WHEN MODELS STRUGGLE WITH NEGATION: LEARNING FROM SOUND AND UNSOUND REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

Direct Preference Optimization (DPO) is only as effective as the preference data it learns from. As human annotation is expensive, synthetic preference pipelines increasingly rely on large language models (LLMs) to generate candidate responses and on LLM judges to identify preferred ones. Yet this pipeline implicitly assumes that the generator can already produce sufficiently reliable responses for the target capability. When models struggle with that capability, preference construction becomes circular: it presupposes the very capability that post-training is intended to improve. We study this problem through negation, a pervasive linguistic phenomenon that remains challenging for LLMs across multiple tasks. We introduce PACE, a framework that constructs preference data without requiring models to first solve the target problem reliably. Given known correct and incorrect answers, PACE elicits corresponding justifications that reflect contrasting sound and plausible but unsound reasoning. PACE further uses pairwise comparisons with Bradley–Terry aggregation to rank candidate justifications within each answer-conditioned set while mitigating positional bias. Across five negation-focused benchmarks spanning question answering, natural language inference, commonsense reasoning, and information retrieval, DPO training on approximately 5K PACE preference pairs yields gains across four Llama and Qwen backbone LLMs, with statistically significant improvements in 17 out of 20 model-benchmark settings. Controlled experiments show that removing justifications conditioned on incorrect answers consistently weakens performance, demonstrating that plausible but incorrect reasoning provides an important learning signal. Our results suggest that reliable preference construction is a critical prerequisite for effective preference optimization when models struggle with the target capability. To our knowledge, we release the first preference dataset for negation understanding, along with our implementation at `https://anonymous.4open.science/r/NEG`.

1 INTRODUCTION

Preference-based post-training is a promising approach for improving large language models (LLMs). Direct Preference Optimization (DPO) directly learns from preference data without explicit reward modeling or reinforcement learning (Rafailov et al., 2023; Ziegler et al., 2019; Ouyang et al., 2022). Its effectiveness, however, critically depends on the quality of the preference data (Morimura et al., 2024; Zhang et al., 2026). As large-scale human annotation is costly, recent synthetic pipelines increasingly use LLMs to generate candidate responses and LLM judges to identify preferred ones (Cui et al., 2023). This standard pipeline therefore places two upstream requirements on preference construction: generators must produce informative candidate responses, and preference extractors must rank those candidates reliably.

However, neither requirement is guaranteed to hold. Standard synthetic pipelines implicitly assume that the generator can already produce sufficiently reliable responses for the target capability. When this assumption fails, preference construction becomes circular: it presupposes the very capability that post-training is intended to improve, resulting in noisy preference data. Preference extraction can also be unreliable. Although LLM judges provide a scalable alternative to human annotation, their rankings may be unreliable when they struggle with the target capability and sensitive to response order, leading to positional bias (Wang et al., 2024b; Shi et al., 2025). These two failure modes motivate us to reconsider both stages of preference construction.

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

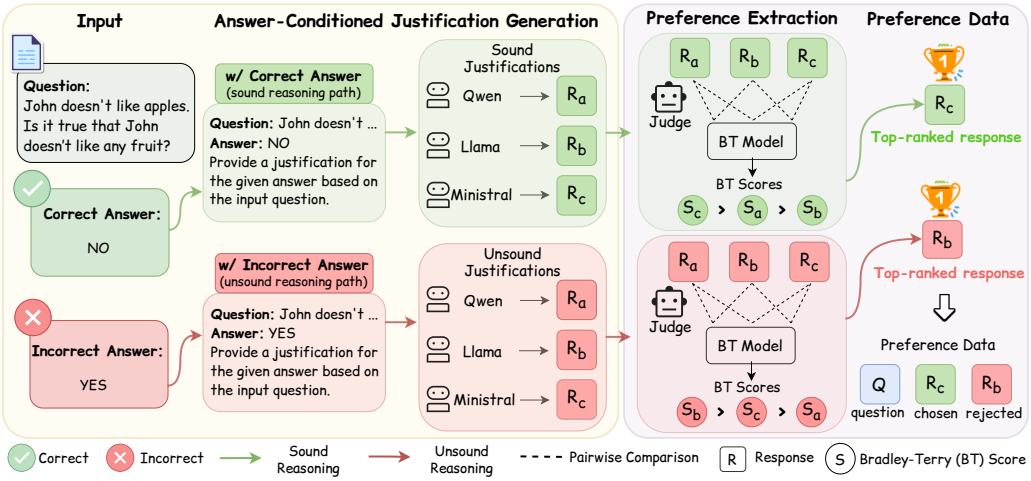

Figure 1: Overview of **PACE**. We generate answer-conditioned justifications to obtain preference data including sound and unsound reasoning paths. Preference extraction relies on pairwise comparisons with an LLM-as-a-judge and Bradley–Terry (BT) ranking. This figure omits the filtering step.

The first failure mode is particularly acute for negation. Negation is a common linguistic phenomenon, appearing in 22–30% of English sentences depending on domain and genre (Hossain et al., 2022). Correctly interpreting negation requires sensitivity to subtle changes in meaning, where small errors can reverse the intended conclusion (Varshney et al., 2025; Asher & Bhar, 2024). Prior work has shown that LLMs perform significantly worse on the same tasks when the input contains negation (Chen et al., 2023; Seo et al., 2025; Kim et al., 2025). Negation therefore exposes a key weakness of synthetic preference construction: standard free-form generation requires models to reliably use the very capability that preference-based post-training is intended to improve.

Prior work on negation has largely focused on benchmark construction, task-specific fine-tuning, and additional pre-training, while existing negation resources typically provide task labels rather than preference supervision (Geiger et al., 2020b; Rezaei & Blanco, 2025). Preference-based post-training therefore offers a complementary direction, but leaves open how to construct reliable preference data when the generator itself is unreliable on negation.

These limitations motivate two research questions: **RQ1:** How can we construct informative candidate responses when models are unreliable on the target capability? **RQ2:** How can we reliably extract preferences from these candidates while mitigating systematic biases from LLM judges?

To address these questions, we introduce **PACE** (**P**reference construction via **A**nswer-**C**onditioned generation and Pairwise **E**valuation), a framework for constructing preference data when models are unreliable on the target capability (Figure 1). For RQ1, rather than asking models to freely solve each problem, PACE conditions generation on known correct and incorrect answers and elicits multiple candidate justifications for each. These candidates reflect contrasting sound and plausible but unsound reasoning for correct and incorrect answers, respectively, thereby bypassing the need for the generator LLM to reliably solve the target problem. For RQ2, PACE uses pairwise comparisons with Bradley–Terry aggregation to rank candidate justifications within each answer-conditioned set while mitigating positional bias. The top-ranked correct- and incorrect-answer-conditioned justifications are then paired as the chosen and rejected responses for DPO.

Across five negation-focused benchmarks spanning question answering, natural language inference, commonsense reasoning, and information retrieval, DPO training on approximately 5K PACE preference pairs yields gains for Llama and Qwen models, with 17 of 20 gains statistically significant. Further experiments show that removing justifications conditioned on incorrect answers consistently weakens performance, while pairwise comparison with Bradley–Terry aggregation reduces positional bias and generally yields stronger downstream performance. Our contributions are as follows:

- We identify a key challenge in synthetic preference construction when models are unreliable on the target capability, and study this problem in a challenging setting: negation in text.

- We (a) show that existing strategies for constructing preference data underperform on negation understanding and (b) present PACE, an answer-conditioned framework that elicits justifications reflecting sound and unsound reasoning for negation post-training.
- We use pairwise comparisons with Bradley–Terry (BT) aggregation to obtain stable preference labels, thereby mitigating positional bias in LLM-based preference extraction.
- We show that LLMs post-trained on PACE preference data consistently outperform strong baselines across multiple downstream tasks, achieving state-of-the-art performance on several. We release, to our knowledge, the first preference dataset for negation understanding.

2 Related Work

Negation is a fundamental linguistic phenomenon that appears across human languages (Horn & Wansing, 2025). Recent work has examined whether language models can understand and reason over negation, introducing challenging benchmarks across multiple downstream tasks (Geiger et al., 2020a; Truong et al., 2022b; Ravichander et al., 2022; García-Ferrero et al., 2023). These studies show that negation remains challenging even for modern LLMs, with performance often degrading when negation affects the intended meaning (Asher & Bhar, 2024; Varshney et al., 2025; Seo et al., 2025; Kim et al., 2025). Prior work has sought to improve negation understanding through paraphrasing, task-specific fine-tuning and prompting, and additional pre-training (Hosseini et al., 2021; Hossain & Blanco, 2022; She et al., 2023). However, preference-based post-training for negation remains largely unexplored. Our work studies this direction while addressing the additional challenge of constructing reliable preference data when models themselves struggle with negation. Additional related work on negation is discussed in Appendix C.

Preference learning. Preference-based post-training has become a widely used paradigm for aligning LLMs with human preferences (Ouyang et al., 2022; Tie et al., 2025). Unlike standard reinforcement learning from human feedback (Christiano et al., 2017; Bai et al., 2022a), Direct Preference Optimization (DPO) directly optimizes preference pairs without an explicit reward model or reinforcement learning (Rafailov et al., 2023). Its simplicity has made it a widely used approach for preference-based post-training. Our work is complementary to preference-optimization methods: we use DPO as the downstream optimizer and focus instead on constructing reliable preference data when models are unreliable on the target capability.

Synthetic preference data construction. The effectiveness of preference-based post-training depends critically on preference-data quality (Albalak et al., 2024; Zhang et al., 2026). Because large-scale human annotation is costly, recent work increasingly uses LLMs to generate candidate responses and extract preferences (Bai et al., 2022b; Cui et al., 2023). Recent studies further show that the quality, selection, and composition of preference pairs substantially affect downstream preference optimization (Morimura et al., 2024; Wang et al., 2025; Zhang et al., 2026). However, these interventions generally operate after candidate responses have already been generated. PACE instead addresses an earlier failure mode: when the generator is unreliable on the target capability, free-form generation itself may fail to provide reliable candidate responses. By conditioning justification generation on known correct and incorrect answers, PACE constructs informative candidate responses without requiring the generator to first solve the target problem reliably.

Preference extraction strategies. LLM-as-a-judge has been widely adopted to reduce the cost of manual annotation (Zheng et al., 2023; Gu et al., 2024). However, automatic judging can suffer from ranking instability and positional bias (Wang et al., 2024a; Koo et al., 2024). Prior work has used stronger proprietary judges and repeated evaluations to improve reliability, at higher inference cost. PACE instead uses pairwise comparisons with Bradley–Terry aggregation to extract preferences while mitigating positional bias, without relying on a larger judge. Additional related work on preference learning, data construction, and preference extraction is discussed in Appendix C.

3 Methodology

We begin by briefly reviewing the standard pipeline for synthetic preference data construction, which typically consists of two components: response generation and preference extraction. We then introduce our proposed **PACE** framework (Figure 1); the full algorithm is in Appendix G.

Background: Standard Pipeline for Synthetic Preference Data Construction. In a standard pipeline for synthetic preference data construction (Cui et al., 2023; Chiang et al., 2024; Ji et al., 2025), one or more LLMs are used to generate a set of k candidate responses $\mathcal{R}$ conditioned on a prompt x: $\mathcal{R} = \{r_i\}_{i=1}^{k}, \quad r_i \sim p(r \mid x)$, where $p(r \mid x)$ denotes the model's output distribution. These responses are then annotated with preference labels, either by human annotators or automatically using an LLM-as-a-judge, a process commonly referred to as preference extraction. To reduce costs, many recent approaches adopt LLM-as-a-judge for automatic preference labeling (Zheng et al., 2023; Gu et al., 2024; Li et al., 2025). In particular, a common strategy is to ask an LLM judge to rank all candidate responses $\mathcal{R}$ and select the most and least preferred responses as the *c*hosen and *r*ejected responses. This results in a preference pair (r_c, r_r) for the given prompt x; the preference relation is denoted as $r_c \succ r_r$. Collecting such pairs over a collection of prompts results in a preference dataset $\mathcal{D} = \{(x^{(i)}, r_c^{(i)}, r_r^{(i)})\}_{i=1}^{n}$, which can be used for post-training.

The standard pipeline for preference data construction can be easily automated but has several limitations. First, if generating a response to the given prompt requires language understanding that the LLM has not mastered (e.g., negation), the generated responses may be unreliable or exhibit flawed reasoning, resulting in noisy preference data. Second, during the preference extraction stage, direct selection by a judge model is prone to positional bias.

3.1 PACE: Constructing Preference Data When Models Struggle

We propose PACE, **P**reference construction via **A**nswer-**C**onditioned generation and Pairwise **E**valuation, a framework that addresses the limitations of the standard pipeline, as shown in Figure 1. First, we use LLMs to generate justifications of correct and incorrect answers for problems that require negation understanding, thereby capturing sound and plausible but unsound reasoning. Second, we filter low-quality or inconsistent justifications. Third, we extract preferences by ranking justifications reflecting sound and unsound reasoning while mitigating positional bias, then pair the highest-ranked justifications with their corresponding answers to construct preference data.

