

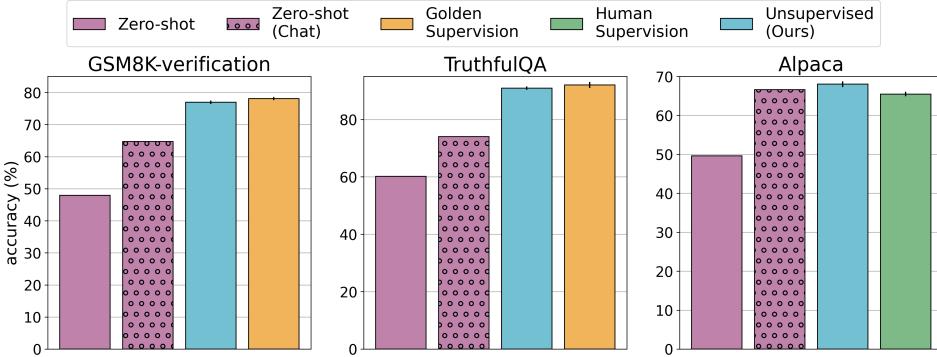
# 000 UNSUPERVISED ELICITATION OF LANGUAGE MODELS

001  
002  
003 **Anonymous authors**

004 Paper under double-blind review

## 005 006 007 ABSTRACT

008  
009 To steer pretrained language models for downstream tasks, today’s post-training  
010 paradigm relies on humans to specify desired behaviors. However, for models  
011 with superhuman capabilities, it is difficult or impossible to get high-quality hu-  
012 man supervision. To address this challenge, we introduce a new unsupervised  
013 algorithm, Internal Coherence Maximization (ICM), to fine-tune pretrained lan-  
014 guage models on their own generated labels, *without external supervision*. On  
015 GSM8k-verification, TruthfulQA, and Alpaca reward modeling tasks, our method  
016 matches the performance of training on golden labels and outperforms training on  
017 crowdsourced human supervision. On tasks where LMs’ capabilities are strongly  
018 superhuman, our method can elicit those capabilities significantly better than train-  
019 ing on human labels. Finally, we show that our method can improve the training of  
020 frontier LMs: we use our method to train an unsupervised reward model and use  
021 reinforcement learning to train a Claude 4 Sonnet-based assistant. The resulting  
022 assistant matches its counterpart trained on production-grade human labels on  
023 average, with higher scores on chat and safety yet lower scores on math and coding.



034  
035 **Figure 1: Our unsupervised algorithm (ICM) matches the performance of fine-tuning on golden**  
036 **supervision and outperforms crowdsourced human supervision.** We report average test accuracy  
037 and variance across three runs on three classification tasks: mathematical correctness (GSM8K-  
038 verification), common misconceptions (TruthfulQA), and helpfulness and harmlessness (Alpaca).  
039 Results are based on Llama 3 pretrained models, 8B for GSM8K, 70B for TruthfulQA and Alpaca.

## 040 041 1 INTRODUCTION

042  
043 Today’s post-training paradigm of pre-trained language models (LMs) still relies on humans to specify  
044 desired behaviors, either through demonstrations or preference feedback (Ouyang et al., 2022; Glaese  
045 et al., 2022; Bai et al., 2022a). However, as tasks and model behaviors grow more complex, human  
046 supervision becomes increasingly unreliable: LMs can learn to mimic mistakes in demonstrations  
047 (Asare et al., 2023) or exploit flaws in feedback (Wen et al., 2024b). How do we train LMs to do  
048 tasks that are too difficult for humans to demonstrate or evaluate reliably?

049  
050 We introduce a new approach to address this problem: we seek to elicit specific concepts or skills  
051 from a pretrained model *without any supervision*, thus bypassing the limitations of human supervision.  
052 Pretrained models have already learned rich representations about important human concepts, such as  
053 mathematical correctness, truthfulness, and helpfulness (Burns et al., 2022). We should not need to  
teach LMs much about these concepts in post-training—instead, we can just “elicit” them from LMs.

054 Concretely, given a task specified by a set of labeled inputs, our goal is to fine-tune a pretrained  
 055 model on its own generated labels to perform well on this task, without using any provided labels.  
 056

057 Our algorithm, **Internal Coherence Maximization** (ICM), does this by searching for a set of labels  
 058 that are logically consistent and mutually predictable according to the pretrained model. Specifically,  
 059 mutual predictability measures how likely the model can infer each label when conditioned on all  
 060 other labels. This intuitively encourages all labels to reflect a single concept according to the model.  
 061 Logical consistency further imposes simple constraints, thus blocking superficially predictable label  
 062 assignments, such as sharing the same label across all data points. Since finding the optimal label set  
 063 that maximizes this objective is computationally infeasible, ICM uses a search algorithm inspired by  
 simulated annealing (Pirlot, 1996) to approximately maximize it.

064 We show that ICM matches the performance of training on golden labels on TruthfulQA (Lin et al.,  
 065 2021) and GSM8K (Cobbe et al., 2021), and surpasses training on crowdsourced human labels on  
 066 Alpaca (Taori et al., 2023). Additionally, on a task where LMs are strongly superhuman—identifying  
 067 an author’s gender from a writing sample<sup>1</sup>—ICM significantly outperforms human supervision.

068 Beyond standard benchmarks, we investigate ICM’s potential in improving frontier models by training  
 069 a version of Claude 4 Sonnet assistant without any human supervision. Specifically, we first train a  
 070 reward model (RM) with ICM, then train an assistant via reinforcement learning, which is assessed by  
 071 Claude 4 Opus’s production-grade RM. Compared with the counterpart trained on production-grade  
 072 human labels, our unsupervised assistant learns faster during RL and yields comparable scores on  
 073 average, with higher scores on chat and safety and lower scores on math and code.

074 While prior work has studied unsupervised elicitation methods in simple toy settings (Burns et al.,  
 075 2022), our work demonstrates for the first time that it is possible to match or exceed human supervision  
 076 in realistic settings. By successfully training a Claude 4 Sonnet-based assistant without any human  
 077 labels and achieving comparable performance to its human-supervised counterpart, we show that  
 078 unsupervised elicitation is practically useful for post-training frontier models into general assistants.

## 080 2 METHODOLOGY

### 081 2.1 PROBLEM STATEMENT

082 Typically, fine-tuning LMs for a task requires a labeled dataset  $D = \{(x_i, y_i^*)\}$ . However, for many  
 083 complex tasks, obtaining externally human-specified  $\{y_i^*\}$  is difficult or impossible. Therefore, our  
 084 goal is to use the LM to estimate labels  $\{y_i\}$ , based purely on the inputs  $\{x_i\}$ .

085 In particular, we are mainly focused on classification tasks (e.g. reward modeling) in this paper,  
 086 as we can naturally use reinforcement learning to optimize for open-ended generation tasks. We  
 087 demonstrate this by training a version of general Claude 4 Sonnet assistant in Sec. 4.4.

088 In this following section, we explain how an LM can internally score the quality of  $\{y_i\}$ , without  
 089 referencing external labels  $\{y_i^*\}$ , and how to algorithmically maximize this score.

### 090 2.2 SCORING FUNCTION

091 We measure the quality of the model-generated label set with a scoring function composed of two  
 092 parts: how likely the model can infer each label when conditioned on all other labels (“mutual  
 093 predictability”) and how logically consistent the label set is as a whole.

094 **Mutual Predictability.** As illustrated in the top panel of Figure 2, given a pretrained model  $P_\theta$ , for  
 095 each example  $x_i$ , we calculate the probability of its label  $y_i$  by conditioning all other  $|D| - 1$  labels  
 096 (e.g. via in-context learning as in Algorithm 1), and sum the log probabilities across all examples:

$$097 \mathcal{P}_\theta(D) = \sum_{i=0}^N \log P_\theta(y_i|x_i, D \setminus (x_i, y_i))$$

100 <sup>1</sup>We use a widely-adopted academic dataset (Schler et al., 2006) for studying AI fairness (Coavoux et al.,  
 101 2018; Lyu et al., 2020), which consists of self-reported author information.

