

BAYESIAN POST TRAINING ENHANCEMENT OF REGRESSION MODELS WITH CALIBRATED RANKINGS

Kevin Tirta Wijaya¹, Bing Hu², Hans-Peter Seidel¹, Wojciech Matusik³, Vahid Babaei^{1,4,5}

¹Max Planck Institute for Informatics, ²University of Waterloo,

³Massachusetts Institute of Technology, ⁴Fraunhofer SCAI, ⁵University of Bonn

kevintirta.w@gmail.com, b25hu@uwaterloo.ca,

vahid.babaei@scai.fraunhofer.de

ABSTRACT

Accurate regression models are essential for scientific discovery, yet high-quality numeric labels are scarce and expensive. In contrast, rankings (especially pairwise) are easier to obtain from domain experts or artificial intelligence judges. We introduce RANKREFINE++, a novel plug-and-play method that improves a base regressor’s prediction for a query by leveraging pairwise rankings between the query and reference items with known labels. RANKREFINE++ performs a Bayesian update that combines a Gaussian likelihood from the regressor and the Bradley-Terry likelihood from the ranker. This yields a strictly log-concave posterior with a unique maximum likelihood solution and fast Newton updates. We show that prior state-of-the-art is a special case of our framework, and we identify a fundamental failure mode: Bradley-Terry likelihoods suffer from scale mismatch and curvature dominance when the number of reference items is large, which can degrade performance. From this analysis, we derive a calibration method to adjust the information originating from the expert rankings. RANKREFINE++ shows a stunning 97.65% median improvement across 12 datasets over previous state-of-the-art method using a realistically-accurate ranker, and runs efficiently on a consumer-grade CPU.

1 INTRODUCTION

Regression underpins many scientific and engineering pipelines. In materials science and drug discovery, for example, structure-to-property predictors (i.e., through regression) can guide inverse molecular design loops by replacing expensive experiments and simulations with fast surrogates (Sanchez-Lengeling & Aspuru-Guzik, 2018; Liang et al., 2021; Allen & Tkatchenko, 2022). Yet, in many domains, collecting accurate absolute labels (e.g., molecule x is toxic at 1 nM) is slow and expensive (Wijaya et al., 2024), leading to data scarcity and limiting the accuracy of learning-based regressors.

An alternative to absolute labels is pairwise ranking (e.g., is molecule x more toxic than molecule y?). Pairwise rankings can be collected from human experts or from computational models, including general-purpose large language models (LLMs), at relatively low cost and with competitive accuracy. For humans, pairwise ranking reduces cognitive load (Routh et al., 2023) and mitigate scale-interpretation bias (Hoeijmakers et al., 2024) compared to directly predicting the absolute labels. Similarly, LLMs exhibit strong pairwise ranking ability in technical domains (Guo et al., 2023; Sun et al., 2025). This raises a central question: **can we enhance a pretrained regressor using only a handful of absolute labels plus inexpensive pairwise rankings without retraining the regressor?**

We propose RANKREFINE++, a post-training enhancement method that combines a regressor’s prediction with *calibrated* expert rankings via Bayesian inference. Given a pretrained regressor and an external pairwise ranker that compares a query item to a small set of labeled references, RANKREFINE++ forms a posterior over the unknown scalar by combining (i) a Gaussian likelihood centered at the regressor’s prediction, and (ii) a Bradley-Terry likelihood for pairwise rankings. We show that the posterior is strictly log-concave, yielding a unique solution and enabling fast optimization.

An additional key insight is that the default Bradley-Terry model can be mismatched to the behavior of pairwise rankers, leading to biased estimates and increased errors. **RANKREFINE++** addresses this with (i) a learned temperature that calibrates the sigmoid slope of the Bradley-Terry model, and (ii) an accuracy-aware soft gate that regulates the influence of the ranker.

Specifically, our contributions are:

1. We introduce **RANKREFINE++** which combines a regressor likelihood with a calibrated ranker likelihood to enhance predictions without retraining. We prove that the state-of-the-art, RankRefine (Wijaya et al., 2025), is a special case under the Gaussian assumption.
2. We show how an uncalibrated Bradley-Terry model can bias the estimates when ranker likelihood dominates, and we provide a temperature calibration with accuracy-aware soft gating mechanism to mitigate the issue.
3. Across 12 cross-domain datasets (including real-world molecular datasets), **RANKREFINE++** significantly improves over existing post-training enhancement methods (Wijaya et al. (2025); Yan et al. (2024), and 3 other baselines) and remains effective when using imperfect LLM rankers. Specifically, **RANKREFINE++** achieves a 19.33% median MAE reduction relying on 30 reference samples and a ranker with 66% accuracy, which translates to a stunning 97.65% relative improvement compared to RankRefine’s 3.38% median MAE reduction.

Together, these results position **RANKREFINE++** as a practical and principled way to leverage readily-available pairwise information to enhance scalar regressor in data-scarce domains. Source code is available at <https://github.com/kiirta/regref>.

2 BACKGROUND

Pairwise Comparison Models. In pairwise (binary) comparisons, the probability of item i is preferred to x_j is often modeled as $P(x_i > x_j) = F(y_i - y_j)$, where y are latent scores and F is a cumulative distribution function (Cattelan, 2012). Classic pairwise comparison models include Thurstone-Mosteller with a Gaussian link function (Thurstone, 1927; Mosteller, 1951) and Bradley-Terry with a logistic link function (Bradley & Terry, 1952). These models provide a one-dimensional likelihood for the unknown scalar label that can be estimated using MAP or MLE estimation.

LLM-as-a-judge. General-purpose LLMs have demonstrated strong performance on relative comparisons across various domains (Qin et al., 2024; Wu et al., 2024; Guo et al., 2023), with evidence of better performance in pairwise comparison compared to score prediction (Zheng et al., 2023). With widely available web APIs (OpenAI, 2025; Anthropic, 2025; Google, 2025), collecting comparisons at scale is increasingly easy, making LLMs a convenient external ranker when numeric absolute labels are scarce.

Regression Refinement using Pairwise Rankings Two representative approaches in this area are the Projection method (Yan et al., 2024) and RankRefine (Wijaya et al., 2025). Projection constrains the regressor’s prediction to a feasible interval implied by non-contradictory pairwise outcomes. Meanwhile, RankRefine fuses the regressor’s output with a rank-only estimate via inverse variance weighting. Our proposed method introduces a different paradigm: it reframes fusion as a Bayesian inference, where the regressor and ranker contribute likelihoods that jointly determine the posterior.

Relation to Learning from Human Feedback. Recent works on fine-tuning with human feedback (Christiano et al., 2017; Ziegler et al., 2019; Ouyang et al., 2022) also use pairwise rankings modeled via Bradley-Terry, but with a different goal. They (i) learn a global reward model and (ii) fine-tune a pretrained model accordingly. In contrast, we apply a per-query, post training prediction correction by combining the regressor’s likelihood with a calibrated ranker likelihood. This produces a one-dimensional posterior without modifying the regressor’s parameters.

3 METHOD

In this section, we describe the base formulation of **RANKREFINE++** (Section 3.1), along with the analyses of its behavior (Section 3.2) and proposed modifications for improving the performance (Section 3.3). We provide the detailed proofs of the analysis in the Appendix.

3.1 BAYESIAN INFERENCE ON EXPERT RANKINGS TO IMPROVE REGRESSION MODELS

Let f be a regressor trained on $\mathcal{D} = \{(x_j, y_j)\}_{j=1}^N$, with scalar labels $y_j \in \mathbb{R}$ and prediction $f(x_j) = y_j^{\text{re}}$. Let R be an expert pairwise ranker that returns a binary comparison: $R(x_a, x_b) = 1$ if it predicts $y_a > y_b$, and $R(x_a, x_b) = 0$ otherwise. We are given a reference set $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^k$ with known y_i , which can be, but not always is, the regressor's training set \mathcal{C} . For a query x_0 with unknown $y_0 \in \mathbb{R}$, we collect expert rankings against all references: $G = \{r_i = R(x_0, x_i)\}_{i=1}^k$ where $r_i \in \{0, 1\}$.

