Syntax-Aware Retrieval Augmentation for Neural Symbolic Regression

Canmiao Zhou[1], Han Huang[2,3]*

[1]School of Software Engineering, South China University of Technology,
[2]Guangzhou Zhisuan Linghang Technology Co., Ltd.,
[3]Key Laboratory of Symbolic Computation and Knowledge Engineering of Ministry of Education, Jilin University

canmiaozhou@163.com hhan@scut.edu.cn

Abstract

Symbolic regression is a powerful technique for discovering mathematical expressions that best fit observed data. While neural symbolic regression methods based on large-scale pretrained models perform well on simple tasks, the reliance on fixed parametric knowledge typically limits their generalization to complex and diverse data distributions. To address this challenge, we propose a syntax-aware retrieval-augmented mechanism that leverages the syntactic structure of symbolic expressions to perform context-aware retrieval from a preconstructed token datastore during inference. This mechanism enables the model to incorporate highly relevant non-parametric prior information to assist in expression generation. Additionally, we design an entropy-based confidence network that dynamically adjusts the fusion strength between neural and retrieved components by estimating predictive uncertainty. Extensive experiments on multiple symbolic regression benchmarks demonstrate that the proposed method significantly outperforms representative baselines, validating the effectiveness of retrieval augmentation in enhancing the generalization performance of neural symbolic regression models.

1 Introduction

Symbolic regression is a fundamental subfield of interpretable machine learning that aims to discover explicit analytical expressions from data, thereby revealing the underlying physical principles or functional relationships. (Makke and Chawla, 2024). By capturing latent structures and intrinsic dependencies within the data, it has demonstrated significant potential in fields such as scientific discovery (Kim et al., 2021) and engineering modeling (Rogers et al., 2024).

Conventional symbolic regression techniques predominantly rely on genetic programming (Zhong et al., 2020; Burlacu et al., 2023), employing evolutionary strategies to explore the combinatorial solution space of symbolic expressions. While these approaches preserve interpretability by enabling intuitive refinement of expressions, their exhaustive search nature incurs substantial computational overhead. More recently, rapid advancements in deep learning have driven the development of numerous neural approaches to symbolic regression. Among these, a particularly promising paradigm employs pretrained language models, such as Transformers, to formulate the task of generating symbolic expressions from data as a sequence-to-sequence translation task (Kamienny et al., 2022; Valipour et al., 2021; Biggio et al., 2021; Li et al., 2025). This neural paradigm enables the end-to-end generation of symbolic expressions within a few inference steps, thereby significantly improving the efficiency of symbolic regression.

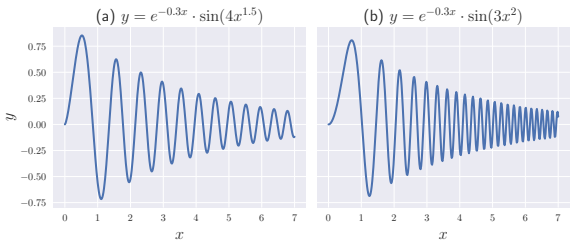


Figure 1: An illustration of two distinct damped oscillation functions exhibiting analogous oscillatory dynamics with differing damping coefficients. The structural similarity in their exponential decay profiles and phase-space trajectories enables cross-prediction of dynamical parameters through prior knowledge transfer.

Overall, most advances in neural symbolic regression have focused on improving model architectures and training paradigms to improve the mapping from data to symbolic expressions. Nevertheless, these approaches typically rely solely on the parametric knowledge encoded within the model parameters after training, neglecting the effective

*Corresponding author.

Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, pages 13149–13159
November 4-9, 2025

utilization of the rich information embedded in existing instances of similar symbolic expressions. Concretely, symbolic expressions frequently exhibit similar mathematical or physical properties due to recurring patterns of operators or structurally analogous formulations. For example, as illustrated in Fig. 1, two distinct damped oscillation functions display comparable structural characteristics and exhibit analogous oscillatory behavior. In such cases, one expression can serve as a valuable reference when inferring the other. This observation highlights the potential of leveraging structurally or functionally similar prior expressions to inform the generation of new ones, thus offering a promising direction for improving the performance of neural symbolic regression models.

To this end, we propose a syntax-aware retrieval augmentation mechanism (SRASR) to facilitate the effective utilization of knowledge embedded in structurally similar symbolic expression instances during expression generation. Specifically, we formulate symbolic regression as an autoregressive task, wherein prefix traversal sequences of expression trees are generated token by token. Within this framework, a pretrained symbolic regression model encodes partially generated prefix sequences into latent representations, which are utilized to perform K-nearest neighbor (KNN) (Kramer and Kramer, 2013) retrieval in a syntax-conditioned latent space. The symbolic tokens associated with the retrieved neighbors are subsequently aggregated to form a non-parametric probability distribution, serving as an auxiliary prior to guide the prediction of subsequent tokens. Moreover, to address the potential reliability gap between the neural predictive distribution and the KNN-derived prior, we introduce an entropy-based adaptive interpolation strategy that dynamically balances their contributions based on predictive uncertainty, thereby facilitating a more context-sensitive integration of parametric and non-parametric knowledge. We validate the effectiveness of the proposed method through extensive experiments on multiple classical symbolic regression benchmarks. The results consistently demonstrate that SRASR significantly outperforms existing baselines in generating symbolic expressions with superior fitting accuracy.

2 Related Work

Symbolic Regression Existing symbolic regression methods can be broadly categorized into two paradigms: search-based and supervised learning-based approaches. Genetic programming-based symbolic regression (GPSR) exemplifies the former, employing evolutionary algorithms to iteratively optimize expression structure and parameters for optimal data fitting (Burlacu et al., 2023). However, due to the NP-hard nature of symbolic regression (Virgolin and Pissis, 2022), such methods typically suffer from combinatorial explosion and high complexity (Petersen et al., 2021). To address these challenges, recent studies (Petersen et al., 2021; Liu et al., 2023a) have integrated reinforcement learning (RL) to improve search efficiency. For example, DSR (Petersen et al., 2021) employs a recurrent neural network trained with a risk-seeking policy gradient to model probabilistic distributions over expression skeletons. Despite such advancements, search-based approaches typically require task-specific optimization or retraining, which hinders their ability to generalize across tasks and reuse learned knowledge.

