LEARNING EXECUTABLE SCORING CRITERIA FOR ACTION QUALITY ASSESSMENT

Anonymous authors
Paper under double-blind review

ABSTRACT

Action quality assessment (AQA) relies on scoring criteria that determine which evidence matters and how it contributes to a quality judgment. Existing methods typically encode these criteria implicitly in learned mappings or specify them explicitly through hand-designed evaluation structures. The latter fix how evidence is organized and combined, preventing score supervision from revising these structural choices. Yet translating domain knowledge into an effective scoring structure requires accounting for the reliability and complementarity of model-derived evidence, which must be established from data. We therefore formulate scoring criteria themselves as learnable structures: domain knowledge guides the construction of candidate criteria, while score supervision selects among these alternatives and calibrates their numerical realization. To this end, we propose Learning Executable **Scor**ing **Cri**teria for Action Quality Assessment (**CriScore**), a neuro-symbolic framework that represents learned scoring criteria as executable task-level programs. Domain knowledge guides the induction of evaluation concepts, evidence bindings, and scoring relations within a predefined evidence and operation space. Overall-score supervision trains the evidence predictors and supports program selection and numerical calibration, adapting the induced criteria to observed assessment performance. The resulting program is reused across new observations through neuro-symbolic quality reasoning, producing quality scores together with execution traces. Experiments across four AQA datasets demonstrate the effectiveness of CriScore, supporting the value of learnable scoring criteria for action quality assessment. The code and data are available at `https://anonymous.4open.science/r/CriScore/`.

1 INTRODUCTION

Action quality assessment (AQA) aims to quantify how well an observed action is performed according to domain-specific evaluation requirements (Majeedi et al., 2024; Zhou et al., 2026a). For example, judging a dive involves considering body control during flight and execution near entry, then determining how evidence from these stages contributes to an overall judgment. These requirements reflect **domain evaluation knowledge**, which specifies which aspects of a performance matter and how they should influence the assessment. **Scoring criteria** connect such knowledge with numerical evidence extracted from videos or motion sequences, specifying which evidence to use and how to combine it into a quality score.

Existing AQA methods differ in how they form these criteria. Learning-based methods infer quality directly from annotated performances, for example through stage-wise reference comparisons (Qi et al., 2025) or human-centric spatial-temporal parsing (Xu et al., 2025). These methods capture rich performance differences from data, while their scoring logic remains largely implicit in the learned mapping. In contrast, explicit-rule approaches organize assessment using predefined domain structures, such as hierarchical expert rules (Okamoto & Parmar, 2024), domain-specified causal relations (Han et al., 2025), or official scoring rules (Li & Leung, 2026). These approaches make evidence selection and composition more inspectable, but their concepts and task-specific relations require prior design. Score supervision can refine evidence predictions and numerical parameters within that design, while the prescribed organization of evidence remains fixed.

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

Translating domain evaluation knowledge into an effective scoring structure requires accounting for the properties of the available evidence. In the diving example, the importance of flight control and entry does not determine how reliably the available predictors estimate quality from those observations or whether their predictions complement one another. Predictors operating on different observation views may yield largely redundant estimates, while a source associated with an important evaluation aspect may produce noisy evidence. The reliability and complementarity of these predictions must be assessed from annotated performances, which also provide a basis for comparing alternative ways of organizing and combining them.

We therefore formulate **scoring criteria themselves as learnable structures**. Domain evaluation knowledge guides the construction of candidate criteria, while score supervision selects among these alternatives and calibrates their numerical realization. Executable programs make evidence use explicit and allow candidate criteria to be evaluated through their predicted scores.

To this end, we propose **CriScore**, an adaptive neuro-symbolic framework that couples learned quality evidence with executable task-level scoring criteria. In *scoring criteria induction*, a language model uses domain evaluation knowledge and training diagnostics to link evaluation concepts to available evidence sources and specify how their outputs contribute to the score. These bindings and relations are compiled into executable candidate programs within a predefined evidence and operation space. To determine which candidates best explain observed scores, *supervised rule calibration* trains the evidence predictors with overall-score supervision and uses held-out training predictions to select among the programs and calibrate their numerical parameters. With the selected program and fitted components fixed, *neuro-symbolic quality reasoning* applies the learned scoring relations to new observation-dependent evidence, producing quality scores together with execution traces.

Our contributions are as follows:

- We formulate scoring criteria as learnable structures for AQA, combining knowledge-guided construction with score-supervised selection and calibration to connect domain evaluation requirements with numerical evidence.
- We propose CriScore, an adaptive neuro-symbolic framework that integrates scoring criteria induction, supervised rule calibration, and neuro-symbolic quality reasoning into executable task-level programs that can be reused across observations.
- We evaluate CriScore on four AQA datasets and analyze criterion construction, program selection, and execution, demonstrating the effectiveness of learnable scoring criteria and their contribution beyond the underlying evidence predictors.

2 PRELIMINARIES

2.1 DEFINITIONS

Definition 1 (Evidence-Grounded Evaluation Concept). An evaluation concept represents an aspect of performance that is relevant to the task requirements. Let $\mathcal{C}$ denote the set of selected concepts and $\mathcal{J}$ index the available evidence sources. For an action observation i, source $j \in \mathcal{J}$ provides input x_{ij} to a quality predictor h_{θ_j}, yielding the quality estimate $z_{ij} = h_{\theta_j}(x_{ij}) \in \mathbb{R}$. An evidence binding $\mathcal{B} \subseteq \mathcal{C} \times \mathcal{J}$ associates each concept with one or more available sources. For a concept $c \in \mathcal{C}$, its associated sources and evidence are

$$\mathcal{J}_c = \{j \in \mathcal{J} : (c, j) \in \mathcal{B}\}, \qquad z_i(c) = (z_{ij})_{j \in \mathcal{J}_c}. \tag{1}$$

The concept specifies the evaluation role of its bound evidence, while the corresponding predictors are supervised by the overall quality score rather than concept-specific labels.

Definition 2 (Executable Scoring Criteria). Executable scoring criteria specify how evidence-grounded concepts contribute to the final quality assessment. We represent the criterion structure as $\mathcal{Q} = (\mathcal{C}, \mathcal{B}, \mathcal{R})$, where $\mathcal{R}$ specifies the scoring operations and their dependencies over the evidence bound by $\mathcal{B}$. A predefined specification $\mathbb{V}$ declares the available evidence sources and constrains their use in the scoring computation.

Let $\mathcal{O}$ denote the operations declared by $\mathbb{V}$, with $\mathcal{X}_o$ and Φ_o specifying the input bindings and numerical parameter domain of each operation $o \in \mathcal{O}$. Each relation $r \in \mathcal{R}$ invokes an operation o_r

with input binding u_r. The admissible criterion space induced by $\mathbb{V}$ is

$$\mathcal{A}_{\mathbb{V}} = \left\{ \mathcal{Q} \;\middle|\; \begin{array}{ll} \mathcal{B} \subseteq \mathcal{C} \times \mathcal{J}, & \mathcal{J}_c \neq \emptyset \quad (\forall c \in \mathcal{C}), \\ o_r \in \mathcal{O}, & u_r \in \mathcal{X}_{o_r} \quad (\forall r \in \mathcal{R}) \end{array} \right\}, \tag{2}$$

where the relations are additionally required to admit an executable dependency order and define a final score output. The pair $\mathcal{P} = (\mathcal{Q}, \psi)$ defines an instantiated executable scoring program, where ψ collects the numerical quantities fitted or selected after criteria induction. The numerical realization must satisfy the operation-specific domains Φ_o and preserve any settings fixed in $\mathcal{Q}$.

2.2 Problem Formulation

Let $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$ denote a dataset of action observations, where x_i is the observed sequence and $y_i \in \mathbb{R}$ is its reference quality score. Given a natural-language evaluation description τ, specification $\mathbb{V}$, and labeled training data $\mathcal{D}_{\mathrm{tr}} \subseteq \mathcal{D}$, our goal is to learn an admissible criterion structure together with the predictive and numerical components required for its execution.

Let Θ^* collect the fitted parameters of the source predictors and the base scorer, and let ψ^* denote the calibrated numerical parameters associated with the selected criterion structure. The learning problem is summarized as

$$(\tau, \mathbb{V}, \mathcal{D}_{\mathrm{tr}}) \longmapsto (\mathcal{Q}^*, \Theta^*, \psi^*), \qquad \mathcal{Q}^* \in \mathcal{A}_{\mathbb{V}}. \tag{3}$$

For an evaluation population indexed by $\mathcal{U}$ with $n = |\mathcal{U}|$ observations, let $\mathbf{X}_{\mathcal{U}}$ collect their inputs. The resulting assessor $F_{\mathcal{Q}^*, \Theta^*, \psi^*}$ produces

$$\hat{\mathbf{y}}_{\mathcal{U}} = F_{\mathcal{Q}^*, \Theta^*, \psi^*}(\mathbf{X}_{\mathcal{U}}) \in \mathbb{R}^n. \tag{4}$$

The selected criteria and all fitted components are fixed before evaluation. Evaluation labels are used only to measure agreement between predicted and reference quality scores.

3 Methodology

As illustrated in Figure 1, CriScore comprises three stages: scoring criteria induction, supervised rule calibration, and neuro-symbolic quality reasoning. Criteria induction constructs candidate scoring structures from domain evaluation knowledge and available training diagnostics. Supervised calibration trains the evidence predictors and selects a criterion structure together with its numerical realization. The resulting assessor executes the selected criteria with frozen parameters to produce quality scores and execution traces.

3.1 Scoring Criteria Induction

CriScore induces task-level scoring criteria by grounding domain evaluation knowledge in the evidence available to the assessor. Given an evaluation description τ, the specification $\mathbb{V}$, and training-side diagnostics H_{diag}, the induction stage determines an evaluation structure for subsequent scoring. Numerical choices that remain open are resolved through supervised calibration in Section 3.2.

Concept Identification and Evidence Binding. CriScore identifies evaluation concepts $\mathcal{C}$ from the task requirements encoded in τ and grounds them in the evidence sources declared by $\mathbb{V}$. Source descriptions characterize the information represented by each source, whereas H_{diag} reflects its empirical utility on the training data. These signals establish the evidence binding $\mathcal{B} \subseteq \mathcal{C} \times \mathcal{J}$, which associates each concept with its supporting evidence sources. A language model maps the evaluation description to this structured representation subject to the constraints of $\mathbb{V}$.