Generating Answer-Conditioned Justifications. While the standard pipeline prompts LLMs for a response to a prompt x (e.g., the solution to a problem), PACE prompts LLMs to generate a justification for an answer y to the prompt x, where y could be correct (y^+) or incorrect (y^-). We refer to the LLM output conditioned on a prompt x and answer y as *justification* rather than *response* when necessary to point out that the LLM response is a justification for the given answer.

We prompt multiple *generator* LLMs to generate justifications that support the given answer, where each justification is sampled as $j \sim p(j \mid x, y)$. Formally, to construct sets of sound ($\mathcal{R}^+$) and unsound ($\mathcal{R}^-$) responses, we generate justifications conditioned on the prompt x and the ground truth answer (y^+) or the flipped answer (y^-, an incorrect answer). Here, r_i^+ combines the correct answer y^+ with its justification j_i^+, while r_i^- combines the incorrect answer y^- with its justification j_i^-:

$$\mathcal{R}^+ = \{r_i^+\}_{i=1}^{k}, \quad j_i^+ \sim p(j \mid x, y^+), \quad \mathcal{R}^- = \{r_i^-\}_{i=1}^{k}, \quad j_i^- \sim p(j \mid x, y^-).$$

Although the two generations are conditioned on the same prompt x but different answers (the correct or incorrect answer), this subtle difference induces meaningful differences in the generated justifications. Justifications in $\mathcal{R}^+$ display sound reasoning, while justifications in $\mathcal{R}^-$ illustrate plausible but unsound reasoning, since justifications for incorrect answers necessarily involve flawed reasoning. This design captures contrasting ways in which a model may succeed or fail on the target capability. For example, in negation understanding, a model may reach an incorrect answer by misinterpreting the scope of negation or its interaction with the rest of the sentence. This step therefore provides the contrasting sound and unsound justifications needed to construct preference pairs. We refer the reader to Appendix D.1 for the prompt used to generate justifications. As we shall see (Section 4), including unsound-reasoning justifications in preference data is crucial.

Filtering Justifications. LLMs are not guaranteed to follow instructions. In the previous step, they sometimes do not generate anything, generate nothing but a written refusal to generate a justification (even when conditioned on correct answers), or simply repeat the prompt x. We define two filters to alleviate this issue. First, we implement a battery of deterministic rules to filter out, among others, empty "justifications" and justifications that support an answer different from the conditioned answer.

Second, we enforce a consistency check. To this end, we use a *judge* LLM. For each pair of responses conditioned on a prompt x and either the correct (y^+) or incorrect (y^-) answer, we present the pair to the judge in both orders, (r_a, r_b) and (r_b, r_a), and ask which justification better supports the provided answer. We discard both responses if the judge's preference changes with order. Importantly, this filtering reflects judgment consistency rather than task difficulty. See Appendix D.2 for details.

Extracting Preferences via Pairwise Comparisons and Bradley-Terry Ranking. Standard preference data construction typically relies on a judge model, often a large and proprietary LLM, to directly select preferred responses. However, this approach suffers from positional bias, especially when multiple candidates are compared simultaneously (Zhu et al., 2024).

To address these issues, we propose a two-step preference extraction procedure, as illustrated in Figure 1. Our design has two objectives. First, to reduce positional bias and obtain more reliable preference signals. Second, to enable smaller judge models to produce stable preference rankings comparable to those obtained with larger models via direct ranking of all responses.

The first step uses the judge LLM to rank all *pairs* of responses (r_a, r_b) generated conditioned on a problem x and either the correct (y^+) or incorrect answer (y^-). This step results in pairwise rankings of the sound and plausible but unsound candidates generated for a problem x. The second step generates an overall ranking of these candidates using the Bradley–Terry (BT) model (Bradley & Terry, 1952), a probabilistic model that assigns a latent score to each response by combining the pairwise rankings. More formally, given the pairwise rankings, the BT model estimates a latent score (BT score) s_i for each response r_i. The probability that r_a is preferred over r_b is defined as:

$$P(r_a \succ r_b) = \frac{\exp(s_a)}{\exp(s_a) + \exp(s_b)}.$$

Based on the BT scores, we select the highest-scoring responses from $\mathcal{R}^+$ and $\mathcal{R}^-$ as the chosen and rejected responses for a given problem x, denoted as r_c and r_r, respectively. Since responses within each set ($\mathcal{R}^+$ or $\mathcal{R}^-$) are conditioned on the same answer (correct or incorrect), we select the justifications that better support the provided answer. Thus, selecting the top-ranked response from each set yields the most informative sound and plausible but unsound candidates. The final preference data consist of triples (x, r_c, r_r): a problem, a chosen response, and a rejected response.

4 Experiments and Results

In this section, we evaluate PACE on negation understanding, a capability that LLMs struggle with. First, we detail the datasets and implementation, including the construction of preference data through two key stages: response generation and preference ranking. We then evaluate PACE across five downstream benchmarks and study the effects of different generation and ranking strategies.

4.1 Experimental Setup

Source Corpora to Generate Justifications. To construct a negation-focused preference dataset, we start with the training splits of three existing benchmarks targeting negation in different tasks: CondaQA (Ravichander et al., 2022), TIND (García-Ferrero et al., 2023), and ScoNe (She et al., 2023). We randomly sample instances from each dataset, ensuring a balanced distribution from each dataset and label. We then apply our framework to construct preference data via answer-conditioned justification generation and preference extraction, as described in Section 3. The result is approximately 5K preference pairs; Appendix D provides further details, including human evaluation.

Models. PACE uses LLMs for two roles (Section 3). The *generator* is used to generate justifications conditioned on a problem x and (correct or incorrect) answer y. We use small LLMs from three different families as generators: Qwen3-4B (Yang et al., 2025), Llama3.2-3B (Grattafiori et al., 2024), and Ministral3-3B (Liu et al., 2026). This choice minimizes cost (small sizes) while generating diverse justifications. In the preference extraction stage, we use a *judge* LLM to rank pairs of justifications. We adopt Llama3-8B (Grattafiori et al., 2024) in the main experiments. Additionally, we also consider a larger judge, Qwen3-32B (Yang et al., 2025). Finally, we evaluate the benefits of post-training with PACE using decoder-only *backbone* LLMs. We use two model families and sizes: Llama3.2-3B, Llama3-8B, Qwen2.5-3B, and Qwen2.5-7B (instruct variants).

Table 1: Accuracy (%) of off-the-shelf backbone LLMs and models post-trained on preference data constructed by PACE or two response-generation variants. Δ indicates change from the off-the-shelf model; * indicates a significant improvement of PACE over the off-the-shelf model under McNemar's test ($p < 0.05$). *Best Prior* reports the best published decoder-only LLM prompting result. Best per-model results are bolded, and new state-of-the-art results are highlighted. Not generating responses conditioned on answers is always detrimental (one exception: Llama3.2-3B on NevIR), as is not conditioning on incorrect answers (no exceptions). PACE improves all LLMs across all benchmarks.

		CondaQA		ScoNe		TIND		NeQA		NevIR	
		Acc.	(Δ)	Acc.	(Δ)	Acc.	(Δ)	Acc.	(Δ)	Acc.	(Δ)
Best Prior	(13B–175B LLMs)	54.00		64.00		57.80		71.00		n/a	
Llama3.2-3B	Off-the-shelf	58.56		52.08		51.99		63.00		12.65	
	Post-trained w/ PACE	**60.19**	(+1.63)*	**60.33**	(+8.25)*	**53.63**	(+1.64)*	**71.00**	(+8.00)*	13.38	(+0.73)
	w/o answers	51.73	(-6.83)	45.50	(-6.58)	52.95	(+0.96)	70.00	(+7.00)	**18.22**	(+5.57)
	w/o incorrect answers	55.41	(-3.15)	57.25	(+5.17)	53.15	(+1.16)	69.00	(+6.00)	7.66	(-4.99)
Llama3-8B	Off-the-shelf	52.86		61.17		54.87		61.00		4.63	
	Post-trained w/ PACE	**54.90**	(+2.04)*	**83.92**	(+22.75)*	**64.51**	(+9.64)*	**79.00**	(+18.00)*	**20.39**	(+15.76)*
	w/o answers	33.43	(-19.43)	50.00	(-11.17)	53.82	(-1.05)	67.00	(+6.00)	15.91	(+11.28)
	w/o incorrect answers	17.25	(-35.61)	50.00	(-11.17)	54.57	(-0.30)	62.00	(+1.00)	15.47	(+10.84)
Qwen2.5-3B	Off-the-shelf	55.12		56.75		51.25		50.00		27.04	
	Post-trained w/ PACE	**57.56**	(+2.44)*	**69.17**	(+12.42)*	**55.52**	(+4.27)*	**56.00**	(+6.00)*	**29.14**	(+2.10)*
	w/o answers	51.80	(-3.32)	50.83	(-5.92)	50.20	(-1.05)	54.00	(+4.00)	25.02	(-2.02)
	w/o incorrect answers	51.01	(-4.11)	51.33	(-5.42)	51.65	(+0.40)	55.00	(+5.00)	23.79	(-3.25)
Qwen2.5-7B	Off-the-shelf	41.67		71.33		61.67		46.00		40.93	
	Post-trained w/ PACE	**42.64**	(+0.97)*	**73.00**	(+1.67)*	**62.12**	(+0.45)*	**53.00**	(+7.00)	**41.36**	(+0.43)
	w/o answers	40.76	(-0.91)	71.00	(-0.33)	61.69	(+0.02)	48.00	(+2.00)	40.71	(-0.22)
	w/o incorrect answers	40.46	(-1.21)	71.33	(+0.00)	61.99	(+0.32)	52.00	(+6.00)	41.07	(+0.14)

Training Setup. For DPO post-training, we follow Rafailov et al. (2023) with $\beta = 0.1$ and adopt Low-Rank Adaptation (LoRA) (Hu et al., 2022). We use a unified set of hyperparameters across all models without extensive tuning and select the best checkpoint based on validation reward accuracy. Additional training details are provided in Appendix E.

Baselines and Ablations. We evaluate our framework by comparing off-the-shelf backbone LLMs with models post-trained on preference data constructed using PACE and its variants. Specifically, we have three options for **response generation**:

- PACE, which generates justifications reflecting both sound and unsound reasoning.
- The standard pipeline in which responses are generated conditioned only on the prompt x (i.e., the problem without answer, Section 3).
- A variant of PACE in which justifications are generated conditioned on the prompt x and correct answer (y^+). This variant discards $\mathcal{R}^-$ (i.e., unsound reasoning).

For **preference extraction**, we experiment with the following:

- PACE, using Llama3-8B as the judge for pairwise comparisons and BT aggregation.
- Direct selection with Llama3-8B, in which the same judge as PACE directly selects the best response from a candidate set (from $\mathcal{R}^+$ or $\mathcal{R}^-$) in a single pass.
- Direct selection with Qwen3-32B, which uses the same strategy as the previous setting but with a much larger judge to examine whether judge scale improves preference extraction.

Evaluation. Backbone models (off-the-shelf and post-trained with several variants of preference data) are evaluated with the test splits of five benchmarks that require negation understanding: CondaQA, TIND, ScoNe, NeQA (Zhang et al., 2023), and NevIR (Weller et al., 2024). These benchmarks span a range of tasks such as question answering, natural language inference, commonsense reasoning, and information retrieval, with a focus on negation understanding. We adopt a zero-shot evaluation setting with a unified prompting format across all models. For fair comparison, we follow the evaluation metrics used in the original benchmark papers. See Appendix H for additional details.

4.2 POST-TRAINING WITH PACE AND ALTERNATIVE RESPONSE GENERATIONS

Table 1 presents results with four off-the-shelf LLMs and after post-training with PACE on five negation-oriented benchmarks covering several tasks (Section 4.1).

Post-training with preference data obtained with PACE is beneficial across all LLMs and benchmarks, and outperforms prior work. We observe consistent improvements across both model families (Llama and Qwen) and sizes (3B and 7/8B). Moreover, 17 of the 20 improvements over the corresponding off-the-shelf models are statistically significant under McNemar's test (McNemar, 1947) ($p < 0.05$), even when some absolute gains are modest. While larger off-the-shelf models generally outperform their smaller counterparts (exceptions: Qwen on CondaQA and NeQA), scaling alone is insufficient for robust negation understanding. PACE also remains highly effective at larger model scales, with Llama3-8B gaining substantially more than Llama3.2-3B on several benchmarks.

For NevIR, originally designed for information retrieval, we use the same metric as the original work, pairwise accuracy, whose random baseline is 25%. Because the paired queries differ primarily in negation, models that overlook the negation can produce the same ranking for both queries, resulting in low pairwise accuracy. This is consistent with the original NevIR finding that nearly all evaluated models perform below random ranking (Weller et al., 2024). For the remaining benchmarks, prior best prompting results rely on substantially larger decoder-only models (e.g., Vicuna-13B (Chiang et al., 2023) and InstructGPT-175B (Ouyang et al., 2022)), whereas PACE enables much smaller 3B–8B decoder-only models to outperform them under the same evaluation setting.

