学习专注：通过梯度引导的标记修剪进行因果注意力蒸馏

Yiju Guo[♪], **Wenkai Yang**[♪], **Zexu Sun**[♪], **Ning Ding**[♩], **Zhiyuan Liu**[♩], **Yankai Lin**[♪]✉
[♪]Gaoling School of Artificial Intelligence, Renmin University of China
[♩] Department of Computer Science and Technology, Tsinghua University
{ yijuguo, yankailin } @ruc.edu.cn

Abstract

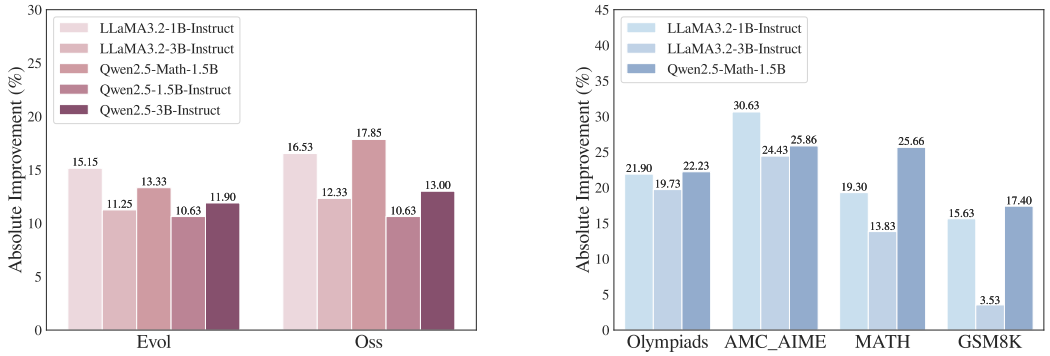
大型语言模型（LLMs）在上下文理解方面展示了显著的改进。然而，它们在长上下文推理和生成过程中关注真正关键信息的能力仍然滞后于发展的步伐。具体而言，我们的初步实验揭示了一些干扰模式会在推理过程中误导模型的注意力，而去除这些模式会大幅提高推理准确性和生成质量。我们将此现象归因于训练数据中的虚假相关性，这妨碍了模型推断真实因果指导——响应关系的能力。这种现象可能导致冗余的推理过程，可能造成显著的推理开销，更为关键的是，生成错误或次优的响应。为此，我们推出了一个名为学习聚焦（LeaF）的两阶段框架，利用基于干预的推理来解开混杂因素。在第一阶段，LeaF 利用与高级教师的梯度比较，基于训练语料的因果关系自动识别混杂的标记。随后，在第二阶段，通过蒸馏过程中剪除这些标记，实现干预，使学生的注意力对齐于教师在真正关键的上下文标记上的聚焦分布。实验结果表明，LeaF 不仅在各种数学推理和代码生成基准测试中取得了绝对的改进，而且还通过抑制推理过程中对混杂标记的注意力，提供了一个更具可解释性和可靠性的推理模型。

1 引言

大型语言模型（LLMs）在自然语言处理任务中取得了显著的成功，展示了在语境理解 [? ? ?] 和语言生成 [? ?] 方面的强大能力。尽管取得了这些进展，LLMs 仍然难以专注于真正关键的信息，特别是在长文本推理 [? ? ?] 和基于复杂指令的任务 [? ?] 中，这对推理准确性和生成质量产生了不利影响。为了系统地研究这一现象，我们首先通过对教师和学生敏感性进行基于梯度的比较来识别干扰模式，然后评估学生模型在 NuminaMath-CoT [?] 和 AceCode-87K [?] 上的表现。令人惊讶的是，正如图 1 所示，简单地修剪干扰模式就带来了显著的提升——在 MATH 训练语料库 [?] 上平均准确率提高了超过 20 %，而在代码训练语料库 [?] 上则提高了超过 10 %。此外，我们观察到，在 AMC_AIME [?] 上的提升比在 GSM8K [?] 上更显著，这表明复杂的推理问题可能包含更多干扰模式，这些模式会干扰模型的推理。这些发现表明，减轻干扰模式的影响对于提高 LLM 推理的鲁棒性和准确性是必不可少的。我们在图 2 中进一步展示了一个代表性案例。从指令中移除干扰模式，无需额外训练，帮助模型专注于关键信息并增强推理能力。这表明改善对相关细节的注意力是推进模型推理的一个有前途的方向。基于我们的分析，我们采用因果视角（图 3 ）来解释和缓解观察到的现象。我们提出学习聚焦（LeaF），一个两阶段框架，将干扰模式视为 LLM 推理中的伪因子。在第一阶段，LeaF 通过高容量教师模型与学生模型之间的梯度比较来识别混淆的 token。然后，通过跨度剪枝生成反事实样本，从每个指令中去除检测到混淆的 token 的连续跨度。在第二阶段，LeaF 引入混合蒸馏损失，最小化两个 KL 散度：一个用于原始样本（标准蒸馏），另一个用于反事实样本（反事实蒸馏）。这种复合目标鼓励学生模型通过比

✉ Corresponding author: Yankai Lin (yankailin@ruc.edu.cn).

Preprint. Under review.



