hat{s}_{\mathrm{shot}} \neq \hat{s}_{\mathrm{exact}}] \leq \sum_{j \neq i^\star} \exp\left(-\frac{M\Delta_j^2}{2}\right) \leq (K-1)\exp\left(-\frac{M\Delta^2}{2}\right). \tag{23}$$

The proof is provided in Appendix A.4. Theorem 4.4 shows that, for any fixed $\Delta > 0$, disagreement between finite-shot and exact selection decreases exponentially with M.

Corollary 4.5 (Shot complexity)**.** *For any target failure probability* $\delta \in (0, 1)$*,* $M \geq 2\Delta^{-2}\log((K-1)/\delta)$ *is sufficient to guarantee* $\Pr[\hat{s}_{\mathrm{shot}} = \hat{s}_{\mathrm{exact}}] \geq 1 - \delta$*. Hence,* $M = \mathcal{O}(\Delta^{-2}\log(K/\delta))$*.*

D. Multi-valid finite-shot recovery: Exact candidate agreement is stronger than necessary when several interpretations are valid. Let $i_\mathcal{V}^\star = \arg\max_{i \in \mathcal{V}(x)} p_i$ be the highest-probability valid candidate and define the valid-invalid margin $\Delta_\mathcal{V} = p_{i_\mathcal{V}^\star} - \max_{j \notin \mathcal{V}(x)} p_j$.

Corollary 4.6 (Finite-shot valid-set recovery)**.** *If* $\Delta_\mathcal{V} > 0$*, then*

$$\Pr[\hat{s}_{\mathrm{shot}} \notin \mathcal{S}_{\mathrm{valid}}(x)] \leq \big(K - |\mathcal{V}(x)|\big)\exp\left(-\frac{M\Delta_\mathcal{V}^2}{2}\right). \tag{24}$$

See Appendix A.6. Corollary 4.6 captures the ambiguity-aware setting directly: finite-shot execution need not reproduce the same valid candidate as exact inference, provided the empirical winner remains within the valid-candidate set.

Remark 1. Lemma 4.1 and Proposition 4.2 characterize the supervision mechanism. Single-reference training concentrates supervision on one interpretation, whereas QMULTI optimizes total probability over the verified valid-candidate subspace.

Remark 2. Theorem 4.3, Theorem 4.4, and Corollary 4.6 identify complementary robustness conditions. Structural perturbations are controlled by the ratio between E_D and the learned candidate margin Δ, while sampling uncertainty is controlled by $M\Delta^2$, or by $M\Delta_{\mathcal{V}}^2$ when correctness is defined over the valid-candidate set. These results connect the theory to the disagreement ablations, ambiguity analysis, finite-shot experiments, and quantum-noise evaluation.

5 EXPERIMENTS

We evaluate the proposed framework through four research questions: **RQ1:** Does joint quantum reasoning improve ambiguity-aware structured generation over deterministic generation, candidate-based methods, and classical graph reasoning? **RQ2:** What are the contributions of cross-candidate interaction, structural disagreement information, and multi-valid supervision? **RQ3:** How does the framework behave as ambiguity and candidate-set size increase? **RQ4:** How robust is the framework to finite-shot measurement, circuit depth, and quantum noise?

5.1 EXPERIMENTAL SETUP

We evaluate on *nvBench* Luo et al. (2026) (7,878 natural-language-to-visualization instances) and *Spider* Yu et al. (2018) (10,181 text-to-SQL examples), preserving the benchmark partitions; Spider results use the public development split. LLaMA-2-13B Touvron et al. (2023) generates up to $k_{\max} = 5$ candidates per input at temperature $T = 0.7$. Candidates are parsed, canonicalized, deduplicated, cached, and shared across all candidate-based methods. We distinguish the verified valid-output set $\mathcal{Y}_{\text{ver}}(x)$ from the potentially larger set of semantically valid interpretations and define $\mathcal{S}_{\text{valid}}(x) = \mathcal{S}(x) \cap \mathcal{Y}_{\text{ver}}(x)$. Dataset-specific verification procedures are described in Appendix F.4.

For ambiguity analysis, covered examples are grouped by $|\mathcal{S}_{\text{valid}}(x)|$ into low (1), medium (2–3), and high (≥ 4) *candidate-set ambiguity*. Uncovered examples remain in end-to-end and coverage evaluations. We compare JOINTQ-MULTI against deterministic generation An et al. (2024), likelihood- and LLM-based candidate reranking Pourreza et al. (2025), independent graph reasoning with single-reference (GRAPH+SINGLE) Shao et al. (2020); Zhang et al. (2026b) and multi-valid (GRAPH+MULTI-VALID) supervision Hou et al. (2026), and construct an interaction-parameter-matched CLASSICAL JOINT-SET, CANDIDATE-WISE QUANTUM, and JOINT QUANTUM+SINGLE. Unless otherwise specified, JOINTQ-MULTI uses $q_c = 3$ candidate qubits, $q_f = 4$ feature qubits, $R = 5$ disagreement categories, and $L = 2$ reasoning layers. The main results use exact state-vector inference and report means over three random seeds; finite-shot and noise robustness are evaluated separately. Full experimental details are provided in Appendices D–F.

5.2 MAIN RESULTS

Overall Performance (RQ1). Table 1 shows that performance improves as the models incorporate stronger reasoning over the candidate set. GRAPH+MULTI-VALID reaches 58.9 EM on nvBench and 70.3 on Spider, while CLASSICAL JOINT-SET improves to 60.1 and 71.2, indicating a benefit from cross-candidate interaction beyond independent graph scoring. JOINTQ-MULTI achieves the best performance at 62.5 and 73.4 EM, outperforming the interaction-parameter-matched CLASSICAL JOINT-SET by 2.4 and 2.2 points and CANDIDATE-WISE QUANTUM by 3.1 and 2.6 points, respectively. Its additional gain over JOINT QUANTUM+SINGLE further supports the contribution of multi-valid supervision.

Table 1: Exact Match (EM) on nvBench and Spider over three seeds.

Method	**nvBench**	**Spider**
Deterministic LLM	48.3	62.5
Candidate + Likelihood	52.1	65.2
Candidate + LLM Rerank	54.7	66.8
Graph + Single	57.1 ± 0.4	68.8 ± 0.3
Graph + Multi-valid	58.9 ± 0.3	70.3 ± 0.4
Classical Joint-Set	60.1 ± 0.4	71.2 ± 0.3
Candidate-wise Quantum	59.4 ± 0.3	70.8 ± 0.3
Joint Quantum + Single	61.2 ± 0.3	72.0 ± 0.3
JointQ-Multi	$\mathbf{62.5 \pm 0.2}$	$\mathbf{73.4 \pm 0.2}$

Table 2: Exact, multi-reference, and semantic/functional evaluation on nvBench and Spider. VSM denotes Visualization Semantic Match and EX execution accuracy. Trainable-model results are mean ± standard deviation over three seeds.

	nvBench			Spider		
Method	**EM**	**Multi-EM**	**VSM**	**EM**	**Multi-EM**	**EX**
Deterministic LLM	48.3	52.7	56.2	62.5	66.1	69.4
Candidate + Likelihood	52.1	58.9	62.1	65.2	70.4	73.3
Candidate + LLM Rerank	54.7	61.3	64.8	66.8	72.6	75.2
Graph + Multi-valid	58.9 ± 0.3	68.4 ± 0.4	71.2 ± 0.4	70.3 ± 0.4	78.2 ± 0.4	81.0 ± 0.4
Classical Joint-Set	60.1 ± 0.4	70.2 ± 0.4	73.0 ± 0.3	71.2 ± 0.3	79.4 ± 0.3	82.0 ± 0.3
Candidate-wise Quantum	59.4 ± 0.3	69.1 ± 0.4	71.9 ± 0.4	70.8 ± 0.3	78.8 ± 0.4	81.4 ± 0.4
JointQ-Multi	$\mathbf{62.5 \pm 0.2}$	$\mathbf{72.6 \pm 0.3}$	$\mathbf{75.8 \pm 0.3}$	$\mathbf{73.4 \pm 0.2}$	$\mathbf{81.5 \pm 0.3}$	$\mathbf{84.3 \pm 0.3}$

Table 3: Candidate coverage and conditional selection performance. Conditional selection is evaluated when Coverage@5$= 1$.

Metric	**nvBench**	**Spider**
Coverage@1	52.7	66.1
Coverage@3	69.8	79.7
Coverage@5	78.6	86.1
Conditional Selection	92.4	94.7
End-to-End Multi-EM	72.6 ± 0.3	81.5 ± 0.3

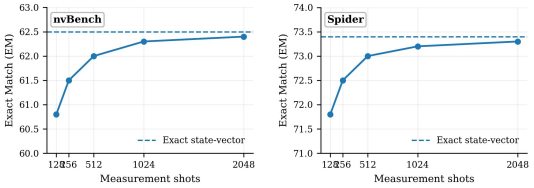


Figure 2: Finite-shot sensitivity of JOINTQ-MULTI; dashed lines denote exact state-vector performance.

Semantic and Functional Evaluation (RQ1–RQ2). Because exact matching can penalize structurally different but semantically equivalent outputs, we additionally report Visualization Semantic Match (VSM) for nvBench and execution accuracy (EX) for Spider. Table 2 shows that the performance pattern remains consistent under these broader criteria: JOINTQ-MULTI reaches 75.8 VSM and 84.3 EX, outperforming CLASSICAL JOINT-SET by 2.8 and 2.3 points. The gains beyond Multi-EM indicate that some remaining exact-match errors correspond to functionally equivalent outputs.

Candidate Coverage and Selection Accuracy. Candidate-based reasoning requires at least one verified valid interpretation in the generated pool. Table 3 shows that Coverage@5 reaches 78.6% on nvBench and 86.1% on Spider, while JOINTQ-MULTI selects a valid candidate in 92.4% and 94.7% of covered cases. The resulting Multi-EM scores show that remaining errors arise from both missing valid candidates and incorrect selection among retained candidates.

Finite-Shot Measurement (RQ4). Figure 2 shows convergence toward exact state-vector inference as the measurement budget increases. Pooled selection Agreement rises from 91.6% at 128 shots to 98.0% at 1024 shots, with an EM difference of at most 0.2 points at 1024 shots.

Performance under Increasing Ambiguity (RQ3). Table 4 shows that performance decreases as candidate-set ambiguity increases, but JOINTQ-MULTI degrades more slowly. Relative to CLASSICAL JOINT-SET, its advantage grows from 1.2 to 2.8 and 5.1 points across low, medium, and high ambiguity on nvBench, and from 1.2 to 2.2 and 4.4 points on Spider. The widening gap indicates that joint reasoning becomes increasingly beneficial when multiple valid interpretations compete.

Effect of Cross-Candidate Interaction (RQ2). Table 5 shows that joint reasoning alone provides limited gains when disagreement information is removed or shuffled. Disagreement-conditioned joint reasoning with single-reference supervision reaches 61.2 and 72.0 EM, while adding multi-valid supervision yields 62.5 and 73.4. These results support complementary contributions from joint interaction, meaningful disagreement features, and multi-valid supervision.

Effect of Candidate-Set Size (RQ3). Table 6 shows a clear gain from $k_{\max} = 3$ to 5, while increasing to 8 provides only marginal improvement despite quadratic growth in pairwise interactions. We therefore use $k_{\max} = 5$ as a practical trade-off between candidate coverage, accuracy, and interaction cost.

Table 4: Exact Match across candidate-set ambiguity levels over three seeds.

Method	nvBench			Spider		
	Low	Med.	High	Low	Med.	High
Graph + Multi-valid	63.8 ± 0.3	57.4 ± 0.4	48.9 ± 0.6	75.4 ± 0.3	69.2 ± 0.4	60.1 ± 0.6
Classical Joint-Set	64.2 ± 0.4	59.5 ± 0.4	52.7 ± 0.5	75.8 ± 0.3	70.7 ± 0.4	63.0 ± 0.5
Candidate-wise Quantum	64.0 ± 0.3	58.1 ± 0.5	50.2 ± 0.6	75.5 ± 0.3	69.8 ± 0.4	61.5 ± 0.6
Joint Quantum + Single	65.0 ± 0.3	60.9 ± 0.4	54.6 ± 0.5	76.6 ± 0.3	71.6 ± 0.3	64.8 ± 0.5
JointQ-Multi	$\mathbf{65.4 \pm 0.2}$	$\mathbf{62.3 \pm 0.3}$	$\mathbf{57.8 \pm 0.4}$	$\mathbf{77.0 \pm 0.2}$	$\mathbf{72.9 \pm 0.3}$	$\mathbf{67.4 \pm 0.4}$

Table 5: Cross-candidate reasoning ablation. Results are mean $\pm$ standard deviation over three seeds.

Variant	nvBench	Spider
Candidate-wise Quantum	59.4 ± 0.3	70.8 ± 0.3
JointQ w/o disagreement	60.0 ± 0.4	71.2 ± 0.3
JointQ + shuffled disagreement	59.7 ± 0.4	71.0 ± 0.4
Joint Quantum + Single	61.2 ± 0.3	72.0 ± 0.3
JointQ-Multi	$\mathbf{62.5 \pm 0.2}$	$\mathbf{73.4 \pm 0.2}$

Table 6: Effect of maximum candidate-set size $k_{\max}$.

$k_{\max}$	nvBench	Spider
3	60.2 ± 0.3	71.4 ± 0.3
5	$\mathbf{62.5 \pm 0.2}$	$\mathbf{73.4 \pm 0.2}$
8	62.7 ± 0.3	73.6 ± 0.3

5.3 Comparison with Other Approaches

Table 7 summarizes the comparison with deterministic, candidate-based, classical, and quantum alternatives. JointQ-Multi achieves 62.5 EM on nvBench and 73.4 on Spider, exceeding the interaction-parameter-matched Classical Joint-Set by 2.4 and 2.2 points and Candidate-wise Quantum by 3.1 and 2.6 points. Its additional gains of 1.3 and 1.4 points over Joint Quantum+Single further support the contribution of multi-valid supervision.

Table 7: EM comparison on nvBench and Spider. Trainable results are mean $\pm$ standard deviation over three seeds.

Method	nvBench	Spider
Deterministic LLM	48.3	62.5
Candidate + Likelihood	52.1	65.2
Candidate + LLM Rerank	54.7	66.8
Graph + Single	57.1 ± 0.4	68.8 ± 0.3
Graph + Multi-valid	58.9 ± 0.3	70.3 ± 0.4
Classical Joint-Set	60.1 ± 0.4	71.2 ± 0.3
Candidate-wise Quantum	59.4 ± 0.3	70.8 ± 0.3
Joint Quantum + Single	61.2 ± 0.3	72.0 ± 0.3
JointQ-Multi	$\mathbf{62.5 \pm 0.2}$	$\mathbf{73.4 \pm 0.2}$

5.4 Limitations

Our framework has several limitations. Performance depends on candidate coverage, as the reasoning module cannot recover a correct interpretation absent from the generated set. Relational graph construction is task-specific and requires suitable definitions of nodes, relations, and disagreements for each structured-output domain. Dense pairwise interactions scale quadratically with candidate-set size, making the current formulation better suited to small sets. Finally, most quantum evaluations rely on exact state-vector, finite-shot, and noisy simulations, while current hardware introduces additional connectivity, noise, decoherence, and compilation constraints. We therefore make no claim of computational speedup or quantum advantage; we investigate joint quantum representations as a mechanism for reasoning over competing structured interpretations.

6 Conclusion

In this paper, we presented a hybrid quantum-classical model for ambiguity-aware structured generation that jointly reasons over competing interpretations. Candidate structures are represented as relational graphs and encoded in a shared candidate-indexed quantum state, where typed structural disagreements condition cross-candidate interactions. A multi-valid objective increases probability over the valid-candidate subspace rather than targeting a single reference. Experiments on text-to-visualization and text-to-SQL show improvements over independent graph, candidate-wise quantum, and interaction-parameter-matched classical joint-reasoning baselines, with larger gains under higher ambiguity. Finite-shot and robustness analyses further characterize performance under quantum-execution constraints. These findings support modeling ambiguity across the full hypothesis set rather than evaluating interpretations independently.

AI USE STATEMENT

Generative AI tools were used to assist with language editing, organization, and manuscript drafting. The authors independently developed and verified the research methodology, mathematical formulation, experimental design, results, and conclusions, and reviewed all AI-assisted content for technical accuracy and consistency. The authors take full responsibility for the scientific content of the manuscript.

ETHICS STATEMENT

This work studies ambiguity-aware structured generation and quantum–classical reasoning using established benchmark datasets. It does not involve human participants, collection of private or personally identifiable information, or deployment in high-stakes decision-making settings. We do not introduce a new dataset or collect user data. The proposed methodology is intended for research on structured prediction and quantum machine learning. We explicitly report limitations related to candidate generation, task-specific graph construction, computational cost, and current quantum-hardware constraints. The authors are not aware of additional ethical concerns specific to this work.

REPRODUCIBILITY STATEMENT

We provide detailed information to support reproducibility throughout the main paper and supplementary material. Appendix B describes quantum state preparation, feature encoding, disagreement-conditioned interactions, circuit decomposition, and measurement. Appendix C provides the complete algorithm together with computational and quantum-resource complexity analyses. Appendix D reports the datasets, candidate-generation procedure, baselines, training hyperparameters, simulation settings, noise models, and hardware-evaluation protocol. Appendix E provides additional ablations and robustness analyses, while Appendix F documents prompting, candidate canonicalization, validity assignment, and preprocessing procedures. Random seeds, measurement budgets, and model configurations are reported explicitly. The supplementary material includes the implementation code and scripts required to reproduce the reported experiments.

REFERENCES

Shamminuj Aktar, Andreas Bärtschi, Abdel-Hameed A Badawy, and Stephan Eidenbenz. Quantum graph transformer for nlp sentiment classification. *arXiv preprint arXiv:2506.07937*, 2025.

Shengnan An, Zexiong Ma, Zeqi Lin, Nanning Zheng, Jian-Guang Lou, and Weizhu Chen. Make your llm fully utilize the context. *Advances in Neural Information Processing Systems*, 37:62160–62188, 2024.

Adithya Bhaskar, Tushar Tomar, Ashutosh Sathe, and Sunita Sarawagi. Benchmarking and improving text-to-sql generation under ambiguity. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 7053–7074, 2023.

Ben Bogin, Jonathan Berant, and Matt Gardner. Representing schema structure with graph neural networks for text-to-sql parsing. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4560–4565, 2019.

Ruichu Cai, Jinjie Yuan, Boyan Xu, and Zhifeng Hao. Sadga: Structure-aware dual graph aggregation network for text-to-sql. *Advances in Neural Information Processing Systems*, 34:7664–7676, 2021.

Yitao Cai and Xiaojun Wan. Igsql: Database schema interaction graph based neural model for context-dependent text-to-sql generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 6903–6912, 2020.

Yun Chen, Jingkang Wang, Ze Yang, Sivabalan Manivasagam, and Raquel Urtasun. G3r: Gradient guided generalizable reconstruction. In *European Conference on Computer Vision*, pp. 305–323. Springer, 2024.

Zhi Chen, Lu Chen, Yanbin Zhao, Ruisheng Cao, Zihan Xu, Su Zhu, and Kai Yu. Shadowgnn: Graph projection neural network for text-to-sql parser. In *Proceedings of the 2021 conference of the North American chapter of the association for computational linguistics: human language technologies*, pp. 5567–5577, 2021.

Leonardo Cotta, Christopher Morris, and Bruno Ribeiro. Reconstruction for powerful graph representations. *Advances in Neural Information Processing Systems*, 34:1713–1726, 2021.

Alexander D'Amour, Katherine Heller, Dan Moldovan, Ben Adlam, Babak Alipanahi, Alex Beutel, Christina Chen, Jonathan Deaton, Jacob Eisenstein, Matthew D Hoffman, et al. Underspecification presents challenges for credibility in modern machine learning. *Journal of Machine Learning Research*, 23(226):1–61, 2022.

Yixin Dong, Charlie F Ruan, Yaxing Cai, Ziyi Xu, Yilong Zhao, Ruihang Lai, and Tianqi Chen. Xgrammar: Flexible and efficient structured generation engine for large language models. *Proceedings of Machine Learning and Systems*, 7, 2025.

Vijay Prakash Dwivedi, Sri Jaladi, Yangyi Shen, Federico López, Charilaos Kanatsoulis, Rishi Puri, Matthias Fey, and Jure Leskovec. Relational graph transformer. In *International Conference on Learning Representations*, volume 2026, pp. 154348–154369, 2026.

Omar Fawzi, Nicolas Flammarion, Aurélien Garivier, and Aadil Oufkir. On adaptivity in quantum testing. *Transactions on Machine Learning Research*, 2023.

Matthias Fey, Weihua Hu, Kexin Huang, Jan Eric Lenssen, Rishabh Ranjan, Joshua Robinson, Rex Ying, Jiaxuan You, and Jure Leskovec. Position: Relational deep learning-graph representation learning on relational databases. In *Forty-first International Conference on Machine Learning*, 2024.

Yingqi Gao, Yifu Liu, Xiaoxia Li, Xiaorong Shi, Yin Zhu, Yiming Wang, Shiqi Li, Wei Li, Yuntao Hong, Zhiling Luo, et al. A preview of xiyan-sql: A multi-generator ensemble framework for text-to-sql. *arXiv preprint arXiv:2411.08599*, 2024.

Jiaxin Ge, Zora Zhiruo Wang, Xuhui Zhou, Yi-Hao Peng, Sanjay Subramanian, Qinyue Tan, Maarten Sap, Alane Suhr, Daniel Fried, Graham Neubig, et al. Autopresent: Designing structured visuals from scratch. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2902–2911. IEEE, 2025.

Xing Han, Hsing-Huan Chung, Joydeep Ghosh, Paul Liang, and Suchi Saria. Building massively multimodal foundation models with interaction-aware mixture-of-experts. In *International Conference on Learning Representations*, volume 2026, pp. 37694–37721, 2026.

Yu Hou, Zaifu Zhan, Min Zeng, Yifan Wu, Shuang Zhou, Xiaoyi Chen, Huixue Zhou, Meijia Song, and Rui Zhang. Auditing frontier general-purpose large language models in biomedical tasks: reasoning gains, extraction limits, and benchmark reliability. *Research Square*, pp. rs–3, 2026.