Post-training for negation requires sound and unsound reasoning. PACE is grounded in collecting preference pairs containing justifications for correct and incorrect answers (i.e., sound and unsound reasoning). Replacing the Answer-Conditioned Justification Generation step (Section 3.1) with either generation without answer conditioning or generation conditioned only on correct answers leads to worse results in almost all settings. The drops are more substantial with models that benefit more from post-training with PACE. In fact, post-training with preference data derived from justifications generated without conditioning on answers or incorrect answers is often detrimental (i.e., the off-the-shelf counterparts obtain better results). For example, Llama3.2-3B accuracy on ScoNe drops from 60.33 to 45.50 (w/o answers) and 57.25 (w/o incorrect answers). Crucially, these changes do not stem from the amount of preference data collected, as in all settings we cap the amount to 5K preference pairs (Section 4.1). We hypothesize the benefits of post-training with justifications conditioned on correct and incorrect answers are due to:

- LLMs are unreliable when the input contains negation (Section 2), thus not conditioning on an answer leads to low-quality or incorrect responses. Further, it is impossible to know which reasoning paths are unsound, resulting in noisy preference data.
- Understanding how reasoning involving negation fails is as important as how it succeeds. Prior work (Zhu et al., 2025) shows that incorporating negative signals can improve reasoning. PACE introduces a critical modification: generating justifications conditioned on both correct and incorrect answers. Thus, post-training exposes LLMs to sound and unsound reasoning paths. Empirically, learning what is wrong is as important as what is correct.

4.3 PREFERENCE EXTRACTION: PACE AND DIRECT SELECTION

Having established that generating preference data with sound and unsound reasoning is required for negation-focused post-training, we now explore preference extraction strategies. Recall that PACE (Section 3.1) uses (a) pairwise comparisons with Llama3-8B as a judge and (b) BT ranking to obtain the global ranking. An obvious alternative is to directly use an LLM-as-a-judge to create the global ranking. To this end, we experiment with Llama3-8B and Qwen3-32B (Table 2).

Pairwise comparison with BT ranking outperforms direct selection with Llama3-8B as the judge. Across most settings, extracting preferences with Llama3-8B as a judge for pairwise comparisons coupled with BT ranking outperforms ranking directly. The only exception is a minor drop on ScoNe (60.33 vs. 60.83). Further, direct selection exhibits unstable behavior, and in some cases even degrades performance below the off-the-shelf Llama3.2-3B (e.g., NevIR). These results indicate that aggregating pairwise comparisons via BT provides a more reliable mechanism for identifying high-quality responses from the same candidate pool, reflecting both sound and unsound reasoning.

Table 2: Accuracy scores for off-the-shelf backbone LLMs and models post-trained with preferences extracted using (a) pairwise comparisons and BT ranking (PACE) and (b) direct selection with Llama3-8B and Qwen3-32B. Here, Δ and * are defined as in the previous table, while the following annotations differ. Bold indicates that PACE outperforms direct selection with the same judge (Llama3-8B), and highlighting indicates that PACE outperforms direct selection with the much larger Qwen3-32B. PACE outperforms or obtains comparable results to direct selection with either judge.

		CondaQA	ScoNe	TIND	NeQA	NevIR
Backbone LLM	Setting	Acc. (Δ)	Acc. (Δ)	Acc. (Δ)	Acc. (Δ)	Acc. (Δ)
Llama3.2-3B	Off-the-shelf	58.56	52.08	51.99	63.00	12.65
	Post-trained, preferences extracted via:					
	Llama3-8B, pairwise & BT	**60.19** (+1.63)*	60.33 (+8.25)*	**53.63** (+1.64)*	**71.00** (+8.00)*	**13.38** (+0.73)
	Llama3-8B, direct selection	59.60 (+1.04)	**60.83** (+8.75)	53.31 (+1.32)	70.00 (+7.00)	9.62 (-3.03)
	Qwen3-32B, direct selection	59.41 (+0.85)	60.75 (+8.67)	53.33 (+1.34)	71.00 (+8.00)	11.42 (-1.23)
Llama3-8B	Off-the-shelf	52.86	61.17	54.87	61.00	4.63
	Post-trained, preferences extracted via:					
	Llama3-8B, pairwise & BT	**54.90** (+2.04)*	**83.92** (+22.75)*	**64.51** (+9.64)*	79.00 (+18.00)*	**20.39** (+15.76)*
	Llama3-8B, direct selection	53.02 (+0.16)	77.42 (+16.25)	61.16 (+6.29)	79.00 (+18.00)	19.67 (+15.04)
	Qwen3-32B, direct selection	57.35 (+4.49)	78.00 (+16.83)	59.39 (+4.52)	76.00 (+15.00)	19.67 (+15.04)
Qwen2.5-3B	Off-the-shelf	55.12	56.75	51.25	50.00	27.04
	Post-trained, preferences extracted via:					
	Llama3-8B, pairwise & BT	**57.56** (+2.44)*	**69.17** (+12.42)*	**55.52** (+4.27)*	56.00 (+6.00)*	**29.14** (+2.10)*
	Llama3-8B, direct selection	56.49 (+1.37)	66.58 (+9.83)	55.19 (+3.94)	56.00 (+6.00)	27.84 (+0.80)
	Qwen3-32B, direct selection	56.60 (+1.48)	66.33 (+9.58)	54.95 (+3.70)	56.00 (+6.00)	28.71 (+1.67)
Qwen2.5-7B	Off-the-shelf	41.67	71.33	61.67	46.00	40.93
	Post-trained, preferences extracted via:					
	Llama3-8B, pairwise & BT	**42.64** (+0.97)*	**73.00** (+1.67)*	**62.12** (+0.45)*	**53.00** (+7.00)	**41.36** (+0.43)
	Llama3-8B, direct selection	42.46 (+0.79)	72.58 (+1.25)	62.05 (+0.38)	49.00 (+3.00)	41.14 (+0.21)
	Qwen3-32B, direct selection	41.92 (+0.25)	72.75 (+1.42)	62.07 (+0.40)	51.00 (+5.00)	40.49 (-0.44)

It is worth noting that the improvements are not due to using the same LLM both to judge pairwise comparisons and as a backbone model. Indeed, the pairwise judge is always Llama3-8B, and we observe benefits of BT ranking across all backbone LLMs. Further, a larger LLM as a direct judge (Qwen3-32B) barely yields any improvement compared to Llama3-8B.

Pairwise comparisons and BT ranking mitigate positional bias in direct selection. Figure 2 shows the distribution of chosen preference depending on the position they are presented using three preference extraction strategies: (a) PACE (pairwise comparisons and BT) and (b) direct selection with two judges: Llama3-8B and Qwen3-32B. PACE clearly chooses more uniformly from all positions. Further, it chooses generations in different positions depending on the source dataset (ScoNe: first; TIND, CondaQA: third). In contrast, the distributions of direct selection with either judge show a clear bias towards the first choice. It is worth noting that the larger judge (Qwen3-32B) is only slightly less biased, suggesting that scale alone may not address positional bias. Based on this analysis, we argue that the benefits of pairwise comparisons coupled with BT ranking stem in part from mitigating the positional bias of direct LLM selection. See Appendix F.4 for more details.

4.4 Further Analysis

We conduct additional experiments to further understand the effects of PACE. First, we study preference-data scaling to examine how the amount of PACE preference data affects downstream performance. Second, leveraging the negation-type annotations in TIND, we conduct a finer-grained analysis to examine where the improvements in negation understanding arise across different negation types. Finally, we observe that several benchmarks also contain affirmative examples and examine how PACE affects performance on these examples. We summarize the main observations below, with detailed results and further analysis provided in Appendix F.

More PACE preference data generally yields stronger downstream performance. To study the effect of preference-data scale, we randomly sample 1K, 2K, 3.5K, and 4.5K training pairs from the PACE preference dataset. We perform DPO post-training with each subset and evaluate the resulting models on NeQA and ScoNe, representing question answering and natural language inference, respectively. As shown in the scaling curves (Figure 9) in Appendix F.1, performance

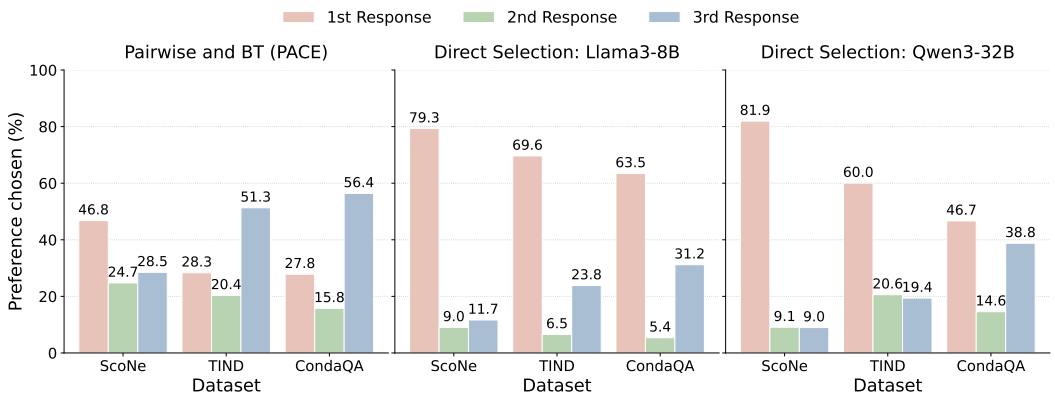

Figure 2: Distribution of chosen preferences depending on the position they are presented. Direct selection with both Llama3-8B and Qwen3-32B (middle and right) is biased toward the first response. In contrast, *pairwise and BT* (PACE), which uses Llama3-8B as a pairwise judge, selects preferences more uniformly across all positions (i.e., mitigates positional bias).

generally improves across both benchmarks as the number of preference pairs increases, with smaller gains from 3.5K to 4.5K, suggesting diminishing returns at larger data scales.

PACE improves both negation types, with larger gains on verbal negation. TIND distinguishes verbal negation, where negation directly modifies a verb, from non-verbal negation. Off-the-shelf models consistently perform worse on non-verbal negation, suggesting that it is the more challenging of the two. Across all models, PACE improves performance on both negation types, with consistently larger gains on verbal negation. This may reflect the more direct relation between the negation marker and predicate in verbal cases, while non-verbal negation involves more varied scope interactions. Detailed results are reported in Table 5 in Appendix F.2.

PACE generally improves performance on affirmative examples as well. Finally, we examine whether post-training with PACE affects performance on affirmative examples. Both CondaQA and NevIR contain examples without negation, allowing us to evaluate these subsets separately. Across the eight model-benchmark settings, PACE improves affirmative-example accuracy in seven, with only a small decrease of 1.50 percentage points in one setting. The gains can also be substantial, reaching up to 7.96 percentage points. Full results are reported in Appendix F.3, Table 7. These results suggest that PACE improves negation understanding without sacrificing performance on affirmative inputs, with its benefits often extending to non-negated examples as well.

5 Conclusion

In this work, we identify a key challenge in synthetic preference construction when models struggle with the target capability, such as negation understanding, which prior work has repeatedly shown to remain challenging for LLMs. We introduce **PACE** to address this challenge through answer-conditioned generation of sound and plausible but unsound justifications, together with pairwise comparisons and BT ranking for preference extraction. Using negation understanding as a challenging testbed, we show that DPO post-training with approximately 5K PACE preference pairs improves Llama and Qwen models of different sizes across multiple tasks and benchmarks, with most improvements statistically significant. Our analyses further reveal two key sources of these gains. First, conditioning generation on both correct and incorrect answers is important: removing incorrect-answer-conditioned justifications or answer conditioning altogether consistently weakens performance and is often detrimental relative to the off-the-shelf models. Second, pairwise comparisons with BT ranking provide more reliable preference extraction than direct LLM judging and substantially mitigate positional bias, which remains present even with a much larger judge such as Qwen3-32B. While our experiments focus on negation, a common yet challenging capability for LLMs, PACE targets a broader class of preference-construction problems where models struggle with the target capability. Exploring PACE in other such settings is a promising direction for future work.

AI USE STATEMENT

In this work, we used generative AI tools to generate justifications used in constructing our preference dataset, as well as to assist with language editing, manuscript organization, and identifying potentially relevant literature. All AI-assisted text was manually reviewed and revised by the authors, and references suggested by generative AI were independently verified through established scholarly sources and original publication venues. The core research ideas and methodological contributions were conceived by the authors, who also independently evaluated all scientific claims and conclusions. We take full responsibility for the final content of this work, including its text, claims, and citations.

REPRODUCIBILITY STATEMENT

We provide the complete PACE algorithm, data construction procedure, prompts, filtering criteria, preference extraction details, and DPO training configuration in the main paper and Appendix. We also report the datasets, hyperparameters, evaluation protocols, and additional experimental results needed to reproduce our experiments. Our implementation and constructed preference data are available in the anonymous repository provided in the paper.

REFERENCES

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Alon Albalak, Yanai Elazar, Sang Michael Xie, Shayne Longpre, Nathan Lambert, Xinyi Wang, Niklas Muennighoff, Bairu Hou, Liangming Pan, Haewon Jeong, et al. A survey on data selection for language models. *arXiv preprint arXiv:2402.16827*, 2024.