To address the limitations of search-based methods, recent works have increasingly adopted supervised learning approaches that leverage deep neural networks to directly model the complex mapping from data to symbolic expressions using large-scale datasets, thereby eliminating the need for exhaustive task-specific search. Representative models such as SymbolicGPT (Valipour et al., 2021), NeSymReS (Biggio et al., 2021), End2End (Kamienny et al., 2022), and MMSR (Li et al., 2025) adopt Transformer-based architectures with varying design and training strategies. SymbolicGPT utilizes a modified T-Net for enhanced feature extraction, while NeSymReS employs a Set-Transformer to ensure permutation invariance in encoding unordered data points. End2End applies a standard Transformer encoder-decoder for end-to-end expression generation, and MMSR incorporates multimodal learning to exploit the intrinsic multimodal nature of symbolic regression. While these models advance architecture and training, they typically do not exploit prior symbolic expressions as guidance, limiting their capacity to incorporate valuable prior knowledge into prediction.

Nearest Neighbor Retrieval Methods in NLP Nearest neighbor retrieval methods have achieved significant success in various natural language processing (NLP) tasks, including machine translation (Liu et al., 2023b; Nishida

et al., 2024) and language modeling (Shi et al., 2022; Trotta et al., 2022). These methods typically construct a datastore by associating decoder output vectors from a pretrained language model with their corresponding target tokens. During inference, kNN retrieval is performed to identify contextually relevant entries from the constructed datastore to form a reference distribution, which is then integrated with the model's predictions to enhance accuracy.

However, prior studies have indicated that direct application of kNN retrieval tends to introduce noise into prediction (Jiang et al., 2022; Yuan et al., 2024; Lv et al., 2024). This issue is especially critical in symbolic regression, which involves symbolic expressions that are highly sensitive to syntactic correctness. To address this, we propose constructing separate kNN datastores based on token categories within symbolic expressions, thereby preventing the disruption of syntactic structures caused by indiscriminate retrieval. Additionally, we introduce an entropy-based confidence adjustment mechanism to mitigate noise in the kNN-derived distributions, enhancing the robustness of retrieval-augmented predictions.

3 Retrieval-Augmented Symbolic Regression

In this section, we introduce the proposed SRASR, which is designed to integrate syntax-constrained nearest neighbor retrieval into the expression generation process of pretrained models, thereby enabling effective utilization of prior knowledge embedded in existing expression instances to enhance prediction accuracy. As depicted in Fig. 2, the framework consists of two essential components: syntax-aware datastore construction and retrieval-augmented prediction, which are detailed respectively in the subsequent subsections.

3.1 Syntax-Aware Datastore Construction

In mathematical expressions, operator tokens typically occur with higher frequency, while constant tokens appear exclusively at the leaf nodes of expression trees, resulting in relatively lower frequencies, as intuitively illustrated by the expression tree in Fig. 2. However, the vocabulary space of constant tokens is substantially larger and sparser than that of operators, a disparity also reflected in the statistics reported in Table 1. Consequently, storing both types of tokens within a unified retrieval datastore can lead to several issues: on one hand, constant tokens are less likely to retrieve semantically relevant neighbors effectively; on the other hand, the sparsity of constant tokens may introduce noise during the retrieval of operator tokens, thereby causing retrieval biases that compromise the syntactic validity of the generated expressions.

Token Type	Vocabulary Size	Datastore Samples
Operator	55	9,089,028
Constant	10,200	5,055,706

Table 1: Statistics of vocabulary size and datastore samples for operator and constant tokens.

Therefore, motivated by the distinct syntactic roles of tokens within symbolic expressions, we propose constructing separate retrieval datastores for operator and constant tokens. Operator tokens delineate the structural skeleton of the expression tree, whereas constant tokens govern its numerical evaluation. This syntactic separation in datastore design mitigates cross-category interference in the retrieval space, thereby improving the semantic relevance of retrieved neighbors and preserving the structural validity of generated expressions.

Specifically, for each data point-expression pair $(\mathbf{p}, \mathbf{s})$ from the training set $(\mathcal{P}, \mathcal{S})$, where $\mathbf{p} = \{(x_l, y_l)\}_{l=1}^{n}$ represents a set of numerical observations and $\mathbf{s} = \{s_1, s_2, ..., s_m\}$ denotes the prefix traversal sequence of the corresponding symbolic expression tree, the input $\mathbf{p}$ is initially encoded by a pretrained symbolic regression model $f_\theta(\cdot)$ to obtain a contextual representation for conditional generation. During the autoregressive decoding process, the decoder hidden state vector $h_i = f_\theta(\mathbf{p}, \mathbf{s}_{<i})$ and the associated target token s_i are recorded at each step i. Subsequently, each pair (h_i, s_i) is stored in the appropriate datastore, based on the syntactic category of s_i, to facilitate syntax-aware neighbor retrieval during inference. The construction of the operator datastore D_{optor} and the constant datastore D_{const} can be mathematically formulated as follows:

$$D_{\text{optor}} = \bigcup_{(\mathbf{p},\mathbf{s})\in(\mathcal{P},\mathcal{S})} \{(h_i, s_i) \mid \forall s_i \in (\mathbf{s} \cap \mathbf{o})\} \tag{1}$$

$$D_{\text{const}} = \bigcup_{(\mathbf{p},\mathbf{s})\in(\mathcal{P},\mathcal{S})} \{(h_i, s_i) \mid \forall s_i \in (\mathbf{s} \cap \mathbf{c})\} \tag{2}$$

where $\mathbf{o}$ and $\mathbf{c}$ denote the sets of operator tokens and constant tokens, respectively. Given that vari-

