A Modular Multi-Tool Evolutionary Algorithm Framework for RNA Inverse Folding

Kaiyu Nie
Department of Electrical and Computer Engineering
Concordia University
Montreal, Canada
ka_nie@mail.concordia.ca

Nawwaf Kharma*
Department of Electrical and Computer Engineering
Concordia University
Montreal, Canada
nawwaf.kharma@concordia.ca

***Abstract*—Most RNA inverse folding methods rely on a single structure predictor, often resulting in sequences that overfit specific algorithmic biases. To address this, we present a predictor-agnostic Evolutionary Algorithm framework that decouples optimization logic from folding engines. On the Eterna100 benchmark, our framework achieves a 68% solve rate using single-tool modes, matching the performance of state-of-the-art reinforcement learning methods. Crucially, to mitigate the risk of model-specific errors, we introduce a consensus optimization strategy that seeks sequences validated by multiple, distinct folding models (e.g., thermodynamic and deep learning). By filtering out predictor-specific artifacts, this approach acts as a structural "safety net," effectively reducing the occurrence of extreme prediction outliers and yielding more robust candidates for downstream experimental testing. Finally, the framework features a modular "plug-and-play" architecture that supports granular, nucleotide-level constraint control, enabling precise customization for diverse design tasks.**

***Index Terms*—RNA Inverse Folding, Evolutionary Algorithms, Consensus Optimization, RNA Design, Multi-Model Framework**

I. Introduction

A. Background and Motivation

The RNA inverse folding problem seeks to generate nucleotide sequences whose predicted secondary structure matches a predefined target structure. Formally, given a target structure S, the objective is to find sequences **x** such that a folding function f(x) = S under a specified RNA structure prediction model [2] [3]. This problem constitutes a core task in computational RNA design and has been addressed using evolutionary algorithms, constraint programming, and more recently, deep generative models [1] [4] [5] [6] [20].

RNA inverse folding is essential for numerous applications in synthetic biology and biomedicine. In mRNA therapeutics, RNA secondary structure influences translation efficiency, molecular stability, and immune response [7] [8]. In ribozyme engineering and RNA-based biosensors, functional activity depends critically on the formation of specific structural motifs [9] [10]. These applications require not only structural correctness but also robustness against the uncertainty inherent in the use of incomplete models for the prediction of RNA structure.

*Corresponding author.

A major challenge in RNA inverse folding arises from the inconsistency among RNA secondary structure predictors. Widely used tools such as ViennaRNA [3], CONTRAfold [11], IPknot [12], ProbKnot [13], and MXfold2 [14] rely on different thermodynamic assumptions, probabilistic models, or learning-based objectives. This creates a cross-validation failure: a sequence optimized for one model often fails when validated by another predictor. This inconsistency reduces the likelihood of correct functionality of the designed sequences. Solutions optimized for one model often fail to transfer to others.

Most existing inverse folding methods are tightly coupled to a single structure predictor, which serves as the sole oracle for optimization or training [1] [2] [5]. Such coupling implicitly assumes that the selected predictor provides a sufficient and accurate representation of RNA folding behavior. In practice, this assumption rarely holds, especially for structures involving pseudoknots, long-range interactions, or non-canonical base pairs. As a result, many designed sequences exhibit strong predictor-specific overfitting and lack robustness across folding models.

Current methods suffer from predictor-specific overfitting. By relying on a single oracle, these algorithms essentially gamble on the accuracy of one specific model. There is a critical need for a predictor-independent framework that accounts for the inherent uncertainty of computational structure prediction. Addressing this gap is critical for improving the generalizability of RNA design methods and for enabling more reliable deployment in experimental settings.

B. Our Approach

To bridge this gap, we propose a modular, tool-independent Evolutionary Algorithm (EA) framework, called cFold (consensus Fold). Instead of being "married" to one predictor, our framework treats folding engines as "pluggable modules." This allows users to:

1) Swap engines (thermodynamic vs. ML-based) without changing the core code;
2) Integrate a weighted penalty mechanism into the fitness function, enabling the algorithm to handle complex, user-defined constraints that may conflict with pure structural stability;