Miriam Anschütz, Diego Miguel Lozano, and Georg Groh. This is not correct! negation-aware evaluation of language generation systems. In *Proceedings of the 16th International Natural Language Generation Conference*, pp. 163–175, 2023.

Nicholas Asher and Swarnadeep Bhar. Strong hallucinations from negation and how to fix them. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 12670–12687, 2024.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022a.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073*, 2022b.

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.

Collin Burns, Pavel Izmailov, Jan Hendrik Kirchner, Bowen Baker, Leo Gao, Leopold Aschenbrenner, Yining Chen, Adrien Ecoffet, Manas R. Joglekar, Jan Leike, Ilya Sutskever, Jeff Wu, and OpenAI. Weak-to-strong generalization: Eliciting strong capabilities with weak supervision. In *International Conference on Machine Learning*, 2023. URL https://api.semanticscholar.org/CorpusID:266312608.

Jiangjie Chen, Wei Shi, Ziquan Fu, Sijie Cheng, Lei Li, and Yanghua Xiao. Say what you mean! large language models speak too positively about negative commonsense knowledge. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9890–9908, 2023.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality, March 2023. URL https://lmsys.org/blog/2023-03-30-vicuna/.

Wei-Lin Chiang, Lianmin Zheng, Ying Sheng, Anastasios Nikolas Angelopoulos, Tianle Li, Dacheng Li, Hao Zhang, Banghua Zhu, Michael Jordan, Joseph E Gonzalez, et al. Chatbot arena: An open platform for evaluating llms by human preference. *arXiv preprint arXiv:2403.04132*, 2024.

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.

Ganqu Cui, Lifan Yuan, Ning Ding, Guanming Yao, Bingxiang He, Wei Zhu, Yuan Ni, Guotong Xie, Ruobing Xie, Yankai Lin, et al. Ultrafeedback: Boosting language models with scaled ai feedback. *arXiv preprint arXiv:2310.01377*, 2023.

Xun Deng, Han Zhong, Rui Ai, Fuli Feng, Zheng Wang, and Xiangnan He. Less is more: Improving llm alignment via preference data selection. *arXiv preprint arXiv:2502.14560*, 2025.

Peter Devine. Are you sure? rank them again: Repeated ranking for better preference datasets. In Jonne Sälevä and Abraham Owodunni (eds.), *Proceedings of the Fourth Workshop on Multilingual Representation Learning (MRL 2024)*, pp. 93–105, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.mrl-1.5. URL https://aclanthology.org/2024.mrl-1.5/.

Kawin Ethayarajh, Winnie Xu, Niklas Muennighoff, Dan Jurafsky, and Douwe Kiela. Model alignment as prospect theoretic optimization. In *Forty-first International Conference on Machine Learning*, 2024.

Iker García-Ferrero, Begoña Altuna, Javier Alvez, Itziar Gonzalez-Dios, and German Rigau. This is not a dataset: A large negation benchmark to challenge large language models. In *Proceedings of the 2023 conference on empirical methods in natural language processing*, pp. 8596–8615, 2023.

Atticus Geiger, Kyle Richardson, and Christopher Potts. Neural natural language inference models partially embed theories of lexical entailment and negation. In Afra Alishahi, Yonatan Belinkov, Grzegorz Chrupała, Dieuwke Hupkes, Yuval Pinter, and Hassan Sajjad (eds.), *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pp. 163–173, Online, November 2020a. Association for Computational Linguistics. doi: 10.18653/v1/2020.blackboxnlp-1.16. URL https://aclanthology.org/2020.blackboxnlp-1.16/.

Atticus Geiger, Kyle Richardson, and Christopher Potts. Neural natural language inference models partially embed theories of lexical entailment and negation. In *Proceedings of the third blackboxnlp workshop on analyzing and interpreting neural networks for NLP*, pp. 163–173, 2020b.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Jiawei Gu, Xuhui Jiang, Zhichao Shi, Hexiang Tan, Xuehao Zhai, Chengjin Xu, Wei Li, Yinghan Shen, Shengjie Ma, Honghao Liu, et al. A survey on llm-as-a-judge. *The Innovation*, 2024.

Jiwoo Hong, Noah Lee, and James Thorne. Orpo: Monolithic preference optimization without reference model. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 11170–11189, 2024.

Laurence R. Horn and Heinrich Wansing. Negation. In Edward N. Zalta and Uri Nodelman (eds.), *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, Spring 2025 edition, 2025.

Md Mosharaf Hossain and Eduardo Blanco. Leveraging affirmative interpretations from negation improves natural language understanding. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 5833–5847, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. URL https://aclanthology.org/2022.emnlp-main.393.

Md Mosharaf Hossain, Antonios Anastasopoulos, Eduardo Blanco, and Alexis Palmer. It's not a non-issue: Negation as a source of error in machine translation. In Trevor Cohn, Yulan He, and Yang Liu (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2020*, pp. 3869–3885, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.findings-emnlp.345. URL https://aclanthology.org/2020.findings-emnlp.345/.

Md Mosharaf Hossain, Dhivya Chinnappa, and Eduardo Blanco. An analysis of negation in natural language understanding corpora. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio (eds.), *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 716–723, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-short.81. URL https://aclanthology.org/2022.acl-short.81/.

Arian Hosseini, Siva Reddy, Dzmitry Bahdanau, R Devon Hjelm, Alessandro Sordoni, and Aaron Courville. Understanding by understanding not: Modeling negation in language models. In Kristina Toutanova, Anna Rumshisky, Luke Zettlemoyer, Dilek Hakkani-Tur, Iz Beltagy, Steven Bethard, Ryan Cotterell, Tanmoy Chakraborty, and Yichao Zhou (eds.), *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 1301–1312, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.102. URL https://aclanthology.org/2021.naacl-main.102/.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Liang Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *Iclr*, 1(2):3, 2022.

Jiaming Ji, Mickel Liu, Josef Dai, Xuehai Pan, Chi Zhang, Ce Bian, Boyuan Chen, Ruiyang Sun, Yizhou Wang, and Yaodong Yang. Beavertails: Towards improved safety alignment of llm via a human-preference dataset. *Advances in Neural Information Processing Systems*, 36:24678–24704, 2023.

Jiaming Ji, Xinyu Chen, Rui Pan, Conghui Zhang, Han Zhu, Jiahao Li, Donghai Hong, Boyuan Chen, Jiayi Zhou, Kaile Wang, et al. Safe rlhf-v: Safe reinforcement learning from multi-modal human feedback. *arXiv preprint arXiv:2503.17682*, 2025.

Aditya Khandelwal and Suraj Sawant. Negbert: a transfer learning approach for negation detection and scope resolution. In *Proceedings of the twelfth language resources and evaluation conference*, pp. 5739–5748, 2020.

Jinsung Kim, Seonmin Koo, and Heui-Seok Lim. Semantic inversion, identical replies: Revisiting negation blindness in large language models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 21445–21482, 2025.

Ryan Koo, Minhwa Lee, Vipul Raheja, Jong Inn Park, Zae Myung Kim, and Dongyeop Kang. Benchmarking cognitive biases in large language models as evaluators. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 517–545, 2024.

Dawei Li, Bohan Jiang, Liangjie Huang, Alimohammad Beigi, Chengshuai Zhao, Zhen Tan, Amrita Bhattacharjee, Yuxuan Jiang, Canyu Chen, Tianhao Wu, Kai Shu, Lu Cheng, and Huan Liu. From generation to judgment: Opportunities and challenges of LLM-as-a-judge. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 2757–2791, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.138. URL https://aclanthology.org/2025.emnlp-main.138/.

Hao Li and Wei Lu. Learning with structured representations for negation scope extraction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 533–539, 2018.

Chen Lin, Steven Bethard, Dmitriy Dligach, Farig Sadeque, Guergana Savova, and Timothy A Miller. Does bert need domain adaptation for clinical negation detection? *Journal of the American Medical Informatics Association*, 27(4):584–591, 2020.

Alexander H Liu, Kartik Khandelwal, Sandeep Subramanian, Victor Jouault, Abhinav Rastogi, Adrien Sadé, Alan Jeffares, Albert Jiang, Alexandre Cahill, Alexandre Gavaudan, et al. Ministral 3. *arXiv preprint arXiv:2601.08584*, 2026.

Ahmed Mahany, Heba Khaled, Nouh Sabri Elmitwally, Naif Aljohani, and Said Ghoniemy. Negation and speculation in nlp: A survey, corpora, methods, and applications. *Applied Sciences*, 12(10): 5209, 2022.

Ian R. McKenzie, Alexander Lyzhov, Michael Pieler, Alicia Parrish, Aaron Mueller, Ameya Prabhu, Euan McLean, Aaron Kirtland, Alexis Ross, Alisa Liu, Andrew Gritsevskiy, Daniel Wurgaft, Derik Kauffman, Gabriel Recchia, Jiacheng Liu, Joe Cavanagh, Max Weiss, Sicong Huang, The Floating Droid, Tom Tseng, Tomasz Korbak, Xudong Shen, Yuhui Zhang, Zhengping Zhou, Najoung Kim, Samuel R. Bowman, and Ethan Perez. Inverse scaling: When bigger isn't better, 2024. URL https://arxiv.org/abs/2306.09479.

Quinn McNemar. Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, 12(2):153–157, 1947.

Yu Meng, Mengzhou Xia, and Danqi Chen. Simpo: Simple preference optimization with a reference-free reward. *Advances in Neural Information Processing Systems*, 37:124198–124235, 2024.

Roser Morante and Walter Daelemans. A metalearning approach to processing the scope of negation. In *Proceedings of the thirteenth conference on computational natural language learning (CoNLL-2009)*, pp. 21–29, 2009.

Roser Morante and Walter Daelemans. ConanDoyle-neg: Annotation of negation cues and their scope in conan doyle stories. In Nicoletta Calzolari, Khalid Choukri, Thierry Declerck, Mehmet Uğur Doğan, Bente Maegaard, Joseph Mariani, Asuncion Moreno, Jan Odijk, and Stelios Piperidis (eds.), *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pp. 1563–1568, Istanbul, Turkey, May 2012. European Language Resources Association (ELRA). URL https://aclanthology.org/L12-1077/.

Tetsuro Morimura, Mitsuki Sakamoto, Yuu Jinnai, Kenshi Abe, and Kaito Ariu. Filtered direct preference optimization. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 22729–22770, 2024.

Ha-Thanh Nguyen and Ken Satoh. Balancing exploration and exploitation in llm using soft rllf for enhanced negation understanding. *arXiv preprint arXiv:2403.01185*, 2024.

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.

Geoffrey K Pullum and Rodney Huddleston. Negation, chapter 9, 2002.

Zhong Qian, Peifeng Li, Qiaoming Zhu, Guodong Zhou, Zhunchen Luo, and Wei Luo. Speculation and negation scope detection via convolutional neural networks. In *Proceedings of the 2016 conference on empirical methods in natural language processing*, pp. 815–825, 2016.

Zhen Qin, Rolf Jagerman, Kai Hui, Honglei Zhuang, Junru Wu, Le Yan, Jiaming Shen, Tianqi Liu, Jialu Liu, Donald Metzler, Xuanhui Wang, and Michael Bendersky. Large language models are effective text rankers with pairwise ranking prompting. In Kevin Duh, Helena Gomez, and Steven Bethard (eds.), *Findings of the Association for Computational Linguistics: NAACL 2024*, pp. 1504–1518, Mexico City, Mexico, June 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-naacl.97. URL https://aclanthology.org/2024.findings-naacl.97/.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741, 2023.

Abhilasha Ravichander, Matt Gardner, and Ana Marasović. Condaqa: A contrastive reading comprehension dataset for reasoning about negation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 8729–8755, 2022.

MohammadHossein Rezaei and Eduardo Blanco. Making language models robust against negation. In Luis Chiruzzo, Alan Ritter, and Lu Wang (eds.), *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 8123–8142, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. ISBN 979-8-89176-189-6. doi: 10.18653/v1/2025.naacl-long.413. URL https://aclanthology.org/2025.naacl-long.413/.

Corby Rosset, Ching-An Cheng, Arindam Mitra, Michael Santacroce, Ahmed Awadallah, and Tengyang Xie. Direct nash optimization: Teaching language models to self-improve with general preferences. *ArXiv*, abs/2404.03715, 2024. URL https://api.semanticscholar.org/CorpusID:268987488.

Jérémy Scheurer, Jon Ander Campos, Tomasz Korbak, Jun Shern Chan, Angelica Chen, Kyunghyun Cho, and Ethan Perez. Training language models with language feedback at scale. *arXiv preprint arXiv:2303.16755*, 2023.

Jaehyung Seo, Hyeonseok Moon, and Heui-Seok Lim. The impact of negated text on hallucination with large language models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 13565–13583, 2025.

Jingyuan S She, Christopher Potts, Samuel Bowman, and Atticus Geiger. Scone: Benchmarking negation reasoning in language models with fine-tuning and in-context learning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 1803–1821, 2023.

