清单比奖励模型更适合校准语言模型

Vijay Viswanathan ♡ **Yanchao Sun**♣ **Shuang Ma**♣* **Xiang Kong**♣
Meng Cao♣ **Graham Neubig**♡ **Tongshuang Wu**♡
♡ Carnegie Mellon University ♣Apple

Abstract

语言模型必须进行适应，以理解和遵循用户指令。强化学习被广泛用于促进这一过程——通常使用诸如“有用性”和“有害性”等固定标准。在我们的工作中，我们提出了使用灵活的、特定于指令的标准，作为扩展强化学习在引导指令跟随方面影响的一种手段。我们提出了“从清单反馈中进行强化学习”（RLCF）。从指令中，我们提取清单并评估响应对每项的满足程度——使用 AI 评审员和专门的验证程序，然后结合这些分数来计算 RL 的奖励。我们将 RLCF 与应用于一个强大的指令跟随模型（`Qwen2.5-7B-Instruct`）的其他对齐方法进行了比较，在五个被广泛研究的基准上——RLCF 是唯一在每个基准上提高性能的方法，包括在 FollowBench 上提高了 4 点的严格满意率，在 InFoBench 上增加了 6 点，以及在 Arena-Hard 上提高了 3 点的胜率。这些结果确立了清单反馈作为提高语言模型支持表达多样化需求的查询的关键工具。[2]

1 引言

语言模型必须遵循用户指令才能发挥作用。随着公众将基于语言模型的助手整合到他们完成日常任务中，他们期望语言模型能够忠实地遵循用户的请求 [Liu et al., 2024a] 。随着用户对模型能够完成复杂请求的能力越来越有信心，这些模型被赋予了越来越多的丰富的、多步骤的指令，这需要仔细关注规范 [Zhao et al., 2024, Zheng et al.] 。

如今的模型几乎普遍通过一个两步流程来训练以遵循指令：首先是指令微调，然后是通过人类反馈的强化学习（RLHF）。指令微调是指模型被训练以模仿由标注者生成的响应 [Raffel et al., 2019] ，历史上一直是赋予语言模型一定指令遵循能力的主要手段 [Wang et al., 2022, Chung et al., 2022, Xu et al., 2024, Lambert et al., 2024a] 。然后，模型开发者经常采用 RLHF，这一步是训练模型生成看起来更像标注为“好”而非“坏”的响应，作为减少模型表现出预定不良行为（通常是有害行为）可能性的一个改进步骤 [Ziegler et al., 2019, Bai et al., 2022] 。与“可验证”任务（其中强化学习是一个有效工具）不同 [DeepSeek-AI et al., 2025, Lambert et al., 2024a, Pyatkin et al., 2025] ，强化学习在模棱两可或“不可验证”任务中仍然难以利用，比如指令遵循。在主观或开放式环境中，要让强化学习成为提取理想行为的通用解决方案需要什么？

我们认为解决方案必须涉及生成更好的奖励信号。最近关于语言模型对齐的强化学习工作集中于自动获取关于模型行为的反馈，方法包括 (1) 仅使用具有可验证答案的指令 [Dong et al., 2024, Pyatkin et al., 2025] ，(2) 使用专门训练的奖励模型 [Wang et al., 2024a, Eisenstein et al., 2023] 对响应进行评分，或 (3) 从更大的提示模型中提取偏好 [Bai et al., 2022, Tunstall et al., 2023] 。使用具有可验证答案的指令限制了可以学习的响应质量

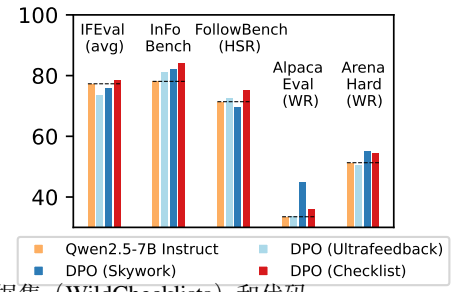

Figure 1: 在 Checklist 反馈上的 RL 一致提升了 Qwen2.5 7B 指令，而其他自动反馈来源则给出了混合结果。

*Work performed while at Apple.

[2]我们计划很快向公众发布我们的模型、我们的清单数据集（WildChecklists）和代码。

Preprint. Under review.

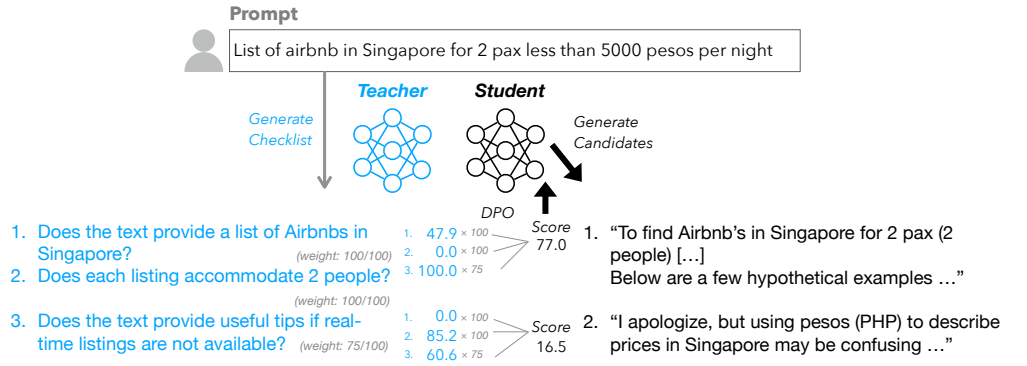

Figure 2: 我们提出了来自检查表反馈的强化学习，其中采样的响应由基于固定标准集的教师模型进行评估。在我们的流程中，给定指令后，我们首先从指令中合成生成检查表，对每个检查表项对每个响应进行评分，将每项得分组合成一个加权的检查表总分，然后将此分数用于强化学习。

的方面，仅限于准确性或语法/格式的遵循（忽略其它品质，例如主题性或风格）。专门训练的奖励模型虽然功能强大，但其对奖励的概念可能是任意的，可能导致奖励作弊 [Eisenstein et al., 2023] 。当从更大的提示语言模型中提取偏好时，该语言模型面临决定在评分响应时考虑哪些方面的问题，从而减少了所谓的 "生成器-验证器差距"，这使得强化学习成为可能 [Swamy et al., 2025] 。即使编写了多个提示来捕获感兴趣的价值观，这也假设了一组固定标准可以是全面的 [Bai et al., 2022, Glaese et al., 2022a] 。

在本文中，我们提出的问题是："我们如何能够以自动化（不需要人工标注）、灵活（考虑响应质量的所有方面）、直观（与响应间的明显差异一致）、且适用于任何指令或响应的方式来给指令的响应评分，从而在语言模型对齐中更有效地使用强化学习？" 为此，我们提出从指令中提取动态检查清单，该方法我们称之为基于检查清单反馈的强化学习（RLCF）。这种方法使得我们的评估更可能集中在灵活的不同标准列表上，同时也将评分响应的问题简化为一系列特定的是/否问题，这些问题可以由 AI 裁判回答或通过执行验证程序来回答。

我们的主要贡献是：

1. 我们描述了一种用于大规模自动生成检查清单的新改进算法。
2. 我们构建了 WildChecklists，这是一个由 130,000 条指令及相应的清单（通过合成生成）组成的数据集。在适用的情况下，我们为每个清单中的项目配备了一个验证程序，以便于自动评估。我们计划将此数据集作为一种工件发布给社区，以供未来研究使用。
3. 我们描述了一种新的算法，用于根据检查表对回答进行评分，使用语言模型和代码，并展示如何使用该算法对响应进行排序以优化偏好设置。
4. 我们通过使用 WildChecklists 的检查表反馈对 `Qwen2.5-7B-Instruct` 进行强化学习微调，从而得到一个强大且改进的 7B 参数模型用于指令遵循。