Relation Induction. Given $\mathcal{C}$ and $\mathcal{B}$, CriScore induces relations $\mathcal{R}$ that define how the selected evidence enters the scoring computation. Each relation invokes an operation permitted by $\mathbb{V}$, restricting the induced criteria to the predefined computation space.

Let ξ denote an accepted structured induction entry. If ξ specifies criterion fields, K_ξ indexes the fixed fields and t_ξ denotes their values. Alternatively, ξ may identify a registered criterion template $\mathcal{Q}_{\kappa_\xi} \in \mathcal{A}_{\mathbb{V}}$. The corresponding candidate family is

$$\mathcal{A}_\xi = \begin{cases} \{\mathcal{Q} \in \mathcal{A}_{\mathbb{V}} : \mathcal{Q}|_{K_\xi} = t_\xi\}, & \text{criterion fields,} \\ \{\mathcal{Q}_{\kappa_\xi}\}, & \text{template identifier.} \end{cases} \tag{5}$$

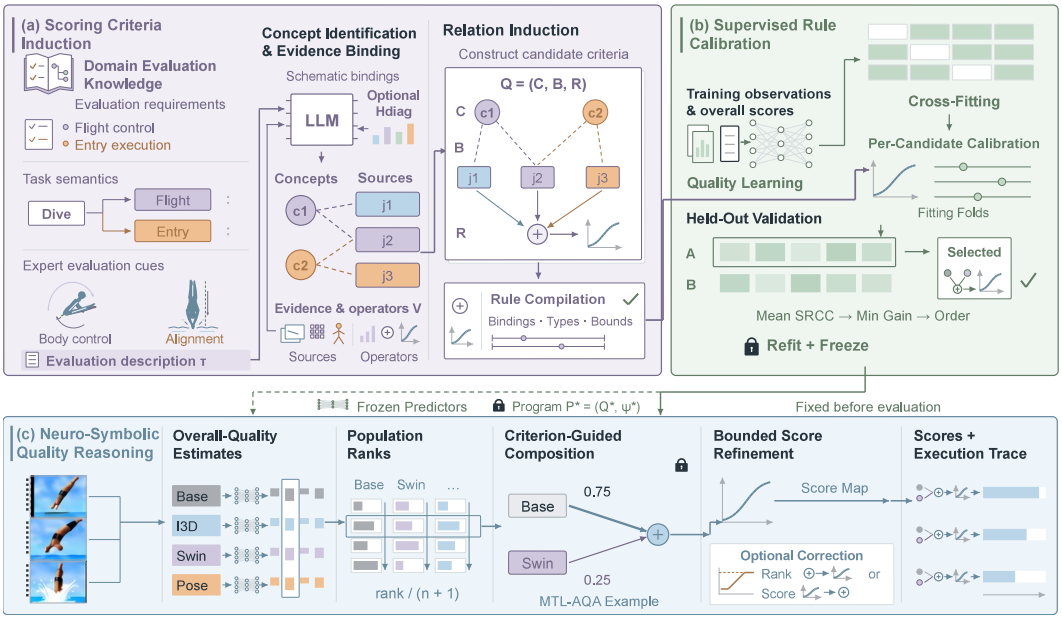


Figure 1: Overview of CriScore. (a) Knowledge-guided scoring criteria induction. (b) Score-supervised program selection and calibration. (c) Neuro-symbolic quality reasoning with the learned executable scoring program, illustrated with an MTL-AQA example.

Only fields affecting numerical execution constrain $\mathcal{A}_\xi$, while unspecified numerical choices remain available for supervised calibration. Explanatory content introduces no additional relations.

Rule Compilation. Compilation translates the candidate criteria in $\mathcal{A}_\xi$ into executable scoring structures. The compiler resolves each bound source set $\mathcal{J}_c$ and validates the operations specified by $\mathcal{R}$. For each relation $r \in \mathcal{R}$, the compiler verifies that $u_r \in \mathcal{X}_{o_r}$ and constrains its numerical parameters to Φ_{o_r}. The bindings and relations determine the active sources and composition operations, while numerical choices left open by the structure are supplied by ψ during calibration.

3.2 Supervised Rule Calibration

Score supervision serves two roles after criteria induction. It trains the predictors that provide numerical quality evidence and resolves the numerical choices left open within the induced candidate family $\mathcal{A}_\xi$. All fitting and selection are performed using the training partition $\mathcal{D}_{\mathrm{tr}}$.

Quality Evidence Learning. For each source $j \in \mathcal{J}$, the predictor h_{θ_j} estimates the overall quality from its source-specific input x_{ij}. Let $\mathcal{I}_{\mathrm{tr}}$ index the observations in $\mathcal{D}_{\mathrm{tr}}$. The predictor is fitted by

$$\mathcal{L}_j(\theta_j) = \frac{1}{|\mathcal{I}_{\mathrm{tr}}|} \sum_{i \in \mathcal{I}_{\mathrm{tr}}} \ell_j\big(h_{\theta_j}(x_{ij}), y_i\big) + \Omega_j(\theta_j), \tag{6}$$

where ℓ_j denotes the prediction loss and Ω_j represents the applicable regularization term. The resulting prediction $z_{ij} = h_{\theta_j}(x_{ij})$ provides numerical evidence for the bindings defined in $\mathcal{B}$. Its supervision target remains the overall quality score rather than a concept-specific label. A base scorer f_θ is trained under the same supervision and produces $s_i = f_\theta(x_i)$.

Cross-Fitted Validation. Program calibration uses predictions from observations excluded from the corresponding predictor fitting. Let $\mathcal{I}_1, \ldots, \mathcal{I}_L$ partition the training indices into inner validation folds. For fold ℓ, the source predictors are fitted on $I_{\mathrm{tr}} \setminus \mathcal{I}_\ell$. Their predictions on $\mathcal{I}_\ell$ form

$$\mathbf{Z}_\ell = \Big[h_{\theta_j^{(-\ell)}}(x_{ij})\Big]_{i \in \mathcal{I}_\ell,\, j \in \mathcal{J}} \in \mathbb{R}^{|\mathcal{I}_\ell| \times |\mathcal{J}|}. \tag{7}$$

The base scorer follows the same fold exclusion and yields predictions $\mathbf{s}_\ell$ on $\mathcal{I}_\ell$. Let $\widehat{\Theta}^{(-\ell)}$ collect the parameters of the source predictors and base scorer fitted on $\mathcal{I}_{-\ell}$. Let $\mathbf{y}_\ell = [y_i]_{i \in \mathcal{I}_\ell}$ denote the corresponding reference scores.

For each candidate structure $\mathcal{Q}_p$, the numerical components $\widehat{\psi}_p^{(-\ell)}$ are fitted or selected using only $\mathcal{I}_{-\ell}$. When score mappings or residual corrections are enabled, their fitting inputs are generated through cross-fitting within $\mathcal{I}_{-\ell}$. Any selection of open hyperparameters at this stage also uses only this fitting partition. The held-out fold $\mathcal{I}_\ell$ is used to evaluate the resulting candidate.

Program Calibration and Selection. Let $\mathcal{Q}_1, \ldots, \mathcal{Q}_K \in \mathcal{A}_\xi$ denote the candidate criterion structures. Each candidate remains fixed across validation folds, while its predictor parameters and numerical components are estimated from the corresponding fitting partitions. Executing candidate $\mathcal{Q}_p$ on fold ℓ produces

$$\hat{\mathbf{y}}_\ell^{\,p} = F_{\mathcal{Q}_p, \widehat{\Theta}^{(-\ell)}, \widehat{\psi}_p^{(-\ell)}}\left(\mathbf{X}_{\mathcal{I}_\ell}\right). \tag{8}$$

Its mean validation performance is

$$M_p = \frac{1}{L}\sum_{\ell=1}^{L} \rho_{p\ell}, \qquad \rho_{p\ell} = \rho_S\left(\mathbf{y}_\ell, \hat{\mathbf{y}}_\ell^{\,p}\right), \tag{9}$$

where ρ_S is Spearman correlation. As a secondary tie-breaking statistic, we define the minimum gain over the corresponding base predictions as

$$G_p = \min_{1\le \ell \le L} \Delta_{p\ell}, \qquad \Delta_{p\ell} = \rho_{p\ell} - \rho_S\left(\mathbf{y}_\ell, \mathbf{s}_\ell\right). \tag{10}$$

Selection prioritizes mean validation performance M_p, followed by the minimum fold gain G_p for equal means and the fixed candidate order for any remaining tie:

$$p^* = \arg\max_{1\le p\le K}{}_{\text{lex}}\big(M_p, G_p, -p\big), \qquad \mathcal{Q}^* = \mathcal{Q}_{p^*}. \tag{11}$$

The comparison uses unrounded statistics. Fields fixed during criteria induction remain unchanged throughout calibration.

Parameter Freezing. After selecting $\mathcal{Q}^*$, the source predictors and base scorer are refitted on the designated training observations to obtain Θ^*. The numerical parameters ψ^* are estimated from training-side cross-fitted predictions. The resulting program $\mathcal{P}^* = (\mathcal{Q}^*, \psi^*)$ and predictor parameters Θ^* are fixed before evaluation, with no subsequent adjustment using evaluation labels.

3.3 Neuro-Symbolic Quality Reasoning

The executor retrieves the quality evidence specified by $\mathcal{B}$ and applies the relations in $\mathcal{R}$ according to their dependencies. It records the active sources, parameters, and score updates.

Evidence Normalization. Rank-based programs place predictions on a common scale over an evaluation population indexed by $\mathcal{U}$ with $n = |\mathcal{U}|$ observations. With average ranks for ties, the source and base evidence are

$$e_{ij} = \frac{\text{rank}_{\mathcal{U}}(z_{ij})}{n+1}, \qquad b_i = \frac{\text{rank}_{\mathcal{U}}(s_i)}{n+1}. \tag{12}$$

Higher values indicate higher predicted quality within the population. Normalization depends on population predictions without accessing evaluation labels. Appendix D provides details on tie handling. The program and fitted parameters remain fixed, although the normalized evidence for an observation may vary with its evaluation population.

Criterion-Guided Composition. For program $\mathcal{P}_p = (\mathcal{Q}_p, \psi_p)$, the compiled structure determines the active source set $\mathcal{S}_p \subseteq \mathcal{J}$ and the composition operations. For the rank-based composition, the selected sources receive nonnegative priorities α_{pj} that sum to one. Temperature $T_p > 0$ controls their concentration, with $T_p = 1$ preserving the original priorities. The resulting weights w_{pj} and blend strength $\beta_p \in [0,1]$ define

$$w_{pj} = \frac{\alpha_{pj}^{1/T_p}}{\sum_{k\in\mathcal{S}_p} \alpha_{pk}^{1/T_p}}, \quad j \in \mathcal{S}_p, \qquad r_i^p = (1-\beta_p)b_i + \beta_p \sum_{j\in\mathcal{S}_p} w_{pj} e_{ij}. \tag{13}$$

Recorded source contributions support execution replay and controlled removal of selected inputs.

