Skeleton-Preserving Coefficient Refinement for Multimodal Physical Formula Discovery

Anonymous ACL submission

Abstract

Discovering physical formulas from multimodal scientific data has recently been advanced by vision-language models (VLMs) that map images, text, and trajectories to symbolic expressions. However, the prevailing post-hoc coefficient-fitting strategy (SR^2) systematically corrupts the predicted formula skeleton: in our analysis, SR^2 alters the skeleton in 82.1% of strictly comparable samples and decreases the structural score in 100% of altered cases (mean drop 0.22), reducing average S_{struct} from 0.83 to 0.65. This reveals a fundamental structure-precision dilemma: refinement procedures that jointly search over structures and coefficients can, and in practice do, trade skeleton correctness for marginal coefficient gains. We address this by reformulating refinement as a constrained problem in which the upstream skeleton is fixed and only coefficients are updated, making structural integrity an architectural invariant rather than an optimization outcome. We propose **HAD** (*Hybrid Autoregressive-Diffusion*), a skeleton-preserving coefficient refinement module that reuses an upstream VLM's symbolic token sequence and refines only the continuous coefficients at numeric positions via an autoregressive-conditioned diffusion head. A decoupled two-stage training schedule resolves the $\sim$600M/$\sim$2M AR–diffusion parameter mismatch. On PHYSSYMBOL, HAD improves S_{struct} from 0.857 to 0.871 and S_{acc} from 0.472 to 0.482 over the strongest VLM baseline, while eliminating SR^2's structural corruption and reducing refinement cost from $\sim$12.5 s to $\sim$0.8 s per sample.

1 Introduction

Deriving and discovering physical formulas from large-scale scientific data is a core component of scientific discovery, enabling the transformation of observed phenomena into interpretable and generalizable mathematical laws (Schmidt and Lipson, 2009; Brunton et al., 2016; Champion et al., 2019). Traditional formal reasoning approaches treat physical formulas as sequences of discrete symbols and typically employ relatively lightweight model architectures, such as multilayer perceptrons (MLPs) or small Transformer modules, for regression-based prediction (Martius and Lampert, 2016; Cranmer et al., 2020; La Cava et al., 2021; Grayeli et al., 2024). While these methods have achieved remarkable success on specific tasks, they suffer from a fundamental limitation: they operate on unimodal, formalized symbolic inputs and cannot integrate the multimodal data indispensable in real-world scientific exploration (e.g., microscopy images, experimental videos, textual descriptions, and numerical measurements). This restriction severely limits the applicability of automated scientific discovery systems in complex practical scenarios (Yi et al., 2019; Bakhtin et al., 2019; Bear et al., 2021; Lu et al., 2022, 2023).

The recently proposed model for formula discovery (Liu et al., 2025) represents a significant advance in this domain, which is one of the first works leveraging multimodal large models to formula discovery and formal reasoning. It supports joint image and text inputs and successfully maps multimodal scientific data to formalized physical formulas. However, this model exhibits an inherent deficiency in its mechanism: during inference, it employs the SR^2 algorithm (Cranmer, 2023) for numerical fitting, which essentially adds residual terms to the original predicted formula and optimizes coefficients via data fitting. Although this strategy improves numerical precision to some extent, it fundamentally compromises the structural integrity and physical interpretability of the original formula. More critically, removing the SR^2 module results in severely degraded coefficient prediction accuracy, failing to meet practical application requirements. This trade-off, sacrificing the formula skeleton to gain coefficient accuracy, re-

veals a fundamental contradiction of this model.

To make this dilemma concrete, we conducted a quantitative analysis of VLM with SR^2 on the PHYSSYMBOL benchmark (Liu et al., 2025). On 100 test samples, SR^2 altered the formula skeleton in 82.1% of strictly comparable cases, and in 100% of altered cases the structural score decreased, with a mean drop of 0.22. The average structural score fell from 0.826 (before SR^2) to 0.647 (after SR^2)—an absolute reduction of 21.8%. SR^2 thus does not occasionally damage the skeleton; it does so *systematically*. Moreover, the iterative genetic-programming search underlying SR^2 incurs ~ 12.5 s per sample, an order of magnitude slower than learned alternatives.

We argue that formula skeletons and numerical coefficients are not independent optimization targets but deeply coupled components: the skeleton defines the functional space in which coefficients reside, while precise coefficients validate and reinforce the effectiveness of structural hypotheses (Brunton et al., 2016; Champion et al., 2019). Based on this insight, we propose a **H**ybrid **A**utoregressive and **D**iffusion model for Formula Discovery (HAD) that aims to simultaneously optimize both formula skeletons and numerical coefficients within a unified framework, addressing the structure–precision dilemma. Figure 1 illustrates the overall pipeline.

HAD addresses this dilemma by reformulating refinement as a *skeleton-preserving* operation. Given an upstream VLM's symbolic hypothesis, HAD reuses the predicted symbolic token sequence at inference time and replaces only the continuous values at numeric positions through an autoregressive-conditioned conditional diffusion head. This design provides a token-level architectural guarantee that the discrete formula structure is never altered during refinement, while the diffusion head models the multimodal distribution of physical coefficients under structural context.

Our main contributions are summarized as follows:

(1) **A new problem formulation with quantitative motivation.** We reformulate post-hoc refinement as *skeleton-preserving coefficient refinement* and provide the first quantitative evidence that the prevailing SR^2 approach alters skeletons in 82.1% of comparable samples with 100% structural degradation.

(2) **A skeleton-preserving refinement architecture.** HAD couples a pretrained AR backbone with a lightweight conditional diffusion head at numeric positions, and—unlike end-to-end AR–diffusion hybrids such as UniGenX (Zhang et al., 2025a)—reuses the upstream symbolic token sequence to enforce token-level skeleton preservation by construction.

(3) **A decoupled two-stage training schedule** that resolves the $\sim$600M/$\sim$2M AR–diffusion parameter asymmetry, yielding improvements in both structural and coefficient accuracy over the strongest VLM baseline while reducing refinement cost from ~ 12.5 s to ~ 0.8 s per sample.

2 Related Work

Symbolic Regression and Formula Discovery. Formula discovery is typically cast as symbolic regression over discrete tokens (La Cava et al., 2021; Radwan et al., 2024), evolving from genetic programming and evolutionary algorithms to deep learning methods based on RNNs and Transformers (Petersen et al., 2021; Valipour et al., 2021). Since the release of ChatGPT (OpenAI, 2024), large language models (LLMs) have shown strong potential for scientific discovery (Wang et al., 2023; Reddy and Shojaee, 2025), yet their use in formula discovery remains largely unexplored (Reddy and Shojaee, 2025). Across both traditional genetic algorithms and neural symbolic regression, a key limitation is that they operate on unimodal, pre-formalized symbolic inputs (Zhang et al., 2025b), whereas real-world scientific exploration relies on multimodal data—images, text, videos, and numerical measurements (Liu et al., 2025; Su et al., 2025; Li et al., 2025). From microscopic images to textual protocols and video-recorded physical phenomena, fusing these modalities is central to modern practice, and methods that ignore this requirement face severe constraints in authentic scientific scenarios (Bai et al., 2025; Zhang et al., 2025b). A recent direction tackles this gap by inducing symbolic *formula skeletons* from multimodal inputs. Notably, VIPER-R1 (Liu et al., 2025) maps images, text, and trajectories to formal expressions but relies at inference on an external SR^2 stage to fit coefficients, which can alter the predicted structure and introduce nontrivial search cost. This highlights the need for methods that preserve the skeleton while estimating coefficients efficiently.