Lin Shi, Chiyu Ma, Wenhua Liang, Xingjian Diao, Weicheng Ma, and Soroush Vosoughi. Judging the judges: A systematic study of position bias in llm-as-a-judge. In *Proceedings of the 14th International Joint Conference on Natural Language Processing and the 4th Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics*, pp. 292–314, 2025.

Rituraj Singh, Rahul Kumar, and Vivek Sridhar. Nlms: Augmenting negation in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 13104–13116, 2023.

Tian Tan and Eduardo Blanco. Not all negation cues are equal: Affixal negations yield better negation understanding. *arXiv preprint arXiv:2609.13685*, 2026.

Guiyao Tie, Zeli Zhao, Dingjie Song, Fuyang Wei, Rong Zhou, Yurou Dai, Wen Yin, Zhejian Yang, Jiangyue Yan, Yao Su, et al. A survey on post-training of large language models. *arXiv preprint arXiv:2503.06072*, 2025.

Thinh Truong, Timothy Baldwin, Trevor Cohn, and Karin Verspoor. Improving negation detection with negation-focused pre-training. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 4188–4193, 2022a.

Thinh Hung Truong, Julia Otmakhova, Timothy Baldwin, Trevor Cohn, Jey Han Lau, and Karin Verspoor. Not another negation benchmark: The nan-nli test suite for sub-clausal negation. In *Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 883–894, 2022b.

Neeraj Varshney, Satyam Raj, Venkatesh Mishra, Agneet Chatterjee, Amir Saeidi, Ritika Sarkar, and Chitta Baral. Investigating and addressing hallucinations of llms in tasks involving negation. In *Proceedings of the 5th workshop on trustworthy NLP (TrustNLP 2025)*, pp. 580–598, 2025.

Peiyi Wang, Lei Li, Liang Chen, Zefan Cai, Dawei Zhu, Binghuai Lin, Yunbo Cao, Lingpeng Kong, Qi Liu, Tianyu Liu, and Zhifang Sui. Large language models are not fair evaluators. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9440–9450, Bangkok,

Thailand, August 2024a. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.511. URL https://aclanthology.org/2024.acl-long.511/.

Peiyi Wang, Lei Li, Liang Chen, Zefan Cai, Dawei Zhu, Binghuai Lin, Yunbo Cao, Lingpeng Kong, Qi Liu, Tianyu Liu, et al. Large language models are not fair evaluators. In *Proceedings of the 62nd annual meeting of the association for computational linguistics (volume 1: Long papers)*, pp. 9440–9450, 2024b.

Yifan Wang, Runjin Chen, Bolian Li, David Cho, Yihe Deng, Ruqi Zhang, Tianlong Chen, Zhangyang Wang, Ananth Grama, and Junyuan Hong. More is less: The pitfalls of multi-model synthetic preference data in dpo safety alignment. *arXiv preprint arXiv:2504.02193*, 2025.

Jason Wei, Najoung Kim, Yi Tay, and Quoc Le. Inverse scaling can become U-shaped. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 15580–15591, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.963. URL https://aclanthology.org/2023.emnlp-main.963/.

Orion Weller, Dawn Lawrie, and Benjamin Van Durme. Nevir: Negation in neural information retrieval. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2274–2287, 2024.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zhihao Fan. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*, 2024.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Yuhui Zhang, Michihiro Yasunaga, Zhengping Zhou, Jeff Z HaoChen, James Zou, Percy Liang, and Serena Yeung. Beyond positive scaling: How negation impacts scaling trends of language models. In *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 7479–7498, 2023.

Zizhuo Zhang, Qizhou Wang, Shanshan Ye, Jianing Zhu, Jiangchao Yao, Bo Han, and Masashi Sugiyama. Towards understanding valuable preference data for large language model alignment. In *International Conference on Learning Representations*, volume 2026, pp. 24817–24844, 2026.

Yiyun Zhao and Steven Bethard. How does bert's attention change when you fine-tune? an analysis methodology and a case study in negation scope. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 4729–4747, 2020.

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.

Banghua Zhu, Evan Frick, Tianhao Wu, Hanlin Zhu, Karthik Ganesan, Wei-Lin Chiang, Jian Zhang, and Jiantao Jiao. Starling-7b: Improving helpfulness and harmlessness with rlaif. In *First Conference on Language Modeling*, 2024.

Xinyu Zhu, Mengzhou Xia, Zhepei Wei, Wei-Lin Chen, Danqi Chen, and Yu Meng. The surprising effectiveness of negative reinforcement in llm reasoning. *arXiv preprint arXiv:2506.01347*, 2025.

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 Future Work

Negation is a complex linguistic phenomenon involving diverse reasoning challenges. Our current framework constructs unsound reasoning in a relatively coarse-grained manner, primarily by conditioning generation on flipped answer labels. While effective, this approach does not explicitly distinguish between different types of negation reasoning errors, such as misinterpreting negation scope, misidentifying the negated target, or making faulty commonsense inferences. Our finer-grained analysis across different negation types also suggests that some forms of negation remain more challenging than others. Future work could therefore construct more fine-grained unsound justifications that explicitly capture these error types, potentially enabling preference post-training to target specific weaknesses in negation understanding.

In addition, PACE currently uses a single judge model for pairwise preference extraction. Future work could investigate multiple judges and cross-model agreement to further improve the robustness of preference extraction and mitigate positional bias. Our current evaluation uses zero-shot prompting. Future work could further evaluate PACE under few-shot settings to examine whether its benefits remain consistent when additional in-context examples are provided.

Although PACE demonstrates strong effectiveness on negation understanding, its current formulation assumes access to a gold answer and a well-defined contrasting incorrect answer, making it most directly applicable to binary-answer tasks. Future work could extend PACE to tasks with more complex answer spaces, such as multiple-choice and open-ended question answering, and evaluate it on other challenging capabilities.

B Broader Impacts

Our work studies reliable synthetic preference construction when LLMs struggle with the capability being optimized, using negation understanding as a challenging testbed. By conditioning generation on known correct and incorrect answers and improving preference extraction through pairwise comparisons and Bradley–Terry ranking, PACE provides a way to construct useful preference data without requiring the generator to first solve the target problem reliably.

Improved negation understanding may benefit downstream applications that require precise reasoning over negation, such as question answering, information retrieval, and decision-support systems, where overlooking or misinterpreting negation can change the intended meaning and lead to incorrect outputs. More broadly, the preference-construction principles explored in PACE may also be useful for other capabilities that LLMs struggle with, although such applications require further evaluation.

At the same time, negation is linguistically diverse, and our current approach does not capture all forms of negation or reasoning errors. Models post-trained with PACE may therefore still produce incorrect outputs. Particular caution is warranted in high-stakes settings such as healthcare or legal applications, where errors in interpreting negation may have serious consequences.

C Additional Related Work

Negation. Beyond the representative studies discussed in the main text, prior work on negation has extensively investigated cue detection, scope resolution, and related semantic phenomena (Qian et al., 2016; Li & Lu, 2018; Lin et al., 2020; Truong et al., 2022a). Earlier computational studies have similarly examined negation detection, scope, and negation-aware language representations (Morante & Daelemans, 2009; 2012; Khandelwal & Sawant, 2020; Zhao & Bethard, 2020). Additional analyses have examined the impact of negation on neural language models and downstream prediction (Hossain et al., 2020; Mahany et al., 2022). Prior work has sought to improve negation understanding through affirmative interpretations and paraphrasing (Hossain & Blanco, 2022), task-specific fine-tuning and prompting (She et al., 2023; Zhang et al., 2023; Anschütz et al., 2023), and additional pre-training (Hosseini et al., 2021; Singh et al., 2023; Tan & Blanco, 2026). Some of these approaches introduce additional inference-time processing by generating or incorporating affirmative interpretations, while others require additional task-specific training or pre-training. In contrast, PACE performs preference-based post-training offline and requires no additional inference-time generation or input augmentation once the model is trained. More recently, reinforcement learning from logical

feedback has also been investigated as a way to improve negation understanding (Nguyen & Satoh, 2024). These studies complement the directions discussed in the main text, but do not construct pairwise preference data for negation post-training.

Preference learning. Beyond DPO, subsequent work has proposed alternative preference-optimization objectives. KTO formulates preference optimization through a prospect-theoretic objective, while ORPO and SimPO introduce alternative reference-free formulations (Ethayarajh et al., 2024; Hong et al., 2024; Meng et al., 2024). These methods primarily investigate how preference signals should be optimized once available, whereas our work focuses on the upstream construction of the preference data itself.

Synthetic preference data construction. A growing body of work studies the construction and quality of preference supervision for post-training (Albalak et al., 2024; Deng et al., 2025; Zhang et al., 2026). Related efforts have explored AI feedback and preference datasets for large-scale alignment (Bai et al., 2022b; Ji et al., 2023). For example, Cui et al. (2023) introduce UltraFeedback, which generates diverse candidate responses and uses an LLM evaluator to provide large-scale feedback for preference learning. Other studies investigate filtering, selection, and composition of preference pairs and demonstrate their importance for downstream preference optimization (Morimura et al., 2024; Wang et al., 2025; Zhang et al., 2026). These approaches primarily improve preference supervision after candidate responses are available. PACE complements this line of work by addressing candidate generation itself when the generator is unreliable on the target capability.

Preference extraction. Beyond the LLM-as-a-judge approaches discussed in the main text, prior work has explored stronger proprietary models as judges, repeated evaluation, and prompt-engineering strategies to improve automatic evaluation (Achiam et al., 2023; Zhu et al., 2024; Qin et al., 2024; Devine, 2024). Additional work has studied LLM-based evaluation and preference generation more broadly (Li et al., 2025). Related preference-learning strategies also use outputs from models of different strengths or human supervision to construct preferred and rejected responses (Burns et al., 2023; Rosset et al., 2024). These directions provide alternative mechanisms for obtaining preference signals, whereas PACE focuses on extracting reliable preferences from answer-conditioned candidate justifications using pairwise evaluation and Bradley–Terry aggregation.

D Preference Dataset Construction and Human Evaluation

D.1 Response Generation Details

Initial Seed Dataset. To construct our preference dataset, we use the training splits of three negation-related datasets: CondaQA, ScoNe, and TIND. These datasets cover complementary task settings, including question answering, natural language inference with a focus on negation scope, and commonsense reasoning. CondaQA is a question-answering dataset in which answering questions over passages requires understanding negation. ScoNe is a natural language inference dataset in which determining entailment requires understanding the effect and scope of negation in the premises and hypotheses. TIND is a question-answering dataset that evaluates commonsense reasoning. Its questions are short and polar, requiring `YES` or `NO` answers, and cannot be answered solely from the information explicitly provided (see Figure 1 for an example).

To build a balanced *initial seed dataset* serving as the starting point of our pipeline, we sample approximately equal numbers of instances from the training split of each source dataset. All three datasets contain both negated and affirmative examples. For CondaQA, we retain all questions with `YES` or `NO` answers, which account for 94.33% of the data, and exclude the remaining 5.67% whose answers are `Don't know` or short phrases. For ScoNe, a binary classification dataset with `entailment` and `neutral` labels organized into six subsets corresponding to different types of negation, we perform balanced random sampling within each subset. For TIND, the largest negation-focused dataset with `true`/`false` labels, we randomly sample from the training set. We further normalize all answer labels into a unified `YES`/`NO` format to ensure consistency across datasets. Statistics of the resulting initial seed dataset are reported in Table 3.

Prompts for Response Generation. Based on the seed dataset, we construct prompts for LLM-based response generation under different prompting strategies. Specifically, we design a set of standardized prompt templates, including: (1) a standard setting without providing the answer as

You are given a reasoning question. Your task is to answer the question and provide a short justification.

Input:
QUESTION: {question}

Output EXACTLY two lines:
Answer: Yes or No
Justification: Provide one short justification (at most 60 words) based on the input. If the input contains a negation cue, explicitly explain its scope and how it affects the answer. If there is no negation cue, do not invent one.

Figure 3: Prompt template for the standard prompting strategy.

You are given an input with a proposed answer. Your task is to justify the proposed answer.

Input:
The proposed answer to the question below is: {answer}

QUESTION: {question}

Output EXACTLY two lines:
Answer: {answer}
Justification: Provide one short justification (at most 60 words) based on the input. If the input contains a negation cue, explicitly explain its scope and how it affects the answer. If there is no negation cue, do not invent one.

Figure 4: Prompt template for the answer-given prompting strategy.

shown in Figure 3, and (2) an answer-conditioned prompting template, as illustrated in Figure 4. By inserting the correct answer for each question instance, we construct prompts used for generating responses conditioned on the correct answer. Conversely, by inserting a constructed incorrect answer, we obtain prompts used for generating responses conditioned on incorrect answers.

In the **Input** section, we provide the question instance used for response generation. It is worth noting that, unlike TIND and ScoNe, samples in CondaQA consist of both a question and an associated passage. Therefore, prompts constructed for CondaQA additionally include the corresponding passage in the input section. Figure 5 shows a real example prompt constructed from a TIND sample under the answer-conditioned prompting strategy for response generation.