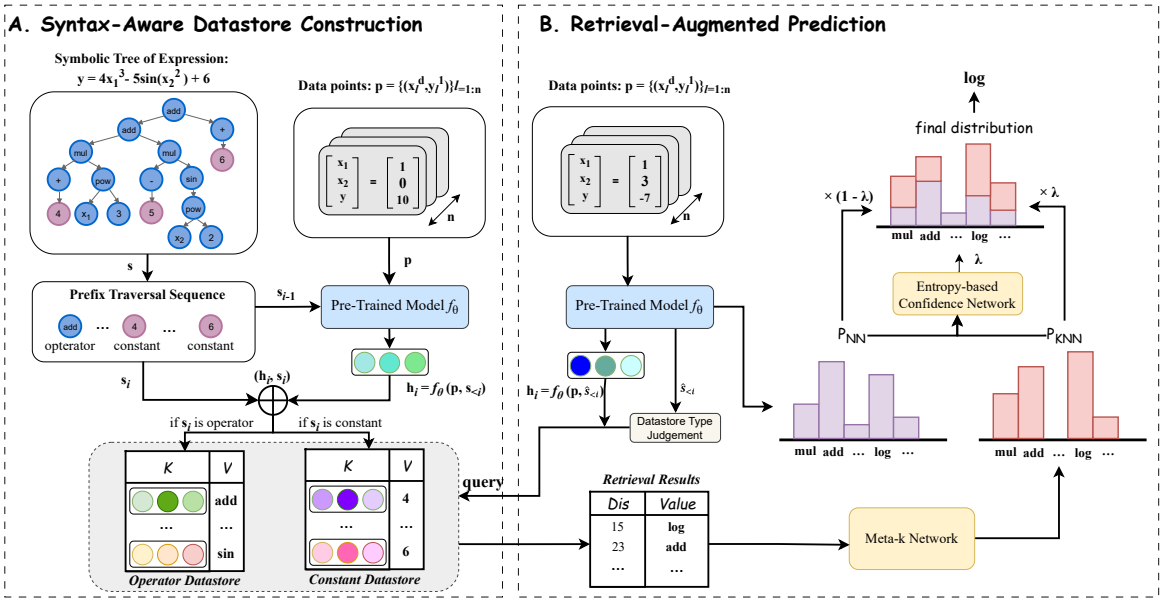


Figure 2: Workflow of the proposed syntax-aware retrieval-augmented symbolic regression method, including syntax-aware datastore construction and retrieval-augmented prediction processes.

ables (e.g., 'x_1', 'x_2', ...) and the sign tokens of constants (i.e., '+' and '−') possess a limited type space and play a structural role similar to operators in symbolic expressions, they are categorized as operator-type tokens and stored in the operator-specific datastore accordingly.

3.2 Retrieval-Augmented Prediction

As illustrated in Fig. 2, the retrieval-augmented prediction mechanism enhances the autoregressive decoding process of the pretrained symbolic regression model by retrieving contextually relevant neighbors from syntax-aware datastores constructed in subsection 3.1.

Specifically, at the i-th decoding step during inference with the pretrained model, the type of the target token is inferred based on the previously generated prefix of the expression. If the preceding token $\hat{s}_{i-1}$ is sign '+' or '−', the subsequent token is assumed to be a constant, and the constant-specific datastore D_{const} is selected for retrieval. Otherwise, the operator-specific datastore D_{optor} is chosen. Subsequently, k nearest neighbors are retrieved from the selected datastore based on the Euclidean distance between the current decoder output representation h_i and the stored keys, yielding a set of distance-token pairs $\mathcal{N} = \{(d_j, v_j)\}_{j=1}^{k}$, where each d_j denotes the Euclidean distance between the query h_i and the retrieved key h_j. The resulting token distribution over the vocabulary is computed as follows:

$$P_{\text{KNN}_k}(s_i|\mathbf{p}, \hat{\mathbf{s}}_{<i}) \propto \sum_{(d_j,v_j)\in\mathcal{N}} \mathbb{1}_{s_i=v_j} \exp(\frac{-d_j}{\tau}) \tag{3}$$

where $\hat{s}$ denotes the predicted prefix sequence of the expression tree, τ is the temperature parameter for scaling the distance values.

The choice of the hyperparameter k critically affects the constructed KNN probability distribution: an excessively large k may introduce noise from irrelevant neighbors, thereby impairing the model's prediction, while an overly small k risks overlooking semantically important tokens. This trade-off is particularly pronounced in the operator datastore due to its inherently limited vocabulary space (fewer than 90 tokens), which makes the selection of an appropriate k especially crucial. To address this, inspired by prior work on learnable retrieval configuration, we employ a meta-k network (Zheng et al., 2021) to adaptively select the optimal k based on the current decoding context. This dynamic strategy effectively balances retrieval robustness with prediction accuracy, yielding an adaptive KNN distribution P_{KNN}.

Furthermore, to effectively leverage the retrieved results for refining the original distribution P_{NN} produced by the pretrained model, we perform an interpolation between P_{NN} and the KNN-based distribution P_{KNN} to obtain the final predictive dis-

tribution P, formulated as:

$$P = \lambda P_{\text{NN}} + (1 - \lambda) P_{\text{KNN}} \tag{4}$$

In this process, dynamically adjusting the interpolation weight λ is crucial for achieving an appropriate balance between the contributions of both distributions to the final prediction. To this end, we introduce an entropy-based confidence network that quantifies the uncertainty of each distribution through its entropy and leverages these entropy features to adaptively determine the fusion weight.

The following provides a detailed description of the meta-k network and the entropy-based confidence network.

Meta-k Network Given an upper bound K for nearest neighbor retrieval, the meta-k network adaptively infers the optimal retrieval size by leveraging context-aware features to assess the relative importance of candidate neighbor subsets. Specifically, for the top-K retrieved neighbors, two feature sequences are constructed to characterize the retrieval context. The distance sequence $\mathbf{d} = \{d_1, d_2, ..., d_K\}$ captures the local density of neighbors by measuring their distances to the query representation. The diversity sequence $\mathbf{c} = \{c_1, c_2, ..., c_K\}$ reflects semantic variation, with each c_j representing the number of distinct tokens among the top-j neighbors. These two sequences are concatenated and fed into a lightweight two-layer perceptron $f_{\text{meta}}(\cdot)$, which yields a confidence distribution over a predefined candidate set of retrieval sizes $\mathcal{S} = \{0, 1, 2, 4, \ldots, 2^{\lfloor log_2 K \rfloor}\}$, i.e., $p(k) = \text{softmax}(f_{\text{meta}}([\mathbf{d}; \mathbf{c}]))$. Subsequently, the final KNN predictive distribution is computed as a weighted sum over the candidate subset-specific distributions:

$$P_{\text{KNN}}(s_i|\mathbf{p}, \hat{\mathbf{s}}_{<i}) \propto \sum_{k_r \in \mathcal{S}} p(k_r) \cdot P_{\text{KNN}_{k_r}}(s_i|\mathbf{p}, \hat{\mathbf{s}}_{<i}) \tag{5}$$

This mechanism enables context-dependent selection of the most effective neighbor subset size, balancing local density and semantic diversity to optimize retrieval utility.