Assuming conditional independence of regressor and ranker given the true Bayes' rule yields

$$\underbrace{p(y_0 | y_0^{\text{re}}, G)}_{\text{posterior}} / \underbrace{p(y_0^{\text{re}} | y_0)}_{\text{reg. likelihood}} \underbrace{p(G | y_0)}_{\text{rank likelihood}} \underbrace{p(y_0)}_{\text{prior}} \quad (1)$$

with Gaussian regressor likelihood $p(y_0^{\text{re}} | y_0) = \mathcal{N}(y_0^{\text{re}}; y_0, \frac{\sigma^2}{\alpha})$ and rank likelihood

$$p(r_i = 1 | y_0; y_i) = \text{s}(y_0 - y_i); \quad \text{s}(z) = \frac{1}{1 + e^{-z}}; \quad (2)$$

$$p(G | y_0) = \prod_{i=1}^k \text{s}(y_0 - y_i)^{r_i} (1 - \text{s}(y_0 - y_i))^{1 - r_i}; \quad (3)$$

which is the product of all pairwise comparisons. Under the Bradley-Terry model (Bradley & Terry, 1952).

With the two likelihoods, the base formulation of RANKREFINE++ enhances a regressor prediction with expert rankings using a maximum a posteriori (MAP) estimation (Murphy, 2022) which maximizes with the objective function

$$L(y) = \frac{1}{2} \frac{\sigma^2}{\alpha} (y_0^{\text{re}} - y)^2 + \sum_{i=1}^k r_i \log \text{s}(y - y_i) + (1 - r_i) \log(1 - \text{s}(y - y_i)) + \log p(y); \quad (4)$$

If we assume a flat (uninformative) prior, i.e. $p(y) = 1$, RANKREFINE++ reduces to a maximum likelihood estimation (MLE) (Murphy, 2022). The objective function of the RANKREFINE++ (Equation 4) is strictly log-concave so we can obtain the maximum efficiently.

Lemma 3.1. (Strict log-concavity.) If $\frac{\sigma^2}{\alpha} > 0$ and $\log p(y_0)$ is concave (including the flat prior), then $\log p(y_0 | y_0^{\text{re}}, G)$ is strictly concave in y_0 .

Corollary 3.2. (Existence and uniqueness.) The maximum a posteriori (and maximum likelihood under a flat prior) estimate exists and is unique.

Proposition 3.3. (RankRefine (Wijaya et al., 2025) is a special case of RANKREFINE++ under Gaussian assumption.) Let $L_{\text{BT}}(y)$ denote the rank-only log-likelihood and $y_0^{\text{ra}} = \arg \max_y L_{\text{BT}}(y)$. By assuming Gaussianity on the ranker likelihood (similar to Wijaya et al. (2025)), the second-order expansion of $L_{\text{BT}}(y)$ around y_0^{ra} yields

$$p(G | y) \approx \mathcal{N}(y; y_0^{\text{ra}}, \frac{\sigma^2}{\alpha}); \quad \frac{\sigma^2}{\alpha} = \sum_{i=1}^k \text{s}(y_0^{\text{ra}} - y_i) (1 - \text{s}(y_0^{\text{ra}} - y_i))^{-1}; \quad (5)$$

Combining this with the Gaussian regressor likelihood and a flat prior gives the inverse-variance weighted (IVW) (Cochran & Carroll, 1953) estimator, i.e., RankRefine (Wijaya et al., 2025), for which the estimator is,

$$y_0^{\text{rr}} = \frac{y_0^{\text{re}} = \frac{\sigma^2}{\alpha} + y_0^{\text{ra}} = \frac{\sigma^2}{\alpha}}{1 = \frac{\sigma^2}{\alpha} + 1 = \frac{\sigma^2}{\alpha}}; \quad (6)$$

Proof. The gradient and curvature of $L_{\text{BT}}(y)$ are

$$L_{\text{BT}}^0(y) = \sum_{i=1}^k r_i - \text{s}(y - y_i); \quad L_{\text{BT}}^{00}(y) = - \sum_{i=1}^k \text{s}(y - y_i) (1 - \text{s}(y - y_i)); \quad (7)$$

The second-order Taylor expansion of L_{BT} around $y = y_0^{ra}$ is

$$L_{BT}(y) \approx L_{BT}(y_0^{ra}) + L'_{BT}(y_0^{ra})(y - y_0^{ra}) + \frac{1}{2}L''_{BT}(y_0^{ra})(y - y_0^{ra})^2 \quad (8)$$

The first term is constant and the second term is zero, therefore,

$$L_{BT}(y) \approx \frac{1}{2} \sum_{i=1}^k s(y_0^{ra} - y_i)(1 - s(y_0^{ra} - y_i))(y - y_0^{ra})^2 \quad (9)$$

Similar to RankRefine, we make a strong assumption of Gaussianity for the BT likelihood,

$$p(G | y_0) / \exp(L_{BT}(y)) = \exp\left(-\frac{1}{2} \sum_{i=1}^k s(y_0^{ra} - y_i)(1 - s(y_0^{ra} - y_i))(y - y_0^{ra})^2\right) \mathcal{N}(y_0^{ra}; \frac{2}{k}) \quad (10)$$

$$y_0^{ra} = y_0^{ra}; \quad \frac{2}{k} = \frac{h}{\sum_{i=1}^k s(y_0^{ra} - y_i)(1 - s(y_0^{ra} - y_i))}$$

The joint likelihood of the conditionally independent regressor and ranker is now a product of two Gaussians. The MLE, i.e., MAP estimate with a flat prior, then solves a weighted least-squares problem whose maximizer is the inverse-variance weighted average.

3.2 ANALYSIS OF POTENTIAL PERFORMANCE DEGRADATION OF BASE RANKREFINE++

Under the base formulation, RANKREFINE++ can degrade when the reference size and ranker accuracy exceed certain thresholds. The effect is dataset-dependent and arises from a mismatch between the Bradley-Terry model and the behavior of real-world rankers.

Separating the rank likelihood from the objective in Equation 4 and maximizing it yields the rank-only MLE y_0^{ra} . If the ranker were perfectly accurate and the likelihood perfectly specified, then $y_0^{ra} = 2(y_m; y_{m+1})$, where $y_m < y_0 < y_{m+1}$ are the closest references and r_i is the number of references ranked below y_i . In practice, the sigmoid in the Bradley-Terry model induces soft- vs. hard-count mismatch: the rank-only target solves a soft-count equation and need not lie in $(y_m; y_{m+1})$.

Lemma 3.4 (Rank-only MLE may target a pseudo ground truth.) Let $u_i(y) = s(y - y_i)$ and $m = \sum_{i=1}^k r_i$. The Bradley-Terry log-likelihood $L_{BT}(y)$ is strictly concave and its unique maximizer y_0^{ra} satisfies

$$L'_{BT}(y_0^{ra}) = 0 \iff \sum_{i=1}^k u_i(y_0^{ra}) = m \quad (11)$$

Because m is a hard count while u_i are sigmoid softcounts, the solution to $\sum_{i=1}^k u_i(y_0^{ra}) = m$ is a pseudo target y_0^{ra} , which can lie outside $(y_m; y_{m+1})$ even under a perfectly accurate ranker, biasing y_0^{ra} . See Figure 1 for illustration.

Bias from the rank-only target can increase the overall error if the rank likelihood dominates the regressor likelihood.

Rank likelihood dominance is governed by its curvature (Fisher information (Ly et al., 2017)), which grows with k .

Lemma 3.5. (Rank curvature (Fisher information) scales with k .) The Fisher information of the rank log-likelihood is $I_{rank}(y) = \sum_{i=1}^k u_i(y)(1 - u_i(y))$; so $I_{rank}(y)$ grows linearly with k . Moreover, $I_{rank}(y) \approx \frac{k}{4}$.