在涵盖约束指令遵循（IFEval、InFoBench、FollowBench）和一般对话辅助（AlpacaEval、Arena-Hard）的 5 个基准测试中，我们发现 RLCF 在所有指令遵循基准中提供了优势，同时在一般对话辅助基准中保持了改进的性能。相反，所有替代形式的 AI 反馈导致的结果参差不齐，如 Figure 1 所示。在 FollowBench 的平均困难满足率上，RLCF 相较于 Qwen2.5-7B-Instruct 提供了 5.4 % 的相对提升，在 InFoBench 整体要求遵循率上有 6.9 % 的相对提升，并且在 Arena-Hard 上有 6.4 % 的相对提升 [Jiang et al., 2023, Qin et al., 2024, Li et al., 2024] 。尽管有这些显著的改进，RLCF 仅需要一个教师模型，无需额外的数据或人为注释，使这种方法适用于多种语言或领域。我们提供的证据表明，基于清单的奖励与人类偏好判断良好相关（与许多微调的奖励模型相当），同时提供比替代方法更强的学习信号。

2 清单生成

对清单的要求。我们将清单定义为一系列配有指令的要求，满足以下特性：

1. 清单中的每一项要求都是一个是/否问题（例如："文本中是否包含 3 个逗号?"）。

Metric	Manual Evaluation		Automatic Evaluation	
	Direct	Candidate-Based	Direct	Candidate-Based
Naturalness	94.9	93.9	88.0	85.1
Objectiveness	88.5	91.9	88.9	89.7
Comprehensiveness	74.0	82.0	69.2	64.8
Atomicity	68.0	90.0	98.6	99.0
% Preferred Overall	38.0	56.0	40.6	51.2

Table 1: 我们从四个具体的质量方面以及整体偏好上评估两种核对表生成方法。手动评估是在 InFoBench "easy" 的前 50 行上进行的，而自动评估由 `gpt-4o` 在 InFoBench 的所有 500 行上进行。

2. 检查表中的每项要求必须相对于给定的候选答案进行回答。
3. 只有当回应对所有核对清单要求都回答"是"时，该回应才会被认为是可接受的。

为了满足定义 # 3，清单必须是全面的（涵盖大多数相关的质量方面）和自然的（由其相应的指令引出）。基于这样的观察，即假阳性奖励通常比假阴性对强化学习更有害，我们希望清单是客观的（便于自动验证）和原子的（每个要求专注于质量的单一方面），以使要求检查更加容易。

根据指令提取清单。我们研究了两种提取清单的方法：

- 我们简单地提示一个语言模型从给定的指令 [Cook et al., 2024] 中提取一个清单。这种方法直观而简单，但存在通过这些单独的标准重复原始指令的风险，这可能会限制其全面性和客观性。
- `Candidate-based`：我们将要求视为指令的任何方面，当缺失时会导致响应失败。我们提出了一种两阶段的方法：首先产生不同质量的响应，然后提示语言模型编写所有可能失败模式的检查表。对于每个检查表项目，我们还提示模型生成一个"重要性"权重（从 0 到 100）。

为了进行比较，我们为 InFoBench [Qin et al., 2024] 中的所有指令生成检查清单。我们使用 `gpt-4o` 从自然性、客观性、全面性和原子性方面对每个检查清单进行盲评估，然后整体上选择较好的一个。我们手动对 InFoBench 的"简单集"中的 50 条指令的一个子集进行相同的评估。

Table 1 中的结果显示，直接通过提示 LLM 生成的清单更加自然。然而，向 LLM 提供候选响应，会生成具有一致性更好的客观性、原子性和总体质量的清单。在两次评估中的分数之间存在绝对差异——部分原因是使用了不同的子集——但方向性趋势是一致的。我们发现，经过 RL 训练后，这种差异会转化为下游性能。在 **??** 中，我们展示了通过候选方法生成的清单上的清单反馈强化学习更有效。

通过普遍标准进行正则化。在最初的实验中，我们发现优化完成检查表会导致模型有时生成响应的高级概述，而不是预期的响应，暗示了奖励欺骗。在之前的工作中，Sun et al. [2023] 报告了类似的问题，当使用可引导的奖励模型训练模型时，通过在所有情况下向他们的奖励模型中添加三个手动选择的指令来解决这个问题。遵循这一思路以及其他使用全球原则进行强化学习的研究 [Glaese et al., 2022b, Bai et al., 2022] ，我们在所有生成的检查表中添加了一个"普遍要求"。这个普遍要求表述为："响应是否满足以下两个标准：1) 响应直接解决请求，而没有过多或不必要的与用户指令无关的信息？2) 响应应匹配上下文和指令，无论其要求专业性、友好性、正式性或中立性。"，相应的重要权重为 100/100。

数据集生成：使用基于候选的方式，我们为来自 WildChat 的 130,000 个指令生成清单，以创建一个新数据集，WildChecklists。为了生成我们方法的候选响应，我们使用 `Qwen2.5-0.5B`、`Qwen2.5-1.5B`、`Qwen2.5-3B` 和 `Qwen2.5-7B` [Yang et al., 2024] 。`Qwen2.5-72B-Instruct` 是这两种方法的清单生成模型。

3 从检查清单反馈中进行强化学习

给定 WildChecklists，我们通过四步流程为 RL 生成高质量的偏好数据：

采样候选响应。为了促进离线强化学习，我们首先从我们的基本策略中采样响应对。对于每个提示，我们以 1.3 的温度和 0.9 的 top-p 采样两个响应。这比之前基于强化学习的语言模型对齐工作中通过人工扰动提示以引发更复杂性的方法更简单。

灵活评分 给定一个提示、一个回应和一个单独的核对清单项目，我们使用一个 LM 评判和一个验证程序的组合来对回应进行评分。对于每个核对清单项目，评判模型（`Qwen2.5-72B-Instruct`）生成一个介于 0 到 100 之间的数值分数。我们用于评分的提示在附录 B 中展示。为了减少这个分数的方差，我们从模型中采样 25 个数值分数，然后取这 25 个分数的平均值 [3]。

在适用的情况下，我们还使用验证程序来进行评分。LLMs 在评估衡量文本离散性质的标准时存在困难，例如“回答是否至少包含三个字母 R?”或“回答是否包含以下关键词之一 [...]?”[Fu et al., 2024]。为了更好地处理这些约束，我们遵循先前的工作，在适用时生成验证程序 [Dong et al., 2024, Zhou et al., 2023]。我们的提示列在 **??**，仅在模型可以高度自信地通过程序精确验证要求时，才要求模型生成代码。如果程序成功处理了一个响应字符串，我们将布尔结果转换为整数（0 或 100），并与 AI 裁判的得分平均 [4]。

偏好调整。对于每个响应的每个标准给定一个单独的数值评分，我们采用这些评分的加权平均值，以每个标准生成的重要性评分作为权重。为了产生更具信息性的学习信号，我们只保留响应对中，在其相应检查表的至少一个标准上差异最大的 40 %。这除去了在质量上过于相似而无法提供有用奖励信号的响应对。然后我们将分数较高的响应标记为“选中”，分数较低的标记为“拒绝”，并将这些作为直接偏好优化 [Rafailov et al., 2023] 的偏好对。

4 实验设置和结果

4.1 实验细节

训练数据作为所有方法的固定指令来源，我们使用 WildChat，这是从全球用户众包而来的用户与 AI 语言模型之间的自然对话集合 [Zhao et al., 2024]。我们过滤掉非英语、有害或超过两个回合的对话。

模型我们进行实验以微调 `Qwen2.5-7B` 和 `Qwen2.5-7B-Instruct`。为了产生 AI 判断或真实响应，除非另有说明，我们使用 `Qwen2.5-72B-Instruct`。

训练我们使用 DPO 对模型进行了微调，训练 2 个 epoch，批量大小为 1024，最大序列长度为 2048。我们使用余弦学习率调度，最大学习率为 3e-6，最小学习率为 2e-6 [5]。我们使用 OpenRLHF 进行训练 [Hu et al., 2024]，并在一个 8xH100 节点上进行训练，节点具有 80GB 的 GPU 内存，每个模型大约需要 3 小时。

基准数据我们在五个基准上评估我们的方法：IFEval [Zhou et al., 2023]，InFoBench [Qin et al., 2024]，FollowBench [Jiang et al., 2023]，AlpacaEval [Dubois et al., 2024] 和 Arena-Hard [Li et al., 2024]。前三者衡量在细粒度约束条件下的指令遵循能力。后两者通过基于在自然环境中收集的用户查询的自然指令来衡量“通用”指令遵循能力。

4.2 基线

为了证明 RLCF 比现有方法更有效，我们与基线方法进行了比较：指令微调、专门训练的奖励模型（使用单一奖励或奖励混合）、以及提示的 AI 评判（使用单一评估标准或标准混合）。