Symbolic regression with continuous coefficients. Several lines of work integrate continuous numerical values into symbolic regression.

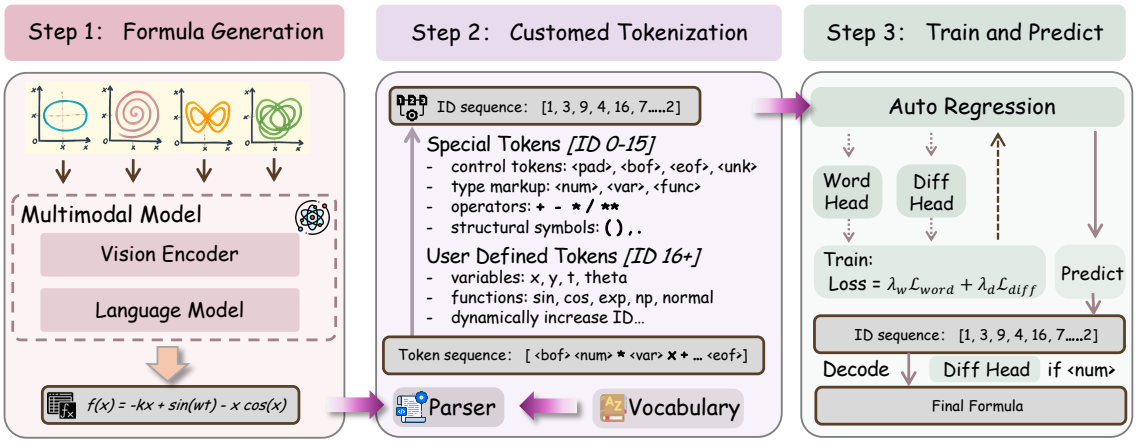

Figure 1: **Overview of the HAD Pipeline.** Step 1: A multimodal model generates an initial formula from scientific inputs. Step 2: HAD reconstructs the formula via dual autoregressive (symbolic) and diffusion (coefficient) pathways. Step 3: The refined formula with precise coefficients is produced.

Kamienny et al. (Kamienny et al., 2022) proposed end-to-end SR by encoding constants as sign/mantissa/exponent tokens within a Transformer decoder, optionally refined by BFGS post-hoc. SymFormer (Vastl et al., 2024) generates expressions with parallel symbol-classification and constant-regression heads, supervised by a combined cross-entropy and MSE loss. Both methods re-generate the entire expression end-to-end from data points, rather than refining an existing symbolic hypothesis, so they offer no architectural mechanism for skeleton preservation in multimodal formula discovery pipelines.

AR–diffusion hybrids for science. Hybrid AR–diffusion architectures have been applied to multimodal generation and scientific generation; the closest is UniGenX (Zhang et al., 2025a), which jointly trains an AR head over tokens and a diffusion head over continuous values for end-to-end scientific design. HAD adopts a similar routing but differs in *task* (post-hoc refinement on an upstream VLM hypothesis, not end-to-end generation), *guarantee* (reused symbolic token sequence preserve the skeleton by construction), and *training* (a two-stage schedule that handles the $\sim$600M/$\sim$2M parameter asymmetry under which joint training fails). A full differentiation table is provided in the supplementary material.

3 Methodology

3.1 Unified Representation for Symbols and Numerical Values

HAD takes formalized physical formulas as input and reconstructs them to improve both structural accuracy and coefficient precision (Zheng et al., 2023). Since the autoregressive module predicts symbolic tokens while the diffusion module generates numerical coefficients conditioned on hidden states, the architecture requires input data that distinguishes between these two types (Zhang et al., 2024). We therefore adopt a unified symbol-number sequence representation: a formula parsing module decomposes physical formulas into hybrid token sequences, inserting special tokens to mark symbolic and numerical positions. This hybrid representation serves as the foundation for our dual-pathway framework (Wu et al., 2024).

Scope of skeleton preservation. Because the diffusion pathway only emits values at positions already marked as numeric by the upstream predictor, the discrete formula structure passes through HAD unchanged. This skeleton preservation is a property of the architecture, not of the optimization target, and holds regardless of the diffusion head's output. One caveat is that our representation does not distinguish coefficients from exponents: refining a numeric value in an exponent position can alter the functional form (e.g., $x^2 \rightarrow x^3$). We therefore report both a token-level structural score and SymPy canonical equivalence; excluding exponent positions from refinement is a straightforward extension that does not affect our main conclusions.

3.2 Dual-Pathway Architecture

At each generation step, the model determines whether the next token is symbolic (dispatched to the AR pathway) or numerical (dispatched to the diffusion pathway). Formally, the generation of element s_i follows:

$$p(\mathbf{s}) = \prod_{i=1}^{|\mathbf{s}|} p(s_i \mid \mathbf{s}_{<i}),$$

$$p(s_i \mid \mathbf{s}_{<i}) = \begin{cases} p_{\text{AR}}^{\text{TopP}}(w_i \mid \mathbf{s}_{<i}) & \text{if } s_i \in \mathcal{V}_{\text{sym}}, \\ p_{\text{diff}}(x_i \mid \mathbf{c}_i) & \text{if } s_i \in \mathcal{V}_{\text{num}}. \end{cases} \tag{1}$$

where $\mathbf{s}_{<i}$ denotes the prefix before position i, $\mathcal{V}_{\text{sym}}$ the symbolic vocabulary (operators, variables, functions), $\mathcal{V}_{\text{num}}$ the numerical positions, $p_{\text{AR}}^{\text{TopP}}$ the autoregressive distribution with Top-p sampling, p_{diff} the conditional diffusion distribution, and $\mathbf{c}_i = f_{\text{enc}}(\mathbf{s}_{<i}; \theta_{\text{enc}})$ the contextual representation. The coupling is *structural-to-numerical* at inference: AR hidden states condition the diffusion process via AdaLN, so coefficient generation is structure-aware. During training, the AR and diffusion modules are optimized in separate stages (Section 4); at inference, AR hidden states serve as conditioning for the diffusion head.

3.3 Autoregressive Pathway for Symbolic Generation

3.3.1 Architecture

We initialize the AR pathway with a pretrained language model (Qwen-0.6B) rather than training from scratch. Physical formula data exhibits a long-tailed distribution of operators and variables: common patterns (e.g., linear terms, simple polynomials) dominate while complex structures are rare. We observed that models trained from scratch on this data consistently overfit to high-frequency templates regardless of augmentation or regularization, whereas pretrained language models provide compositional reasoning capabilities that mitigate this issue on small, domain-specific datasets (Virgolin and Pissis, 2022). We adapt the model to our hybrid representation by introducing a numerical embedding layer (Section 3.3.2).