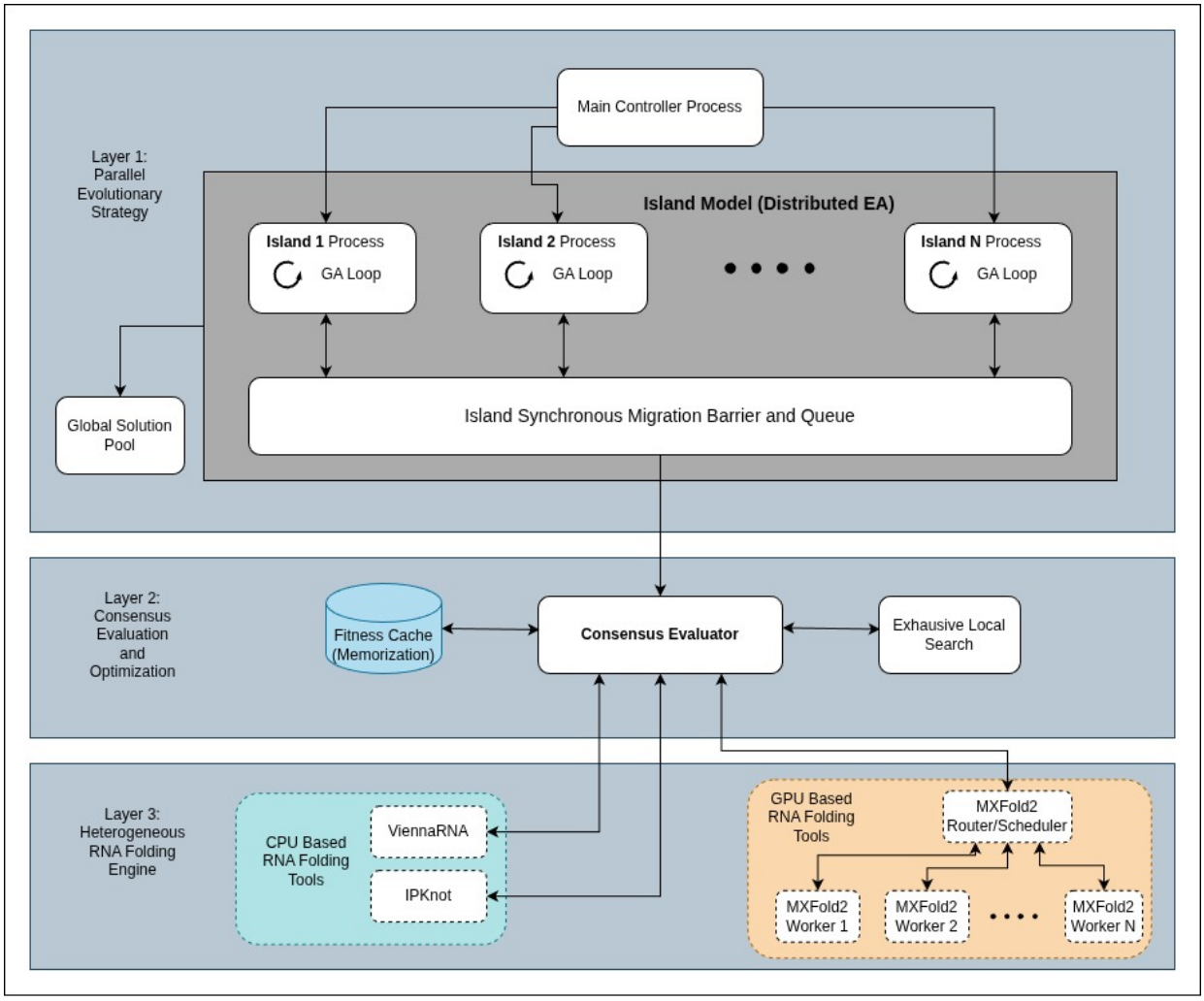

Fig. 1. Architecture of the proposed modular EA framework. Layer 1 implements a parallel island model with shared solution pool. Layer 2 handles fitness evaluation with optional caching and local search. Layer 3 contains the pluggable folding backends (dashed border), supporting both CPU-based tools and GPU model servers. Bidirectional arrows indicate data flow in both **directions**. Sequences are **sent** downwards to the RNA folding engine, and structures are **sent** upwards to the evaluator.

3) Optimize for consensus, seeking sequences that satisfy multiple models simultaneously to improve experimental robustness.

C. Contributions

- **A Modular "Plug-and-Play" Architecture:** We deliver the first inverse folding framework that decouples EA logic from the folding oracle. It supports a diverse ecosystem of tools (ViennaRNA [3], MXFold2 [14]) via a unified interface.
- **Granular Constraint Optimization via Weighted Penalties:** We introduce a position-wise weighting scheme that transforms binary constraints into a continuous loss landscape. Unlike traditional methods that strictly prune the search space, our approach allows the optimizer to navigate partial constraint violations. This is critical for solving over-constrained puzzles, where the algorithm must balance structural fidelity against sequence composition requirements (e.g., satisfying IUPAC codes while minimizing folding energy).
- **Risk Mitigation via Consensus Optimization:** We demonstrate that optimizing for the intersection of discordant predictors functions as a **"safety net."** Instead of merely chasing average accuracy, this strategy significantly raises the *reliability floor*: it reduces the worst-case prediction error by 17–34% (118 vs. 142/180) compared to single-tool baselines, providing a more robust pathway for experimental RNA design.