Bounded Score Refinement. Let g_p denote the fitted mapping from the composed evidence r_i^p to the task-specific score scale. Without further refinement, the prediction is $\hat{y}_i = g_p(r_i^p)$. A non-decreasing mapping does not reverse the ordering, although it may introduce ties. A registered residual correction may further adjust the resulting values and rankings. Its correction coordinate and activation rule are specified by the registered operator. For program $\mathcal{P}_p$, the correction is

$$\delta_i^p = \gamma_i^p \eta_p \operatorname{clip}(d_i^p, -B_p, B_p), \qquad |\delta_i^p| \le \gamma_i^p \eta_p B_p, \tag{14}$$

where d_i^p is the predicted correction, $B_p \ge 0$ its bound, $\eta_p \in [0,1]$ its strength, and $\gamma_i^p \in \{0,1\}$ its activation. The correction predictor uses the inputs declared by the registered operator and is trained in the coordinate of the correction. An inactive correction gives $\delta_i^p = 0$, while an unconditional correction sets $\gamma_i^p = 1$.

The final prediction depends on the declared correction coordinate:

$$\hat{y}_i = \begin{cases} g_p(r_i^p + \delta_i^p), & \text{rank-coordinate correction,} \\ g_p(r_i^p) + \delta_i^p, & \text{score-coordinate correction.} \end{cases} \tag{15}$$

The rank-coordinate branch requires g_p to be defined on the corrected inputs. The bound limits the update magnitude in the declared coordinate. Appendix D gives the corresponding pairwise ordering property. Complete evidence or a declared fallback ensures valid execution.

4 Experiments

In this section, we conduct extensive experiments to evaluate the effectiveness of CriScore. Our evaluation is organized around the following research questions:

- **RQ1:** How well does CriScore generalize quality assessment across diverse action domains?
- **RQ2:** What are the contributions of its key components to assessment performance?
- **RQ3:** How consistently do the induced programs perform across evaluation conditions?
- **RQ4:** How do the induced criteria shape individual quality predictions?
- **RQ5:** What computational overhead does CriScore introduce during preparation and assessment?

4.1 Experimental Setup

Datasets. We evaluate CriScore on four AQA datasets spanning individual diving, synchronized diving, and rehabilitation assessment: MTL-AQA (Parmar & Morris, 2019), FineDiving (Xu et al., 2022), TASD-2 (Gao et al., 2020), and UI-PRMD (Vakanski et al., 2018). Dataset statistics and available observations are detailed in Appendix A.1.

Baselines. We compare CriScore with 33 baselines across four families: feature-based regression, temporal modeling, graph-based motion modeling, and AQA-specific methods. The complete baseline set and implementation details are provided in Appendix A.2.

Protocol and Implementation. Model fitting, induction diagnostics, and supervised calibration use only training partitions, while evaluation follows dataset-specific held-out protocols. We evaluate quality ranking using SRCC and score prediction using NMSE, MAE, and RMSE. DeepSeek-V4-Pro is the default model for criteria induction. Evaluation protocols, metric definitions, and component controls are provided in Appendix B.

4.2 Quality Prediction (RQ1)

Table 1 compares CriScore with baselines across four AQA datasets and multiple evaluation metrics. Across the 33 baselines, CriScore achieves the best performance on all four datasets and evaluation metrics, with an average relative SRCC gain of 2.19% over the strongest baseline on each dataset. The improvement is consistent across different modeling paradigms, including feature-based regression, temporal and graph-based models, and methods specifically designed for AQA. Notably, CriScore maintains its advantage over recent AQA methods that incorporate relative-quality learning, structured action modeling, domain knowledge, or uncertainty modeling. The gains in SRCC are accompanied by lower score errors, indicating that the induced scoring criteria improve both quality ranking and numerical score prediction.

Table 1: Quality performance on four AQA datasets using SRCC (% ↑), NMSE (↓), MAE (↓), and RMSE (↓). Bold and underlined values indicate the best and second-best results, respectively.

Method	MTL-AQA				TASD-2				FineDiving				UI-PRMD			
	SRCC	NMSE	MAE	RMSE	SRCC	NMSE	MAE	RMSE	SRCC	NMSE	MAE	RMSE	SRCC	NMSE	MAE	RMSE
Feature-Based Regression																
Ridge	69.167	0.618	11.043	14.225	80.964	0.408	5.931	8.383	79.534	0.354	6.934	9.099	61.176	0.810	0.094	0.122
SVR	76.668	0.466	8.972	12.353	73.030	0.671	8.081	10.757	74.553	0.445	7.653	10.210	64.063	0.606	0.084	0.109
General Temporal Models																
ModernTCN	30.407	11.881	59.707	62.353	78.283	0.444	6.490	8.744	78.061	0.436	7.433	10.065	51.341	1.056	0.109	0.140
PatchTST	36.227	12.343	60.953	63.554	25.272	0.999	11.287	13.119	78.114	0.379	7.189	9.411	23.881	1.018	0.115	0.143
TimesNet	78.413	0.434	9.158	11.919	78.260	0.400	6.121	8.300	79.690	0.358	6.900	9.144	59.792	0.679	0.088	0.118
iTransformer	34.116	12.341	60.949	63.550	13.183	1.020	11.408	13.257	78.754	0.374	7.137	9.349	43.533	0.950	0.105	0.137
TimeMixer	36.115	8.618	50.063	53.105	75.846	0.511	7.209	9.386	79.631	0.349	6.770	9.032	46.845	0.775	0.094	0.125
MambaTS	25.146	7.320	45.661	48.943	76.166	0.474	6.565	9.042	75.824	0.414	7.215	9.837	45.211	0.985	0.113	0.137
Graph-Based Motion Models																
ST-GCN	15.289	4.691	35.335	39.183	82.353	0.372	5.803	8.004	78.231	0.424	6.942	9.314	25.285	1.826	0.153	0.192
CTR-GCN	18.310	11.096	57.530	60.259	81.134	0.423	6.007	8.540	79.115	0.385	6.817	9.141	24.507	1.848	0.154	0.191
EfficientGCN	21.343	13.153	63.086	65.608	82.701	0.399	6.110	8.294	78.892	0.398	6.854	9.215	58.285	0.805	0.097	0.128
InfoGCN	16.007	7.984	47.967	51.117	78.590	0.799	8.428	11.737	80.108	0.364	6.741	8.992	57.999	0.629	0.088	0.113
HD-GCN	38.467	12.971	62.616	65.152	77.673	0.440	6.418	8.711	79.841	0.371	6.788	9.014	25.792	1.588	0.143	0.177
AQA-Specific Methods																
MUSDL	82.532	0.330	7.488	10.398	79.304	0.432	5.357	8.633	78.455	0.406	6.912	9.281	48.657	0.854	0.101	0.132
CoRe	84.807	0.301	7.297	9.922	81.076	0.405	6.222	8.355	78.908	0.478	8.007	10.559	32.381	1.391	0.128	0.169
T2CR	80.595	0.477	9.596	12.499	85.198	0.338	5.733	7.634	80.652	0.345	6.708	8.854	7.085	1.117	0.118	0.150
Uni-FineParser	79.408	0.377	8.298	11.102	76.298	0.515	6.483	9.420	80.787	0.342	6.671	8.824	45.898	0.884	0.104	0.129
PHI	82.277	0.377	8.381	11.100	64.407	0.695	8.167	10.943	81.051	0.331	6.649	8.795	54.322	0.752	0.084	0.125
FineCausal	80.597	0.351	7.941	10.714	79.714	0.448	6.236	8.782	80.344	0.354	6.712	8.918	65.432	0.655	0.087	0.116
UIL-AQA	80.772	0.342	7.644	10.575	77.672	0.491	6.265	9.201	79.568	0.384	6.801	9.082	52.125	0.761	0.091	0.124
RICA^2	81.473	0.399	8.501	11.434	80.042	0.444	6.107	8.746	81.012	0.337	6.658	8.784	71.241	0.514	0.077	0.105
CaFlow	80.591	0.576	10.929	13.727	79.669	0.450	6.016	8.809	70.991	0.511	8.321	10.792	54.921	0.722	0.085	0.122
MCMoE	81.334	0.351	7.970	10.718	76.065	0.485	6.775	9.143	79.402	0.386	6.829	9.117	55.679	0.742	0.092	0.121
DAE	76.435	0.587	10.576	13.864	81.157	0.405	6.223	8.355	79.458	0.354	6.873	9.095	47.039	0.883	0.102	0.133
FineParser	78.052	0.407	8.825	11.541	76.288	0.466	6.555	8.966	79.924	0.374	6.753	9.021	42.344	0.912	0.108	0.135
HGCN-AQA	77.421	0.717	12.145	15.316	71.652	0.520	7.338	9.464	77.900	0.459	7.765	10.260	39.814	1.185	0.117	0.152
ASGTN	80.886	0.412	8.958	11.606	78.950	0.515	6.780	9.423	77.104	0.463	7.848	10.346	43.237	0.791	0.092	0.130
HP-MCoRe	81.160	0.536	10.334	13.245	70.906	0.659	7.892	10.656	80.895	0.339	6.662	8.811	68.958	0.584	0.081	0.109
MAGR	80.650	0.395	8.325	11.370	81.232	0.365	5.915	7.933	77.516	0.415	7.615	9.841	3.265	1.129	0.123	0.154
MAGR++	81.041	0.358	8.122	10.826	80.478	0.355	5.684	7.824	77.859	0.428	6.961	9.351	1.550	1.186	0.125	0.152
TPT	81.268	0.386	8.421	11.233	78.188	0.487	6.145	9.164	78.028	0.514	8.382	10.925	54.896	0.948	0.088	0.133
BriMA	80.640	0.483	9.442	12.575	75.968	0.487	6.885	9.160	78.954	0.395	6.842	9.191	54.232	0.757	0.093	0.126
PAMFN	79.562	0.366	8.112	10.939	74.357	0.510	6.969	9.376	79.108	0.391	6.837	9.168	56.897	0.724	0.088	0.119
CriScore	**86.115**	**0.276**	**6.729**	**9.511**	**86.964**	**0.312**	**5.110**	**7.331**	**81.337**	**0.329**	**6.630**	**8.766**	**74.667**	**0.422**	**0.076**	**0.099**

4.3 Contributions of Criteria Induction and Quality Reasoning (RQ2)

Table 2 examines criteria induction and quality reasoning while holding the underlying evidence predictions fixed. Removing semantic or diagnostic information consistently degrades performance, with larger losses from shuffled evidence bindings than from uniform weights, highlighting the importance of identifying and organizing relevant evidence. Likewise, replacing structured reasoning with weighted fusion or removing scoring relations reduces performance, while individual operations have smaller and more task-dependent effects. These results show that CriScore benefits primarily from the induced evidence structure and its composition during quality reasoning.