3.3.2 Numerical Embedding Fusion

Beyond generating symbolic tokens, the autoregressive module must also perceive the positions of numerical tokens within the sequence. This spatial awareness is critical for ensuring that subsequently generated coefficients are embedded at appropriate positions. To achieve this, we employ a numerical embedding fusion mechanism:

Fusion Strategy. At each position i in the sequence, the input representation is determined by the token type:

$$\mathbf{e}_i = \begin{cases} f_{\text{TokenEmbed}}(s_i) & \textit{if } m_i = 0, \\ f_{\text{NumEmbed}}(c_i) & \textit{if } m_i = 1, \end{cases} \tag{2}$$

where $f_{\text{TokenEmbed}}(\cdot)$ denotes the pretrained token embedding, $f_{\text{NumEmbed}}(c_i) = \mathbf{W}_{\text{num}} \cdot c_i + \mathbf{b}_{\text{num}}$ is a trainable linear projection ($\mathbf{W}_{\text{num}} \in \mathbb{R}^{d \times 1}$, d being the hidden dimension), and $m_i \in \{0, 1\}$ is the binary mask ($m_i = 0$ for symbolic, $m_i = 1$ for numerical tokens).

This fusion provides **contextual awareness** by incorporating numerical values into the embedding space, and **position guidance** through distinct embedding patterns that mark where coefficients should appear. During training, the model learns bidirectional associations: symbolic tokens influence numerical placement, while numerical values inform the selection of surrounding operators and variables.

3.4 Diffusion Pathway for Coefficient Generation

We instantiate the coefficient head as a conditional diffusion model rather than a direct regression head. Physical coefficient distributions are generally non-Gaussian and can be multimodal: coefficients of the same operator type across different physical regimes (e.g., damping coefficients in stiff vs. underdamped systems) cluster at distinct values. A regression head trained with MSE predicts the conditional mean and tends to oversmooth toward template values, whereas a diffusion head models the full conditional distribution and can sample from the appropriate mode given the AR hidden state.

3.4.1 Structure-Conditioned Generation

A key aspect of our approach is that the diffusion model directly incorporates structural information from the autoregressive pathway as conditioning, enabling structure-aware coefficient generation without requiring destructive post-hoc optimization (He et al., 2023).

Conditioning Mechanism. At each position i where a coefficient needs to be generated, the diffusion model receives the corresponding hidden state $\mathbf{h}_i$ from the autoregressive module as the conditioning vector. This hidden state encodes three types of information: **local context** (the operators and variables immediately surrounding position i), **global structure** (the overall formula pattern captured through self-attention in preceding layers),

and **semantic meaning** (the physical or mathematical interpretation suggested by the symbolic skeleton). The conditioned generation process can be formalized as:

$$c_0^{(i)} = f_{\text{diff}}(\mathbf{h}_i; \epsilon_\theta, K), \tag{3}$$

where $f_{\text{diff}}(\cdot)$ denotes the K-step reverse diffusion process starting from Gaussian noise, guided by the denoising network ϵ_θ conditioned on $\mathbf{h}_i$.

Inference Efficiency. During inference, when the autoregressive model predicts a numerical token, the diffusion module executes K denoising steps (typically $K = 1000$) to generate a single coefficient. While this incurs some computational overhead compared to direct regression, it remains far more efficient than SR^2's exhaustive search, and the generation can be further accelerated using DDIM (Song et al., 2020) with fewer sampling steps (e.g., $K = 50$) without degrading precision below the τ=0.01 accuracy threshold.

3.4.2 Denoising Network with AdaLN

The conditional diffusion module activates exclusively at numeric positions (around 15% of tokens), utilizing a lightweight three-layer MLP with AdaLN to predict noise $\epsilon_\theta = \text{Denoiser}(c_t, \mathbf{h}_{\text{cond}})$ based on diffusion timestep t and autoregressive hidden states $\mathbf{h}_{\text{AR}}$. More details are in Appendix F.

4 Training Procedure

4.1 Two-Stage Training Strategy

We adopt a **two-stage training strategy** to ensure effective optimization of both the autoregressive and diffusion pathways. This design is motivated by the substantial disparity in parameter scales and learning dynamics between the two modules: the autoregressive module (Qwen-0.6B) contains $\sim$600M parameters, while the lightweight diffusion denoiser has only $\sim$2M parameters.

Challenge of direct joint training. A naive single-stage objective that updates both pathways from the start leads to an optimization conflict. When training is continued for sufficient epochs to fully optimize the diffusion pathway, the large autoregressive backbone rapidly overfits the training distribution, causing validation performance to diverge. Conversely, when training is stopped early to prevent AR overfitting, the diffusion pathway remains under-trained and fails to generate accurate coefficients. This dilemma arises from the imbalanced capacity and different convergence rates of the two modules: the pretrained AR backbone quickly memorizes symbolic and template-level patterns, whereas the small diffusion denoiser requires more iterations to learn the conditional coefficient distribution.

Solution: Staged Optimization. Given the substantial differences in model parameters between the two components, we adopt a two-stage training strategy. Figures 3 and 4 illustrate the loss curves for both stages of our training process.

Stage I. In the first training stage, as shown in Figure 3, we freeze the diffusion model and perform supervised fine-tuning (SFT) exclusively on the autoregressive model using the augmented dataset of 50k samples.

Stage II. In the second training stage, as depicted in Figure 4, we train the diffusion model while the autoregressive component inherits the weights from the first stage and remains frozen. This approach prevents the degradation of previously acquired knowledge structures during subsequent training. For the diffusion model training, we do not employ early stopping but instead set a fixed training duration of 80 epochs. As evident from the figure, the model achieves stable convergence upon completing the full 80 epochs.

This decoupling ensures that each pathway receives adequate training while maintaining overall model stability.

4.2 Stage I: Autoregressive Pathway Pre-finetuning

In Stage I, we exclusively train the autoregressive pathway to establish a strong foundation for the subsequent joint training phase.

Roles of the Autoregressive Module. The AR pathway fulfills two key functions: **unified sequence modeling**, which captures the distribution of hybrid symbol-number sequences, and **dynamic routing**, which acts as a dispatcher to determine whether to generate a symbolic token directly or activate the diffusion pathway for numerical synthesis.

Loss Function Composition. In Stage I, we freeze all parameters of the diffusion network f_ϕ and exclusively train the autoregressive module, including the pretrained LLM, the numerical embedding layer g_θ, and the optional projection head, so that the model first acquires a robust capability for

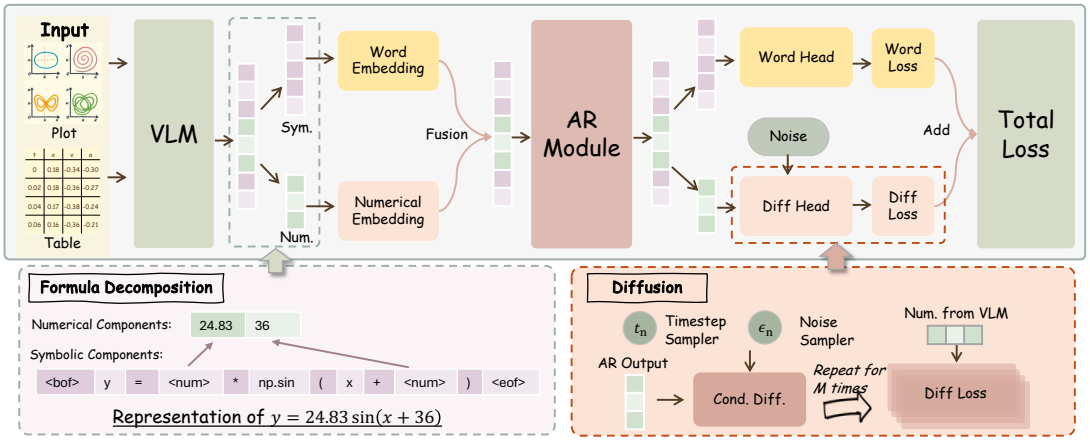

Figure 2: **Training Pipeline of HAD Model.** The framework consists of two pathways: the autoregressive pathway processes word embeddings through a Word Head to generate symbolic structures, while the diffusion pathway processes numerical embeddings through a Diff Head to predict coefficients. Both pathways are independently optimized with their respective loss functions (Word Loss and Diff Loss) to form the Total Loss. A pretrained multimodal model (e.g., VIPER-R1) provides initial embeddings for both modalities.