Manzong Huang, Chenyang Bu, Yi He, Xingrui Zhuo, and Xindong Wu. Relink: Constructing query-driven evidence graph on-the-fly for graphrag. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 31202–31210, 2026.

Suruchi Kumari and Pravendra Singh. Annotation ambiguity aware semi-supervised medical image segmentation. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10404–10413. IEEE, 2025.

Devin Lange, Pengwei Sui, Shanghua Gao, Marinka Zitnik, and Nils Gehlenborg. Dqvis dataset: Natural language to biomedical visualization. *Advances in Neural Information Processing Systems*, 38, 2026.

Juho Lee, Yoonho Lee, Jungtaek Kim, Adam Kosiorek, Seungjin Choi, and Yee Whye Teh. Set transformer: A framework for attention-based permutation-invariant neural networks. In *International conference on machine learning*, pp. 3744–3753. PMLR, 2019.

Fanding Li, Xiangyu Li, Xianghe Su, Xingyu Qiu, Suyu Dong, Wei Wang, Kuanquan Wang, Gongning Luo, and Shuo Li. Ambiguity-aware truncated flow matching for ambiguous medical image segmentation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 6082–6090, 2026.

Hongda Liu, Yunfan Liu, Min Ren, Hao Wang, Yunlong Wang, and Zhenan Sun. Revealing key details to see differences: A novel prototypical perspective for skeleton-based action recognition. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 29248–29257. IEEE, 2025.

Tianqi Luo, Chuhan Huang, Leixian Shen, Boyan Li, Shuyu Shen, Wei Zeng, Nan Tang, and Yuyu Luo. nvbench 2.0: Resolving ambiguity in text-to-visualization through stepwise reasoning. *Advances in Neural Information Processing Systems*, 38, 2026.

Lezhi Ma, Shangqing Liu, Yi Li, Xiaofei Xie, and Lei Bu. Specgen: Automated generation of formal program specifications via large language models. In *2025 IEEE/ACM 47th International Conference on Software Engineering (ICSE)*, pp. 16–28. IEEE, 2025.

Rohan Mukherjee, Yeming Wen, Dipak Chaudhari, Thomas Reps, Swarat Chaudhuri, and Christopher Jermaine. Neural program generation modulo static analysis. *Advances in Neural Information Processing Systems*, 34:18984–18996, 2021.

Bo Ni, Yu Wang, Lu Cheng, Erik Blasch, and Tyler Derr. Towards trustworthy knowledge graph reasoning: An uncertainty aware perspective. In *Proceedings of the AAAI conference on artificial intelligence*, volume 39, pp. 12417–12425, 2025.

Yuansheng Ni, Songcheng Cai, Xiangchao Chen, Jiarong Liang, Zhiheng Lyu, Jiaqi Deng, Kai Zou, Ping Nie, Fei Yuan, Xiang Yue, et al. Viscoder2: Building multi-language visualization coding agents. In *International Conference on Learning Representations*, volume 2026, pp. 104353–104422, 2026.

Mohammadreza Pourreza and Davood Rafiei. Evaluating cross-domain text-to-sql models and benchmarks. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 1601–1611, 2023.

Mohammadreza Pourreza, Hailong Li, Ruoxi Sun, Yeounoh Chung, Shayan Talaei, Gaurav Tarlok Kakkar, Yu Gan, Amin Saberi, Fatma Ozcan, and Sercan Arik. Chase-sql: Multi-path reasoning and preference optimized candidate selection in text-to-sql. In *International Conference on Learning Representations*, volume 2025, pp. 60385–60415, 2025.

Yixian Qiu, Lirandë Pira, and Patrick Rebentrost. Quantum learning with tunable loss functions. *npj Quantum Information*, 2026.

Depanshu Sani, Mehar Khurana, and Saket Anand. Active learning for animal re-identification with ambiguity-aware sampling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 39182–39190, 2026.

Bo Shao, Yeyun Gong, Weizhen Qi, Guihong Cao, Jianshu Ji, and Xiaola Lin. Graph-based transformer with cross-candidate verification for semantic parsing. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 8807–8814, 2020.

Xiaodong Su, Yang Gu, Peng Wang, Wei Gu, Lincheng Qi, and Jingwei He. A robust natural language text-to-sql generation framework with dynamic strategies based on llms. *Scientific Reports*, 16(1):7892, 2026.

Puoya Tabaghi and Yusu Wang. Universal representation of permutation-invariant functions on vectors and tensors. In *International Conference on Algorithmic Learning Theory*, pp. 1134–1187. PMLR, 2024.

Yehui Tang and Junchi Yan. Graphqntk: Quantum neural tangent kernel for graph data. *Advances in neural information processing systems*, 35:6104–6118, 2022.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Mikaela Angelina Uy, Ricardo Martin-Brualla, Leonidas Guibas, and Ke Li. Scade: Nerfs from space carving with ambiguity-aware depth estimates. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16518–16527. IEEE, 2023.

Sanidhya Vijayvargiya, Xuhui Zhou, Akhila Yerukola, Maarten Sap, and Graham Neubig. Ambigswe: Interactive agents to overcome underspecificity in software engineering. In *International Conference on Learning Representations*, volume 2026, pp. 146716–146737, 2026.

Bailin Wang, Richard Shin, Xiaodong Liu, Oleksandr Polozov, and Matthew Richardson. Rat-sql: Relation-aware schema encoding and linking for text-to-sql parsers. In *Proceedings of the 58th annual meeting of the association for computational linguistics*, pp. 7567–7578, 2020.

Daniel Widdowson and Vitaliy Kurlin. Resolving the data ambiguity for periodic crystals. *Advances in Neural Information Processing Systems*, 35:24625–24638, 2022.

Carl Yang, Peiye Zhuang, Wenhan Shi, Alan Luu, and Pan Li. Conditional structure generation through graph variational generative adversarial nets. *Advances in neural information processing systems*, 32, 2019.

Xinyu Ye, Hao Xiong, Jianhao Huang, Ziang Chen, Jia Wang, and Junchi Yan. On designing general and expressive quantum graph neural networks with applications to milp instance representation. In *International Conference on Learning Representations*, volume 2025, pp. 68460–68485, 2025.

Chengxuan Ying, Tianle Cai, Shengjie Luo, Shuxin Zheng, Guolin Ke, Di He, Yanming Shen, and Tie-Yan Liu. Do transformers really perform badly for graph representation? *Advances in neural information processing systems*, 34:28877–28888, 2021.

Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, et al. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-sql task. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pp. 3911–3921, 2018.

Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Russ R Salakhutdinov, and Alexander Smola. Deep sets. *Advances in neural information processing systems*, 30, 2017.

Fengji Zhang, Xinyao Niu, Chengyang Ying, Guancheng Lin, Zhongkai Hao, Zhou Fan, Chengen Huang, Jacky Keung, Bei Chen, and Junyang Lin. A ^{2} search: Ambiguity-aware question answering with reinforcement learning. In *International Conference on Learning Representations*, volume 2026, pp. 136394–136448, 2026a.

Yuanhe Zhang, Ilja Kuzborskij, Jason Lee, Chenlei Leng, and Fanghui Liu. Dag-math: Graph-of-thought guided mathematical reasoning in llms. In *International Conference on Learning Representations*, volume 2026, pp. 74347–74389, 2026b.

Lianmin Zheng, Liangsheng Yin, Zhiqiang Xie, Chuyue Sun, Jeff Huang, Cody H Yu, Shiyi Cao, Christos Kozyrakis, Ion Stoica, Joseph E Gonzalez, et al. Sglang: Efficient execution of structured language model programs. *Advances in neural information processing systems*, 37:62557–62583, 2024.

Yilun Zheng, Xiang Li, Sitao Luan, Xiaojiang Peng, and Lihui Chen. Let your features tell the differences: Understanding graph convolution by feature splitting. In *International Conference on Learning Representations*, volume 2025, pp. 61124–61157, 2025.

A Theoretical Properties and Proofs

This appendix provides the complete proofs for the theoretical results in Section 4. The analysis uses the implemented product-form ambiguity circuit rather than the conceptual exact Hamiltonian evolution.

For an input x, let the active candidate set contain K candidates and let $\mathcal{V}(x) \subseteq \{1, \ldots, K\}$ denote the verified valid-candidate indices. For the final joint state $|\Psi_L\rangle$, define

$$\Pi_i = |i-1\rangle\langle i-1|_c \otimes I_f, \tag{25}$$

so that the probability assigned to candidate i is

$$p_i = \langle \Psi_L | \Pi_i | \Psi_L \rangle. \tag{26}$$

The valid-subspace projector is

$$\Pi_{\mathcal{V}} = \sum_{i \in \mathcal{V}(x)} \Pi_i, \tag{27}$$

and therefore

$$P_{\mathcal{V}} = \langle \Psi_L | \Pi_{\mathcal{V}} | \Psi_L \rangle = \sum_{i \in \mathcal{V}(x)} p_i. \tag{28}$$

The multi-valid objective is

$$\mathcal{L}_{\mathrm{QMulti}} = -\log(P_{\mathcal{V}} + \epsilon), \qquad \epsilon > 0. \tag{29}$$

A.1 Proof of Lemma 4.1

Proof. Let $\mathbf{p}$ and $\mathbf{p}'$ be two candidate probability distributions satisfying

$$\sum_{i \in \mathcal{V}(x)} p_i = \sum_{i \in \mathcal{V}(x)} p'_i. \tag{30}$$

Define

$$P_{\mathcal{V}} = \sum_{i \in \mathcal{V}(x)} p_i \tag{31}$$

and

$$P'_{\mathcal{V}} = \sum_{i \in \mathcal{V}(x)} p'_i. \tag{32}$$

By equation 30,

$$P_{\mathcal{V}} = P'_{\mathcal{V}}. \tag{33}$$

Substituting into the multi-valid objective gives

$$\begin{aligned} \mathcal{L}_{\mathrm{QMulti}}(\mathbf{p}) &= -\log(P_{\mathcal{V}} + \epsilon) \\ &= -\log(P'_{\mathcal{V}} + \epsilon) \\ &= \mathcal{L}_{\mathrm{QMulti}}(\mathbf{p}'). \end{aligned} \tag{34}$$

Therefore, the objective depends only on the aggregate probability assigned to the valid-candidate subspace.

The same result can be written directly for a redistribution of probability. Let

$$p'_i = p_i + \delta_i \tag{35}$$

for $i \in \mathcal{V}(x)$, where

$$\sum_{i \in \mathcal{V}(x)} \delta_i = 0. \tag{36}$$

Then

$$\begin{aligned} P'_{\mathcal{V}} &= \sum_{i \in \mathcal{V}(x)} p'_i \\ &= \sum_{i \in \mathcal{V}(x)} (p_i + \delta_i) \\ &= \sum_{i \in \mathcal{V}(x)} p_i + \sum_{i \in \mathcal{V}(x)} \delta_i \\ &= P_{\mathcal{V}} + 0 \\ &= P_{\mathcal{V}}. \end{aligned} \tag{37}$$

Hence

$$\begin{aligned} \mathcal{L}_{\mathrm{QMulti}}(\mathbf{p}') &= -\log(P'_{\mathcal{V}} + \epsilon) \\ &= -\log(P_{\mathcal{V}} + \epsilon) \\ &= \mathcal{L}_{\mathrm{QMulti}}(\mathbf{p}). \end{aligned} \tag{38}$$

Thus, any probability redistribution within the valid-candidate subspace that preserves total valid probability leaves the loss unchanged. □

A.2 Proof of Proposition 4.2

Proof. We first derive the gradient of the multi-valid objective. By definition,

$$\mathcal{L}_{\mathrm{QMulti}} = -\log(P_{\mathcal{V}} + \epsilon), \tag{39}$$

where

$$P_{\mathcal{V}} = \sum_{i \in \mathcal{V}(x)} p_i. \tag{40}$$

Applying the chain rule yields

$$\begin{aligned} \nabla_\theta \mathcal{L}_{\mathrm{QMulti}} &= -\nabla_\theta \log(P_{\mathcal{V}} + \epsilon) \\ &= -\frac{1}{P_{\mathcal{V}} + \epsilon} \nabla_\theta P_{\mathcal{V}} \\ &= -\frac{1}{P_{\mathcal{V}} + \epsilon} \nabla_\theta \left(\sum_{i \in \mathcal{V}(x)} p_i \right) \\ &= -\frac{1}{P_{\mathcal{V}} + \epsilon} \sum_{i \in \mathcal{V}(x)} \nabla_\theta p_i. \end{aligned} \tag{41}$$

Thus, the optimization signal is determined by the aggregate probability assigned to the valid-candidate set.

We next compare the multi-valid objective with single-reference supervision. Let $r \in \mathcal{V}(x)$ denote a designated valid reference. Since candidate probabilities are non-negative,

$$p_i \geq 0, \tag{42}$$

we have

$$\begin{aligned} P_{\mathcal{V}} &= \sum_{i \in \mathcal{V}(x)} p_i \\ &= p_r + \sum_{\substack{i \in \mathcal{V}(x) \\ i \neq r}} p_i \\ &\geq p_r. \end{aligned} \tag{43}$$

Adding ϵ to both sides gives

$$P_{\mathcal{V}} + \epsilon \geq p_r + \epsilon. \tag{44}$$

Because the logarithm is monotonically increasing,

$$\log(P_{\mathcal{V}} + \epsilon) \geq \log(p_r + \epsilon). \tag{45}$$

Multiplying by -1 reverses the inequality:

$$-\log(P_{\mathcal{V}} + \epsilon) \leq -\log(p_r + \epsilon). \tag{46}$$

Therefore,

$$\mathcal{L}_{\text{QMulti}} \leq \mathcal{L}_{\text{QSingle}}. \tag{47}$$

The difference between the two objectives can be written explicitly as

$$\begin{aligned} \mathcal{L}_{\text{QSingle}} - \mathcal{L}_{\text{QMulti}} &= -\log(p_r + \epsilon) + \log(P_{\mathcal{V}} + \epsilon) \\ &= \log(P_{\mathcal{V}} + \epsilon) - \log(p_r + \epsilon) \\ &= \log\left(\frac{P_{\mathcal{V}} + \epsilon}{p_r + \epsilon}\right) \\ &\geq 0. \end{aligned} \tag{48}$$

Equality occurs precisely when

$$P_{\mathcal{V}} = p_r, \tag{49}$$

which, because all probabilities are non-negative, is equivalent to

$$\sum_{\substack{i \in \mathcal{V}(x) \\ i \neq r}} p_i = 0. \tag{50}$$

Hence the multi-valid objective does not introduce an additional penalty for probability assigned to another verified valid interpretation. □

A.3 Proof of Theorem 4.3

Proof. We prove the result in seven steps.

Step 1: Norm of the interaction generator. For candidate pair (i, j), recall

$$T_{ij} = |i-1\rangle\langle j-1| + |j-1\rangle\langle i-1|. \tag{51}$$

The operator is Hermitian:

$$\begin{aligned} T_{ij}^\dagger &= (|i-1\rangle\langle j-1|)^\dagger + (|j-1\rangle\langle i-1|)^\dagger \\ &= |j-1\rangle\langle i-1| + |i-1\rangle\langle j-1| \\ &= T_{ij}. \end{aligned} \tag{52}$$

Moreover,

$$\begin{aligned} T_{ij}^2 &= (|i-1\rangle\langle j-1| + |j-1\rangle\langle i-1|)^2 \\ &= |i-1\rangle\langle i-1| + |j-1\rangle\langle j-1|. \end{aligned} \tag{53}$$

Thus T_{ij}^2 is the projector onto the two-dimensional subspace spanned by $|i-1\rangle$ and $|j-1\rangle$. Consequently,

$$\|T_{ij}\|_2 = 1. \tag{54}$$

Each P_r is a Pauli operator or Pauli string, so

$$P_r^\dagger P_r = I \tag{55}$$

and therefore

$$\|P_r\|_2 = 1. \tag{56}$$

For

$$A_{ijr} = T_{ij} \otimes P_r, \tag{57}$$

the multiplicativity of the operator norm under tensor products gives

$$\begin{aligned} \|A_{ijr}\|_2 &= \|T_{ij} \otimes P_r\|_2 \\ &= \|T_{ij}\|_2 \|P_r\|_2 \\ &= 1 \cdot 1 \\ &= 1. \end{aligned} \tag{58}$$

Step 2: Perturbation of one interaction factor. For a fixed layer ℓ, pair (i, j), and relation r, define

$$U_{ijr}^{(\ell)} = \exp\left[-i\theta_r^{(\ell)} d_{ij}^{(r)} A_{ijr}\right] \tag{59}$$

and

$$\widetilde{U}_{ijr}^{(\ell)} = \exp\left[-i\theta_r^{(\ell)} \widetilde{d}_{ij}^{(r)} A_{ijr}\right]. \tag{60}$$

Since

$$\widetilde{d}_{ij}^{(r)} = d_{ij}^{(r)} + \xi_{ij}^{(r)}, \tag{61}$$

we obtain

$$\begin{aligned}
\widetilde{U}_{ijr}^{(\ell)} &= \exp\left[-i\theta_r^{(\ell)} \left(d_{ij}^{(r)} + \xi_{ij}^{(r)}\right) A_{ijr}\right] \\
&= \exp\left[-i\theta_r^{(\ell)} d_{ij}^{(r)} A_{ijr}\right] \exp\left[-i\theta_r^{(\ell)} \xi_{ij}^{(r)} A_{ijr}\right] \\
&= U_{ijr}^{(\ell)} \exp\left[-i\theta_r^{(\ell)} \xi_{ij}^{(r)} A_{ijr}\right].
\end{aligned} \tag{62}$$

The factorization is exact because both exponentials are generated by the same operator A_{ijr} and therefore commute.

Hence

$$\begin{aligned}
\left\| U_{ijr}^{(\ell)} - \widetilde{U}_{ijr}^{(\ell)} \right\|_2 &= \left\| U_{ijr}^{(\ell)} \left[I - \exp\left(-i\theta_r^{(\ell)} \xi_{ij}^{(r)} A_{ijr}\right)\right] \right\|_2 \\
&= \left\| I - \exp\left(-i\theta_r^{(\ell)} \xi_{ij}^{(r)} A_{ijr}\right) \right\|_2.
\end{aligned} \tag{63}$$

The second equality follows because multiplication by a unitary preserves the operator norm.

Let

$$A_{ijr} = \sum_{\lambda} \lambda \Pi_\lambda \tag{64}$$

be the spectral decomposition of A_{ijr}. Then

$$\begin{aligned}
\left\| I - e^{-i\theta_r^{(\ell)} \xi_{ij}^{(r)} A_{ijr}} \right\|_2 &= \max_\lambda \left| 1 - e^{-i\theta_r^{(\ell)} \xi_{ij}^{(r)} \lambda} \right| \\
&= 2 \max_\lambda \left| \sin\left(\frac{\theta_r^{(\ell)} \xi_{ij}^{(r)} \lambda}{2}\right) \right| \\
&\leq 2 \max_\lambda \left| \frac{\theta_r^{(\ell)} \xi_{ij}^{(r)} \lambda}{2} \right| \\
&= |\theta_r^{(\ell)}||\xi_{ij}^{(r)}| \max_\lambda |\lambda| \\
&= |\theta_r^{(\ell)}||\xi_{ij}^{(r)}| \|A_{ijr}\|_2 \\
&= |\theta_r^{(\ell)}||\xi_{ij}^{(r)}|.
\end{aligned} \tag{65}$$

We used $|\sin z| \leq |z|$ and equation 58.