Entropy-based Confidence Network In symbolic regression, prediction targets are governed by strict syntactic rules and exhibit strong inter-symbol dependencies. Therefore, ensuring consistency and coordination between the model-generated distribution and the retrieval-augmented distribution is critical for robust inference. To this end, we design an entropy-based confidence network that adaptively estimates the interpolation weight λ based on both contextual information and distributional uncertainty.

Specifically, the confidence network takes as input a concatenation of context-aware retrieval features and entropy-based signals derived from the predictive distributions. The interpolation weight λ is computed as:

$$\lambda = \text{sigmoid}(f_{\text{conf}}([\mathbf{d}; \mathbf{c}; \mathbf{e}])) \tag{6}$$

where $\mathbf{d}$ and $\mathbf{c}$ represent the distance and diversity feature sequences of the retrieved neighbors, respectively, as defined in the meta-k network. $\mathbf{e}$ is an entropy-aware feature defined by:

$$\mathbf{e} = f_{\text{uncertainty}}(\text{JS}(P_{\text{NN}}, P_{\text{KNN}}), \mathcal{H}(P_{\text{KNN}})) \tag{7}$$

where $\text{JS}(P_{\text{NN}}, P_{\text{KNN}})$ denotes the Jensen-Shannon divergence between the neural predictive distribution P_{NN} and the KNN-derived distribution P_{KNN}, quantifying their semantic discrepancy; $\mathcal{H}(P_{\text{KNN}})$ represents the entropy of the KNN distribution, measuring its inherent uncertainty. The two uncertainty metrics are fed into $f_{\text{uncertainty}}$ to produce a unified entropy representation $\mathbf{e}$, which is concatenated with contextual features and mapped through f_{conf}, followed by a sigmoid activation to yield the final interpolation weight λ. Both $f_{\text{uncertainty}}$ and f_{conf} are lightweight two-layer feed-forward networks.

The proposed entropy-based network dynamically modulates the interpolation between neural generation and retrieval augmentation by leveraging both contextual information and distributional uncertainties, thereby enhancing the robustness of the final predictions.

4 Experiment

In this section, we conduct experiments on publicly available datasets and compare the proposed SRASR against various competitive baselines to evaluate its effectiveness in symbolic regression. Our source code is released at https://github.com/marasimc/SRASR.

4.1 Experimental Settings

Benchmarks. We conduct comparative analyses on several widely used symbolic regression datasets, including Nguyen (Uy et al., 2011), Keijzer (Keijzer, 2003), Livermore (Mundhenk

et al., 2021), R (Krawiec and Pawlak, 2013), Jin (Jin et al., 2019), Neat (Trujillo et al., 2016), and Strogatz. In particular, the Strogatz dataset is derived from SRBench (La Cava et al., 2021), a comprehensive benchmark suite for symbolic regression that encompasses diverse data distributions and a wide range of equation types.

Evaluation Metrics. We assess the performance of various symbolic regression models based on the fitting accuracy of the generated symbolic expressions. To this end, we employ the coefficient of determination (R^2) (Cava et al., 2021) as the evaluation metric, which is widely adopted in the symbolic regression literature and is defined as:

$$R^2 = 1 - \frac{\sum_i^{N_{\text{test}}} (y_i - \hat{y}_i)^2}{\sum_i^{N_{\text{test}}} (y_i - \bar{y})^2}$$

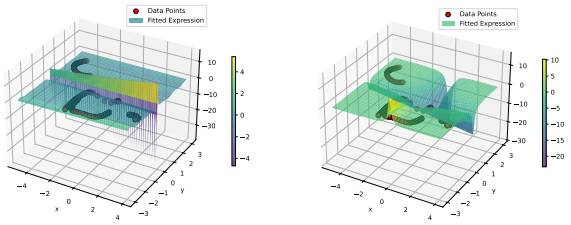
where N_{test} denotes the number of test samples, y_i and $\hat{y}_i$ represent the ground truth and predicted values of the i-th test sample, respectively, and $\bar{y}$ is the mean of the ground truth values over the test set.

Baselines. To evaluate the effectiveness of the proposed method, we conduct comparisons against several state-of-the-art baselines. These include NeSymReS (Biggio et al., 2021) and End2End (Kamienny et al., 2022), two representative Transformer-based neural symbolic regression models; DySymNet (Li et al., 2024), a recent approach leveraging dynamic symbolic networks; and GPlearn (Stephens, 2022), a classical genetic programming-based symbolic regression method. Additionally, we consider DSR (Petersen et al., 2021), a search-based approach that integrates recurrent neural networks with reinforcement learning strategies; uDSR (Landajuela et al., 2022), which unifies five symbolic regression strategies into a cohesive architecture to enhance overall performance; and RSRM (Xu et al., 2024), a reinforcement-learning-based approach that combines genetic programming and modulated subtree discovery to extract complex mathematical expressions from limited data. All baseline methods are publicly available, and we adopt their default hyperparameter configurations to ensure consistency and fairness in comparison.

Implementation Details. In our retrieval-augmented framework, we adopt End2End (Kamienny et al., 2022) as the pretrained backbone and construct the datastore following the dataset generation strategy of its original implementation. To enable efficient retrieval from the constructed datastore, we utilize Faiss (Johnson et al., 2021) for nearest-neighbor search. During retrieval-augmented inference, the maximum number of nearest neighbors is set to 4 for the operator datastore and 64 for the constant datastore. A detailed sensitivity analysis of these two hyperparameters is presented in Section 4.3. Both the meta-k network and the confidence network are implemented with a hidden layer size of 32. Model training is performed using the Adam optimizer with a learning rate of 10^{-5} and a batch size of 64.

4.2 Main Results

As shown in Table 2, we perform a comprehensive evaluation of the proposed SRASR on several widely used symbolic regression benchmark datasets. The results indicate that SRASR consistently outperforms both classical search-based symbolic regression approaches (including GPlearn, DSR, uDSR, and RSRM) and recent neural symbolic regression models (i.e., NeSymReS, End2End, and DySymNet) across all datasets. In particular, compared to the pretrained End2End model employed as our backbone, SRASR achieves robust and consistent performance improvements, yielding an average gain of 3.53% in the R^2 score (increasing from 0.9511 to 0.9847). This notable performance gain underscores the efficacy of the proposed syntax-aware retrieval augmentation mechanism in improving the accuracy of symbolic expression generation.