Figure 1: Soft- vs. hard-count mismatch from Lemma 3.4. We generate a synthetic reference set g and ground truth y_0 . From Equation 11, the rank-only MLE y_0^{ra} is the intersection between $u_i(y)$ and m , and $(y_m; y_{m+1})$ forms the feasible range. We can see that $u_i(y)$ with a step function correctly intersects the feasible range. However, using a sigmoid function results in a biased intersection outside the feasible range.

Lemma 3.6. (Info-weighted Newton step and dominance) As k grows, the optimization process using Newton’s method (Murphy, 2022) to solve Equation 4 is dominated by the rank likelihood term. That is, denoting $\mathbf{g}_{\text{tot}}(y)$ and $\mathbf{I}_{\text{tot}}(y)$ as the total gradient and Fisher information of both likelihoods at y , \mathbf{I}_{reg} as the Fisher information of the regressor likelihood, and $\mathbf{y}^{\text{ra}}(y)$ as the target of the ranker likelihood term at y , a Newton step is

$$y \leftarrow y + \frac{\mathbf{g}_{\text{tot}}(y)}{\mathbf{I}_{\text{tot}}(y)} = \frac{\mathbf{I}_{\text{reg}} \mathbf{y}_0^{\text{re}} + \mathbf{I}_{\text{rank}}(y) \mathbf{y}^{\text{ra}}(y)}{\mathbf{I}_{\text{reg}} + \mathbf{I}_{\text{rank}}(y)}, \quad (12)$$

which is an information-weighted average using Fisher information as the weights.

Since $\mathbf{I}_{\text{reg}} = 1 = \frac{2}{\sigma_{\text{re}}^2}$ is constant, Lemma 3.5 implies the rank likelihood term eventually dominates as k grows. Apart from Newton, for a first-order step $y \leftarrow y + \mathbf{g}_{\text{tot}}(y)$, the same decomposition yields $y \leftarrow y + [\mathbf{I}_{\text{reg}}(\mathbf{y}_0^{\text{re}} - y) + \mathbf{I}_{\text{rank}}(y)(\mathbf{y}^{\text{ra}}(y) - y)]$, showing similar information-weighted pull. The dominance of the rank likelihood then follows from Lemma 3.5.

Dominance alone is harmless, but combined with Lemma 3.4, a biased rank likelihood target steers the refinement away from \mathbf{y}_0 . Note that dominance occurs only when both the ranker accuracy and the regressor accuracy are sufficiently large. Moreover, the accuracy threshold decreases with k .

Lemma 3.7. (Accuracy threshold for rank dominance.) Define

$$p = \frac{1}{k} \sum_{i=1}^k \mathbb{1}(y_0 > y_i); \quad \rho(y) = \frac{1}{k} \sum_{i=1}^k u_i(y); \quad (13)$$

Let $a \in [0.5; 1]$ denote the ranker accuracy. Then, at any iteration, the rank likelihood dominates the regressor likelihood whenever

$$a > \frac{1}{2} + \frac{|\mathbf{y} \cdot \mathbf{y}_0^{\text{re}}|}{2k \rho(y) p}; \quad (14)$$

The right hand side of the inequality is the threshold accuracy $a_{\text{th}}(y)$, and it decreases as k grows for a fixed y , \mathbf{y}_0^{re} , and p . Therefore, the required ranker accuracy such that the rank term dominates the Newton step shrinks as k grows.

Corollary 3.8. (Rank likelihood can degrade base RANKREFINE++) Under the base formulation of RANKREFINE++, the rank-only target can be biased (Lemma 3.4). This may lead to a performance degradation when the ranker likelihood term dominates (Lemma 3.6). In general, ranker likelihood dominance grows with the reference set size and the ranker accuracy. When k is large, the required accuracy for the ranker likelihood to dominate is lower (Lemma 3.7).

3.3 RANKREFINE++ WITH CALIBRATED EXPERT RANKINGS

The degradation stems from the Bradley-Terry modeling mismatch (Lemma 3.4): when many pairwise gaps ($y_a - y_b$) fall on the sigmoid’s transition region rather than its saturated tails, the rank-only target becomes biased. This is a dataset-scale issue; datasets with small average gaps ($y_a - y_b$) have higher probability of placing many pairs on the transition slope.

We address this by introducing a temperature to adjust the slope, i.e., replace $\sigma(y - y_i)$ with $v_i(y; \tau) = \sigma(\frac{y - y_i}{\tau})$. This (i) aligns the logistic slope with label units so that the tempered score $v_i(\mathbf{y}_0^{\text{ra}}; \tau)$ can match the observed count $\sum_i r_i$, removing the soft- vs. hard-count mismatch, and (ii) sets the rank curvature to

$$\mathbf{I}_{\text{rank}}(\mathbf{y}; \tau) = \sum_i \frac{v_i(y; \tau) (1 - v_i(y; \tau))}{\tau^2}; \quad (15)$$

thereby controlling rank dominance at large k . We estimate τ for a dataset via a one-parameter logistic fit on the reference set, thus requiring no extra labeled data,

$$\Pr(r = 1 \mid y_a, y_b; \tau) = \sigma(\tau(y_a - y_b)); \quad (y_a, y_b) \in D; \quad (16)$$

and set $\tau = \tau_{\text{cal}} = 1/\tau$. With this calibration of expert rankings, the Newton update is an information-weighted average with temperature-controlled curvature,

$$y \leftarrow y + \frac{\mathbf{I}_{\text{reg}} \mathbf{y}_0^{\text{re}} + \mathbf{I}_{\text{rank}}(\mathbf{y}; \tau) \mathbf{y}^{\text{ra}}(\mathbf{y}; \tau)}{\mathbf{I}_{\text{reg}} + \mathbf{I}_{\text{rank}}(\mathbf{y}; \tau)}. \quad (17)$$

Figure 2: Illustration of RANKREFINE++. For a query x_0 , RANKREFINE++ collects the pairwise rankings from an expert ranker and converts them into a ranker likelihood via the calibrated Bradley-Terry model. The ranker likelihood is then fused with the regressor likelihood, and optionally with a prior, to define a posterior. The refined prediction y_0^r maximizes the posterior distribution.

$$\text{where } y_0^r(y; \cdot) = y + \frac{\sum_{i=1}^k (r_i - v_i(y; \cdot))}{I_{\text{rank}}(y; \cdot)}.$$

Setting $\alpha < 1$ increases the rank likelihood curvature (via the α^2 factor) and shifts the update toward the rank target. When the ranker is accurate, this permits rank dominance without the bias highlighted by Lemma 3.4, improving performance over the naive RANKREFINE++.

On the other hand, when the ranker is less accurate, overly large curvature would let noisy rank signals dominate and harm performance. We therefore apply accuracy-aware soft gate

$$w(\alpha) = 1 + (\alpha_{\text{cal}} - 1) \cdot \alpha; \quad \alpha \in [0, 1]; \quad (18)$$

$$w(\alpha) = \max(0, 2\alpha - 1) \cdot \alpha \in [0, 1]; \quad (19)$$

Thus $w(\alpha) = 1$ when $\alpha = 0.5$, $w(\alpha) = \alpha_{\text{cal}}$ as $\alpha \rightarrow 1$, and intermediate α produces a smooth interpolation. The final RANKREFINE++ with calibrated expert rankings is summarized in Algorithm 1, and illustrated in Figure 2.

Algorithm 1 RANKREFINE++ Algorithm

Inputs: $x_0, y_0^e, \alpha, \alpha_{\text{cal}}, G = f(x_i; y_i)_{i=1}^k; D = f(y_i)_{i=1}^k; R; (\text{optional: } p(y))$
 Collect Comparisons: $r_i = R(x_0; x_i)$
 Estimate Temperature: $t = \text{stdev}(y_a - y_b)$ on labeled pairs from D ; set $\alpha_{\text{cal}} = 1 - \alpha$. Estimate the ranker accuracy; set $\alpha = 1 + (\alpha_{\text{cal}} - 1) \cdot (\max(0, 2\alpha - 1))$.
 MAP / MLE Optimization: Maximize Equation 4 with (\cdot) replaced by $(y_0^r(\cdot))$. With Newton steps described on Equation 17, iterate to convergence.
 Output: y_0^r and approximate uncertainty $\sigma_{\text{post}} = (I_{\text{reg}} + I_{\text{rank}}(y_0^r; \cdot))^{-1}$.