Therefore,

$$\boxed{\left\| U_{ijr}^{(\ell)} - \widetilde{U}_{ijr}^{(\ell)} \right\|_2 \leq |\theta_r^{(\ell)}||\xi_{ij}^{(r)}|}. \tag{66}$$

Step 3: Perturbation of the complete product-form circuit. Flatten all interaction factors across the L reasoning layers into the fixed execution order used by the implementation. Let

$$N = LR\binom{K}{2} \tag{67}$$

be the total number of interaction factors. Write

$$\mathcal{U} = U_N U_{N-1} \cdots U_1 \tag{68}$$

and

$$\widetilde{\mathcal{U}} = \widetilde{U}_N \widetilde{U}_{N-1} \cdots \widetilde{U}_1. \tag{69}$$

The difference between the products has the telescoping expansion

$$\begin{aligned}
\mathcal{U} - \widetilde{\mathcal{U}} &= U_N U_{N-1} \cdots U_2 \left(U_1 - \widetilde{U}_1\right) \\
&\quad + U_N U_{N-1} \cdots U_3 \left(U_2 - \widetilde{U}_2\right) \widetilde{U}_1 \\
&\quad + \cdots \\
&\quad + \left(U_N - \widetilde{U}_N\right) \widetilde{U}_{N-1} \cdots \widetilde{U}_1 \\
&= \sum_{m=1}^{N} U_N \cdots U_{m+1} \left(U_m - \widetilde{U}_m\right) \widetilde{U}_{m-1} \cdots \widetilde{U}_1.
\end{aligned} \tag{70}$$

Taking the operator norm and applying the triangle inequality,

$$\begin{aligned}
\left\|\mathcal{U} - \widetilde{\mathcal{U}}\right\|_2 &\le \sum_{m=1}^{N} \left\|U_N \cdots U_{m+1} \left(U_m - \widetilde{U}_m\right) \widetilde{U}_{m-1} \cdots \widetilde{U}_1\right\|_2 \\
&\le \sum_{m=1}^{N} \|U_N\|_2 \cdots \|U_{m+1}\|_2 \left\|U_m - \widetilde{U}_m\right\|_2 \\
&\qquad \times \|\widetilde{U}_{m-1}\|_2 \cdots \|\widetilde{U}_1\|_2.
\end{aligned} \tag{71}$$

Every factor is unitary, so

$$\|U_m\|_2 = \|\widetilde{U}_m\|_2 = 1. \tag{72}$$

Hence

$$\begin{aligned}
\left\|\mathcal{U} - \widetilde{\mathcal{U}}\right\|_2 &\le \sum_{m=1}^{N} \left\|U_m - \widetilde{U}_m\right\|_2 \\
&\le \sum_{\ell=1}^{L} \sum_{1 \le i < j \le K} \sum_{r=1}^{R} |\theta_r^{(\ell)}| |\xi_{ij}^{(r)}| \\
&= E_D.
\end{aligned} \tag{73}$$

Step 4: Final-state perturbation. Both circuits act on the same normalized initial state $|\Psi_0\rangle$. Therefore,

$$\begin{aligned}
\left\||\Psi_L\rangle - |\widetilde{\Psi}_L\rangle\right\|_2 &= \left\|\mathcal{U}|\Psi_0\rangle - \widetilde{\mathcal{U}}|\Psi_0\rangle\right\|_2 \\
&= \left\|\left(\mathcal{U} - \widetilde{\mathcal{U}}\right)|\Psi_0\rangle\right\|_2 \\
&\le \left\|\mathcal{U} - \widetilde{\mathcal{U}}\right\|_2 \||\Psi_0\rangle\|_2 \\
&= \left\|\mathcal{U} - \widetilde{\mathcal{U}}\right\|_2 \\
&\le E_D.
\end{aligned} \tag{74}$$

Step 5: Candidate-probability perturbation. Define

$$\rho = |\Psi_L\rangle\langle\Psi_L| \tag{75}$$

and

$$\widetilde{\rho} = |\widetilde{\Psi}_L\rangle\langle\widetilde{\Psi}_L|. \tag{76}$$

Their trace distance is

$$D_{\mathrm{tr}}(\rho, \widetilde{\rho}) = \frac{1}{2} \|\rho - \widetilde{\rho}\|_1. \tag{77}$$

For pure states,

$$D_{\mathrm{tr}}\left(\rho,\widetilde{\rho}\right)=\sqrt{1-\left|\langle\Psi_L|\widetilde{\Psi}_L\rangle\right|^2}. \tag{78}$$

Let

$$c=\langle\Psi_L|\widetilde{\Psi}_L\rangle. \tag{79}$$

Then

$$\begin{aligned} D_{\mathrm{tr}}^2 &= 1-|c|^2 \\ &= (1-|c|)(1+|c|) \\ &\le 2(1-|c|) \\ &\le 2(1-\operatorname{Re}c). \end{aligned} \tag{80}$$

On the other hand,

$$\begin{aligned} \left\||\Psi_L\rangle-|\widetilde{\Psi}_L\rangle\right\|_2^2 &= \langle\Psi_L|\Psi_L\rangle+\langle\widetilde{\Psi}_L|\widetilde{\Psi}_L\rangle \\ &\quad -\langle\Psi_L|\widetilde{\Psi}_L\rangle-\langle\widetilde{\Psi}_L|\Psi_L\rangle \\ &= 2-c-c^* \\ &= 2-2\operatorname{Re}c \\ &= 2(1-\operatorname{Re}c). \end{aligned} \tag{81}$$

Combining the two bounds,

$$\begin{aligned} D_{\mathrm{tr}}\left(\rho,\widetilde{\rho}\right) &\le \left\||\Psi_L\rangle-|\widetilde{\Psi}_L\rangle\right\|_2 \\ &\le E_D. \end{aligned} \tag{82}$$

Candidate probabilities are obtained by measuring the candidate register. Measurement cannot increase distinguishability, so

$$\begin{aligned} \mathrm{TV}\left(\mathbf{p},\widetilde{\mathbf{p}}\right) &= \frac{1}{2}\sum_i |p_i-\widetilde{p}_i| \\ &\le D_{\mathrm{tr}}\left(\rho,\widetilde{\rho}\right) \\ &\le E_D. \end{aligned} \tag{83}$$

Since total variation distance is at most one,

$$\boxed{\mathrm{TV}\left(\mathbf{p},\widetilde{\mathbf{p}}\right)\le\min\{1,E_D\}}. \tag{84}$$

For any individual candidate,

$$\begin{aligned} |p_i-\widetilde{p}_i| &\le \mathrm{TV}\left(\mathbf{p},\widetilde{\mathbf{p}}\right) \\ &\le E_D. \end{aligned} \tag{85}$$

Similarly,

$$\begin{aligned} |P_{\mathcal{V}}-\widetilde{P}_{\mathcal{V}}| &= \left|\sum_{i\in\mathcal{V}(x)} p_i-\sum_{i\in\mathcal{V}(x)}\widetilde{p}_i\right| \\ &\le \mathrm{TV}\left(\mathbf{p},\widetilde{\mathbf{p}}\right) \\ &\le E_D. \end{aligned} \tag{86}$$

Step 6: Multi-valid loss stability. Suppose

$$P_{\mathcal{V}}\ge\mu \tag{87}$$

and

$$\widetilde{P}_{\mathcal{V}}\ge\mu. \tag{88}$$

Then

$$\begin{aligned}\left|\mathcal{L}_{\mathrm{QMulti}}-\widetilde{\mathcal{L}}_{\mathrm{QMulti}}\right| &= \left|\log(P_{\mathcal{V}}+\epsilon)-\log(\widetilde{P}_{\mathcal{V}}+\epsilon)\right| \\ &= \left|\int_{\widetilde{P}_{\mathcal{V}}}^{P_{\mathcal{V}}} \frac{1}{u+\epsilon}\,du\right| \\ &\le \frac{|P_{\mathcal{V}}-\widetilde{P}_{\mathcal{V}}|}{\mu+\epsilon} \\ &\le \frac{E_D}{\mu+\epsilon}. \end{aligned} \tag{89}$$

Step 7: Preservation of the selected candidate. Let

$$i^\star = \arg\max_i p_i \tag{90}$$

and define

$$\Delta = p_{i^\star} - \max_{j\neq i^\star} p_j. \tag{91}$$

For any competitor $j \neq i^\star$,

$$\begin{aligned}\widetilde{p}_{i^\star}-\widetilde{p}_j &= p_{i^\star}-p_j+(\widetilde{p}_{i^\star}-p_{i^\star})-(\widetilde{p}_j-p_j) \\ &\ge p_{i^\star}-p_j-|\widetilde{p}_{i^\star}-p_{i^\star}|-|\widetilde{p}_j-p_j| \\ &\ge p_{i^\star}-p_j-E_D-E_D \\ &= p_{i^\star}-p_j-2E_D \\ &\ge \Delta-2E_D. \end{aligned} \tag{92}$$

If

$$\Delta > 2E_D, \tag{93}$$

then

$$\widetilde{p}_{i^\star}-\widetilde{p}_j > 0 \tag{94}$$

for every $j \neq i^\star$. Therefore,

$$\arg\max_i \widetilde{p}_i = i^\star. \tag{95}$$

This completes the proof. □

A.4 Proof of Theorem 4.4

Proof. Let

$$i^\star = \arg\max_i p_i \tag{96}$$

be the unique exact-state prediction. For each competitor $j \neq i^\star$, define

$$\Delta_j = p_{i^\star} - p_j > 0. \tag{97}$$

Consider M independent measurements of the candidate register. Let Y_m denote the candidate outcome of shot m, where

$$\Pr[Y_m = i] = p_i. \tag{98}$$

For a fixed competitor j, define

$$Z_m^{(j)} = \mathbf{1}\{Y_m = i^\star\} - \mathbf{1}\{Y_m = j\}. \tag{99}$$

Then

$$Z_m^{(j)} \in \{-1, 0, 1\}. \tag{100}$$

Its expectation is

$$\begin{aligned}\mathbb{E}\left[Z_m^{(j)}\right] &= \mathbb{E}\left[\mathbf{1}\{Y_m = i^\star\}\right] - \mathbb{E}\left[\mathbf{1}\{Y_m = j\}\right] \\ &= \Pr[Y_m = i^\star] - \Pr[Y_m = j] \\ &= p_{i^\star} - p_j \\ &= \Delta_j. \end{aligned} \tag{101}$$

The empirical probability difference satisfies

$$\begin{aligned}\widehat{p}_{i^\star} - \widehat{p}_j &= \frac{N_{i^\star}}{M} - \frac{N_j}{M} \\ &= \frac{1}{M}\sum_{m=1}^{M} \mathbf{1}\{Y_m = i^\star\} - \frac{1}{M}\sum_{m=1}^{M} \mathbf{1}\{Y_m = j\} \\ &= \frac{1}{M}\sum_{m=1}^{M} \left[\mathbf{1}\{Y_m = i^\star\} - \mathbf{1}\{Y_m = j\}\right] \\ &= \frac{1}{M}\sum_{m=1}^{M} Z_m^{(j)}. \end{aligned} \tag{102}$$

Candidate j can defeat or tie $i^\star$ only if

$$\widehat{p}_j \geq \widehat{p}_{i^\star}. \tag{103}$$

Using equation 102,

$$\begin{aligned}\Pr\left[\widehat{p}_j \geq \widehat{p}_{i^\star}\right] &= \Pr\left[\frac{1}{M}\sum_{m=1}^{M} Z_m^{(j)} \leq 0\right] \\ &= \Pr\left[\frac{1}{M}\sum_{m=1}^{M} Z_m^{(j)} - \Delta_j \leq -\Delta_j\right]. \end{aligned} \tag{104}$$

Since the M shots are independent and $Z_m^{(j)} \in [-1, 1]$, Hoeffding's inequality gives

$$\begin{aligned}\Pr\left[\frac{1}{M}\sum_{m=1}^{M} Z_m^{(j)} - \Delta_j \leq -\Delta_j\right] &\leq \exp\left[-\frac{2M^2\Delta_j^2}{\sum_{m=1}^{M}(1-(-1))^2}\right] \\ &= \exp\left[-\frac{2M^2\Delta_j^2}{4M}\right] \\ &= \exp\left(-\frac{M\Delta_j^2}{2}\right). \end{aligned} \tag{105}$$

Therefore,

$$\Pr\left[\widehat{p}_j \geq \widehat{p}_{i^\star}\right] \leq \exp\left(-\frac{M\Delta_j^2}{2}\right). \tag{106}$$

If finite-shot inference differs from exact inference, at least one competitor must equal or exceed the empirical probability of $i^\star$. Hence

$$\{\hat{s}_{\text{shot}} \neq \hat{s}_{\text{exact}}\} \subseteq \bigcup_{j \neq i^\star} \{\widehat{p}_j \geq \widehat{p}_{i^\star}\}. \tag{107}$$

Applying the union bound,

$$\begin{aligned}\Pr\left[\hat{s}_{\text{shot}} \neq \hat{s}_{\text{exact}}\right] &\leq \sum_{j \neq i^\star} \Pr\left[\widehat{p}_j \geq \widehat{p}_{i^\star}\right] \\ &\leq \sum_{j \neq i^\star} \exp\left(-\frac{M\Delta_j^2}{2}\right). \end{aligned} \tag{108}$$

Define

$$\Delta = \min_{j \neq i^\star} \Delta_j. \tag{109}$$

Then $\Delta_j \geq \Delta$ for every $j \neq i^\star$, and therefore

$$\sum_{j \neq i^\star} \exp\left(-\frac{M\Delta_j^2}{2}\right) \leq \sum_{j \neq i^\star} \exp\left(-\frac{M\Delta^2}{2}\right) = (K-1)\exp\left(-\frac{M\Delta^2}{2}\right). \tag{110}$$

Combining the bounds gives

$$\boxed{\Pr\left[\hat{s}_{\text{shot}} \neq \hat{s}_{\text{exact}}\right] \leq \sum_{j \neq i^\star} \exp\left(-\frac{M\Delta_j^2}{2}\right) \leq (K-1)\exp\left(-\frac{M\Delta^2}{2}\right)}. \tag{111}$$

The use of the non-strict event $\widehat{p}_j \geq \widehat{p}_{i^\star}$ also covers empirical ties. □

A.5 Proof of Corollary 4.5

Proof. From Theorem 4.4,

$$\Pr\left[\hat{s}_{\text{shot}} \neq \hat{s}_{\text{exact}}\right] \leq (K-1)\exp\left(-\frac{M\Delta^2}{2}\right). \tag{112}$$

To guarantee failure probability at most δ, it is sufficient that

$$(K-1)\exp\left(-\frac{M\Delta^2}{2}\right) \leq \delta. \tag{113}$$

Solving for M,

$$\begin{aligned}
(K-1)\exp\left(-\frac{M\Delta^2}{2}\right) &\leq \delta \\
\exp\left(-\frac{M\Delta^2}{2}\right) &\leq \frac{\delta}{K-1} \\
-\frac{M\Delta^2}{2} &\leq \log\left(\frac{\delta}{K-1}\right) \\
&= -\log\left(\frac{K-1}{\delta}\right) \\
\frac{M\Delta^2}{2} &\geq \log\left(\frac{K-1}{\delta}\right) \\
M &\geq \frac{2}{\Delta^2}\log\left(\frac{K-1}{\delta}\right).
\end{aligned} \tag{114}$$

Therefore,

$$\Pr\left[\hat{s}_{\text{shot}} = \hat{s}_{\text{exact}}\right] \geq 1-\delta. \tag{115}$$

Ignoring constants,

$$M = \mathcal{O}\left(\frac{1}{\Delta^2}\log\frac{K}{\delta}\right). \tag{116}$$

□

A.6 Proof of Corollary 4.6

Proof. Let

$$\mathcal{V} = \mathcal{V}(x) \tag{117}$$

and define the invalid set

$$\mathcal{I} = \{1, \ldots, K\} \setminus \mathcal{V}. \tag{118}$$

If $\mathcal{I} = \varnothing$, every candidate is valid and

$$\Pr\left[\hat{s}_{\text{shot}} \notin \mathcal{S}_{\text{valid}}(x)\right] = 0. \tag{119}$$

Otherwise, let

$$i^\star_{\mathcal{V}} = \arg\max_{i \in \mathcal{V}} p_i \tag{120}$$

and define

$$\Delta_{\mathcal{V}} = p_{i^\star_{\mathcal{V}}} - \max_{j \in \mathcal{I}} p_j > 0. \tag{121}$$

For each $j \in \mathcal{I}$, define

$$\Delta_{\mathcal{V},j} = p_{i^\star_{\mathcal{V}}} - p_j. \tag{122}$$

Then

$$\begin{aligned} \Delta_{\mathcal{V},j} &= p_{i^\star_{\mathcal{V}}} - p_j \\ &\geq p_{i^\star_{\mathcal{V}}} - \max_{m \in \mathcal{I}} p_m \\ &= \Delta_{\mathcal{V}}. \end{aligned} \tag{123}$$

Define

$$Z_m^{(j)} = \mathbf{1}\{Y_m = i^\star_{\mathcal{V}}\} - \mathbf{1}\{Y_m = j\}. \tag{124}$$

Then

$$\begin{aligned} \mathbb{E}\left[Z_m^{(j)}\right] &= p_{i^\star_{\mathcal{V}}} - p_j \\ &= \Delta_{\mathcal{V},j}. \end{aligned} \tag{125}$$

Applying the same Hoeffding argument,

$$\begin{aligned} \Pr\left[\widehat{p}_j \geq \widehat{p}_{i^\star_{\mathcal{V}}}\right] &\leq \exp\left(-\frac{M\Delta^2_{\mathcal{V},j}}{2}\right) \\ &\leq \exp\left(-\frac{M\Delta^2_{\mathcal{V}}}{2}\right). \end{aligned} \tag{126}$$

If the finite-shot winner is invalid, at least one invalid candidate must equal or exceed the empirical probability of $i^\star_{\mathcal{V}}$. Therefore,

$$\begin{aligned} \Pr\left[\hat{s}_{\text{shot}} \notin \mathcal{S}_{\text{valid}}(x)\right] &\leq \sum_{j \in \mathcal{I}} \Pr\left[\widehat{p}_j \geq \widehat{p}_{i^\star_{\mathcal{V}}}\right] \\ &\leq \sum_{j \in \mathcal{I}} \exp\left(-\frac{M\Delta^2_{\mathcal{V}}}{2}\right) \\ &= |\mathcal{I}| \exp\left(-\frac{M\Delta^2_{\mathcal{V}}}{2}\right) \\ &= (K - |\mathcal{V}(x)|) \exp\left(-\frac{M\Delta^2_{\mathcal{V}}}{2}\right). \end{aligned} \tag{127}$$

This completes the proof. □

A.7 Combined Disagreement and Finite-Shot Stability

The previous results isolate two distinct sources of uncertainty: perturbations of the structural disagreement representation and finite-shot measurement. They can also be combined into a single bound.

Proposition A.1 (Combined structural and sampling stability)**.** *Let $i^\star = \arg\max_i p_i$ be the unique prediction under the unperturbed exact circuit with margin*

$$\Delta = p_{i^\star} - \max_{j \neq i^\star} p_j. \tag{128}$$

Suppose

$$E_D < \frac{\Delta}{2}. \tag{129}$$

If M shots are then drawn from the perturbed circuit,

$$\Pr\left[\hat{s}_{\text{shot}}^{\text{pert}} \neq i^\star\right] \leq (K-1)\exp\left[-\frac{M(\Delta - 2E_D)^2}{2}\right]. \tag{130}$$

Proof. From Theorem 4.3,

$$|\widetilde{p}_i - p_i| \leq E_D \tag{131}$$

for every candidate i. Therefore, for any $j \neq i^\star$,

$$\begin{aligned}\widetilde{p}_{i^\star} - \widetilde{p}_j &= p_{i^\star} - p_j + (\widetilde{p}_{i^\star} - p_{i^\star}) - (\widetilde{p}_j - p_j) \\ &\geq p_{i^\star} - p_j - |\widetilde{p}_{i^\star} - p_{i^\star}| - |\widetilde{p}_j - p_j| \\ &\geq \Delta - 2E_D. \end{aligned} \tag{132}$$

Since $E_D < \Delta/2$,

$$\Delta - 2E_D > 0. \tag{133}$$

Applying Theorem 4.4 to the perturbed distribution yields

$$\Pr\left[\hat{s}_{\text{shot}}^{\text{pert}} \neq i^\star\right] \leq (K-1)\exp\left[-\frac{M(\Delta - 2E_D)^2}{2}\right]. \tag{134}$$

This proves the proposition. □

Proposition A.1 shows that disagreement perturbations reduce the guaranteed candidate margin from Δ to at least $\Delta - 2E_D$, after which the finite-shot concentration result applies.

A.8 Scope of the Theoretical Results

Fixed candidate set. The disagreement-stability analysis compares two executions containing the same K active candidates. It does not analyze perturbations that alter the candidate-generation process or replace one candidate with another.

Fixed candidate representations. Theorem 4.3 perturbs the pairwise disagreement coefficients $d_{ij}^{(r)}$ while holding the initial candidate-feature states $|\phi_i\rangle_f$ fixed. Thus, E_D characterizes sensitivity of the ambiguity-reasoning module rather than errors that simultaneously modify candidate generation, graph encoding, and state preparation.

Fixed interaction ordering. The unperturbed and perturbed circuits use the same lexicographic product ordering over candidate pairs and relation types. Because distinct interaction generators need not commute, the theorem does not claim permutation invariance of the product-form implementation.

Fixed learned parameters. The disagreement analysis holds $\theta_r^{(\ell)}$ fixed. It therefore characterizes inference-time sensitivity of a trained model rather than the effect of disagreement perturbations throughout optimization.

Independent finite-shot measurements. Theorem 4.4 and Corollary 4.6 assume that the M measurement outcomes are independent draws from a fixed candidate probability distribution.

Hardware noise. Finite-shot uncertainty and hardware noise are distinct. Finite-shot measurement estimates a fixed underlying probability distribution using a finite number of samples, whereas gate and readout errors may change the underlying probability distribution itself.

B Quantum Circuit Construction

This section realizes the joint quantum representation and cross-candidate ambiguity reasoning presented in Section 3 at the circuit level. It describes the candidate and feature registers, explicit joint-state preparation, relational feature encoding, disagreement-conditioned interactions, circuit decomposition, and candidate measurement. The implementation uses standard controlled quantum operations and does not assume quantum random-access memory (QRAM) or constant-cost arbitrary state loading.

B.1 Candidate and Feature Registers

We configure the model to accommodate at most $k_{\max}$ candidate structures. For a given input x, preprocessing retains

$$\mathcal{S}(x) = s_1, \ldots, s_K, \qquad K = |\mathcal{S}(x)| \leq k_{\max}, \tag{135}$$

where K is the number of active candidates. We assign q_c qubits to the candidate register and q_f qubits to the feature register, with

$$q_c = \lceil \log_2 k_{\max} \rceil . \tag{136}$$

The candidate register has capacity for the configured candidate set, while the feature register holds quantum-encoded relational data. The computational-basis state $|i-1\rangle_c$ corresponds to candidate s_i. Under the default setting $k_{\max} = 5$, three candidate qubits suffice:

$$q_c = \lceil \log_2 5 \rceil = 3. \tag{137}$$

For an input with K retained candidates, only the first K computational-basis states are active. We define the *active candidate subspace* as

$$\mathcal{H} * \mathrm{act}(x) = \mathrm{span}\left\{|0\rangle, \ldots, |K-1\rangle\right\}. \tag{138}$$

The remaining basis states are inactive. The candidate-transition operators defined below act only within $\mathcal{H} * \mathrm{act}(x)$, and inactive states have zero amplitude after ideal state preparation. Any probability that leaks into inactive states during noisy execution is retained as invalid probability mass rather than removed by post-selection. The total number of logical qubits is therefore

$$q_{\mathrm{total}} = q_c + q_f + q_a, \tag{139}$$

where q_a denotes optional ancilla qubits required by a specific circuit decomposition.

B.2 Explicit Joint-State Preparation

We first prepare a candidate-register state over the K active interpretations:

$$|\psi_c\rangle = \sum_{i=1}^{K} \sqrt{\pi_i}, |i-1\rangle_c, \qquad \sum_{i=1}^{K} \pi_i = 1. \tag{140}$$

Unless otherwise noted, we use a uniform prior over the active candidates,

$$\pi_i = \frac{1}{K}, \qquad i \in 1, \ldots, K. \tag{141}$$

This prevents the candidate generator's initial choice from carrying over to the ambiguity-reasoning step. When K is not a power of two, Hadamard gates alone cannot produce the desired active-state superposition. We therefore synthesize equation 140 using a limited sequence of controlled rotations. Because the candidate sets considered in this work are small, this preparation remains tractable and does not require QRAM. We then control a feature-encoding operation on each active candidate basis state. Let $U_{\mathrm{enc}}(\tilde{\mathbf{g}} * i)$ denote the encoder associated with candidate s_i, and let

$$\Pi * \mathrm{inactive} = I_c - \sum_{i=1}^{K} |i-1\rangle\langle i-1| \tag{142}$$

project onto the unused candidate basis states. The full preparation operator is

$$U_{\mathrm{prep}} = \sum_{i=1}^{K} |i-1\rangle\langle i-1| * c \otimes U * \mathrm{enc}(\tilde{\mathbf{g}} * i) \quad + \Pi * \mathrm{inactive} \otimes I_f. \tag{143}$$

Applying U_{prep} yields

$$|\Psi_0(x)\rangle = U_{\text{prep}}\left(|\psi_c\rangle \otimes |0\rangle_f^{\otimes q_f}\right) \quad = \sum_{i=1}^{K} \sqrt{\pi_i}, |i-1\rangle_c |\phi_i\rangle_f, \tag{144}$$

where $|\phi_i\rangle_f$ is the quantum encoding of the relational representation of candidate graph G_i. The probability mass is normalized over the interpretations active for the input, yielding the joint candidate state described in Section 3.2.