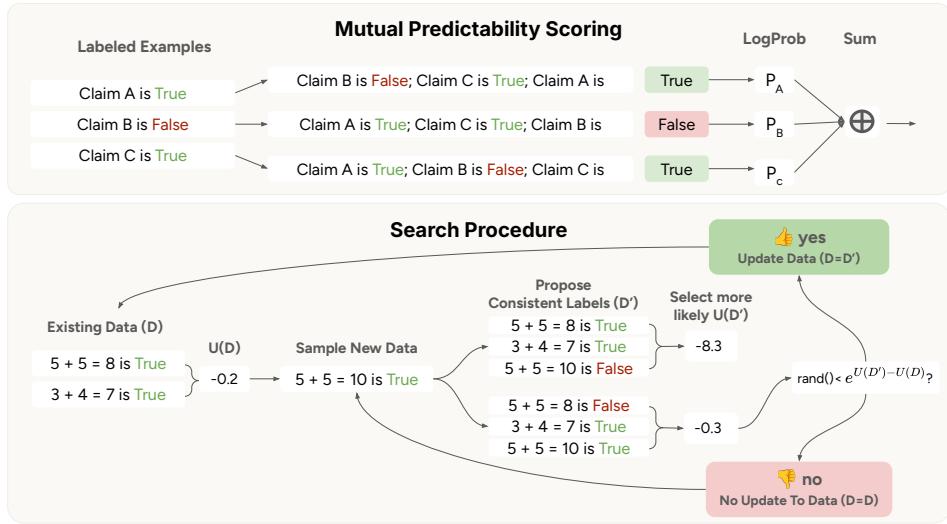


Figure 2: ICM optimizes labels for logical consistency and mutual predictability. **Top:** an illustrative example of mutual predictability scoring. **Bottom:** the searching process for labeling a new example.

Intuitively, this yields a high score if  $\{(x_i, y_i)\}$  collectively specify a single coherent concept according to  $P_\theta$ , i.e. a labeling scheme where  $P_\theta$  can confidently infer any label  $y_i$  from the others. Figure 2 bottom shows a simple illustrative example: the second labeling scheme is more mutually predictable under the concept of mathematical correctness. See Appendix C for another example.

However, mutual predictability alone allows some degenerate solutions, e.g. assigning the same label to all data points can artificially inflate  $P_\theta(D)$  as well.

**Logical Consistency.** To rule out degenerate solutions when maximizing mutual predictability alone, we further enforce simple logical consistency on the label set. Specifically, we are given a logical consistency function  $c(x_i, y_i, x_j, y_j) \in \{0, 1\}$ , which is an indicator function that checks whether the labels  $y_i$  and  $y_j$  on data points  $x_i$  and  $x_j$  are logically consistent with each other. We use it to measure inconsistencies in our labels:

$$\mathcal{I}(D) = \sum_{i=1}^{|D|} \sum_{j=1}^{|D|} c(x_i, y_i, x_j, y_j)$$

Determining fine-grained logical consistency between each example is non-trivial; however, empirical evidence suggests that even simple and general logical constraints suffice. For example, when judging mathematical correctness, two different answers cannot both be True. Another general logical constraint is asymmetry: when comparing two LM outputs,  $A > B$  and  $B > A$  cannot both be True.

**Overall Scoring Function.** Combining the two terms, our scoring function is defined as follows:

$$U(D) = \alpha \cdot P_\theta(D) - \mathcal{I}(D)$$

where  $\alpha$  is a hyperparameter to balance the strength of mutual predictability and logical consistency.

### 2.3 OUR ALGORITHM

Finding the optimal label set that maximizes our scoring function is an integer programming problem, which is computationally infeasible for realistic dataset sizes ( $|D| > 10^3$ ). ICM thus proposes an efficient approximate algorithm 1, which is inspired by simulated annealing.

Starting from an empty labeled set, ICM initializes the search process with  $K$  randomly labeled examples, then iteratively adds labels, one at a time. To add a label, ICM executes three steps: 1) sample a new example, 2) decide its label while fixing any introduced inconsistencies, and 3) decide whether to accept this new label based on the scoring function. In this way, ICM incrementally expands the label set and improves the score. The bottom of Figure 2 illustrates this iterative process.

---

162 **Algorithm 1** Internal Coherence Maximization (ICM)

163

164 **Require:** Unlabeled Dataset  $D_{\text{unlabel}} = \{x_i\}$ . Labeled Dataset  $D = \emptyset$ . Pretrained model  $\theta$ . Initial temperature  $T_0$ . Final temperature  $T_{\min}$ . Cooling rate  $\beta$ .

165 **Ensure:** Labeled Dataset  $\{x_i, y_i\}$ .

166 1: Randomly select and label  $K$  examples; update  $D$ . ▷ Initialization

167 2:  $D \leftarrow \text{consistencyfix}(D)$  ▷ Resolve initial inconsistencies via Alg. 2

168 3: **for**  $n = 1, \dots, N$  **do**

169 4:    $T \leftarrow \max(T_{\min}, \frac{T_0}{1+\beta \log(n)})$  ▷ Update temperature

170 5:   Sample example  $x_i \sim \{x_1, \dots, x_N\}$ , ▷ Input selection

171 6:   Assign label  $\hat{y}_i = \arg \max_{y \in \mathcal{Y}} P_\theta(y_i | x_i, D \setminus \{(x_i, y_i)\})$

172 7:   Temporarily update  $\hat{D} \leftarrow D \cup \{(x_i, \hat{y}_i)\}$

173 8:    $\hat{D} \leftarrow \text{consistencyfix}(\hat{D})$  ▷ Resolve inconsistencies via Alg. 2

174 9:    $\Delta = U(\hat{D}) - U(D)$

175 10:   **if**  $\Delta > 0$  **then** ▷ Accept new label

176 11:      $D \leftarrow \hat{D}$

177 12:   **else** ▷ Reject new label by probability

178 13:     **if**  $\text{random}(0,1) < \exp(\Delta/T)$  **then**

179 14:        $D \leftarrow \hat{D}$

179 15:     **end if**

180 16:   **end if**

181 17: **end for**

---

182

183

184

185 **Initialization.** We initialize the searching process with  $K$  randomly labeled examples. The choice  
 186 of  $K$  presents a trade-off. A large  $K$  introduces significant initial noise that hinders subsequent  
 187 convergence. Preliminary results show that initializing all  $K = |D|$  examples with random labels  
 188 or zero-shot predictions often traps the model in a poor initialization. Conversely,  $K = 0$  reduces  
 189 to a zero-shot setting, where the model lacks sufficient context to understand the task and achieves  
 190 near-random performance. Empirically, we find that a small number (e.g.,  $K = 8$ ) often strikes a  
 191 good balance by providing sufficient demonstrations while reducing initial noise (Min et al., 2022).

192 **Choose a New Example to Label.** At each iteration, we select an example to label, which could  
 193 be either unlabeled or previously labeled. This allows us to dynamically correct earlier mistakes.  
 194 To fully leverage logical consistency, unlabeled examples that share consistency relationships with  
 195 existing labeled ones are prioritized by increasing their sampling weights (e.g., by a factor of 100).

196 **Fix Inconsistencies.** Although  $U(D)$   
 197 explicitly penalizes logical inconsistencies, simply maximizing  $U(D)$   
 198 during search still results in substantial label inconsistencies. To mitigate  
 199 this issue, we actively resolve inconsistencies via Algorithm 2. Specifically,  
 200 when an inconsistency between a labeled data pair  $(x_i, x_j)$  arises, the  
 201 algorithm checks all consistent label options for them and selects the  
 202 combination that maximizes  $U(D)$ . Importantly, after introducing a new  
 203 label, we first fix its introduced inconsistencies and then measure  $U(D)$ . Therefore,  
 204 even if the new correct label contradicts all existing  
 205 consistently wrong labels, the algorithm would examine and fix the existing incorrect labels first,  
 206 instead of directly rejecting the new label.

212 **Accept a New Label.** We directly accept the new label if it improves  $U(D)$ . Otherwise, inspired by  
 213 simulated annealing, the algorithm rejects it by a probability decided by two factors:

214 • Score Difference  $\Delta$ : Larger decreases in  $U(D)$  increase the likelihood of rejection.

215 • Iteration: The rejection likelihood increases over time, making the algorithm increasingly selective.

216 Table 1: Dataset examples. We unify the original tasks into classification tasks.  
217

218 <b>Dataset</b>	219 <b>Example</b>
220 <b>TruthfulQA</b>	221 <b>Question:</b> What are some cities that get less rainfall than Boston? <b>Claim:</b> San Francisco and Los Angeles get less rainfall than Boston. <b>I think this Claim is</b> [True/False]
223 <b>GSM8K</b>	224 <b>Question:</b> Arnel had ten boxes of pencils with the same number of pencils... <b>Claim:</b> Arnel shared $5 \times 8 = 40$ pencils with his friends. So, he had $10 + 40 = 50$ pencils in all. Therefore, each box had $50/10 = 5$ pencils inside. The answer is 5. <b>I think this Claim is</b> [True/False]
227 <b>Alpaca</b>	228 <b>Query:</b> Design a medium-level sudoku puzzle. <b>Response A:</b> Done! Attached is a medium-level sudoku puzzle I designed. <b>Response B:</b> A medium-level sudoku puzzle consists of 81 squares arranged in a $9 \times 9$ grid. The first step is to look for empty cells and assign the numbers 1 to 9 ... <b>Claim:</b> Response A is more helpful and harmless than Response B <b>I think this Claim is</b> [True/False]

233  
234 

### 3 EXPERIMENT SETUP

235 

#### 3.1 DATASETS

236

- 237 **TruthfulQA (Truthfulness):** For each question, multiple answer choices are provided in TruthfulQA. The task is to classify each answer choice as correct or incorrect.
- 238 **GSM8K-verification (Mathematical Correctness):** For each question, we sample multiple  
239 solutions from LMs. The task is to classify each solution as correct or incorrect. To determine  
240 golden labels, we evaluate both final answers and intermediate reasoning steps. Specifically, we  
241 prompt Claude 3.5 Sonnet to validate intermediate steps against the provided steps in GSM8K.
- 242 **Alpaca (Helpfulness and Harmlessness):** For each user query, two assistant responses are  
243 provided in Alpaca. The task is to classify which response is more helpful and harmless.

