Vintage Code, Modern Judges: Meta-Validation in Low Data Regimes

Ora Fandina, Gal Amram, Eitan Farchi, Shmulik Froimovich, Raviv Gal
Wesam Ibraheem, Rami Katan, Alice Podolsky, and Orna Raz
IBM Research, Israel
{ora.nova.fandina, gal.amram, farchi, shmulik.froimovich, ravivg, wesam, rami.katan, alice.podolsky, ornar}@ibm.com

***Abstract*—Application modernization in legacy languages such as COBOL, PL/I, and REXX faces an acute shortage of resources, both in expert availability and in high-quality human evaluation data. While Large Language Models as a Judge (LaaJ) offer a scalable alternative to expert review, their reliability must be validated before being trusted in high-stakes workflows. Without principled validation, organizations risk a circular evaluation loop, where unverified LaaJs are used to assess model outputs, potentially reinforcing unreliable judgments and compromising downstream deployment decisions.**

Although various automated approaches to validating LaaJs have been proposed, alignment with human judgment remains a widely used and conceptually grounded validation strategy. In many real-world domains, the availability of human-labeled evaluation data is severely limited, making it difficult to assess how well a LaaJ aligns with human judgment.

We introduce SparseAlign, a formal framework for assessing LaaJ alignment with sparse human-labeled data. SparseAlign combines a novel pairwise-confidence concept with a score-sensitive alignment metric that jointly capture ranking consistency and score proximity, enabling reliable evaluator selection even when traditional statistical methods are ineffective due to limited annotated examples.

SparseAlign was applied internally to select LaaJs for COBOL code explanation. The top-aligned evaluators were integrated into assessment workflows, guiding model release decisions.

I. Introduction

The modernization of legacy software systems, particularly those implemented in languages such as COBOL, PL/I, and REXX, remains a strategic imperative for organizations operating critical infrastructure. However, this process is constrained by severe resource limitations, most notably the diminishing availability of domain experts and the absence of large, high-quality datasets for evaluation purposes. As noted in recent work on assessing code transformations in modernization pipelines, the lack of gold-standard references and task-specific benchmarks significantly hinders the reliable evaluation of system outputs [4]

In response to the scarcity of expert evaluators, Large Language Models as a Judge have emerged as a scalable alternative to human assessment for tasks such as code translation, refactoring validation, and documentation generation. These systems leverage the generative and reasoning capabilities of large-language models to provide flexible, semantic evaluations beyond surface-level similarity [8]. However, LaaJs exhibit well-documented weaknesses, including susceptibility to positional and verbosity biases, prompt sensitivity, and self-preference effects [11]. Such vulnerabilities raise concerns regarding their suitability for deployment in high-stakes workflows, where subtle inaccuracies in judgment may lead to costly or dangerous outcomes.

	Model 1	Model 2	Model 3	Model 4	Model 5	Model 6
Human_1 | Code_ID_26	6	5	6	6	6	6
Human_1 | Code_ID_27	6	6	6	4	6	4
Human_1 | Code_ID_33	2	1	4	1	5	1
Human_1 | Code_ID_6	5	6	6	5	4	6
Human_2 | Code_ID_25	5	4	6	1	4	6
Human_3 | Code_ID_12	6	6		4	5	5
Human_3 | Code_ID_37	6	6		7	4	6
Human_4 | Code_ID_11	2	5	6	1	3	5
Human_4 | Code_ID_15	5	6	6	5	4	6
Human_4 | Code_ID_32	5	5	6	5	5	5
Human_5 | Code_ID_1	7	7	5	6	4	6
Human_5 | Code_ID_3	6	2	2	2	2	2
Human_5 | Code_ID_7	5	2	1	3	4	4
Human_6 | Code_ID_17	6	5	5	5	5	6
Human_6 | Code_ID_29	5	3		5	5	5
Human_6 | Code_ID_38	6	5	5	5	5	6
Human_6 | Code_ID_39	5	5	5	5	5	5
Human_6 | Code_ID_4	5	5	4	1	5	6
Human_6 | Code_ID_40	4	6	6	5	5	6
Human_6 | Code_ID_41	5	4	3	6	5	5
Human_6 | Code_ID_42	6	7	6	6	5	6
Human_6 | Code_ID_5	5	5	5	4	4	6
Human_6 | Code_ID_8	5	5	3	5	5	6
Human_7 | Code_ID_28	4	4		4	3	3
Human_7 | Code_ID_30	5	5	6	4	5	5

Fig. 1. Human evaluation on COBOL: Six models explained the same 25 samples; seven annotators received sparse, disjoint subsets.