B.3 Relational Feature Encoding

The classical relational encoder produces a projected representation $\tilde{\mathbf{g}} * i \in \mathbb{R}^{d_q}$ for each candidate graph G_i. Before quantum encoding, each feature dimension is normalized to a finite interval and interpreted as a rotation angle. We use *angle encoding* because it yields a compact circuit without amplitude loading. Let $\mathbf{z} * i = [z * i, 1, \ldots, z * i, d_q]$ denote the normalized feature vector. One encoding block is defined as

$$U_{\text{enc}}^{(b)}(\mathbf{z} * i) = \prod * m = 1^{q_f} R_Z\left(z_{i,2m}^{(b)}\right) R_Y\left(z_{i,2m-1}^{(b)}\right). \tag{145}$$

When $d_q > 2q_f$, the feature vector is divided into B blocks and encoded through *feature reuploading*:

$$U_{\text{enc}}(\mathbf{z} * i) = \prod * b = 1^{B} U_{\text{enc}}^{(b)}(\mathbf{z}_i^{(b)}). \tag{146}$$

Zero padding is applied if the final block contains fewer than $2q_f$ values. This design keeps the feature-register size fixed even when the projected relational representation has higher dimensionality. The same encoding procedure is used for all candidates and their corresponding quantum baselines to ensure a controlled comparison.

B.4 Structural Disagreement Features

The joint representation alone does not specify how competing interpretations differ. We therefore compute explicit relational disagreement signals for each pair of candidate graphs. For relation category r, let $E_i^{(r)}$ be the set of canonicalized relations of type r in candidate graph G_i. We define the normalized disagreement as

$$d_{ij}^{(r)} = 1 - \frac{2\left|E_i^{(r)} \cap E_j^{(r)}\right|}{\left|E_i^{(r)}\right| + \left|E_j^{(r)}\right|}. \tag{147}$$

If both relation sets are empty, we set $d_{ij}^{(r)} = 0$. The disagreement measure satisfies

$$0 \le d_{ij}^{(r)} \le 1, \qquad d_{ij}^{(r)} = d_{ji}^{(r)}. \tag{148}$$

A value near zero indicates strong structural agreement, while a higher value indicates greater disagreement for relation type r. For Spider, the relation categories include join structure, grouping, aggregation, condition attachment, and value relations. For nvBench, they include attribute selection, encoding assignment, aggregation, grouping, and filtering. We canonicalize relations before comparison so that the measure captures structural differences rather than serialization order. The complete disagreement vector is

$$\mathbf{d} * ij = \left[d * ij^{(1)}, \ldots, d_{ij}^{(R)}\right]. \tag{149}$$

These features are computed classically and condition the strength of cross-candidate quantum interactions.

B.5 Disagreement-Conditioned Quantum Interactions

For an input with K active candidates, each pair (i, j), where $1 \le i < j \le K$, is associated with the Hermitian two-level transition operator

$$T_{ij} = |i-1\rangle\langle j-1| + |j-1\rangle\langle i-1|. \tag{150}$$

This operator is zero outside the two-dimensional candidate subspace spanned by s_i and s_j. Each disagreement category r is also associated with a fixed Hermitian Pauli operator P_r acting on the feature register:

$$\begin{aligned} P_1 &= X_1, \quad P_2 = Y_1, \qquad P_3 = X_2, \\ P_4 &= Y_2, \quad P_5 = X_1 X_2, \end{aligned} \tag{151}$$

where X_m and Y_m act on feature qubit m and as identity on all other feature qubits. For all feature-register sizes considered in our experiments, $q_f \geq 2$, and the operator bank has a maximum Pauli weight of two. The five relation indices map to the task-specific disagreement categories defined during graph comparison:

r	Spider	nvBench	P_r
1	Join structure	Attribute selection	X_1
2	Grouping	Encoding assignment	Y_1
3	Aggregation	Aggregation	X_2
4	Condition attachment	Grouping	Y_2
5	Value relations	Filtering	X_1X_2

The Pauli assignment is fixed before training and shared across candidate pairs, datasets, random seeds, and matching quantum baselines. The disagreement value $d_{ij}^{(r)}$ carries task-specific information, while P_r provides a distinct low-weight quantum generator for each relation index. The assignment is not intended to encode semantic similarity between relation categories across tasks. At reasoning layer ℓ, candidate pair (i, j) and relation type r induce the interaction

$$U_{ijr}^{(\ell)} = \exp\left[-i\theta_r^{(\ell)} d_{ij}^{(r)} T_{ij} \otimes P_r\right], \tag{152}$$

where $\theta_r^{(\ell)}$ is trainable and shared across all active candidate pairs. This relation-based parameter sharing keeps the number of trainable ambiguity parameters independent of K and prevents an arbitrary candidate index from acquiring a specific semantic role. The corresponding ambiguity Hamiltonian is

$$H_{\text{amb}}^{(\ell)} = \sum_{1 \leq i < j \leq K} \sum_{r=1}^{R} \theta_r^{(\ell)} d_{ij}^{(r)} T_{ij} \otimes P_r. \tag{153}$$

Conceptually, its exact evolution is

$$U_{\text{amb,exact}}^{(\ell)} = \exp\left(-iH_{\text{amb}}^{(\ell)}\right). \tag{154}$$

In our experiments, we instead use the first-order product-form approximation

$$\widetilde{U}_{\text{amb}}^{(\ell)} = \prod_{i=1}^{K-1} \prod_{j=i+1}^{K} \prod_{r=1}^{R} U_{ijr}^{(\ell)}. \tag{155}$$

We use a fixed lexicographic execution order over i, then j, and then r. Training, noisy simulation, finite-shot simulation, exact state-vector evaluation, and physical-device execution all use this ordering. Because the pairwise interaction generators need not commute, the product-form implementation is not fully invariant to candidate ordering. We therefore assess candidate-order sensitivity empirically in Appendix E.7. After L implemented product-form layers, the final state is

$$|\Psi_L(x)\rangle = \widetilde{U}_{\text{amb}}^{(L)} \cdots \widetilde{U}_{\text{amb}}^{(1)} |\Psi_0(x)\rangle. \tag{156}$$

The disagreement terms directly regulate cross-candidate interactions. Greater disagreement increases the magnitude of the effective interaction angle

$$\alpha_{ijr}^{(\ell)} = \theta_r^{(\ell)} d_{ij}^{(r)}, \tag{157}$$

allowing the quantum evolution to respond to conflicting structural decisions while learning the relative importance of each disagreement category through $\theta_r^{(\ell)}$. If candidates i and j agree on relation type r, then $d_{ij}^{(r)} \approx 0$ and

$$U_{ijr}^{(\ell)} \approx I. \tag{158}$$

B.6 Circuit Decomposition

The interaction in equation 152 can be implemented using standard two-level-unitary synthesis. For each candidate pair (i, j), a reversible basis transformation first maps $|i-1\rangle_c$ and $|j-1\rangle_c$ to basis states that differ on a selected target qubit. Candidate-controlled operations then couple this target qubit to the feature-register operator P_r to implement the two-level transition. Finally, the basis transformation is reversed. For a Pauli string

$$P_r = \sigma_{a_1} \otimes \cdots \otimes \sigma_{a_m}, \tag{159}$$

the exponential

$$\exp\left(-i\alpha P_r\right), \qquad \alpha = \theta_r^{(\ell)} d_{ij}^{(r)}, \tag{160}$$

is implemented by transforming non-Z Pauli operators to the Z basis, computing their parity with entangling gates, applying an $R_Z(2\alpha)$ rotation, and reversing the parity and basis transformations. Decomposing multi-controlled candidate operations may require optional ancilla qubits. Appendix C analyzes asymptotic circuit costs and logical interaction counts. Appendix D describes how compiled circuit depth, physical-qubit mapping, and available single- and two-qubit gate counts are recorded for physical-device executions.

B.7 Measurement and Candidate Selection

After joint ambiguity reasoning, the probability of each structured interpretation is given by the candidate-register marginal. For candidate s_i,

$$p_i = \langle \Psi_L | \left(|i-1\rangle\langle i-1|_c \otimes I_f \right) |\Psi_L\rangle. \tag{161}$$

Under finite-shot execution with M measurements, the empirical probability is

$$\hat{p}_i = \frac{N_i}{M}, \tag{162}$$

where N_i is the number of measurements yielding candidate basis state $|i-1\rangle_c$. In ideal state-vector evaluation, all probability mass remains within the active candidate subspace:

$$\sum_{i=1}^{K} p_i = 1. \tag{163}$$

When $K < 2^{q_c}$, noise may cause probability to leak into inactive computational-basis states. We retain this probability as invalid mass rather than renormalizing the active outcomes, avoiding artificial gains from post-selection. The final structured prediction is selected from the active candidates:

$$\hat{s} = \arg \max_{s_i \in \mathcal{S}(x)} \hat{p}_i. \tag{164}$$

For exact state-vector evaluation, the same selection rule uses p_i in place of $\hat{p}_i$. Thus, the candidate-probability definition and selection rule apply to both ideal state-vector and noisy finite-shot evaluations.

C Algorithm and Complexity Analysis

This section describes the full reasoning and inference procedure and analyzes its classical, quantum, and simulation-resource requirements. The framework is hybrid: candidate generation, graph construction, relational encoding, and pairwise disagreement extraction are performed classically, while joint ambiguity reasoning and candidate-probability estimation use the quantum circuit detailed in Appendix B.

C.1 Joint Quantum Ambiguity-Reasoning Algorithm

Given a natural-language input x, the structured generator produces up to $k_{\max}$ candidate structures. After parsing, canonicalization, and duplicate removal, the retained candidate set is

$$\mathcal{S}(x) = \{s_1, \ldots, s_K\}, \qquad K = |\mathcal{S}(x)| \le k_{\max}. \tag{165}$$

Algorithm 1 JOINTQ-MULTI: Joint Quantum Reasoning over Competing Interpretations

Require: Input x; generator q_ϕ; maximum candidate size $k_{\max}$; relational encoder RelEnc; reasoning layers L; relation types R; execution mode mode; measurement shots M

Ensure: Candidate probabilities and prediction $\hat{s}$

1: Generate up to $k_{\max}$ candidate structures using $q_\phi(\cdot \mid x)$
2: Parse, canonicalize, and remove duplicate candidates
3: $\mathcal{S}(x) \leftarrow \{s_1, \ldots, s_K\}$, where $K = |\mathcal{S}(x)| \le k_{\max}$
4: **for** $i = 1$ to K **do**
5: Construct relational graph G_i from s_i
6: $\mathbf{g}_i \leftarrow \mathrm{RelEnc}(G_i, x)$
7: $\tilde{\mathbf{g}}_i \leftarrow W_p \mathbf{g}_i + \mathbf{b}_p$
8: **end for**
9: **for** $i = 1$ to K **do**
10: **for** $j = i + 1$ to K **do**
11: $\mathbf{d}_{ij} \leftarrow D(G_i, G_j, x)$
12: **end for**
13: **end for**
14: Set uniform prior $\pi_i \leftarrow 1/K$ for $i = 1, \ldots, K$
15: Prepare candidate prior
16: $|\psi_c\rangle \leftarrow \sum_{i=1}^{K} \sqrt{\pi_i}\, |i-1\rangle_c$
17: Prepare joint candidate-feature state
18: $|\Psi_0\rangle \leftarrow \sum_{i=1}^{K} \sqrt{\pi_i}\, |i-1\rangle_c U_{\mathrm{enc}}(\tilde{\mathbf{g}}_i)|0\rangle_f^{\otimes q_f}$
19: **for** $\ell = 1$ to L **do**
20: **for** $i = 1$ to K **do**
21: **for** $j = i + 1$ to K **do**
22: **for** $r = 1$ to R **do**
23: Apply $U_{ijr}^{(\ell)} = \exp\Big(-i\theta_r^{(\ell)} d_{ij}^{(r)} T_{ij} \otimes P_r\Big)$
24: **end for**
25: **end for**
26: **end for**
27: **end for**
28: **if** mode $=$ EXACT **then**
29: **for** $i = 1$ to K **do**
30: $p_i \leftarrow \langle \Psi_L | (|i-1\rangle\langle i-1|_c \otimes I_f) | \Psi_L \rangle$
31: $\tilde{p}_i \leftarrow p_i$
32: **end for**
33: **else**
34: Execute the candidate-register measurement using M shots
35: **for** $i = 1$ to K **do**
36: $\tilde{p}_i \leftarrow N_i / M$
37: **end for**
38: **end if**
39: $\hat{s} \leftarrow \arg\max_{s_i \in \mathcal{S}(x)} \tilde{p}_i$
40: **return** $\{\tilde{p}_i\}_{i=1}^{K}, \hat{s}$

Each active candidate is converted into a relational graph G_i, from which a compact representation $\tilde{\mathbf{g}}_i$ is obtained. Pairwise structural disagreements $\mathbf{d}_{ij}$ are then computed across the active candidate graphs. These representations and disagreements determine the initial joint quantum state and the disagreement-conditioned ambiguity-reasoning circuit.

Algorithm 1 summarizes the forward reasoning and candidate-selection procedure. *Lines 1–3* generate the candidate structures, parse and canonicalize them, remove duplicates, and retain at most $k_{\max}$ active candidates. *Lines 4–8* construct a relational graph for each retained candidate, encode the graph, and project its representation into the feature space used by the quantum circuit. *Lines 9–13* compare each distinct pair of candidate graphs and compute the corresponding disagreement vector $\mathbf{d}_{ij}$. *Lines 14–18* prepare the initial joint quantum state. A uniform prior assigns equal weight

to the K active candidates. Candidate-controlled feature encoding then associates each candidate basis state with its encoded relational representation, producing $|\Psi_0(x)\rangle$. *Lines 19–27* apply the ambiguity-reasoning layers. In each layer, the algorithm iterates over candidate pairs and relation types, applying an interaction determined by the disagreement value $d_{ij}^{(r)}$ and trainable parameter $\theta_r^{(\ell)}$. These interactions produce the final state $|\Psi_L(x)\rangle$. *Lines 28–38* obtain candidate probabilities according to the execution mode. In EXACT mode, *Lines 29–32* compute each probability from the candidate-register marginal of $|\Psi_L(x)\rangle$. Otherwise, *Lines 34–37* use M measurement shots to estimate each probability from its observed count. Finally, *Lines 39–40* select and return the active candidate with the highest probability.

In the initial training experiments, we use exact state-vector probabilities to compute the quantum multi-valid objective. For an input with valid candidate index set $\mathcal{I}_{\text{valid}}(x)$,

$$\mathcal{L}_{\text{QMulti}} = -\log\left(\sum_{i\in\mathcal{I}_{\text{valid}}(x)} p_\theta(s_i \mid x, \mathcal{G}(x)) + \epsilon\right). \tag{166}$$

The classical projection parameters and trainable quantum parameters are optimized jointly. Finite-shot and noisy evaluations use the learned parameters and replace exact candidate probabilities with measurement-based estimates under the corresponding execution conditions. Validity annotations are used only for training and evaluation; they are not required during inference.

C.2 CLASSICAL PREPROCESSING COMPLEXITY

We first analyze the classical operations performed before the quantum circuit. For an input x, let $K = |\mathcal{S}(x)| \le k_{\max}$ be the number of active candidates retained after parsing, canonicalization, and duplicate removal. Let $\mathcal{C}_G$ denote the cost of constructing and encoding one candidate graph. Processing all K active candidates therefore costs

$$\mathcal{C}_{\text{graph}} = \mathcal{O}\left(K\mathcal{C}_G\right). \tag{167}$$

The method also computes disagreement features for every pair of active candidate graphs. The number of pairs is

$$N_{\text{pair}} = \binom{K}{2} = \frac{K(K-1)}{2}. \tag{168}$$

Let $\mathcal{C}_D$ denote the average cost of comparing two candidate graphs for one relation category. Across R disagreement categories, the extraction cost is

$$\mathcal{C}_{\text{dis}} = \mathcal{O}\left(RK^2\mathcal{C}_D\right). \tag{169}$$

Combining these steps, the classical preprocessing cost of the ambiguity-reasoning phase is

$$\mathcal{C}_{\text{classical}} = \mathcal{O}\left(K\mathcal{C}_G + RK^2\mathcal{C}_D\right). \tag{170}$$

Since $K \le k_{\max}$, its worst-case cost at the configured candidate capacity is

$$\mathcal{C}_{\text{classical}} = \mathcal{O}\left(k_{\max}\mathcal{C}_G + Rk_{\max}^2\mathcal{C}_D\right). \tag{171}$$

This analysis excludes candidate generation by the base LLM. Candidate generation is performed once, and the resulting candidate pools are stored and reused across the candidate-based reasoning methods. The reported preprocessing complexity therefore covers graph construction and disagreement extraction immediately before joint ambiguity reasoning.

C.3 QUBIT COMPLEXITY

The configured maximum candidate-set size $k_{\max}$ determines the candidate-register size:

$$q_c = \lceil \log_2 k_{\max} \rceil. \tag{172}$$

The register capacity is fixed, although only $K(x) \le k_{\max}$ candidate states are active for a given input. Including q_f feature qubits and q_a optional ancilla qubits, the total logical-qubit requirement is

$$q_{\text{total}} = \lceil \log_2 k_{\max} \rceil + q_f + q_a. \tag{173}$$

Thus, the candidate-register requirement grows logarithmically with the configured candidate capacity:

$$q_c = \mathcal{O}\left(\log k_{\max}\right). \tag{174}$$

In the default configuration, $k_{\max} = 5$ and $q_f = 4$, giving

$$q_c = \lceil \log_2 5 \rceil = 3 \tag{175}$$

and

$$q_{\text{total}} = 7 + q_a, \tag{176}$$

where q_a depends on the chosen decomposition of multi-controlled operations.

C.4 State-Preparation Complexity

Preparing the joint state for an input with K active candidates requires a prior over the K candidate states and conditional encoding of each candidate representation. Let $\mathcal{C}_{\text{enc}}$ denote the logical cost of encoding one candidate feature vector, and let $\mathcal{C}_{\text{prior}}(K, q_c)$ denote the logical cost of preparing the candidate superposition in the q_c-qubit register. The preparation cost is

$$\mathcal{C}_{\text{prep}} = \mathcal{C}_{\text{prior}}(K, q_c) + K\mathcal{C}_{\text{enc}}. \tag{177}$$

For the angle-encoding scheme in Appendix B.3, the number of feature re-uploading blocks is

$$B = \left\lceil \frac{d_q}{2q_f} \right\rceil. \tag{178}$$

Each candidate requires $\mathcal{O}(q_f B)$ data-dependent rotations, so

$$\mathcal{C}_{\text{enc}} = \mathcal{O}\left(q_f B\right). \tag{179}$$

Hence, at the logical-operation level,

$$\mathcal{C}_{\text{prep}} = \mathcal{O}\left(\mathcal{C}_{\text{prior}}(K, q_c) + Kq_f B\right). \tag{180}$$

The number of candidate-controlled feature encodings depends on the active-candidate count K, while the configured capacity $k_{\max}$ determines the candidate-register size through q_c. Physical gate counts also depend on the native gate set, hardware connectivity, and decomposition of controlled operations.

C.5 Cross-Candidate Quantum Interaction Complexity

For an input with K active candidates, each ambiguity-reasoning layer applies one relation-conditioned interaction for every active candidate pair and disagreement category. The number of logical interaction factors per layer is therefore

$$N_{\text{int}}^{(1)} = R\binom{K}{2} = R\frac{K(K-1)}{2}. \tag{181}$$

With L reasoning layers, the implemented product-form circuit contains

$$N_{\text{int}} = LR\frac{K(K-1)}{2}. \tag{182}$$

Consequently,

$$N_{\text{int}} = \mathcal{O}\left(LRK^2\right). \tag{183}$$

Since $K \le k_{\max}$, the worst-case interaction count at the configured candidate capacity is

$$N_{\text{int}} = \mathcal{O}\left(LRk_{\max}^2\right). \tag{184}$$

Although the interaction count grows quadratically with K, parameters are shared across candidate pairs. Each layer has one trainable parameter per relation type, giving the ambiguity module

$$P_{\text{amb}} = LR \tag{185}$$

trainable interaction parameters. Increasing K therefore increases circuit execution cost without directly increasing the number of trainable ambiguity parameters.

For a full candidate pool in the default configuration,

$$K = k_{\max} = 5, \qquad R = 5, \qquad L = 2. \tag{186}$$

There are

$$N_{\text{pair}} = \binom{5}{2} = 10 \tag{187}$$

active candidate pairs and

$$N_{\text{int}} = 2 \times 5 \times 10 = 100 \tag{188}$$

logical disagreement-conditioned interaction factors. These factors are not native physical two-qubit gates. Inputs with $K < 5$ require fewer pairwise interaction factors.

C.6 Circuit Depth and Physical Gate Cost

Let $\mathcal{C}_{\text{pair}}(q_c, w_r)$ denote the compiled physical-gate cost of one interaction $U_{ijr}^{(\ell)}$, where w_r is the Pauli weight of P_r. For an input with K active candidates, the total physical-gate cost of the ambiguity-reasoning interactions is

$$\mathcal{C}_{\text{amb}} = L \sum_{1 \le i < j \le K} \sum_{r=1}^{R} \mathcal{C}_{\text{pair}}(q_c, w_r). \tag{189}$$

Using an average Pauli weight $\bar{w}$ to summarize the per-interaction cost, the scaling is

$$\mathcal{C}_{\text{amb}} = \mathcal{O}\left(LRK^2 \mathcal{C}_{\text{pair}}(q_c, \bar{w})\right). \tag{190}$$

Every P_r in the fixed operator bank of Appendix B.5 has Pauli weight at most two. Under fully serial execution, the interaction depth scales as

$$D_{\text{amb}} = \mathcal{O}\left(LRK^2 D_{\text{pair}}\right), \tag{191}$$

where D_{pair} denotes the compiled depth of a typical pairwise interaction. The compiled depth may be lower when commuting or non-conflicting operations can be scheduled in parallel. In the worst case, $K = k_{\max}$, so the interaction count scales quadratically with the configured candidate capacity. We report logical interaction counts separately from compiled hardware resources because physical requirements also depend on circuit decomposition, hardware connectivity, routing, and native gate support. For physical-device executions, transpiled depth, physical-qubit mapping, and available single- and two-qubit gate counts are recorded separately, as detailed in Appendix D.