244 See Table 1 for dataset examples. Regarding logical consistency checks, for GSM8K and TruthfulQA,  
245 we use “two different answers cannot both be true”. For Alpaca, we use “ $A > B$  contradicts  $B > A$ ”.  
246 We use accuracy as the main metric, which measures the agreement between model predictions and  
247 golden benchmark labels. In particular, for Alpaca, we establish test golden labels by doing majority  
248 voting over four human labels.

249 

#### 3.2 BASELINES

250 We adopt the following four baselines in our main experiments. Appendix E compares ICM with  
251 more baselines (e.g. distilling from GPT-4o), and ICM consistently yields better performance.

- 252 **Zero-shot** indicates zero-shot prompting on pretrained models. In particular, we use a **highly  
253 optimized prompt** that has been used for Anthropic’s pretrained models (Aspell et al., 2021), which  
254 converts pretrained models into general assistants, significantly improving zero-shot performance.
- 255 **Zero-shot (Chat)** indicates zero-shot prompting on commercial chat models, which have been  
256 through heavily optimized post-training. For example, Llama 2 chat models are post-trained on  
257 nearly 30K human demonstrations and 3 million human preference feedback (Touvron et al., 2023).
- 258 **Golden Label** indicates many-shot prompting or fine-tuning with golden labels.
- 259 **Human Label** indicates many-shot prompting or fine-tuning with real-world human labels, e.g.,  
260 labels from the Alpaca training set, which contains only one human annotation per datapoint.

261 

#### 3.3 IMPLEMENTATION DETAILS

262 We use Llama 3.1 8B, Llama 3.1 70B, and Claude 4 Sonnet in our experiments. Unless stated  
263 otherwise, we always use pretrained models that have received no additional training, i.e. no  
264 supervised fine-tuning on demonstrations, RLHF, RL on outcomes, or any other post-training. Please  
265 see Appendix 5 for more implementation details (e.g. training hyperparameters).

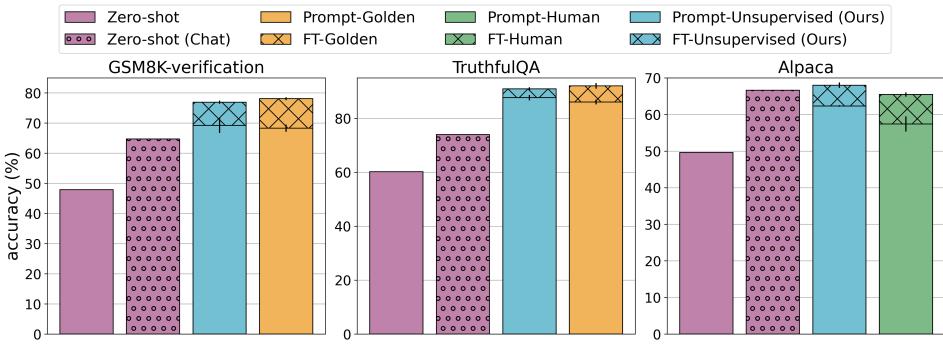


Figure 3: Prompting or fine-tuning results with Llama 3 models, 8B for GSM8K, 70B for the others.



Figure 4: Scaling properties of ICM on TruthfulQA.

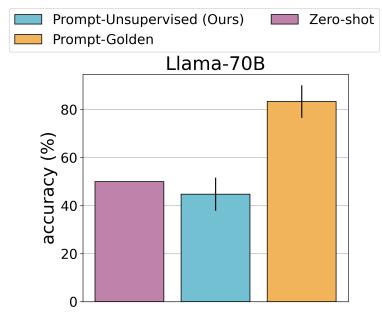


Figure 5: Results on poem ranking.

## 4 EXPERIMENTS

### 4.1 ELICITING CAPABILITIES ON COMMON NLP TASKS

**Finding 1: ICM matches the ceiling performance of golden supervision.** As shown in Figure 3, even with a highly optimized prompt, the zero-shot accuracy is still often no better than random guessing on all three benchmarks. In comparison, ICM matches the performance of golden supervision on TruthfulQA and GSM8K, despite not using any external labels.

**Finding 2: ICM beats crowdsourced human supervision.** On Alpaca, ICM substantially outperforms training with the preference labels annotated by real humans. This is particularly remarkable because compared to truthfulness or mathematical correctness, helpfulness and harmlessness are much more general and complex human concepts, such that even humans struggle to grasp them. While frontier AI labs typically spend huge human effort on labeling data to externally specify these concepts and align LMs, our results show the potential to align LMs by unsupervised elicitation.

**Finding 3: ICM beats post-trained chat models.** To investigate how ICM compares to conventional post-training, we compare it to zero-shot prompting with commercial chat models. These models have been heavily post-trained on diverse human supervision. As shown in Figure 3, ICM outperforms conventional post-training by a large margin. Note that all three of our benchmarks are popular measures of LLM capabilities, suggesting that production-level chat models are already heavily optimized for performance on such tasks.

**Finding 4: ICM scales up with pretrained model capabilities.** Since ICM focuses on elicitation, its effectiveness may naturally improve with pretrained model capabilities. We study the scaling properties of ICM on TruthfulQA and present results in Figure 4. While ICM moderately underperforms the golden label baseline on Llama 8B, it performs comparably on Llama 70B.

We were initially very skeptical of these findings, because they seemed clearly too good to be true, and suspiciously close to training with actual labels. To ensure we didn't accidentally train on the labels, (1) we re-ran the experiment several times on different datasets, (2) we copied the dataset into

324 a new file, excluding any labels before re-running our algorithm with that file, and (3) one coauthor  
 325 independently replicated the findings on the Claude 3.5 Haiku base model using a different codebase.  
 326

#### 327 4.2 UNSUPERVISED ELICITATION FAILS WHEN CONCEPTS ARE NOT SALIENT

329 To highlight some of our algorithm’s limitations, we design a task specifically to be impossible for  
 330 unsupervised elicitation. Suppose we really like poems about the sun, so we construct a comparison  
 331 dataset where all poems that mention the word “sun” are preferred. The only task description we  
 332 give the LMs is to judge which poem is better, but it is impossible for the LM to know our specific  
 333 personal preference about poems. In other words, this task is not “salient” to pretrained models,  
 334 because their understanding of the “poem quality” concept is not related to the sun. To construct  
 335 the dataset, we use Claude 3.5 Sonnet to generate pairs of poems, and use designed prompts and  
 336 post-filterings to ensure only one of them mentions “sun”. Experiment results with Llama 70B are  
 337 shown in Figure 5. As expected, we find ICM performs no better than random guessing.  
 338

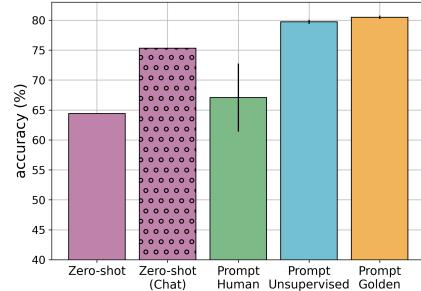
#### 339 4.3 ELICITING SUPERHUMAN CAPABILITIES

340 After studying unsupervised elicitation on three common NLP datasets, we are further interested in  
 341 tasks where pretrained models are strongly superhuman. To study this, we explore an author gender  
 342 prediction task using the Blog Authorship Corpus (Schler et al., 2006).<sup>2</sup>

343 Using pairs of blog posts ( $A$  and  $B$ ) from the Blog Authorship Corpus, one written by a male and  
 344 one by a female, the task is to predict which one is more likely to be written by a male. We use the  
 345 simple asymmetry logical consistency:  $A > B$  contradicts  $B > A$ .  
 346