(a) Fitting result generated by SRASR

(b) Fitting result generated by End2End

Figure 3: Comparison of fitting results obtained by SRASR and End2End on the *strogatz_shearflow1* dataset. The underlying dynamics are described by the differential equation $x' = \cot(y) \cdot \cos(x)$, representing one state of a two-dimensional shear flow system. In both visualizations, red dots represent the sampled data points from the dataset, while the surface illustrates the symbolic expression inferred by the corresponding symbolic regression model.

Benchmark	SRASR (ours)	End2End	NeSymReS	DySymReS	GPlearn	DSR	uDSR	RSRM
Nguyen	**0.9992**$_{\pm 0.0000}$	0.9988$_{\pm 0.0000}$	0.9804$_{\pm 0.0000}$	0.9059$_{\pm 0.0074}$	0.9106$_{\pm 0.0000}$	0.9732$_{\pm 0.0018}$	0.9906$_{\pm 0.0000}$	0.9972$_{\pm 0.0009}$
Keijzer	**0.9937**$_{\pm 0.0000}$	0.9736$_{\pm 0.0000}$	0.9384$_{\pm 0.0000}$	0.9569$_{\pm 0.0501}$	0.6986$_{\pm 0.0520}$	0.8204$_{\pm 0.0020}$	0.9523$_{\pm 0.0000}$	0.9725$_{\pm 0.0040}$
Livermore	**0.9879**$_{\pm 0.0000}$	0.9875$_{\pm 0.0000}$	0.9642$_{\pm 0.0000}$	0.8990$_{\pm 0.0113}$	0.7515$_{\pm 0.0256}$	0.8006$_{\pm 0.0105}$	0.8993$_{\pm 0.0000}$	0.9821$_{\pm 0.0004}$
R	**0.9999**$_{\pm 0.0000}$	0.9998$_{\pm 0.0000}$	0.9765$_{\pm 0.0000}$	0.8654$_{\pm 0.1335}$	0.6113$_{\pm 0.0099}$	0.9139$_{\pm 0.0044}$	0.9811$_{\pm 0.0000}$	0.9976$_{\pm 0.0023}$
Jin	**0.9999**$_{\pm 0.0000}$	0.9625$_{\pm 0.0000}$	0.9396$_{\pm 0.0000}$	0.9508$_{\pm 0.0308}$	0.8578$_{\pm 0.0291}$	0.8002$_{\pm 0.0000}$	0.9276$_{\pm 0.0000}$	0.9550$_{\pm 0.0035}$
Neat	**0.9674**$_{\pm 0.0000}$	0.9163$_{\pm 0.0000}$	0.9121$_{\pm 0.0000}$	0.9419$_{\pm 0.0397}$	0.7637$_{\pm 0.0897}$	0.8162$_{\pm 0.0022}$	0.8755$_{\pm 0.0000}$	0.9205$_{\pm 0.0007}$
Strogatz	**0.9449**$_{\pm 0.0000}$	0.8192$_{\pm 0.0000}$	0.8069$_{\pm 0.0000}$	0.8642$_{\pm 0.0568}$	0.4768$_{\pm 0.0000}$	0.6704$_{\pm 0.0070}$	0.8780$_{\pm 0.0000}$	0.9191$_{\pm 0.0001}$
Average	**0.9847**	0.9511	0.9312	0.9120	0.7243	0.8278	0.9292	0.9634

Table 2: Comparison of performance between SRASR and baseline methods in terms of the coefficient of determination (R^2) (↑). For each method, the reported R^2 values represent the mean over ten independent runs, with 95% confidence intervals provided.

Notably, SRASR demonstrates a substantial performance improvement over End2End on the Strogatz benchmark, with an increase of 0.1257 in the R^2 metric. We attribute this improvement to SRASR's enhanced capability in modeling nonlinear dynamics and chaotic behavior. Specifically, the Strogatz dataset comprises a collection of challenging tasks involving complex nonlinear dynamic behaviors and chaotic regimes, which are characterized by intricate symbolic dependencies and highly uncertain functional landscapes. As a representative example, we examine a two-dimensional shear flow system governed by the differential equation $x' = \cot(y) \cdot \cos(x)$. Fig. 3 presents a comparison of the fitting surfaces generated by SRASR and End2End on this specific task. The visualization clearly reveals that SRASR generates symbolic expressions that closely fit the observed data, whereas the expression produced by End2End exhibits noticeable deviations. This observation supports our claim that the proposed syntax-aware retrieval mechanism enables SRASR to more effectively capture complex symbolic dependencies, thereby yielding superior performance in modeling tasks characterized by structural complexity and dynamical uncertainty.

4.3 Hyperparameter Analysis

In this experiment, we investigate the effect of key retrieval hyperparameters on the performance of SRASR, specifically the maximum number of neighbors for the operator and constant datastores, denoted as k_1 and k_2, respectively. As shown in Table 3, given the relatively small number of operator token types, we consider smaller values of k_1, whereas, due to the larger number of constant token types, we evaluate larger values of k_2.

We observe that increasing k_1 from 4 to 8 leads to improved performance on certain datasets when k_2 is set to 16 or 32; however, the same increase in k_1 results in a performance decline when k_2 is set to 64. Overall, the best performance is achieved with $k_1 = 4$ and $k_2 = 64$, which is consistent with our expectations. Theoretically, each datastore has an optimal retrieval threshold: values set too low may omit informative tokens, whereas values set too high may introduce excessive noise. In practice, however, since operator and constant symbols jointly determine both the structural and numerical aspects of the generated expressions, their thresholds are inherently interdependent. Consequently, an imbalanced threshold in one datastore can not only degrade its own retrieval quality but also weaken the contribution of the other. This interdependence underscores the importance of carefully tuning both thresholds in conjunction to achieve stable performance.