Given these vulnerabilities, rigorous validation of LaaJs is essential prior to their adoption in production workflows. Several methodologies have been proposed for evaluator validation [3], [7].

A commonly adopted strategy is to measure alignment with human evaluations, treating human-derived rankings as the reference standard. However, this approach presupposes access to sufficient and well-distributed human annotations, which is often violated in practice, particularly in specialized domains where expert evaluations are scarce and costly. For example, in one instance of our real-world internal settings, we had as few as 25 annotated samples per model (see Figure 1). Under such constraints, conventional statistical tools may yield inconclusive or unstable results.

To address this gap, we introduce SparseAlign, a lightweight, interpretable framework designed to validate evaluator alignment under extreme data sparsity.

Alignment Under Sparse Human Data. SparseAlign provides a principled formalism for quantifying agreement between LLM-based evaluators and expert human judgments in

2151-0849/25/$31.00 ©2025 IEEE
DOI 10.1109/ASEW67777.2025.00015

low-data regimes. Specifically designed for industrial settings where annotated examples are scarce, it builds on and extends techniques from order statistics.

The key contributions are as follows: 1) *Sparse Signal Extraction*: A method for inferring model ranking signals from non-overlapping, minimal human feedback. 2) *Confidence-Weighted Comparison*: A novel definition of pairwise confidence to quantify the reliability of individual ranking preferences. 3) *Alignment Metric*: An interpretable alignment score that combines confidence-weighted ranking agreement and absolute score proximity, yielding a robust measure of evaluator reliability.

While our framework is broadly applicable across tasks and evaluation metrics, we demonstrate its utility on a use case drawn from internal modernization workflows: ranking LaaJs by their ability to generate accurate explanations for COBOL programs.

Related Work. Recent surveys outline LaaJ applications and limitations [5], [8]. Validating these evaluators remains difficult under sparse supervision: Guerdan *et al.* show that standard protocols may favor weak judges in low-agreement settings [6], while Li *et al.* propose rubric-based multi-agent setups that assume dense annotations [9].

Legacy code modernization poses further challenges due to limited benchmarks. Diggs *et al.* show that automated metrics poorly reflect human preferences in legacy systems [2]. Vo *et al.* find LaaJs effective for Bash code evaluation when enhanced with bidirectional matching [10].

Ongoing Work. To validate SparseAlign's robustness, we adopt a simulation-based approach [1], using virtual LaaJs with tunable quality to test metric sensitivity under controlled noise. While the full framework is under development, we present a preliminary variant with two virtual LaaJs to illustrate SparseAlign's ability to distinguish evaluator quality.

II. Details and Experiments

Let $M_1, M_2, \ldots, M_k$ denote LLM models, each of which generates explanations for a predefined set of COBOL code snippets. Let (C_i, E_i) represent the input–output pair for model M_i, where C_i is the set of COBOL snippets and E_i is the corresponding set of explanations generated by M_i.

The outputs are evaluated by a set of human experts $H_1, \ldots, H_t$ and a set of candidate LaaJ evaluators $\text{LaaJ}_1, \ldots, \text{LaaJ}_n$. Each human expert H_j is assigned a subset of the samples from C_i for evaluation. We denote this subset as $(C_i[H_j], E_i[H_j]) \subseteq (C_i, E_i)$.

We assume limited evaluation data, with the total number of samples $\sum_i |C_i|$ being small. SparseAlign is designed to operate even under extreme human evaluation data constraints, such as when each sample is rated by only one evaluator (i.e., $C_i[H_j] \cap C_i[H_{j'}] = \emptyset$ for $j \neq j'$), which precludes traditional inter-annotator agreement analysis and necessitates alternative strategies for aggregating and comparing model-level rankings.