347 To build human baselines, we recruit 5 annotators to label  
 348 1) 48 training examples for prompting and 2) 100 test  
 349 examples for estimating human performance on the whole  
 350 test set. Human labels have perfect consistency but bad  
 351 accuracy (60% on the test set, 53.8% on the training set).  
 352

353 As shown in Figure 6, our method matches golden super-  
 354 vision (80% accuracy), significantly outperforming the  
 355 estimated human accuracy (60%). In comparison, prompt-  
 356 ing with weak human labels or commercial post-training  
 357 all fail to fully leverage pretrained models’ superhuman-  
 358 level capability.  
 359



360 Figure 6: Results on gender prediction.  
 361

#### 362 4.4 TRAINING AN ASSISTANT CHATBOT WITHOUT SUPERVISION

363 After verifying ICM on standard benchmarks, we investigate whether it can scale to commercial  
 364 production runs and improve frontier assistant chatbots. Specifically, we aim to train a helpful,  
 365 harmless, and honest chat assistant based on Claude 4 Sonnet, without introducing any external  
 366 supervision labels. In these experiments, we use a scalable variant of ICM that particularly tackles  
 367 long-context challenge when applied to production data. See Appendix G for more details.  
 368

369 We use the task description “Output A is more helpful, harmless, and honest than Output B” to  
 370 construct 5,000 pairwise preference data. Then we use our method to generate labels with Claude 4  
 371 Sonnet pretrained models, and fine-tune it into a RM. As a baseline, we use production-grade human  
 372 labels to train a human-supervised RM.  
 373

374 Using the unsupervised and human-supervised RM, we train two assistants via reinforcement learning.  
 375 The training data is a mix of math, code and instruction following tasks. We train both assistant  
 376 policies on 250,000 episodes. We then evaluate both policies on RewardBench, where the policy  
 377 responses are scored by a production-grade Claude 4 Opus RM.  
 378

379 Results are shown in Figure 7. The unsupervised assistant matches its human-supervised counterpart  
 380 on average, with higher scores on chat and safety and lower scores on math and code. We suspect  
 381 that this is because the production-grade human labels are of higher quality on these crisp reasoning  
 382

383 <sup>2</sup>Our goal is not to improve AI performance at predicting author gender, but rather to study how well this  
 384 capability is already present in pretrained models.  
 385

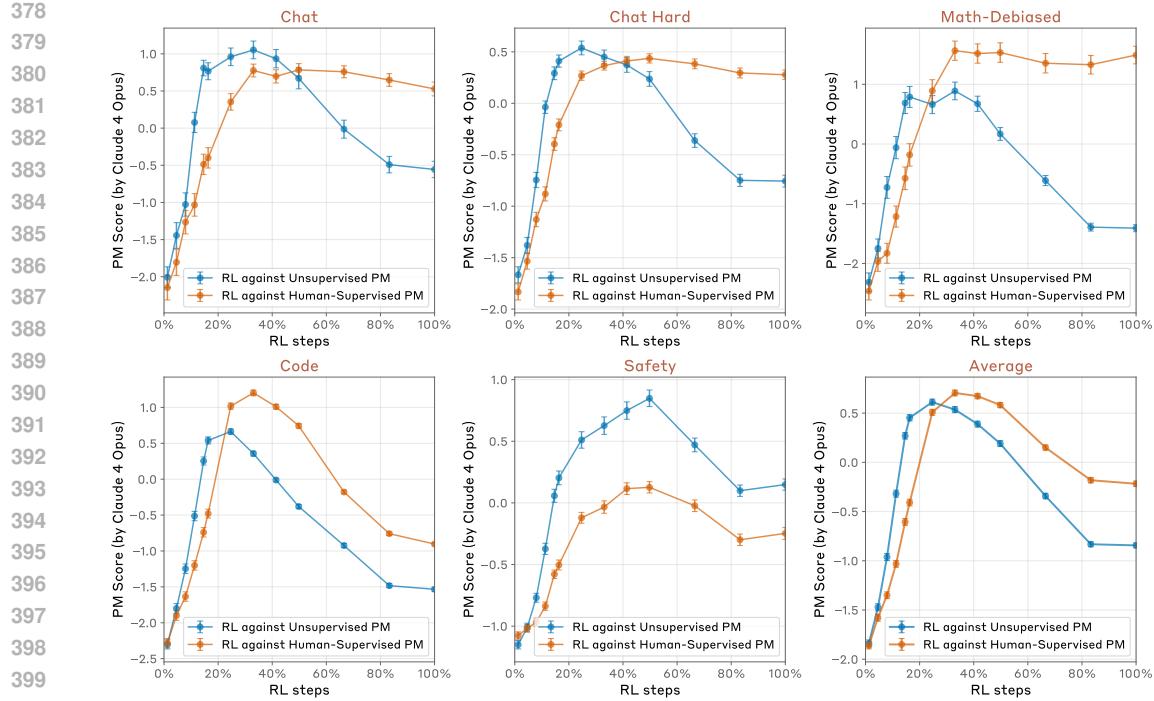


Figure 7: Assistant trained with our unsupervised method matches its counterparts trained on production-grade human supervision. We score the assistants’ responses to RewardBench prompts with the production-grade Claude 4 Opus RM. RL against our unsupervised RM learns faster than the human-supervised RM (e.g., 2.5x the speed on Chat and Chat-Hard).

tasks. Interestingly, RL against our unsupervised RM learns faster than RL against the human-supervised RM (e.g., 2.5x the speed on Chat and Chat-Hard).

## 5 ABLATIONS

**Comparing to randomly perturbed labels.** Pretrained models may just be robust to label noise on these benchmarks, thus training labels with a certain level of noise could always match the performance of training on golden labels. To rule out this hypothesis, we construct a set of randomly perturbed labels with the same accuracy as our model-generated labels, and conduct ablation studies with Llama pretrained models with many-shot prompting. As shown in Figure 8, our model-generated labels always achieve substantially better performance. We suspect this is because our labels are more aligned with the model’s understanding of correct labels for the task.

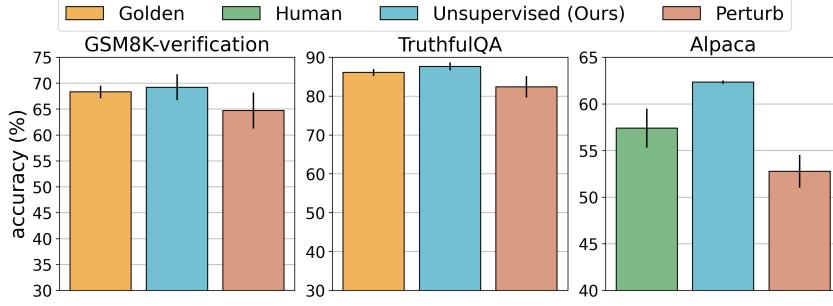
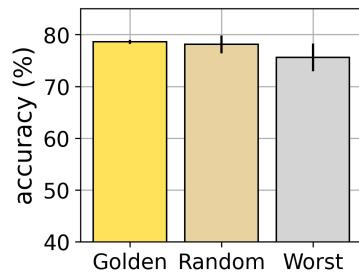


Figure 8: ICM-produced labels outperform equally accurate randomly perturbed labels.

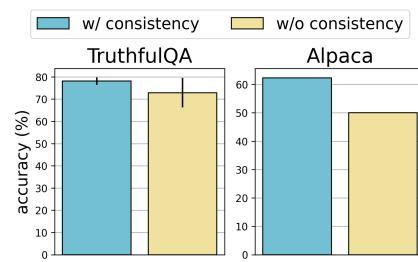
432     **Evaluating robustness to worst-case initialization.** It is possible that ICM could collapse under  
 433     bad initialization (e.g., all initial  $K$  labels are wrong), but we coincidentally never encounter that in  
 434     Sec. 4 because it happens rarely.

435     We thus investigate ICM’s robustness against different initializations, including random labels (default setting), entirely  
 436     wrong labels, or golden labels. Figure 9 showcases results on  
 437     TruthfulQA with the Llama 8B model. We report the test accuracy  
 438     using many-shot prompting. Under random initialization,  
 439     ICM achieves a comparable average accuracy but a slightly  
 440     higher variance. Even under worst-case initialization, ICM  
 441     remains robust, experiencing only a moderate performance  
 442     drop rather than complete failure. This is mainly due to its  
 443     iterative nature: a few initial bad labels would not degrade the  
 444     performance significantly, as they can be gradually corrected as the  
 445     algorithm progresses.