4 EXPERIMENTS

To demonstrate the benefits of RANKREFINE++ in data-scarce domains, we evaluate on 9 molecular property prediction datasets from the TDC ADMET regression task (Huang et al., 2021): Caco-2 (Wang et al., 2016), Clearance Microsome and Clearance Hepatocyte (Di et al., 2012), log Half-Life (Obach et al., 2008), FreeSolv (Mobley & Guthrie, 2014), Lipophilicity (Wu et al., 2018), PPBR, Solubility (Sorkun et al., 2019), and VDss (Lombardo & Jing, 2016). For each experiment, we sample $N = 100$ molecules from the original training split and $n = 100$ molecules from the original test split. We repeat this train/test resampling for 10 random seeds, similar to a Monte Carlo cross-validation. The reference set is sampled from the training set with $n \in \{3, 10, 20, 30, 50, 100\}$.

Two types of base regressors are used: random forest (RF) (Ho, 1995) and multilayer perceptron (MLP) (Rumelhart et al., 1986), both trained on the training labels. We also use three tabular regression datasets: crop-yield prediction from sensor data (Soundankar, 2025), student-performance prediction (Cortez, 2014), international-education cost estimation (Shamim, 2025).

Figure 3: MAE ratio (lower is better) as a function of oracle ranker accuracy on nine TDC ADMET datasets ($k = 30$). We report the mean and standard deviation across 10 random splits. RANKREFINE++ (MLE-GatedTemp variant) outperforms projection and RankRe ne across ranker accuracies, with especially strong gains at moderate, real-world accuracy. Results for other datasets with similar trends are available in Appendix.

We evaluate four RANKREFINE++ variants: MAP with a Gaussian prior, MLE, MLE with temperature scaling (MLE-Temp), and MLE with temperature scaling and soft gating (MLE-GatedTemp). The full RANKREFINE++, i.e., MLE-GatedTemp, is the default variant we use in the experiments. We compare our method to two post-training regression enhancement baselines: (i) Projection-based approach (Yan et al., 2024) and (ii) RankRe ne (Wijaya et al., 2025). Additionally, we adapt two pairwise ranking models to create rank-only regression enhancement baselines: (i) Bradley-Terry (rank-only MLE using the Bradley-Terry model), (ii) Thurstone (rank-only MLE using the Thurstone-Mosteller model (Thurstone, 1927; Mosteller, 1951)). Following Wijaya et al. (2025), we report $\text{MAE}_{\text{post}} = \text{MAE}_{\text{base}}$ which is the ratio between the post-enhancement Mean Absolute Error (MAE) and the MAE of the regressor-only predictions (lower is better).

We use two types of rankers: (i) oracle ranker to study the effect of ranker accuracy, and (ii) LLM ranker to demonstrate real-world use cases. For the oracle ranker, we set (y_0, y_i) and then flip the outcomes with probability $1 - a$ to simulate a ranker with accuracy $a \in [0.5; 1]$. For LLM rankers, we prompt publicly-available models to compare a list of pairs of molecular text representations (SMILES (Weininger, 1988)) and measure the accuracy on a small validation set.

Additional discussions and experiments on LLM usage, temperature calibration, multi-target regression, and reference set size are available in Appendix.

4.1 MAIN RESULTS WITH ORACLE RANKERS

Across all nine ADMET datasets ($k = 30$), RANKREFINE++ (MLE-GatedTemp variant) consistently outperforms previous state-of-the-art, RankRe ne, and 3 other baselines (Figure 3). Gains over recent custom-built enhancement methods, RankRe ne and Projection, are largest in the mid-

Figure 4: Effects of reference set size and temperature scaling / soft gating on the Clearance Hepatocyte dataset. The shown methods are the four variants of RANKREFINE++, with ours being MLE-GatedTemp. We show the MAE ratio (lower is better) as a function of oracle ranker accuracy a for $k \in \{3, 10, 20, 30, 50, 100\}$. Without temperature scaling, MAP and MLE variants degrade at larger k due to rank curvature dominance. Temperature (MLE-Temp) fixes the scale mismatch, and soft-gating (MLE-GatedTemp) further improves robustness when ranker accuracy is low.

accuracy regime (65% - 90%). At lower ranker accuracies (65%), RANKREFINE++ matches or exceeds RankRe-ne and clearly outperforms Projection. At high ranker accuracies, RankRe-ne performance plateaus, while RANKREFINE++ continues to improve, converging to Projection's performance at perfect (yet unrealistic) ranker accuracy.

4.2 ABLATION: PRIOR, TEMPERATURE, AND ACCURACY-AWARE SOFT GATING

Figure 4 shows that MAP/MLE variants of RANKREFINE++ can degrade as k grows, confirming that rank-dominance in the Newton step and the soft-hard count mismatch on the rank term can bias the enhancement (Corollary 3.8). The rank-only Bradley-Terry and Thurstone models in Figure 3 also exhibit similar degradation, validating that the source of degradation is the ranker likelihood. Interestingly, MAP degrades less than MLE, indicating that prior can act as a regularizer. As proposed in Lemma 3.7, the ranker accuracy threshold at which degradation appears shifts lower as k increases.

Our full method (MLE-GatedTemp variant) solves the performance degradation issue. Temperature scaling aligns the sigmoid slope to the label scale and controls the rank curvature, mitigating degradation at high ranker accuracy, but can worsen performance at low ranker accuracy (Figure 4, MLE-Temp). Adding accuracy-aware soft-gating removes this trade-off (Figure 4, MLE-GatedTemp), enabling full RANKREFINE++ to deliver strong performance across the complete ranker accuracy range. Note that Figure 4 also confirms that our enhancement method works with a reference set as small as $k = 3$, which is highly practical in the real-world setting.

4.3 CROSS-DOMAIN GENERALITY AND REGRESSOR DIVERSITY

To test cross-domain and cross-model generalization, we add three non-molecular tabular datasets and run both Random Forest (RF) and Multilayer Perceptron (MLP) regression models. Figure 5 shows that RANKREFINE++ yields < 1 across the ranker accuracy range, improving both RF and MLP models, and proving its generalization capability to other domains and regression models.

Figure 5: MAE ratio on three non-molecular tabular datasets with Random Forest (RF) and Multi-layer Perceptron (MLP) as base regressor models. We show that the refinements are generalizable to other domains and regressor models. Baseline (non-refined) MAE ratios: International Education Cost: 0.0926 (MLP), 0.0609 (RF); Smart Farming Yield: 0.1448 (MLP), 0.1361 (RF); Student Exam Performance: 0.0727 (MLP), 0.0819 (RF).

Dataset Name	Half Life	FreeSolv	PPBR	Solubility	VDss
ChatGPT5 PRA (%)	62.20 ± 1.89	62.27 ± 2.00	63.29 ± 2.65	62.87 ± 3.27	65.95 ± 1.96
RANKREFINE++ ()	0.952 ± 0.057	0.997 ± 0.021	0.928 ± 0.045	0.985 ± 0.015	0.853 ± 0.194
RankRefine ()	0.977 ± 0.024	1.002 ± 0.022	0.968 ± 0.016	0.985 ± 0.020	0.952 ± 0.055
Projection ()	1.056 ± 0.171	1.119 ± 0.100	1.007 ± 0.026	0.992 ± 0.039	0.949 ± 0.082
Dataset Name	Half Life	FreeSolv	PPBR	Solubility	VDss
Claude4 PRA (%)	52.36 ± 2.22	72.47 ± 1.40	51.73 ± 3.25	60.41 ± 2.45	60.85 ± 2.12
RANKREFINE++ ()	0.955 ± 0.058	0.977 ± 0.011	0.965 ± 0.042	0.990 ± 0.010	0.850 ± 0.191
RankRefine ()	0.981 ± 0.010	1.070 ± 0.052	0.979 ± 0.019	0.988 ± 0.016	0.951 ± 0.056
Projection ()	1.124 ± 0.279	1.153 ± 0.126	1.031 ± 0.039	0.995 ± 0.014	0.959 ± 0.038

Table 1: Results using LLMs (ChatGPT5, Claude4) as external rankers, measured on ten train/test splits with $N = 50; k = 20$. PRA stands for pairwise ranking accuracy. Best result for each dataset-ranker is bolded. **RANKREFINE++** generally outperforms other refinement methods.