Benchmark	k_1=4			k_1=8		
	k_2=16	k_2=32	k_2=64	k_2=16	k_2=32	k_2=64
Nguyen	0.9984	0.9989	**0.9992**	0.9989	0.9987	<u>0.9991</u>
Keijzer	<u>0.9917</u>	0.9805	**0.9937**	0.9897	0.9897	0.9902
Livermore	0.9761	<u>0.9886</u>	0.9879	**0.9921**	0.9869	0.9816
R	0.9996	0.9997	**0.9999**	**0.9999**	<u>0.9998</u>	0.9993
Jin	0.9626	**0.9999**	**0.9999**	<u>0.9979</u>	0.9913	0.9670
Neat	<u>0.9572</u>	0.9487	**0.9674**	0.9236	0.9540	0.9478
Strogatz	0.8293	0.8623	**0.9449**	0.8524	<u>0.8949</u>	0.8640
Average	0.9593	0.9684	**0.9847**	0.9649	<u>0.9736</u>	0.9641

Table 3: Results of the retrieval hyperparameter study. The best and second-best results are highlighted in bold and underlined, respectively.

Our empirical results indicate that the configuration $k_1 = 4, k_2 = 64$ achieves a favorable balance between recall and noise, yielding superior performance in complex modeling tasks. This setting is also consistent with the token distribution, as operator tokens are relatively sparse and thus benefit from a smaller threshold, whereas constant tokens are more abundant and require a larger threshold to ensure sufficient coverage. Accordingly, we adopt $k_1 = 4, k_2 = 64$ as the default hyperparameter

setting in our experiments.

4.4 Ablation Study

In this section, we perform ablation experiments to assess the effect of several core components of SRASR, including the syntax-aware retrieval module, the meta-k network, and the entropy-based confidence network. Considering the significant performance differences observed between SRASR and End2End on the Jin, Neat, and Strogatz datasets in the previous experiment, we select them for subsequent ablation studies.

Effect of the syntax-aware retrieval mechanism. To evaluate the impact of the syntax-aware retrieval mechanism in SRASR, we compare two retrieval-augmented strategies under identical model settings. In the syntax-aware setting, separate datastores are maintained for operators and constants, from which 4 and 64 nearest neighbors are retrieved respectively, consistent with the main experimental setup. In contrast, the syntax-agnostic variant employs a unified datastore that merges operator and constant tokens, retrieving 34 neighbors to match the average total used in the syntax-aware setting. In both cases, the interpolation weight λ between neural and retrieval probabilities is fixed at 0.5.

As illustrated in Fig. 4(a), both retrieval strategies improve the performance of the pretrained symbolic regression model (End2End) across multiple benchmarks, demonstrating the overall effectiveness of retrieval-based augmentation. Notably, the syntax-aware retrieval approach achieves consistently superior improvements compared to its syntax-agnostic counterpart. This performance gap underscores the advantage of incorporating syntactic structure during retrieval, which enables the model to retrieve more semantically relevant and structurally consistent neighbors. By aligning retrieved examples with the grammatical roles of target tokens (e.g., operators vs. constants), the syntax-aware strategy provides more contextually appropriate guidance during expression generation.

Effect of the meta-k network. As described in section 3.2, the meta-k network adaptively selects the optimal number of neighbors based on task-specific semantic features, with the goal of improving the flexibility and accuracy of retrieval-augmented generation. To evaluate its effectiveness, we compare two SRASR variants without the confidence network and under a fixed interpolation weight of 0.5: one without the meta-k network and the other with it. The comparison results are shown in Fig. 4(b).

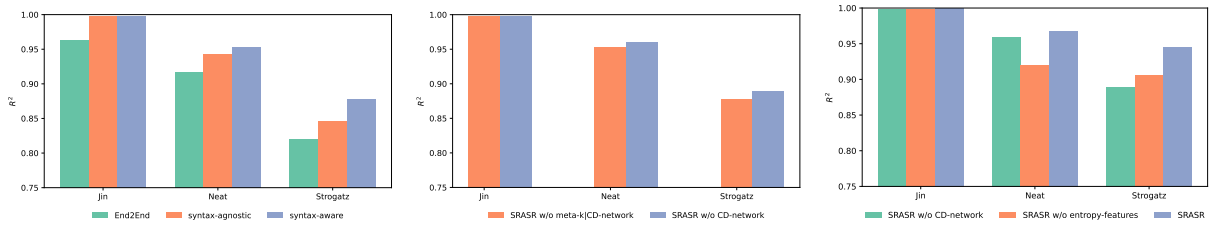
The results indicate that incorporating the meta-k network leads to modest but consistent improvements on the Neat and Strogatz benchmarks. Performance on the Jin dataset shows minimal improvement, as the fixed retrieval configuration already provides strong performance. These results demonstrate that the meta-k network plays a beneficial role in adaptively adjusting the retrieval strength, offering stable performance gains across tasks. While the improvements are not dramatic, they highlight the value of incorporating retrieval adaptivity in symbolic expression generation.

Benchmark	End2End	w/o JS&$\mathcal{H}$	w/ $\mathcal{H}$	w/ JS	w/ JS&$\mathcal{H}$
Nguyen	0.9988	0.9981	0.9989	0.9989	0.9992
Keijzer	0.9736	0.9932	0.9918	0.9957	0.9937
Livermore	0.9875	0.9872	0.9826	0.9828	0.9879
R	0.9998	0.9996	0.9998	0.9998	0.9999
Jin	0.9625	0.9984	0.9919	0.9919	0.9999
Neat	0.9163	0.9247	0.9654	0.9656	0.9674
Strogatz	0.8192	0.8652	0.8891	0.8890	0.9449
Average	0.9511	0.9666	0.9742	0.9748	0.9847

Table 4: Ablation study on the effect of different entropy-based features on SRASR performance.

Effect of the entropy-based confidence network. To assess the effect of the confidence network, particularly the contribution of entropy-based features within SRASR, we compare three variants: (1) SRASR without the confidence network, employing a fixed interpolation weight $\lambda = 0.5$; (2) SRASR with a confidence network utilizing only contextual features to learn the interpolation weight; and (3) SRASR with the full entropy-based confidence network. The comparative results are presented in Fig. 4(c).