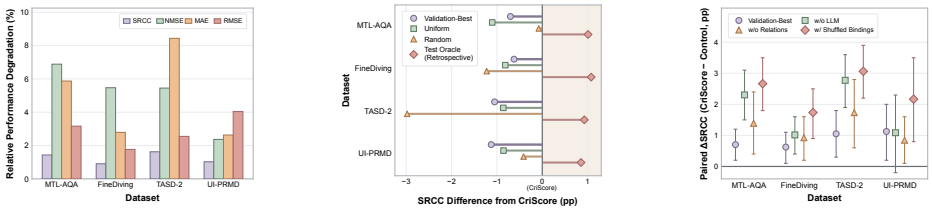
Figure 2a further shows consistent degradation across ranking and score-accuracy measures when supervised calibration is removed. Together, the results support complementary contributions from criteria induction, neuro-symbolic quality reasoning, and supervised calibration.

4.4 Program Selection and Robustness (RQ3)

Figures 2b and 2c evaluate program choice while holding the underlying evidence predictions fixed. Figure 2b shows that the selected programs consistently outperform random program references, indicating that the gains cannot be attributed solely to the learned evidence predictors. Figure 2c further examines whether these differences persist across evaluation units. Eleven of the twelve paired 95% confidence intervals exclude zero, showing that the advantage is generally stable across data partitions and random seeds. The only exception occurs on UI-PRMD for the w/o LLM control, where the mean difference remains positive but the interval crosses zero. Detailed program controls

Table 2: Criteria induction and quality-reasoning controls under fixed evidence predictions. Values denote SRCC decrease (pp) relative to CriScore.

Dataset	Criteria Induction						Quality Reasoning							
	No Diag.	No Sem.	No LLM	No S.+D.	Equal W.	Shuffle B.	Fusion	No Rel.	No Corr.	No Constr.	No Fall.	Alt. Comp.	No Cal.	No Exec.
MTL-AQA	1.995	1.165	2.305	2.963	1.015	2.665	1.444	1.391	0.663	0.224	0.061	1.065	1.235	1.525
TASD-2	2.114	1.844	2.774	3.450	0.954	3.064	1.812	1.733	—	—	—	1.164	1.414	1.944
FineDiving	0.827	0.887	1.017	1.906	0.587	1.737	0.992	0.925	0.213	0.080	0.026	0.487	0.737	1.107
UI-PRMD	0.847	0.717	1.087	1.842	0.557	2.167	0.953	0.846	0.292	0.155	—	0.517	0.767	0.977



(a) Calibration Effect (b) Program Selection (c) Program Robustness

Figure 2: Analysis of supervised calibration, program selection, and robustness. (a) Relative performance degradation without supervised calibration. (b) SRCC differences between CriScore and alternative program-selection strategies under fixed evidence predictions. (c) Mean paired SRCC gains over program controls across evaluation units, with 95% CIs.

and paired statistics are provided in Appendices C.2 and C.3. Additional induction-model sensitivity and output-validity analyses are reported in Appendix C.4.

4.5 Criteria-Guided Quality Reasoning (RQ4)

Figure 3 traces how the induced criteria produce an individual quality prediction through a numerical replay on TASD-2. For the illustrated observation, the selected program combines evidence from the base scorer, I3D, and VideoMAE without post-composition correction. Criterion-guided composition yields a normalized rank of 0.47826, which is subsequently mapped to a predicted score of 56.70 against a reference score of 57.60. The replay thus makes the path from selected evidence to the final prediction explicit.

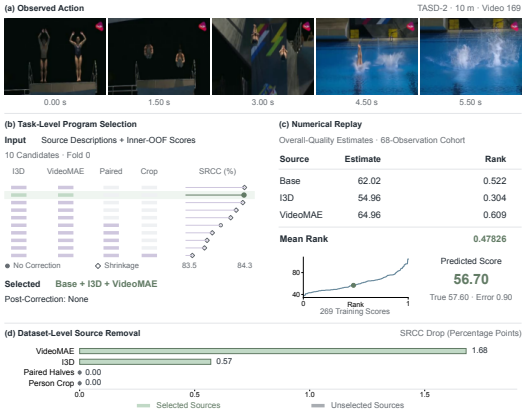
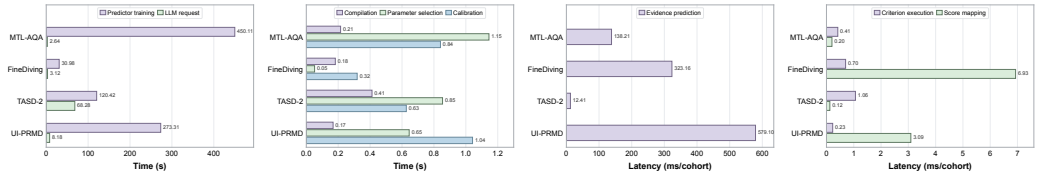


Figure 3: The TASD-2 replay connects observations, selection, scoring, and dataset-level source removal. Ranks use 68 validation samples and score mapping uses 269 training labels.

Source removal further evaluates whether the selected evidence materially contributes to assessment performance. Removing VideoMAE and I3D reduces dataset-level SRCC by 1.68 and 0.57 percentage points, respectively, whereas removing either unselected source has no effect. This consistency between criterion selection and source-removal effects shows that the reasoning trace reflects the evidence contributing to the resulting scores. Appendix C.5 extends the evidence-deletion analysis to all four tasks.

4.6 Preparation and Scoring Efficiency (RQ5)

Figure 4 separates one-time task preparation from repeated scoring with frozen criteria. Preparation is dominated by predictor training, while criteria induction and subsequent compilation, selection, and calibration introduce comparatively smaller costs. Although the induction cost varies across tasks, it is incurred only during preparation and does not enter the repeated scoring path. After freezing, the computational cost shifts primarily to evidence prediction. Neuro-symbolic quality



(a) Major Preparation Costs (b) Minor Preparation Costs (c) Evidence Prediction (d) Quality Reasoning

Figure 4: Computational efficiency of CriScore. (a)-(b) report task-level preparation costs, while panels (c)-(d) report repeated scoring latency per cohort. Separate scales are used to distinguish major and smaller computational stages.

reasoning itself remains lightweight, with criterion execution requiring only 0.228–1.062 ms per cohort. Together with score mapping, the frozen assessor processes 509.729–5,100.956 samples/s from cached inputs. These results show that the main computational cost is concentrated in task-level preparation, while the induced criteria can be reused efficiently for subsequent assessment. Detailed timing protocols and measured stages are provided in Appendix C.6.

5 Related Work

Action Quality Assessment. Learning-based AQA methods infer quality from annotated performances through learned representations and scoring functions. HP-MCoRe (Qi et al., 2025) captures quality differences through stage-wise contrastive comparisons, while Uni-FineParser (Xu et al., 2025) models fine-grained action structure through human-centric spatial and temporal parsing. Another line of work incorporates domain knowledge through predefined evaluation structures. NSAQA (Okamoto & Parmar, 2024) executes hierarchical expert rules over action symbols, FineCausal (Han et al., 2025) introduces expert-specified causal structures, and FineGrade (Li & Leung, 2026) incorporates official scoring rules into quality prediction. These approaches either encode scoring logic implicitly in learned mappings or rely on evaluation structures specified in advance. CriScore instead induces executable scoring criteria from domain evaluation knowledge and adapts their numerical realization through overall-score supervision.

Neuro-Symbolic Reasoning. Neuro-symbolic approaches combine learned representations with explicit programs to support structured and executable reasoning. VPD (Hu et al., 2024) executes and verifies candidate programs before distilling their reasoning into vision-language models, while VADAR (Marsili et al., 2025) synthesizes functions and programs for compositional 3D spatial reasoning. Vision-Language Programs (Wüst et al., 2025) compile structured visual descriptions into neuro-symbolic programs, while Visual Set Program Synthesizer (Cheng et al., 2026) separates symbolic program generation from grounded execution for filtering, comparison, and aggregation. SYNAPSE (Modak et al., 2025) combines visual parsing, language models, and program synthesis to learn user preferences from limited visual demonstrations. CriScore extends neuro-symbolic reasoning to quality assessment by inducing task-level scoring criteria within a predefined evidence and operation space, grounded and calibrated by overall-score supervision.

6 Conclusion

In this paper, we addressed the action quality assessment (AQA) problem by introducing CriScore, a neuro-symbolic framework for learning executable scoring criteria from domain evaluation knowledge and score supervision. CriScore represents scoring criteria as executable task-level programs, with domain knowledge guiding candidate construction within a predefined evidence and operation space and overall-score supervision supporting program selection and numerical calibration. The resulting program is frozen before evaluation and reused across new observations through traceable neuro-symbolic quality reasoning. Extensive experiments across four AQA datasets demonstrate the effectiveness of CriScore, with an average relative SRCC gain of 2.19% over the strongest baselines. Component and fixed-evidence analyses further support the roles of criteria induction, calibration, and program selection. Overall, these results support treating scoring criteria as learnable structures that connect domain knowledge with data-driven quality assessment.

AI USE STATEMENT

In this work, we used generative AI tools for manuscript language polishing, implementing basic utility functions, and debugging code. We did not use these tools to assist with theoretical development, mathematical claims or proofs, hypothesis formulation, research design, experimental design, or result interpretation. Synthetic data generation and qualitative or thematic data analysis were not applicable to this work. The authors reviewed all AI-assisted text for factual and technical accuracy and manually inspected and tested all AI-assisted code changes for correctness. We take responsibility for the final content of this work, including all text, claims, code, and other artifacts produced with the aid of generative AI.

REFERENCES

Yang Bai, Desen Zhou, Songyang Zhang, Jian Wang, Errui Ding, Yu Guan, Yang Long, and Jingdong Wang. Action quality assessment with temporal parsing transformer. In *European Conference on Computer Vision*, pp. 422–438, 2022.