(a) Code Accuracy Improvement (b) MATH Accuracy Improvement

Figure 1: 通过从数学和代码训练语料库的小模型中去除混杂符号实现了准确性的提高。结果表明性能显著提高，在数学语料库上提高了超过 20 %，在代码语料库上提高了超过 10 %。（有关这些类别的更多细节，参见附录 **??** 。）

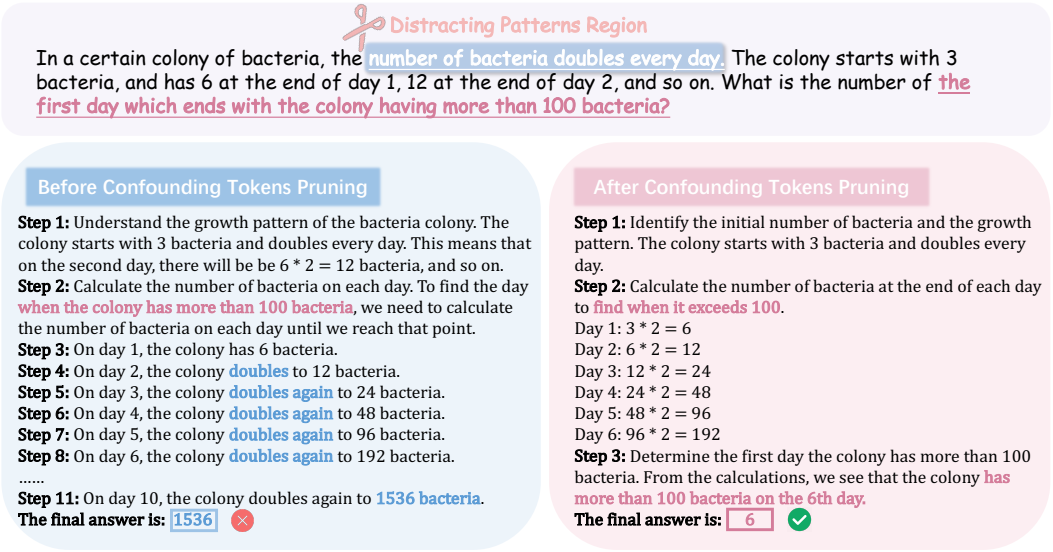


Figure 2: 修剪干扰模式前后的推理比较。蓝色阴影区域 表示已修剪的混淆标记。粉色高亮 标记需要关注的区域，而 蓝色高亮 显示过度关注导致错误的地方。

较教师模型在剪枝混淆 token 前后的输出来捕获真正的因果依赖，提高 LLM 推理的鲁棒性和可解释性。

我们通过全面实验表明，Learning to Focus (LeaF) 在推理期间显著增强了关键标记识别和注意力一致性，从而提高了下游任务的性能，如数学推理和代码生成。与标准的知识蒸馏相比，LeaF 在 GSM8K [**?**] 、MATH [**?**] 和 OlympiadBench [**?**] 上对 LLaMA-1B/3B-Instruct [**?**] 和 Qwen2.5-Math-1.5B [**?**] 实现了平均 2.41 % 的准确率提升。在代码生成方面，LeaF 在 HumanEval+ [**?**] 、LeetCode [**?**] 和 LivecodeBench [**?**] 上平均提高了 2.48 % 。这些结果验证了我们的假设，即增强对关键信息的关注对提高推理性能至关重要。第 **??** 节的注意力可视化进一步展示了 LeaF 的可解释性。

2 方法论

2.1 因果框架

根据 Pearl 的结构因果模型 [**?**] ，我们建立一个 DAG G 来模拟不同组件之间的因果关系（参见图 3 ）。我们假设 (X, A, Y) 的分布忠实于 DAG。在这个框架中，$X = [x_1, x_2, \ldots, x_n]$ 代表输入的标记，而 Y 是模型的输出。我们将混淆标记定义为一个子集 A ，通过引入与输

出 Y 和补充输入 $X \setminus A$ 的虚假相关性来掩盖真实的因果关系。这些误导性的依赖关系扭曲了模型的注意力机制，并偏向其推理过程，最终产生不可靠的预测。

模型的理想行为是消除由混杂的标记 A 引入的虚假相关。当 A 同时影响 X 和 Y 时（参见图 3 中的虚线箭头），观察到的条件分布变为：

$$P(Y \mid X_i = x) = \sum_A P(Y \mid X_i = x, A)\, P(A \mid X_i = x), \quad (1)$$

，这偏离了干预分布 $P(Y \mid \mathrm{do}(X_i))$ ，反映了 A 通过虚假路径对 Y 的间接影响所引入的偏差。

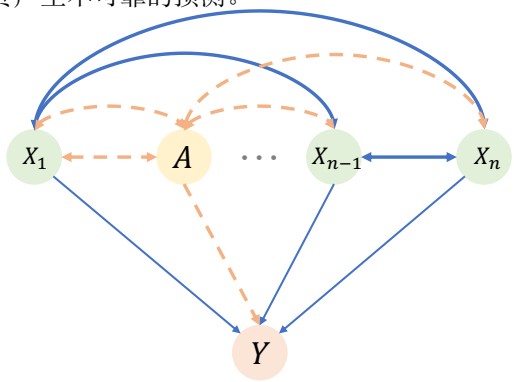


Figure 3: 推理过程的因果图。X 表示输入提示，而 Y 表示模型的输出。在 X 中的一个子集的标记，被识别为混杂标记（A），引入了破坏推理过程的虚假相关性。我们的方法检测和屏蔽 A ，有效去除了从 A 到 Y 的虚假边缘，并恢复了真正的因果依赖。