D.2 Filtering Details

In this section, we provide a more detailed description of our filtering process. Using the prompt templates and initial seed dataset described above, we generate candidate responses with short justifications from LLMs. However, To ensure the quality of the candidate responses, we apply several post-processing checks and filtering steps to the generated responses. These mainly include rule-based filtering and a consistency check. Filtering statistics are reported in Table 3.

Rule-Based Filtering. Our rule-based filtering mainly focuses on parsing and extracting answers and justifications from the raw responses generated by LLMs. Overall, the filtering process consists of three stages: (1) raw response-level filtering, (2) consistency checking between valid responses and the given answer, and (3) model-group-level filtering.

Using the answer-conditioned prompting strategy as an example, we first clean the raw response text by removing unnecessary artifacts such as redundant newline characters and residual conversational tokens (e.g., “assistant”). In practice, we observe that some models tend to partially repeat or echo the input prompt before producing their final prediction. We therefore parse the answer by scanning the response text backward from the end. Similarly, justifications are extracted by searching backward for the keyword “Justification”. Responses from which both the answer and justification can be successfully extracted are regarded as valid responses.

You are given an input with a proposed answer. Your task is to justify the proposed answer.

Input:
The proposed answer to the question below is: NO.

QUESTION: Is the following statement true: In certain contexts a flat is classified as a place?

Output EXACTLY two lines:
Answer: NO

Justification: Provide one short justification (at most 60 words) based on the input. If the input contains a negation cue, explicitly explain its scope and how it affects the answer. If there is no negation cue, do not invent one.

Figure 5: An example answer-conditioned prompt on the TIND dataset.

Table 3: Statistics of seed dataset construction and filtering stages across the three source datasets.

	CondaQA	ScoNe	TIND
Source Training Set	5,382	6,012	268,505
Initial Seed Dataset	5,077	5,076	5,076
# YES	2,485	2,540	2,525
# NO	2,592	2,536	2,551
After Filtering			
Rule-Based	4,535	3,273	3,108
Consistency	2,078	1,458	1,496

Next, we verify whether the extracted answer is consistent with the given answer in the prompt. Due to the difficulty of negation reasoning, models may still generate incorrect answers and corresponding justifications even when provided with the correct answer. Conversely, under incorrect-answer conditioning, some models reject the provided answer, argue against it, and output the correct answer instead. Since our framework requires candidate responses conditioned on the provided answer, we discard responses whose extracted answer is inconsistent with the given answer.

Finally, each question instance is expected to receive one valid response from each of the three response-generation LLMs. Samples with fewer than three valid candidate responses are discarded.

Consistency Check. The consistency check can be viewed as a post-processing step after response generation. Its goal is to remove samples that are highly sensitive to positional bias in LLM-as-a-judge settings. That is, the judge's preference should remain unchanged regardless of the order in which the two responses are presented.

After rule-based filtering, each question instance contains two candidate response sets, denoted as R^+ and R^-, corresponding to responses generated under correct-answer and incorrect-answer conditioning, respectively. For each response set, pairwise comparisons are performed within the same set. Using R^+ as an example, we construct all response pairs:

$$(r_a, r_b) \in R^+ \times R^+, \quad r_a \neq r_b.$$

The same procedure is applied to R^-. We then swap the response order to obtain (r_b, r_a) and present both pairs to the judge model for comparison. If the judge produces inconsistent preferences for the two orders, the response pair is regarded as unstable and cannot be consistently ranked.

At the sample level, if the majority of candidate pairs within either R^+ or R^- cannot be consistently ranked, we discard the entire question instance. Intuitively, samples that survive this consistency check are expected to be less affected by positional bias.

After this stage, we obtain the final seed dataset of question instances, which is used throughout all subsequent experiments, including analyses of response-generation and preference-extraction strategies. The resulting dataset contains 5,032 samples, with statistics reported in Table 3.

Importantly, the consistency check does not filter examples based on correctness or task difficulty, but only on whether the judge's preference remains consistent across response orders. Thus, a difficult

example is retained as long as the judge makes the same choice in both orders, even if that choice is incorrect. In contrast, order-sensitive examples, which may include semantically ambiguous or borderline cases, are removed. Our human evaluation in Section D.4 further shows that annotators do not reach perfect agreement on all sampled examples, suggesting that the filtering does not simply reduce the dataset to trivial cases.

D.3 Preference Extraction Details

Preparing Responses for Preference Extraction. As discussed above, after filtering, we obtain a seed dataset of question instances. Using different prompting strategies, we prompt LLMs to generate responses to these questions. After extracting valid answers and justifications, we further apply a unified formatting step across all experiments. Specifically, we concatenate the extracted justification and answer in the format: {*justification*} *Therefore, the answer is* {*answer*}.

The motivation behind this design is to encourage the judge model to focus more on the justification, which contains the underlying reasoning process, rather than relying primarily on the final answer itself. After this normalization step, the resulting responses are used as candidate responses for preference extraction by the judge model.

Prompts for Preference Extraction. Under our correct- and incorrect-answer-conditioned prompting strategy, we explore two preference extraction approaches: Direct Selection and Pairwise Comparison with Bradley–Terry (BT) aggregation. Figures 6 and 7 illustrate the corresponding prompt templates used for judge-based preference evaluation.

For direct selection strategy, all candidate responses are presented to the judge model simultaneously, which directly selects the top-ranked response. In contrast, the pairwise strategy compares candidate responses in pairs and uses the resulting pairwise preferences to fit a global ranking through the BT model, from which the top-ranked response is selected.

PACE Preference Dataset. Together with their corresponding question instances, the selected chosen and rejected responses form training triplets (q, r_c, r_r), which constitute our final preference dataset for subsequent DPO training. In total, the dataset contains 5,032 question instances obtained from the training splits of the three source datasets after filtering. We then merge samples from all source datasets, remove duplicated instances, and randomly split the preference dataset into training and validation sets with a 9:1 ratio, where 10% of the samples are used for validation.

D.4 Human Evaluation

To assess the quality of the preference dataset constructed through our **PACE** framework, we conducted a human evaluation. We randomly sampled 1% of the constructed preference pairs and asked two annotators to independently evaluate each sample. We examined the following four aspects to evaluate the quality of the generated responses under our framework: We examined the following four aspects to evaluate the quality of the generated responses under our framework:

- whether the question contains negation,
- whether human annotators agree with the LLM-selected preferred response,
- whether the generated justification exhibits the expected reasoning quality: for correct-answer-conditioned responses, whether the reasoning is logically sound; and for incorrect-answer-conditioned responses, whether it reflects plausible but flawed reasoning rather than hallucinated premises or arbitrary rationalization, and
- whether the justification consistently supports the given answer rather than contradicting it.

Our analysis shows that approximately 52% of the evaluated samples contain negation, indicating that our dataset is nearly balanced between affirmative and negation instances. The average agreement between the two human annotators and the LLM preference selection reached 97%, while the two annotators selected the same preferred response for 98% of the evaluated samples.

In addition, all evaluated justifications were relevant to the input questions, suggesting that the LLMs rarely generated completely unrelated content. Furthermore, 94% of the justifications correctly supported their associated answers. As discussed earlier, negation reasoning remains challenging for LLMs, and models may still generate incorrect reasoning paths that support wrong answers even

You are comparing three candidate responses to the same question.
The given answer for this question is: {given_answer}.
All three responses attempt to justify this answer.

Choose the response whose justification more clearly and logically supports the given answer, and stays relevant to the question or passage.

Input:
{question}

Response A:
{response_a}

Response B:
{response_b}

Response C:
{response_c}

Output exactly one line without explanation:
Winner: A
or
Winner: B
or
Winner: C

Figure 6: Prompt template used for direct-selection preference evaluation by the judge model.

You are comparing two candidate responses to the same question.
The given answer for this question is: {given_answer}.
Both responses attempt to justify this answer.

Choose the response whose justification more clearly and logically supports the given answer, and stays relevant to the question or passage.

Input:
{question}

Response A:
{response_a}

Response B:
{response_b}

Output exactly one line:
Winner: A
or
Winner: B

Figure 7: Prompt template used for pairwise preference comparison by the judge model.

when conditioned on a given answer. Conversely, when conditioned on an incorrect answer, models may instead produce the correct answer and justification, thereby contradicting the provided answer.

Overall, these results provide strong evidence that the preference dataset constructed by **PACE** is reliable and of high quality.

D.5 Example of a Preference Data Instance

Figure 8 presents a real example from the constructed preference dataset. In this example, `Response 1` is generated conditioned on the correct answer and therefore demonstrates a sound reasoning process. It correctly interprets the semantics of the statement and recognizes that the term "presentation" does not inherently denote an event that repeats. As a result, the justification is logically consistent with both the input question and the final answer.

Answer the question. If a passage is provided, base your answer on the passage.

Question: Is the following statement true: Presentation does not stand for an event that repeats?

Response 1:
The statement "Presentation does not stand for an event that repeats" is true because "presentation" typically refers to a one-time event, such as a speech, lecture, or exhibition, where the same content is delivered or displayed only once. Therefore, the answer is YES.

Response 2:
The statement claims "presentation does not stand for an event that repeats," but the term "presentation" can indeed refer to repeated events, e.g., presentations in conferences, workshops, or lectures. The negation is misleadingly absolute. Therefore, the answer is NO.

Figure 8: Example preference pair from the preference dataset created by PACE.

Table 4: Detailed hyperparameter configuration for DPO fine-tuning.

Parameter	β	r	α	Dropout	Warmup ratio	Epochs	Batch size	Learning rate	Precision
Value	0.1	16	32	0.05	0.05	2	8	1×10^{-5}	bfloat16

In contrast, `Response 2` is generated conditioned on an intentionally incorrect answer, forcing the model to construct a justification that supports an incorrect conclusion. Importantly, the resulting reasoning is not a completely unrelated hallucination or nonsensical fabrication. Instead, it remains highly relevant to the input question and attempts to provide a plausible explanation by referring to repeated conference presentations, workshops, or lectures. However, the response contains a semantic-level reasoning error. The original statement concerns whether the word "presentation" semantically stands for a repeating event, whereas `Response 2` incorrectly shifts the discussion toward whether presentations can occur multiple times in practice.

In other words, it mistakenly equates "presentation can be repeated" with "presentation stands for a repeating event." Consequently, although the reasoning appears coherent and plausible on the surface, its underlying logic is flawed and ultimately leads to an incorrect conclusion.

Therefore, such unsound reasoning paths can still provide valuable learning signals for post-training. By exposing models to plausible but incorrect reasoning that remains grounded in the input question, the model can learn to recognize and avoid subtle reasoning errors. Unlike completely irrelevant or nonsensical hallucinated responses, these unsound reasoning paths constitute meaningful negative examples for PACE.

E Experimental Details

DPO training. We conduct DPO training on four backbone models: Llama3.2-3B, Llama3-8B (Grattafiori et al., 2024), Qwen2.5-3B, and Qwen2.5-7B (Yang et al., 2024), using their instruction-tuned variants. Following Rafailov et al. (2023), we use a fixed value of $\beta = 0.1$ for all models. To reduce computational cost, we adopt Low-Rank Adaptation (LoRA) (Hu et al., 2022) for parameter-efficient fine-tuning. We use the AdamW optimizer with a learning rate of 1×10^{-5}, a warmup ratio of 0.05, and a cosine learning rate scheduler. Each model is trained for up to two epochs.

During training, we evaluate the model on the validation set every 100 steps. We enable early stopping with a patience of 3, terminating training if the validation metric does not improve for three consecutive evaluation rounds. The checkpoint with the highest validation reward accuracy is selected as the final model. This schedule allows early stopping to operate independently of the epoch count, since multiple evaluations can occur within a single epoch.

Each model is trained on approximately 4.5K preference pairs, with an additional 0.5K preference pairs used as the validation set. We use the same training and validation splits across experiments to ensure comparability between different variants. We adopt a unified set of hyperparameters across

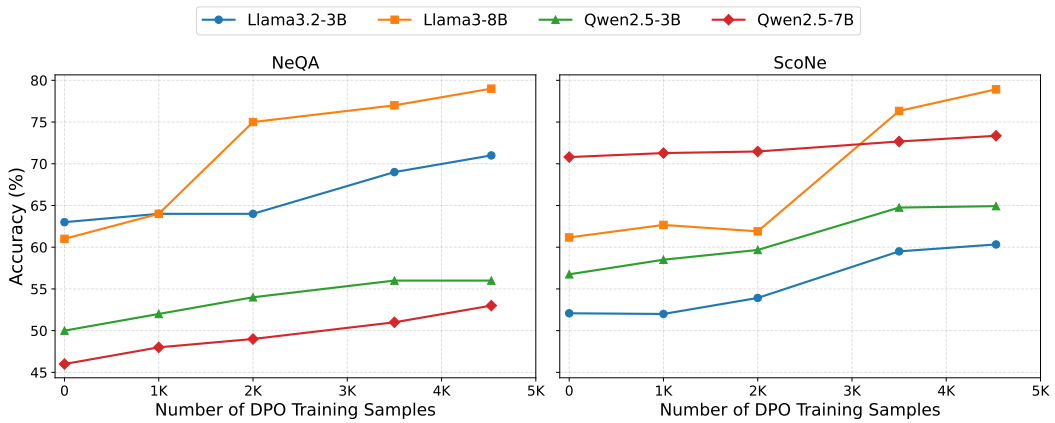

Figure 9: Performance on NeQA and ScoNe with increasing numbers of PACE preference pairs used for DPO post-training. Zero samples denote the corresponding off-the-shelf models. Performance generally improves as more preference data are used, with smaller gains at larger data scales.

all backbone models without extensive model-specific tuning, and largely use commonly adopted default settings. Table 4 reports the complete hyperparameter configuration used for DPO training.