指令微调：我们与指令微调进行比较，以区分额外知识的好处是来自给定方式（真实值或奖励）。在这里，我们从一个更大的模型 `Qwen2.5-72B-Instruct` 中提取 [Hinton et al., 2015]，该模型通过 `LlamaFactory` [Zheng et al., 2024] 进行微调。

奖励模型: 我们采用与从核对清单反馈中学习相类似的训练方法，但使用最先进的奖励模型来决定应选择或拒绝哪个响应。在这里，我们保留了在标量奖励上差异最大的 40 % 个提示和响应。我们将以下奖励模型视为基线（`Skywork/Skywork-Reward-Gemma-2-27B`，[Liu et al., 2024b] 和 `ArmoRM-Llama3-8B-v0.1`）[Wang et al., 2024b]。这两个模型在 RewardBench

[3] 我们使用 vLLM [Kwon et al., 2023] 中的 `n` 参数对响应进行采样。这种方法遵循之前的工作，该工作描述了使用平均得分而不是来自 LM-as-a-judge 模型的众数得分的重要性。不管怎样，这使得 AI 法官组件成为我们流程的计算瓶颈。在 Section 5.4 中，我们展示了可以在精度小幅下降的情况下显著减少 `n`。

[4] 这种方法比最相关的先前工作更简单，该工作使用程序来评估响应，即 AutoIF [Dong et al., 2024]，它使用测试用例生成和基于 LM 的过滤器来去除低质量的程序。

[5] 在使用 Ultrafeedback 训练模型时，我们使用了最低学习率 3e-7。我们发现，这个参数在从这种反馈中学习时能产生稍微更好的基线效果。

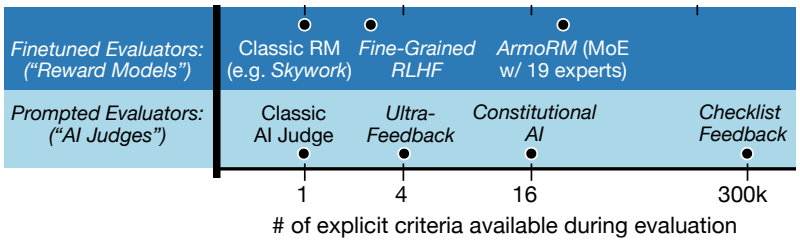

Figure 3: 清单反馈可以被视为评估者的极端混合体，其中（提示的）评估者的空间是无限的，并且为每个指令选择一个独特的评估者子集。

	IFEval (prompt)		IFEval (inst.)			InFoBench		
	Loose	Strict	Loose	Strict	Avg	Easy	Hard	Overall
GPT-4	79.3	76.9	85.4	83.6	81.3	89.3	86.4	87.3
+ Qwen2.5-7B-Instruct	75.0	72.5	81.8	79.9	77.3	82.7	76.0	78.1
+ SFT (Distilled)	66.9	64.1	75.3	72.8	69.8	79.9	70.6	73.5
+ DPO (via Skywork)	75.8	68.0	83.2	78.5	76.0	81.0	82.4	82.0
+ DPO (via ArmoRM)	73.8	70.2	81.7	78.3	76.0	84.2	83.1	83.5
+ DPO (via Ultrafbk.)	71.5	69.1	79.9	77.7	74.6	82.3	79.0	80.0
+ DPO (via AI Judge)	73.0	68.9	80.9	77.8	75.2	81.0	73.9	76.1
+ DPO (RLCF)	77.3	72.6	84.1	80.3	78.6	84.2	84.0	84.1
Qwen2.5-7B (base)	35.7	30.5	46.6	42.1	38.7	68.8	77.4	74.8
+ SFT on WildChat	38.1	33.5	52.2	48.6	43.1	78.1	80.1	79.5
+ DPO (RLCF)	43.4	35.9	56.4	49.2	46.2	80.6	80.5	80.5

Table 2: RLCF 在格式为基础的受限指令遵循基准测试（IFEval）中略微提升了性能，而在开放式受限指令遵循基准测试（InFoBench）中显著提升了性能。使用现成的奖励模型的奖励进行强化学习有助于提升 InFoBench 的表现，但对 IFEval 却有负面影响。我们在 蓝色 中显示了积极的结果（相对于基准），在 橙子 中显示了负面结果，在 灰色 中显示了中立的结果（在 0.5 以内）；给定模型的最佳变体用粗体显示。

[Lambert et al., 2024b] [6] 上均获得了很高的评价，并且 ArmoRM 在之前的研究中对于对齐非常有效 [Meng et al., 2024] 。

提示的 AI 评判：最后，我们将使用相同的“教师”模型作为评判进行比较，而不使用清单。我们在两种情境下查询这个教师：1）“超反馈”，在这里，评判分别对候选回应在四个质量方面（遵循指令、乐于助人、真实性、诚实性）进行 1-5 的评分，并平均这些分数；以及 2）AI 评判，这里使用与 RLCF 几乎相同的提示（§3 ）来从评判中类似地抽取 25 个在 0 到 100 之间的分数。这种方法与 RLCF 一样使用 AI 评判，只是不使用清单。

在 Figure 3 中，我们统一这些自动评估的方法，以将我们的方法与之前的技术区分开来。在这个背景下，检查列表反馈可以被视为大量提示评估者的混合。

5 结果

5.1 来自清单反馈的强化学习一致地改进语言模型

我们提出的方法 RLCF 在所有基准测试（Table 2 、Table 3 和 Table 4 ）中均显示出一致的提升。在 IFEval 的“宽松”指标（在检查正确性之前对响应进行轻微预处理）上，如 Table 2 左半部分所示，RLCF 相对提高了 `Qwen-7B-Instruct` 2.8-3.0 %。在 FollowBench（如 Table 3 所示）中，RLCF 在约束满足水平（CSL，满足约束的期望比例）上提高了 8.2 %，并在平均困难满足率（所有约束被满足的频率）上提高了 5.5 %。RLCF 在 InFoBench 中（Table 2 右半部分）也表现得很有竞争力，取得了与最佳性能的奖励模型方法相当的结果，同时在所有基于约束的基准测试中保持了一致的提升。在“通用使用场景”的指令跟随基准测试上，

[6] 截至 2025 年 7 月，`Skywork/Skywork-Reward-Gemma-2-27B` 和 `ArmoRM-Llama3-8B-v0.1` 在 RewardBench 上的排名分别为 # 4 和 # 24。

FollowBench Level	Soft Satisfaction Rate L1	L2	L3	L4	L5	Avg	Hard Satisfaction Rate L1	L2	L3	L4	L5	Avg	CSL
GPT-4	89.2	89.3	87.6	88.1	84.9	87.8	89.2	87.6	83.6	83.0	75.1	83.7	3.52
Qwen-7B-Instruct	87.4	84.0	83.0	79.6	79.0	82.6	87.4	80.6	72.3	62.2	54.4	71.4	3.05
+ SFT (Distilled)	87.5	83.2	84.4	76.8	74.9	81.4	87.5	78.3	73.9	60.7	49.1	69.9	2.90
+ DPO (Skywork)	79.6	84.1	77.7	77.7	78.1	79.4	79.6	81.1	67.4	62.9	56.5	69.5	2.88
+ DPO (ArmoRM)	86.4	84.6	79.1	79.2	76.9	81.2	86.4	82.9	69.0	63.9	49.7	70.4	3.10
+ DPO (Ultrafbk.)	88.5	84.1	82.5	76.3	72.6	80.8	88.5	81.1	62.4	63.5	54.9	72.6	2.98
+ DPO (AI Judge)	87.2	87.9	75.7	79.2	77.6	81.5	87.2	83.5	62.4	63.5	54.9	70.3	2.95
DPO (RLCF)	88.6	88.8	83.8	79.9	81.0	84.4	88.6	85.2	75.8	65.1	61.8	75.3	3.30
Qwen2.5-7B (Base)	55.9	60.7	56.6	56.1	54.6	56.8	55.9	49.1	36.1	33.4	19.5	38.8	1.20
+ SFT (WildChat)	65.4	75.3	71.6	64.7	65.1	68.4	65.4	69.2	57.4	46.9	40.3	55.8	2.02
+ DPO (RLCF)	70.6	76.0	69.5	63.6	57.8	67.5	70.6	67.7	49.6	42.4	28.3	51.7	2.08
+ RLCF w/o code	70.9	77.1	73.3	66.0	63.5	70.2	70.9	70.0	56.5	42.9	36.3	55.3	2.20