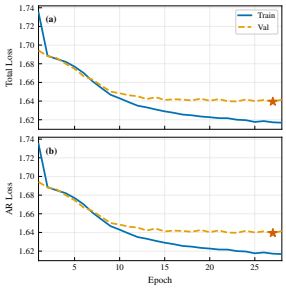

Figure 3: Loss curves for Stage I training, where we freeze the diffusion model and train only the autoregressive model.

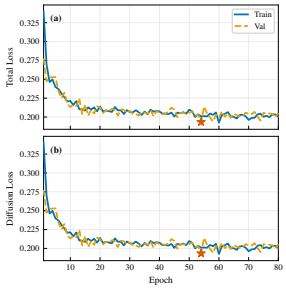

Figure 4: Loss curves for Stage II training, where we freeze the AR model and train only the diffusion model.

structural sequence prediction before continuous-value generation is introduced.

Training Objective. Let $\mathbf{s} = (s_1, s_2, \ldots, s_T)$ denote an input token sequence, where each s_t belongs to either the symbolic vocabulary $\mathcal{V}$ or the special placeholder that marks a continuous numerical position. We define the set of valid (non-padding) positions as $\mathcal{M} = \{t \mid s_t \neq \text{padding token}\}$. For positions where $s_t =$ <num>, the associated ground-truth scalar v_t is injected through the numerical embedding layer g_θ to replace the default token embedding, thereby enabling the backbone to perceive continuous values within the symbolic stream.

The Stage I loss follows the standard autoregressive next-token prediction objective with label smoothing. Concretely, we minimize:

$$\mathcal{L}_{\text{AR}} = -\frac{1}{|\mathcal{M}|} \sum_{t \in \mathcal{M}} \Bigg[(1 - \epsilon_{\text{ls}}) \log p_\theta(s_{t+1} \mid s_{\leq t}) + \frac{\epsilon_{\text{ls}}}{|\mathcal{V}|} \sum_{w \in \mathcal{V}} \log p_\theta(w \mid s_{\leq t}) \Bigg]. \tag{4}$$

where ϵ_{ls} is the label smoothing coefficient (set to 0.1 in all experiments) and $p_\theta(\cdot \mid s_{\leq t})$ is the softmax distribution produced by the LLM head at position t. The overall Stage I objective is $\mathcal{L}_{\text{Stage-I}} = \mathcal{L}_{\text{AR}}$. Upon convergence of Stage I, all autoregressive parameters are frozen and the training proceeds to Stage II , which activates the diffusion network f_ϕ to learn continuous value generation conditioned on the hidden states produced by the now-fixed backbone.

Training Configuration. We fine-tune for 50 epochs with batch size 64 using AdamW (Loshchilov and Hutter, 2017) with learning rate 5×10^{-5} and a cosine schedule, freezing the first 8 Transformer layers to preserve Qwen's pretrained knowledge. Training converges when validation loss plateaus, typically within 30 epochs (∼2.5 hours on 8 A800 GPUs).

4.3 Stage II: Conditional Diffusion Training

In Stage II, we freeze the entire AR module and activate the diffusion network f_ϕ. The frozen backbone serves as a fixed condition encoder whose hidden states are treated as constants during diffusion training.

Numerical Position Selection. Let $\mathcal{N} = \{t \in \mathcal{M} \mid s_{t+1} = \texttt{<num>}\}$ denote the set of valid positions whose *next* token is a numerical placeholder. For each $t \in \mathcal{N}$, we extract a pair $(\mathbf{h}_t, x_t)$, where $\mathbf{h}_t \in \mathbb{R}^d$ is the hidden state produced by the frozen backbone at position t, and $x_t \in \mathbb{R}$ is the ground-truth scalar associated with position $t{+}1$. Positions that are either padding or non-numerical are excluded from the diffusion objective entirely.

Training Objective. We adopt the standard denoising diffusion probabilistic model (DDPM) (Ho et al., 2020) formulation with an ϵ-prediction parameterization. Given a linear noise schedule $\{\beta_k\}_{k=1}^{K}$ from $\beta_1 = 10^{-4}$ to $\beta_K = 0.02$ with $K{=}1000$ timesteps and the cumulative product $\bar{\alpha}_k = \prod_{i=1}^{k}(1-\beta_i)$, the forward process corrupts each ground-truth value x_0 as:

$$x_k = \sqrt{\bar{\alpha}_k}\, x_0 + \sqrt{1-\bar{\alpha}_k}\, \epsilon, \quad \epsilon \sim \mathcal{N}(0, I). \quad (5)$$

The diffusion network f_ϕ is trained to recover the added noise from the corrupted input, conditioned on the backbone hidden state. The Stage II loss is:

$$\mathcal{L}_{\text{Diff}} = \frac{1}{|\mathcal{N}|} \sum_{t \in \mathcal{N}} \mathbb{E}_{k,\epsilon} \big\| \epsilon - f_\phi(x_k, k, \mathbf{h}_t) \big\|^2, \quad (6)$$

where k is uniformly sampled from $\{1, \ldots, K\}$ for each numerical position independently, and $\mathbf{h}_t$ is treated as a *constant* with respect to ϕ since the backbone is frozen. Scalars are denoised independently since contextual dependencies are already encapsulated in $\mathbf{h}_t$, and the 1D scalar nature of the targets allows a compact MLP with AdaLN (Peebles and Xie, 2023) instead of heavier U-Net or Transformer architectures. The overall Stage II objective is $\mathcal{L}_{\text{Stage-II}} = \mathcal{L}_{\text{Diff}}$.

Training Configuration. Stage II training runs for 80 epochs with batch size 64 and AdamW (Loshchilov and Hutter, 2017) with $\beta_1 = 0.9$, $\beta_2 = 0.999$, and weight decay 0.01. We apply gradient clipping with threshold 1.0 for stability and mixed-precision (FP16) to accelerate computation; training completes in approximately 0.8 hours on 8 A800 GPUs.

5 Experiments

5.1 Experimental Setting

We benchmark HAD on the PhysSymbol dataset (4,500/500 train–validation split) by first extracting formula pairs $\mathcal{D} = \{(F_i^{\text{VIPER}}, F_i^{\text{GT}})\}_{i=1}^{5000}$ via a pretrained VIPER-R1 model to generate hybrid token sequences. Evaluation relies on Structural Score (S_{struct}) for skeleton exact match and Accuracy Score (S_{acc}) for joint structure and coefficient precision within $\tau = 0.01$. For more details, please refer to Appendix E.