4.4 LLM AS IMPERFECT YET PRACTICAL RANKER

We replace the oracle ranker with off-the-shelf LLMs to obtain noisy but scalable expert rankings. With ChatGPT5 Thinking and Claude Sonnet 4 as rankers at $N = 50$ and $k = 20$, **RANKREFINE++** generally achieves the best across *ve* ADMET datasets. Detailed results are shown in Table 1.

4.5 RUNTIME ANALYSIS ON VARYING REFERENCE SET SIZE

Excluding the ranking predictions, **RANKREFINE++** runs in less than 1 ms on an Intel i7-13700 CPU when the reference set size is 1000 covering realistic scenarios in data-scarce domains. The average running time to predict expert rankings using ChatGPT5-Thinking for 1000 queries and 50 references (50,000 pairs in total) in parallel is 152.4 ± 54.7 seconds. Therefore, **RANKREFINE++** can be run efficiently on consumer-grade computers, and LLM inference is not a significant bottleneck.

4.6 RANKREFINE IS A SPECIAL CASE OF RANKREFINE++.

We showed that RankRefine (Wijaya et al., 2025) is a special case of **RANKREFINE++** in Proposition 3.3. We verify this empirically by comparing enhanced predictions from RankRefine and **RANKREFINE++** under the Gaussianity assumptions. Across *ve* ADMET tasks, the mean absolute difference μ_y is at the level of machine epsilon (Table 2), confirming that Rankrefine is a special case of **RANKREFINE++** with Gaussianity assumptions.

Dataset Name	Half Life	FreeSolv	PPBR	Solubility	VDss
$y \cdot 10^{-7}$	0.028 ± 0.028	0.030 ± 5.71	8.51 ± 1.42	0.029 ± 1.33	3.24 ± 2.96

Table 2: Mean absolute difference between the refined predictions of RankRefine and REREFINE++ with Gaussianity assumptions. The values are around the range of machine epsilon for a floating point precision, confirming that RankRefine is a special case of REREFINE++.

5 LIMITATIONS

While REREFINE++ significantly outperforms prior arts, there are several aspects that can be explored further in future works.

- Noise model for the oracle ranker. We inject pairwise flips uniformly to reach the desired oracle ranker accuracy. Real rankers (human or LLM) might exhibit systematic biases, which could negatively affect the Bayesian inference at lower ranker accuracies.
- Dependence on regressor uncertainty. REREFINE++ requires a base regressor that can quantify uncertainty, which many off-the-shelf regressors lack. A straightforward solution is to use ensembling or Monte Carlo dropout.
- Reference-set selection. Regression improvements may depend on the composition of the reference set (coverage, label diversity, resolution). Future works can explore more sophisticated, sequential sampling strategy to select the references.

6 CONCLUSION

We introduced REREFINE++, framing post training regression enhancement as a problem of Bayesian inference rather than heuristic fusion. This perspective not only generalizes prior state-of-the-art, but also reveals subtle failure modes, explains when and why performance degrades, and offers principled solutions through temperature calibration and gating. Empirically, REREFINE++ delivers consistent gains across 12 diverse datasets, operates effectively with both oracle and noisy LLM rankers, and runs under 1 ms per query. We believe REREFINE++ can serve as a practical bridge between label-scarce applications and the growing availability of pairwise signals from LLMs.

REFERENCES

- Alice EA Allen and Alexandre Tkatchenko. Machine learning of material properties: Predictive and interpretable multilinear models. *Science advances* 8(18):eabm7185, 2022.
- Anthropic. Anthropic api guide. <https://docs.anthropic.com/en/api/overview>, 2025. Accessed: 2025-09-16.
- Ralph Allan Bradley and Milton E Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika* 39(3/4):324–345, 1952.
- Manuela Cattelan. Models for paired comparison data: A review with emphasis on dependent data. *Statistical Science* pp. 412–433, 2012.
- Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. *Advances in neural information processing systems* 30, 2017.
- William G Cochran and Sarah Porter Carroll. A sampling investigation of the efficiency of weighting inversely as the estimated variance. *Biometrics* 9(4):447–459, 1953.
- Paulo Cortez. Student Performance. UCI Machine Learning Repository, 2014.
- Li Di, Christopher Keefer, Dennis O Scott, Timothy J Strelevitz, George Chang, Yi-An Bi, Yurong Lai, Jonathon Duckworth, Katherine Fenner, Matthew D Troutman, et al. Mechanistic insights from comparing intrinsic clearance values between human liver microsomes and hepatocytes to guide drug design. *European journal of medicinal chemistry* 57:441–448, 2012.

- Google. Gemini developer api <https://ai.google.dev/gemini-api/docs>, 2025. Accessed: 2025-09-16.
- Taicheng Guo, Bozhao Nan, Zhenwen Liang, Zhichun Guo, Nitesh Chawla, Olaf Wiest, Xiangliang Zhang, et al. What can large language models do in chemistry? a comprehensive benchmark on eight tasks. *Advances in Neural Information Processing Systems*, 36:59662–59688, 2023.
- Tin Kam Ho. Random decision forests. *Proceedings of 3rd international conference on document analysis and recognition* volume 1, pp. 278–282. IEEE, 1995.
- Eva JI Hoeijmakers, Bibi Martens, Babs MF Hendriks, Casper Muhl, Razvan L Miclea, Walter H Backes, Joachim E Wildberger, Frank M Zijta, Hester A Gietema, Patricia J Nelemans, et al. How subjective ct image quality assessment becomes surprisingly reliable: pairwise comparisons instead of likert scale. *European Radiology*, 34(7):4494–4503, 2024.
- Kexin Huang, Tianfan Fu, Wenhao Gao, Yue Zhao, Yusuf Roohani, Jure Leskovec, Connor Coley, Cao Xiao, Jimeng Sun, and Marinka Zitnik. Therapeutics data commons: Machine learning datasets and tasks for drug discovery and development. *Advances in neural information processing systems*, 2021.
- Qiaohao Liang, Aldair E Gongora, Zekun Ren, Armi Tiihonen, Zhe Liu, Shijing Sun, James R Deaneault, Daniil Bash, Flore Mekki-Berrada, Saif A Khan, et al. Benchmarking the performance of bayesian optimization across multiple experimental materials science domains. *Computational Materials*, 7(1):188, 2021.
- Franco Lombardo and Yankang Jing. In silico prediction of volume of distribution in humans. extensive data set and the exploration of linear and nonlinear methods coupled with molecular interaction fields descriptors. *Journal of chemical information and modeling*, 16(10):2042–2052, 2016.
- Alexander Ly, Maarten Marsman, Josine Verhagen, Raoul PPP Grasman, and Eric-Jan Wagenmakers. A tutorial on sher information. *Journal of Mathematical Psychology*, 90:40–55, 2017.
- David L Mobley and J Peter Guthrie. Freesolv: a database of experimental and calculated hydration free energies, with input les. *Journal of computer-aided molecular design*, 18:711–720, 2014.
- Frederick Mosteller. Remarks on the method of paired comparisons: iii. a test of significance for paired comparisons when equal standard deviations and equal correlations are assumed. *Biometrika*, 16(2):207–218, 1951.
- Kevin P Murphy. *Probabilistic machine learning: an introduction*. MIT press, 2022.
- R Scott Obach, Franco Lombardo, and Nigel J Waters. Trend analysis of a database of intravenous pharmacokinetic parameters in humans for 670 drug compounds. *Drugs, Metabolism and Disposition*, 36(7):1385–1405, 2008.
- OpenAI. Openai api reference. <https://platform.openai.com/docs/api-reference/introduction>, 2025. Accessed: 2025-09-16.
- Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.
- Zhen Qin, Rolf Jagerman, Kai Hui, Honglei Zhuang, Junru Wu, Le Yan, Jiaming Shen, Tianqi Liu, Jialu Liu, Donald Metzler, et al. Large language models are effective text rankers with pairwise ranking prompting. *Findings of the Association for Computational Linguistics: NAACL 2024* pp. 1504–1518, 2024.
- Jennifer Routh, Sharmini Julita Paramasivam, Peter Cockcroft, Sarah Wood, John Remnant, Corélie Westermann, Alison Reid, Patricia Pawson, Sheena Warman, Vishna Devi Nadarajah, et al. Rating and ranking preparedness characteristics important for veterinary workplace clinical training: a novel application of pairwise comparisons and the elo algorithm. *Frontiers in Medicine*, 10:1128058, 2023.