All models are trained on a single NVIDIA H100 GPU with 80GB of memory. Under our training setup, DPO training on the proposed preference dataset requires approximately 1–3 hours depending on the backbone model. Despite this relatively simple and unified training setup, all models consistently achieve improved performance across downstream tasks after DPO training on our preference dataset, demonstrating the effectiveness of the constructed preference data.

Evaluation. During evaluation, we use deterministic decoding (`do_sample=False` with no temperature or top-p sampling), yielding identical outputs across repeated runs. In addition, all experiments are conducted in a zero-shot setting using bf16 precision. Due to the relatively large size of the CondaQA and TIND test sets, we use a batch size of 64 during evaluation, which requires approximately 3 hours to complete. For the remaining datasets, batch evaluation is not used.

F Additional Results and Analysis

In this section, we present additional experimental results and analyses that complement the main paper. Specifically, we (1) examine how performance scales with the amount of PACE preference data, (2) provide a fine-grained analysis of verbal and non-verbal negation on TIND, (3) analyze performance on affirmative examples, and (4) further examine positional bias by separating correct- and incorrect-answer-conditioned responses. These results provide additional insights into the behavior and robustness of PACE.

F.1 Preference-data scaling.

To further examine the effect of preference-data scale, we conduct a scaling experiment using the preference dataset constructed by PACE. From the 4.5K training pairs, we randomly sample subsets of 1K, 2K, 3.5K, and 4.5K pairs for DPO post-training, while keeping the same 0.5K validation set fixed across all settings. We conduct the scaling experiment on all four backbone models. To ensure comparability across different training-set sizes, all models are trained for the same number of epochs, with evaluation performed at the end of each epoch. All other training hyperparameters, including the learning rate and DPO configuration, are kept identical to those used for training on the full PACE preference dataset. We evaluate each resulting model on NeQA and ScoNe, representing question answering and natural language inference, respectively. The off-the-shelf models are included as the 0-sample baselines.

As shown in Figure 9, performance generally improves as more PACE preference data are used for post-training across both benchmarks and all four backbone models. The gains tend to diminish

Table 5: Accuracy (%) on TIND negated examples, with results broken down by verbal and non-verbal negation. **All** denotes accuracy over all negated examples, including both types. Values in parentheses indicate absolute improvements over the corresponding off-the-shelf model.

Model	Setting	Verbal	Non-verbal	All
Llama3.2-3B	Off-the-shelf	51.34	49.62	50.17
	PACE	54.85 (+3.51)	51.11 (+1.49)	52.30 (+2.13)
Llama3-8B	Off-the-shelf	53.48	49.69	50.90
	PACE	66.55 (+13.07)	57.46 (+7.77)	60.36 (+9.46)
Qwen2.5-3B	Off-the-shelf	46.42	46.08	46.19
	PACE	46.60 (+0.18)	46.16 (+0.08)	46.30 (+0.11)
Qwen2.5-7B	Off-the-shelf	51.23	45.75	47.50
	PACE	52.69 (+1.46)	45.82 (+0.07)	48.02 (+0.52)

at larger data scales, with relatively smaller improvements from 3.5K to 4.5K samples, suggesting that performance begins to approach saturation under the current setup. Although Qwen2.5-7B shows a comparatively modest scaling trend on ScoNe, its performance after post-training with the 4.5K training set remains significantly higher than that of the off-the-shelf model according to our significance test.

Overall, these results indicate that increasing the amount of PACE preference data generally provides stronger supervision, while the marginal benefit decreases as the dataset grows. The consistent gains with additional PACE samples also suggest that the constructed preference data provides useful and reliable supervision.

F.2 Fine-Grained Analysis of Negation Types on TIND

TIND (García-Ferrero et al., 2023) is a large, semi-automatically generated dataset containing approximately 400K descriptive sentences about commonsense knowledge. An important property of TIND is that its negated examples distinguish between *verbal* and *non-verbal* negation. In verbal negation, the negation marker is associated with the verb and directly affects the predicate, whereas in non-verbal negation, it is associated with other sentence constituents such as objects or adjuncts. For illustration, a verbal-negation example may state that "a bird does *not* live underwater," while a non-verbal example may express that "in *no* season do leaves remain unchanged." This distinction allows us to examine more closely where the gains from PACE arise. We therefore partition the negated TIND examples according to these two negation types and compute accuracy separately for each subset. Detailed results are reported in Table 5, from which we identify three consistent observations regarding the effects of PACE across negation types.

Non-verbal negation is more challenging for off-the-shelf models. All four off-the-shelf models perform worse on non-verbal than verbal negation. For example, Llama3-8B achieves 53.48% accuracy on verbal negation but only 49.69% on non-verbal negation, while Qwen2.5-7B achieves 51.23% and 45.75%, respectively. This pattern suggests that non-verbal negation may be more challenging for LLMs, potentially because negation interacts with a wider range of sentence constituents and scope configurations.

PACE improves performance on both negation types. DPO post-training with PACE improves performance on both verbal and non-verbal negation across all four backbone models. For verbal negation, the gains range from +0.18 to +13.07 points, while improvements on non-verbal negation range from +0.07 to +7.77 points. These results indicate that the benefits of PACE are not restricted to a single form of negation.

PACE yields larger gains on verbal than non-verbal negation. The improvement is consistently larger for verbal negation across all four models. For instance, Llama3.2-3B improves by +3.51 points on verbal negation compared with +1.49 on non-verbal negation, while Llama3-8B improves by +13.07 and +7.77 points, respectively. A similar pattern holds for both Qwen models. Together with the lower off-the-shelf performance on non-verbal examples, this suggests that non-verbal negation may present a more difficult learning problem, with its associated patterns being less readily captured during preference-based post-training.

Table 6: Distribution of affirmative and negated examples in the CondaQA and NevIR test sets.

Dataset	Affirmative	Negated	Total
CondaQA	1,807 (24.96%)	5,433 (75.04%)	7,240
NevIR	1,269 (45.88%)	1,497 (54.12%)	2,766

Table 7: Accuracy (%) on affirmative examples from CondaQA and NevIR before and after DPO post-training with PACE. Gain denotes the absolute change in percentage points (pp).

Benchmark	Model	Off-the-shelf	PACE	Gain
CondQA	Llama3.2-3B	51.02	54.12	+3.10
	Llama3-8B	46.60	51.74	+5.15
	Qwen2.5-3B	47.15	49.86	+2.71
	Qwen2.5-7B	34.70	35.14	+0.44
NevIR	Llama3.2-3B	50.83	49.33	-1.50
	Llama3-8B	44.29	52.25	+7.96
	Qwen2.5-3B	63.20	63.44	+0.24
	Qwen2.5-7B	67.61	68.48	+0.87

Overall, PACE improves both verbal and non-verbal negation, while the larger gains on verbal cases suggest that non-verbal negation remains more challenging to learn.

F.3 Performance on Affirmative Examples

Although our primary evaluation focuses on negation understanding, CondaQA and NevIR also contain affirmative examples that do not include negation. The corresponding statistics are reported in Table 6. We therefore examine whether post-training with PACE affects performance on these examples and whether its benefits come at the cost of performance on non-negated inputs. For each benchmark, we isolate the affirmative subset and compare the corresponding off-the-shelf backbone with the same model after DPO post-training on PACE preference data.

As shown in Table 7, PACE improves accuracy on affirmative examples in seven of the eight model-benchmark settings. The only decrease is observed for Llama3.2-3B in one benchmark, where accuracy decreases by 1.50 percentage points. In contrast, several settings show considerably larger improvements, with gains reaching 7.96 percentage points. In particular, Llama3-8B shows substantial improvements on the affirmative subsets of both benchmarks.

Overall, these results indicate that improving negation understanding with PACE does not generally come at the cost of performance on affirmative examples. Instead, PACE preserves or improves affirmative-example performance in most settings, suggesting that its benefits are not limited to negated inputs. One possible explanation is that learning from contrasting correct- and incorrect-answer justifications may improve the model's ability to distinguish semantically appropriate from inappropriate interpretations more generally.

F.4 Fine-Grained Analysis of Positional Bias

In the main paper, we have shown that the pairwise comparison and BT aggregation strategies in our **PACE** framework are less susceptible to positional bias than the commonly used direct selection approach. Here, we provide a more fine-grained analysis by comparing preference distributions along the two reasoning paths induced by our answer-conditioned prompting strategy.

Specifically, we examine whether the preference distribution differs between responses conditioned on the correct answer and those conditioned on an incorrect answer, corresponding to sound and unsound reasoning paths, respectively. This allows us to analyze positional bias separately for the two types of candidate responses used in our preference construction pipeline. To ensure a fair comparison between preference extraction strategies, we use the same judge model, Llama3-8B, across all settings. The resulting preference distributions are shown in Figure 10.

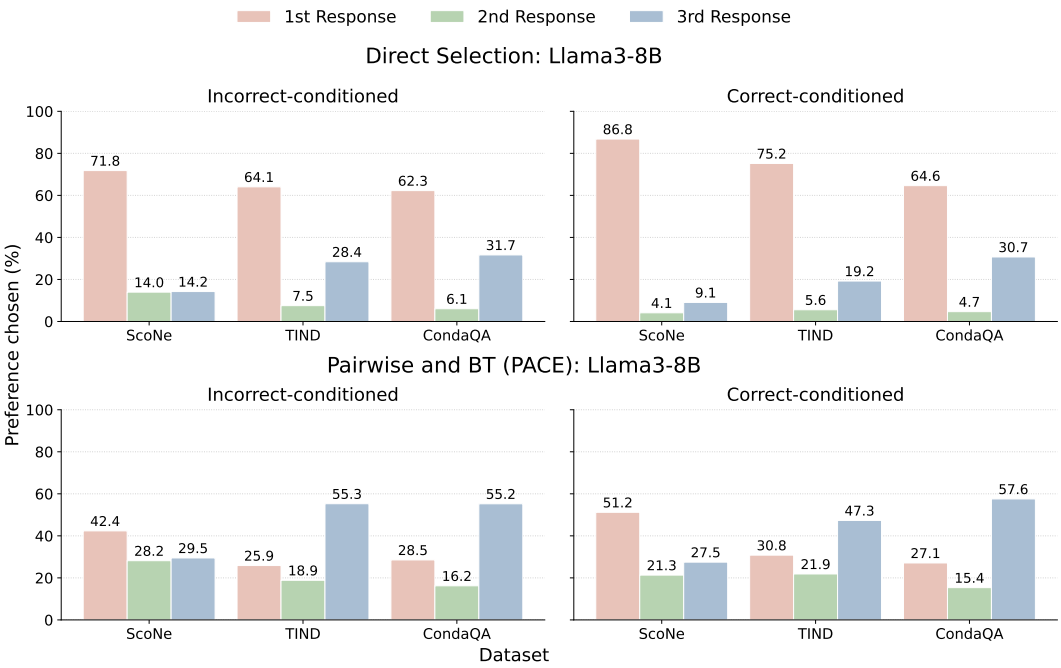

Figure 10: Distribution of chosen preferences depending on the position in which they are presented under Direct judging and Pairwise BT (PACE) using Llama3-8B as the judge model. Results show the positional distribution (1st, 2nd, or 3rd) of top-ranked responses for samples generated using correct-answer and incorrect-answer prompting strategies across three source datasets.

Under both correct-conditioned and incorrect-conditioned prompting settings, the Direct judging strategy exhibits clear positional bias, consistently favoring the first response as the top-ranked choice. In contrast, our Pairwise BT strategy substantially mitigates this effect across both prompting settings.

These results further suggest that positional bias primarily arises from the preference extraction strategy itself, and is minimally affected by the prompting setting used for response generation. Overall, positional bias appears slightly weaker when preference extraction is performed on responses generated under incorrect-answer conditioning compared to correct-answer conditioning.

G Formalization of the PACE Pipeline

Algorithm 1 provides a formal summary of the complete PACE preference-construction procedure. Given a seed dataset

$$\mathcal{D} = \{(x_i, y_i^+)\}_{i=1}^N,$$

each instance consists of a problem x_i and its known correct answer y_i^+. Under the binary answer setting considered in this work, we obtain the corresponding incorrect answer as

$$y_i^- = \mathrm{Flip}(y_i^+).$$

For compactness, we use $a \in \{+, -\}$ to denote the correct- and incorrect-answer-conditioned branches.