5.2 Results

As shown in Table 1, we evaluate multiple open-source and closed-source vision-language models on a moderate-difficulty subset of the PhysSymbol benchmark (the same subset used by VIPER-R1 (Liu et al., 2025) to reveal cross-model performance differences). All closed-source models perform poorly ($S_{\text{struct}} < 0.30$). Among open-source models, VIPER-R1-7B significantly outperforms all others. Applying HAD as a skeleton-preserving refinement module on top of VIPER-R1-7B improves both metrics: S_{struct} increases from 0.857 to 0.871 and S_{acc} from 0.472 to 0.482. Critically, HAD achieves these gains *without* the SR^2 module: the original VIPER-R1 skeleton is preserved by construction, so the gains are pure coefficient-level improvements rather than restructured formulas that happen to fit the data. We additionally report Edit Distance Similarity (S_{EDS}, higher is better: 0.3831 vs. 0.3799) and Raw Edit Distance (S_{RED}, lower is better: 51.26 vs. 52.19), confirming improvements under general string-similarity metrics. The next subsection provides direct evidence that this skeleton preservation is a property SR^2 fundamentally lacks.

Out-of-distribution evaluation. HAD demonstrates strong generalization on an unseen test set ($S_{\text{acc}} = 0.3017$, $S_{\text{struct}} = 0.3943$) and is theoretically compatible with any multimodal upstream system producing formula hypotheses.For more details, please refer to Appendix C.

Computational efficiency. HAD's diffusion pathway delivers an order-of-magnitude speedup over VIPER-R1's SR^2 module ($\sim$0.8 s vs. $\sim$12.5 s per sample) with a total training budget of 3.3 h on 8$\times$A800 GPUs.For more details, please refer to Appendix D.

Qualitative Analysis. Qualitative results show that HAD successfully preserves the formula skeleton while leveraging its diffusion pathway to refine coefficient errors to within the $\tau = 0.01$ threshold across various complex formulas.For more details, please refer to Appendix B.

Category	Method	PhysSymbol	
		$S_{\text{struct}} \uparrow$	$S_{\text{acc}} \uparrow$
Closed-source VLMs	GPT-4o	0.2675	0.0549
	GPT-5	0.1252	0.0613
	Claude-Sonnet-4.5	0.2857	0.0204
	Claude-Haiku-4.5	0.2631	0.0406
	Gemini-2.5-Pro	0.0017	0.0067
Open-source VLMs	GLM-4.5V	0.0432	0.0134
	Qwen-VL-Max	0.0108	0.0233
	Qwen2.5-VL-72B-Instruct	0.0727	0.2424
	Qwen3-VL-30B-A3B-Instruct	0.0761	0.0343
	VIPER-R1-7B	0.8570	0.4721
Ours	**HAD**	**0.8710**	**0.4817**

Table 1: Comparison of vision-language models (VLMs) and our HAD model on the PhysSymbol test set. For both metrics reported, higher values refer to better performance.

SR^2 Systematically Corrupts the Formula Skeleton. Empirical analysis shows that VIPER-R1's SR^2 module systematically corrupts formula skeletons, causing mean S_{struct} to drop from 0.826 to 0.647 (-21.8%), whereas HAD by construction guarantees 100% skeleton preservation by reusing the upstream token IDs verbatim.For more details, please refer to Appendix A.

5.3 Necessity of Two-Stage Training

Ablation studies show that *Direct Joint Training* causes severe overfitting and gradient conflict, as the 600M AR module dominates the 2M diffusion module, leading to poor convergence ($S_{\text{struct}} = 0.523$ / $S_{\text{acc}} = 0.287$). *One-Stage (AR only)* achieves good structural accuracy but lacks precise coefficient generation ($S_{\text{struct}} = 0.761$ / $S_{\text{acc}} = 0.472$). In contrast, our proposed *Two-Stage* optimization addresses this bottleneck by first establishing a stable AR backbone and then training the diffusion module on frozen hidden states, delivering superior performance ($S_{\text{struct}} = 0.871$ / $S_{\text{acc}} = 0.482$). For more details, please refer to Appendix H.

6 Discussion

HAD's design centers on a single architectural commitment: at inference time, the upstream symbolic token sequence are reused verbatim, so refinement cannot alter the formula skeleton. This commitment differentiates HAD from both SR^2 and from end-to-end AR–diffusion approaches (e.g., UniGenX (Zhang et al., 2025a)) that re-generate the full sequence and therefore cannot offer token-level structural guarantees. The remaining design choices follow from this commitment: the diffusion head operates on a 1-D coefficient target, so a lightweight MLP with AdaLN suffices; AR hidden states provide structural context (operator adjacency, variable scope, formula depth) that the head exploits as conditioning; and the two-stage schedule decouples optimization of the $\sim$600M backbone from that of the $\sim$2M denoiser. Figure 6 confirms that direct joint training of these mismatched modules fails to converge.

7 Conclusion

We introduced **HAD**, a skeleton-preserving coefficient refinement module for multimodal physical formula discovery. Given a symbolic hypothesis from an upstream VLM, HAD reuses the predicted symbolic token sequence at inference time and refines only the continuous coefficients at numeric positions positions via an autoregressive-conditioned diffusion head, guaranteeing token-level structural integrity by construction. We provided the first quantitative evidence that the prevailing SR^2 approach systematically corrupts skeletons (82.1% of comparable samples, 100% degradation), motivating this design. A decoupled two-stage schedule resolves the $\sim$600M/$\sim$2M AR–diffusion parameter asymmetry under which joint training fails. On PHYSSYMBOL, HAD improves S_{struct} from 0.857 to 0.871 and S_{acc} from 0.472 to 0.482 over the strongest VLM baseline, while reducing refinement cost from $\sim$12.5 s to $\sim$0.8 s per sample. We view HAD as a *skeleton-preserving refinement interface*, distinct from end-to-end SR and generic AR-diffusion scientific generators, and hope this perspective generalizes to other domains where discrete structures and continuous parameters must be jointly modeled.

Limitations

HAD is designed for coefficient refinement under a fixed symbolic hypothesis, and is therefore complementary to methods that revise the formula structure itself. In addition, the current representation treats numerical literals uniformly, while future work could further distinguish continuous physical coefficients from structural numerical constants. Our experiments mainly focus on PhysSymbol and VIPER-R1-generated hypotheses, leaving broader validation across upstream models, domains, and real experimental settings as future work. Finally, two-stage training and diffusion sampling introduce additional complexity, motivating faster samplers and lightweight distillation in future extensions.

References

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, and 1 others. 2025. Qwen2. 5-vl technical report. *arXiv preprint arXiv:2502.13923*.

Anton Bakhtin, Laurens van der Maaten, Justin Johnson, Laura Gustafson, and Ross Girshick. 2019. Phyre: A new benchmark for physical reasoning. *Advances in Neural Information Processing Systems*, 32.

Daniel M Bear, Elias Wang, Damian Mrowca, Felix J Binder, Hsiao-Yu Fish Tung, RT Pramod, Cameron Holdaway, Sirui Tao, Kevin Smith, Fan-Yun Sun, and 1 others. 2021. Physion: Evaluating physical prediction from vision in humans and machines. *arXiv preprint arXiv:2106.08261*.

Steven L Brunton, Joshua L Proctor, and J Nathan Kutz. 2016. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the national academy of sciences*, 113(15):3932–3937.

Kathleen Champion, Bethany Lusch, J Nathan Kutz, and Steven L Brunton. 2019. Data-driven discovery of coordinates and governing equations. *Proceedings of the National Academy of Sciences*, 116(45):22445–22451.