- David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986.
- Benjamin Sanchez-Lengeling and An Aspuru-Guzik. Inverse molecular design using machine learning: Generative models for matter engineering. *Science*, 361(6400):360–365, 2018.
- Adil Shamim. Cost of international education, 2025. URL <https://www.kaggle.com/datasets/adilshamim8/cost-of-international-education>
- Murat Cihan Sorkun, Abhishek Khetan, and Ölsüman Er. Aqsolddb, a curated reference set of aqueous solubility and 2d descriptors for a diverse set of compounds. *Scientific data*, 6(1):143, 2019.
- Atharva Soundankar. Smart farming sensor data for yield prediction, 2025. URL <https://www.kaggle.com/datasets/atharvasoundankar/smart-farming-sensor-data-for-yield-prediction>
- Michael Sun, Gang Liu, Weize Yuan, Wojciech Matusik, and Jie Chen. Foundation molecular grammar: Multi-modal foundation models induce interpretable molecular graph languages. In International Conference on Machine Learning, 2025.
- LL Thurstone. A law of comparative judgment. *Psychological Review*, 34(4):273–286, 1927.
- Ning-Ning Wang, Jie Dong, Yin-Hua Deng, Min-Feng Zhu, Ming Wen, Zhi-Jiang Yao, Ai-Ping Lu, Jian-Bing Wang, and Dong-Sheng Cao. Adme properties evaluation in drug discovery: prediction of caco-2 cell permeability using a combination of nsga-ii and boosting. *Journal of chemical information and modeling*, 16(4):763–773, 2016.
- David Weininger. Smiles, a chemical language and information system. 1. introduction to methodology and encoding rules. *Journal of chemical information and computer sciences*, 28(1):31–36, 1988.
- Kevin Tirta Wijaya, Minghao Guo, Michael Sun, Hans-Peter Seidel, Wojciech Matusik, and Vahid Babaei. Two-stage pretraining for molecular property prediction in the wild. *arXiv preprint arXiv:2411.03537*, 2024.
- Kevin Tirta Wijaya, Michael Sun, Minghao Guo, Hans-Peter Seidel, Wojciech Matusik, and Vahid Babaei. Post hoc regression refinement via pairwise ranking. *arXiv preprint arXiv:2508.16495*, 2025.
- Tong Wu, Guandao Yang, Zhibing Li, Kai Zhang, Ziwei Liu, Leonidas Guibas, Dahua Lin, and Gordon Wetzstein. Gpt-4v (ision) is a human-aligned evaluator for text-to-3d generation. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 2227–2238, 2024.
- Zhenqin Wu, Bharath Ramsundar, Evan N Feinberg, Joseph Gomes, Caleb Geniesse, Aneesh S Pappu, Karl Leswing, and Vijay Pande. Moleculenet: a benchmark for molecular machine learning. *Chemical science*, 9(2):513–530, 2018.
- Le Yan, Zhen Qin, Honglei Zhuang, Rolf Jagerman, Xuanhui Wang, Michael Bendersky, and Harrie Oosterhuis. Consolidating ranking and relevance predictions of large language models through post-processing. In Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, pp. 410–423, 2024.
- Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in neural information processing systems*, 36:46595–46623, 2023.
- Daniel M Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*, 2019.

A APPENDIX

A.1 DETAILED PROOFS

Lemma 3.1. Proof. The regressor log-likelihood contributes $-\frac{1}{2} \frac{1}{\sigma^2} (y_0^{\text{re}} - y_0)^2$ with curvature $1 = \frac{2}{\sigma^2} < 0$. Each BT term adds $\log s(y_0 - y_i)$ or $\log(1 - s(y_0 - y_i))$, whose second derivative is $-s(1-s) > 0$. Sum of concave terms and a concave prior is concave; the Gaussian term's strictly negative curvature makes the sum strictly concave.

Lemma 3.4. Proof. $L_{\text{BT}}^0(y)$ (Proposition 3.3.) is strictly decreasing, therefore $L_{\text{BT}}(y)$ is strictly concave. Setting $L_{\text{BT}}^0(y) = 0$ gives the equation. Since $F(y) = \sum_{i=1}^k u_i(y)$ is continuous and strictly increasing with $F(-1) = 0$, $F(1) = k$, there is a unique solution at the observed count

Let $y_1 \leq \dots \leq y_k$ be the ordered reference set and suppose they are in $(y_m; y_{m+1})$,

- if most $(y_0 - y_i)$ are in the sigmoid saturated regime, F behaves like a hard count, so $F(y_0) \approx m$, $F(y_0) = m$, and $y_0^{\text{ra}} = y_0 - y_0$ up to the resolution $(y_{m+1} - y_m)$.
- if many $(y_0 - y_i)$ lie in the transition region, $F(y_0) \approx m$, $F(y_0) = m$, and $y_0^{\text{ra}} = y_0 - \delta y_0$. This shift does not induce rank bias as long as $\delta < (y_m; y_{m+1})$.
- rank bias arises if the shift exits the interval, i.e., $y_0 - y_j < \min(y_0 - y_m; y_{m+1} - y_0)$, which is more likely when the number of $(y_0 - y_i)$ that lie in the transition region is high.

Lemma 3.6. Proof. Let $I_{\text{reg}} = 1 = \frac{2}{\sigma^2}$ be the Fisher information of the regressor term and write the total gradient and Fisher information at y as

$$\begin{aligned} g_{\text{tot}}(y) &= g_{\text{reg}}(y) + g_{\text{rank}}(y) \\ &= -\frac{1}{\sigma^2} (y - y_0^{\text{re}}) + \sum_{i=1}^k (r_i - u_i(y)); \end{aligned} \quad (20)$$

$$I_{\text{tot}}(y) = I_{\text{reg}} + I_{\text{rank}}(y); \quad (21)$$

Define the local rank target for a single Newton step

$$\begin{aligned} y^{\text{ra}}(y) &= y + \frac{g_{\text{rank}}(y)}{I_{\text{rank}}(y)} \\ &= y + \frac{\sum_{i=1}^k (r_i - u_i(y))}{I_{\text{rank}}(y)}. \end{aligned} \quad (22)$$

By substituting $g_{\text{reg}}(y) = I_{\text{reg}} (y_0^{\text{re}} - y)$ and $g_{\text{rank}}(y) = I_{\text{rank}}(y) (y^{\text{ra}}(y) - y)$, a single Newton step is an information-weighted average as written in Equation 12.