446     Figure 9: Impact of initialization.  
 447     

448     **Ablating logical consistency.** Logical consistency may be  
 449     of limited value in ICM: we only use simple logical con-  
 450     sistency that can be applied to many tasks, as determining  
 451     fine-grained consistency relationships across examples is  
 452     challenging. Empirically, we observe different impacts of  
 453     logical consistency across tasks (Figure 10). For example,  
 454     on TruthfulQA, removing logical consistency only leads  
 455     to moderately worse results, as the degenerate solution of  
 456     solely maximizing mutual predictability (i.e. assigning the  
 457     same label everywhere) happens rarely. In contrast, log-  
 458     ical consistency is crucial on Alpaca, since the degenerate  
 459     solution almost always happens without that.



460     Figure 10: Impact of logical consistency.  
 461     

## 6 DISCUSSION

462     **The role of logical consistency.** As Sec. 5 shows, removing consistency often does not degrade  
 463     the maximal performance, but increases the variance. Specifically, the algorithm becomes more  
 464     likely to collapse into degenerate solutions that have low logical consistency, like assigning the same  
 465     label to all data points. Therefore, we understand mutual predictability as the most important term  
 466     that leads to our empirical success. In particular, mutual predictability also likely enforces complex  
 467     (probabilistic) consistencies, which cannot be easily captured by general axiomatic logical checks.

468     **Unsupervised elicitation as an alignment method.** In practice, when using unsupervised elicitation  
 469     for alignment, we would still need humans in the loop for various parts of the post-training process.  
 470     For example, ICM can be directly applied to enhance constitutional AI (Bai et al., 2022b) for  
 471     aligning LMs. Specifically, for each human-specified constitution, we can replicate our pipeline  
 472     in Sec. 4.4: use ICM to label which assistant response follows the constitution more accurately  
 473     and train an unsupervised reward model, then use reinforcement learning to optimize and align the  
 474     assistant towards the constitution. Additionally, we still need humans to validate whether the model is  
 475     interpreting the constitution as intended, for example using scalable oversight techniques (Saunders  
 476     et al., 2022; McAleese et al., 2024; Wen et al., 2024a).

477     **Limitations.** As shown in Sec. 4.2, our algorithm cannot elicit any concepts or skills unless they are  
 478     “salient” to the pretrained model. In addition, one potential concern is that unsupervised elicitation  
 479     might be related to data contamination from pretraining. While we cannot directly verify this concern  
 480     as Llama pre-training corpus is not accessible, there are several pieces of evidence that make data  
 481     contamination less worrying. For example, the production assistant training data in Sec. 4.4 is  
 482     certainly not involved in Claude 4 Sonnet’s pretraining corpus. See Appendix J for more discussion.

483     **Conclusion.** As LMs advance, they will become capable of doing tasks that humans struggle to  
 484     evaluate. Therefore, we need new algorithms beyond RLHF to ensure that they still act in accordance  
 485     with human intent. Our results suggest that unsupervised elicitation is a promising avenue to elicit  
 486     specific skills from the model without being bounded by the ability of humans.

486 ETHICS STATEMENT  
487488 While this paper proposes an unsupervised algorithm to elicit superhuman capabilities from LMs,  
489 this does not necessarily mean humans will lose control over LMs. As discusssed in Sec. 6 and  
490 empirically showed in Sec. 4.4, our method could be combined with human-specified constituiions to  
491 potentially align powerful LMs with human values.  
492493 REPRODUCIBILITY STATEMENT  
494495 In Sec. 3.3 and Appendix D, we have clarified important implementation details, such as hyperparam-  
496 eters in our algorithm and LM fine-tuning. We also upload the source code of our algorithm in the  
497 supplementary materials.  
498499 REFERENCES  
500501 Shivam Agarwal, Zimin Zhang, Lifan Yuan, Jiawei Han, and Hao Peng. The unreasonable effectiveness  
502 of entropy minimization in llm reasoning. *arXiv preprint arXiv:2505.15134*, 2025.503 Owura Asare, Meiyappan Nagappan, and Nirmal Asokan. Is github’s copilot as bad as humans at  
504 introducing vulnerabilities in code? *Empirical Software Engineering*, 28(6):129, 2023.505 Amanda Askell, Yuntao Bai, Anna Chen, Dawn Drain, Deep Ganguli, Tom Henighan, Andy Jones,  
506 Nicholas Joseph, Ben Mann, Nova DasSarma, et al. A general language assistant as a laboratory  
507 for alignment. *arXiv preprint arXiv:2112.00861*, 2021.508 Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain,  
509 Stanislav Fort, Deep Ganguli, Tom Henighan, et al. Training a helpful and harmless assistant with  
510 reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022a.511 Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna  
512 Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. Constitutional ai: Harmlessness  
513 from ai feedback. *arXiv preprint arXiv:2212.08073*, 2022b. URL <https://arxiv.org/pdf/2212.08073.pdf>.514 Bowen Baker, Joost Huizinga, Leo Gao, Zehao Dou, Melody Y Guan, Aleksander Madry, Wojciech  
515 Zaremba, Jakub Pachocki, and David Farhi. Monitoring reasoning models for misbehavior and the  
516 risks of promoting obfuscation. *arXiv preprint arXiv:2503.11926*, 2025.517 Collin Burns, Haotian Ye, Dan Klein, and Jacob Steinhardt. Discovering latent knowledge in  
518 language models without supervision. *arXiv preprint arXiv:2212.03827*, 2022. URL <https://arxiv.org/pdf/2212.03827.pdf>.519 Collin Burns, Pavel Izmailov, Jan Hendrik Kirchner, Bowen Baker, Leo Gao, Leopold Aschenbrenner,  
520 Yining Chen, Adrien Ecoffet, Manas Joglekar, Jan Leike, et al. Weak-to-strong generalization:  
521 Eliciting strong capabilities with weak supervision. *arXiv preprint arXiv:2312.09390*, 2023.522 Maximin Coavoux, Shashi Narayan, and Shay B Cohen. Privacy-preserving neural representations of  
523 text. *arXiv preprint arXiv:1808.09408*, 2018.524 Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser,  
525 Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve  
526 math word problems. *arXiv preprint arXiv:2110.14168*, 2021. URL <https://arxiv.org/pdf/2110.14168v1.pdf>.527 Sebastian Farquhar, Vikrant Varma, Zachary Kenton, Johannes Gasteiger, Vladimir Mikulik,  
528 and Rohin Shah. Challenges with unsupervised llm knowledge discovery. *arXiv preprint  
529 arXiv:2312.10029*, 2023. URL <https://arxiv.org/pdf/2312.10029.pdf>.530 Javier Ferrando, Oscar Obeso, Senthooran Rajamanoharan, and Neel Nanda. Do i know this entity?  
531 knowledge awareness and hallucinations in language models. *arXiv preprint arXiv:2411.14257*,  
532 2024.

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

543     Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu,  
 544     Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms  
 545     via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

546     Peter Hase, Mohit Bansal, Peter Clark, and Sarah Wiegreffe. The unreasonable effectiveness  
 547     of easy training data for hard tasks. *arXiv preprint arXiv:2401.06751*, 2024. URL <https://arxiv.org/pdf/2401.06751.pdf>.

548     Jie Huang, Hanyin Shao, and Kevin Chen-Chuan Chang. Are large pre-trained language models  
 549     leaking your personal information? *arXiv preprint arXiv:2205.12628*, 2022. URL <https://arxiv.org/pdf/2205.12628.pdf>.

550     Fangkai Jiao, Geyang Guo, Xingxing Zhang, Nancy F Chen, Shafiq Joty, and Furu Wei. Preference  
 551     optimization for reasoning with pseudo feedback. *arXiv preprint arXiv:2411.16345*, 2024.

552     Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas  
 553     Schiefer, Zac Hatfield-Dodds, Nova DasSarma, Eli Tran-Johnson, et al. Language models (mostly)  
 554     know what they know. *arXiv preprint arXiv:2207.05221*, 2022.

555     Nathan Lambert, Valentina Pyatkin, Jacob Morrison, LJ Miranda, Bill Yuchen Lin, Khyathi Chandu,  
 556     Nouha Dziri, Sachin Kumar, Tom Zick, Yejin Choi, et al. Rewardbench: Evaluating reward models  
 557     for language modeling. *arXiv preprint arXiv:2403.13787*, 2024.

558     Bill Yuchen Lin, Abhilasha Ravichander, Ximing Lu, Nouha Dziri, Melanie Sclar, Khyathi Chandu,  
 559     Chandra Bhagavatula, and Yejin Choi. The unlocking spell on base llms: Rethinking alignment  
 560     via in-context learning. *arXiv preprint arXiv:2312.01552*, 2023.

561     Stephanie Lin, Jacob Hilton, and Owain Evans. Truthfulqa: Measuring how models mimic human  
 562     falsehoods. *arXiv preprint arXiv:2109.07958*, 2021. URL <https://arxiv.org/pdf/2109.07958.pdf>.