为了阻断这些非因果影响，我们提出了因果剪枝方法，该方法在蒸馏之前去除了 A 的影响。这促使学生模型学习基于真实因果结构的注意力模式，从而提高鲁棒性和可解释性。

为了消除虚假的依赖关系，我们引入了学习专注（LeaF）框架（图 **??** ），该框架由两个主要阶段组成：（1）混淆标记检测（第 **??** 节），在该阶段，LeaF 通过教师-学生基于梯度的比较识别混淆标记，并通过修剪这些标记来构建反事实样本。（2）因果注意力蒸馏（第 2.1.1 节），在该阶段，LeaF 通过一种混合蒸馏损失捕捉因果依赖，该损失使学生在原始样本和反事实样本上与教师对齐。我们将在下文中详细讨论它们。

为了识别引入虚假相关性的混淆标记 A ，我们采用了一种基于梯度的方法 [**?** **?**] 来定量测量每个标记对模型输出 Y 的影响。具体来说，我们利用了教师模型 $\boldsymbol{\theta}_T$ 和学生模型 $\boldsymbol{\theta}_S$ 的梯度敏感性。我们专注于由学生错误预测但由教师正确处理的数据实例，以隔离混淆标记。对于每个标记 $x_i \in X$ ，我们计算两个模型对该标记的损失梯度：

这些梯度反映了每个模型对 x_i 中扰动的敏感性。为了实现具有不同梯度尺度的模型之间的标记级比较，我们对敏感性值应用最小-最大归一化。

为了识别混淆标记，我们通过计算每个标记的梯度差异来捕捉教师模型和学生模型之间在标记级注意力的差异：

为了确保跨实例的梯度差异一致性，我们对梯度差异进行了归一化，并将一个令牌 x_i 分类为干扰令牌，如果：(i) 在推理过程中，它从学生模型接收到显著的注意力，但从教师模型接收到的注意力可以忽略不计，如方程 2 中所述，并且 (ii) 移除该令牌会导致两个模型都做出正确预测。

$$\frac{\Delta\hat{g}_i - \min_j \Delta\hat{g}_j}{\max_j \Delta\hat{g}_j - \min_j \Delta\hat{g}_j} \leq \tau_{\text{confounder}}, \quad (2)$$

其中 $\tau_{\text{confounder}}$ 是通过在验证集上的统计分析确定的阈值。关于阈值的敏感性分析在第 4.3 节中展示。从直觉上讲，干扰令牌捕获了教师和学生模型之间的敏感性差异，这表明存在虚假的依赖关系。

此外，我们还探索了基于困惑度的检测，但没有教师指导，它往往捕捉的是模型不确定性的标志性符号，而不是真正的混淆因素，尤其是在像奥林匹克竞赛这样具有挑战性的任务中。因此，我们采用基于梯度的检测作为我们的主要策略（详见章节 4.1 ）。

剪枝策略。 我们研究了两种修剪策略来从指令 X 中去除混杂标记：（1）集合修剪，去除所有已识别的混杂因子 A ，得到 $X \setminus A$ 。（2）范围修剪，每次仅去除一个连续的混杂范围 A_i ，得到 $X \setminus A_i$ 。初步实验表明范围修剪优于集合修剪（见附录 **??** ），因为同时修剪所有干扰模式会破坏句子完整性。因此，我们通过范围修剪策略构建对比样本：

$$\mathcal{D}_{\text{pruned}} = \left\{(X \setminus A_i, y)\right\}_{i=1}^{k},$$

其中每个 A_i 表示一个不同的混杂范围。该增强过程鼓励模型学习独立于特定混杂因子的推理路径。

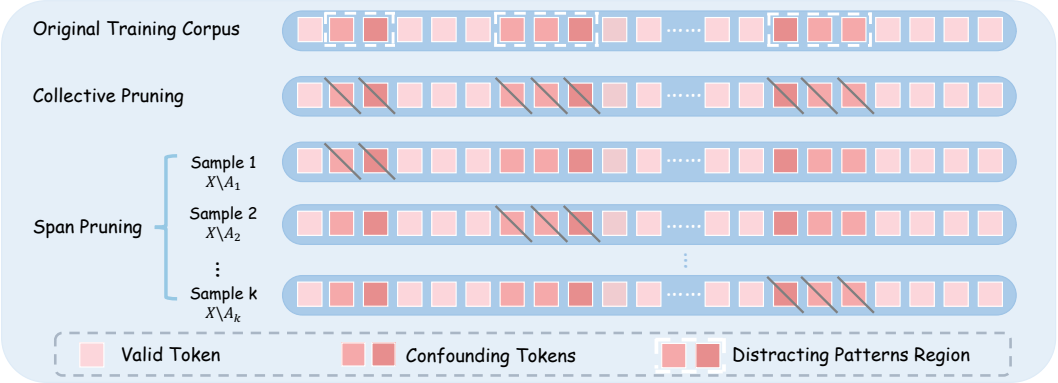


Figure 4: 集合剪枝和跨度剪枝的示例。

2.1.1 因果注意力蒸馏

在生成原始和对比样本后，我们优化两个互补的蒸馏目标以引导学生关注真实的因果依赖关系：

标准蒸馏。在原始指令上对齐学生的输出分布与教师的输出分布：

$$\mathcal{L}_{kd} = D_{\mathrm{KL}}\big(p_T(y \mid X) \,\|\, p_S(y \mid X)\big),$$