Miles Cranmer. 2023. Interpretable machine learning for science with pysr and symbolicregression. jl. *arXiv preprint arXiv:2305.01582*.

Miles Cranmer, Alvaro Sanchez Gonzalez, Peter Battaglia, Rui Xu, Kyle Cranmer, David Spergel, and Shirley Ho. 2020. Discovering symbolic models from deep learning with inductive biases. *Advances in neural information processing systems*, 33:17429–17442.

Arya Grayeli, Atharva Sehgal, Omar Costilla Reyes, Miles Cranmer, and Swarat Chaudhuri. 2024. Symbolic regression with a learned concept library. *Advances in Neural Information Processing Systems*, 37:44678–44709.

Haoran He, Chenjia Bai, Kang Xu, Zhuoran Yang, Weinan Zhang, Dong Wang, Bin Zhao, and Xuelong Li. 2023. Diffusion model is an effective planner and data synthesizer for multi-task reinforcement learning. *Advances in neural information processing systems*, 36:64896–64917.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851.

Pierre-Alexandre Kamienny, Stéphane d'Ascoli, Guillaume Lample, and François Charton. 2022. End-to-end symbolic regression with transformers. *Advances in Neural Information Processing Systems*, 35:10269–10281.

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

Ruikun Li, Yan Lu, Shixiang Tang, Biqing Qi, and Wanli Ouyang. 2025. Mllm-based discovery of intrinsic coordinates and governing equations from high-dimensional data. *arXiv preprint arXiv:2505.11940*.

Jiaqi Liu, Songning Lai, Pengze Li, Di Yu, Wenjie Zhou, Yiyang Zhou, Peng Xia, Zijun Wang, Xi Chen, Shixiang Tang, and 1 others. 2025. Mimicking the physicist's eye: A vlm-centric approach for physics formula discovery. *arXiv preprint arXiv:2508.17380*.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Pan Lu, Hritik Bansal, Tony Xia, Jiacheng Liu, Chunyuan Li, Hannaneh Hajishirzi, Hao Cheng, Kai-Wei Chang, Michel Galley, and Jianfeng Gao. 2023. Mathvista: Evaluating mathematical reasoning of foundation models in visual contexts. *arXiv preprint arXiv:2310.02255*.

Pan Lu, Swaroop Mishra, Tanglin Xia, Liang Qiu, Kai-Wei Chang, Song-Chun Zhu, Oyvind Tafjord, Peter Clark, and Ashwin Kalyan. 2022. Learn to explain: Multimodal reasoning via thought chains for science question answering. *Advances in Neural Information Processing Systems*, 35:2507–2521.

Georg Martius and Christoph H Lampert. 2016. Extrapolation and learning equations. *arXiv preprint arXiv:1610.02995*.

OpenAI. 2024. Gpt-4o system card. *Preprint*, arXiv:2410.21276.

William Peebles and Saining Xie. 2023. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4195–4205.

Brenden K. Petersen, Mikel Landajuela, T. Nathan Mundhenk, Claudio P. Santiago, Soo K. Kim, and Joanne T. Kim. 2021. Deep symbolic regression: Recovering mathematical expressions from data via risk-seeking policy gradients. In *International Conference on Learning Representations (ICLR)*.

Yousef Radwan, Gabriel Kronberger, and Stephan Winkler. 2024. A comparison of recent algorithms for symbolic regression to genetic programming. *Preprint*, arXiv:2406.03585.

Chandan K Reddy and Parshin Shojaee. 2025. Towards scientific discovery with generative ai: Progress, opportunities, and challenges. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 28601–28609.

Michael Schmidt and Hod Lipson. 2009. Distilling free-form natural laws from experimental data. *Science*, 324(5923):81–85.

Jiaming Song, Chenlin Meng, and Stefano Ermon. 2020. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*.

Zhaochen Su, Peng Xia, Hangyu Guo, Zhenhua Liu, Yan Ma, Xiaoye Qu, Jiaqi Liu, Yanshu Li, Kaide Zeng, Zhengyuan Yang, and 1 others. 2025. Thinking with images for multimodal reasoning: foundations, methods, and future frontiers. *arXiv preprint arXiv:2506.23918*.

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

Martin Vastl, Jonáš Kulhánek, Jiří Kubalík, Erik Derner, and Robert Babuška. 2024. Symformer: End-to-end symbolic regression using transformer-based architecture. *IEEE Access*, 12:37840–37849.

Marco Virgolin and Solon P Pissis. 2022. Symbolic regression is NP-hard. *arXiv preprint arXiv:2207.01018*.

Hanchen Wang, Tianfan Fu, Yuanqi Du, Wenhao Gao, Kexin Huang, Ziming Liu, and 1 others. 2023. Scientific discovery in the age of artificial intelligence. *Nature*, 620:47–60.

Zhaofeng Wu, Linlu Qiu, Alexis Ross, Ekin Akyürek, Boyuan Chen, Bailin Wang, Najoung Kim, Jacob Andreas, and Yoon Kim. 2024. Reasoning or reciting? exploring the capabilities and limitations of language models through counterfactual tasks. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*.

Kexin Yi, Chuang Gan, Yunzhu Li, Pushmeet Kohli, Jiajun Wu, Antonio Torralba, and Joshua B Tenenbaum. 2019. Clevrer: Collision events for video representation and reasoning. *arXiv preprint arXiv:1910.01442*.

Gongbo Zhang, Yanting Li, Renqian Luo, Pipi Hu, Yang Yang, Zeru Zhao, Lingbo Li, Guoqing Liu, Zun Wang, Ran Bi, Kaiyuan Gao, Liya Guo, Yu Xie, Chang Liu, Jia Zhang, Tian Xie, Robert Pinsler, Claudio Zeni, Ziheng Lu, and 15 others. 2025a. UniGenX: a unified generative foundation model that couples sequence, structure and function to accelerate scientific design across proteins, molecules and materials. *arXiv preprint arXiv:2503.06687*.

Jingyi Zhang, Jiaxing Huang, Sheng Jin, and Shijian Lu. 2024. Vision-language models for vision tasks: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 46(8):5625–5644.

Zhen Zhang, Zhichu Ren, Chia-Wei Hsu, Weibin Chen, Zhang-Wei Hong, and et al. 2025b. A multimodal robotic platform for multi-element electrocatalyst discovery. *Nature*. Published 23 Sept 2025, doi:10.1038/s41586-025-09640-5.

Yizhen Zheng, Huan Yee Koh, Jiaxin Ju, Anh TN Nguyen, Lauren T May, Geoffrey I Webb, and Shirui Pan. 2023. Large language models for scientific synthesis, inference and explanation. *arXiv preprint arXiv:2310.07984*.

Appendix

A SR^2 Systematically Corrupts the Formula Skeleton

A central motivation of HAD is that SR^2, while improving coefficient fit, alters the formula skeleton in unpredictable ways. To quantify this, we ran VIPER-R1-7B on 100 PHYSSYMBOL test samples and compared S_{struct} before and after the SR^2 stage. Of the 96 samples with parseable SR^2 output, 39 were strictly comparable via SymPy canonical form; the remaining 57 contained non-standard operators or compositions that SymPy could not canonicalize and are conservatively treated as "unchanged" for the lower bound.