Each human evaluator scores their assigned samples on a fixed scale from 1 to s, using shared evaluation guidelines to promote consistency. Each candidate LaaJ also scores *all samples* on the same scale, but operates according to its own instruction prompt, which defines its evaluation behavior and may vary significantly across candidates and serve as the primary configuration parameter for each LaaJ.

The framework consists of two main parts:

1) **Human-Rank**: An algorithm for deriving a reliable consensus ranking of models from limited, disjoint human evaluation data.
2) **Align-Score**: A metric that quantifies how closely a candidate LaaJ replicates the human-derived model ranking.

A. Human-Rank Algorithm

To derive a consensus model ranking from sparse and imbalanced human evaluations, we begin by computing the average score for each model across all samples it was evaluated on.

When the difference between the average scores of two models, say M_i and M_j, is smaller than a threshold $\delta = \frac{1}{6} \cdot \text{median}\{\sigma_i\}_{i=1}^{k}$, where σ_i is the standard deviation of scores for model M_i, we consider the models tied in the global ranking. This threshold scales with the typical variability in human scores across models, ensuring that only differences substantially larger than the inherent score dispersion are considered meaningful.

A limitation of this rule is that the tie relation is not transitive: it is possible for M_1 to tie with M_2, and for M_2 to tie with M_3, while M_1 and M_3 differ by more than δ.

To reduce such inconsistencies, we apply a greedy clustering procedure: models are sorted by their average score in decreasing order, and ties are formed sequentially, adding a model to the current tie group only if its inclusion does not cause the group's score span to exceed δ threshold.

Greedy Tie-Clustering. Set $\delta = \frac{1}{6} \cdot \text{median}\{\sigma_i\}_{i=1}^{k}$ where σ_i is the standard deviation of scores for model M_i. Sort models by average score μ_i in descending order (highest to lowest). Start the first cluster with the top model.

For each remaining model M_i in sorted order:

1) Check if its μ_i differs by at most δ from every model in the current (last) cluster.
2) If yes, add it to that cluster.
3) If no, start a new cluster with this model.

Output the clusters in the order they were formed.

This procedure produces a partial order over the models, where each model belongs to exactly one tie cluster. A cluster may contain a single model or multiple models whose relative preference is interpreted as undecidable.

Breaking Ties: Weighted Pairwise Voting with Confidence. When models are placed in the same tie cluster because their average human scores differ by less than a threshold δ, we refine the ordering within the cluster using *weighted pairwise voting* with an associated *confidence score*. The goal is to use the strongest available human preferences to refine ties, while ensuring that no preference cycles are created and that ties remain when evidence is weak or conflicting.

Let $\mathcal{H}_{i,j}$ be the set of humans who evaluated outputs from both M_i and M_j. For each human $H \in \mathcal{H}_{i,j}$:

- Let μ_H^i and μ_H^j be the average scores H assigned to outputs from M_i and M_j, respectively.
- Let $n_H = |S_H^i| + |S_H^j|$ be the total number of samples H evaluated for this pair.
- Let $w_H = \frac{n_H}{\sum_{H' \in \mathcal{H}_{i,j}} n_{H'}}$ be the normalized weight of annotator H for this pair.

Each annotator contributes w_H to the model with the higher average score for this pair; in case of equality, w_H is split equally. The total weighted votes are:

$$V(M_i, M_j) = \sum_{H \in \mathcal{H}_{i,j}} w_H \cdot 1[\mu_H^i > \mu_H^j] + \tfrac{1}{2} w_H \cdot 1[\mu_H^i = \mu_H^j]$$

$$V(M_j, M_i) = \sum_{H \in \mathcal{H}_{i,j}} w_H \cdot 1[\mu_H^j > \mu_H^i] + \tfrac{1}{2} w_H \cdot 1[\mu_H^i = \mu_H^j]$$

The **confidence score** for the comparison is:

$$\text{conf}(M_i, M_j) = |V(M_i, M_j) - V(M_j, M_i)| \in [0, 1],$$

which quantifies the net strength of weighted human preference between the models.