Answer-conditioned response generation. PACE generates candidate responses conditioned on both answers. For each problem x_i and answer y_i^a, the generators $\mathcal{G}$ produce a candidate set

$$\mathcal{R}_i^a = \mathrm{Generate}(x_i, y_i^a, \mathcal{G}).$$

Each candidate response can be represented as

$$r = (y_i^a, z),$$

Algorithm 1: PACE Preference Data Construction

Input : Seed dataset $\mathcal{D} = \{(x_i, y_i^+)\}_{i=1}^N$; generators $\mathcal{G}$; judge model J
Output : Preference dataset $\mathcal{P} = \{(x_i, r_i^c, r_i^r)\}$

Function `ExtractPreference`$(\mathcal{R}, x, y, J)$**:**
  $\mathcal{C} \leftarrow \emptyset$; `// pairwise preferences`
  foreach $(r_a, r_b) \in \mathrm{Pairs}(\mathcal{R})$ **do**
    $c_{ab} \leftarrow J(x, y, r_a, r_b)$;
    $\mathcal{C} \leftarrow \mathcal{C} \cup \{c_{ab}\}$;
  $\mathbf{s} \leftarrow \mathrm{BT}(\mathcal{C})$; `// Bradley--Terry scores`
  return $\arg\max_{r_j \in \mathcal{R}} s_j$;

$\mathcal{P} \leftarrow \emptyset$;
foreach $(x_i, y_i^+) \in \mathcal{D}$ **do**
  $y_i^- \leftarrow \mathrm{Flip}(y_i^+)$;
  foreach $a \in \{+, -\}$ **do**
    $\mathcal{R}_i^a \leftarrow \mathrm{Generate}(x_i, y_i^a, \mathcal{G})$;
    $\mathcal{R}_i^a \leftarrow \mathrm{Filter}(\mathcal{R}_i^a, x_i, y_i^a)$;
    $\hat{r}_i^a \leftarrow$ `ExtractPreference`$(\mathcal{R}_i^a, x_i, y_i^a, J)$;
  $r_i^c \leftarrow \hat{r}_i^+$;
  $r_i^r \leftarrow \hat{r}_i^-$;
  $\mathcal{P} \leftarrow \mathcal{P} \cup \{(x_i, r_i^c, r_i^r)\}$;
return $\mathcal{P}$;

where y_i^a is the provided answer and z is the generated justification supporting that answer. Thus, $\mathcal{R}_i^+$ contains responses conditioned on the known correct answer, whereas $\mathcal{R}_i^-$ contains responses conditioned on the corresponding incorrect answer. Conceptually, these two sets provide candidates reflecting sound and plausible but unsound reasoning, respectively, without requiring the generator to determine the correct answer itself.

Candidate filtering. Before preference extraction, PACE removes low-quality or inconsistent responses using the filtering procedure described in the main text. Formally,

$$\widetilde{\mathcal{R}}_i^a = \mathrm{Filter}(\mathcal{R}_i^a, x_i, y_i^a).$$

For simplicity, Algorithm 1 reuses the notation $\mathcal{R}_i^a$ for the filtered candidate set. This stage ensures that subsequent preference extraction operates only on candidates that satisfy the required quality and consistency constraints.

Pairwise preference extraction. PACE extracts preferences independently from each answer-conditioned candidate set. Given a filtered set $\mathcal{R}$, the judge model J compares every candidate pair (r_p, r_q) conditioned on the corresponding problem x and answer y. Each comparison produces a pairwise preference

$$c_{pq} = J(x, y, r_p, r_q),$$

and the complete set of pairwise judgments is

$$\mathcal{C} = \{c_{pq} : (r_p, r_q) \in \mathrm{Pairs}(\mathcal{R})\}.$$

Rather than selecting a response directly from a single judgment, PACE aggregates the pairwise preferences using the Bradley–Terry (BT) model (Bradley & Terry, 1952). For candidate responses r_p and r_q with latent scores s_p and s_q, the BT model defines

$$P(r_p \succ r_q) = \frac{\exp(s_p)}{\exp(s_p) + \exp(s_q)}.$$

The scores are estimated from the complete set of pairwise judgments $\mathcal{C}$, yielding

$$\mathbf{s} = \mathrm{BT}(\mathcal{C}),$$

where s_j denotes the BT score for candidate r_j. PACE then selects the highest-ranked response:

$$\hat{r} = \arg\max_{r_j \in \mathcal{R}} s_j.$$

Applying this procedure separately to the two answer-conditioned sets gives

$$\hat{r}_i^+ = \text{ExtractPreference}(\mathcal{R}_i^+, x_i, y_i^+, J)$$

and

$$\hat{r}_i^- = \text{ExtractPreference}(\mathcal{R}_i^-, x_i, y_i^-, J).$$

Preference-pair construction. The highest-ranked correct-answer-conditioned response is used as the chosen response,

$$r_i^c = \hat{r}_i^+,$$

while the highest-ranked incorrect-answer-conditioned response becomes the rejected response,

$$r_i^r = \hat{r}_i^-.$$

PACE therefore constructs the final preference dataset as

$$\mathcal{P} = \{(x_i, r_i^c, r_i^r)\}_{i=1}^N.$$

Each triple contains a problem together with a chosen and rejected response, providing the preference supervision used for subsequent DPO post-training. PACE itself terminates at the construction of $\mathcal{P}$; DPO is used only as the downstream preference optimizer.

H Benchmark Details

We evaluate all DPO-trained models in a zero-shot setting on the test splits of five downstream benchmarks. Following the original evaluation protocols of each benchmark, we report accuracy for all datasets except NevIR, where the evaluation metric is pairwise accuracy. Below, we provide detailed descriptions of the benchmarks and summarize previously reported zero-shot results of decoder-only LLMs on these benchmarks as a reference.

H.1 Benchmark Descriptions

CondaQA is a large contrastive question answering benchmark designed to evaluate contextual negation reasoning. Each instance consists of a question paired with a passage containing a negated statement relevant to the question. Answers may take several forms, including Yes, No, Don't know, or a span extracted from the passage. The dataset contains 5,382 training instances, 1,110 validation instances, and 7,240 test instances.

To construct the PACE preference data, we use only training instances with `YES` or `NO` answers, yielding 5,077 instances (94.33%). We discard the remaining 305 instances (5.67%), whose answers are `Don't know` or text spans and therefore cannot be directly converted into incorrect answers through label flipping. We follow the same zero-shot evaluation setting and evaluation metric (accuracy) used in the original work. Figure 11 shows an example evaluation prompt.

ScoNe is a negation-focused lexical entailment benchmark for natural language inference (NLI), designed to isolate how the scope of negation affects entailment relations. The dataset is organized into six contrast splits, each containing 1,002 training instances and 200 test instances, for a total of 1,200 test instances. The six splits differ in the number of negations and whether they take scope over the lexical item relevant to the entailment decision:

- *no_negation*: no negation is present.
- *one_not_scoped*: one negation is present but does not scope over the relevant lexical item.
- *two_not_scoped*: two negations are present, neither scopes over the relevant lexical item.
- *two_scoped*: two negations are present; one scopes over the relevant lexical item, while the other scopes over the first negation, canceling its effect.
- *one_scoped*: one negation scopes over the relevant lexical item.
- *one_scoped_one_not_scoped*: two negations are present, but only one scopes over the relevant lexical item and affects the entailment relation.

Benchmark	Model	Accuracy (%)
CondaQA	GPT-3 (davinci)	43.72
	InstructGPT (text-davinci-002)	**54.00**
ScoNe	InstructGPT (text-davinci-003)	**64.00**
TIND	Vicuna-13B	**57.80**
	WizardLM-30B	57.30
NeQA	InstructGPT (text-davinci-002)	56.00
	InstructGPT (text-davinci-003)	**71.00**

Table 8: Previously reported results of decoder-only LLMs on the evaluation benchmarks. Bold indicates the best prior result for each benchmark.

ScoNe also evaluates multiple prompting templates to study their impact on model performance. We follow the *Hypothesis Q* template used in the original work:

Assume that **Premise**. Is it then definitely true that **Hypothesis**? Answer yes or no.

Our complete evaluation prompt example is shown in Figure 12.

TIND is currently the largest benchmark for negation and commonsense reasoning, comprising 381,300 artificially generated sentences in standard English. Its test split contains 90,281 instances labeled as either `True` or `False`, with or without negation. The dataset was also subjected to extensive manual inspection to ensure the quality of the generated examples.

Following the negation taxonomy of Pullum & Huddleston (2002), TIND further distinguishes different types of negation. In our analysis, we focus on the *verbal* and *non-verbal* negation categories. For verbal negation, the negation marker (e.g., *not*) is associated with and directly affects the verb, whereas non-verbal negation (e.g., *no*) is associated with other sentence constituents such as objects or adjuncts.

Following Scheurer et al. (2023), TIND adopts a log-probability-based evaluation strategy in which each sentence is evaluated independently as a binary `True`/`False` prediction. Given a prompt pt, the prediction A is determined from the normalized probabilities derived from the token log-probabilities:

$$A = \begin{cases} \texttt{True}, & \text{if } \dfrac{p(\texttt{True} \mid pt)}{p(\texttt{True} \mid pt) + p(\texttt{False} \mid pt)} > 0.5, \\ \texttt{False}, & \text{otherwise.} \end{cases}$$

For fair comparison, we follow the original evaluation setup, including the metric, prompt template, and log-probability-based prediction procedure. An example prompt is shown in Figure 12.

NeQA is a multiple-choice commonsense question answering benchmark designed to evaluate negation understanding in a simple setting. Each instance contains a negated question with two answer choices, one correct and one incorrect. The test set includes 100 negation-related questions. NeQA also exhibits inverse scaling behavior (Wei et al., 2023; McKenzie et al., 2024), with performance not consistently improving as model size increases, further highlighting the difficulty of negation for current LLMs. An example evaluation prompt is shown in Figure 12.

NevIR is a benchmark for evaluating negation understanding in information retrieval, specifically document ranking given a query. Each original instance consists of two contrastive queries and two documents, where each query corresponds to a different relevant document and the queries differ primarily in the presence or absence of negation. The original task evaluates whether a retrieval model assigns a higher score to the relevant document for both queries.

To adapt NevIR to our decoder-only LLM evaluation setting, we reformulate each original instance into two samples, each containing one query and two candidate documents, and ask the model to select the relevant document. This results in 2,766 evaluation samples. For the main evaluation, we retain the original pairwise accuracy metric: an original query pair is counted as correct only if the model selects the correct document for both queries. For our affirmative-example analysis, however, we separate the reformulated samples according to whether the query contains negation and report accuracy over individual samples.

In this task, you're expected to write answers to questions involving reasoning about negation. The answer to the question should be "yes", "no", "don't know" or a phrase in the passage. Questions can have only one correct answer.

Passage: In October 2010, Daniela Soleri - Paolo Soleri's daughter - resigned from the Cosanti Foundation board, citing abuse by her father. She stated that some of Soleri's inner circle had been told decades earlier, but they remained actionless about it at the time. After the resignation, Soleri stepped down as chairman, but the board made no public statement on the reasons.

Question: Is it likely that people in Paolo's inner circle talked to him about the abuse of his daughter?

Answer:

Figure 11: Zero-shot prompt example used to evaluate LLMs with CondaQA.

NeQA

The following are multiple choice questions (with answers) about common sense.

Question: Crying is not part of ?
A. life
B. love
Answer:

ScoNe

Question: Assume that the man does not own a dog and does not own a cat. Is it then definitely true that the man does not own an animal and does not own a cat? Only answer yes or no.

Answer:

TIND

Is the following statement True or False? Answer only True or False.

Question: Notice refers to an announcement containing information about an event.

Answer:

Figure 12: Zero-shot prompt examples used for evaluation on NeQA, ScoNe, and TIND.

The original NevIR study does not report results for decoder-only LLMs and therefore provides no directly comparable baseline for our setting. We design our own prompting template and adopt a log-probability-based evaluation strategy similar to that used for TIND. The evaluation prompt is shown in Figure 13.

H.2 Prior Results on the Evaluation Benchmarks

To contextualize our results, Table 8 summarizes previously reported results of decoder-only LLMs on CondaQA, ScoNe, TIND, and NeQA. These results are taken from the original benchmark papers and provide the source of the *Best Prior* results reported in the main paper. For each benchmark, we report the evaluated model and its published zero-shot performance for reference.

In this task, you will be given two candidate documents and a query. Your goal is to decide which document answers the query.

##DOCUMENT 1:
In 2012 Dyson published (with William H. Press) a fundamental new result about the prisoner's dilemma in the Proceedings of the National Academy of Sciences of the United States of America. He wrote a foreword to a treatise on psychic phenomena in which he concluded that "ESP is real... but cannot be tested with the clumsy tools of science".

##DOCUMENT 2:
In 2012 Dyson published (with William H. Press) a fundamental new result about the prisoner's dilemma in the Proceedings of the National Academy of Sciences of the United States of America. He wrote a foreword to a treatise on psychic phenomena in which he concluded that "ESP is real... and has been tested with the clumsy tools of science".

Output "1" if the first document contains the answer, or "2" if the second document contains the answer.

You MUST answer with exactly one character: "1" or "2".

Query:
Who said that ESP is real though not testable by science? Only output "1" or "2".

Response:

Figure 13: Zero-shot prompt example used to evaluate LLMs on NevIR.