其中 p_T 和 p_S 分别表示教师和学生的输出分布。

反事实蒸馏。使学生在反事实指令 $X \setminus A$ 上的输出分布与教师的一致（混淆因子已修剪）：我们通过权重因子 λ 来综合这些目标：其中 $\lambda \in [0, 1]$ 控制标准蒸馏与反事实蒸馏之间的权衡。这个复合损失引导学生在保持语义知识的同时，强制执行真正的因果依赖关系。

对于我们的方法，我们考虑两种变体：（1）指令级修剪：我们只在指令中检测和掩盖混淆的标记，并在指令级掩盖的样本上执行我们的 LeaF。（2）同时进行指令和响应级修剪：我们也可以将先前生成的标记视为上下文输入，并掩盖那些对后续生成具有误导性的标记，以帮助模型生成更准确的续接。因此，我们在指令和先前的生成中同时检测和修剪混淆标记。

Figure 5: 响应拆分策略的说明：语言 CoT，指令级裁剪和响应级裁剪（2 段和 3 段拆分）。高亮的白色区域代表输入，blue underlined areas 代表用于交叉熵损失计算的输出。

3 实验

3.1 实验设置

训练数据集。我们进行实验以评估我们提出的学习聚焦框架（LeaF）方法在数学推理和代码生成任务上的有效性。对于代码生成，我们从 AceCode-87k [**?**] 数据集中随机选择了一个包含 120k 实例的子集。对于数学基准测试，为确保模型在任务中遇到等数量的混扰标记，我

Table 1: 对于 Instruct 模型（LLaMA 3.2–1B/3B-Instruct）和基础模型（Qwen 2.5–Math-1.5B）在 MathBench 和 CodeBench 上的表现，采用了三种剪枝方案（无掩码，Instruct 级别掩码和 Response 级别掩码），其中 Instr Mask 指的是 Instruct 级别剪枝，Resp Mask 指的是 Response 级别剪枝。最好的和第二好的结果分别以粗体和 underlined 标出。

Model	MathBench				CodeBench			
	GSM8K	MATH	Olympiad-Bench	Avg.	Human-Eval+	Leet-Code	Livecode-Bench	Avg.
Teacher Model								
LLaMA3.3-70B-Instruct	95.60	70.40	36.50	67.50	78.05	53.90	45.02	58.99
Qwen2.5-72B-Instruct	95.45	73.80	41.25	70.17	81.71	69.40	54.42	68.51
LLaMA3.2-1B-Instruct								
Instruct Model (Pre-KD)	44.88	24.20	5.79	24.96	29.27	7.22	9.68	15.39
KD w/o Mask	56.79	33.40	8.90	33.03	32.32	6.11	13.74	17.39
LeaF (Instr Mask)	57.70	35.40	10.09	34.40	39.02	6.67	13.60	19.76
LeaF (Instr & Resp Mask)	58.98	35.20	9.94	34.71	39.63	7.22	12.48	19.77
LLaMA3.2-3B-Instruct								
Instruct Model (Pre-KD)	76.88	42.80	13.20	44.29	48.78	13.89	20.34	27.67
KD w/o Mask	82.87	49.00	18.99	50.29	54.88	16.67	24.12	31.89
LeaF (Instr Mask)	83.09	51.80	20.77	51.88	55.49	19.44	25.39	33.44
LeaF (Instr & Resp Mask)	84.69	52.40	22.55	53.21	56.10	21.67	25.81	34.53
Qwen2.5-Math-1.5B								
Base Model (Pre-KD)	65.20	41.40	21.96	42.85	35.37	6.67	1.26	14.43
KD w/o Mask	82.18	67.80	31.16	60.38	41.46	7.78	10.10	19.78
LeaF (Instr Mask)	84.69	68.60	32.79	62.03	42.68	9.94	10.80	20.97
LeaF (Instr & Resp Mask)	85.29	70.60	31.75	62.54	43.29	9.94	13.04	21.92

们从 NuminaMath-CoT [**?**] 中的以下子集中随机选择了 30k 实例：Olympiads [**?**] 、AMC _ AIME [**?**] 、GSM8K [**?**] 和 MATH [**?**] 。

评估数据集。在数学任务的评估中，我们选择了三个难度不同的广泛使用的基准测试，包括 GSM8K [**?**] 、MATH [**?**] 和 OlympiadBench [**?**] 。在代码领域，我们对 HumanEval+ [**?**] 、LeetCode [**?**] 和 LivecodeBench (v4) [**?**] 进行评估，提供对编码性能各个方面的全面评估。关于数学和代码基准测试的详细描述在附录 **??** 中提供。