II. Methods

A. Framework Overview

Our framework follows a modular design that separates the evolutionary search logic from the structure prediction backend. Fig 1 illustrates the overall architecture.

The core principle is decoupling: the EA handles population management, selection, mutation, and fitness aggrega-

tion, while folding predictors are encapsulated as independent evaluation modules. This separation ensures that adding or replacing a predictor does not require modifications to the core algorithm.

B. Modular Tool Integration

To achieve tool independence, we abstract the folding predictor into a unified interface contract. Rather than hard-coding dependencies, the framework interacts with predictors through a standardized abstraction layer.

a) Standardized Interface: We define a generic predictor interface that maps an RNA sequence to its secondary structure. Integrating a new tool requires only encapsulating its logic within a wrapper class that adheres to this interface. This design effectively decouples the evolutionary search logic from the underlying prediction mechanics. The EA explicitly remains oblivious to the specific tool being used, interacting solely with the abstract wrapper.

b) Hybrid Execution Environment.: The framework supports a heterogeneous computing environment:

- **Direct Execution (CPU):** Lightweight tools (e.g., ViennaRNA) are invoked directly via system calls for minimal latency.
- **Asynchronous Inference Service (GPU):** For heavy deep learning models (MXFold2, UFold), we implement a Client-Server architecture to reduce initialization overhead. A persistent model server performs **micro-batching**—aggregating incoming sequence queries from multiple EA workers into a single tensor. This allows for parallel inference on the GPU, significantly increasing throughput compared to sequential execution.

C. Fitness Evaluation

1) Single-Tool Fitness: When a single predictor is enabled, fitness is computed as the structural distance between the predicted and target structures:

$$f(\mathbf{x}) = d\big(\text{Fold}(\mathbf{x}), S_{\text{target}}\big) \tag{1}$$

where $d(\cdot,\cdot)$ denotes the base-pair distance, defined as the number of base pairs present in one structure but absent in the other.

2) Multi-Tool Consensus Fitness: When multiple predictors are enabled, we compute a **consensus structure** by taking the intersection of the predicted base pairs across all tools. This strict intersection policy effectively acts as a high-pass filter, accepting only base pairs agreed upon by all tools. This filters out model-specific errors.

$$P_{\text{consensus}} = \bigcap_{t \in \mathcal{T}} P_t(\mathbf{x}). \tag{2}$$

Here, $P_t(\mathbf{x})$ denotes the set of base pairs predicted by tool t, and $\mathcal{T}$ represents the set of enabled tools. We enforce a strict intersection policy ($\bigcap$) rather than majority voting. This choice is motivated by both theoretical and practical considerations. From a theoretical perspective, majority voting is not well-defined in a dual-predictor framework without an additional tie-breaking model. From a practical perspective, strict consensus emphasizes precision over recall, which is preferable in wet-lab settings, where false positives incur substantial experimental cost, while false negatives impose only limited computational cost. The fitness function is then defined as

$$f(\mathbf{x}) = d\big(P_{\text{consensus}}, P_{\text{target}}\big). \tag{3}$$

where P_{target} denotes the set of base pairs specified by the target structure. This intersection-based consensus is strict: a base pair is included only if it is predicted by *all* enabled tools. Consequently, a sequence achieves zero fitness (the optimum fitness value) only when every tool independently predicts the target structure exactly.

3) Weighted Fitness with Constraints: To support position-level constraints under both single-tool and consensus folding modes, we define a unified fitness function:

$$F(\mathbf{x}) = \underbrace{d(S_{\text{pred}}, S_{\text{target}})}_{\text{Structural distance}} + \underbrace{\sum_{(i,j)\in\Delta} w_{ij}^{s}}_{\text{Constraint penalty}} + \lambda \cdot \underbrace{\sum_{i:x_i \notin \text{IUPAC}_i} w_i^{q}}_{\text{Sequence penalty}}. \tag{4}$$

Here,

- S_{pred} denotes the predicted structure, obtained either from a single folding tool or from the consensus intersection set $P_{\text{consensus}}$ in multi-tool mode;
- Δ is the set of base pairs required by the target structure but missing in S_{pred};
- $w_{ij}^{s} \in [0,1]$ is a user-defined weight that reflects the importance of critical motifs (e.g., binding sites) even when the consensus predictor is uncertain;
- w_i^{q} penalizes violations of position-wise IUPAC constraints, and λ controls the trade-off between structural and sequence-level penalties.

This formulation decouples constraint enforcement from the underlying prediction logic, enabling constrained optimization under both single-tool and consensus folding paradigms.

D. Evolutionary Operators

1) Structure-Aware Adaptive Mutation: Rather than employing a uniform mutation rate, we adapt the mutation probability at each position according to the current mismatch pattern, as summarized in Table I.

TABLE I
ADAPTIVE MUTATION RATES BASED ON STRUCTURAL CONTEXT.