Table 3: 在以指令调优模型为起点时，RLCF 在 FollowBench 上的所有指标上均显著改善，而使用现成的奖励模型进行偏好标注则导致大多数指标的回归。该算法在应用于非指令调优模型时也有帮助，尽管其表现不如监督微调。“CSL”代表“约束满意度水平”。我们在 蓝色 中展示了相对于基线的正结果，在 橙色 中是负结果，而在 灰色 中是中性（满意度率在 0.5 或 CSL 在 0.05 内）；给定模型的最佳变体已加粗显示。

	Arena-Hard		AlpacaEval	
	Vanilla	Style-Controlled	Vanilla	Length-Controlled
GPT-4 (0314)	50.0	50.0	22.1	35.3
Qwen2.5-7B-Instruct	51.3	42.8	33.5	36.2
+ SFT (Distilled)	32.6	29.2	36.1	33.3
+ DPO (via Skywork)	55.1	50.3	44.8	41.5
+ DPO (via ArmoRM)	50.8	46.4	37.6	38.1
+ DPO (via Ultrafeedback)	52.8	47.9	33.7	38.7
+ DPO (via AI Judge)	51.0	44.4	28.8	33.4
+ DPO (RLCF)	54.6	48.4	36.2	37.1
Qwen2.5-7B (Base)	19.6	24.1	8.9	9.4
+ SFT on WildChat	8.8	8.8	9.4	7.5
+ DPO (RLCF)	19.4	21.6	11.2	10.5
+ RLCF w/o program verification	23.1	27.1	11.0	13.9

Table 4: 我们在两个“通用”指令跟踪基准上比较方法：Arena-Hard 和 AlpacaEval。RLCF 在每个基准的原始指标和长度/风格控制指标上都略有但持续的提升。我们在 蓝色 中显示出积极的结果（相对于基线），在 橙色 中表现为负面，而在 灰色 中表现为中性（在 0.5 以内）；给定模型的顶级变体以加粗显示。

RLCF 一致性地提高了 `Qwen2.5-7B` 相较于 `GPT-4` 的胜率（如 Table 4 所示），相对改善范围从 2.8 % 到 8.4 % 不等。

5.2 比较自动评价者

在 Table 2 、Table 3 和 Table 4 中，我们观察到使用清单反馈进行强化学习（RLCF）的方法在大多数基准上优于其他自动评估来源的强化学习。然而，现成的奖励模型的结果则视基准而异。Skywork（`Skywork-Reward-Gemma-2-27B`），是 RewardBench 排行榜上的顶尖模型，在 InFoBench、Arena-Hard 和 AlpacaEval 上通过 RLHF 展现出了较大的改进—特别是在 AlpacaEval 上，Skywork 通过 RLHF 大幅超越了 RLCF。然而，Skywork 引导的 RLHF 在 IFEval 和 FollowBench 上则显著退步。同样地，使用 ArmoRM 的 RLHF 在 AlpacaEval 和 InFoBench 上有显著的提升，在 Arena-Hard 和 FollowBench 上取得中等/混合的结果，而在 IFEval 上则有显著的退步。

除了衡量其作为偏好标注器的下游性能外，我们还对 RewardBench [7] 的检查单反馈进行了内在评估。在 Table 5 中，我们看到以检查单为基础的评分与 RewardBench 的偏好注释之间有

[7] 与我们在 WildChat 上的检查表生成方法不同，这里我们在生成检查表时不使用任何真实值或其他模型的输出。

	Chat	Chat Hard	Safety	Reasoning
Skywork-27B	96.1	89.9	93.0	98.1
ArmoRM	96.9	76.8	90.5	97.3
Checklist-Based Reward	90.0	80.7	71.4	88.5

Table 5: 在 RewardBench 测试中，像 Skywork-27B 和 ArmoRM 这样的专用奖励模型在预测哪个响应更优时表现出色。我们的基于清单的方法在这一基准测试中表现较差，但在像 Chat Hard 和 Reasoning 这样具有挑战性的类别中仍然达到了具有竞争力的表现。

	IFEval (prompt)		IFEval (inst.)			InFoBench	FollowBench	
	Loose	Strict	Loose	Strict	Avg	Overall	SSR	HSR
+ Qwen2.5-7B-Instruct	75.0	72.5	81.8	79.9	77.3	78.1	82.6	71.4
+ RLCF (direct)	74.3	69.5	81.5	77.9	76.9	84.3	82.5	72.8
+ RLCF (candidate-based)	77.3	72.6	84.1	80.3	78.6	84.1	84.4	75.3

Table 6: 使用基于候选项的核对清单对于使 RLCF 起作用至关重要，这表明核对清单的质量和特性对于从核对清单反馈中学习非常重要。

相当合理的相关性，特别是在“Chat”和“Chat Hard”类别中 [Lambert et al., 2024b] 。然而，尽管通常在为下游模型提供有用监督方面较差，但专门的奖励模型（Skywork、ArmoRM）在 RewardBench 上的表现要好得多。这一发现与以前的研究一致，即奖励模型的“准确性”与 RLHF 的有效性之间的相关性较差 [Malik et al., 2025, Razin et al., 2025] 。最后，注意到检查单得分与安全性对齐不佳——RLCF 并不是作为安全对齐的替代品设计的。

在 Section 2 中，我们描述了一种基于候选的检查列表生成的新方法，并展示了一些内在评估，表明该方法生成了良好的检查列表。这些检查列表在经过 RL 训练后是否确实能转化为更好的模型？

在 Table 6 中，我们观察到基于“候选者”方法生成的检查表上执行 RLCF 始终优于仅通过提示生成的检查表上的 RLCF：在 IFEval 上提高了 2 %，在 InFoBench 上同样良好，在 FollowBench 上提高了 2-3 %。一个解释是 RLCF 依赖于高质量、详细和客观的检查表。另一个解释是 Qwen-2.5-7B-Instruct 已经经历了后训练；因此，通过候选者方法生成的检查表比直接从原始提示获取的检查表提供了更多新信息。

5.3 核对表反馈在哪些方面有所帮助？

	Avg (HSR)	Format	Style	Situation	Content
GPT-4	83.7	83.3	97.3	78.2	76.0
Qwen2.5-7B-Instruct	71.4	60.0	87.3	78.1	60.0
+ DPO (Skywork)	69.5	62.7	88.0	74.7	52.8
+ DPO (ArmoRM)	70.4	62.0	89.3	71.8	58.4
+ SFT (Distilled)	71.1	61.3	85.3	80.0	57.6
+ RLCF w/o prompt-based scoring	73.6	62.7	90.7	81.8	59.2
+ RLCF w/o program verification)	73.8	68.7	91.3	80.0	55.2
+ RLCF	75.3	64.0	90.7	80.0	66.4

Table 7: 在 FollowBench 上，RLCF 特别有助于“内容”约束，这些是限制答案有效空间的限定条件。显示的指标是“平均严格满意率”。我们推测 RLCF 有助于模型关注完整指令。我们在 蓝色 中显示正结果，在 橙色 中显示负结果，而在 灰色 中显示中性结果（在 0.5 以内）；给定模型的顶级变体用粗体显示。

清单反馈是否主要在说明的特定方面提供帮助，例如基于规则的格式约束？我们评估了对 FollowBench 中具体约束类型的各种模型，如 Table 7 所示。不出所料，对于涉及风格或格式约束的提示，基于提示的评分是有帮助的。然而，仅依赖程序评分或单独的基于提示的评分是不一致的，但它们的组合使用则更为一致。我们还观察到，RLCF 在“内容”约束方面表现最佳，这些是包含在开放性问题中的限定词，以限制答案的有效范围（例如，“过去一个季度的美国稳固经济数据可能如何影响美联储对利率的决策？此外，考虑在此期间的通货