基础模型和基线。我们在两个不同的模型家族，即 LLaMA 家族 [**? ?**] 和 Qwen 家族 [**?**] 上进行了全面的实验，涵盖了各种规模的模型。对于基于 LLaMA 的实验，我们使用 LLaMA3.2-1B-Instruct [**?**] 和 LLaMA3.2-3B-Instruct [**?**] 作为学生模型，而它们的教师模型设置为 LLaMA3.3-70B-Instruct [**?**] 。对于基于 Qwen 的实验，我们使用 Qwen2.5-Math-1.5 [**?**] 作为学生模型，并以 Qwen2.5-72B-Instruct [**?**] 作为教师模型。我们的比较基线是标准的知识蒸馏方法，其中不涉及剪枝过程。

训练和评估设置。(1) 在训练过程中，模型使用 Alpaca-LoRA 框架进行训练，采用全参数 logits 知识蒸馏和余弦学习率调度，最大学习率为 10^{-5} ，共进行三个周期。LLaMA 模型的批量大小为 64，Qwen 模型的批量大小为 32。详细的超参数和平台信息在附录 **??** 中。(2) 在评估中，教师模型和学生模型在数学和代码任务上使用贪婪解码进行评估。代码任务的最大生成长度为 1024 个 tokens，而数学任务的最大生成长度为 16,384 个 tokens。推理时遵循官方聊天模板。完整的评估细节在附录 **??** 中。

3.2 主要结果

主要结果展示在表 1 中。我们可以从结果中得出几个结论：（1）对于开源基线，标准蒸馏和我们的方法在数学和代码领域的几乎所有任务中都提升了模型的性能。（2）在性能方面，LeaF 在使用相同的训练语料库时始终优于标准知识蒸馏。这表明 LeaF 可能使模型能够更有效地关注关键信息，从而增强其推理能力。（3）从指令级到响应级的剪枝扩展在 LLaMA 和 Qwen 系列的大多数任务中进一步提高了性能，表明响应级的干扰模式影响后续生成。我们假设指令级和响应级的干扰模式不同，同时学习这两者可以增强模型的推理能力。基于响应拆分策略的更详细分析在第 4.2 节中提供。

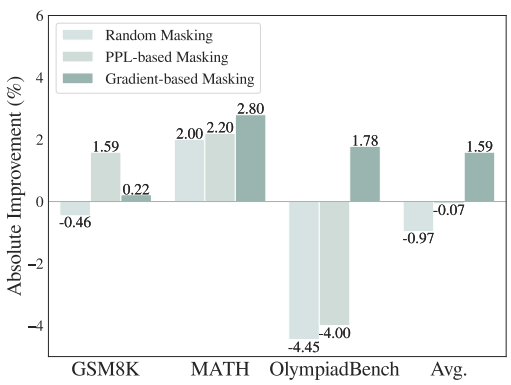


Figure 6: 遮掩策略与基线（KD）相比的准确性提升比较。

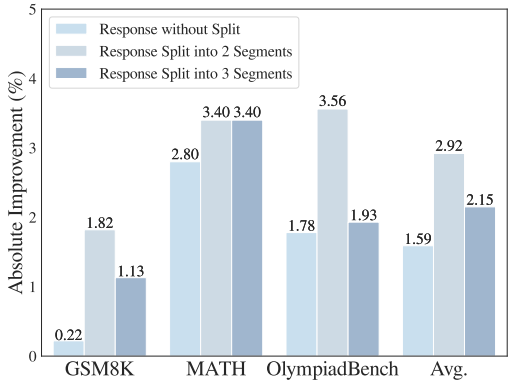


Figure 7: 分割策略与基线（KD）相比的准确性提升比较。

4 进一步分析

在本节中，我们进行掩蔽策略分析（第 4.1 节）、响应拆分分析（第 4.2 节）和阈值敏感性分析（第 4.3 节）。最后，我们开展一个案例研究（第 **??** 节）以说明我们方法的可解释性。

4.1 掩蔽策略分析

为了展示我们的基于梯度的掩码策略的有效性，我们将 LeaF 与另外两种选择的标记掩码进行比较：（1）随机掩码，其中由我们的基于梯度的方法识别的相同数量的标记在每个实例中随机掩码；（2）基于困惑度（PPL）的掩码，其中具有最高困惑度分数的标记以相同的比例进行掩码，遵循与我们的方法中用于生成增强训练数据的数据筛选过程相同的流程。我们在 GSM8K、MATH 和 OlympiadBench 数据集上评估所有三种掩码策略：随机掩码、基于困惑度的掩码和基于梯度的掩码。