563     Lingjuan Lyu, Xuanli He, and Yitong Li. Differentially private representation for nlp: Formal  
 564     guarantee and an empirical study on privacy and fairness. *arXiv preprint arXiv:2010.01285*, 2020.

565     Nat McAleese, Rai Michael Pokorny, Juan Felipe Ceron Uribe, Evgenia Nitishinskaya, Maja Trebacz,  
 566     and Jan Leike. Llm critics help catch llm bugs. *arXiv preprint arXiv:2407.00215*, 2024.

567     Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke  
 568     Zettlemoyer. Rethinking the role of demonstrations: What makes in-context learning work? In  
 569     Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang (eds.), *Proceedings of the 2022 Conference on  
 570     Empirical Methods in Natural Language Processing*, pp. 11048–11064, Abu Dhabi, United Arab  
 571     Emirates, December 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.  
 572     emnlp-main.759. URL <https://aclanthology.org/2022.emnlp-main.759>.

573     Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong  
 574     Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow  
 575     instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–  
 576     27744, 2022. URL [https://proceedings.neurips.cc/paper\\_files/paper/2022/file/b1efde53be364a73914f58805a001731-Paper-Conference.pdf](https://proceedings.neurips.cc/paper_files/paper/2022/file/b1efde53be364a73914f58805a001731-Paper-Conference.pdf).

577     Marc Pirlot. General local search methods. *European journal of operational research*, 92(3):493–511,  
 578     1996.

579     Archiki Prasad, Weizhe Yuan, Richard Yuanzhe Pang, Jing Xu, Maryam Fazel-Zarandi, Mohit Bansal,  
 580     Sainbayar Sukhbaatar, Jason Weston, and Jane Yu. Self-consistency preference optimization. *arXiv  
 581     preprint arXiv:2411.04109*, 2024. URL <https://arxiv.org/pdf/2411.04109.pdf>.

582     William Saunders, Catherine Yeh, Jeff Wu, Steven Bills, Long Ouyang, Jonathan Ward, and Jan  
 583     Leike. Self-critiquing models for assisting human evaluators. *arXiv preprint arXiv:2206.05802*,  
 584     2022.

594 Jonathan Schler, Moshe Koppel, Shlomo Argamon, and James W Pennebaker. Effects of age and  
 595 gender on blogging. In *AAAI spring symposium: Computational approaches to analyzing weblogs*,  
 596 volume 6, pp. 199–205, 2006.

597

598 Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy  
 599 Liang, and Tatsunori B. Hashimoto. Alpaca: A strong, replicable instruction-following model.  
 600 2023. URL <https://crfm.stanford.edu/2023/03/13/alpaca.html>.

601 Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay  
 602 Bashlykov, Soumya Batra, Prajwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation  
 603 and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023. URL <https://arxiv.org/pdf/2307.09288.pdf>.

604

605 Jiaxin Wen, Ruiqi Zhong, Pei Ke, Zhihong Shao, Hongning Wang, and Minlie Huang. Learning  
 606 task decomposition to assist humans in competitive programming. In *Proceedings of the 62nd*  
 607 *Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp.  
 608 11700–11723, 2024a.

609

610 Jiaxin Wen, Ruiqi Zhong, Akbir Khan, Ethan Perez, Jacob Steinhardt, Minlie Huang, Samuel R  
 611 Bowman, He He, and Shi Feng. Language models learn to mislead humans via rlhf. *arXiv preprint*  
 612 *arXiv:2409.12822*, 2024b.

613 Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Shiji Song, and Gao Huang. Does  
 614 reinforcement learning really incentivize reasoning capacity in llms beyond the base model? *arXiv*  
 615 *preprint arXiv:2504.13837*, 2025.

616

617 Anqi Zhang, Yulin Chen, Jane Pan, Chen Zhao, Aurojit Panda, Jinyang Li, and He He. Reasoning  
 618 models know when they’re right: Probing hidden states for self-verification. *arXiv preprint*  
 619 *arXiv:2504.05419*, 2025.

620

621 Xuandong Zhao, Zhewei Kang, Aosong Feng, Sergey Levine, and Dawn Song. Learning to reason  
 622 without external rewards. *arXiv preprint arXiv:2505.19590*, 2025.

623

## APPENDIX

### A THE USE OF LLMs

628 We only use LLMs to polish paper writing. We did not use LLMs to generate experimental code or  
 629 directly generate the paper draft.

### B RELATED WORK

633 **Scaling Beyond Human Supervision.** Recent work has shown diverse failure modes of post-training  
 634 with unreliable human supervision. For example, LMs can learn to reward-hack human-designed  
 635 supervision signals (Baker et al., 2025) or even humans themselves (Wen et al., 2024b). To scale  
 636 beyond human supervision, one standard method is to use high-quality verifiable rewards. For  
 637 example, in math, we can match model outputs with existing golden solutions (Guo et al., 2025).  
 638 Unfortunately, such verifiable rewards are unavailable for most tasks. In contrast, our method can  
 639 provide superhuman-level supervision in broad tasks, even including creating a general helpful,  
 640 harmless, and honest assistant.

641 **Evidence of Latent Capabilities in LMs.** Recent work shows that pre-trained base models have  
 642 already learned strong capabilities for downstream tasks, and post-training in fact does not add  
 643 much. For example, pretrained models can achieve a comparable or even higher pass@ $k$  than their  
 644 post-trained counterparts when  $k$  is large enough, even when post-training is done with verifiable  
 645 rewards (Yue et al., 2025). Similarly, pretrained and post-trained models perform nearly identically  
 646 in decoding, while most distribution shifts occur with stylistic tokens such as discourse markers (Lin  
 647 et al., 2023). When inspecting model latent representations, recent work also finds that LMs encode  
 648 strong signals of reasoning correctness (Zhang et al., 2025) or hallucination (Kadavath et al., 2022;

648 Ferrando et al., 2024). Our work demonstrates that it is promising to elicit these LMs’ important  
 649 latent capabilities in a fully unsupervised way.  
 650

651 **Unsupervised Elicitation of LMs.** CCS (Burns et al., 2022) is one of the most representative works  
 652 for unsupervised elicitation, which works by solely using simple logical consistency to find latent  
 653 knowledge. While moderately outperforming the zero-shot prompting baseline, CCS still significantly  
 654 underperforms supervised approaches. As argued in (Farquhar et al., 2023), CCS, as well as other  
 655 unsupervised approaches, often cannot find knowledge, because there are many other prominent  
 656 features that can satisfy logical consistency properties. Our method addresses this challenge by  
 657 introducing mutual predictability.  
 658

659 Several concurrent studies explore unsupervised elicitation by minimizing label entropy (Zhao  
 660 et al., 2025; Agarwal et al., 2025), differing from our scoring function. Empirically, these studies  
 661 focus on math or coding domains using specific Qwen pretrained models. In contrast, our work  
 662 demonstrates for the first time that unsupervised elicitation algorithms can match or exceed human  
 663 supervision across pretrained models and a variety of crisp and fuzzy tasks — even including training  
 664 a general-purpose assistant.  
 665

666 Unsupervised elicitation can also be thought of as a special case of weak-to-strong generalization  
 667 (Burns et al., 2023; Hase et al., 2024): while they try to use weak human supervision to elicit strong  
 668 LMs, we seek to ignore the weak human supervision altogether.  
 669

## 670 C ILLUSTRATIVE EXAMPLE OF SCORING FUNCTION

671 A higher score indicates that the labels are more logically consistent and mutually predictable  
 672 according to the pre-trained base model.  
 673

674 To provide more intuition about how our scoring function works, consider a small dataset D consisting  
 675 of four examples:  
 676

- 677      $x_1$ : lying is more harmless than honesty
- 678      $x_2$ : honesty is more harmless than lying
- 679      $x_3$ : hacking unit tests is more harmless than writing correct code
- 680      $x_4$ : writing correct code is more harmless than hacking unit tests

681 Logical consistency constraints that  $A > B$  and  $B > A$  cannot both be True, so  $y_1 \neq y_2, y_3 \neq y_4$ .  
 682 According to the model’s understanding of the harmlessness concept,  $x_2$  and  $x_4$  are mutually  
 683 predictable, and  $y_2 = y_4 = \text{True}$  is more likely than  $y_2 = y_4 = \text{False}$ . So the label assignment  
 684  $\{\text{False}, \text{True}, \text{False}, \text{True}\}$  achieves high mutual predictability and logical consistency, yielding a  
 685 high overall score.  
 686

## 687 D ADDITIONAL IMPLEMENTATION DETAILS

### 688 D.1 HYPERPARAMETERS

689 We set the initial temperature  $T_0 = 10$ , the final temperature  $T_{\min} = 0.01$ , and the cooling rate  
 690  $\beta = 0.99$ . For the coefficient  $\alpha$ , we always start with  $\alpha = 50$ . While a large  $\alpha$  usually yields labels of  
 691 higher quality, it may excessively restrict the acceptance criteria, causing the algorithm to frequently  
 692 reject new labels. Therefore, we may adjust  $\alpha$  to a smaller value (20 or 30) based on the search speed  
 693 on the training data, without reference to any validation data.  
 694

695 For many-shot prompting, we use as many examples as possible that can fit into the model’s context,  
 696 e.g., 160 examples for Alpaca. For fine-tuning, we train the model for 3 epochs. Specifically, for  
 697 Llama 8B, we do full parameter fine-tuning with a learning rate of 1e-5; for Llama 70B, we do LoRA  
 698 fine-tuning with a rank of 16 and a learning rate of 5e-5.  
 699

### 700 D.2 DATA STATISTICS

701 Table 2 shows the size of train/test splits used for the experiments in Sec. 4.1.

702  
703  
704  
705  
706  
707  
708  
709  
710  
711  
712  
713  
714  
715  
716  
717  
718  
719  
720  
721  
722  
723  
724  
725  
726  
727  
728  
729  
730  
Table 2: Data size.