Xiuding Cai, Xueyao Wang, Yaoyao Zhu, and Yu Yao. MambaTS: Improved selective state space models for long-term time series forecasting. *Pattern Recognition*, 2026. URL https://arxiv.org/abs/2405.16440.

Yuxin Chen, Ziqi Zhang, Chunfeng Yuan, Bing Li, Ying Deng, and Weiming Hu. Channel-wise topology refinement graph convolution for skeleton-based action recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021.

Zehua Cheng, Wei Dai, Wenhu Zhang, Thomas Lukasiewicz, and Jiahao Sun. Visual set program synthesizer. In *IEEE International Conference on Multimedia and Expo*, 2026. URL https://arxiv.org/abs/2603.15997.

Hyung-Gun Chi, Myoung Hoon Ha, Seunggeun Chi, Sang Wan Lee, Qixing Huang, and Karthik Ramani. InfoGCN: Representation learning for human skeleton-based action recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

Xu Dong, Xinran Liu, Wanqing Li, Anthony Adeyemi-Ejeye, and Andrew Gilbert. UIL-AQA: Uncertainty-aware clip-level interpretable action quality assessment. *International Journal of Computer Vision*, 134(1):24, 2026. doi: 10.1007/s11263-025-02638-6.

Jibin Gao, Wei-Shi Zheng, Jia-Hui Pan, Chengying Gao, Yaowei Wang, Wei Zeng, and Jianhuang Lai. An asymmetric modeling for action assessment. In *European Conference on Computer Vision*, pp. 222–238, 2020.

Ruisheng Han, Kanglei Zhou, Amir Atapour-Abarghouei, Xiaohui Liang, and Hubert P. H. Shum. FineCausal: A causal-based framework for interpretable fine-grained action quality assessment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 6064–6073, 2025.

Ruisheng Han, Kanglei Zhou, Shuang Chen, Amir Atapour-Abarghouei, and Hubert P. H. Shum. CaFlow: Enhancing long-term action quality assessment with causal counterfactual flow. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 8231–8241, 2026. doi: 10.1109/WACV61042.2026.00794.

Yushi Hu, Otilia Stretcu, Chun-Ta Lu, Krishnamurthy Viswanathan, Kenji Hata, Enming Luo, Ranjay Krishna, and Ariel Fuxman. Visual program distillation: Distilling tools and programmatic reasoning into vision-language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9590–9601, 2024.

Ali Ismail-Fawaz, Maxime Devanne, Stefano Berretti, Jonathan Weber, and Germain Forestier. A standardized benchmark for skeleton-based rehabilitation assessment using deep learning. In *IEEE International Conference on Automatic Face and Gesture Recognition (FG)*, pp. 1–9, 2026. URL https://arxiv.org/abs/2507.21018.

Xiao Ke, Huangbiao Xu, Xiaofeng Lin, and Wenzhong Guo. Two-path target-aware contrastive regression for action quality assessment. *Information Sciences*, 664:120347, 2024. doi: 10.1016/j.ins.2024.120347.

Jungho Lee, Minhyeok Lee, Dogyoon Lee, and Sangyoun Lee. Hierarchically decomposed graph convolutional networks for skeleton-based action recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023.

Yicong Li and Howard Leung. FineGrade: A rule-consistent scoring framework for fine-grained action quality assessment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Findings*, pp. 8258–8267, 2026.

Jiang Liu, Huasheng Wang, Wei Zhou, Katarzyna Stawarz, Padraig Corcoran, Ying Chen, and Hantao Liu. Adaptive spatiotemporal graph transformer network for action quality assessment. *IEEE Transactions on Circuits and Systems for Video Technology*, 35(7):6628–6639, 2025. doi: 10.1109/TCSVT.2025.3541456.

Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. iTransformer: Inverted transformers are effective for time series forecasting. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=JePfAI8fah.

Donghao Luo and Xue Wang. ModernTCN: A modern pure convolution structure for general time series analysis. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=vpJMJerXHU.

Abrar Majeedi, Viswanatha Reddy Gajjala, Satya Sai Srinath Namburi, and Yin Li. RICA^2: Rubric-informed, calibrated assessment of actions. In *European Conference on Computer Vision*, 2024.

Damiano Marsili, Rohun Agrawal, Yisong Yue, and Georgia Gkioxari. Visual agentic ai for spatial reasoning with a dynamic api. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 19446–19455, 2025.

Sadanand Modak, Noah Tobias Patton, Isil Dillig, and Joydeep Biswas. SYNAPSE: SYmbolic neural-aided preference synthesis engine. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 27529–27537, 2025. doi: 10.1609/aaai.v39i26.34965.

Yuqi Nie, Nam H. Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=Jbdc0vTOcol.

Lauren Okamoto and Paritosh Parmar. Hierarchical neurosymbolic approach for comprehensive and explainable action quality assessment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 3204–3213, 2024.

Paritosh Parmar and Brendan Tran Morris. What and how well you performed? a multitask learning approach to action quality assessment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019.

Mengshi Qi, Hao Ye, Jiaxuan Peng, and Huadong Ma. Action quality assessment via hierarchical pose-guided multi-stage contrastive regression. *IEEE Transactions on Image Processing*, 34:6461–6474, 2025. doi: 10.1109/TIP.2025.3613952.

Yi-Fan Song, Zhang Zhang, Caifeng Shan, and Liang Wang. Constructing stronger and faster baselines for skeleton-based action recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

Yansong Tang, Zanlin Ni, Jiahuan Zhou, Danyang Zhang, Jiwen Lu, Ying Wu, and Jie Zhou. Uncertainty-aware score distribution learning for action quality assessment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9839–9848, 2020.

Aleksandar Vakanski, Hyung-pil Jun, David Paul, and Russell Baker. A data set of human body movements for physical rehabilitation exercises. *Data*, 3(1):2, 2018. doi: 10.3390/data3010002.

Shiyu Wang, Haixu Wu, Xiaoming Shi, Tengge Hu, Huakun Luo, Lintao Ma, James Y. Zhang, and Jun Zhou. TimeMixer: Decomposable multiscale mixing for time series forecasting. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=7oLshfEIC2.

Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. TimesNet: Temporal 2d-variation modeling for general time series analysis. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=ju_Uqw384Oq.

Antonia Wüst, Wolfgang Stammer, Hikaru Shindo, Lukas Helff, Devendra Singh Dhami, and Kristian Kersting. Synthesizing visual concepts as vision-language programs. *arXiv preprint arXiv:2511.18964*, 2025. URL https://arxiv.org/abs/2511.18964.

Huangbiao Xu, Huanqi Wu, Xiao Ke, Junyi Wu, Rui Xu, and Jinglin Xu. MCMoE: Completing missing modalities with mixture of experts for incomplete multimodal action quality assessment. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2026. doi: 10.1609/aaai.v40i13.38104.

Jinglin Xu, Yongming Rao, Xumin Yu, Guangyi Chen, Jie Zhou, and Jiwen Lu. FineDiving: A fine-grained dataset for procedure-aware action quality assessment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

Jinglin Xu, Sibo Yin, Guohao Zhao, Zishuo Wang, and Yuxin Peng. FineParser: A fine-grained spatio-temporal action parser for human-centric action quality assessment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14628–14637, 2024.

Jinglin Xu, Sibo Yin, and Yuxin Peng. Human-centric fine-grained action quality assessment. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(8):6242–6255, 2025. doi: 10.1109/TPAMI.2025.3556935.

Sijie Yan, Yuanjun Xiong, and Dahua Lin. Spatial temporal graph convolutional networks for skeleton-based action recognition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2018.

Xumin Yu, Yongming Rao, Wenliang Zhao, Jiwen Lu, and Jie Zhou. Group-aware contrastive regression for action quality assessment. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021.

Ling-An Zeng and Wei-Shi Zheng. Multimodal action quality assessment. *IEEE Transactions on Image Processing*, 33:1600–1613, 2024. doi: 10.1109/TIP.2024.3362135.

Boyu Zhang, Jiayuan Chen, Yinfei Xu, Hui Zhang, Xu Yang, and Xin Geng. Auto-encoding score distribution regression for action quality assessment. *Neural Computing and Applications*, 36(2):929–942, 2024. doi: 10.1007/s00521-023-09068-w.

Kanglei Zhou, Yue Ma, Hubert P. H. Shum, and Xiaohui Liang. Hierarchical graph convolutional networks for action quality assessment. *IEEE Transactions on Circuits and Systems for Video Technology*, 2023. doi: 10.1109/TCSVT.2023.3281413.

Kanglei Zhou, Liyuan Wang, Xingxing Zhang, Hubert P. H. Shum, Frederick W. B. Li, Jianguo Li, and Xiaohui Liang. MAGR: Manifold-aligned graph regularization for continual action quality assessment. In *European Conference on Computer Vision*, pp. 375–392, 2024.

Kanglei Zhou, Qingyi Pan, Xingxing Zhang, Hubert P. H. Shum, Frederick W. B. Li, Xiaohui Liang, and Liyuan Wang. Continual action quality assessment via adaptive manifold-aligned graph regularization. *arXiv preprint arXiv:2510.06842*, 2025a.

Kanglei Zhou, Hubert P. H. Shum, Frederick W. B. Li, Xingxing Zhang, and Xiaohui Liang. PHI: Bridging domain shift in long-term action quality assessment via progressive hierarchical instruction. *IEEE Transactions on Image Processing*, 2025b.

Kanglei Zhou, Ruizhi Cai, Liyuan Wang, Hubert P. H. Shum, and Xiaohui Liang. A comprehensive survey of action quality assessment: Method and benchmark. *Pattern Recognition*, pp. 113933, 2026a. doi: 10.1016/j.patcog.2026.113933.

Kanglei Zhou, Chang Li, Qingyi Pan, and Liyuan Wang. BriMA: Bridged modality adaptation for multi-modal continual action quality assessment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026b.

A Experimental Details

A.1 Dataset Details

The datasets provide video or motion observations for assessing individual sports, synchronized sports, and rehabilitation movements. Their recorded observations and annotations are described below. Appendix B.1 specifies the data partitions used for evaluation.

MTL-AQA. MTL-AQA contains 1,412 competitive diving videos collected from 16 international events (Parmar & Morris, 2019). It covers individual and synchronized springboard and platform events, with varied viewpoints and backgrounds. Final quality scores provide supervision, while action classes, difficulty degrees, and commentary describe the performed actions.

TASD-2. TASD-2 provides frontal-view synchronized-diving sequences with annotations for execution, synchronization, and final quality scores (Gao et al., 2020). These paired performances support quality assessment of action execution and coordination between athletes.