Situation	Rate	Rationale
Correct within a perfect motif	0.5%	Preserve high-quality regions
Correct pair in an imperfect stem	1%	Minor refinement
Completely mismatched region	10%	Encourage exploration
Incorrect base pair	25%	Targeted repair

For positions (i, j) constrained to form a base pair, we employ a compensatory mutation strategy: a mutation is applied to one nucleotide, and its partner is deterministically updated to a compatible complement to enforce a valid Watson–Crick

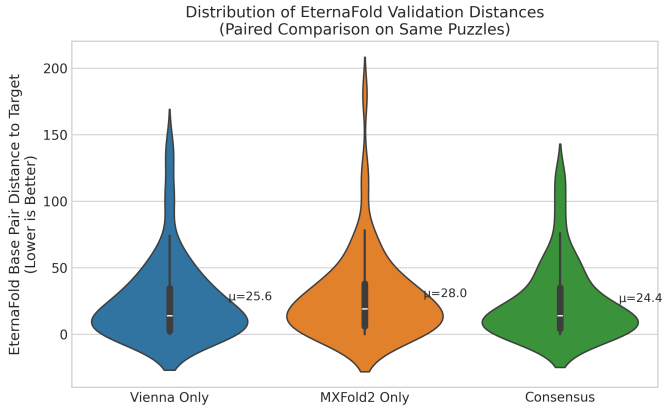

Fig. 2. **Distribution of EternaFold validation distances.** The violin plot reveals that while median performance is similar across methods, the Consensus approach (green) exhibits a significantly shorter and narrower upper tail compared to Vienna (blue) and MXFold2 (orange). This visually confirms the reduction in high-error outliers reported in Table IV.

TABLE II
EXPERIMENTAL CONFIGURATIONS.

Configuration	Fitness Tool(s)	Purpose
Vienna-only	ViennaRNA	Thermodynamic baseline
MXFold2-only	MXFold2	ML and thermodynamic mixed baseline
Consensus	ViennaRNA + MXFold2	Multi-tool optimization

or Wobble pair (A–U, G–C, or G–U). This strategy preserves structural validity throughout the search process.

2) Selection and Population Management: We employ tournament selection with a window size of k [19] to balance selection pressure and population diversity. In each generation, the following steps are performed:

1) Select parents using k-tournament selection with $k = 5$;
2) Generate $\lambda = 7 \times \mu$ offspring through mutation;
3) Evaluate the fitness of the offspring utilizing the enabled folding tools;
4) Select the best μ individuals for the next generation while preserving elite solutions.

3) Parallel Island Model: To prevent premature convergence and exploit multi-core architectures, we employ a parallel Island Model [19]. The population is partitioned into distinct subpopulations that evolve independently (with occasional migration) using the operators described above. To share genetic information, we employ a synchronous ring migration topology: every G generations, the elite individual from each island is transferred to its neighbor, replacing the worst-performing individual in the destination island. This mechanism balances local exploration with global exploitation.

III. EXPERIMENT

A. Experimental Setup

a) Benchmark: We evaluate our framework on the Eterna100 benchmark [18], a collection of 100 RNA design puzzles curated from the Eterna citizen science platform. The puzzles range from simple hairpins (16 nucleotides) to complex multi-branch structures (over 400 nucleotides), providing a comprehensive test of inverse folding capability.

b) Experimental Configurations: To evaluate both the flexibility of the framework and the effect of multi-tool consensus, we consider three experimental configurations as shown in Table II. While the Consensus configuration solves slightly fewer puzzles (64%), this reduction *is* expected as it imposes a stricter acceptance criterion, effectively filtering out solutions that are likely expressions of a single predictor's bias.

All configurations use identical EA parameters: population size $\mu = 100$, offspring count $\lambda = 700$, tournament size $k = 5$, and a maximum of 200 generations with early stopping (a patience of 20 generations).

c) Independent Validation: To assess whether solutions generalize beyond the training predictor(s), we validate all solved sequences using EternaFold [17]. Crucially, EternaFold is *not* used during optimization—it serves purely as an independent oracle.

TABLE III
SOLVE RATE COMPARISON ON ETERNA100. RESULTS FOR BASELINE METHODS ARE OBTAINED FROM [1]. OUR MODULAR FRAMEWORK (IN BOLD) MATCHES STATE-OF-THE-ART RL METHODS WHILE SUPPORTING ARBITRARY BACKEND PREDICTORS.