Table 2 reports the results. Mean S_{struct} drops from 0.826 to 0.647 (-0.180, -21.8%). SR^2 altered the skeleton in $32/39$ strictly comparable samples (82.1%), with a conservative lower bound of 33.3% over all 96 matched samples. Crucially, in 100% of altered cases ($32/32$), S_{struct} decreased (mean drop 0.22, median 0.18, max 0.50); we did not observe a single case in which SR^2 altered the skeleton in a way that *increased* S_{struct}.

These results substantiate the structure–precision dilemma motivating HAD: SR^2 systematically corrupts a large fraction of skeletons with no observed structural improvement. HAD avoids this by construction—at inference, the upstream symbolic token sequence are reused verbatim, so the token-level skeleton is guaranteed to be preserved regardless of the diffusion head's coefficient predictions.

B Qualitative Analysis.

Figure 5 presents three representative examples comparing VIPER-R1's initial predictions with HAD's refined outputs and groundtruth formulas. In each case, HAD preserves the structural skeleton while correcting coefficient values to closely match the ground truth, illustrating the complementary roles of the AR and diffusion pathways. For instance, in a 4-term formula with polynomial, interaction, and trigonometric components (Example 1), VIPER-R1's coefficients deviate by up to 0.09 from ground truth; HAD's diffusion pathway corrects all five coefficients, reducing the maximum error to within the τ=0.01 threshold. Similar refinement is observed across formulas of varying complexity, including 5-term expressions with stochastic noise terms.

C Out-of-distribution evaluation.

On a held-out test set featuring a larger vocabulary and unseen operator combinations, HAD achieves Formula Accuracy 0.3017 and Structural Score 0.3943, indicating that the refinement module generalizes beyond training templates. As a refinement back-end, HAD is in principle compatible with any multimodal system that produces symbolic formula hypotheses, though the present experiments validate it specifically on VIPER-R1; broader upstream-model validation is left to future work.

D Computational efficiency.

VIPER-R1's SR^2 module requires iterative genetic programming and BFGS optimization at $\sim$12.5 s per sample, whereas HAD's diffusion pathway generates coefficients via DDIM (K=50) in $\sim$0.8 s per sample, an order-of-magnitude speedup. The total training budget is 2.5 h (Stage I) + 0.8 h (Stage II) on 8$\times$A800 GPUs.

E Experimental Setting

Dataset. We benchmark HAD on the PHYSSYMBOL dataset (Liu et al., 2025), which contains 5,000 multimodal experimental samples spanning visual observations (trajectory plots, experimental setups), textual descriptions, numerical measurements, and ground-truth formulas. Unless otherwise stated, all main results in this section use the **original** 5,000**-sample set** with a $90/10$ train–validation split (4,500 / 500) and the held-out PHYSSYMBOL benchmark test set for final evaluation.

Data preprocessing. Since HAD refines VIPER-R1's initial predictions rather than operating on raw multimodal inputs, we first apply the pretrained VIPER-R1 model to all 5,000 samples to obtain formula reconstruction pairs

$$\mathcal{D} = \left\{ \left(F_i^{\text{VIPER}},\, F_i^{\text{GT}} \right) \right\}_{i=1}^{5000}.$$

Each pair is then processed through our formula parser to produce hybrid token sequences $(S_i,\, C_i,\, M_i)$.

Evaluation Metrics. Following VIPER-R1, we adopt two metrics. **Structural Score** (S_{struct}) measures skeleton correctness: both predicted and

Metric	Value
Aggregate structural score	
Mean S_{struct} before SR^2	0.826
Mean S_{struct} after SR^2	0.647
Absolute drop	-0.180 (-21.8%)
Per-sample skeleton alteration	
Altered skeletons (strict, 39 samples)	32 (82.1%)
Altered skeletons (lower bound, 96 samples)	$\geq 33.3\%$
Of altered: cases with S_{struct} decrease	32/32 (100%)
Drop distribution (mean / med. / Q25 / Q75 / max)	0.22 / 0.18 / 0.15 / 0.28 / 0.50

Table 2: Quantitative evidence that SR^2 corrupts formula skeletons. Measured on 100 PHYSSYMBOL held-out test samples; "strict" denotes samples for which both predicted and ground-truth formulas are canonicalizable by SymPy.

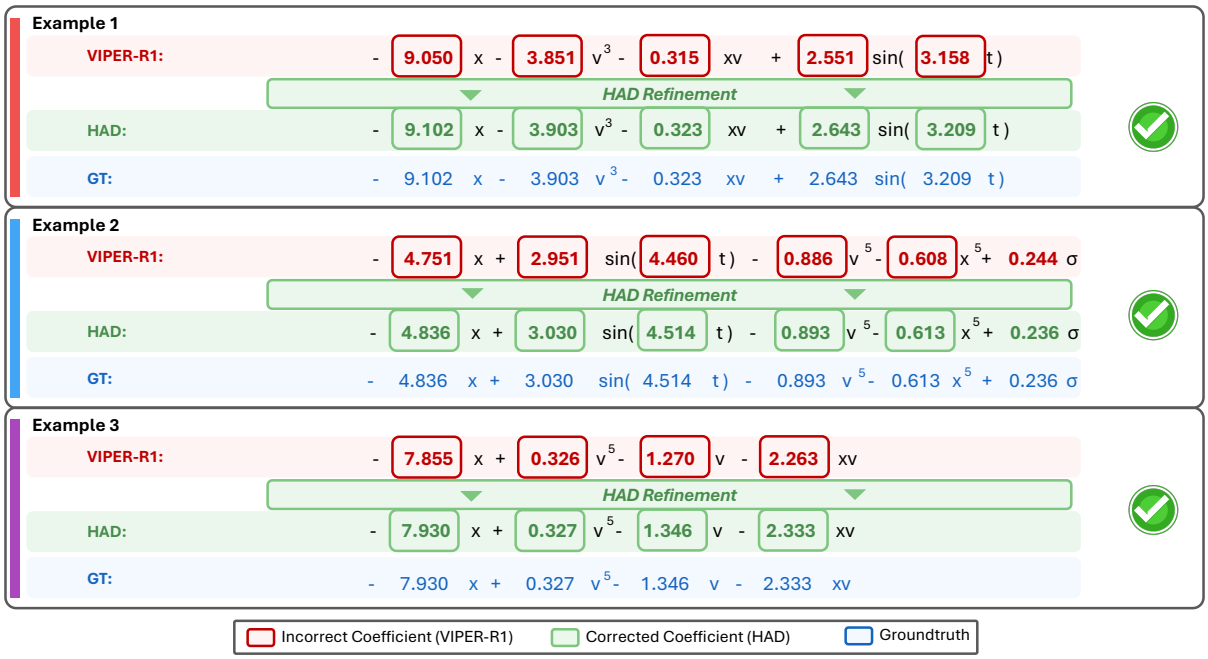

Figure 5: **Qualitative Examples.** Selected formula predictions from VIPER-R1 (input) and HAD (refined output) compared to the ground truth. HAD preserves the symbolic skeleton while correcting coefficient values (highlighted in **bold**).

ground-truth formulas are normalized by replacing all coefficients with a placeholder $\langle c \rangle$, and an exact match is computed: $S_{\text{struct}} = \frac{1}{N}\sum_{i=1}^{N} \mathbb{1}[\tilde{F}_i^{\text{pred}} = \tilde{F}_i^{\text{GT}}]$. **Accuracy Score** ($S_{\text{acc}}$) requires both structural match *and* coefficient precision within tolerance τ=0.01 (detailed formulations in supplementary material): $S_{\text{acc}} = \frac{1}{N}\sum_{i=1}^{N} \mathbb{1}[\tilde{F}_i^{\text{pred}} = \tilde{F}_i^{\text{GT}} \wedge \max_j |c_{i,j}^{\text{pred}} - c_{i,j}^{\text{GT}}| < \tau]$.