FineDiving. FineDiving contains 3,000 diving videos covering 52 action types, 29 sub-action types, and 23 difficulty degrees (Xu et al., 2022). Procedure annotations describe each dive as consecutive action steps, alongside the quality scores used for supervision.

UI-PRMD. UI-PRMD records ten healthy subjects performing ten rehabilitation exercises in correct and incorrect manners using Vicon and Kinect (Vakanski et al., 2018). The observations contain full-body joint positions and angles. For continuous quality regression, we use the segmented UI-PRMD_REG datasets released in Rehab-Pile, which associate each motion sequence with a continuous target $y \in [0, 1]$ (Ismail-Fawaz et al., 2026).

A.2 Baseline Methods

We compare four baseline families using the data partitions and evaluation populations defined in Appendix B.1. Each cited method is adapted to the registered inputs and scalar quality targets, while retaining the corresponding source modules, prediction heads, and four-metric aggregates for comparison. FineDiving uses 32-step cached sequences with six splash-mask descriptors and 54 metadata features; UI-PRMD uses 66 joint-coordinate values and ten action indicators over the same sequence length. CaFlow and PHI retain the released objectives, detach operations, and score heads; the flow branches do not contribute to the score-head predictions used in the reported comparisons.

Feature-based regression.

- Ridge fits L_2-regularized linear regression to aggregate sequence statistics. The fixed-dimensional input provides a stable linear reference for quality prediction.
- SVR models nonlinear feature–score relations with a radial-basis kernel. Training selects regularization and kernel parameters, which are frozen for held-out prediction.

General temporal models.

- PatchTST (Nie et al., 2023) divides cached sequences into temporal patches, uses a shared Transformer across channels, and pools the resulting representation for regression.
- TimesNet (Wu et al., 2023) selects dominant periods and folds sequences into multiscale two-dimensional patterns. Its FineDiving and UI-PRMD adapters retain FFT-based period selection and amplitude-weighted Inception blocks, followed by a quality head.
- iTransformer (Liu et al., 2024) embeds temporal information within variable tokens and applies attention across feature dimensions. Its pooled output predicts action quality.
- TimeMixer (Wang et al., 2024) decomposes observations across temporal scales, exchanges information across resolutions, and aggregates the mixed representation over time.
- ModernTCN (Luo & Wang, 2024) models temporal and channel interactions with large-kernel convolutions. FineDiving and UI-PRMD retain large/small-kernel branches and grouped variable and channel feed-forward blocks, replacing forecasting with regression.
- MambaTS (Cai et al., 2026) uses selective state-space layers for long-range propagation with linear sequence complexity, then pools their temporal representations for regression.

Graph-based motion models.

- ST-GCN (Yan et al., 2018) combines spatial graph convolution over joint nodes with temporal convolution, globally pooling motion features for quality regression.
- CTR-GCN (Chen et al., 2021) refines the physical skeleton graph with motion-dependent, channel-specific joint relations, then aggregates the adaptive features across time.
- EfficientGCN (Song et al., 2022) applies efficient graph operations and staged fusion to multiple skeleton streams, using the fused motion descriptor for quality regression.
- InfoGCN (Chi et al., 2022) encodes skeletal motion with an information-bottleneck objective, pooling its features for optimization against continuous quality labels.
- HD-GCN (Lee et al., 2023) partitions the skeleton into hierarchical graph subsets, aggregates their structural features, and temporally pools the result for regression.

AQA-specific methods.

- DAE (Zhang et al., 2024) learns and samples from a quality-score distribution. Its distributional output is converted to an expectation for the four-metric comparison.
- MUSDL (Tang et al., 2020) learns a discrete score distribution with uncertainty-aware supervision and predicts its expectation on the dataset-specific quality scale.
- CoRe (Yu et al., 2021) compares examples grouped by score proximity and reconstructs predictions from references. Its FineDiving and UI-PRMD regression trees predict differences from pooled cached features and combine ten training-reference votes.
- T2CR (Ke et al., 2024) combines two target-aware representations for relative-quality contrast and numerical regression, using the registered partition for prediction.
- TPT (Bai et al., 2022) learns temporal part queries attending to quality-relevant action phases. The part representations support contrastive learning and are aggregated for regression.
- FineParser (Xu et al., 2024) parses human-centered spatial and temporal cues into structured action features, aggregating them across the procedure for quality regression.
- HGCN-AQA (Zhou et al., 2023) propagates quality information through clip- and scene-level graphs, fusing their hierarchical features for continuous score regression.
- Uni-FineParser (Xu et al., 2025) unifies human-centered temporal parsing and quality prediction across tasks, assessing observations from the registered temporal inputs.
- PHI (Zhou et al., 2025b) organizes evidence with hierarchical instructions, retaining the released instruction objective and score head for the available registered inputs.
- ASGTN (Liu et al., 2025) combines an adaptive space-time graph and a Transformer for local joint relations and longer temporal dependencies, pooling their features for regression.
- FineCausal (Han et al., 2025) separates action evidence from potentially spurious contextual factors, feeding this action-focused representation to the released score predictor.
- UIL-AQA (Dong et al., 2026) models clip-level uncertainty, separates difficulty from execution quality, and combines both factors in its head for calibrated assessment.
- HP-MCoRe (Qi et al., 2025) builds pose-guided, multistage relative-quality comparisons, aggregating stage-level contrastive features to reconstruct an absolute score.
- RICA^2 (Majeedi et al., 2024) aligns visual evidence with task-specific rubric dimensions describing observable action aspects, then combines criterion estimates for assessment.
- CaFlow (Han et al., 2026) models quality-related temporal variation with counterfactual regularization and a bidirectional time-conditioned flow. The adapter retains the released objective and score head; flow branches do not alter the reported inference score.
- MAGR (Zhou et al., 2024) aligns replayed and current features and regularizes a score-aware graph. Each outer fitting set has five action-ordered stages and replay capacity 50.
- MAGR++ (Zhou et al., 2025a) adds manifold alignment to replay-based graph learning. Like the MAGR adapter, it uses five stages, a temporal descriptor encoder, replay capacity 50, and one stochastic score draw for each observation, with its random seed held fixed.

- MCMoE (Xu et al., 2026) reconstructs unavailable modality information and combines modality-specific expert predictions, using a quality head for available inputs.
- BriMA (Zhou et al., 2026b) couples modality recovery with modality-aware replay under changing availability, fusing both representations for quality regression.
- PAMFN (Zeng & Zheng, 2024) learns sample-dependent RGB, optical-flow, and audio fusion weights, routing available modality features through its quality-prediction network.

A.3 Computational Environment

Server configurations. Experiments are run on a server with Ubuntu 22.04.5 LTS, two Intel Xeon Gold 5318Y CPUs (48 physical cores in total), and 256 GiB of system memory. The server is equipped with three NVIDIA A800 GPUs with 80 GB of memory each and one NVIDIA RTX A6000 GPU with 48 GB.

Execution environments. We use Python 3.10.12, NumPy 1.26.4, SciPy 1.11.4, and scikit-learn 1.4.2. CPU-based experiments share the server resources, whereas each GPU-based experiment is assigned a single accelerator.

B Evaluation and Program Controls

B.1 Data Splits and Evaluation Populations

All four datasets use an 8:2 train/test split. Predictor fitting, program selection, calibration, and any induction diagnostics are performed within the training partition. Numerical hyperparameters are selected through training-side validation. The frozen assessor is evaluated on the held-out test partition without using test labels for fitting or selection.

B.2 Evaluation Metrics

For an evaluation population $\mathcal{U}$ with $n = |\mathcal{U}|$, let $\mathbf{y} = (y_i)_{i\in\mathcal{U}}$ and $\widehat{\mathbf{y}} = (\widehat{y}_i)_{i\in\mathcal{U}}$ denote the reference and predicted scores. With $\bar{y} = n^{-1}\sum_{i\in\mathcal{U}} y_i$, the four evaluation metrics are

$$\begin{aligned} \mathrm{SRCC} &= 100\ \mathrm{Corr}(\mathrm{rank}(\mathbf{y}), \mathrm{rank}(\widehat{\mathbf{y}})), \quad \mathrm{NMSE} = \frac{\sum_{i\in\mathcal{U}}(y_i - \widehat{y}_i)^2}{\sum_{i\in\mathcal{U}}(y_i - \bar{y})^2}, \\ \mathrm{MAE} &= \frac{1}{n}\sum_{i\in\mathcal{U}} |y_i - \widehat{y}_i|, \qquad\qquad \mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{i\in\mathcal{U}}(y_i - \widehat{y}_i)^2}. \end{aligned} \tag{16}$$

Here, $\mathrm{rank}(\cdot)$ assigns average ranks to ties, and $\mathrm{Corr}(\cdot,\cdot)$ denotes Pearson correlation. Score errors are evaluated after applying each prediction set's frozen score reconstruction. Relative SRCC gain is computed against the strongest baseline on each dataset and then averaged equally across the four datasets.

B.3 Component Controls

The language controls remove language-guided choices while retaining each task's evidence bank. The execution controls bypass the compiled program and return the designated base prediction, removing criterion-guided composition and associated calibration together. Both controls are evaluated using the same held-out protocol as CriScore.

C Extended Experimental Results

C.1 Non-language Comparisons and Intervention Scope

Shared inputs and search budget. Table 3 compares Uniform Fusion, Optimized Weighted Fusion, Non-language Program Selection, and CriScore using the same evidence predictions, training-side validation folds, and score-mapping capacity. Uniform Fusion averages all available rank-

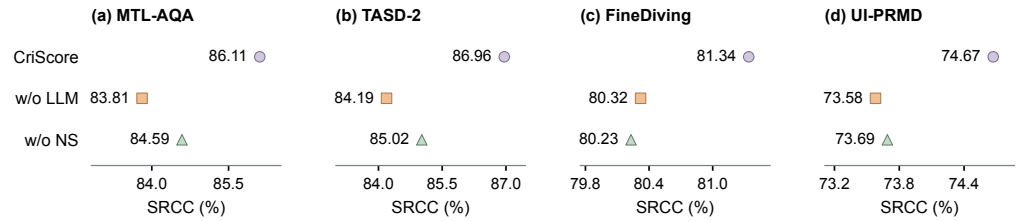


Figure 5: SRCC for CriScore, removal of language induction, and removal of neuro-symbolic execution under the same evidence bank within each assessment task.

normalized predictions and serves as a fixed reference. The three search-based methods use a common upper limit of $K = 64$ distinct candidate evaluations. All fitted mappings use training-side data; test labels do not enter candidate construction or selection.