我们将结果展示在图 6 中。我们的观察如下：（1）基于梯度的掩码（我们的方法）始终优于两个基准，在 MATH 和 OlympiadBench 上获得最高准确率。（2）随机掩码在 GSM8K 和 Olympiad 上导致性能下降，尽管在 MATH 上略有提高。这表明在没有经过信息选择的情况下天真地掩码标记会削弱蒸馏性能。（3）基于 PPL 的掩码在 GSM8K 和 MATH 上提供了适度的改进，但在 OlympiadBench 上的表现与随机掩码相当，表明其在复杂任务中的局限性。我们的分析是，虽然在简单环境中困惑度可能足以检测混淆标记，但在具有挑战性的基准上缺乏所需的敏感性。这强调了在复杂推理场景中需要高级教师模型来指导标记选择的必要性。

4.2 响应拆分策略

我们比较了三种响应拆分策略，考虑到了在指令和响应层面都存在干扰模式。我们评估了无拆分的响应（指令层面的剪枝）、两段拆分和三段拆分。这些策略的示意图见图 5 。

结果。 根据图 7 中的结果，如下所示：(1) 在响应层面的剪枝（包括两段和三段拆分）显著优于指令层面的剪枝，证明了将混淆标记的学习从指令层面扩展到响应层面的重要性和好处。我们推测，这种改进来自于指令层面和响应层面的干扰模式的差异，这表明结合两个层面可以进一步提升模型性能。(2) 三段拆分的性能可与两段拆分相媲美，表明在响应层面的进一步细分收效减小。我们推测，响应层面的干扰模式呈现出一定的规律性，而两段拆分生成的数据足以让模型有效学习这些模式，从而使额外的细分变得不必要。

4.3 阈值灵敏度分析

我们对用于混淆标记修剪的阈值进行敏感性分析，评估 LLaMA3.2-1B-Instruct 和 LLaMA3.2-3B-Instruct 作为学生模型的性能。如图 8 和图 9 所示，我们展示了这些模型在不同阈值水平下 MathBench（GSM8K, MATH, OlympiadBench）上的平均实验结果。

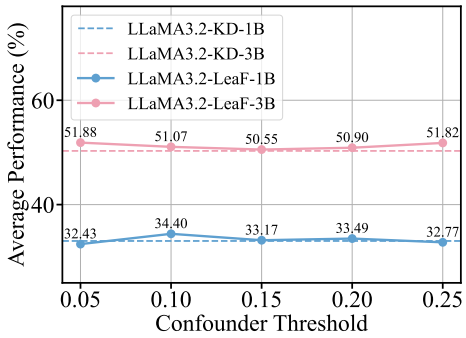


Figure 8: MathBench 中的指导级别。

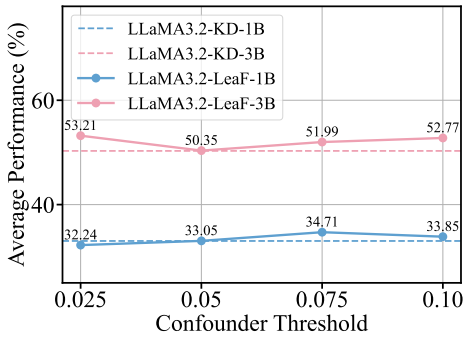


Figure 9: MathBench 中的响应级别。

结果。 我们观察到以下结果：(1) 指令级别：LLaMa3.2-LeaF-1B 在阈值为 0.10 时表现最佳，而 LLaMa3.2-LeaF-3B 在 0.05 时表现最佳。(2) 响应级别：LLaMa3.2-LeaF-1B 在 0.15 时表现最佳，而 LLaMa3.2-LeaF-3B 在 0.10 时表现最佳。(3) 对于指令级别和响应级别，LLaMa3.2-LeaF-1B 在误导性标记阈值较高时实现了最佳性能，而 LLaMa3.2-LeaF-3B 则不是。这表明较小的模型由于更容易受到混淆标记的影响，能够从更高的阈值中受益，这样可以更有效地过滤掉破坏性标记。

我们从可解释性的角度进行了一项案例研究，以显示与标准知识蒸馏（KD）相比，LeaF 能够使模型专注于关键信息，避免混淆标记。可解释性案例研究如图 **??** 所示。它说明了 LeaF 如何使模型更加关注关键信息，例如“实数”、“所有”、“都是实数”。通过识别所有根为实数的要求，LeaF 引导模型通过一个连贯的推理链：识别 $x = -1$ 作为一个明显的实根，并应用判别条件以确保二次因子也产生实解。相对而言，KD 模型忽略了这一约束，并错误地将 AM-GM 不等式应用于可能的负值，导致了一个错误的最终答案。这表明 LeaF 能够使模型更好地专注于关键信息。

推理一致性。一系列现有的研究集中于解码阶段的一致性。自洽性 [**?**] 通过采样多个推理链并采用多数投票来稳定最终答案。为了进一步降低推理成本，自适应一致性 [**?**] 和早期评分自洽性 [**?**] 分别引入了基于狄利克雷和基于窗口的停止标准。推理感知自洽性 [**?**] 通过对样本质量和推理路径重要性加权来增强答案的一致性。然而，这些方法主要集中在稳定最终答案，而没有解决模型在生成过程中的保持和利用关键上下文信息的能力。我们定义一致性为答案稳定性和上下文依从性，并引入一种蒸馏策略，系统地抑制误导信号的影响，增强学生对关键信息的关注，并在推理过程中促进更稳健的上下文依从性。