Method	Category	Solved	Rate
RNAinverse	Local search	60/100	60%
antaRNA	Ant colony	58/100	58%
MCTS-RNA	Tree search	57/100	57%
LEARNA	Reinforcement learning	67/100	67%
Meta-LEARNA	Variation of LEARNA	68/100	68%
Ours (Vienna)	**Modular EA (Vienna)**	**67/100**	**67%**
Ours (MXFold2)	**Modular EA (MXFold2)**	**68/100**	**68%**
Ours (Consensus)	**Modular EA (Both)**	**64/100**	**64%**

TABLE IV
VALIDATION RESULTS ON 63 COMMONLY SOLVED PUZZLES. CONSENSUS OPTIMIZATION REDUCES WORST-CASE FAILURES AND OUTLIER VARIANCE.

Metric	Vienna	MXFold2	Consensus
Mean distance	25.6 ± 31.0	28.0 ± 32.3	$\mathbf{24.4 \pm 28.6}$
90th percentile	58.0	64.6	**57.4**
Maximum distance	142	180	**118**
Failure Rate ($d > 30$)	33.3%	34.9%	**28.6%**

B. Solve Rate Comparison

Table III compares our framework with established baselines. Our framework achieves a solve rate of 67% with ViennaRNA and 68% with MXFold2, matching the performance of LEARNA and Meta-LEARNA.

C. Cross-Validation Analysis

To investigate whether consensus optimization yields more robust solutions, we analyze the 63 puzzles solved by all three configurations.

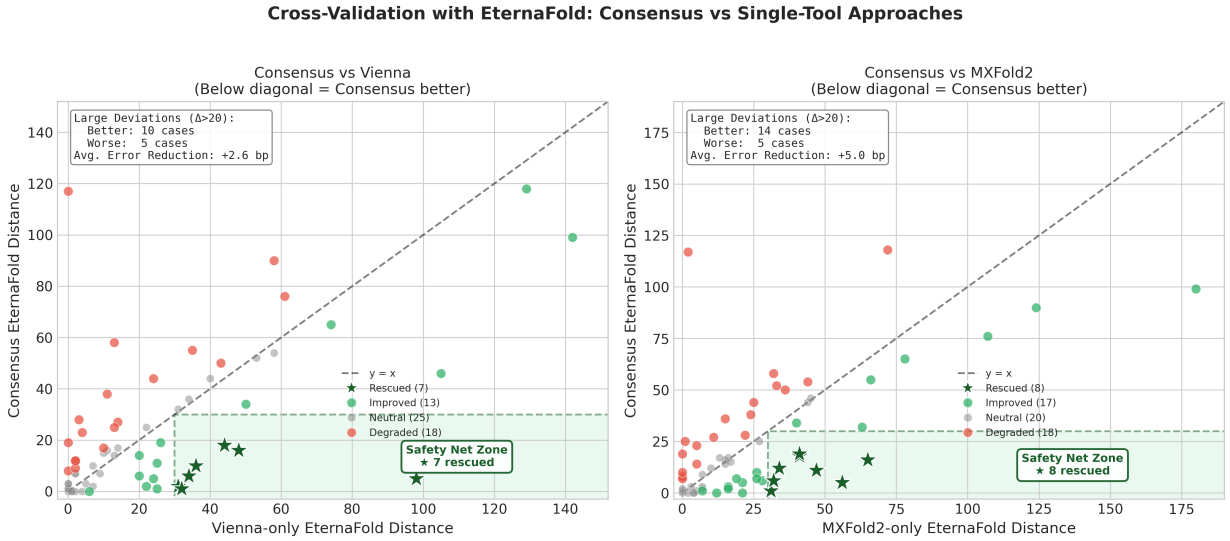

Fig. 3. **Paired validation comparison against independent oracle (EternaFold).** Each point represents one puzzle; points below $y = x$ indicate improved accuracy under Consensus. The "Safety Net Zone" (bottom-right) highlights catastrophic single-tool failures ($d > 30$ bp) successfully rescued by the consensus strategy (green stars).