Data-driven weight search. Optimized Weighted Fusion assigns a weight variable to every available source and searches the nonnegative unit simplex at a resolution of 0.1. For m sources, its grid and cardinality are

$$\mathcal{W}_m = \left\{ \mathbf{w} \in \{0, 0.1, \ldots, 1\}^m : \sum_{j=1}^{m} w_j = 1 \right\}, \qquad |\mathcal{W}_m| = \binom{m+9}{m-1}. \tag{17}$$

For three sources, the grid contains 66 combinations, including single-source vertices and approximately equal weights. When the grid has at most 64 candidates, all are evaluated; otherwise, 64 are sampled uniformly without replacement using a fixed random seed. Zero weights permit a source to make no contribution. Thus this control optimizes numerical weights without language-derived priorities, subject to the same evaluation cap as program search.

Non-language candidate construction. Non-language Program Selection and CriScore share a finite set of legal program configurations $(\mathcal{Q}, \psi)$ with $\mathcal{Q} \in \mathcal{A}_{\mathbb{V}}$; searched numerical settings are restricted to declared discrete parameter sets. The non-language control samples a nonempty source subset uniformly, then samples uniformly among its supported aggregation operators and their legal parameter settings. Temperature and blending strength take values in $T_p \in \{0.5, 1, 2\}$ and $\beta_p \in \{0.25, 0.5, 0.75, 1\}$, respectively. Source weights, where required, come from the discrete simplex grid rather than language-generated priorities. A supported optional residual correction is enabled or disabled with equal probability; its fitting and score mapping follow the shared training-side procedure. This is hierarchical random sampling, not a claim of uniform sampling over all complete programs. CriScore uses task semantics and, when available, training diagnostics to propose candidates in the same declared space.

Deduplication and selection. A legal candidate is canonicalized by its source assignments, active operators, and discrete parameters, then deduplicated by a deterministic hash. Inactive parameter fields do not create distinct candidates, and duplicate proposals are not reevaluated. One evaluation is counted only after a distinct candidate completes all prescribed training-side validation folds and yields M_p and G_p; individual folds do not count separately. For each program-search method, the number of evaluated candidates is bounded by

$$K_{\text{eval}} \le \min\big(K_{\text{legal}}, 64\big), \qquad p^* = \arg\operatorname*{lex\,max}_{p \in \mathcal{E}} (M_p, G_p, -p), \tag{18}$$

where K_{legal} is the number of legal configurations, $\mathcal{E}$ is the set of evaluated candidates, and p is their fixed index. The same lexicographic rule selects the weighted-fusion configuration: maximize mean validation correlation, break ties by worst-fold gain, and then prefer the smaller candidate index, without a positive-gain threshold. The cap controls candidate evaluations rather than total generation cost.

CriScore improves SRCC over non-language program selection by 0.774, 0.640, 0.425, and 0.614 percentage points on MTL-AQA, TASD-2, FineDiving, and UI-PRMD, respectively, averaging 0.613 points. Gains over optimized weighted fusion range from 0.679 to 1.249 points and aver-

Table 3: Fusion and program-selection controls using shared evidence predictions. Search-based methods have a common cap of 64 candidate evaluations; Uniform Fusion is fixed. Values are SRCC (%).

Method	MTL-AQA	TASD-2	FineDiving	UI-PRMD
Uniform Fusion	84.185	85.284	80.213	73.284
Optimized Weighted Fusion	85.032	85.715	80.658	73.916
Non-language Program Selection	85.341	86.324	80.912	74.053
CriScore	**86.115**	**86.964**	**81.337**	**74.667**

age 0.941 points. Weight search tests whether numerical reweighting accounts for the gains; non-language program search additionally allows source and operation choices. The results support language-assisted candidate construction within the evaluated search spaces and budget. This comparison does not establish superiority under arbitrary search budgets or statistical stability across repeated runs. On TASD-2, equal weighting of all available sources differs from equal weighting of a selected subset: the source set remains part of the program even when its weights are uniform.

Induction inputs and fixed-program interventions. Table 4 defines the changed and fixed quantities for two types of intervention. Removing semantics or diagnostics changes the information available for induction and requires regenerating and selecting a program; its effect can involve both structure and parameters. By contrast, equal weighting, residual removal, and binding permutation operate on a fixed program while preserving the quantities listed in the table. Equal weighting is applicable only when the selected set contains multiple sources with nonuniform weights. Residual removal is applicable only when an additive correction is active; setting $\gamma_i^p = 0$ disables that correction while retaining score mapping. Absent or already identical operations are marked N/A rather than treated as performance-reducing interventions.

Interpretation of the reported configurations. Figure 5, Table 2, and Figure 2a compare the reported configurations under the same evidence bank within each task. The configurations labeled No Diag., No Sem., and No S.+D. in Table 2 have average SRCC gaps of 1.446, 1.153, and 2.540 points relative to CriScore. Separately, the w/o LLM configuration in Figure 5 reduces SRCC by 2.305, 2.774, 1.017, and 1.087 points across the four datasets, averaging 1.796 points. Joint removal of the two induction inputs and removal of the complete language-induction stage are distinct configurations. Appendix B.3 specifies the complete-stage replacements: bypassing execution returns the base prediction and removes composition and calibration together. These comparisons measure the performance changes caused by restricting the candidate-generation or evaluation space and reselecting the scoring program under the same evidence bank. Table 4 provides definitions for isolated interventions, not a retrospective assignment of these definitions to every plotted result. We retain the reported comparisons without assigning a multi-operation difference to one operator; dashes in Table 2 denote unused operations.

C.2 Frozen-Program Controls and Evaluation Cells

Figure 6 holds numerical evidence fixed while changing the scoring program. For MTL-AQA, FineDiving, and UI-PRMD, the random reference is the 95th percentile of 64 legal programs; TASD-2 uses its frozen aggregate reference. The worst legal program is selected using training-side performance and then evaluated on the same frozen evidence. The selected programs exceed the random references by 0.08, 2.98, 1.23, and 0.41 SRCC points on MTL-AQA, TASD-2, FineDiving, and UI-PRMD, respectively. Their gains over the training-side worst programs are 40.56, 1.30, 14.70, and 3.06 points. Program choice affects ranking under fixed predictions.

Panel (a) records how often language induction and execution improve SRCC across evaluation cells. MTL-AQA has positive effects in 5/5 and 4/5 registered cells, respectively. The other tasks use five folds and three random seeds: the corresponding counts are 14/15 and 15/15 for TASD-2, 15/15 and 15/15 for FineDiving, and 10/15 and 10/15 for UI-PRMD. Mean gains recur across all FineDiving cells, while rehabilitation results vary across partitions and seeds.

Table 4: Definitions of isolated interventions and the quantities held fixed. These definitions distinguish operator-level controls from the reported configuration-level comparisons.

Variant	Changed input or operation	Fixed quantities
w/o Diagnostics	Remove supplied training diagnostics from the language input; repeat induction and selection.	Task description, evidence interface, predictors, and evaluation budget.
w/o Semantics	Remove domain evaluation knowledge; repeat induction and selection.	Required execution interface, diagnostics, predictors, and evaluation budget.
w/ Equal Weights	Set weights within the selected source set to equal values.	Selected sources, base blending strength, score mapping, and other operations.
w/o Residual Correction	Set $\gamma_i^p = 0$ to disable an active additive correction.	Predictors, selected sources, composition weights, score mapping, and other operations.
w/ Shuffled Bindings	Permute source-to-input assignments using recorded random seeds.	Prediction arrays, operation structure, and numerical parameters.

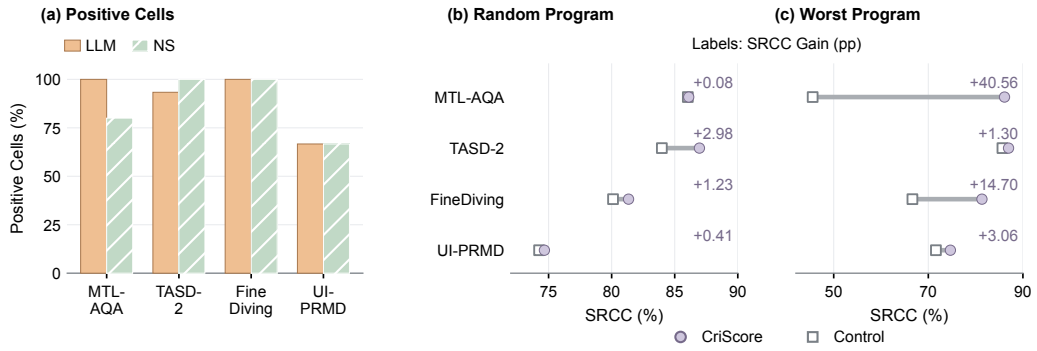


Figure 6: Frozen-program controls: (a) positive component effects across cells, (b) random-program references, and (c) training-side worst programs. Numerical evidence is fixed.

C.3 Program Selection and Paired Robustness

Figure 2b compares the selected program with uniform combination, the random reference, and a retrospective test oracle under fixed evidence predictions. The oracle uses evaluation outcomes to identify the best candidate and serves only as an upper reference. The selected program exceeds uniform combination and the random reference on each task while remaining below the oracle. These comparisons concern complete program configurations; they do not isolate individual selection criteria or tie-breakers.

Figure 2c pairs CriScore with removal of language induction, removal of relations, and permuted bindings within each evaluation unit. The 95% confidence intervals exclude zero in 11 of the 12 dataset–control pairs, with the reported p-values below 0.05. The exception is UI-PRMD versus w/o LLM: its mean gain is 1.087 points, its interval is $[-0.200, 2.300]$, and $p = 0.085$. This exception agrees with the variable rehabilitation gains in Figure 6(a), distinguishing a positive mean from a consistently positive effect across evaluation units.