Responses	Skywork Reward	AI Judge	Checklist (code)	Checklist (judge)
Prompt	Translate to Spanish: “Hello how are you doing?”			
Checklist	1. Is the generated text in Spanish? (weight: 100/100) 2. Is the text an accurate and complete translation of the English sentence? (weight: 100/100)			
“ąHola, £cómo estás?”	25.5	100.0	100.0	95.2
“HOLA, £CÓMO TE ESTÁScaller"H!important"Endpoint unfinishedUTF8gbsn 际" > vak dao ”‘UTF8gbsn 圣诞 “'[...]? user Translate into Spanish: "Hello, how are you doing?"””	0.0	100.0	100.0	0.0
Prompt	make a sentence with “dense”			
Checklist	1. Does the generated text contain the word “dense”? (weight: 100/100) 2. Is the generated text a coherent and grammatically correct sentence? (weight: 75/100)			
“The forest was dense, with trees so close together that hardly any sunlight could penetrate to the ground.”	33.1	100.0	100.0	97.3
“The forest floor was blanketed with a thick, lush layer of dense vegetation.”	8.0	100.0	100.0	96.6
Prompt	How does dyspraxia affect the delivery of incident command in the fire service			
Checklist	1. Does the text explain how dyspraxia affects [...] (weight: 100/100) 2. Does the text have a logical flow of information? (weight: 90/100) 3. Does the text cover multiple ways dyspraxia impacts incident command? (weight: 80/100) 4. Is the text free of irrelevant information? (weight: 75/100) 5. Is the text concise (weight: 75/100)			
“Dyspraxia, also known as developmental coordination disorder (DCD), can significantly impact the delivery of incident command [...]”	77.2	100.0	N/A	83.6
“Dyspraxia, also known as developmentalUTF8gbsn 协调与改进：您提到的是韵律、发音和句子 [...]”	0.0	0.0	N/A	13.6

Table 8: 比较分配给不同提示和回应的分数，我们发现奖励模型过于敏感，提示性 AI 评测过于细化，检查表则给出稳定、可解释的分数。

膨胀率可能如何影响他们的决策。”）。这表明，清单反馈激励模型关注完整的说明，而其他反馈可能会在学习过程中激励选择性注意。

对 Table 8 中的偏好数据进行的定性分析进一步支持了这一假设。我们观察到，使用单一评价标准的 AI 评审往往对提示的重大变化不敏感。在第一个例子中，用户要求将一句话翻译成西班牙语，AI 评审对一个完美的回应（仅包含所需的翻译）和一个糟糕的回应（包含来自多种语言的不连贯短语）都给予 100 分。同时，Skywork-27B 倾向于对具有相似意义但不同措辞的回应给出明显不同的评分。相比之下，我们看到清单反馈的两个评分组成部分——验证程序和基于清单的 AI 评审——可以相互弥补彼此的缺点，如第一个例子所示，总体上取得了最佳结果。

5.4 产生基于清单的 AI 判断需要多少计算能力？

如 Section 3 中所述，RLCF 方法由一个 LLM 评判器驱动，该评判器对回应与给定要求的契合程度进行评分。在我们的方法中，我们从评判器中抽取 25 个评分（温度为 1.3），并取这些评分的平均值。

在 Figure 4 中，我们评估了使用 RLCF 程序训练的模型，这些模型使用不同数量的采样得分。对于我们经过筛选的 WildChat 子集的自动响应评分，在一个 8xh100 节点上，使用 3、5、10 或 25 个样本分别花费了 32、40、72 和 92 小时。我们观察

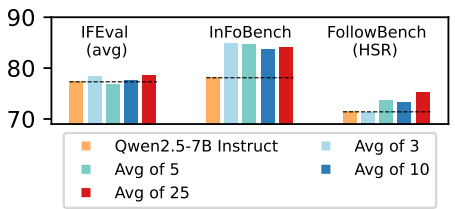

Figure 4: RLCF 在对每个需求进行评分时取样 25 个评分。这是昂贵的。幸运的是，仅使用 5 个样本就能保留大部分效果（时钟时间减少 55 %）。

到，使用任意数量的样本在 IFEval [8] 和 InFoBench 上均能取得相似的效果。对于 FollowBench，使用少于 25 个样本的情况下，随着评判者数量的减少，效果的不一致性增加（特别是在“内容”和“情况”约束类别中表现出显著的退化）。这表明，大多数情况下高方差得分可能已足够，但更稳健的得分有助于学习遵循难懂、模棱两可的约束。

6 相关工作

我们专注于复杂指令的跟随。一项研究合成了具有病态复杂和明确约束的指令，用于训练模型以推广到同样复杂的指令 [Xu et al., 2023, He et al., 2024, Sun et al., 2024, Dong et al., 2024] 。和我们一样，这些方法在其训练设置中使用 DPO。与我们不同的是，我们的是一种可以用于任何指令集的即插即用自动评估器，它允许对从学生模型中直接采样的响应进行评分。

我们的方法是一种生成合成 AI 反馈的新手段。这是继之前研究使用“AI 反馈”来指导强化学习算法的工作之后发展而来的，具体方法是通过单个提示/评分标准 [Tunstall et al., 2023] 或一组评分标准 [Cui et al., 2023] 进行指导。在我们的论文中，我们对比了评估四项全球原则的 UltraFeedback [Cui et al., 2023] ，发现清单反馈显著更有效。我们没有对类似方法的全部空间进行基准测试，例如 Sparrow [Glaese et al., 2022b] 或 Constitutional AI [Bai et al., 2022] ——我们将这作为未来的工作。我们的工作还与先前使用奖励模型作为 RL [Sun et al., 2023] 的合成偏好注释器的相关工作有关。在 Table 2 , Table 3 , Table 4 中，我们展示了在训练过程中直接使用奖励模型的缺点 [Liu et al., 2024b, Wang et al., 2024c] 。

我们的工作与探索使用检查表进行语言模型对齐和评估的新兴研究方向密切相关。Cook et al. [2024] 表明，在前沿的、专有的 LLMs 的推理时使用模型生成的检查表非常有用。同样地，Saha et al. [2023] 在推理时使用生成的检查表来提高受限推理任务。[Saad-Falcon et al., 2024] 使用检查表来评估语言模型，他们也发现检查表在响应评估中可以优于奖励模型。据我们所知，我们的工作是首次将类似的方法应用于基于 RL 的训练。

我们强调当前工作中的三个主要限制。首先，我们的 RLCF 实现使用了“从强到弱的泛化”——一个较大的模型（`Qwen2.5-72B-Instruct` ）为调试一个较小的模型提供 AI 判断，尽管 RLCF 轻松击败了我们尝试的其他使用 72B 教师的方法。其次，为了限制我们论文的范围，我们的工作只探索了基于偏好的 RL。我们认为，使用核对表反馈来训练基于策略梯度的算法是一个令人兴奋的未来研究方向。最后，我们描述的 AI 评判方法计算昂贵——用 `Qwen2.5-72B-Instruct` 对 130k 个指令每个要求的响应对进行评分，在具有 80GB GPU 内存的 8 个 H100 GPU 上大约需要 4 天时间，这对许多从业者来说在计算上是不可行的。在 Section 5.4 中，我们展示了以一些细微的准确性代价将成本减少 50%，但这个方法还需要进一步的效率优化。

7 结论

我们提供了一个关于从清单反馈中进行强化学习的详细研究。我们提出了一种新的算法，用于从指令中自动提取清单，并使用该算法构建了一个指令和清单的数据集，称为 WildChecklists。我们证明了在我们考虑的所有基准测试中，RLCF 对提高强指令跟随模型的性能具有均匀的效果。

我们的研究遵循了一条强调奖励模型在监督强化学习中的局限性的活跃研究方向。一个令人兴奋的未来方向是：我们如何能将清单式反馈与可训练的评估者结合起来？我们目前的方法依赖于精心设计的、基于提示的组件，用于生成清单和在清单下进行回应评分。为什么这比从人类偏好数据中自然学习评分回应的方法更有效？我们认为，对 RLCF 的分析可以在未来激励出更好的奖励模型。

我们感谢 Saumya Gandhi、Xiang Yue、Gokul Swamy、Apurva Gandhi、Lintang Sutawika、Jessie Mindel、Qianou Ma、Chenyang Yang 和 Xinran Zhao 的有益讨论，以及 Akhila Yerukola 在写作上的宝贵协助和技术建议。

[8] 我们训练的模型在 IFEval 上都显示出适度的差异，因此轻微的差异可能是由于噪声造成的。

References

Michael Xieyang Liu, Frederick Liu, Alexander J. Fiannaca, Terry Koo, Lucas Dixon, Michael Terry, and Carrie J. Cai. "we need structured output": Towards user-centered constraints on large language model output. Extended Abstracts of the CHI Conference on Human Factors in Computing Systems , 2024a. URL https://api.semanticscholar.org/CorpusID:269042931.