On the Strogatz dataset, the confidence network relying solely on contextual features achieves a modest improvement over the fixed-weight baseline. However, on the Neat dataset, this approach leads to performance degradation, indicating that context-only confidence estimation lacks robustness across different tasks. In contrast, the integration of entropy-based features consistently improves performance across all evaluated datasets, demonstrating their stabilizing effect. These findings highlight the critical role of the entropy features in confidence estimation, as they effectively enhance the network's capacity for accurate confi-



(a) Ablation study results of the syntax-aware retrieval mechanism

(b) Ablation study results of the meta-k network

(c) Ablation study results of the entropy-based confidence network

Figure 4: Ablation study of SRASR on Jin, Neat, and Strogatz benchmarks.

dence assessment, thereby enabling more effective control over retrieval fusion and ultimately improving the robustness of symbolic expression generation. Overall, the experiment confirms the value of the entropy-based confidence network in improving the reliability and performance of SRASR.

Furthermore, we investigate the influence of two distinct entropy-based features on the performance of SRASR. Specifically, the feature JS denotes the Jensen-Shannon divergence between the neural prediction distribution P_{NN} and the k-nearest neighbor distribution P_{KNN}, whereas the feature $\mathcal{H}$ corresponds to the entropy of the distribution P_{KNN}. As shown in Table 4, each feature individually contributes to improving model performance, while their combination yields further gains. This indicates that the two features capture complementary aspects of predictive uncertainty, with the JS divergence reflecting distributional discrepancies between neural and retrieved predictions, and the entropy characterizing the inherent uncertainty within the retrieved neighborhood. Their integration thus enables a more robust and fine-grained estimation of model confidence.

5 Conclusion

In this work, we propose a syntax-aware retrieval-augmented mechanism for symbolic regression. Our approach leverages the syntactic structure of symbolic expressions to enable adaptive retrieval of pre-stored tokens, thereby incorporating non-parametric prior knowledge from retrieval results to enhance the expression generation process of neural networks. Moreover, we design an entropy-based confidence network that dynamically adjusts the retrieval interpolation weight during inference, enhancing uncertainty estimation and model robustness. Extensive experiments demonstrate that SRASR significantly outperforms existing state-of-the-art methods across multiple challenging benchmarks, particularly excelling at modeling highly nonlinear and chaotic dynamic systems. These results highlight the superiority of our approach in enhancing neural symbolic regression performance on complex tasks. Future work will focus on developing more efficient data storage and retrieval architectures to improve computational efficiency. Additionally, we plan to extend our framework to large-scale scientific data modeling, further broadening the applicability of symbolic regression to complex real-world problems.

Limitations

Despite the demonstrated effectiveness of SRASR in enhancing symbolic regression via syntax-aware retrieval, the retrieval process inevitably introduces additional computational overhead during inference, resulting in a modest increase in decoding time compared to standard pretrained models. Detailed experimental results on solving time are provided in Appendix A. One potential approach to mitigate this limitation is to develop more efficient data storage and retrieval structures to reduce retrieval latency.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (62276103), the Innovation Team Project of General Colleges and Universities in Guangdong Province (2023KCXTD002), the Research and Development Project on Key Technologies for Intelligent Sensing and Analysis of Urban Events Based on Low-Altitude Drones (2024BQ010011), the 2023 Special Program for Audit Theory Research, Guangdong Provincial Philosophy and Social Science Planning (GD23SJZ09), and the Fundamental Research Funds for the Central Universities, JLU (93K172024K24).

References

Luca Biggio, Tommaso Bendinelli, Alexander Neitz, Aurelien Lucchi, and Giambattista Parascandolo. 2021. Neural symbolic regression that scales. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 936–945. PMLR.

Bogdan Burlacu, Kaifeng Yang, and Michael Affenzeller. 2023. Population diversity and inheritance in genetic programming for symbolic regression. *Natural Computing*, pages 1–36.

William La Cava, Patryk Orzechowski, Bogdan Burlacu, Fabricio Olivetti de Franca, Marco Virgolin, Ying Jin, Michael Kommenda, and Jason H. Moore. 2021. Contemporary symbolic regression methods and their relative performance. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*.

Hui Jiang, Ziyao Lu, Fandong Meng, Chulun Zhou, Jie Zhou, Degen Huang, and Jinsong Su. 2022. Towards robust k-nearest-neighbor machine translation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 5468–5477, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Ying Jin, Weilin Fu, Jian Kang, Jiadong Guo, and Jian Guo. 2019. Bayesian symbolic regression. *arXiv:1910.08892*.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2021. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data*, 7(3):535–547.

Pierre-alexandre Kamienny, Stéphane d'Ascoli, Guillaume Lample, and Francois Charton. 2022. End-to-end symbolic regression with transformers. In *Advances in Neural Information Processing Systems*, volume 35, pages 10269–10281. Curran Associates, Inc.

Maarten Keijzer. 2003. Improving symbolic regression with interval arithmetic and linear scaling. In *Proceedings of the 6th European Conference on Genetic Programming*, EuroGP'03, page 70–82, Berlin, Heidelberg. Springer-Verlag.

Samuel Kim, Peter Y. Lu, Srijon Mukherjee, Michael Gilbert, Li Jing, Vladimir Čeperić, and Marin Soljačić. 2021. Integration of neural network-based symbolic regression in deep learning for scientific discovery. *IEEE Transactions on Neural Networks and Learning Systems*, 32(9):4166–4177.

Oliver Kramer and Oliver Kramer. 2013. K-nearest neighbors. *Dimensionality reduction with unsupervised nearest neighbors*, pages 13–23.

Krzysztof Krawiec and Tomasz Pawlak. 2013. Approximating geometric crossover by semantic backpropagation. In *Proceedings of the 15th Annual Conference on Genetic and Evolutionary Computation*, GECCO '13, page 941–948, New York, NY, USA. Association for Computing Machinery.

William La Cava, Bogdan Burlacu, Marco Virgolin, Michael Kommenda, Patryk Orzechowski, Fabrício Olivetti de França, Ying Jin, and Jason H Moore. 2021. Contemporary symbolic regression methods and their relative performance. *Advances in neural information processing systems*, 2021(DB1):1.

Mikel Landajuela, Chak Shing Lee, Jiachen Yang, Ruben Glatt, Claudio P Santiago, Ignacio Aravena, Terrell Mundhenk, Garrett Mulcahy, and Brenden K Petersen. 2022. A unified framework for deep symbolic regression. *Advances in Neural Information Processing Systems*, 35:33985–33998.

Wenqiang Li, Weijun Li, Lina Yu, Min Wu, Linjun Sun, Jingyi Liu, Yanjie Li, Shu Wei, Deng Yusong, and Meilan Hao. 2024. A neural-guided dynamic symbolic network for exploring mathematical expressions from data. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 28222–28242. PMLR.