链式思维知识蒸馏。我们的工作属于知识蒸馏范式 [**? ?**]，这是一种在复杂任务中提升开源模型性能的广泛采用的技术，例如数学问题解决 [**? ?**] 和代码生成 [**? ?**]。CoT-KD [**?**] 首次尝试采用由高能力教师生成的链式思维解释对学生模型进行微调，从而使其具备高级推理能力。最近的进展大致可以分为两个互补的方向：（1）数据导向的方法旨在通过提高训练数据的质量和多样性来增强蒸馏效果（例如，CD [**?**]，SCORE [**?**]，和 Skintern [**?**]）。（2）模型导向的方法通过增强模型架构和推理策略来提高效率和推理能力（例如，PRR [**?**] 和 ATM [**?**]）。然而，现有方法主要关注输出模仿，而我们的方法则显式蒸馏教师在推理过程中捕捉关键信息的能力到学生模型中，从而使学生模型具备上下文感知推理能力。

评论令牌识别。现有研究已经探讨了各种策略来识别和减轻语言模型推理过程中冗余步骤 [**? ? ?**] 或不太重要的令牌 [**? ?**]。诸如 LLMLingua [**?**] 之类的方法依赖于模型的自我评估来确定令牌的重要性，这可能由于模型有限的推理能力而引入偏见。更新的方法进一步在不同训练阶段优化令牌选择。RHO-1 [**?**] 在监督微调阶段引入选择性语言建模（SLM），优先考虑信息性令牌以提高效率和推理准确性。TokenSkip [**?**] 专注于思维链阶段，有选择地跳过低影响令牌以压缩推理路径而不牺牲性能。同时，cDPO [**?**] 通过对比学习隔离关键令牌来增强直接偏好优化。然而，这些方法主要集中在输出令牌上，而忽视了来自指令阶段的上下文信息的影响。相比之下，我们的方法引入了先进的教师模型，通过梯度差异识别混淆令牌，建立指令和输出之间更强的连接，实现更全面和有效的令牌筛选。

我们的工作有以下局限性：（1）对高级教师模型的依赖：我们的方法依赖于一个教师模型来识别混淆符号。探索能够使模型自我改进以精确关注关键符号并提升推理能力的方法是一个有趣且重要的未来研究方向。（2）对长文本任务的有限扩展性：由于学生模型的固有限

制，我们仅在数学和代码任务中验证了我们的方法，将其适用于长文本及其他领域的研究留待未来探讨。

在本文中，我们提出了学习聚焦框架（LeaF），这是一种提高大型语言模型一致性的全新策略。通过利用因果分析和基于梯度的剪枝，我们的方法有效地识别并消除混淆符号，使学生模型能够捕捉更可靠的因果依赖性。实验结果显示，在数学和代码基准测试中无论是准确性还是鲁棒性都有显著提高。在未来的研究中，通过自我改进机制实现模型一致性而不依赖于高级模型仍是一个值得进一步探索的方向。

我们比较了小模型和大模型的梯度热图，分析了在推理过程中它们对关键符号的注意力和关注的差异。

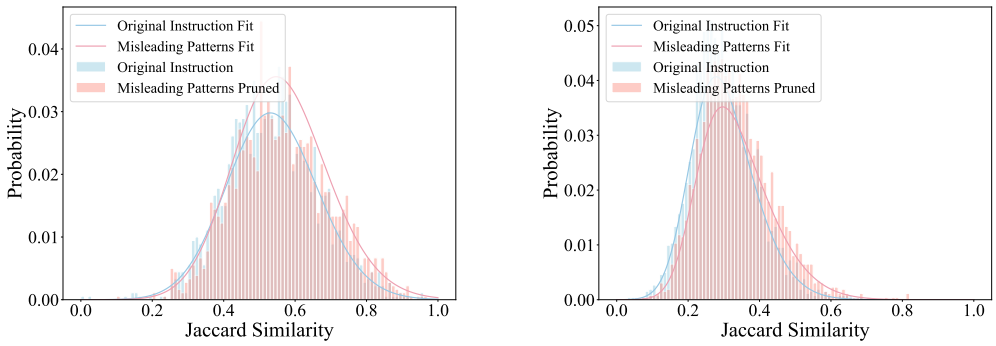
结果。 图 **??** 展示了 Llama3.3-70B-Instruct 成功地捕捉到了一个关键的语境关系："五个蓝莓容器可以用来换两个西葫芦。"梯度热图显示教师模型将注意力紧密对准相关的标记，而学生模型的注意力则更为分散。这个观察促使我们提出假设：通过修剪干扰模式，我们可以引导学生模型更好地关注显著信息，从而增强其推理能力。为了评估这一点，我们在两种设置下进行试点研究：（1）评估在修剪干扰模式后数学和代码基准准确性提升（附录 **??**），以及（2）评估响应质量改善（附录 4.4 ）。