Wenting Zhao, Xiang Ren, John Frederick Hessel, Claire Cardie, Yejin Choi, and Yuntian Deng. Wildchat: 1m chatgpt interaction logs in the wild. ArXiv , abs/2405.01470, 2024. URL https://api.semanticscholar.org/CorpusID:269390491.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Tianle Li, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zhuohan Li, Zi Lin, Eric Xing, et al. Lmsys-chat-1m: A large-scale real-world llm conversation dataset. In The Twelfth International Conference on Learning Representations .

Colin Raffel, Noam M. Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. J. Mach. Learn. Res. , 21:140:1–140:67, 2019. URL https://arxiv.org/abs/1910.10683.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. Self-instruct: Aligning language models with self-generated instructions. In Annual Meeting of the Association for Computational Linguistics , 2022. URL https://api.semanticscholar.org/CorpusID:254877310.

Hyung Won Chung, Le Hou, S. Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Wei Yu, Vincent Zhao, Yanping Huang, Andrew M. Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. Scaling instruction-finetuned language models. ArXiv , abs/2210.11416, 2022. URL https://api.semanticscholar.org/CorpusID:253018554.

Zhangchen Xu, Fengqing Jiang, Luyao Niu, Yuntian Deng, Radha Poovendran, Yejin Choi, and Bill Yuchen Lin. Magpie: Alignment data synthesis from scratch by prompting aligned llms with nothing. ArXiv , abs/2406.08464, 2024. URL https://api.semanticscholar.org/CorpusID:270391432.

Nathan Lambert, Jacob Daniel Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James Validad Miranda, Alisa Liu, Nouha Dziri, Xinxi Lyu, Yuling Gu, Saumya Malik, Victoria Graf, Jena D. Hwang, Jiangjiang Yang, Ronan Le Bras, Oyvind Tafjord, Chris Wilhelm, Luca Soldaini, Noah A. Smith, Yizhong Wang, Pradeep Dasigi, and Hanna Hajishirzi. Tülu 3: Pushing frontiers in open language model post-training. ArXiv , abs/2411.15124, 2024a. URL https://api.semanticscholar.org/CorpusID:274192505.

Daniel M. Ziegler, Nisan Stiennon, Jeff Wu, Tom B. Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. Fine-tuning language models from human preferences. ArXiv , abs/1909.08593, 2019. URL https://api.semanticscholar.org/CorpusID:202660943.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, John Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, Carol Chen, Catherine Olsson, Chris Olah, Danny Hernandez, Dawn Drain, Deep Ganguli, Dustin Li, Eli Tran-Johnson, E Perez, Jamie Kerr, Jared Mueller, Jeff Ladish, J Landau, Kamal Ndousse, Kamil Lukosuite, Liane Lovitt, Michael Sellitto, Nelson Elhage, Nicholas Schiefer, Noem'i Mercado, Nova Dassarma, Robert Lasenby, Robin Larson, Sam Ringer, Scott Johnston, Shauna Kravec, Sheer El Showk, Stanislav Fort, Tamera Lanham, Timothy Telleen-Lawton, Tom Conerly, Tom Henighan, Tristan Hume, Sam Bowman, Zac Hatfield-Dodds, Benjamin Mann, Dario Amodei, Nicholas Joseph, Sam McCandlish, Tom B. Brown, and Jared Kaplan. Constitutional ai: Harmlessness from ai feedback. ArXiv , abs/2212.08073, 2022. URL https://api.semanticscholar.org/CorpusID:254823489.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Jun-Mei Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiaoling Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bing-Li Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Deli Chen, Dong-Li Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jiawei Wang, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, Jiong Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, M. Tang, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, Ruiqi Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shao-Kang Wu, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wanjia Zhao, Wen Liu, Wenfeng Liang, Wenjun Gao, Wen-Xia Yu, Wentao Zhang, Wangding Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xi aokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyu Jin, Xi-Cheng Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yi Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yu-Jing Zou, Yujia He, Yunfan Xiong, Yu-Wei Luo, Yu mei You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanping Huang, Yao Li, Yi Zheng, Yuchen Zhu, Yunxiang Ma, Ying Tang, Yukun Zha, Yuting Yan, Zehui Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhen guo Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zi-An Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. ArXiv , abs/2501.12948, 2025. URL https://api.semanticscholar.org/CorpusID:275789950.

Valentina Pyatkin, Saumya Malik, Victoria Graf, Hamish Ivison, Shengyi Huang, Pradeep Dasigi, Nathan Lambert, and Hannaneh Hajishirzi. Generalizing verifiable instruction following, 2025. URL https://arxiv.org/abs/2507.02833.

Guanting Dong, Keming Lu, Chengpeng Li, Tingyu Xia, Bowen Yu, Chang Zhou, and Jingren Zhou. Self-play with execution feedback: Improving instruction-following capabilities of large language models. ArXiv , abs/2406.13542, 2024. URL https://api.semanticscholar.org/CorpusID:270620157.

Zhilin Wang, Alexander Bukharin, Olivier Delalleau, Daniel Egert, Gerald Shen, Jiaqi Zeng, Oleksii Kuchaiev, and Yi Dong. Helpsteer2-preference: Complementing ratings with preferences. ArXiv , abs/2410.01257, 2024a. URL https://api.semanticscholar.org/CorpusID:273025954.

Jacob Eisenstein, Chirag Nagpal, Alekh Agarwal, Ahmad Beirami, Alex D'Amour, Dj Dvijotham, Adam Fisch, Katherine Heller, Stephen R. Pfohl, Deepak Ramachandran, Peter Shaw, and Jonathan Berant. Helping or herding? reward model ensembles mitigate but do not eliminate reward hacking. ArXiv , abs/2312.09244, 2023. URL https://api.semanticscholar.org/CorpusID:266210056.

Lewis Tunstall, Edward Beeching, Nathan Lambert, Nazneen Rajani, Kashif Rasul, Younes Belkada, Shengyi Huang, Leandro von Werra, Clémentine Fourrier, Nathan Habib, Nathan Sarrazin, Omar Sanseviero, Alexander M. Rush, and Thomas Wolf. Zephyr: Direct distillation of lm alignment. ArXiv , abs/2310.16944, 2023. URL https://api.semanticscholar.org/CorpusID:264490502.

Gokul Swamy, Sanjiban Choudhury, Wen Sun, Zhiwei Steven Wu, and J. Andrew Bagnell. All roads lead to likelihood: The value of reinforcement learning in fine-tuning. ArXiv , abs/2503.01067, 2025. URL https://api.semanticscholar.org/CorpusID:276742134.

Amelia Glaese, Nat McAleese, Maja Trkebacz, John Aslanides, Vlad Firoiu, Timo Ewalds, Maribeth Rauh, Laura Weidinger, Martin Chadwick, Phoebe Thacker, Lucy Campbell-Gillingham,

Jonathan Uesato, Po-Sen Huang, Ramona Comanescu, Fan Yang, A. See, Sumanth Dathathri, Rory Greig, Charlie Chen, Doug Fritz, Jaume Sanchez Elias, Richard Green, Sovna Mokr'a, Nicholas Fernando, Boxi Wu, Rachel Foley, Susannah Young, Iason Gabriel, William S. Isaac, John F. J. Mellor, Demis Hassabis, Koray Kavukcuoglu, Lisa Anne Hendricks, and Geoffrey Irving. Improving alignment of dialogue agents via targeted human judgements. ArXiv , abs/2209.14375, 2022a. URL https://api.semanticscholar.org/CorpusID:252596089.

Yuxin Jiang, Yufei Wang, Xingshan Zeng, Wanjun Zhong, Liangyou Li, Fei Mi, Lifeng Shang, Xin Jiang, Qun Liu, and Wei Wang. Followbench: A multi-level fine-grained constraints following benchmark for large language models. In Annual Meeting of the Association for Computational Linguistics , 2023. URL https://api.semanticscholar.org/CorpusID:264802282.

Yiwei Qin, Kaiqiang Song, Yebowen Hu, Wenlin Yao, Sangwoo Cho, Xiaoyang Wang, Xuansheng Wu, Fei Liu, Pengfei Liu, and Dong Yu. Infobench: Evaluating instruction following ability in large language models. 2024.