Yanjie Li, Jingyi Liu, Min Wu, Lina Yu, Weijun Li, Xin Ning, Wenqiang Li, Meilan Hao, Yusong Deng, and Shu Wei. 2025. Mmsr: Symbolic regression is a multi-modal information fusion task. *Information Fusion*, 114:102681.

Jingyi Liu, Weijun Li, Lina Yu, Min Wu, Linjun Sun, Wenqiang Li, and Yanjie Li. 2023a. Snr: Symbolic network-based rectifiable learning framework for symbolic regression. *Neural Netw.*, 165(C):1021–1034.

Shudong Liu, Xuebo Liu, Derek F. Wong, Zhaocong Li, Wenxiang Jiao, Lidia S. Chao, and Min Zhang. 2023b. kNN-TL: k-nearest-neighbor transfer learning for low-resource neural machine translation. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1878–1891, Toronto, Canada. Association for Computational Linguistics.

Rui Lv, Junliang Guo, Rui Wang, Xu Tan, Tao Qin, and Qi Liu. 2024. N-gram nearest neighbor machine translation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*.

Nour Makke and Sanjay Chawla. 2024. Interpretable scientific discovery with symbolic regression: a review. *Artificial Intelligence Review*, 57(1):2.

Terrell Mundhenk, Mikel Landajuela, Ruben Glatt, Claudio P Santiago, Daniel faissol, and Brenden K Petersen. 2021. Symbolic regression via deep reinforcement learning enhanced genetic programming seeding. In *Advances in Neural Information Processing Systems*, volume 34, pages 24912–24923. Curran Associates, Inc.

Yuto Nishida, Makoto Morishita, Hidetaka Kamigaito, and Taro Watanabe. 2024. Generating diverse translation with perturbed kNN-MT. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics: Student Research Workshop*, pages 9–31, St. Julian's, Malta. Association for Computational Linguistics.

Brenden K Petersen, Mikel Landajuela Larma, Terrell N. Mundhenk, Claudio Prata Santiago, Soo Kyung Kim, and Joanne Taery Kim. 2021. Deep symbolic regression: Recovering mathematical expressions from data via risk-seeking policy gradients. In *International Conference on Learning Representations*.

Alexander W Rogers, Amanda Lane, Cesar Mendoza, Simon Watson, Adam Kowalski, Philip Martin, and Dongda Zhang. 2024. Integrating knowledge-guided symbolic regression and model-based design of experiments to automate process flow diagram development. *Chemical Engineering Science*, 300:120580.

Weijia Shi, Julian Michael, Suchin Gururangan, and Luke Zettlemoyer. 2022. Nearest neighbor zero-shot inference. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 3254–3265, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Trevor Stephens. 2022. gplearn: Genetic programming in python.

Severino Trotta, Lucie Flek, and Charles Welch. 2022. Nearest neighbor language models for stylistic controllable generation. In *Proceedings of the 2nd Workshop on Natural Language Generation, Evaluation, and Metrics (GEM)*, pages 295–305, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

Leonardo Trujillo, Luis Muñoz, Edgar Galván-López, and Sara Silva. 2016. neat genetic programming: Controlling bloat naturally. *Information Sciences*, 333:21–43.

Nguyen Quang Uy, Nguyen Xuan Hoai, Michael O'Neill, Robert I McKay, and Edgar Galván-López. 2011. Semantically-based crossover in genetic programming: application to real-valued symbolic regression. *Genetic Programming and Evolvable Machines*, 12:91–119.

Mojtaba Valipour, Bowen You, Maysum Panju, and Ali Ghodsi. 2021. Symbolicgpt: A generative transformer model for symbolic regression. *arXiv:2106.14131*.

M. Virgolin and Solon P. Pissis. 2022. Symbolic regression is np-hard. *IEEE Transactions on Machine Learning Research*, 2022.

Yilong Xu, Yang Liu, and Hao Sun. 2024. Reinforcement symbolic regression machine. In *The Twelfth International Conference on Learning Representations*.

Bo Yuan, Yulin Chen, Zhen Tan, Wang Jinyan, Huan Liu, and Yin Zhang. 2024. Label distribution learning-enhanced dual-knn for text classification. In *Proceedings of the 2024 SIAM International Conference on Data Mining (SDM)*, pages 400–408. SIAM.

Xin Zheng, Zhirui Zhang, Junliang Guo, Shujian Huang, Boxing Chen, Weihua Luo, and Jiajun Chen. 2021. Adaptive nearest neighbor machine translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 368–374, Online. Association for Computational Linguistics.

Jinghui Zhong, Liang Feng, Wentong Cai, and Yew-Soon Ong. 2020. Multifactorial genetic programming for symbolic regression problems. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 50(11):4492–4505.

A Analysis of Solving Time

To quantify the additional inference overhead introduced by SRASR relative to its pretrained backbone, we conduct a runtime analysis, including RSRM as a reference due to its highest accuracy among baselines.

Benchmark	SRASR (R^2/runtime)	End2End (R^2/runtime)	RSRM (R^2/runtime)
Nguyen	0.9992 / 101.91	0.9998 / 35.27	0.9966 / 1017.44
Keijzer	0.9937 / 102.37	0.9736 / 33.58	0.9747 / 1035.19
Livermore	0.9879 / 131.20	0.9875 / 50.71	0.9822 / 1114.74
R	0.9999 / 14.51	0.9998 / 7.65	0.9985 / 1200.20
Jin	0.9999 / 98.17	0.9625 / 49.63	0.9566 / 1106.39
Neat	0.9674 / 92.17	0.9163 / 30.50	0.9205 / 1001.91
Strogatz	0.9449 / 71.07	0.8192 / 25.05	0.9191 / 1143.19
Average	**0.9847** / 87.34	0.9511 / **33.20**	0.9640 / 1088.44

Table 5: Comparison of SRASR with its underlying pretrained model (End2End) and the strongest baseline (RSRM) in terms of fitting accuracy and solving time (in seconds). All results are reported as the average over three independent runs for each method.

As shown in Table 5, SRASR incurs a modest runtime increase due to the retrieval process, but this overhead is limited. Notably, it consistently outperforms RSRM in accuracy while maintaining substantially faster inference, demonstrating that the additional computational cost is a reasonable trade-off for the achieved performance gains.