C.7 Measurement Complexity

Each shot of the joint circuit produces one candidate-register outcome. Repeating the circuit for M shots yields counts from which the probabilities of all K active candidates are estimated simultaneously:

$$\hat{p}_i = \frac{N_i}{M}, \qquad i \in \{1, \dots, K\}. \tag{192}$$

Finite-shot inference therefore requires

$$\mathcal{C}_{\text{measure}} = \mathcal{O}(M) \tag{193}$$

measurements per example. All K candidate probabilities are estimated from the same joint circuit, and the counts require $\mathcal{O}(M + K)$ classical post-processing.

For a dataset of N_{ex} examples evaluated with M shots each, the total shot count is

$$N_{\text{shots}} = N_{\text{ex}} M. \tag{194}$$

If parameter-shift differentiation were used for finite-shot training, each trainable quantum parameter would require two shifted circuit evaluations. Since the ambiguity module has $P_{\text{amb}} = LR$ trainable parameters, the approximate shot count per training example would be

$$N_{\text{shots}}^{\text{grad}} \approx 2LRM, \tag{195}$$

excluding an optional unshifted forward evaluation. The main experiments instead use exact state-vector probabilities during training to avoid finite-shot gradient-estimation noise. Finite-shot measurement is evaluated separately during inference.

Table 8: Complexity of the proposed joint quantum ambiguity-reasoning framework.

Component	**Complexity / Requirement**
Candidate-register qubits	$\lceil \log_2 k_{\max} \rceil$
Total logical qubits	$\lceil \log_2 k_{\max} \rceil + q_f + q_a$
Graph preprocessing	$\mathcal{O}(K\mathcal{C}_G)$
Pairwise disagreement extraction	$\mathcal{O}(RK^2\mathcal{C}_D)$
State preparation	$\mathcal{O}(\mathcal{C}_{\text{prior}}(K, q_c) + Kq_fB)$
Trainable ambiguity parameters	LR
Quantum interaction factors	$LRK(K-1)/2$
Worst-case interaction scaling	$\mathcal{O}(LRk_{\max}^2)$
Finite-shot inference	$\mathcal{O}(M)$ shots/example
State-vector memory	$\mathcal{O}(2^{q_{\text{total}}})$

Table 9: Logical resource configurations for the candidate-set-size analysis, assuming the full configured candidate pool is retained ($K = k_{\max}$). Logical-qubit counts exclude optional ancillas.

$k_{\max}$	q_c	q_f	L	**Logical Qubits**	**Candidate Pairs**	**Interactions**
3	2	4	2	6	3	30
5	3	4	2	7	10	100
8	3	4	2	7	28	280

C.8 State-Vector Simulation Complexity

Exact state-vector simulation scales exponentially with the total number of simulated qubits. At the configured maximum candidate capacity, the circuit uses

$$q_{\text{total}} = \lceil \log_2 k_{\max} \rceil + q_f + q_a \tag{196}$$

qubits and has a state vector containing

$$2^{q_{\text{total}}} \tag{197}$$

complex amplitudes. Its memory requirement therefore scales as

$$\mathcal{C}_{\text{memory}} = \mathcal{O}\left(2^{q_{\text{total}}}\right). \tag{198}$$

Simulation runtime also depends on the number and decomposition of circuit operations, including the $\mathcal{O}(LRK^2)$ logical interaction factors in the ambiguity-reasoning module. Increasing the active-candidate count K increases the number of pairwise interactions executed, while increasing $k_{\max}$ can increase the state-vector dimension through the candidate-register width. These costs motivate the compact $q_f = 4$ feature register, the default candidate capacity $k_{\max} = 5$, and separate sensitivity analyses of feature-register size, candidate-set size, and ambiguity-reasoning depth.

C.9 Complexity Summary

Table 8 summarizes the computational and quantum-resource requirements of the proposed framework. Here, $K = |\mathcal{S}(x)| \leq k_{\max}$ is the number of active candidates for an input, and $k_{\max}$ is the configured maximum candidate-set size.

The main experiments use $k_{\max} = 5$, $q_f = 4$, $R = 5$, and $L = 2$, as shown in Table 9. When the full candidate pool is retained ($K = k_{\max} = 5$), this configuration uses three candidate qubits and seven logical qubits before optional ancillas, with ten active candidate pairs and 100 logical disagreement-conditioned interaction factors. For inputs with $K < 5$, the candidate-register capacity remains fixed, but fewer pairwise interactions are applied.

The framework has two distinct scaling properties. Parameter sharing keeps the number of trainable ambiguity parameters at LR, regardless of the number of active candidate pairs, while the candidate-register width grows logarithmically with the configured capacity. Dense joint reasoning nevertheless applies interactions across pairs of active candidates, giving $\mathcal{O}(LRK^2)$ interaction

complexity and $\mathcal{O}(LRk_{\max}^2)$ worst-case scaling. This quadratic interaction term is the main scalability constraint of the current formulation and motivates sparse, hierarchical, or selectively active interactions for substantially larger candidate sets.

D Detailed Experimental Setup

This section provides the implementation details for the experiments in Section 5. It describes the datasets, candidate generation, relational graph construction, disagreement extraction, baseline implementations, quantum simulation and hardware settings, noise models, optimization, and evaluation protocol. Unless otherwise specified, every candidate-based method receives the same cached candidate set for each input.

D.1 Datasets and Candidate-Set Ambiguity

We evaluate on **nvBench** Luo et al. (2026) and **Spider** Yu et al. (2018), containing 7,878 natural-language-to-visualization instances and 10,181 text-to-SQL examples, respectively. We preserve the benchmark-defined training and validation partitions. For Spider, reported held-out evaluation is performed on the public development split, for which reference SQL queries and database instances are available. We apply the same candidate-generation, parsing, canonicalization, validity-assignment, and ambiguity-analysis procedures to all compared methods.

For each input x, let $\mathcal{Y}_{\mathrm{ver}}(x)$ denote the set of outputs accepted as valid under the dataset-specific verification procedure described in Appendix F.4, and let $\mathcal{S}(x)$ denote the candidate set retained after parsing, canonicalization, and duplicate removal. The verified valid candidates present in the retained pool are

$$\mathcal{S}_{\mathrm{valid}}(x) = \mathcal{S}(x) \cap \mathcal{Y}_{\mathrm{ver}}(x). \tag{199}$$

We define their number as

$$n_{\mathrm{valid}}(x) = \left|\mathcal{S}_{\mathrm{valid}}(x)\right|. \tag{200}$$

Importantly, $n_{\mathrm{valid}}(x)$ does not estimate the total number of semantically valid interpretations that may exist for an input. It counts only the *verified valid candidates present in the retained candidate pool*. We therefore refer to the resulting partition as *candidate-set ambiguity*, rather than intrinsic or exhaustive semantic ambiguity. For examples with $n_{\mathrm{valid}}(x) \geq 1$, we define

$$\mathrm{Amb}(x) = \begin{cases} \text{Low}, & n_{\mathrm{valid}}(x) = 1, \\ \text{Medium}, & 2 \leq n_{\mathrm{valid}}(x) \leq 3, \\ \text{High}, & n_{\mathrm{valid}}(x) \geq 4. \end{cases} \tag{201}$$

Examples with $n_{\mathrm{valid}}(x) = 0$ contain no verified valid target in the retained candidate pool. They are excluded from the ambiguity strata in equation 201, but are retained in end-to-end evaluation and candidate-generation coverage analysis. Candidate pools, validity assignments, and ambiguity-group assignments are computed once and reused across all compared reasoning models. Consequently, comparisons within each ambiguity group are performed over the same examples and the same retained candidate sets.

D.2 Candidate Generation

We use LLaMA-2-13B Touvron et al. (2023) as the structured-output generator and set the maximum candidate-set size to $k_{\max} = 5$ per input. Candidates are produced using stochastic decoding with temperature $T = 0.7$, top-k sampling with $k_{\mathrm{sample}} = 50$, and a maximum generation length of 256 tokens. The candidate pools are generated once, cached, and shared across all candidate-based models. Generated outputs are parsed and canonicalized before being passed to the reasoning models. Exact duplicates are removed after canonicalization. If fewer than $k_{\max}$ distinct candidates remain, we sample further generations, up to a total of 10 attempts. The retained candidate set is

$$\mathcal{S}(x) = \{s_1, \ldots, s_{K(x)}\}, \qquad K(x) = |\mathcal{S}(x)| \leq k_{\max}. \tag{202}$$

We write K for $K(x)$ when the input is clear from context. If fewer than $k_{\max}$ distinct, successfully parsed outputs are available after the generation budget is exhausted, all available candidates are retained. As described in Appendix B.1, the candidate register still has capacity for $k_{\max}$ candidates,

while basis states without a corresponding retained candidate remain inactive. All candidate-based methods use the same cached candidate pool for each input; the deterministic baseline uses greedy decoding. Sharing the pool ensures that differences among candidate-based methods reflect reasoning and selection rather than candidate generation. For the configured maximum candidate size $k_{\max}$, candidate coverage is measured by

$$\text{Top-}k_{\max} = \mathbb{I}\left[\mathcal{S}(x) \cap \mathcal{Y}_{\text{ver}}(x) \neq \emptyset\right]. \tag{203}$$

This indicator records whether the retained pool contains at least one semantically valid interpretation. It measures candidate-generation coverage independently of the subsequent reasoning model.

D.3 Relational Graph Construction

Each generated candidate s_i is converted into a typed relational graph

$$G_i = (V_i, E_i, \tau_i). \tag{204}$$

Graph construction is task-specific, while the subsequent candidate-reasoning architecture is shared across tasks. For Spider, nodes represent tables, columns, aggregation operators, conditions, values, grouping operations, and joins. Typed edges represent table-column membership, join connectivity, condition attachment, aggregation attachment, grouping relations, and value associations. For nvBench, nodes represent data properties, visualization encoding channels, aggregation operators, grouping choices, filters, and values. Typed edges describe dependencies among these structural components. Graph relations are canonicalized before comparison so that semantically comparable structures receive consistent representations despite differences in serialization order. In particular, differences in the ordering of visualization components or permutations of commutative requirements are not treated as structural differences.

We use a two-layer relation-aware message-passing encoder with hidden dimension 128 and dropout probability 0.1. After graph-level mean pooling, the representation is

$$\mathbf{g}_i = \text{RelEnc}(G_i, x) \in \mathbb{R}^{128}. \tag{205}$$

It is then projected to 16 dimensions:

$$\tilde{\mathbf{g}}_i = W_p \mathbf{g}_i + \mathbf{b}_p \in \mathbb{R}^{16}. \tag{206}$$

As described in Appendix B.3, this 16-dimensional representation is encoded using two feature re-uploading blocks and $q_f = 4$ feature qubits.

D.4 Pairwise Structural Disagreement

We use $R = 5$ task-specific disagreement categories. For Spider, they are

$$\mathcal{R}_{\text{SQL}} = \{\text{Join}, \text{Grouping}, \text{Aggregation}, \text{Condition}, \text{Value}\}. \tag{207}$$

For nvBench, they are

$$\mathcal{R}_{\text{Vis}} = \{\text{Attribute}, \text{Encoding}, \text{Aggregation}, \text{Grouping}, \text{Filtering}\}. \tag{208}$$

For each candidate pair (i, j), the normalized disagreement function in Appendix B.4 produces

$$\mathbf{d}_{ij} = \left[d_{ij}^{(1)}, \ldots, d_{ij}^{(5)}\right]. \tag{209}$$

These vectors are computed once during preprocessing and cached. Both the parameter-matched classical joint-reasoning baseline and the proposed quantum model receive the same disagreement vectors. Thus, differences between the models cannot be attributed to access to different structural disagreement features.

D.5 Baselines

We compare the proposed model with deterministic generation, candidate-based reranking, independent classical graph reasoning, classical joint reasoning, and candidate-wise quantum reasoning. Every candidate-based method receives the same cached candidate set $\mathcal{S}(x) = \{s_1, \ldots, s_K\}$ for each input. Graph-based methods additionally receive the same canonicalized candidate graphs and use the same relational-encoder architecture and projection dimensionality. The encoder and projection parameters are optimized separately within each trainable model, so independently trained methods are not assumed to produce numerically identical latent representations. Unless otherwise stated, trainable candidate-selection models normalize their candidate scores over the same K active candidates.

Deterministic LLM (An et al. (2024)). LLaMA-2-13B uses greedy decoding to produce a single structured output without candidate generation or candidate-level reasoning.

Candidate + Likelihood (Chen et al. (2024)). This baseline selects from the cached candidate pool using the normalized generation likelihood assigned by the candidate generator. No additional candidate-level reasoning model is trained.

Candidate + LLM Rerank (Pourreza et al. (2025)). The generator is used as a reranker over the same cached candidate pool, selecting the candidate judged most compatible with the natural-language input and task context. The reranking procedure is described in Appendix F.2.

Single-Reference Target Selection. For controlled single-reference training, we define one deterministic verified-valid target for every example satisfying $\mathcal{I}_{\text{valid}}(x) \neq \emptyset$. Let s_{ref} denote the designated benchmark reference. If its canonicalized form occurs in the retained candidate pool, the corresponding candidate is used as the single-reference target. Otherwise, we select from the verified-valid retained candidates the candidate with the highest normalized generator likelihood:

$$r(x) = \begin{cases} i_{\text{ref}}, & \exists\, i_{\text{ref}} \text{ such that } \operatorname{Canon}(s_{i_{\text{ref}}}) = \operatorname{Canon}(s_{\text{ref}}), \\ \arg\max\limits_{i \in \mathcal{I}_{\text{valid}}(x)} \ell_{\text{gen}}(s_i \mid x), & \text{otherwise}, \end{cases} \tag{210}$$

where $\ell_{\text{gen}}(s_i \mid x)$ is the normalized generation likelihood stored during candidate generation. Ties are resolved using the fixed cached candidate order. The same target index $r(x)$ is used by all single-reference trainable baselines for that example. Examples with $\mathcal{I}_{\text{valid}}(x) = \emptyset$ contain no verified-valid candidate and are excluded from candidate-selection training for both single-reference and multi-valid objectives. They remain included in end-to-end evaluation and candidate-coverage analysis.

Graph + Single. Each candidate graph is processed independently by the shared relation-aware graph encoder. Let $\tilde{\mathbf{g}}_i \in \mathbb{R}^{d_q}$ denote the projected representation of candidate i. A shared scoring head produces

$$z_i^{\text{G}} = \mathbf{w}_s^\top \tilde{\mathbf{g}}_i + b_s, \tag{211}$$

and candidate probabilities are

$$p_i^{\text{G}} = \frac{\exp(z_i^{\text{G}})}{\sum_{j=1}^{K} \exp(z_j^{\text{G}})}. \tag{212}$$

GRAPH+SINGLE is optimized toward the deterministic single-reference target $r(x)$ defined in Eq. equation 210:

$$\mathcal{L}_{\text{GSingle}} = -\log\left(p_{r(x)}^{\text{G}} + \epsilon\right). \tag{213}$$

Graph + Multi-valid. This baseline uses exactly the same independent graph encoder, projection layer, and scoring head as GRAPH+SINGLE, but changes the supervision objective. Let $\mathcal{I}_{\text{valid}}(x)$ denote the verified valid-candidate indices. We define

$$P_{\text{valid}}^{\text{G}}(x) = \sum_{i \in \mathcal{I}_{\text{valid}}(x)} p_i^{\text{G}}, \tag{214}$$

and optimize

$$\mathcal{L}_{\text{GMulti}} = -\log\left(P_{\text{valid}}^{\text{G}}(x) + \epsilon\right). \tag{215}$$

The comparison between GRAPH+SINGLE and GRAPH+MULTI-VALID therefore isolates the effect of multi-valid supervision without introducing cross-candidate interaction.

Interaction-Parameter-Matched Classical Joint-Set. To compare classical and quantum cross-candidate reasoning, we construct a classical joint model from the same canonicalized candidate graphs, using the same relational-encoder architecture, projection dimensionality, and pairwise disagreement vectors $\mathbf{d}_{ij}$ as the quantum model. The projected representations $\tilde{\mathbf{g}}_i$ are produced by the model's own jointly trained encoder and projection parameters. Candidate states are initialized as

$$\mathbf{h}_i^{(0)} = \tilde{\mathbf{g}}_i. \tag{216}$$

At reasoning layer ℓ, the disagreement-conditioned compatibility between candidates i and j is

$$a_{ij}^{(\ell)} = -\sum_{r=1}^{R} \lambda_r^{(\ell)} d_{ij}^{(r)}, \tag{217}$$

where $\lambda_r^{(\ell)}$ is trainable. For each candidate i, the interaction coefficients over the remaining active candidates are normalized as

$$\alpha_{ij}^{(\ell)} = \frac{\exp\left(a_{ij}^{(\ell)}\right)}{\sum_{\substack{m=1 \\ m\neq i}}^{K} \exp\left(a_{im}^{(\ell)}\right)}, \qquad j \neq i. \tag{218}$$

Candidate states are then updated through residual cross-candidate aggregation:

$$\mathbf{h}_i^{(\ell+1)} = \mathbf{h}_i^{(\ell)} + \sum_{\substack{j=1 \\ j\neq i}}^{K} \alpha_{ij}^{(\ell)} \mathbf{h}_j^{(\ell)}. \tag{219}$$

After L interaction layers, a shared linear scoring head converts the final candidate state into a scalar score:

$$z_i^{\mathrm{CJ}} = \mathbf{w}_s^\top \mathbf{h}_i^{(L)} + b_s. \tag{220}$$

The corresponding candidate distribution is

$$p_i^{\mathrm{CJ}} = \frac{\exp(z_i^{\mathrm{CJ}})}{\sum_{j=1}^{K} \exp(z_j^{\mathrm{CJ}})}. \tag{221}$$

To match the supervision used by JOINTQ-MULTI, the classical joint baseline is trained with the same aggregate valid-candidate objective:

$$P_{\mathrm{valid}}^{\mathrm{CJ}}(x) = \sum_{i \in \mathcal{I}_{\mathrm{valid}}(x)} p_i^{\mathrm{CJ}}, \tag{222}$$

$$\mathcal{L}_{\mathrm{CJ}} = -\log\left(P_{\mathrm{valid}}^{\mathrm{CJ}}(x) + \epsilon\right). \tag{223}$$

The classical interaction mechanism contains one trainable disagreement coefficient for each relation type and reasoning layer:

$$P_{\mathrm{JointSet}}^{\mathrm{int}} = LR. \tag{224}$$

The proposed quantum ambiguity module likewise contains LR trainable interaction coefficients $\{\theta_r^{(\ell)}\}$. Thus, the term *interaction-parameter-matched* refers specifically to the trainable parameters governing cross-candidate interaction; it does not claim equality of the total parameter count of the complete classical and quantum pipelines. The candidate pools, canonicalized graph inputs, encoder architecture, projection dimensionality, and disagreement features are controlled across the comparison, while the encoder and projection parameters are optimized independently within each model. The classical scoring head is shared in form with the independent graph baselines and is not part of the interaction-parameter matching.

Candidate-wise Quantum. To distinguish quantum processing of individual candidates from cross-candidate quantum interaction, this baseline processes every active candidate independently using the same q_f-qubit feature encoding, reasoning depth L, and Pauli operators $\{P_r\}_{r=1}^{R}$ used by the proposed model. It does not use a candidate register, candidate-transition operators, or pairwise disagreement features.

Candidate s_i is encoded as

$$|\phi_i\rangle_f = U_{\mathrm{enc}}(\tilde{\mathbf{g}}_i)|0\rangle_f^{\otimes q_f}. \tag{225}$$

It is then processed independently through L variational layers:

$$|\chi_i\rangle_f = V_{\mathrm{ind}}^{(L)} \cdots V_{\mathrm{ind}}^{(1)} |\phi_i\rangle_f, \tag{226}$$

where

$$V_{\mathrm{ind}}^{(\ell)} = \prod_{r=1}^{R} \exp\left(-i\eta_r^{(\ell)} P_r\right). \tag{227}$$

The parameters $\eta_r^{(\ell)}$ are trainable and shared across all candidates. The candidate-wise quantum reasoning module therefore contains LR trainable quantum parameters, matching the number of trainable interaction coefficients in the proposed joint quantum ambiguity module.

The mean Pauli-Z expectation over the feature register defines a scalar candidate score:

$$z_i^{\mathrm{CQ}} = \frac{1}{q_f} \sum_{m=1}^{q_f} \langle \chi_i | Z_m | \chi_i \rangle. \tag{228}$$

The independently computed scores are normalized over the active candidate set:

$$p_i^{\mathrm{CQ}} = \frac{\exp(z_i^{\mathrm{CQ}})}{\sum_{j=1}^{K} \exp(z_j^{\mathrm{CQ}})}. \tag{229}$$

For a controlled comparison with JOINT QUANTUM+SINGLE, the candidate-wise quantum baseline uses the same deterministic single-reference target $r(x)$ defined in Eq. equation 210:

$$\mathcal{L}_{\mathrm{CQ}} = -\log\left(p_{r(x)}^{\mathrm{CQ}} + \epsilon\right). \tag{230}$$

The principal difference is therefore the reasoning mechanism: CANDIDATE-WISE QUANTUM evaluates each candidate in an independent quantum state, whereas JOINT QUANTUM+SINGLE represents all active candidates jointly and introduces disagreement-conditioned cross-candidate interactions.

Joint Quantum + Single. This variant retains the complete disagreement-conditioned joint quantum architecture but optimizes toward the deterministic single-reference target $r(x)$ defined in Eq. equation 210:

$$\mathcal{L}_{\mathrm{QSingle}} = -\log\left(p_\theta(s_{r(x)} \mid x, \mathcal{G}(x)) + \epsilon\right). \tag{231}$$

The target-selection rule and covered training examples are identical to those used by the other single-reference baselines. This variant therefore preserves joint quantum reasoning while isolating the effect of replacing single-reference supervision with multi-valid supervision.