对于 LLaMA 系列，我们使用 LLaMA3.2-1B/3B-Instruct 作为学生模型，LLaMA3.3-70B-Instruct 作为教师模型。对于 Qwen 系列，我们在初步实验中使用 Qwen2.5-Math-1.5B 作为学生模型，Qwen2.5-72B-Instruct 作为教师模型。对于数学问题求解，我们从 Numina-CoT 数据集中选取一个经过筛选的子集，教师模型生成正确推断的部分，包含 12K 个例子（每个来自 GSM8K、奥林匹克、AMC_AIME 和 MATH 中选取 3K 个）。然后，我们选择学生模型产生错误结果的子集。我们计算每个样本在学生模型和教师模型之间的梯度差异，以识别潜在的干扰模式。最终，移除这些干扰模式，并重新评估学生模型以评估由此产生的准确性改进。结果展示在图 1(b) 中。

对于代码生成，我们从 AceCode 数据集 [**?**] 中构建了一个过滤后的 12K 示例子集，其中教师模型生成了正确的推理。这个子集包含来自 Evol 子集的 6K 样本和来自 Oss 子集的 6K 样本。我们应用与数学领域中相同的干扰模式识别和修剪程序。随后，我们重新评估学生模型，以评估这种干预对代码生成准确性的影响。结果在图 1(a) 中展示。

4.4 通过词元修剪提升生成质量

我们在两种条件下分析学生模型输出与教师模型输出之间的一致性：(1) 原始指令：学生模



(a) Jaccard Similarity in Code Corpus (b) Jaccard Similarity in Math Corpus

Figure 10: 学生模型响应（原始版本与经过指令裁剪的干扰模式）与数学和代码数据集上的真实响应之间的 Jaccard 相似度分布。

型基于原始指令生成输出。(2) 指令中剔除干扰模式：从一个混淆令牌被屏蔽的指令中学生模型生成输出。为了量化模型输出之间的相似性，我们使用了两个广泛采用的度量标准：Jaccard 相似度 [**?**] 。这个度量标准可以有效评估学生模型的输出与教师模型所传达的上下文意义之间的一致性。

结果。 图 10 显示了在去除混淆标记后，学生模型在代码和数学任务上生成的响应的 Jaccard 相似度分布的变化。这表明，通过忽略干扰模式，学生模型不仅提高了推理准确性（图 1），而且生成的响应更符合教师模型，从而提升了输出质量。

我们比较了两种剪枝策略（图 4）：集体剪枝，同时屏蔽所有识别出的混淆标记，和跨距剪枝，仅屏蔽连续的标记跨距。表格 **??** 给出了在两个剪枝方案下，基于数学语料库（86K）训练并在 MATH-500 上评估的 LLaMA3.2-1B/3B-Instruct 模型的结果。(1) 我们发现跨距剪枝显著优于集体剪枝和本地知识蒸馏。(2) 集体剪枝不仅未能带来改进，甚至在 LLaMA3.2-3B-Instruct 模型上降低了性能。我们怀疑同时剪去所有混淆标记扰乱了训练数据的语义连贯性，削弱了模型的学习。因此，我们在所有后续实验中采用跨距剪枝策略。知识蒸馏的完整训练超参数见表格 **??** 。

对于数学问题解决，我们使用 Step-DPO 框架评估 GSM8K、MATH 和 OlympiadBench，并对其数据提取不准确的问题进行了修改。对于代码生成，我们在 HumanEval(+) 和 LeetCode 上使用 EvalPlus 报告 pass@1，在 LiveCodeBench 上使用 Skythought-Evals 框架报告 pass@10。GSM8K 包含 8500 个小学水平的文字题目，每个题目需要 2-8 步基本算术。其自然语言多样性和多步骤结构使其成为链式思维提示的标准衡量指标。

MATH [**?**] 包含 12500 道竞赛风格的问题，分为七个主题（初等代数、代数、数论、计数 & 概率、几何学、中级代数、预备微积分）。每道题都有详细的解答。

OlympiadBench [**?**] 最初是一个包含 8476 个奥林匹克级别数学和物理问题的双语多模态集合。我们筛选掉了基于证明和图像的问题，以获得 674 个纯文本任务，从而能够集中评估高级符号推理能力。

HumanEval+ [**?**] 扩展了原始的 HumanEval，增加了额外的 Python 编程任务和增强的单元测试，旨在针对多样的代码模式实现功能上的正确性。

LeetCode [**?**] 从 LeetCode 平台中选择现实世界的算法挑战，如数组、树、动态规划等，以评估模型生成正确且高效解决方案的能力。

LiveCodeBench [**?**] 提供了一套大规模的真实世界编码任务，配有全面的单元测试和人工偏好注释，从而可以评估功能准确性和编码风格。

5 开源指令模型

下面提供了四种开源模型的下载链接：

- Llama-3.2-1B-Instruct: https://huggingface.co/meta-llama/Llama-3.2-1B-Instruct
- Llama-3.2-3B-Instruct: https://huggingface.co/meta-llama/Llama-3.2-3B-Instruct
- Llama-3.3-70B-Instruct: https://huggingface.co/meta-llama/Llama-3.3-70B-Instruct
- Qwen2.5-Math-1.5B: https://huggingface.co/Qwen/Qwen2.5-Math-1.5B
- Qwen2.5-72B-Instruct：https://huggingface.co/Qwen/Qwen2.5-72B-Instruct