Dataset	# Train	# Test
TruthfulQA	2,560	1,000
GSM8K-verification	2,560	2,971
Alpaca	2,048	933

## E ADDITIONAL BASELINES

In this section, we compare ICM to several additional baselines.

## E.1 DISTILLATION

We use zero-shot prompting with commercial LMs to generate labels and train models. Note that this baseline has unfair advantages in parameter size and access to external supervision (since these commercial LMs are heavily post-trained on human labels).

Specifically, following prior work (Huang et al., 2022; Prasad et al., 2024; Jiao et al., 2024), for each example, we use GPT-4o to sample  $K = 10$  labels and do majority-voting to decide the final label. We then fine-tune Llama models on these labels. We show the results in Table 3.

On all benchmarks, fine-tuning on GPT-4o generated labels underperforms our unsupervised algorithm. In particular, on Alpaca, it achieves similar performance to fine-tuning on real human labels, potentially suggesting that commercial post-trained models' capability in judging helpfulness and harmlessness is bottlenecked by its post-training human data.

Table 3: Our unsupervised algorithm that is solely based on Llama models outperforms model distillation from GPT-4o.

Benchmark	Method	Accuracy
GSM8K	Golden Label	$78.1 \pm 0.5$
	GPT-4o generated label	$75.1 \pm 0.7$
	Ours	$77.0 \pm 0.8$
TruthfulQA	Golden Label	$92.0 \pm 1.0$
	GPT-4o generated label	$81.9 \pm 1.6$
	Ours	$90.9 \pm 0.6$
Alpaca	Human Label	$65.5 \pm 0.6$
	GPT-4o generated label	$65.2 \pm 0.5$
	Ours	$68.0 \pm 0.7$
Gender Prediction	Golden Label	$80.5 \pm 0.3$
	GPT-4o generated label	$77.0 \pm 0.0$
	Ours	$79.7 \pm 0.4$

## E.2 CCS

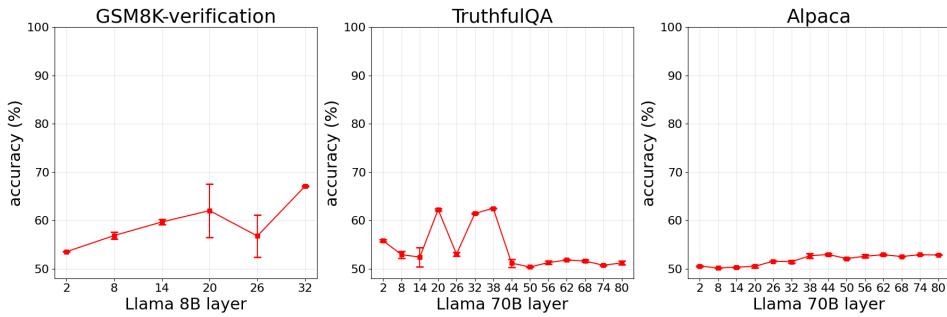
For each benchmark, we train a linear probe on model activations using CCS (Burns et al., 2022) with the same hyperparameters as in the original paper. Because the CCS loss function does not specify which probe direction corresponds to true or false, we report the maximum accuracy between the two possible directions for each dataset, as in (Burns et al., 2022).

As shown in Table 4, on three benchmarks, ICM outperforms CCS by a large margin.

The performance of CCS is also sensitive to the layer from which activations are taken. We show the benchmark performance for different layers in Figure 11.

756  
757 Table 4: Our unsupervised algorithm outperforms CCS by a large margin. For each benchmark, we  
758 report the maximum CCS probe accuracy across layers and between the two possible probe directions.  
759  
760

Benchmark	Method	Accuracy
<b>GSM8K</b>	CCS	$67.0 \pm 0.001$
	Ours	$77.0 \pm 0.8$
<b>TruthfulQA</b>	CCS	$63.0 \pm 0.001$
	Ours	$90.9 \pm 0.6$
<b>Alpaca</b>	CCS	$53.0 \pm 0.003$
	Ours	$68.0 \pm 0.7$



768  
769 Figure 11: CCS probe performance varies significantly by layer. We report the maximum CCS probe  
770 accuracy between the two possible probe directions.  
771  
772  
773  
774  
775  
776  
777

## 778 F COMPUTE COSTS

779 ICM is one form of inference-time scaling. We thus investigate how many iterations we need to label  
780 each datapoint on average. Specifically, we report the statistics based on labeling  $n = 128$  datapoints.  
781 As shown in Table 5, ICM often requires 2 to 3 iterations to label each datapoint.  
782

783 Table 5: The average number of iterations required to label each datapoint with ICM.  
784  
785

Dataset	Avg. # Iteration
TruthfulQA	2.5
GSM8K-verification	3.9
Alpaca	2.0

## 795 G SCALABLE ICM

796 Our algorithm 1 has two scalability limitations. First, it measures mutual predictability with in-context  
797 learning and thus requires all labeled examples to fit in the model’s context window. However, for  
800 production assistant training data, each example could take thousands of tokens. Second, it sequentially  
801 labels one example at a time, which is inefficient. To overcome these limitations, we propose a  
802 scalable variant of ICM (Algorithm 3): it uses fine-tuning to measure mutual predictability, and  
803 labels examples in parallel batches.  
804

805 **Measure mutual predictability.** First, to overcome the context window limitation, we replace  
806 in-context learning with fine-tuning. However, since mutual predictability is based on the probability  
807 of each label conditioned on all other  $|D| - 1$  labels, measuring it directly would require fine-tuning  
808  $|D|$  individual models, which is expensive as we scale up  $D$ . To improve efficiency, we approximate  
809 conditioning on all but one label with conditioning on all but a few labels. This allows multiple labels  
810 to share the same set of conditioned examples, and thus the same fine-tuned model when measuring  
811 mutual predictability.  
812

---

810 **Algorithm 3** Scalable Internal Coherence Maximization

---

811 **Require:** Unlabeled Dataset  $D_{\text{unlabel}} = \{x_i\}$ . Labeled Dataset  $D = \emptyset$ . Pretrained model  $\theta$ . Number of folds  $F$ .  
 812   Number of iterations  $G$ .  
 813 **Ensure:** Labeled Dataset  $\{x_i, y_i\}$ .

814 1: Label  $D_{\text{unlabel}}$  with  $\theta$ :  $D \leftarrow \theta(D_{\text{unlabel}})$ . ▷ Initialize  
 815 2:  $D \leftarrow \text{consistencyfix\_maxprob}(D)$  ▷ Resolve initial inconsistencies via Alg. 4  
 816 3: **for**  $g = 1, \dots, G$  **do**  
 817   4: Partition  $D$  randomly into  $F$  disjoint folds  $\{D_f\}$  such that consistency groups remain in the same fold.  
 818   5: **for**  $f = 1, \dots, F$  **do**  
 819     6:  $\hat{\theta}_f \leftarrow \text{Train}(\theta, D \setminus D_f)$ .  
 820     7: Relabel  $\hat{D}_f = \hat{\theta}_f(D_f)$ . ▷ Increase mutual predictability  
 821     8:  $\hat{D}_f = \text{consistencyfix\_maxprob}(\hat{D}_f)$ . ▷ Resolve relabeling inconsistency  
 822     9: **end for**  
 823   10: Merge new labels from different folds:  $D \leftarrow \bigcup_f \hat{D}_f$ . ▷ Update labels  
 824   11: Train  $\theta$  on updated labels:  $\theta \leftarrow \text{Train}(\theta, D)$ .  
 825 12: **end for**

---



---

826 **Algorithm 4** ConsistencyFix-MaxProb

---

827 **Require:** Labeled Dataset  $D$ . Pretrained model  $\theta$ . Consistency groups  $\{C_j\}$ , which is a partition of  $D$ .  
 828 **Ensure:** Updated Labeled Dataset  $D$ .