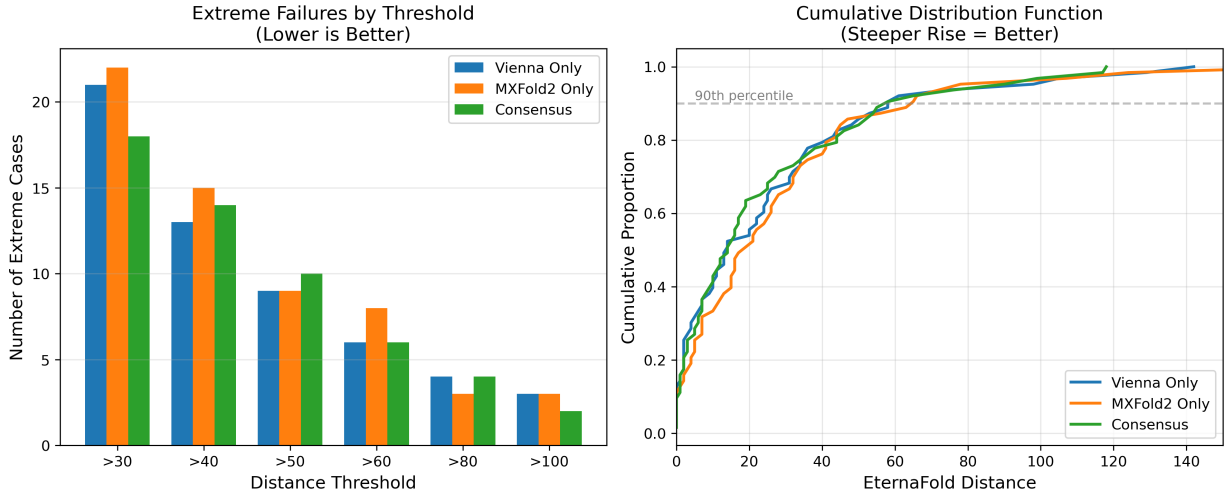

Fig. 4. **Analysis of extreme failures and solution robustness.** (Left) Puzzle counts exceeding various error thresholds; Consensus (green) suppresses extreme outliers. (Right) CDF of validation distances confirms systematically lower failure risk under Consensus optimization.

1) Definition of Extreme Failure: In experimental RNA design, small structural deviations are often tolerable, but large-scale misfolding renders sequences non-functional. We formally define an **"extreme failure"** as a solution whose validation distance exceeds a threshold of $d > 30$, corresponding to approximately 30% structural deviation for a typical 100-nucleotide target—a level at which the designed sequence is unlikely to be functionally viable. This threshold represents a catastrophic structural divergence where the predicted fold bears little functional resemblance to the target.

2) Quantitative Comparison: Table IV summarizes the validation metrics. While the consensus approach shows a modest improvement in mean distance, it significantly reduces the maximum error and the variance of the predictions.

3) Distribution of Validation Errors: To visualize the distribution of these errors, Fig. 2 presents the EternaFold validation distances.

D. Computational Analysis

The Consensus configuration (green) exhibits a much shorter upper tail compared to the single-tool baselines. This suggests that while Consensus optimization may not always find the absolute global minimum for every puzzle, it effectively suppresses the generation of high-risk sequences.

1) Paired Robustness Analysis: To understand individual puzzle performance, Fig. 3 presents a paired comparison between Consensus and single-tool baselines.

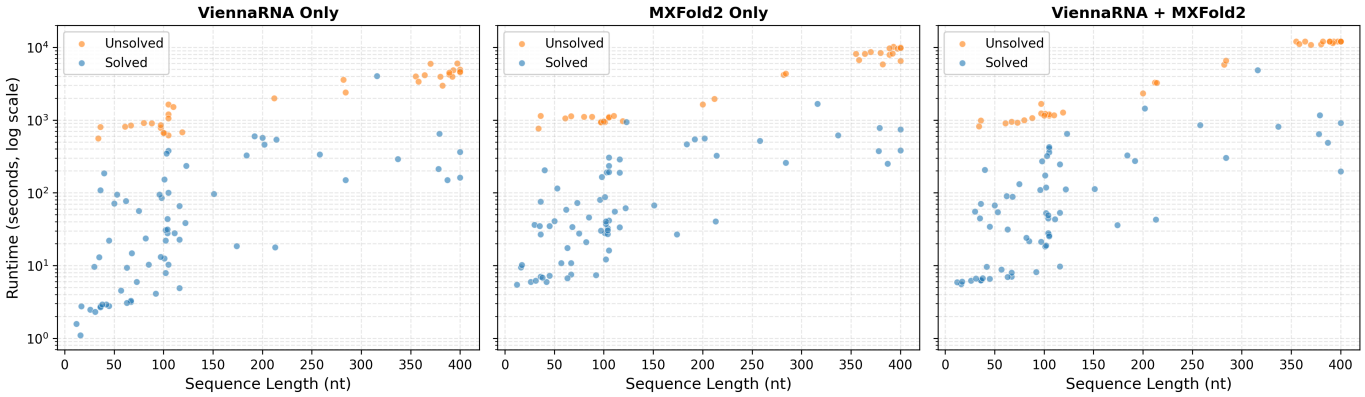

Fig. 5. **Computational cost analysis.** Runtime versus sequence length for Vienna-only (left), MXFold2-only (center), and Consensus (right) configurations. While Consensus optimization increases runtime linearly with the number of tools, the scaling behavior remains polynomial, ensuring scalability for typical RNA design tasks.

For the majority of "easy" puzzles (gray points near the origin), the Consensus approach performs comparably to single-tool baselines, indicating that the strict intersection constraint does not hinder performance on well-behaved targets. The primary advantage emerges in the "Safety Net Zone" (green shaded region).