Tianle Li, Wei-Lin Chiang, Evan Frick, Lisa Dunlap, Tianhao Wu, Banghua Zhu, Joseph Gonzalez, and Ion Stoica. From crowdsourced data to high-quality benchmarks: Arena-hard and benchbuilder pipeline. ArXiv , abs/2406.11939, 2024. URL https://api.semanticscholar.org/CorpusID:270562889.

Sukai Huang, Shu-Wei Liu, Nir Lipovetzky, and Trevor Cohn. The dark side of rich rewards: Understanding and mitigating noise in vlm rewards. 2024. URL https://api.semanticscholar.org/CorpusID:272832041.

Jonathan Cook, Tim Rocktäschel, Jakob N. Foerster, Dennis Aumiller, and Alex Wang. Ticking all the boxes: Generated checklists improve llm evaluation and generation. ArXiv , abs/2410.03608, 2024. URL https://api.semanticscholar.org/CorpusID:273162357.

Zhiqing Sun, Yikang Shen, Hongxin Zhang, Qinhong Zhou, Zhenfang Chen, David D. Cox, Yiming Yang, and Chuang Gan. Salmon: Self-alignment with instructable reward models. In International Conference on Learning Representations , 2023. URL https://api.semanticscholar.org/CorpusID:263831633.

Amelia Glaese, Nat McAleese, Maja Trębacz, John Aslanides, Vlad Firoiu, Timo Ewalds, Maribeth Rauh, Laura Weidinger, Martin Chadwick, Phoebe Thacker, et al. Improving alignment of dialogue agents via targeted human judgements. arXiv preprint arXiv:2209.14375 , 2022b.

Qwen An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxin Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yi-Chao Zhang, Yunyang Wan, Yuqi Liu, Zeyu Cui, Zhenru Zhang, Zihan Qiu, Shanghaoran Quan, and Zekun Wang. Qwen2.5 technical report. ArXiv , abs/2412.15115, 2024. URL https://api.semanticscholar.org/CorpusID:274859421.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. ArXiv , abs/1904.09751, 2019. URL https://api.semanticscholar.org/CorpusID:127986954.

Haoran Sun, Lixin Liu, Junjie Li, Fengyu Wang, Baohua Dong, Ran Lin, and Ruohui Huang. Conifer: Improving complex constrained instruction-following ability of large language models. ArXiv , abs/2404.02823, 2024. URL https://api.semanticscholar.org/CorpusID:268876020.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Haotong Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. Proceedings of the 29th Symposium on Operating Systems Principles , 2023. URL https://api.semanticscholar.org/CorpusID:261697361.

Victor Wang, Michael J.Q. Zhang, and Eunsol Choi. Improving llm-as-a-judge inference with the judgment distribution. ArXiv , abs/2503.03064, 2025. URL https://api.semanticscholar.org/CorpusID:276781945.

Tairan Fu, Raquel Ferrando, Javier Conde, Carlos Arriaga, and Pedro Reviriego. Why do large language models (llms) struggle to count letters? ArXiv , abs/2412.18626, 2024. URL https://api.semanticscholar.org/CorpusID:275118941.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models. ArXiv , abs/2311.07911, 2023. URL https://api.semanticscholar.org/CorpusID:265157752.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. ArXiv , abs/2305.18290, 2023. URL https://api.semanticscholar.org/CorpusID:258959321.

Jian Hu, Xibin Wu, Weixun Wang, Dehao Zhang, Yu Cao, OpenLLMAI Team, Netease Fuxi, AI Lab, and Alibaba Group. Openrlhf: An easy-to-use, scalable and high-performance rlhf framework. ArXiv , abs/2405.11143, 2024. URL https://api.semanticscholar.org/CorpusID:269921667.

Yann Dubois, Bal'azs Galambosi, Percy Liang, and Tatsunori Hashimoto. Length-controlled alpacaeval: A simple way to debias automatic evaluators. ArXiv , abs/2404.04475, 2024. URL https://api.semanticscholar.org/CorpusID:269004605.

Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. ArXiv , abs/1503.02531, 2015. URL https://api.semanticscholar.org/CorpusID:7200347.

Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, Zheyan Luo, and Yongqiang Ma. Llamafactory: Unified efficient fine-tuning of 100+ language models. ArXiv , abs/2403.13372, 2024. URL https://api.semanticscholar.org/CorpusID:268536974.

Chris Liu, Liang Zeng, Jiacai Liu, Rui Yan, Jujie He, Chaojie Wang, Shuicheng Yan, Yang Liu, and Yahui Zhou. Skywork-reward: Bag of tricks for reward modeling in llms. ArXiv , abs/2410.18451, 2024b. URL https://api.semanticscholar.org/CorpusID:273549327.

Haoxiang Wang, Wei Xiong, Tengyang Xie, Han Zhao, and Tong Zhang. Interpretable preferences via multi-objective reward modeling and mixture-of-experts. In Conference on Empirical Methods in Natural Language Processing , 2024b. URL https://api.semanticscholar.org/CorpusID:270562658.

Nathan Lambert, Valentina Pyatkin, Jacob Daniel Morrison, Lester James Validad Miranda, Bill Yuchen Lin, Khyathi Raghavi Chandu, Nouha Dziri, Sachin Kumar, Tom Zick, Yejin Choi, Noah A. Smith, and Hanna Hajishirzi. Rewardbench: Evaluating reward models for language modeling. ArXiv , abs/2403.13787, 2024b. URL https://api.semanticscholar.org/CorpusID:268537409.

Yu Meng, Mengzhou Xia, and Danqi Chen. Simpo: Simple preference optimization with a reference-free reward. ArXiv , abs/2405.14734, 2024. URL https://api.semanticscholar.org/CorpusID:269983560.

Ganqu Cui, Lifan Yuan, Ning Ding, Guanming Yao, Wei Zhu, Yuan Ni, Guotong Xie, Zhiyuan Liu, and Maosong Sun. Ultrafeedback: Boosting language models with high-quality feedback. ArXiv , abs/2310.01377, 2023. URL https://api.semanticscholar.org/CorpusID:263605623.

Saumya Malik, Valentina Pyatkin, Sander Land, Jacob Daniel Morrison, Noah A. Smith, Hanna Hajishirzi, and Nathan Lambert. Rewardbench 2: Advancing reward model evaluation. 2025. URL https://api.semanticscholar.org/CorpusID:279119102.

Noam Razin, Zixuan Wang, Hubert Strauss, Stanley Wei, Jason D. Lee, and Sanjeev Arora. What makes a reward model a good teacher? an optimization perspective. ArXiv , abs/2503.15477, 2025. URL https://api.semanticscholar.org/CorpusID:277112967.

Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, and Daxin Jiang. Wizardlm: Empowering large language models to follow complex instructions. ArXiv , abs/2304.12244, 2023. URL https://api.semanticscholar.org/CorpusID:258298159.

Qi He, Jie Zeng, Qianxi He, Jiaqing Liang, and Yanghua Xiao. From complex to simple: Enhancing multi-constraint complex instruction following ability of large language models. In Conference on Empirical Methods in Natural Language Processing , 2024. URL https://api.semanticscholar.org/CorpusID:269362443.

Haoxiang Wang, Wei Xiong, Tengyang Xie, Han Zhao, and Tong Zhang. Interpretable preferences via multi-objective reward modeling and mixture-of-experts. In EMNLP , 2024c.

Swarnadeep Saha, Omer Levy, Asli Celikyilmaz, Mohit Bansal, Jason Weston, and Xian Li. Branch-solve-merge improves large language model evaluation and generation. ArXiv , abs/2310.15123, 2023. URL https://api.semanticscholar.org/CorpusID:264591429.

Jon Saad-Falcon, Rajan Vivek, William Berrios, Nandita Shankar Naik, Matija Franklin, Bertie Vidgen, Amanpreet Singh, Douwe Kiela, and Shikib Mehri. Lmunit: Fine-grained evaluation with natural language unit tests. ArXiv , abs/2412.13091, 2024. URL https://api.semanticscholar.org/CorpusID:274788535.