To obtain an acyclic partial order within the cluster, we:

1) Form a set of directed edges $(M_i \to M_j)$ for all pairs with $V(M_i, M_j) > V(M_j, M_i)$, assigning each edge strength $s(i, j) = \text{conf}(M_i, M_j)$.
2) Sort edges by descending $s(i, j)$, breaking ties deterministically (e.g., by $|\mu_i - \mu_j|$ or model index).
3) Initialize a directed graph G over the models in the cluster, with no edges.
4) Process edges in sorted order, adding each to G only if it does not create a directed cycle.

The resulting G is a directed acyclic graph: if a path exists from M_i to M_j, we conclude $M_i \succ M_j$; if no path exists in either direction, the models remain tied.

This design uses the full information from weighted human judgments, prioritizes the strongest and most reliable preferences, and ensures the absence of cycles. It refines ties when the evidence is clear, but preserves ties when preferences are inconclusive or contradictory, preventing overinterpretation of sparse evaluation data.

B. Align-Score Metric

We define an alignment score $\textit{align-score} \in [0, 1]$ between a human ranking R_H and a judge ranking R_J. It is based on an *evaluation discrepancy* ϵ_{rank} that combines two components: a confidence-weighted rank disagreement capturing how often and how strongly R_J differs from R_H, and a normalized score error measuring differences in assigned quality scores. The alignment score is computed as $\textit{align-score} = 1 - \epsilon_{\text{rank}}$, so higher values indicate stronger agreement.

The *confidence-weighted rank disagreement* is an adaptation of Kendall's τ distance, where each pairwise inversion is weighted by the human-assigned confidence for that model pair.

For each pair of models (M_i, M_j) in the ordering set $p_{ij} = \text{conf}(M_i, M_j)$ if R_H prefers $M_i \succ M_j$ and R_J reverses or ties;
$p_{ij} = 1$ if $\text{conf}(M_i, M_j) = 0$ and R_J imposes an order; otherwise $p_{ij} = 0$.

The rank disagreement is $\epsilon_{\text{rank}} = \frac{\sum_{i<j} p_{ij}}{\binom{k}{2}}$, where k is the number of models.

For the score error, let $s_H(M_i)$ and $s_J(M_i)$ denote the mean scores assigned to model M_i by humans and by the judge respectively, both normalized to $[0, 1]$. The normalized score error is defined as:

$$\epsilon_{\text{score}} = \frac{1}{k} \sum_{i=1}^{k} |s_H(M_i) - s_J(M_i)| .$$

The overall total evaluation discrepancy is:

$$\varepsilon = \alpha \cdot \epsilon_{\text{rank}} + (1 - \alpha) \cdot \epsilon_{\text{score}},$$

where $\alpha \in [0, 1]$ controls the trade-off between ranking preservation and score proximity.

C. Experimental Results: COBOL Code Explanation

We applied the SparseAlign framework to the COBOL code explanation task to evaluate how well two candidate judges aligned with expert human assessments under extremely limited annotation.

The dataset contained COBOL code snippets extracted from production applications, along with natural-language explanations generated by LLM models under active development by model-training teams. These evaluations were conducted to inform decisions on which models demonstrated superior performance and should be advanced toward production deployment. Each explanation was reviewed by our subject-matter experts (SMEs) according to task-specific evaluation guidelines. The evaluation used a 1–7 scoring scale. A total of 7 human evaluators $(H_1, \ldots, H_7)$ assessed explanations produced by 6 different models $(M_1, \ldots M_6)$. Each model generated explanations for the same 25 code snippets, resulting in 150 (code, explanation) pairs. These pairs were distributed across annotators without overlap, as illustrated in Figure 1.

For this experiment, we evaluated four candidate LaaJs. L_1, based on `llama-405b-instruct`, and L_2, based on `llama-4-maverick`, were both configured with the same evaluation prompt (omitted here for brevity) to isolate the effect of the underlying LLM's capabilities while keeping evaluation criteria fixed.