In this region, a cluster of "rescued" solutions (marked as stars) is observed, where baseline predictors produced catastrophic failures (distances > 100 bp), while the Consensus strategy maintained functional structures (distances < 30 bp). Quantitative analysis (top-left inset) confirms this robust behavior: instances of significant improvement ($\Delta > 20$ bp) outnumber instances of degradation by approximately 2:1 against ViennaRNA and nearly 3:1 against MXFold2.

Although the strict consensus logic occasionally rejects a valid structure (visible as rare red outliers in the top-left), the average error reduction ($+2.6$ bp and $+5.0$ bp) demonstrates that the proposed framework effectively suppresses high-risk prediction artifacts, acting as a reliable safeguard for experimental design.

2) Analysis of Extreme Failures: We further quantify this robustness benefit in Fig 4. The bar chart (left) explicitly counts the number of failures at various thresholds.

Consensus optimization reduces the number of extreme failures ($d > 30$) by approximately 15–20% compared to single-tool baselines. The Cumulative Distribution Function (CDF) on the right confirms this stochastic dominance: the Consensus curve (green) saturates earlier than its counterparts, indicating a systematically lower probability of encountering large validation errors.

While Consensus optimization improves robustness, it incurs a computational cost. Fig 5 illustrates the relationship between sequence length and wall-clock time.

As expected, the Consensus strategy requires more time per generation. However, thanks to the parallel architecture and efficient GPU batching for MXFold2, the computational cost remains comparable to that of the standalone deep learning baseline, representing a justifiable trade-off for the significant reduction in experimental failure risk.

E. Discussion

Several factors may account for the limited statistical significance in the mean distance comparison:

- **Correlated errors**: ViennaRNA [3] and MXFold2 [14] may exhibit similar biases, as MXFold2 is partly trained on thermodynamic data, reducing the orthogonality required for effective consensus;
- **Stricter constraints**: The consensus criterion acts as a strict filter, which may exclude valid solutions that are correctly identified by only one predictor;
- **Sample size**: The paired evaluation set ($N = 63$) may lack sufficient statistical power to detect small effect sizes.

Despite the lack of statistically significant improvement in the *mean* performance, consensus optimization consistently reduces the **worst-case risk**, offering superior stability:

- **Reduced Maximum Error**: The worst-case prediction distance dropped to 118, representing a 17–34% reduction compared to single-tool baselines (142 and 180).
- **Lower Extreme Failure Rate**: The frequency of catastrophic failures ($d > 30$) decreased to 28.6%, compared to 33.3% and 34.9% for individual tools.
- **Lower Standard Deviation**: The standard deviation of prediction distances was reduced (28.6 vs. 31.0/32.3), indicating more consistent performance across diverse puzzles.

For applications where reliability is paramount—such as selecting candidates for costly wet-lab synthesis—consensus optimization provides a "safety net" that effectively filters out the most egregious prediction artifacts.

IV. Conclusion

In this work, we presented a modular Evolutionary Algorithm framework that effectively decouples optimization logic from the underlying folding oracle. Our results on the Eterna100 benchmark demonstrate that this predictor-agnostic

approach is highly competitive: when paired with state-of-the-art predictors, the framework achieves a solve rate of 68%, matching the performance of specialized Deep Reinforcement Learning methods (e.g., LEARNA). This confirms that generic evolutionary search, when properly engineered, remains a top-tier strategy for solving complex RNA inverse folding puzzles.

Beyond standard benchmarking, our investigation into consensus optimization highlights a critical trade-off between average accuracy and worst-case robustness. While combining thermodynamic and deep learning predictors does not guarantee higher mean accuracy due to potential correlated errors, it functions as a vital structural "safety net." By optimizing for the intersection of discordant models, the framework effectively filters out predictor-specific artifacts and significantly reduces the occurrence of catastrophic misfolding. This strategy shifts the paradigm from gambling on a single algorithm's bias to generating robust candidates that are more likely to survive the gap between in silico prediction and in vitro reality.

Finally, the framework's architecture offers lasting utility to the RNA design community. Its "plug-and-play" design acts as a universal adapter, ensuring that the tool remains future-proof as new, more accurate structure predictors emerge. Complementing this flexibility is our introduction of granular, nucleotide-level constraint control, which transforms rigid binary constraints into a continuous optimization landscape. This capability allows researchers to precisely enforce sequence motifs (e.g., for binding sites or synthesis requirements) without sacrificing structural stability, making the framework a versatile tool for diverse synthetic biology applications.

Future work will focus on two key directions. First, we aim to expand the consensus mechanism by incorporating a more diverse set of orthogonal predictors to reduce correlated biases. Second, and most importantly, we plan to move from computational benchmarking to wet-lab validation, synthesizing the designed sequences to verify whether the computational robustness observed in silico translates to physical stability in vitro.

REFERENCES

[1] F. Runge, D. Stoll, S. Falkner, and F. Hutter, "Learning to design RNA," in *Proc. Int. Conf. Learn. Representations (ICLR)*, 2019.