C.4 Language-Model Sensitivity and Generation Reliability

Figure 7(a) compares language models under the same task inputs, admissible program space, output schema, and selection budget. The comparison covers DeepSeek-V4-Pro, DeepSeek-V4-Flash, GPT-6 Astra, Claude Sonnet 5, Gemini 3.8 Flash, and Qwen3.8-Max. DeepSeek-V4-Pro reaches the highest SRCC on MTL-AQA and UI-PRMD, at 86.115% and 74.667%. GPT-6 Astra leads on

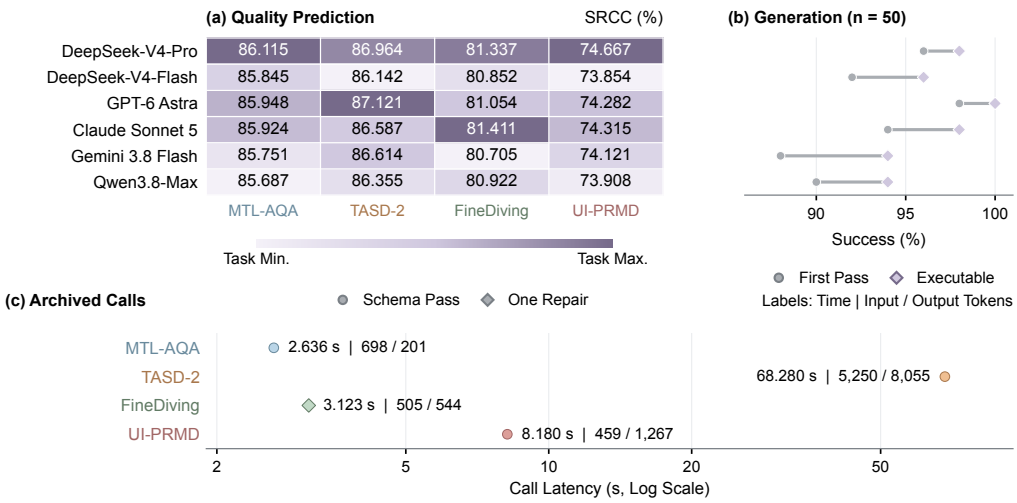


Figure 7: Language-model analysis: (a) SRCC with color scaled per dataset, (b) validity and executability over 50 generations per model, and (c) call times, tokens, and repair.

TASD-2 at 87.121%, and Claude Sonnet 5 leads on FineDiving at 81.411%. The strongest language model varies across these tasks within the shared induction interface.

Panel (b) reports 50 generations per model, separating first-pass validity from final executability after repair. First-pass rates range from 88% to 98%, while final executable rates range from 94% to 100%. GPT-6 Astra reaches 50/50 executable outputs; DeepSeek-V4-Pro and Claude Sonnet 5 reach 49/50, DeepSeek-V4-Flash reaches 48/50, and Gemini 3.8 Flash and Qwen3.8-Max reach 47/50. Repair recovers some initially invalid outputs, while panel (a) separately measures the resulting programs' predictive quality. Panel (c) gives task-call times and token counts.

C.5 Execution Replay and Evidence Deletion

Figure 3 replays TASD-2 video 10m/169 from a 68-observation training-side validation cohort, using a score mapping fitted on 269 observations. The returned identifier resolves to I3D and VideoMAE; the executor retains the base predictor and applies no post-composition correction. On the illustrated validation fold, the returned configuration without correction achieves an SRCC of 84.287%, compared with 84.294% for the shrinkage candidate. These values describe fold-specific performance. Final program selection uses the mean SRCC across the prescribed validation folds. The three predictors provide overall-quality estimates, whose normalized ranks yield the displayed score through the fitted mapping. All 68 validation observations share the same sources, composition, and fitted score map.

Figure 8 extends source removal to all four tasks while retaining the other frozen predictions. Deleting selected evidence reduces SRCC by 1.210 points on MTL-AQA, 1.452 on FineDiving, and 1.150 on UI-PRMD. The selected sources are Swin evidence, difficulty/bounding-box compactness, and prior-weighted motion blocks, respectively. Their unselected-source controls remove pose features, splash area, and velocity blocks, producing changes of 0.002, -0.001, and -0.003 points. For TASD-2, deleting VideoMAE or I3D decreases SRCC by 1.680 or 0.570 points, while deleting paired halves or person crop has zero effect. The small negative decreases denote equally small improvements after deletion. These interventions measure dataset-level ranking sensitivity to source removal. All outputs estimate overall quality under total-score supervision.

C.6 Task-Level Preparation and Repeated Scoring

Figure 4 separates task preparation from repeated scoring using the frozen program and fitted predictors. Figure 7(c) records the task calls underlying the language-request cost, including input/output tokens and initial schema validity. The FineDiving response required one deterministic schema extraction after the initial response; its additional extraction time was not logged. UI-PRMD reused its

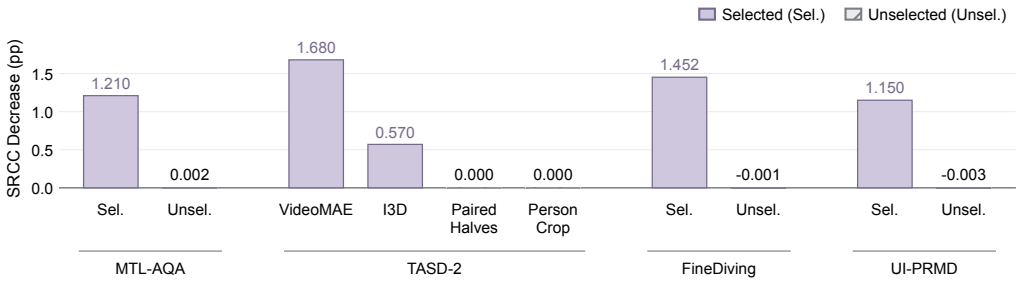


Figure 8: SRCC decreases after deleting selected and unselected evidence. TASD-2 reports VideoMAE and I3D separately, alongside the unselected paired-half and person-crop sources.

earlier prior during final confirmation. The repeated-generation experiment in Figure 7(b) separately measures reliability across 50 generation attempts per language model.

Preparation cost. Figure 4(a)–(b) includes language requests, compilation, predictor training, parameter selection, and calibration. The recorded totals are 454.949, 190.588, 34.654, and 283.350 seconds for MTL-AQA, TASD-2, FineDiving, and UI-PRMD, respectively. Predictor training dominates the first, third, and fourth tasks, while TASD-2's 68.280-second language request forms a larger share of preparation. The same program and predictors support repeated scoring.

FineDiving and UI-PRMD fitting includes inner cross-fitting and a final training fit. Calibration fits a score map for FineDiving and a linear map with residual prediction for UI-PRMD.

Repeated scoring cost. The complete cached-input scoring paths take 139.152, 14.115, 335.748, and 584.625 ms per timing batch on MTL-AQA, TASD-2, FineDiving, and UI-PRMD, respectively. These paths process 2,536.794, 5,100.956, 1,343.270, and 509.729 samples/s, respectively. Criterion execution takes 0.415, 1.062, 0.701, and 0.228 ms per cohort; evidence prediction accounts for most of the repeated numerical computation. Separately measured stage costs need not sum to the total.

All measurements use two CPU threads. FineDiving and UI-PRMD timing uses 30 executions after five warm-up calls. The MTL-AQA mapping and TASD-2 execution measurements use frozen outputs with 200 executions after 20 warm-ups, and the MTL-AQA measurement uses a second server. UI-PRMD's mapping includes residual prediction. Raw-video processing, feature extraction, file I/O, and language calls are excluded from the repeated scoring measurements. Throughput depends on cohort size for rank-normalized inputs.

D Additional Execution Details

D.1 Average-Rank Normalization

The population $\mathcal{U}$ with $n = |\mathcal{U}|$ determines the common comparison set for all sources, as in Equation 12. For a prediction z_{ij}, let a_{ij} count strictly smaller predictions and let t_{ij} count equal predictions, including observation i. The average rank is $a_{ij} + (t_{ij} + 1)/2$, and division by $n + 1$ yields normalized evidence

$$e_{ij} = \frac{2|\{k \in \mathcal{U} : z_{kj} < z_{ij}\}| + |\{k \in \mathcal{U} : z_{kj} = z_{ij}\}| + 1}{2(n+1)}. \tag{19}$$

This transformation uses source predictions and handles ties consistently across the base and auxiliary sources. It uses the declared validation or evaluation population, whose composition can change normalized values even when predictors and program parameters are frozen.

D.2 Fixed Training-Reference Scoring

Fixed-reference alternative. Population-relative normalization and independent-sample scoring are different execution settings. To evaluate the latter, define a training-only reference set $\mathcal{R}_j$ of n_j frozen predictions for each source, including the base predictor. For a new prediction z, a smoothed

empirical mid-distribution transform is

$$F_j^{\mathrm{ref}}(z) = \frac{\frac{1}{2} + \sum_{r \in \mathcal{R}_j} \left[\mathbb{I}(r < z) + \frac{1}{2}\mathbb{I}(r = z)\right]}{n_j + 1}, \qquad e_{ij}^{\mathrm{ref}} = F_j^{\mathrm{ref}}(z_{ij}). \tag{20}$$

The reference predictions, their generating predictors, and the tie convention remain fixed at evaluation time. Other test observations do not enter this transformation. Given identical source predictions and deterministic downstream operations, the fixed-reference transformation makes each prediction independent of evaluation-batch composition.

Training and evaluation separation. Within each validation fold, reference predictions must come from its fitting partition, excluding that fold and all test observations. Calibration must use the same reference transformation as execution. The final assessor freezes a training-only reference together with its predictors and score mapping. Reference size is recorded separately from the number of observations used to fit a score map; the latter does not determine the former. Replacing population-relative ranks with this transform changes the numerical inputs and requires reevaluating predictions and any affected calibration.

D.3 Composition and Correction Bounds

Selected nonnegative weights sum to one. Temperature scaling preserves a zero priority, and the convex combination keeps r_i^p between its smallest and largest input. Since both normalized inputs and the base rank lie in $[0, 1]$, the deviation from the base rank is bounded by $|r_i^p - b_i| \le \beta_p$.

For program $\mathcal{P}_p$, let $v_i^p = r_i^p$ for rank-coordinate correction and $v_i^p = g_p(r_i^p)$ for score-coordinate correction. The corrected coordinate is $\widehat{v}_i^p = v_i^p + \delta_i^p$, with δ_i^p defined in Equation 14. The bounded correction gives $|\widehat{v}_i^p - v_i^p| \le \gamma_i^p \eta_p B_p \le \eta_p B_p$. For two observations that share the frozen correction parameters, their corrected difference therefore satisfies

$$\widehat{v}_i^p - \widehat{v}_k^p = (v_i^p - v_k^p) + \delta_i^p - \delta_k^p \ge v_i^p - v_k^p - 2\eta_p B_p. \tag{21}$$

A positive separation greater than $2\eta_p B_p$ therefore retains its sign in the declared correction coordinate. For rank-coordinate correction, a subsequent nondecreasing g_p cannot reverse this ordering, although it may introduce a tie. Smaller separations can change order, while $\gamma_i^p = 0$ leaves an observation unchanged. The bound concerns the update and does not establish the accuracy of the predictions.