We also included two simulated baselines. L_{random} assigns scores uniformly at random in the range 1–7, representing a completely uninformative judge. $L_{\text{human\_close}}$ is derived from the human evaluation table by perturbing 10% of the grades in each model column by either $+1$ or -1 (clipped to $[1, 7]$). This retains the overall human preference structure while introducing controlled noise, simulating a high-quality but imperfect judge. These simulated LaaJs serve as stress tests for our framework: L_{random} provides a clear negative control and receives a low alignment score, while $L_{\text{human\_close}}$ achieves high alignment as expected. This validates SparseAlign's ability to reliably distinguish between strong and weak evaluators.

First, we derived the human reference ranking using the *Human-Rank* algorithm. Table I reports the mean and standard deviation of human evaluation scores for each model.

TABLE I
MEAN AND STANDARD DEVIATION OF HUMAN SCORES PER MODEL.

	M1	M2	M3	M4	M5	M6
Average	5.08	4.80	4.857	4.20	4.52	5.08
Std. Dev.	1.129	1.470	1.457	1.720	0.900	1.324

We continue with computing the threshold $\delta = \frac{1}{6} \cdot \text{median}(\sigma_i) \sim 0.23$. This threshold determines when differences in mean scores are too small to indicate a meaningful preference. Applying it to the mean human scores produces the following ranking, from most to least preferred, with ties: $< \{M_1, M_6\}, \{M_3, M_2\}, M_5, M_4 >$.

TABLE II
AVERAGE SCORES OF MODELS M1 AND M6 PER HUMAN EVALUATOR

Evaluator	Av. Score M1	Av. Score M6
H1: 4/25	**4.75**	4.25
H2: 1/25	5.00	**6.00**
H3: 2/25	**6.00**	5.50
H4: 3/25	4.00	**5.30**
H5: 3/25	**6.00**	4.00
H7: 10/25	5.20	**5.70**
H8: 2/25	**4.50**	4.00

We then apply the weighted voting scheme to resolve the ties between M_1 and M_6, and between M_3 and M_2. For $\{M_1, M_6\}$, the total normalized votes are $V(M_1) = 4/25 + 2/25 + 3/25 + 2/25 = 0.44$ and $V(M_6) = 1/25 + 3/25 + 10/25 = 0.56$, yielding $M_6 \succ M_1$ with confidence 0.12. For $\{M_3, M_2\}$, we obtain $V(M_2) = 0.52$ and $V(M_3) = 0.48$, giving $M_2 \succ M_3$ with confidence 0.04.

Overall, the *Human-Rank* algorithm yields the following human reference ranking: $R_H = \langle M_6,\ M_1,\ M_2,\ M_3,\ M_5,\ M_4 \rangle$, with $\text{conf}(M_6, M_1) = 0.12$ and $\text{conf}(M_2, M_3) = 0.04$, while all other model pairs have $\text{conf} = 1$.

Next, we ran all the LaaJs on the same set of samples, computed the average scores per model, and derived the corresponding rankings (omitted from submission).

Applying the align-score metric with the human reference ranking R_H, we decomposed the evaluation discrepancy ϵ into its two components: confidence-weighted rank disagreement ϵ_{rank} and normalized score error ϵ_{score}. The results for each candidate LaaJ are shown in Table III and Figure 2.

TABLE III

LaaJ	ϵ_{rank}	ϵ_{score}	*align-score*
Llama-4-maverick	0.30	0.18	**0.76**
Llama-3-405b	0.38	0.14	**0.74**
Random Scores	0.475	0.347	**0.589**
Human_Close	0.211	0.058	**0.866**

The results show that both candidate LaaJs achieve high alignment with human judgments, with Llama-4-maverick slightly outperforming Llama-3-405b due to lower rank disagreement. The human-close LaaJ scores highest, as expected, while the random LaaJ scores substantially lower, confirming SparseAlign's ability to distinguish between high- and low-fidelity evaluators.