Lemma 3.7. Proof. Using $E[r_i] = (1 - a) + (2a - 1)1(y_0 > y_i)$ to model lip errors in a noisy binary ranker gives

$$E[g_{\text{rank}}(y)] = \sum_{i=1}^k E[r_i] - \sum_{i=1}^k u_i(y) = k((1 - a) + (2a - 1)p - p(y)); \quad (23)$$

If $p = 1/2 - p(y)$ or $p = 1/2 + p(y)$, then

$$E[g_{\text{rank}}(y)] = k(2a - 1)j p - p(y)j; \quad (24)$$

Substituting $g_{\text{rank}}(y)$ with $1 = \frac{2}{\sigma^2} (y - y_0^{\text{re}})$, the expected Newton step is rank-dominated whenever

$$(2a - 1)j p - p(y)j > \frac{jy - y_0^{\text{re}}j}{k \frac{2}{\sigma^2}}. \quad (25)$$

A.2 USE OF LARGE LANGUAGE MODELS

We use large language models (LLMs) to (i) polish writing (restructuring sentences and proofread grammars and typos), (ii) search for related works, (iii) predict pairwise rankings of molecule pairs. For writing and search, we use ChatGPT5. For pairwise ranking predictions, we use ChatGPT5 and Claude Sonnet 4 (Copilot version). The prompt that we used for pairwise ranking prediction is as follow,

You are an expert molecular reasoning model tasked with predicting pairwise rankings of molecules based on a described molecular property of interest (e.g., solubility, polarity, etc.).

You will be given json files containing the test molecules to be compared with the reference molecules. The property of interest is described within the json files with the key "description".

* Your Task - Follow These Steps:

1. Read the dataset's description to understand the molecular property being ranked (e.g., "higher solubility", "lower toxicity").
 - Be careful with the measurement unit. For example, a higher number in IC50 could mean lower toxicity.
2. For each molecule pair (test molecule vs. reference molecule):
 - Use your internal knowledge to infer which molecule ranks higher for the property.
 - You may use structural patterns, substrings, atom types, SMARTS-like features, token-level patterns, or other insights you may have.
 - You are encouraged to develop your own heuristics or scoring logic using Python.
3. Assign a "pairwise_rank" to each pair:
 - 1 test molecule ranks higher than reference, meaning $\text{test_property} > \text{reference_property}$
 - 0 test molecule ranks lower or equal to reference

Caution! Only care for the value of the property. For example, if toxicity is measured in IC50, and test_molecule IC50 value is greater than reference_molecule IC50 value, you should output 1.
4. Save your results as a new JSON file with similar structure.

* You Are Allowed To:

1. Write your own Python logic.
2. Use basic Python and string-based pattern recognition
3. Think step-by-step to develop useful ranking heuristics
4. Use helper functions from Cheminformatics libraries, as long as you do not use them to directly predict the property of interest values.

* You Are NOT Allowed To:

1. Use the internet to search for the property values
2. Use cheminformatics libraries like RDKit to directly predict the property of interest values, e.g., solubility of molecule A for solubility datasets.
3. Access files not specified in this prompt

* Each entry in your output JSON should look like this:

```
"test_molecule": "smiles": "...",  
"reference_molecules": [  
  {  
    "id": "...",  
    "smiles": "..."  
  },  
  {  
    "id": "...",  
    "smiles": "..."  
  }  
],  
"pairwise_ranks": 1,
```

},
 that is, put the pairwise_rank predictions inside the reference_molecule. Your output json file should be named "pairwise_ranking_predictions_{split_id}.json"

* Tips for Better Performance:

1. Think aloud: before you begin ranking, describe what is the property of interest and why one molecule might rank higher based on your knowledge
2. Use token or substring patterns (e.g., "more OH groups" or "more aromatic rings")
3. Define scoring rules: e.g., "count('O') - count('N')" to estimate polarity

A.3 EFFECTS OF NUMBER OF SAMPLES FOR TEMPERATURE CALIBRATION

In Section 3.3, the temperature is set to \hat{t}_{cal} , the calibrated value estimated from the labeled reference set. A natural question is the robustness of this procedure in data-scarce regimes, where only a few labeled references are available. Figure 6 shows \hat{t}_{cal} as a function of the calibration set size n_{cal} . Across the 9 TDC ADMET datasets, \hat{t}_{cal} converges at around $n_{cal} = 10$, indicating that optimal calibration can be obtained with only 10 labeled samples. Furthermore, except for PBR, the estimates obtained with $n_{cal} < 10$ are already close to their converged values. This suggests that \hat{t}_{cal} remains reliable even when very limited calibration data are available. Consistent with this observation, in the experiment where we fix the reference set size $n_{ref} = 50$ but vary $n_{cal} \in [2, 3, 50]$ (Figure 7), the resulting performance curves are nearly identical. Together, these results demonstrate that the number of references used for temperature calibration has a negligible effect on overall regression enhancement performance.

A.4 REGRESSION ENHANCEMENT FOR MULTIPLE TARGETS

The discussion and experiments in Sections 3 and 4 focus on scalar regression tasks. In many practical settings, however, the goal is to predict multiple properties simultaneously, i.e., to produce vector-valued outputs. Extending RANKREFINE++ to this setting is straightforward: one can assume independence across output dimensions and apply RANKREFINE++ separately to each component of a target vector \mathbf{R}^d , yielding d independent regression enhancement processes.

In practice, the components y_i are often correlated. For example, in molecular chemistry, aqueous solubility and membrane permeability are often related, as are clearance and half-life. In such cases, dependencies among outputs can be captured by employing a low-rank approximation of the covariance structure, which enables a computationally efficient implementation. Exploring this extension is an interesting direction for future work.

A.5 MORE RESULTS ON TDC ADMET

We show results for more values in Figure 8-Figure 16. In general, RANKREFINE++ can improve the regression error (i.e., $\epsilon < 1$) with a reference set size as small as 3.

Figure 6: Effects of number of samples used to calibrate the temperature on the value of \hat{t}_{cal} .

Figure 7: Effects of number of samples on the regression enhancement. We measure (lower is better) as a function of number of samples used to calibrate the temperature. The reference set size k is set to 50.

Figure 8: More results with varying reference set size for the Caco2Wang dataset.

Figure 9: More results with varying reference set size for the ClearanceHepatocyteAZ dataset

Figure 10: More results with varying reference set size for the ClearanceMicrosomeAZ dataset

Figure 11: More results with varying reference set size for the HalfLife_Obach dataset

Figure 12: More results with varying reference set size for the HydrationFreeEnergyFreeSolv dataset

Figure 13: More results with varying reference set size for the Lipophilicity_AstraZeneca dataset

Figure 14: More results with varying reference set size for the PPBRAZ dataset

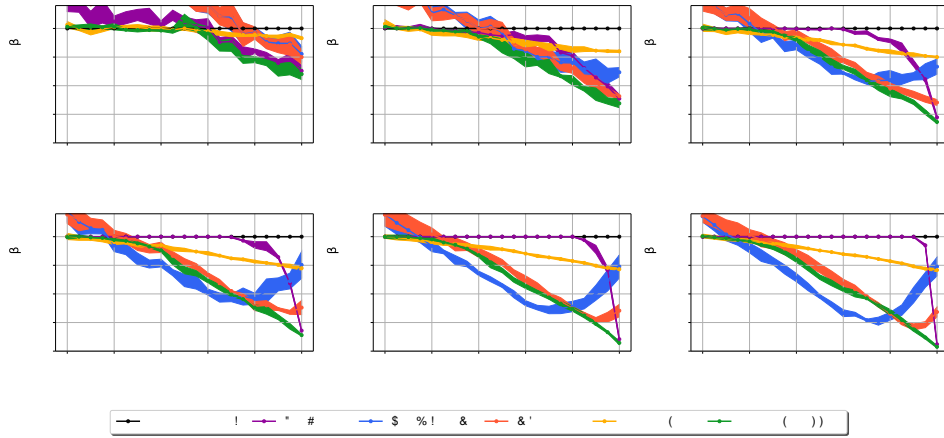


Figure 15: More results with varying reference set size (k) for the Solubility_AqSolDB dataset