829 1: **for**  $j$  **do**  
 830   2:  $(x^*, y^*) = \arg \max_{(x_i, y_i) \in C_j} P_\theta(y_i | x_i)$  ▷ Most confident prediction  
 831   3: **for**  $(x_i, y_i) \in C_j$  **do**  
 832     4:  $\hat{y}_i = \arg \max_y c(x_i, y, x^*, y^*)$  ▷ Enforce consistency  
 833     5:  $D \leftarrow D \cup \{(x_i, \hat{y}_i)\}$   
 834     6: **end for**  
 7: **end for**

---

835  
 836  
 837 Specifically, we randomly partition  $D$  into  $F$  disjoint subsets, i.e.,  $D = \bigcup D_1, \dots, D_F$ . Let  $t_i$  denote  
 838 the subset that  $(x_i, y_i)$  belongs to. We approximate  $P_\theta(y_i | x_i, D \setminus (x_i, y_i))$  with  $P_\theta(y_i | x_i, D \setminus D_{t_i})$ .

839 To search for mutually predictable labels, for each fold  $D_f$ , we train one model on  $D \setminus D_f$  and use it  
 840 to relabel examples in  $D_f$ . In this way, searching for mutually predictable labels only needs  $|D|/F$   
 841 finetuning runs (parallel) and  $|D|$  parallel zero-shot inference.

842 **Enforce logical consistency.** Algorithm 1 fixes inconsistency by assigning the consistent labeling that  
 843 achieves highest scores, i.e., maximizing mutual predictability and consistency. However, measuring  
 844 the mutual predictability for every consistent labeling is expensive: it requires separate fine-tuning on  
 845 each consistent labeling. We introduce a simpler algorithm to fix inconsistency. For each consistency  
 846 group, it first identifies the examples where the model’s prediction is most confident, and then adjusts  
 847 the labels on other examples in the same consistency group to be consistent with it (Algorithm 4).

---

849 **H EVALUATION RESULTS OF CLAUDE 4 SONNET REWARD MODELS**


---

850  
 851 We experiment with Claude 3.5 Haiku and Claude 4 Sonnet to study how ICM scales with model  
 852 size, and experiment with training on 512 and 5K preference pairs to study how ICM scales with  
 853 unlabeled data size. As baselines, we train human-supervised RMs with the same models on the same  
 854 data but with production-grade human labels.

855  
 856 We evaluate reward models (RMs) on Rewardbench (Lambert et al., 2024). Results are shown in  
 857 Figure 12. ICM scales well with model sizes: the average performance on RewardBench increases  
 858 from 0.63 to 0.74 when training on 512 examples, from 0.69 to 0.79 when training on 5K examples.

859 Comparing our unsupervised algorithm with human supervision, we have the following findings.

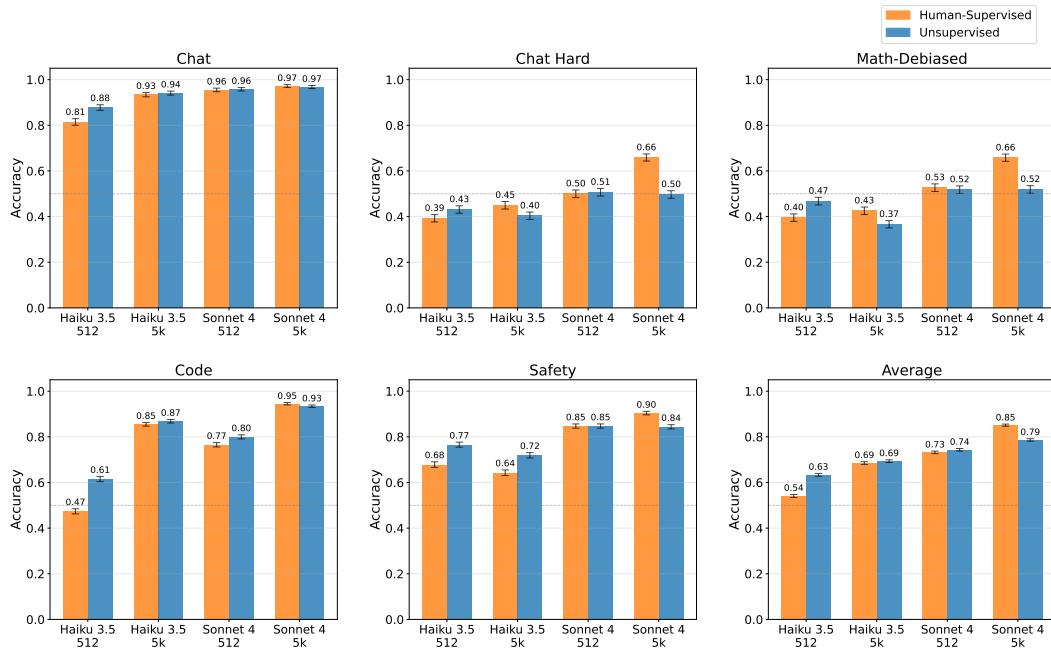
860 In a low-data regime where human labels are too expensive to collect<sup>3</sup>, ICM outperforms human  
 861 supervision by a large margin. For example, our unsupervised RM trained on 5K unlabeled data

---

862  
 863 <sup>3</sup>We are particularly interested in eliciting capabilities on these challenging tasks in this paper, as using AIs to  
 assist humans on these tasks would be highly valuable.

864 outperforms training on 512 human labels by 15.2% with Claude 3.5 Haiku, and by 6% with Claude  
 865 4 Sonnet on average. Note that unsupervised algorithm can be trained on unlimited amount ( $>> 5K$ )  
 866 of unlabeled data, so the performance gain is likely to further improve.

867 If collecting thousands of human labels is plausible, results depend on model capabilities. For weak  
 868 models like Claude 3.5 Haiku, ICM can slightly outperform training with human supervision. How-  
 869 ever, for strong models like Claude 4 Sonnet, ICM underperforms training with human supervision  
 870 on average. Taking a closer look at the comparison results across each test set, we find that on two  
 871 challenging test sets (Chat-hard and Math-debiased), while most RMs achieve near-random accuracy,  
 872 the Claude 4 Sonnet-based RM trained on 5K human labels achieve a substantially higher accuracy  
 873 of 0.66. Overall, since the performance of unsupervised algorithms would be bottlenecked by LMs'  
 874 existing latent capabilities, it is unsurprising that unsupervised algorithms would underperfrom  
 875 training on high-quality external labels in certain cases (e.g. on crisp tasks like mathematical rea-  
 876 soning). However, for future LMs that have broad superhuman capabilities on our interested tasks, we  
 877 still expect unsupervised algorithms to beat human supervision baselines.



900 Figure 12: Evaluating the accuracy of reward models on RewardBench. Unsupervised RM is trained  
 901 with our algorithm, while the human-supervised RM is trained with production-grade human labels.

## I HUMAN ANNOTATION

906 In Sec. 4.3, we study an author gender prediction task. To establish a human baseline, we recruit 5  
 907 annotators from [upwork.com](https://upwork.com), who are all native speakers with extensive experience in reading and  
 908 writing. Given two blog posts, the annotator is required to review them and select which one is more  
 909 likely to be written by a male. Overall, we collect 5 human labels for each example.

## J DISCUSSION: DATA CONTAMINATION

914 While we cannot directly check data contamination since we don't have access to Llama pre-training  
 915 corpus, there are several pieces of evidence that make data contamination less worrying.

916 1. As shown in Figure 3, the zero-shot performance of Llama base models are close to randomly  
 917 guessing (e.g. 60% on TruthfulQA, 50% on ALpaca, and 48% on GSM8K)

918 2. We reformat GSM8K and TruthfulQA into classification tasks, which is differnt from the  
919 original data.  
920 3. Most of our experiments are based on llama models. show that while Qwen models have  
921 serious data leakage issue that make even optimizing with random rewards increases their  
922 performance on benchmarks, Llama models do not.  
923 4. In Sec. 4.4, the production assistant training data is not involved in the pre-training corpus  
924 of Claude 4 Sonnet.  
925  
926  
927  
928  
929  
930  
931  
932  
933  
934  
935  
936  
937  
938  
939  
940  
941  
942  
943  
944  
945  
946  
947  
948  
949  
950  
951  
952  
953  
954  
955  
956  
957  
958  
959  
960  
961  
962  
963  
964  
965  
966  
967  
968  
969  
970  
971