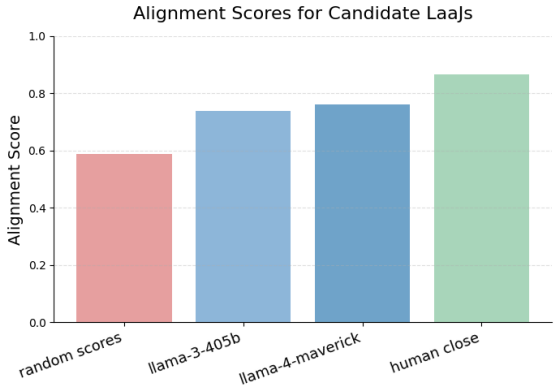

Fig. 2. SparseAlign correctly assigns low alignment to the weak evaluator L_{random} and high alignment to the strong evaluator $L_{\text{human\_close}}$, while quantifying the performance of the two real LaaJs.

REFERENCES

[1] Gal Amram, Eitan Farchi, Shmulik Froimovich, Raviv Gal, and Avi Ziv. Laajmeter: A framework for laaj evaluation, 2025.

[2] Colin Diggs, Michael Doyle, Amit Madan, Siggy Scott, Emily Escamilla, Jacob Zimmer, Naveed Nekoo, Paul Ursino, Michael Bartholf, Zachary Robin, Anand Patel, Chris Glasz, William Macke, Paul Kirk, Jasper Phillips, Arun Sridharan, Doug Wendt, Scott Rosen, Nitin Naik, Justin F. Brunelle, and Samruddhi Thaker. Leveraging llms for legacy code modernization: Challenges and opportunities for llm-generated documentation, 2024.

[3] Ora Nova Fandina, Eitan Farchi, Shmulik Froimovich, Rami Katan, Alice Podolsky, Orna Raz, and Avi Ziv. Automated validation of llm-based evaluators for software engineering artifacts, 2025.

[4] Shmulik Froimovich, Raviv Gal, Wesam Ibraheem, and Avi Ziv. Quality evaluation of cobol to java code transformation, 2025.

[5] Jiawei Gu, Xuhui Jiang, Zhichao Shi, Hexiang Tan, Xuehao Zhai, Chengjin Xu, Wei Li, Yinghan Shen, Shengjie Ma, Honghao Liu, Saizhuo Wang, Kun Zhang, Yuanzhuo Wang, Wen Gao, Lionel Ni, and Jian Guo. A survey on llm-as-a-judge, 2025.

[6] Luke Guerdan, Solon Barocas, Kenneth Holstein, Hanna Wallach, Zhiwei Steven Wu, and Alexandra Chouldechova. Validating llm-as-a-judge systems in the absence of gold labels, 2025.

[7] Sangwon Hyun, Mingyu Guo, and M. Ali Babar. METAL: Metamorphic Testing Framework for Analyzing Large-Language Model Qualities . In *2024 IEEE Conference on Software Testing, Verification and Validation (ICST)*, pages 117–128, Los Alamitos, CA, USA, May 2024. IEEE Computer Society.

[8] Haitao Li, Qian Dong, Junjie Chen, Huixue Su, Yujia Zhou, Qingyao Ai, Ziyi Ye, and Yiqun Liu. Llms-as-judges: A comprehensive survey on llm-based evaluation methods, 2024.

[9] Yuran Li, Jama Hussein Mohamud, Chongren Sun, Di Wu, and Benoit Boulet. Leveraging llms as meta-judges: A multi-agent framework for evaluating llm judgments, 2025.

[10] Ngoc Phuoc An Vo, Brent Paulovicks, and Vadim Sheinin. Llm-as-a-judge for reference-less automatic code validation and refinement for natural language to bash in it automation, 2025.

[11] Jiayi Ye, Yanbo Wang, Yue Huang, Dongping Chen, Qihui Zhang, Nuno Moniz, Tian Gao, Werner Geyer, Chao Huang, Pin-Yu Chen, Nitesh V Chawla, and Xiangliang Zhang. Justice or prejudice? quantifying biases in llm-as-a-judge, 2024.