JointQ-Multi. The complete model jointly represents all K active candidate hypotheses, applies disagreement-conditioned cross-candidate quantum interactions, and optimizes $\mathcal{L}_{\mathrm{QMulti}}$ over the aggregate probability of the verified valid-candidate subspace. Its comparison with JOINT QUANTUM+SINGLE isolates multi-valid supervision, while its comparison with the interaction-parameter-matched CLASSICAL JOINT-SET evaluates the proposed quantum cross-candidate reasoning mechanism under the same candidate information and supervision.

D.6 Default Quantum Configuration

Unless otherwise specified, we use

$$k_{\max} = 5, \qquad q_c = 3, \qquad q_f = 4, \qquad R = 5, \qquad L = 2. \tag{232}$$

The default circuit therefore uses seven logical qubits before optional ancillas:

$$q_{\text{logical}} = q_c + q_f = 7. \tag{233}$$

When all five candidate slots are active, there are ten candidate pairs. With two layers and five relation types, the circuit applies

$$N_{\text{int}} = 2 \times 5 \times 10 = 100 \tag{234}$$

logical disagreement-conditioned interaction factors.

The default finite-shot evaluation uses $M = 1024$ measurements per input. To assess sensitivity to the shot count, we also evaluate

$$M \in \{128, 256, 512, 1024, 2048\}. \tag{235}$$

D.7 Quantum Simulation

We use the Amazon Braket SDK for exact state-vector simulation in the main training and evaluation experiments. State-vector simulation provides exact candidate probabilities, allowing us to compare reasoning architectures without quantum-execution noise or finite-shot uncertainty. We train the full model using exact state-vector probabilities, then evaluate the learned parameters under finite-shot and noisy simulation. This separation keeps stochastic gradient-estimation noise out of the main comparison of reasoning methods. Unless otherwise noted, the primary results use exact state-vector inference; finite-shot and noisy results are reported in the robustness analyses.

D.8 Rigetti Hardware Evaluation

We additionally evaluate the trained ambiguity-reasoning circuit on a superconducting Rigetti quantum processor accessed through Amazon Braket. Model training and parameter optimization are performed using exact state-vector simulation. The resulting model parameters are then held fixed and deployed to the physical QPU without hardware-based fine-tuning or parameter re-optimization.

Because physical-device execution is substantially more costly than simulation, we evaluate a fixed stratified subset of 60 examples from each dataset. For both nvBench and Spider, the subset contains 20 examples from each of the low-, medium-, and high-candidate-set ambiguity groups defined in equation 201. The same 60 examples are used for exact state-vector inference, finite-shot simulation, and physical-device execution, enabling paired comparison across execution conditions.

Hardware execution uses

$$M = 1024 \tag{236}$$

measurement shots per example and the default model configuration

$$k_{\max} = 5, \qquad q_c = 3, \qquad q_f = 4, \qquad R = 5, \qquad L = 2. \tag{237}$$

The logical circuit therefore contains seven register qubits: three candidate-register qubits and four feature-register qubits. Any additional routing operations or auxiliary resources introduced by backend compilation are treated as physical-execution overhead rather than part of the logical model specification.

Exact state-vector inference, finite-shot simulation, and QPU execution use identical candidate pools, graph representations, learned parameters, and circuit-level reasoning settings. The execution backend is therefore the only source of variation among these paired evaluations.

For every submitted hardware circuit, we retain the backend identifier, execution timestamp, logical-to-physical qubit mapping, compiled circuit depth, compiled single- and two-qubit gate counts, measurement shot count, and raw measurement counts. These quantities are stored per circuit rather than summarized as logical interaction counts, because routing, decomposition, and gate counts may vary across compiled instances according to backend connectivity and execution conditions. The corresponding execution-record format is described in Appendix F.8.

Table 10: Configuration used for the physical-device evaluation. The hardware results are obtained on a stratified diagnostic subset rather than the complete benchmark evaluation sets.

Setting	Value
Execution platform	Rigetti QPU via Amazon Braket
Maximum candidate size $k_{\max}$	5
Candidate qubits q_c	3
Feature qubits q_f	4
Logical register qubits	7
Relation types R	5
Reasoning layers L	2
Examples per dataset	60
Low-ambiguity examples per dataset	20
Medium-ambiguity examples per dataset	20
High-ambiguity examples per dataset	20
Shots per example	1024
Total physical-execution shots	122,880

D.9 Noise Simulation

We assess robustness using a controlled depolarizing and readout-noise model. The nominal noise level is

$$\epsilon_q \in \{0, 0.001, 0.005, 0.01, 0.02\}. \tag{238}$$

For each ϵ_q, the one-qubit depolarizing probability is

$$p_{1\mathrm{Q}} = \frac{\epsilon_q}{10}, \tag{239}$$

the two-qubit depolarizing probability is

$$p_{2\mathrm{Q}} = \epsilon_q, \tag{240}$$

and the symmetric readout bit-flip probability is

$$p_{\mathrm{read}} = \epsilon_q. \tag{241}$$

For nonzero ϵ_q, this configuration assigns a higher error probability to two-qubit operations than to single-qubit rotations while maintaining a controlled one-parameter noise sweep. Under noisy execution, leakage into inactive candidate-register states is retained as *invalid probability mass*, without renormalizing the active-state probabilities.

D.10 Training and Hyperparameters

All trainable candidate-selection models are optimized using Adam with an initial learning rate of 1×10^{-3}, weight decay of 1×10^{-5}, and gradient clipping at a maximum norm of 1.0. For graph-based models, the relational encoder and projection layer are trained jointly with the corresponding candidate-reasoning module and output parameters. Thus, projected candidate representations are computed using the current model parameters during training rather than stored as fixed preprocessing outputs.

For the quantum variants, the optimized parameters include the relational encoder, projection layer, and trainable quantum reasoning parameters. For the classical variants, they include the relational encoder, projection layer, corresponding graph- or joint-reasoning parameters, and scoring head. Controlled models use the same encoder architecture, projection dimensionality, input candidate graphs, and training protocol, but their learned parameters are optimized separately for the corresponding objective.

Models are trained for at most 30 epochs with a batch size of 16. Training stops early when the corresponding validation selection performance does not improve for five consecutive epochs. Hyperparameter selection and early stopping use only the validation partition; the held-out evaluation split is not used for model selection. Each trainable configuration is evaluated using three random seeds:

$$\mathcal{Z} = \{13, 42, 2026\}. \tag{242}$$

Table 11: Default training and model hyperparameters.

Hyperparameter	**Value**
Optimizer	Adam
Learning rate	1×10^{-3}
Weight decay	1×10^{-5}
Batch size	16
Maximum epochs	30
Early-stopping patience	5
Gradient clipping	1.0
Random seeds	$\{13, 42, 2026\}$
Graph encoder layers	2
Graph hidden dimension	128
Graph dropout	0.1
Projected dimension d_q	16
Maximum candidate size $k_{\max}$	5
Candidate qubits q_c	3
Feature qubits q_f	4
Relation types R	5
Reasoning layers L	2
Default finite-shot budget M	1024

Using validation performance, we select the learning rate from

$$\eta \in \{5 \times 10^{-4}, 1 \times 10^{-3}, 2 \times 10^{-3}\}, \tag{243}$$

and the ambiguity-reasoning depth from

$$L \in \{1, 2, 3, 4\}. \tag{244}$$

For models without a joint ambiguity-reasoning module, the applicable architectural depth is kept consistent with the corresponding controlled baseline. The selected default configuration uses $L = 2$ and $\eta = 1 \times 10^{-3}$.

Unless otherwise noted, the maximum candidate-set size is $k_{\max} = 5$, requiring $q_c = \lceil \log_2 k_{\max} \rceil = 3$ candidate qubits for the joint quantum model. As described in Appendix D.2, individual inputs may retain fewer active candidates, with $K(x) < k_{\max}$. Only the first $K(x)$ candidate basis states are active for such examples.

All candidate-selection objectives require at least one verified-valid candidate in the retained pool. Examples satisfying $\mathcal{I}_{\text{valid}}(x) = \emptyset$ are therefore excluded from candidate-selection training but remain included in end-to-end evaluation and candidate-coverage analysis. For single-reference variants, the target index $r(x)$ is selected using the deterministic procedure in Eq. equation 210; the same $r(x)$ is used across GRAPH+SINGLE, CANDIDATE-WISE QUANTUM, and JOINT QUANTUM+SINGLE. Multi-valid variants instead optimize the aggregate probability over all indices in $\mathcal{I}_{\text{valid}}(x)$. Table 11 summarizes the default configuration.

D.11 EVALUATION METRICS

We distinguish single-reference exact matching from evaluation against the complete verified reference set available under our protocol. The primary single-reference Exact Match (EM) metric is

$$\text{EM}(x) = \mathbb{I}\left[\text{Canon}(\hat{s}) = \text{Canon}(s_{\text{ref}})\right], \tag{245}$$

where s_{ref} denotes the designated benchmark reference output.

Because an input may have multiple outputs verified as valid under the dataset-specific procedure described in Appendix F.4, we additionally report *Multi-Reference Exact Match*:

$$\text{MultiEM}(x) = \mathbb{I}\left[\text{Canon}(\hat{s}) \in \text{Canon}\left(\mathcal{Y}_{\text{ver}}(x)\right)\right]. \tag{246}$$

Here, $\mathrm{Canon}(\mathcal{Y}_{\mathrm{ver}}(x))$ denotes the set obtained by canonicalizing each member of $\mathcal{Y}_{\mathrm{ver}}(x)$. Multi-EM therefore measures agreement with at least one verified valid output and should not be interpreted as evaluation against an exhaustive set of all semantically equivalent structures.

We separately evaluate whether candidate generation exposes at least one verified valid interpretation. For a retained top-k candidate set $\mathcal{S}_k(x)$, candidate coverage is

$$\mathrm{Coverage@}\, k(x) = \mathbb{I}\left[\mathcal{S}_k(x) \cap \mathcal{Y}_{\mathrm{ver}}(x) \neq \emptyset\right]. \tag{247}$$

Thus, Coverage@$k = 0$ indicates that the retained candidate set contains no output verified as valid under our protocol. In this case, no candidate-selection method operating on the same pool can return a verified valid output. Coverage therefore separates candidate-generation failure from downstream candidate-selection failure.

For finite-shot and noisy quantum execution, we additionally measure *Selection Agreement* relative to exact state-vector inference:

$$\mathrm{Agreement} = \frac{1}{N}\sum_{n=1}^{N} \mathbb{I}\left[\hat{s}_n^{(\mathrm{pert})} = \hat{s}_n^{(\mathrm{exact})}\right], \tag{248}$$

where N denotes the number of evaluated examples for the corresponding dataset and execution setting, $\hat{s}_n^{(\mathrm{exact})}$ is the candidate selected under exact state-vector inference, and $\hat{s}_n^{(\mathrm{pert})}$ is the candidate selected under the finite-shot or noisy setting. Agreement is reported separately for each dataset rather than pooled across nvBench and Spider.

We also report performance separately for the low-, medium-, and high-candidate-set ambiguity groups defined in equation 201.

Task-Level Semantic Evaluation. Exact structural matching remains conservative because distinct representations may preserve the same task-level behavior. We therefore complement EM and Multi-EM with task-specific semantic or functional metrics.

For nvBench, we define a visualization semantic signature

$$\Phi_{\mathrm{viz}}(s) = \{t, A, E, G, H, F\}, \tag{249}$$

where t denotes visualization type, A selected attributes, E attribute-to-encoding mappings, G aggregation operations, H grouping decisions, and F filtering criteria. Representation-level ordering and serialization differences that do not alter the represented visualization are ignored. *Visualization Semantic Match* (VSM) is defined as

$$\mathrm{VSM}(x) = \mathbb{I}\left[\exists\, y \in \mathcal{Y}_{\mathrm{ver}}(x) : \Phi_{\mathrm{viz}}(\hat{s}) = \Phi_{\mathrm{viz}}(y)\right]. \tag{250}$$

VSM therefore evaluates whether the predicted visualization preserves the semantic specification of at least one verified output without requiring exact canonicalized structural identity.

For Spider, we additionally report execution accuracy. Let $\mathrm{Exec}(s, D_x)$ denote the result of executing SQL structure s on the database D_x associated with input x. We define

$$\mathrm{EX}(x) = \mathbb{I}\left[\exists\, y \in \mathcal{Y}_{\mathrm{ver}}(x) : \mathrm{Exec}(\hat{s}, D_x) = \mathrm{Exec}(y, D_x)\right]. \tag{251}$$

Execution equivalence is intentionally broader than Multi-EM: a query may receive credit under EX even when it does not satisfy the stricter canonicalized structural criterion used for membership in $\mathcal{Y}_{\mathrm{ver}}(x)$. Because distinct SQL queries can produce the same result on a particular database instance, EX is used as a complementary functional metric rather than as a substitute for structural evaluation.

D.12 Controlled Comparison Protocol

We design the baseline comparisons to isolate the effects of cross-candidate interaction, disagreement conditioning, quantum processing, and multi-valid supervision while holding the available candidate information and model capacity as comparable as possible.

All candidate-based methods operate on the same cached candidate pools $\mathcal{S}(x)$. All graph-based methods receive the same canonicalized candidate graphs $\mathcal{G}(x)$ and use the same relation-aware graph-encoder architecture and projection dimensionality. The encoder and projection parameters

are trained separately within each model; consequently, the resulting learned candidate representations are not assumed to be numerically identical across independently trained methods.

The classical and quantum joint-reasoning models receive the same precomputed pairwise disagreement vectors $\mathbf{d}_{ij}$ derived from the fixed canonicalized candidate graphs. Candidate-wise and joint quantum variants use the same feature-register size q_f, feature-encoding procedure, Pauli-operator family, and reasoning depth L.

The principal controlled comparisons are

$$\begin{aligned} \textsc{Graph+Single} &\longleftrightarrow \textsc{Graph+Multi-valid}, \\ \textsc{Graph+Multi-valid} &\longleftrightarrow \textsc{Classical Joint-Set}, \\ \textsc{Candidate-wise Quantum} &\longleftrightarrow \textsc{Joint Quantum+Single}, \\ \textsc{Classical Joint-Set} &\longleftrightarrow \textsc{JointQ-Multi}, \\ \textsc{Joint Quantum+Single} &\longleftrightarrow \textsc{JointQ-Multi}. \end{aligned} \tag{252}$$

The comparison between GRAPH+SINGLE and GRAPH+MULTI-VALID preserves the model architecture and changes the supervision objective, thereby measuring the contribution of multi-valid learning without introducing cross-candidate reasoning.

The comparison between GRAPH+MULTI-VALID and CLASSICAL JOINT-SET uses the same candidate pools, graph-input representation, graph-encoder architecture, projection dimensionality, and multi-valid supervision while adding classical cross-candidate interaction. It therefore evaluates the contribution of set-level candidate reasoning relative to independent graph processing.

The comparison between CANDIDATE-WISE QUANTUM and JOINT QUANTUM+SINGLE uses the same single-reference supervision, feature-register size, reasoning depth, and LR trainable quantum reasoning parameters. The principal distinction is whether candidates are processed independently or represented jointly with disagreement-conditioned cross-candidate interactions.

The comparison between CLASSICAL JOINT-SET and JOINTQ-MULTI uses the same retained candidate pools, canonicalized candidate graphs, disagreement features, projection dimensionality, and multi-valid supervision. Their respective cross-candidate interaction modules each contain LR trainable disagreement-dependent coefficients. Thus, this comparison is *interaction-parameter-matched*: it controls the number of trainable parameters directly governing cross-candidate interaction rather than asserting equality of the total end-to-end parameter counts or equality of independently learned latent representations.

Finally, JOINT QUANTUM+SINGLE and JOINTQ-MULTI use the same joint quantum architecture, candidate encoding, disagreement-conditioned interactions, and inference procedure. They differ in the supervision objective, providing the most direct controlled comparison for assessing the contribution of multi-valid learning.

D.13 REPRODUCIBILITY

For each experiment, we retain the random seed, generated candidate set, canonicalized candidate structures, relational graph representations, structural disagreement vectors, trained model parameters, simulator configuration, measurement-shot budget, and noise configuration. Candidate pools are cached so that all controlled baseline comparisons operate on identical generated candidates. For physical-device experiments, we additionally retain the Amazon Braket execution identifier, backend identifier, execution timestamp, logical-to-physical qubit mapping, compiled circuit depth, compiled single- and two-qubit gate counts, measurement-shot count, and raw measurement outcomes for each submitted circuit. The exact state-vector and finite-shot simulator predictions for the same hardware-evaluated examples are stored alongside the corresponding QPU predictions.

Together, these records permit paired reconstruction of the state-vector, finite-shot, noisy-simulation, and physical-device evaluations and allow individual hardware disagreements to be traced back to their candidate distributions and compiled circuit executions.

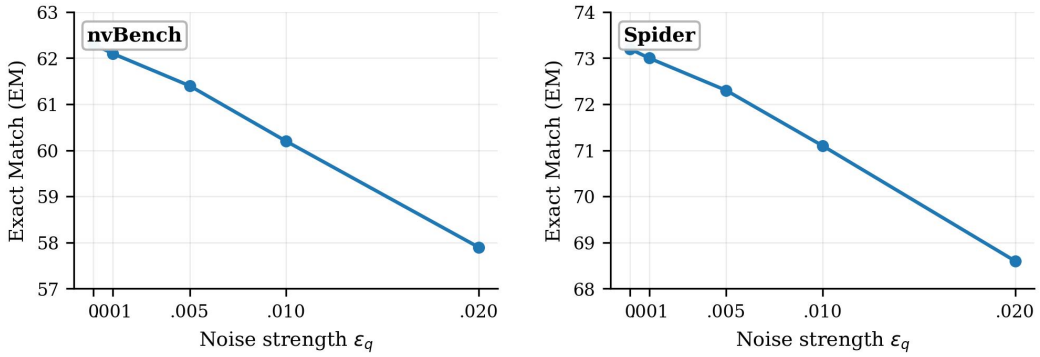


Figure 3: Robustness of JOINTQ-MULTI under increasing simulated quantum noise using 1024 shots.

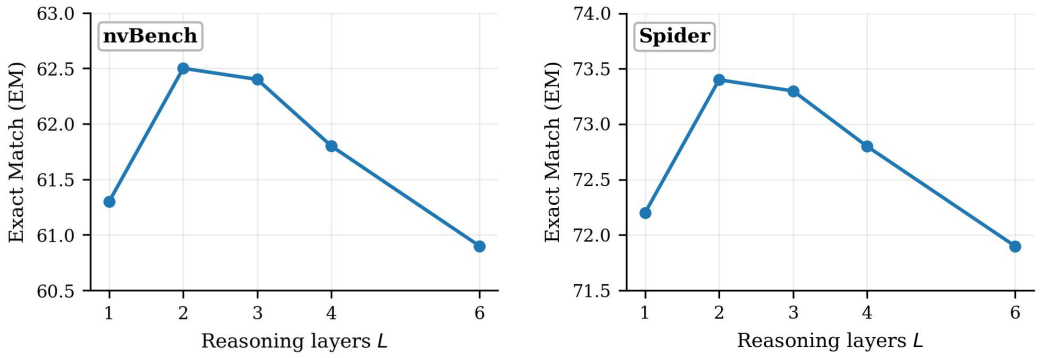


Figure 4: Effect of the number of ambiguity-reasoning layers L on nvBench and Spider.

E ADDITIONAL EXPERIMENTAL ANALYSES

This section presents additional sensitivity and robustness analyses beyond the main results in Section 5. We focus on feature-register size, ambiguity-reasoning depth, finite-shot measurement, quantum noise, candidate ordering, candidate-prior initialization, and physical-device execution. Ambiguity-level performance, candidate-set size, and cross-candidate ablations are reported in the main paper. Unless otherwise noted, experiments use $k_{\max} = 5$, with $K(x) \leq k_{\max}$ active candidates per input, along with $q_c = 3$, $q_f = 4$, $R = 5$, $L = 2$, and $M = 1024$ shots.

E.1 ROBUSTNESS TO QUANTUM NOISE (RQ4).

Figure 3 shows gradual degradation as simulated gate and measurement noise increase. At $\epsilon_q = 0.001$, EM decreases by only 0.2 points on both datasets relative to noiseless 1024-shot inference. Stronger noise produces increasingly larger drops, reaching 57.9 EM on nvBench and 68.6 on Spider at $\epsilon_q = 0.02$. These results suggest that the shallow ambiguity-reasoning circuit remains relatively stable under mild noise but becomes increasingly affected as execution errors accumulate.

E.2 EFFECT OF CIRCUIT DEPTH (RQ4).

Figure 4 shows that increasing the number of ambiguity-reasoning layers from $L = 1$ to $L = 2$ improves EM by 1.2 points on both datasets. Performance remains nearly unchanged at $L = 3$ and decreases for deeper circuits, suggesting that additional interaction layers provide little benefit while increasing circuit complexity. We therefore use $L = 2$ as the default configuration, providing sufficient capacity for cross-candidate reasoning while keeping the circuit shallow.

Table 12: Sensitivity to feature-register size with $k_{\max} = 5$ and $L = 2$. Logical-qubit counts exclude optional ancillas. Results are reported as mean $\pm$ standard deviation over three random seeds.

q_f	Logical Qubits	Re-upload Blocks	nvBench	Spider
2	5	4	60.9 ± 0.4	71.8 ± 0.3
4	7	2	$\mathbf{62.5 \pm 0.2}$	$\mathbf{73.4 \pm 0.2}$
6	9	2	62.6 ± 0.3	73.5 ± 0.3

Table 13: Effect of ambiguity-reasoning depth for the default full candidate pool. Results are reported as mean $\pm$ standard deviation over three random seeds.

L	Interactions	Parameters	nvBench	Spider
1	50	5	61.3 ± 0.3	72.2 ± 0.3
2	100	10	$\mathbf{62.5 \pm 0.2}$	$\mathbf{73.4 \pm 0.2}$
3	150	15	62.4 ± 0.3	73.3 ± 0.3
4	200	20	61.8 ± 0.4	72.8 ± 0.4

E.3 Feature-Register Sensitivity

Table 12 evaluates feature-register sizes $q_f \in \{2, 4, 6\}$. Increasing the register from two to four qubits improves performance and reduces feature re-uploading. Expanding it to six qubits increases the simulated state dimension but provides little additional benefit. We therefore use $q_f = 4$ by default.