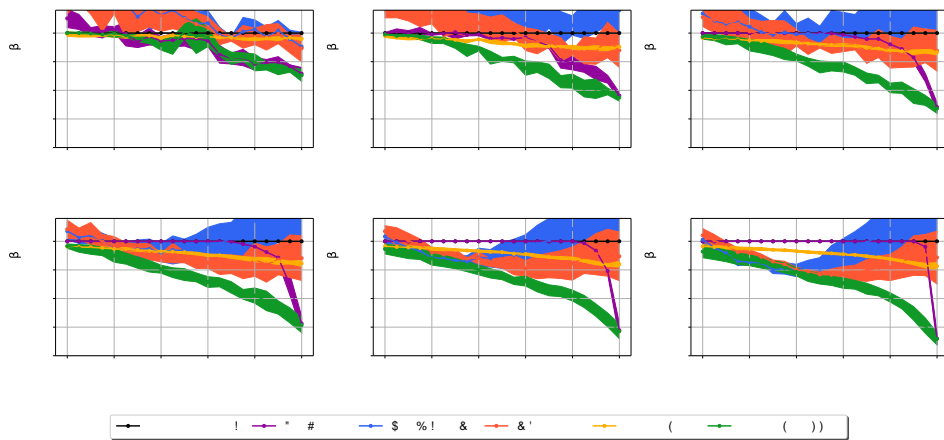


Figure 16: More results with varying reference set size (k) for the VDss.Lombardo dataset

A.6 THE CORE BAYESIAN FORMULATION OF RANKREFINE++ WITHOUT CALIBRATION OUTPERFORMS PRIOR WORKS

Calibration and soft-gating offer a principled way to maximize the performance of RANKREFINE++. Nevertheless, the core Bayesian formulation of RANKREFINE++ already brings substantial improvement over the baselines by itself. As shown in Figure 17, plain RANKREFINE++ without calibration or soft-gating outperforms both RankRefine and Projection across the pairwise ranking accuracy.

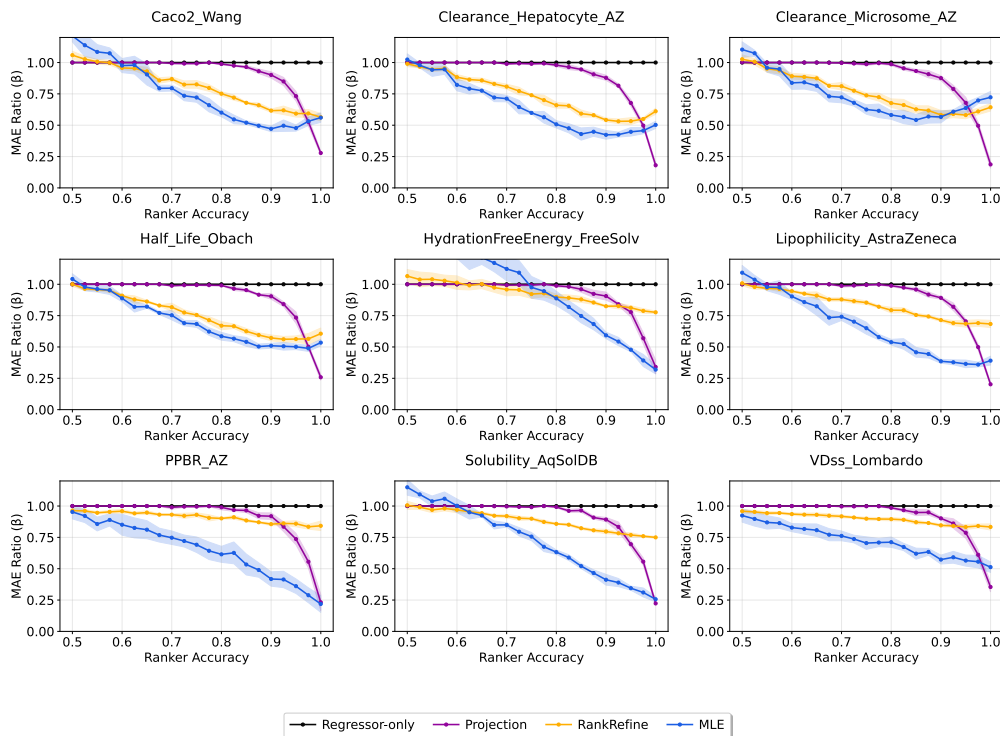


Figure 17: Plain RANKREFINE++ (MLE variant) vs. RankRefine and Projection. Reference set size $k = 20$. Non-calibrated RANKREFINE++ consistently achieves lower error compared to RankRefine and Projection baselines across datasets, often by a large margin

A.7 SENSITIVITY ANALYSIS FOR UNDER/OVERESTIMATION OF REGRESSOR UNCERTAINTY

Figure 18 shows the performance of RANKREFINE++ when the pointwise uncertainty of the regressor is set to the pointwise absolute or squared error, scaled by s . Overestimation (e.g., $s = 10.0$) has minimal impact on the performance, while underestimation brings substantial degradation due to excessive confidence in the imperfect regressor.

A.8 SENSITIVITY ANALYSIS FOR NOISY ESTIMATION OF THE RANKER ACCURACY

The soft-gating mechanism relies on the estimation of the ranker accuracy. Figure 19 shows the effect of noisy estimation of the ranker accuracy. When injecting the ground-truth ranker accuracy ($a \in [0.0, 1.0]$) with a Gaussian noise, there is no substantial performance degradation up to $\sigma^2 = 0.50$. Furthermore, for $\sigma^2 > 0.50$, the performance starts to slightly degrade only at relatively high ranker accuracy, demonstrating the robustness of RANKREFINE++ in most real-world scenarios.

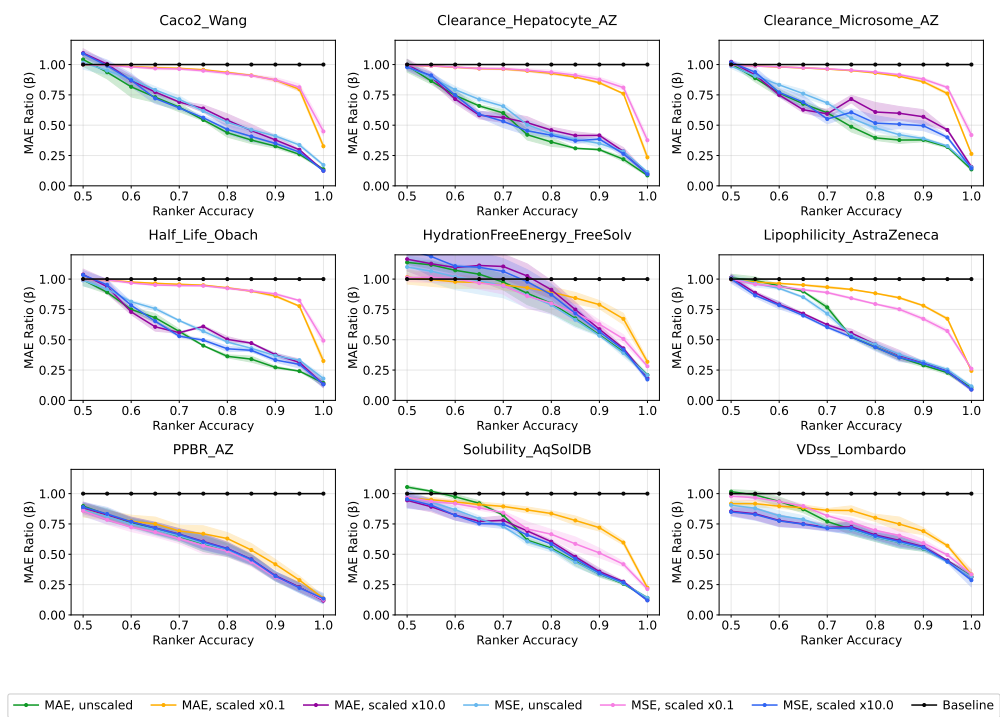


Figure 18: Sensitivity analysis on under or overestimation of regressor uncertainty. Overestimating uncertainty (e.g., $s = 10.0$) has minimal impact on performance, while underestimating it (e.g., $s = 0.1$) degrades performance due to excessive confidence in the regressor.