[2] I. L. Hofacker, "Vienna RNA secondary structure server," *Nucleic Acids Res.*, vol. 31, no. 13, pp. 3429–3431, 2003.

[3] R. Lorenz, S. H. Bernhart, C. Höner zu Siederdissen, H. Tafer, C. Flamm, P. F. Stadler, and I. L. Hofacker, "ViennaRNA Package 2.0," *Algorithms Mol. Biol.*, vol. 6, no. 1, p. 26, 2011.

[4] H. Huang, Z. Lin, D. He, L. Hong, and Y. Li, "RiboDiffusion: tertiary structure-based RNA inverse folding with generative diffusion models," *Bioinformatics*, vol. 40, no. Suppl. 1, pp. i347–i356, 2024.

[5] R. Kleinkauf, T. Houwaart, R. Backofen, and M. Mann, "antaRNA: ant colony-based RNA sequence design," *Bioinformatics*, vol. 31, no. 19, pp. 3114–3121, 2015.

[6] Á. Rubio-Largo, L. Escobar-Encinas, N. Lozano-García, and J. M. Granado-Criado, "Evolutionary Strategy to Enhance an RNA Design Tool Performance," *IEEE Access*, vol. 12, pp. 15582–15593, 2024.

[7] N. Pardi, M. J. Hogan, F. W. Porter, and D. Weissman, "mRNA vaccines — a new era in vaccinology," *Nat. Rev. Drug Discov.*, vol. 17, no. 4, pp. 261–279, 2018.

[8] H. Zhang, L. Zhang, A. Lin, C. Xu, Z. Li, K. Liu, B. Liu, X. Ma, F. Zhao, H. Jiang, C. Chen, H. Shen, H. Li, D. H. Mathews, Y. Zhang, and L. Huang, "Algorithm for optimized mRNA design improves stability and immunogenicity," *Nature*, vol. 621, no. 7978, pp. 396–403, 2023.

[9] R. R. Breaker, "Engineered allosteric ribozymes as biosensor components," *Curr. Opin. Biotechnol.*, vol. 13, no. 1, pp. 31–39, 2002.

[10] A. Serganov and D. J. Patel, "Ribozymes, riboswitches and beyond: regulation of gene expression without proteins," *Nat. Rev. Genet.*, vol. 8, no. 10, pp. 776–790, 2007.

[11] C. B. Do, D. A. Woods, and S. Batzoglou, "CONTRAfold: RNA secondary structure prediction without physics-based models," *Bioinformatics*, vol. 22, no. 14, pp. e90–e98, 2006.

[12] K. Sato, Y. Kato, M. Hamada, T. Akutsu, and K. Asai, "IPknot: fast and accurate prediction of RNA secondary structures with pseudoknots using integer programming," *Bioinformatics*, vol. 27, no. 13, pp. i85–i93, 2011.

[13] S. Bellaousov and D. H. Mathews, "ProbKnot: Fast prediction of RNA secondary structure including pseudoknots," *RNA*, vol. 16, no. 10, pp. 1870–1880, 2010.

[14] K. Sato, M. Akiyama, and Y. Sakakibara, "RNA secondary structure prediction using deep learning with thermodynamic integration," *Nat. Commun.*, vol. 12, no. 1, p. 941, 2021.

[15] S. Bellaousov, J. S. Reuter, M. G. Seetin, and D. H. Mathews, "RNAstructure: web servers for RNA secondary structure prediction and analysis," *Nucleic Acids Res.*, vol. 41, no. W1, pp. W471–W474, 2013.

[16] L. Fu, Y. Cao, J. Wu, Q. Peng, Q. Nie, and X. Xie, "UFold: fast and accurate RNA secondary structure prediction with deep learning," *Nucleic Acids Res.*, vol. 50, no. 3, p. e14, 2022.

[17] H. K. Wayment-Steele, W. Kladwang, A. I. Strom, J. Lee, A. Treuille, A. Becka, Eterna Participants, and R. Das, "RNA secondary structure packages evaluated and improved by high-throughput experiments," *Nat. Methods*, vol. 19, no. 10, pp. 1234–1242, 2022.

[18] R. V. Koodli, B. Rudolfs, J. Romano, H. K. Wayment-Steele, W. A. Dunlap IV, Eterna Participants, and R. Das, "Redesigning the Eterna100 for the Vienna 2 folding engine," *bioRxiv*, 2025, doi: 10.1101/2021.08.26.457839.

[19] A. E. Eiben and J. E. Smith, *Introduction to Evolutionary Computing*, 2nd ed. Berlin, Heidelberg: Springer-Verlag, 2015.

[20] A. Taneda, "MODENA: A multi-objective RNA inverse folding," *Adv. Bioinformatics*, vol. 2011, Art. no. 658979, 2011.