E.4 Effect of Ambiguity-Reasoning Depth

Table 13 evaluates reasoning depths $L \in \{1, 2, 3, 4\}$. Performance improves from one to two layers, changes little at $L = 3$, and shows no further improvement at $L = 4$. We therefore use $L = 2$ by default to balance interaction capacity and circuit complexity.

E.5 Finite-Shot Measurement Analysis

Table 14 evaluates finite-shot inference with

$$M \in \{128, 256, 512, 1024, 2048\}. \tag{253}$$

For each measurement budget, execution is repeated using ten independent sampling seeds while keeping the trained model parameters, candidate pools, and exact state-vector distributions fixed.

For dataset d, let N_d denote the number of evaluated examples, $\hat{s}_{dn}^{(M,r)}$ the candidate selected using M shots under sampling repeat r, and $\hat{s}_{dn}^{(\text{exact})}$ the candidate selected under exact state-vector inference. We compute selection agreement for repeat r by pooling the examples from nvBench and Spider:

$$\text{Agreement}^{(M,r)} = \frac{\sum_{d} \sum_{n=1}^{N_d} \mathbb{I}\left[\hat{s}_{dn}^{(M,r)} = \hat{s}_{dn}^{(\text{exact})}\right]}{\sum_{d} N_d}. \tag{254}$$

The Agreement values in Table 14 are the mean and standard deviation of this pooled quantity across the ten sampling repeats. The nvBench and Spider EM columns are computed separately for the two datasets.

As the shot budget increases, finite-shot candidate selections progressively approach those obtained under exact state-vector inference. Pooled Selection Agreement increases from 91.6% at 128 shots to 98.0% at 1024 shots and 98.8% at 2048 shots. At 1024 shots, finite-shot EM is within 0.2 points of the corresponding exact-inference result on each dataset. We therefore use $M = 1024$ as the default finite-shot measurement budget in experiments requiring sampled quantum execution.

Table 14: Finite-shot sensitivity of JOINTQ-MULTI. nvBench and Spider EM are reported separately, while Pooled Agreement measures candidate-selection agreement with exact state-vector inference across the combined evaluation examples from both datasets. Finite-shot values are mean $\pm$ standard deviation over ten independent sampling executions.

Measurement	Repeats	nvBench EM	Spider EM	Pooled Agreement
128 shots	10	60.8 ± 0.6	71.8 ± 0.5	91.6 ± 1.1
256 shots	10	61.5 ± 0.5	72.5 ± 0.4	94.2 ± 0.9
512 shots	10	62.0 ± 0.4	73.0 ± 0.3	96.5 ± 0.7
1024 shots	10	62.3 ± 0.3	73.2 ± 0.3	98.0 ± 0.4
2048 shots	10	62.4 ± 0.2	73.3 ± 0.2	98.8 ± 0.3
Exact state-vector	–	$\mathbf{62.5 \pm 0.2}$	$\mathbf{73.4 \pm 0.2}$	100.0

Table 15: Performance under simulated quantum noise using 1024 measurement shots. nvBench and Spider EM are reported separately. Pooled Agreement measures candidate-selection agreement with the corresponding noiseless 1024-shot execution across examples from both datasets. Results are mean $\pm$ standard deviation over ten independent executions.

ϵ_q	Repeats	nvBench EM	Spider EM	Pooled Agreement
0	10	62.3 ± 0.3	73.2 ± 0.3	100.0
0.001	10	62.1 ± 0.3	73.0 ± 0.3	98.4 ± 0.4
0.005	10	61.4 ± 0.4	72.3 ± 0.4	95.1 ± 0.6
0.010	10	60.2 ± 0.5	71.1 ± 0.5	91.3 ± 0.9
0.020	10	57.9 ± 0.7	68.6 ± 0.6	84.7 ± 1.3

E.6 ROBUSTNESS TO QUANTUM NOISE

We evaluate robustness to controlled quantum noise using the 1024-shot configuration. For noise level ϵ_q, the simulated single-qubit gate, two-qubit gate, and readout error probabilities are parameterized as

$$p_{1\mathrm{Q}} = \frac{\epsilon_q}{10}, \qquad p_{2\mathrm{Q}} = \epsilon_q, \qquad p_{\mathrm{read}} = \epsilon_q. \tag{255}$$

Each noise configuration is evaluated over ten independent execution seeds. To isolate changes caused by the injected noise, Selection Agreement is measured relative to the corresponding noiseless 1024-shot execution rather than relative to exact state-vector inference. For repeat r, let $\hat{s}_{dn}^{(\epsilon_q,r)}$ denote the prediction under noise level ϵ_q and let $\hat{s}_{dn}^{(0,r)}$ denote the prediction obtained from the corresponding noiseless finite-shot execution. We define pooled noisy agreement as

$$\mathrm{Agreement}^{(\epsilon_q,r)} = \frac{\sum_{d} \sum_{n=1}^{N_d} \mathbb{I}\left[\hat{s}_{dn}^{(\epsilon_q,r)} = \hat{s}_{dn}^{(0,r)}\right]}{\sum_{d} N_d}. \tag{256}$$

The reported Pooled Agreement is the mean and standard deviation of this quantity across the repeated executions.

Performance remains relatively stable under mild simulated noise. At $\epsilon_q = 0.001$, EM decreases by only 0.2 points on both datasets relative to noiseless 1024-shot inference, while pooled candidate-selection agreement remains 98.4%. As the noise level increases, disagreement with the noiseless execution becomes more frequent. At $\epsilon_q = 0.02$, Pooled Agreement decreases to 84.7%, with EM reaching 57.9 on nvBench and 68.6 on Spider.

These results indicate that the shallow ambiguity-reasoning circuit retains most candidate decisions under low simulated noise, whereas accumulated gate and readout errors increasingly perturb the candidate distribution at stronger noise levels. The synthetic-noise analysis is intended to characterize sensitivity to controlled execution errors; physical-device behavior is evaluated separately in Appendix E.9.

Table 16: Candidate-ordering robustness using 20 random permutations per example. Permutation Agreement is reported as mean ± standard deviation.

Method	Permutations	Permutation Agreement
Classical Joint-Set	20	99.8 ± 0.1
Candidate-wise Quantum	20	100.0 ± 0.0
JointQ-Multi	20	99.4 ± 0.2

Table 17: Effect of candidate-prior initialization. Results are reported as mean ± standard deviation over three random seeds.

Prior	τ	nvBench	Spider
Uniform	–	$\mathbf{62.5 \pm 0.2}$	$\mathbf{73.4 \pm 0.2}$
Generator likelihood	1.0	61.8 ± 0.3	72.9 ± 0.3

E.7 Candidate-Ordering Robustness

Exact candidate-permutation invariance is not guaranteed because the first-order product-form implementation applies pairwise interactions in a fixed lexicographic order. We therefore evaluate sensitivity to candidate ordering using $N_{\text{perm}} = 20$ random permutations per example in Table 16. The high agreement suggests that predictions are largely unaffected by arbitrary candidate ordering under the evaluated conditions. The small variation observed for JointQ-Multi is consistent with the fixed ordering of generally noncommuting pairwise interactions. We therefore regard *candidate-order robustness* as an empirical property rather than claim exact permutation invariance.

E.8 Effect of Candidate Prior

The default joint state uses a uniform prior over the K active candidates:

$$\pi_i = \frac{1}{K}, \qquad i \in \{1, \dots, K\}. \tag{257}$$

For a full candidate pool with $K = 5$, this assigns $\pi_i = 0.2$ to each candidate, as shown in Table 17. We compare this initialization with a prior derived from the normalized generator log-likelihood ℓ_i:

$$\pi_i^{\text{gen}} = \frac{\exp(\ell_i/\tau)}{\sum_{j=1}^{K} \exp(\ell_j/\tau)}, \tag{258}$$

with temperature $\tau = 1.0$. The uniform prior performs marginally better, suggesting that incorporating generator confidence into the initial quantum state may reinforce incorrect candidate preferences. We therefore use uniform initialization in the main experiments.

E.9 Physical-Device Results

Table 18 compares exact state-vector inference, 1024-shot simulation, and physical-device execution on the same stratified 60-example subset from each dataset. These experiments are intended as a diagnostic study of execution robustness and are separate from the full-dataset results reported in the main experiments. For nvBench, exact state-vector inference obtains 63.3 EM on the hardware-evaluation subset, while 1024-shot simulation obtains 61.7 EM. The finite-shot simulator reproduces the exact candidate selection on 96.7% of the 60 examples. Physical-device execution obtains 58.3 EM and preserves the exact state-vector selection on 88.3% of the examples. For Spider, exact state-vector inference obtains 73.3 EM on the same type of stratified subset, compared with 71.7 EM under 1024-shot simulation. Finite-shot Selection Agreement is 95.0%. Physical-device execution reaches 68.3 EM and agrees with the exact state-vector selection on 86.7% of the examples.

The larger degradation under physical-device execution is consistent with error sources absent from ideal state-vector inference and only partially represented by finite-shot simulation, including gate

Table 18: Physical-device evaluation of JOINTQ-MULTI on the stratified 60-example subset from each dataset. Selection Agreement is computed separately for each dataset relative to exact state-vector inference on the same examples.

Execution	nvBench EM	Agreement	Spider EM	Agreement
Exact state-vector	**63.3**	100.0	**73.3**	100.0
1024-shot simulator	61.7	96.7	71.7	95.0
Rigetti QPU	58.3	88.3	68.3	86.7

and readout errors, device-connectivity constraints, routing operations, and compiled-circuit overhead. Nevertheless, the majority of candidate selections remain unchanged relative to exact inference in both datasets. Importantly, these results evaluate inference using an already trained model; no parameters are optimized using QPU measurements. The physical-device experiment therefore characterizes the execution robustness of the learned ambiguity-reasoning circuit rather than hardware-aware training. Because the experiment uses a small stratified subset, it should be interpreted as evidence of practical executability and candidate-selection robustness rather than as a full-scale hardware benchmark. We make no claim of computational speedup or quantum advantage from these results.

E.10 SUMMARY OF ADDITIONAL ANALYSES

The additional sensitivity and robustness analyses provide a more complete characterization of the proposed framework. Two ambiguity-reasoning layers and a four-qubit feature register offer a favorable trade-off between circuit complexity and predictive performance. Stronger simulated noise reduces candidate-selection stability, while finite-shot estimates approach exact state-vector predictions as the measurement budget increases. Candidate-ordering tests show strong empirical robustness despite the fixed ordering of the product-form interaction circuit, and the candidate-prior comparison supports uniform initialization. The physical-device protocol further evaluates transfer of the trained reasoning circuit to existing superconducting quantum hardware.

F PROMPTS, CANONICALIZATION, AND REPRODUCIBILITY DETAILS

This section describes the preprocessing procedures and prompt templates used to construct candidate sets and evaluate structured predictions. Candidate generation is independent of the proposed reasoning model, and the generated pools are cached and shared across all candidate-based methods. This prevents differences in LLM sampling from confounding comparisons of reasoning methods.

F.1 CANDIDATE-GENERATION PROMPT

For each input, the generator is instructed to produce one syntactically valid structured output at a time. Stochastic decoding is repeated to obtain up to $k_{\max} = 5$ distinct candidates. The prompt includes the task schema and requests only the structured output, without additional natural-language explanation.

For Spider, we use the following template:

```
You are given a natural-language question and a database schema.

Generate one valid SQL query that answers the question using only
the provided schema.

The question may be ambiguous. Produce one plausible interpretation
of the question. Do not enumerate multiple alternatives in a single
response.

Return only the SQL query without explanation.
```

```
Database schema:
{SCHEMA}

Question:
{QUESTION}

SQL:
```

Repeated stochastic generations from the same prompt provide alternative candidate interpretations. We generate candidates in separate model calls rather than asking the generator to enumerate multiple alternatives within one completion. This avoids introducing explicit within-response dependencies among candidates and allows each candidate to be sampled independently from the same prompt context.

For nvBench, we use the corresponding visualization-generation template:

```
You are given a natural-language visualization request and the
corresponding database/table schema.

Generate one valid visualization specification that satisfies one
plausible interpretation of the request.

The request may be ambiguous and may admit multiple reasonable
interpretations. Use only attributes available in the provided schema.
Do not enumerate multiple alternatives in a single response.

Return only the structured visualization specification without
explanation.

Schema:
{SCHEMA}

Request:
{QUESTION}

Visualization:
```

Candidate generation uses temperature $T = 0.7$, top-k sampling with $k_{\text{sample}} = 50$, and a maximum generation length of 256 tokens. Sampling continues until either five distinct canonicalized candidates are obtained or ten generation attempts have been made. Candidates that fail parsing are discarded, and canonicalized duplicates are removed. If fewer than five distinct successfully parsed candidates remain, all available candidates are retained, as described in Appendix D.2.

F.2 Candidate Reranking Prompt

The Candidate+LLM Rerank baseline operates on exactly the same cached candidate pool as the other candidate-based methods. Unlike the candidate generator, the reranker is not allowed to modify an existing candidate or produce a new output. It receives the natural-language input, the relevant schema, and all retained candidates, and is asked to select the candidate that best satisfies the request.

Providing the schema to the reranker ensures that candidate selection can account for schema compatibility, including table, attribute, join, grouping, aggregation, filtering, and value choices, rather than judging candidates from the natural-language request alone.

For Spider, we use the following template:

```
You are given a natural-language question, a database schema,
and several candidate SQL queries.
```

```
Select the candidate that best answers the question using the
provided schema.

The candidates may differ in important structural details such as
tables, joins, aggregation, grouping, conditions, or values.

Do not modify any candidate and do not generate a new SQL query.
Return only the number of the best candidate.

Database schema:
{SCHEMA}

Question:
{QUESTION}

Candidate 1:
{CANDIDATE_1}

Candidate 2:
{CANDIDATE_2}

...

Candidate K:
{CANDIDATE_K}

Best candidate number:
```

For nvBench, we use the corresponding visualization-reranking template:

```
You are given a natural-language visualization request, the
corresponding schema, and several candidate visualization
specifications.

Select the candidate that best satisfies the request using only
the provided schema.

The candidates may differ in important structural details such as
attribute selection, encoding assignment, aggregation, grouping,
filtering, or visualization type.

Do not modify any candidate and do not generate a new visualization
specification. Return only the number of the best candidate.

Schema:
{SCHEMA}

Request:
{QUESTION}

Candidate 1:
{CANDIDATE_1}

Candidate 2:
{CANDIDATE_2}

...

Candidate K:
```

```
{CANDIDATE_K}

Best candidate number:
```

Here, K denotes the number of distinct, successfully parsed candidates retained for the input. Candidate ordering is fixed for the primary reranking experiment. Sensitivity to candidate permutation is evaluated separately in Appendix E.7.

F.3 Candidate Parsing and Canonicalization

Generated candidates are parsed before being passed to the classical or quantum reasoning models. Candidates that cannot be converted into the required structured representation are discarded. For Spider, SQL candidates are canonicalized by normalizing keyword casing, redundant whitespace, and aliases. Serialization order is normalized only where the corresponding operations are semantically order-invariant. Structural elements, including tables, columns, joins, aggregation operations, grouping variables, conditions, and values, are preserved and extracted for candidate-graph construction. For nvBench, visualization candidates are canonicalized by normalizing representation-level differences in attribute names, encoding assignments, aggregation operators, grouping choices, filters, and other visualization-specific structural fields. Ordering is normalized only for components whose order does not affect the rendered visualization.

Canonicalization precedes duplicate removal. Two candidates s_i and s_j are considered duplicates when

$$\mathrm{Canon}(s_i) = \mathrm{Canon}(s_j). \tag{259}$$

Only one instance is retained in the candidate pool. Canonicalization removes superficial formatting and serialization differences without inferring previously unannotated semantic equivalence. Candidates remain distinct when they differ in substantive structural choices, including selected relations, joins, aggregation, grouping, filters, values, or visualization encodings.

F.4 Validity Assignment

Training with the multi-valid objective requires identifying which retained candidates are supported as valid under the evaluation protocol. Let $\mathcal{Y}_{\mathrm{ver}}(x)$ denote the verified valid-output set associated with input x. Because the form of available supervision differs across benchmarks, the construction of $\mathcal{Y}_{\mathrm{ver}}(x)$ is dataset-specific.

nvBench. For nvBench, we construct $\mathcal{Y}_{\mathrm{ver}}(x)$ from the benchmark-supported visualization specifications associated with the input. All available reference structures are parsed and canonicalized using the same procedure applied to generated candidates. Distinct canonicalized references are retained, allowing multiple benchmark-supported visualization structures to remain valid for the same natural-language request.

Spider. Spider provides a single benchmark reference SQL query for each natural-language input in the evaluated split. We first parse and canonicalize the reference query using the same procedure applied to generated candidates. A generated SQL candidate that differs from the reference surface form is treated as an additional verified alternative only when it remains equivalent under the canonicalized structural comparison and produces the same execution result as the benchmark reference on the corresponding database. Execution agreement alone is not sufficient for assignment to $\mathcal{Y}_{\mathrm{ver}}(x)$, because structurally different SQL queries may produce identical results on a particular database instance without expressing the same query semantics. This conservative combination of canonicalized structural compatibility and execution agreement reduces the risk of introducing accidental execution matches as positive multi-valid targets.

Canonicalized Validity Matching. After constructing $\mathcal{Y}_{\mathrm{ver}}(x)$, each generated candidate $s_i \in \mathcal{S}(x)$ is assigned the validity indicator

$$y_i(x) = \mathbb{I}\left[s_i \in \mathcal{Y}_{\mathrm{ver}}(x)\right], \tag{260}$$

where membership is evaluated after the parsing and canonicalization procedure described in Appendix F.3. The corresponding valid-candidate index set is

$$\mathcal{I}_{\text{valid}}(x) = \{i : y_i(x) = 1\}. \tag{261}$$

We do not assign positive labels solely on the basis of generator likelihood, structural similarity, or unconstrained model judgment. A candidate is treated as valid only when it satisfies the dataset-specific verification criterion described above. This conservative protocol is intended to avoid introducing unsupported alternatives into multi-valid supervision.

The verified set $\mathcal{Y}_{\text{ver}}(x)$ should not be interpreted as an exhaustive enumeration of every semantically valid structure that could express the intended meaning. Accordingly, throughout the paper, the terms *valid candidate*, *Multi-EM*, and *candidate-set ambiguity* refer specifically to validity established by this verification protocol.

Validity information is used only for training supervision and evaluation. At inference time, the reasoning model receives the natural-language input, retained candidate structures, relational graph representations, and pairwise disagreement features, but neither $y_i(x)$ nor $\mathcal{I}_{\text{valid}}(x)$.

If

$$\mathcal{I}_{\text{valid}}(x) = \emptyset, \tag{262}$$

the retained candidate set contains no verified valid target. Such examples remain part of end-to-end evaluation and candidate-generation coverage analysis, but they provide no positive target for the multi-valid candidate-selection objective. We therefore exclude them from the multi-valid selection loss rather than assigning an artificial negative or pseudo-positive target.

This treatment separates candidate-generation failure from candidate-selection failure: the reasoning module can select a verified valid interpretation only when at least one such candidate is present in the retained pool.

F.5 Construction of Pairwise Disagreement Matrices

For each input, the candidate set yields one disagreement matrix per relation type. For relation r, we define

$$D^{(r)} \in \mathbb{R}^{K \times K}, \tag{263}$$

where $K = |\mathcal{S}(x)|$ is the number of active candidates, with entries

$$D_{ij}^{(r)} = d_{ij}^{(r)}. \tag{264}$$

The disagreement function is symmetric, and each candidate has zero disagreement with itself:

$$D_{ij}^{(r)} = D_{ji}^{(r)}, \qquad D_{ii}^{(r)} = 0. \tag{265}$$

For the full default candidate pool, $K = 5$ and $R = 5$, this produces five symmetric 5×5 matrices, each containing ten distinct candidate-pair comparisons. When fewer than five candidates are active, disagreements are computed only among the retained candidates. Unused candidate-register states remain inactive and do not participate in cross-candidate interactions. The disagreement matrices are computed once during preprocessing and stored with the candidate graphs. CLASSICAL JOINT-SET and JOINTQ-MULTI receive identical disagreement information.

F.6 Data Stored for Each Example

After preprocessing, each candidate-based example is stored as

$$\mathcal{E}(x) = \{\mathcal{S}(x), \mathcal{G}(x), \mathcal{D}(x), \mathcal{I}_{\text{valid}}(x)\}, \tag{266}$$

where $\mathcal{S}(x)$ is the retained candidate set, $\mathcal{G}(x) = \{G_1, \ldots, G_K\}$ contains the corresponding canonicalized relational graphs, $\mathcal{D}(x)$ contains the pairwise structural disagreement vectors, and $\mathcal{I}_{\text{valid}}(x)$ contains the verified valid-candidate indices used during training and evaluation.

Candidate-generation outputs, canonicalized graphs, disagreement features, and validity assignments are cached before model training. Consequently, repeated training across random seeds and

controlled baseline comparisons use identical candidate pools and fixed preprocessed structural inputs without regenerating or re-labeling candidates.

Graph embeddings and projected candidate representations are not stored as fixed preprocessing outputs. Instead, during training and inference, the current relational encoder computes

$$\mathbf{g}_i = \mathrm{RelEnc}(G_i, x), \tag{267}$$

followed by the trainable projection

$$\tilde{\mathbf{g}}_i = W_p \mathbf{g}_i + b_p. \tag{268}$$

Thus, $\mathbf{g}_i$ and $\tilde{\mathbf{g}}_i$ may evolve during optimization and may differ across independently trained models, while the underlying candidate structures, graph inputs, disagreement features, and validity assignments remain fixed.

F.7 Randomness and Repeated Evaluation

Trainable experiments use three model-training seeds:

$$\mathcal{Z}_{\mathrm{train}} = \{13, 42, 2026\}. \tag{269}$$

Finite-shot and synthetic-noise evaluations use ten independent measurement seeds per configuration. Candidate-order robustness experiments use twenty random permutations per example. We treat these sources of randomness separately. The generated candidate pool remains fixed across all training and measurement seeds, allowing variation from optimization and quantum measurement to be examined independently.