F Denoising Network with AdaLN

The conditional diffusion module activates only at numeric positions (approximately 15% of tokens in the PHYSSYMBOL training set), while symbolic positions are handled exclusively by the AR pathway. We use a lightweight three-layer MLP with AdaLN (Peebles and Xie, 2023) as the denoising network:

$$\begin{aligned} \mathbf{h}_{\text{cond}} &= \text{MLP}_{\text{time}}(t) + \text{MLP}_{\text{AR}}(\mathbf{h}_{\text{AR}}), \\ \epsilon_\theta &= \text{Denoiser}(c_t, \mathbf{h}_{\text{cond}}), \end{aligned} \tag{7}$$

where $\mathbf{h}_{\text{AR}}$ is the autoregressive hidden state and t is the diffusion timestep. AdaLN (Peebles and Xie, 2023) modulates features via $\mathbf{h}_{\text{cond}}$ for context-aware coefficient generation (see supplementary material for the full specification).

G Challenge of Direct Joint Training.

Directly training both pathways from scratch leads to optimization conflicts. If trained for *sufficient epochs* to fully optimize the diffusion pathway, the

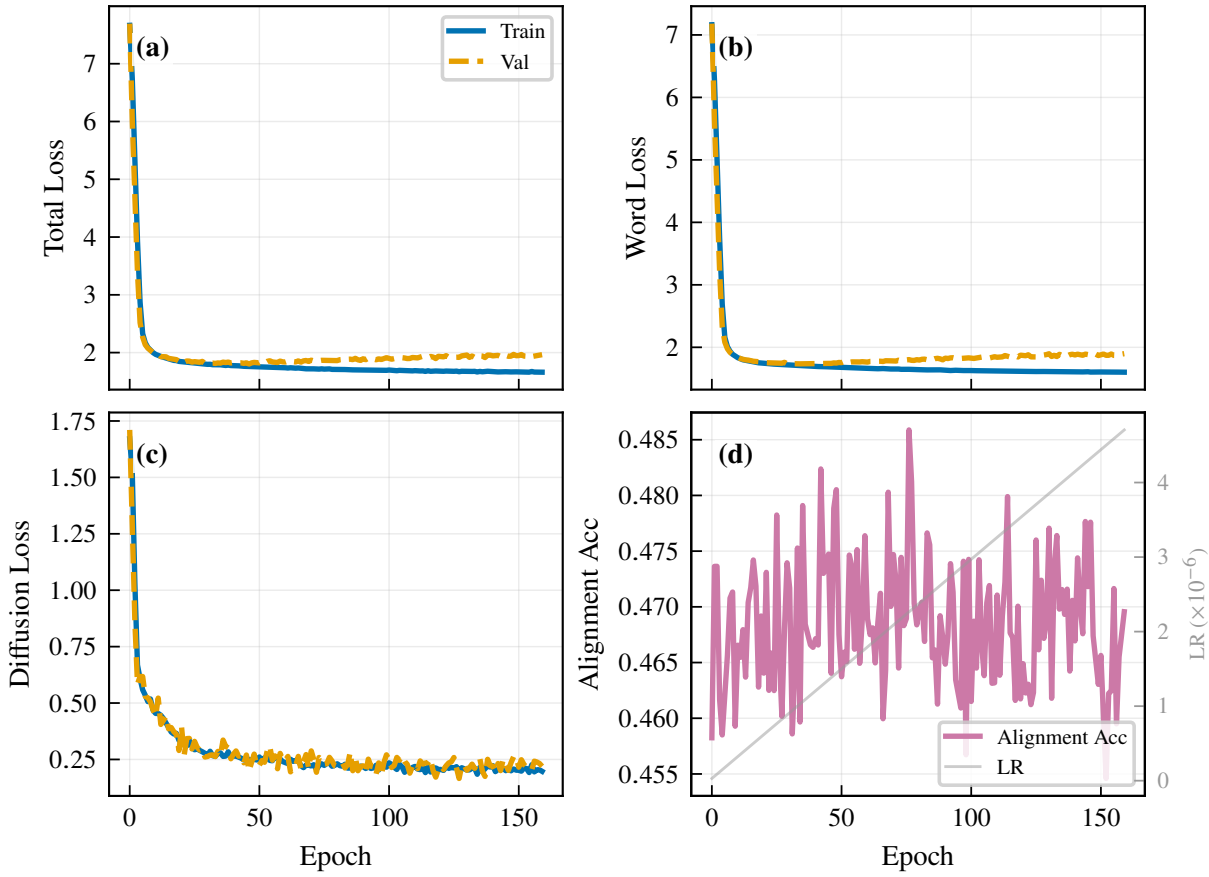

Figure 6: **Ablation Study: Direct Joint Training without Stage I.** Training dynamics over 160 epochs reveal overfitting when skipping supervised fine-tuning. **(a, b)** Total Loss and Word Loss both exhibit training-validation divergence (train ≈1.5, val ≈1.9). **(c)** Diffusion Loss converges to ≈0.25 for both splits. **(d)** Alignment accuracy fluctuates around 0.47 with high variance despite a linear LR warmup.

autoregressive module experiences severe overfitting, causing divergence on the validation set. Conversely, if training terminates *early* to prevent AR overfitting, the diffusion pathway remains undertrained and fails to generate accurate coefficients.

This optimization dilemma stems from the **imbalanced capacity** and **different convergence rates** of the two modules: the large pretrained AR model quickly memorizes training patterns, while the small diffusion denoiser requires more iterations to learn the coefficient distribution.

H Necessity of Two-Stage Training

To validate the necessity of staged optimization, we compare three training configurations: (1) *Direct Joint Training*, where both pathways are trained jointly from the start, (2) *One-Stage (AR only)*, where only the AR pathway is used without diffusion, and (3) *Two-Stage (Ours)*, where Stage I AR pre-finetuning is followed by Stage II diffusion optimization.

As shown in Figure 6, Direct Joint Training exhibits severe optimization pathologies. The 600M-parameter AR module dominates gradient flow and rapidly overfits to training patterns, producing increasingly deterministic hidden states $\mathbf{h}_t$ that fail to provide useful conditioning for the 2M-parameter diffusion denoiser. This triggers a cascading failure: word loss shows significant training-validation divergence (Figure 6a,b), while alignment accuracy fluctuates around 0.47 with high variance (Figure 6d), indicating the model never achieves stable convergence.

One-Stage (AR only) training achieves strong structural accuracy but cannot improve coefficient precision beyond the AR module's inherent limitations, since no diffusion pathway is available for continuous value generation. Quantitatively, Direct Joint Training yields S_{struct}=0.523 / S_{acc}=0.287, One-Stage (AR only) reaches S_{struct}=0.761 / S_{acc}=0.472, while our Two-Stage achieves S_{struct}=0.871 / S_{acc}=0.482, confirming that staged optimization is essential for both struc-

tural and coefficient accuracy. Our Two-Stage approach resolves the gradient conflict by first establishing a stable, well-generalized AR backbone (Stage I), then training the diffusion module on frozen, high-quality hidden states (Stage II).