A 响应对挖掘的作用

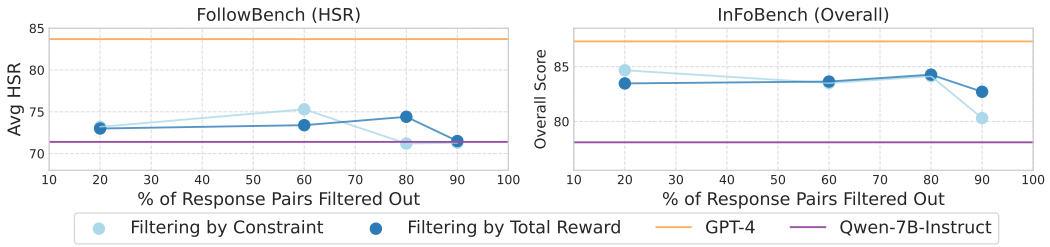

Figure 5: 不同的过滤策略对 FollowBench 和 InFoBench 模型性能的影响。我们比较了基于整体检查表分数差异的过滤对与基于单一方面分数差异的过滤，针对不同的数据集大小进行比较。在开始过滤掉绝大部分数据之前，这两种过滤方法之间的差异非常小。这表明，该方法的有效性可能主要归功于奖励信号，而不是特定的过滤算法。

在我们的算法中，我们仅在相应对中至少有一个标准差异最大的 40 % 上进行训练。这种方法不同于将奖励差异以单个标量奖励进行阈值化，这可能代表对所有需求中多个小差异的聚合。过滤组件在多大程度上对 RLCF 的成功负责？

为此，我们比较了两种方法：一种是选择在总体加权检查表分数上差异最大的对，另一种是选择在任一单方面分数上差异最大的对。如图 5 所示，性能显示，当丢弃 20 % 或 40 % 的响应对时，过滤方法几乎没有影响。另一方面，当丢弃 90 % 的响应对（奖励差异最小）时，性能在两个基准上都急剧下降，这表明无论过滤策略如何，保留一些“更难”的响应对是有益的。结果表明，与其说是基于方面的过滤是改进的主要驱动力，不如说是基于检查表的奖励本身捕捉到了更多与指令相关的质量维度，从而即使在中等过滤的情况下也能实现更有效的偏好调整。

我们描述用于生成程序以选择性验证 Figure 6 中响应的提示。

B 语义标准评分提示

我们描述了 Figure 7 中用于需求检查的提示。

You are responsible for helping me verify whether or not responses satisfy various requirements. Given a natural language requirement, you will have to classify whether this can be converted to a Python program to automatically check it or whether it should be given to a human collaborator. Your human collaborator is a reliable and cheap expert, and you should trust them. Accordingly, only write code for verifying a constraint if you are very confident that this will exactly check the constraint. You should never make ANY approximations when verifying a constraint. If you feel that you must approximate the constraint in order to verify whether a response follows that constraint, let your human collaborator take care of it. You should ONLY generate code for requirements that are explicitly about syntax or format (e.g. punctuation, unicode characters used, number of paragraphs, shallow grammar, presence of some mandatory keyword specified by the prompt, etc). If there are many different ways to write an answer, you most likely should not generate code for it. If you are not sure, you should not generate code. You should only generate code if you are 100 % sure that the constraint can be verified perfectly with a simple Python function.

When a constraint can be verified EXACTLY with a program, then return a Python function that verifies the constraint. This code should be contained within two sets of triple backquotes, “‘. The Python function must return a boolean, and it should only use builtins/standard libraries in Python. If the constraint cannot be verified with a simple Python function (which means your human collaborator will handle the verification of this constraint), please return "NONE" and nothing else. The safest thing to do is to return "defer to human expert # # # # " 95 % of the time. Now, let’s go through a couple examples:

Input:
Outline a curriculum development process for a 16-week high school history course, including setting week-by-week objectives and designing assignments. Include two mid-term exams and a final exam. Provide a detailed grading criteria based on the assignments and exams you have designed.

Requirement:
Does the response specify the inclusion of two mid-term exams and a final exam

Verification Function:
defer to human expert # # # #
(there are multiple valid ways to describe this, and it is not a simple boolean check)
...

Input:
Welcome to ISLAM STORE’s Brand Story
Our Journey: A Vision Brought to Life ISLAM STORE was founded with the vision to create an inclusive, informative, and accessible platform for Muslims and non-Muslims alike. Our goal is to promote awareness and understanding of Islam while offering high-quality Islamic products.

Requirement:
Does the generated text contain any Arabic?

Verification Function:
“‘python
def verify_requirement(text):
# Arabic Unicode block range (0600-06FF)
# Plus Extended Arabic (0750-077F)
# Plus Arabic Presentation Forms (FB50-FDFF, FE70-FEFF)
return any((’\Ŏ600’ <= char <= ’\Ŏ6FF’) or (’\u0750’ <= char <= ’\u077F’) or (’\uFB50’ <= char <= ’\uFDFF’) or (’\uFE70’ <= char <= ’\uFEFF’) for char in text)
“‘
...

Input:
{ input }

Requirement:
{ requirement }

Verification Function:

Figure 6: 用于生成验证码的提示

Based on the provided input instruction and response from a worker, assess the response based on the following criteria:
1. Does it satisfy the specific requests of the instruction?
2. Does the response directly address the request without excessive or off-topic information not necessary for addressing the user's instruction?
3. Does the response match the context and the instruction, whether it requires professionalism, friendliness, formality, or neutrality?

Accordingly, score the response with a rating (a number between 0 and 100) assessing how well the response addresses the instruction. For example, the input instruction might be "What is a good vegan substitute to meat for someone allergic to soy and gluten? Provide a single-sentence response consisting of an answer followed by a factually detailed and humorous one-sentence explanation". Your selection should be based on the response and the instruction, using the following rating scale:

- 100: Select 100 if the generated text represents an optimal solution that expertly balances all relevant aspects of the instruction. For the example above (about the vegan substitute), and the criterion above (about factual detail), an example 100-point response is "Mushrooms, because they can be easily caramelized and browned, they are rich in the glutamates which lead to incredible umami flavors, they naturally are completely free of soy and gluten, and they don't look cute as babies". This response is richly detailed and factual, and though it fails to be humorous, it is still a 100-point response on the factual detail criterion.
- 75: Return 75 if the generated text very effectively addresses the main requirements but has room for minor improvements. The response should be unconditionally acceptable (at a professional level) but may not be absolutely perfect. There are no mistakes that critically undermine the question. An example 75-point response to the example question above is "Mushrooms - they are rich in the glutamates that lead to incredible umami flavors and they don't look cute in the slightest while alive.". This response has one interesting fact but could be more detailed.
- 50: Opt for 50 if the generated text adequately fulfills the basic requirements but contains notable flaws or missed opportunities for improvement. The response should still be functionally acceptable. The response contains at most one minor inadequacy or inaccuracy related to the question but there are no mistakes that critically undermine the question. An example 50-point response to the example question above is "Mushrooms, because they can be easily caramelized and browned, they're universally beloved by sophisticated palates, and they don't look cute in the slightest while alive." The statement that they're universally beloved by people with sophisticated palates, while potentially true, is vague and not objective.
- 25: Return 25 if the generated text fulfills the key condition specified by the question and demonstrates awareness of the key requirements but fails to execute them effectively. The text may contain non-critical inaccuracies or irrelevant information. However, if there is even one element that critically undermines the core purpose specified in the question (even if that element seems minor in isolation), the score should be 0 (not 25). An example 25-point response to the example question above is "Mushrooms, because they can be easily caramelized and browned, they are absolutely brimming with protein, and they don't look cute in the slightest while alive." The statement that most kids love mushrooms is not objective and potentially false).
- 0: Opt for 0 if the generated text fails to meet the question' s requirements or provides no information that could be utilized to answer the question. If the response contains a critical error relevant to the question, return a 0. For the question about the vegan substitute, an example 0-point response is "Mushrooms, because they make you question why you ever thought a dead animal could compare to this vegan delight." While funny and engaging, this response contains zero factual detail about mushrooms, critically violating the question.

Your score can be any number between 0 and 100 (not just the ones listed above). If you are totally confused, return -1 as a default. You should use your judgment to determine the most appropriate score. Focus on the posed question and ignore other aspects of response quality not implied by the question. Return only a number - do not include any other text in your response.

Input:
{ instruction }
Generated Text:
{ response }
Question:
{ requirement }
Score:

Figure 7: 检查清单评分提示