F.8 Quantum Execution Records

To support reproducibility and paired analysis of the physical-device experiments, we retain execution metadata separately for every submitted quantum circuit. For circuit C, we define the associated metadata record as

$$\begin{aligned}\mathcal{M}(C) = \{&\text{backend}, \text{task-id}, \text{timestamp}, M, q_{\mathrm{logical}}, q_{\mathrm{physical}},\\ &D_{\mathrm{compiled}}, N_{1\mathrm{Q}}, N_{2\mathrm{Q}}\},\end{aligned} \tag{270}$$

where M is the number of measurement shots, q_{logical} is the number of logical circuit qubits, q_{physical} records the logical-to-physical qubit mapping, D_{compiled} denotes compiled circuit depth, and $N_{1\mathrm{Q}}$ and $N_{2\mathrm{Q}}$ denote the compiled single- and two-qubit gate counts, respectively. The raw candidate-register measurement counts are retained separately for every structured-generation example. If $N_i(C)$ denotes the number of shots in which the candidate register is measured in the basis state associated with candidate s_i, the physical-device candidate probability estimate is

$$\hat{p}_i^{\mathrm{QPU}} = \frac{N_i(C)}{M}. \tag{271}$$

The selected hardware candidate is therefore

$$\hat{s}_{\mathrm{QPU}} = \arg\max_{s_i \in \mathcal{S}(x)} \hat{p}_i^{\mathrm{QPU}}. \tag{272}$$

For every hardware-evaluated example, we additionally retain the candidate distribution and prediction obtained from exact state-vector inference and the corresponding 1024-shot simulator execution. This creates a one-to-one record linking each physical-device execution to its ideal and finite-shot counterparts. These records enable hardware disagreements to be examined at the candidate level and distinguish changes caused by finite sampling from those introduced by physical execution and compilation. Because compiled depth, gate counts, and physical-qubit mappings may vary across submitted circuits, these quantities are treated as per-execution metadata rather than inferred directly from the logical circuit specification.

G Qualitative Analysis Protocol and Failure Taxonomy

This section describes the qualitative analysis used to examine how JointQ-Multi redistributes probability among competing structured interpretations and to characterize the principal sources of end-to-end failure. For both Spider and nvBench, we use the same cached candidate pools, canonicalized structures, verified validity assignments, relational graphs, and disagreement features used in the quantitative experiments. No candidates are added, removed, or re-labeled specifically for the qualitative analysis.

G.1 Case-Selection Protocol

We examine examples from the low-, medium-, and high-candidate-set ambiguity groups defined in equation 201, together with representative failure cases. Throughout this analysis, validity refers specifically to the verified valid-output set $\mathcal{Y}_{\mathrm{ver}}(x)$ defined in Appendix F.4.

A *successful case* is an input for which the final prediction satisfies

$$\mathrm{Canon}(\hat{s}) \in \mathrm{Canon}\left(\mathcal{Y}_{\mathrm{ver}}(x)\right). \tag{273}$$

A *reasoning failure* occurs when at least one verified valid candidate is present in the retained pool but the model selects an invalid candidate:

$$\mathcal{S}(x) \cap \mathcal{Y}_{\mathrm{ver}}(x) \neq \emptyset, \qquad \hat{s} \notin \mathcal{Y}_{\mathrm{ver}}(x). \tag{274}$$

Inputs satisfying

$$\mathcal{S}(x) \cap \mathcal{Y}_{\mathrm{ver}}(x) = \emptyset \tag{275}$$

are treated separately as *candidate-coverage failures*, because no downstream candidate-selection method operating on the same pool can return a verified valid output.

For each examined covered example, we record: (i) the retained candidate structures; (ii) their verified validity labels; (iii) the dominant pairwise structural disagreement types; (iv) candidate probabilities before and after joint ambiguity reasoning; (v) layer-wise candidate-probability evolution; and (vi) predictions from the principal classical and quantum baselines. This protocol separates failures of candidate generation from failures of candidate discrimination and selection.

G.2 Probability Redistribution under Joint Reasoning

Under the default uniform candidate prior, each of the K active candidates is initialized with

$$\pi_i = \frac{1}{K}, \qquad i \in \{1, \ldots, K\}. \tag{276}$$

Because the candidate-register marginal of the prepared state equals the specified prior, we write

$$p_i^{(0)} = \pi_i. \tag{277}$$

For a full candidate pool with $K = 5$, this gives $p_i^{(0)} = 0.2$ for every active candidate.

We quantify the probability redistribution induced by the complete joint reasoning circuit for candidate s_i as

$$\Delta p_i = p_i^{(L)} - p_i^{(0)} = p_i^{(L)} - \pi_i, \tag{278}$$

where $p_i^{(L)}$ is the candidate-register marginal after the final ambiguity-reasoning layer. A positive Δp_i indicates that the candidate gains probability mass relative to the uniform prior, whereas a negative value indicates suppression.

For the verified valid-candidate index set $\mathcal{I}_{\mathrm{valid}}(x)$, we define the aggregate valid mass after layer ℓ as

$$P_{\mathrm{valid}}^{(\ell)} = \sum_{i \in \mathcal{I}_{\mathrm{valid}}(x)} p_i^{(\ell)}. \tag{279}$$

The total change in verified valid probability induced by joint reasoning is therefore

$$\Delta P_{\mathrm{valid}} = P_{\mathrm{valid}}^{(L)} - P_{\mathrm{valid}}^{(0)} = \sum_{i \in \mathcal{I}_{\mathrm{valid}}(x)} p_i^{(L)} - \frac{|\mathcal{I}_{\mathrm{valid}}(x)|}{K}. \tag{280}$$

A positive $\Delta P_{\mathrm{valid}}$ indicates that joint reasoning moves aggregate probability mass toward candidates verified as valid. Importantly, this quantity does not require probability to be distributed uniformly among the valid candidates, which is consistent with the multi-valid objective.

G.3 Interpreting Structural Disagreement

To examine which structural differences are most strongly weighted by the learned ambiguity-reasoning circuit, we define the magnitude of the disagreement-conditioned coefficient associated with candidate pair (i, j), relation type r, and reasoning layer ℓ as

$$\gamma_{ij}^{(r,\ell)} = \left| \theta_r^{(\ell)} d_{ij}^{(r)} \right|. \tag{281}$$

The absolute value is used because $\theta_r^{(\ell)} d_{ij}^{(r)}$ is a signed interaction coefficient, whereas the present analysis concerns its magnitude.

We aggregate this quantity across the L reasoning layers:

$$\Gamma_{ij}^{(r)} = \sum_{\ell=1}^{L} \gamma_{ij}^{(r,\ell)}. \tag{282}$$

For a fixed candidate pair, relation types with larger $\Gamma_{ij}^{(r)}$ correspond to disagreement dimensions receiving larger learned interaction magnitudes across the circuit.

For Spider, the disagreement dimensions correspond to join structure, grouping, aggregation, condition attachment, and value relations. For nvBench, they correspond to attribute selection, encoding assignment, aggregation, grouping, and filtering.

These quantities should be interpreted as diagnostics of the parameterized interaction mechanism rather than as causal feature attributions. They identify which available disagreement dimensions receive the strongest learned interaction coefficients for a candidate pair, while the final candidate probability results from the complete joint circuit and interactions among all active candidates.

G.4 Layer-Wise Probability Evolution

We additionally inspect how candidate probabilities change across successive ambiguity-reasoning layers. Let

$$p_i^{(\ell)} = \langle \Psi_\ell | \left(|i-1\rangle\langle i-1|_c \otimes I_f \right) | \Psi_\ell \rangle \tag{283}$$

denote the candidate-register marginal probability of candidate s_i after completion of reasoning layer ℓ. The initial value $p_i^{(0)} = \pi_i$ corresponds to the prepared joint state before ambiguity-conditioned interactions are applied.

For the default $L = 2$ configuration, the candidate trajectory is

$$p_i^{(0)} \longrightarrow p_i^{(1)} \longrightarrow p_i^{(2)}. \tag{284}$$

At the same time, the aggregate verified valid probability evolves as

$$P_{\text{valid}}^{(0)} \longrightarrow P_{\text{valid}}^{(1)} \longrightarrow P_{\text{valid}}^{(2)}, \tag{285}$$

where

$$P_{\text{valid}}^{(\ell)} = \sum_{i \in \mathcal{I}_{\text{valid}}(x)} p_i^{(\ell)}. \tag{286}$$

This layer-wise view distinguishes cases in which verified valid candidates are promoted progressively across reasoning layers from cases in which an early probability shift is weakened or reversed by a subsequent interaction layer. It also provides a candidate-level counterpart to the circuit-depth analysis reported in the quantitative experiments.

G.5 Comparison of Candidate-Level Reasoning

For each examined case, we compare candidate-level predictions from Graph+Multi-valid, Classical Joint-Set, Candidate-wise Quantum, Joint Quantum+Single, and JointQ-Multi using exactly the same retained candidate pool. The comparisons follow the controlled protocol described in Appendix D.12.

Comparing GRAPH+MULTI-VALID with CLASSICAL JOINT-SET examines whether introducing classical cross-candidate interaction changes the ranking obtained from independent graph representations while retaining multi-valid supervision.

Comparing CANDIDATE-WISE QUANTUM with JOINT QUANTUM+SINGLE isolates the effect of replacing independent quantum candidate processing with a joint candidate-indexed state and disagreement-conditioned cross-candidate interactions under the same single-reference supervision.

Comparing JOINT QUANTUM+SINGLE with JOINTQ-MULTI holds the joint quantum reasoning architecture fixed and isolates the effect of multi-valid supervision.

Finally, comparing CLASSICAL JOINT-SET with JOINTQ-MULTI contrasts classical and quantum joint reasoning when the candidate pools, projected candidate representations, structural disagreement features, and multi-valid supervision are held fixed. As described in Appendix D.5, this comparison is interaction-parameter-matched: both cross-candidate interaction modules contain LR trainable disagreement-dependent coefficients, although their complete model parameterizations are not claimed to be identical.

The candidate-level analysis therefore focuses on whether and how joint reasoning changes the relative ranking of competing interpretations, how those changes relate to the verified validity of the candidates, and which structural disagreement dimensions receive the largest learned interaction magnitudes.

G.6 REPRESENTATIVE CANDIDATE-LEVEL ANALYSIS

To illustrate how the proposed joint reasoning mechanism operates at the candidate level, we consider representative ambiguity patterns consistent with the candidate structures and disagreement types used throughout the experiments.

Spider Example: Resolving a Join–Condition Ambiguity. Consider an input for which the retained pool contains five SQL candidates differing primarily in join structure and condition attachment. Two candidates, s_2 and s_4, are verified as valid, while the remaining candidates contain incorrect join paths or attach the filtering condition to an incorrect relation. Under the uniform candidate prior,

$$p_i^{(0)} = 0.20, \qquad i \in \{1, \dots, 5\}. \tag{287}$$

A representative layer-wise evolution is

$$\begin{aligned} \mathbf{p}^{(0)} &= [0.20,\ 0.20,\ 0.20,\ 0.20,\ 0.20], \\ \mathbf{p}^{(1)} &= [0.14,\ 0.27,\ 0.16,\ 0.29,\ 0.14], \\ \mathbf{p}^{(2)} &= [0.10,\ 0.31,\ 0.13,\ 0.35,\ 0.11]. \end{aligned} \tag{288}$$

The aggregate probability assigned to the verified valid candidates therefore changes from

$$P_{\text{valid}}^{(0)} = 0.40 \tag{289}$$

to

$$P_{\text{valid}}^{(2)} = 0.66. \tag{290}$$

Thus,

$$\Delta P_{\text{valid}} = +0.26. \tag{291}$$

The largest accumulated disagreement-conditioned interaction magnitudes occur for join structure and condition attachment. In this case, the two invalid candidates most similar to the final winner differ from it primarily in their join path and placement of the filtering predicate. Joint reasoning suppresses these structurally conflicting interpretations while increasing probability on both verified valid alternatives. The final prediction is s_4.

Table 19: Representative candidate-level behavior of JOINTQ-MULTI.

Case	$P_{\text{valid}}^{(0)}$	$P_{\text{valid}}^{(L)}$	ΔP_{valid}	**Outcome**
Spider join/condition ambiguity	0.40	0.66	+0.26	Valid
nvBench encoding ambiguity	0.40	0.64	+0.24	Valid
Reasoning-failure case	0.20	0.29	+0.09	Invalid

nvBench Example: Multiple Valid Visualization Specifications. Consider a visualization request admitting two verified specifications, s_1 and s_3, that express the same intended analysis using different but supported encoding assignments. The remaining candidates differ in aggregation, grouping, or filtering.

A representative probability trajectory is

$$\begin{aligned}\mathbf{p}^{(0)} &= [0.20,\ 0.20,\ 0.20,\ 0.20,\ 0.20],\\ \mathbf{p}^{(1)} &= [0.28,\ 0.15,\ 0.26,\ 0.16,\ 0.15],\\ \mathbf{p}^{(2)} &= [0.33,\ 0.11,\ 0.31,\ 0.13,\ 0.12].\end{aligned} \tag{292}$$

The verified-valid probability mass increases from

$$P_{\text{valid}}^{(0)} = 0.40 \tag{293}$$

to

$$P_{\text{valid}}^{(2)} = 0.64. \tag{294}$$

The dominant disagreement dimensions are aggregation and encoding assignment. Importantly, the multi-valid objective does not require one verified candidate to absorb all probability mass. Instead, both valid interpretations remain competitive while candidates containing incompatible aggregation or filtering choices are suppressed.

Reasoning-Failure Example. Joint reasoning does not resolve every covered candidate set. Consider a case with one verified valid candidate, s_2, and an invalid candidate s_3 that differs only through a structural distinction that is weakly represented by the disagreement features. A representative trajectory is

$$\begin{aligned}\mathbf{p}^{(0)} &= [0.20,\ 0.20,\ 0.20,\ 0.20,\ 0.20],\\ \mathbf{p}^{(1)} &= [0.14,\ 0.25,\ 0.27,\ 0.18,\ 0.16],\\ \mathbf{p}^{(2)} &= [0.11,\ 0.29,\ 0.34,\ 0.14,\ 0.12].\end{aligned} \tag{295}$$

Although the verified candidate gains probability relative to the uniform prior, the competing invalid interpretation gains more and is ultimately selected. Such a case corresponds to an incorrect cross-candidate preference when the valid–invalid pair has nonzero measured structural disagreement, or to insufficient structural distinguishability when the corresponding disagreement vector is zero under the taxonomy of Appendix G.7.

These examples illustrate three behaviors captured by the proposed diagnostics. First, joint reasoning can transfer probability away from structurally conflicting candidates toward the verified-valid subspace. Second, under multi-valid supervision, this probability need not collapse onto a single verified interpretation. Third, increasing the probability of a valid candidate does not guarantee correct final selection when a competing invalid candidate receives an even larger increase. The aggregate failure analysis in Appendix G.8 quantifies how frequently the corresponding failure categories occur over the complete evaluation sets.

G.7 FAILURE ANALYSIS

We analyze failures according to the stage at which the structured-generation pipeline becomes unable to return a verified valid output. Throughout this analysis, validity refers specifically to the verified valid-output set $\mathcal{Y}_{\text{ver}}(x)$ defined in Appendix F.4. Under exact state-vector inference, the categories below are applied hierarchically so that each failed example is assigned to exactly one category.

Candidate-Coverage Failure. A candidate-coverage failure occurs when the retained candidate set contains no verified valid interpretation:

$$\mathcal{S}(x) \cap \mathcal{Y}_{\mathrm{ver}}(x) = \emptyset. \tag{296}$$

In such cases, neither a classical nor a quantum candidate-selection method operating on $\mathcal{S}(x)$ can return an output supported as valid under the verification protocol. These cases are therefore attributed to candidate generation rather than downstream ambiguity reasoning.

Inadequate Structural Distinguishability. We next consider failed examples for which at least one verified valid candidate is present. Let

$$j^* = \arg\max_j p_j \tag{297}$$

denote the candidate selected under exact state-vector inference. For a failed covered example, $j^* \notin \mathcal{I}_{\mathrm{valid}}(x)$. We measure the structural separation between the selected invalid candidate and its closest verified-valid competitor as

$$D_{\mathrm{VI}}(x) = \min_{i \in \mathcal{I}_{\mathrm{valid}}(x)} \|\mathbf{d}_{ij^*}\|_1 . \tag{298}$$

We classify the failure as one of *inadequate structural distinguishability* when

$$D_{\mathrm{VI}}(x) = 0. \tag{299}$$

That is, the disagreement representation assigns no typed structural difference between the selected invalid candidate and at least one verified valid candidate. Because the proposed cross-candidate interactions are conditioned on $\mathbf{d}_{ij}$, the reasoning module receives no disagreement signal with which to distinguish that valid–invalid pair.

Incorrect Cross-Candidate Preference. For a covered example with $D_{\mathrm{VI}}(x) > 0$, the selected invalid candidate is structurally distinguishable from every verified valid candidate under the disagreement representation. Failure in this case is attributed to incorrect cross-candidate preference.

To quantify this behavior, we define the valid–invalid probability margin

$$\Delta_{\mathrm{valid}}(x) = \max_{i \in \mathcal{I}_{\mathrm{valid}}(x)} p_i - \max_{j \notin \mathcal{I}_{\mathrm{valid}}(x)} p_j. \tag{300}$$

A negative value, $\Delta_{\mathrm{valid}}(x) < 0$, indicates that the highest-probability invalid candidate outranks every verified valid candidate. Under the hierarchical classification above, a failed covered example with $D_{\mathrm{VI}}(x) > 0$ is therefore categorized as an incorrect cross-candidate preference.

Measurement- or Noise-Induced Failure. Finite-shot estimation and quantum-execution noise are analyzed separately from the exact-inference categories above. A measurement- or noise-induced failure occurs when exact state-vector inference selects a verified valid candidate but the corresponding finite-shot or noisy execution selects an invalid candidate.

Let i^* denote the highest-probability candidate under exact state-vector inference. Its exact selection margin is

$$\Delta(x) = p_{i^*} - \max_{j \neq i^*} p_j. \tag{301}$$

Small values of $\Delta(x)$ indicate closely competing candidates and therefore greater sensitivity to finite-shot fluctuations or execution noise. This characterization is consistent with the finite-shot sensitivity analysis in Appendix E.5 and Section 5.

G.8 Success and Failure Analysis

We further examine the end-to-end behavior of JointQ-Multi by separating successful resolutions from failures occurring at different stages of the pipeline. Under exact state-vector inference, an example is considered successfully resolved when

$$\mathrm{Canon}(\hat{s}) \in \mathrm{Canon}\left(\mathcal{Y}_{\mathrm{ver}}(x)\right), \tag{302}$$

Table 20: Distribution of successful resolutions and failure modes under exact state-vector inference. Values denote percentages of all evaluated examples. Failure categories are assigned hierarchically and are mutually exclusive.

Outcome	**nvBench**	**Spider**
Successful resolution	**72.6**	**81.5**
Candidate-coverage failure	21.4	13.9
Insufficient structural distinguishability	2.3	1.8
Incorrect cross-candidate preference	3.7	2.8
Total	100.0	100.0

which is equivalent to success under the Multi-EM criterion.

Exact-inference failures are assigned hierarchically to three mutually exclusive categories. First, an example is labeled a *candidate-coverage failure* if $\mathcal{S}(x) \cap \mathcal{Y}_{\mathrm{ver}}(x) = \emptyset$. Among the remaining covered examples, a failed prediction is labeled *insufficient structural distinguishability* when $D_{\mathrm{VI}}(x) = 0$ according to equation 298. All remaining failed covered examples, for which $D_{\mathrm{VI}}(x) > 0$, are classified as *incorrect cross-candidate preference*. These rules partition successful and failed exact-inference examples without overlap. Measurement- and noise-induced failures are evaluated separately because they arise only after replacing exact state-vector probabilities with finite-shot or noisy quantum execution.

Table 20 shows that candidate generation accounts for the largest share of the remaining errors. The retained candidate set contains no verified valid interpretation for 21.4% of nvBench examples and 13.9% of Spider examples. These percentages correspond directly to the complement of Coverage@5: once no verified valid interpretation is present in the candidate pool, downstream candidate selection cannot recover one.

Failures arising after successful candidate coverage are substantially less frequent. Insufficient structural distinguishability accounts for 2.3% of all nvBench examples and 1.8% of all Spider examples. In these cases, the invalid candidate selected by the model has zero measured typed disagreement with at least one verified valid candidate, leaving the ambiguity-reasoning circuit without a structural disagreement signal for that pair.

Incorrect cross-candidate preference accounts for a further 3.7% of nvBench examples and 2.8% of Spider examples. These cases contain at least one verified valid candidate and provide a nonzero structural disagreement signal between the selected invalid candidate and every verified valid alternative, but the learned candidate distribution still ranks an invalid interpretation highest.

To isolate candidate selection from candidate generation, we compute the valid-selection accuracy conditional on successful candidate coverage:

$$\mathrm{SelAcc} = \frac{\sum_x \mathrm{MultiEM}(x)}{\sum_x \mathrm{Coverage@5}(x)}. \tag{303}$$

Because Multi-EM success necessarily implies successful candidate coverage, this quantity measures the fraction of covered examples for which the reasoning model selects a verified valid candidate. It yields approximately 92.4% on nvBench and 94.7% on Spider. Thus, JOINTQ-MULTI resolves most candidate sets once at least one verified valid interpretation is available, while the largest remaining source of end-to-end error is candidate coverage.

Execution-Induced Failures. Finite-shot sampling and quantum-execution noise introduce a separate class of failures not included in Table 20. An execution-induced failure occurs when exact state-vector inference selects a verified valid candidate but finite-shot sampling or noisy execution changes the winner to an invalid candidate. Such failures are expected to occur more frequently when the exact probability margin in equation 301 is small. This behavior is examined separately in the finite-shot, synthetic-noise, and physical-device experiments through both task performance and selection agreement.

G.9 Interpretation and Scope

The qualitative analysis complements aggregate accuracy by examining how candidate probabilities change when competing interpretations differ in localized structural choices. Particularly informative cases involve independently processed candidates with similar scores despite differences in aggregation, grouping, joins, conditions, attributes, or visualization encodings. Comparing these cases across independent and joint reasoning models helps determine whether ranking changes are associated with explicit relationships among competing hypotheses rather than candidate plausibility alone.

The failure taxonomy also shows that joint ambiguity reasoning does not eliminate the fundamental limitations of candidate-based structured generation. Performance still depends on valid candidate coverage and whether the relational representation captures the structural distinctions that determine semantic validity. Small candidate-probability margins also remain sensitive to finite-shot and quantum-execution noise. These limitations motivate future work on improved candidate generation, selective candidate interactions, richer disagreement representations, and noise-aware ambiguity reasoning.