One-shot Entropy Minimization

Anonymous ACL submission

Abstract

We trained 13,440 large language models and found that entropy minimization requires only a single unlabeled data and 10 steps optimization to achieve performance improvements greater than those obtained using thousands of data and carefully designed rewards in rule-based reinforcement learning.

1 Introduction

The post-training phase of large language models (LLMs) has advanced rapidly (Xu et al., 2025a; Xie et al., 2025; Hu et al., 2025; Wang et al., 2025; Cui et al., 2025; Zeng et al., 2025), with models like DeepSeek-R1 (DeepSeek-AI et al., 2025), Kimi-K1.5 (Team et al., 2025), and OpenAI o-series (OpenAI, 2024, 2025) demonstrating remarkable reasoning abilities. However, preparing for Reinforcement Learning (RL) is never an easy task. It often requires a large amount of high-quality ground truth labeled data, along with the careful design of rule-based rewards to maximize advantage signals and prevent reward hacking. In contrast, Entropy Minimization (EM) is entirely unsupervised. We used the EM method to train 13,440 large language models in order to eliminate randomness in training as much as possible and ensure that the experimental results and observed patterns are reliable. Our rigorous study demonstrates that using just a single piece of unlabeled data, the performance already surpasses that of traditional RL. Moreover, they typically converge within just 10 training steps, which is significantly faster than the thousands of steps often required for RL. EM is based on two direct and simple assumptions: (1) The sampling process in generation of large language models is inherently stochastic, (2) Correct answers generally have lower entropy than incorrect ones. Our study reveals that EM and RL share the same goal: unlocking the pretrained model's latent potential without adding new knowledge(Liu et al., 2025). Both rely on a process we call "token reranking" to maximize the model's performance. We find that entropy minimization has the capacity to rival RL in the post-training phase.

Our main contributions are as follows:

- We propose ***One-shot Entropy Minimization***, a surprisingly powerful and fully unsupervised method that rivals or surpasses reinforcement learning using just a single unlabeled data.
- We conduct an in-depth analysis of the effectiveness of One-shot EM, making it a highly reasonable approach. We find that it shares many core properties with RL, yet drives model behavior in the opposite direction when viewed through the lens of ***logits shift***.
- We extensively evaluate EM and identify temperature as a key factor for both training and inference, and EM shows an opposite trend to RL with respect to inference-time temperature.
- We reveal that EM is a distribution shaping tool rather than a learning method by analyzing logit shift in Section 3.3 effect and the inconsistency in loss-reasoning curve over confidence in Section 3.4 of EM.

2 Method

2.1 Entropy Minimization Algorithm

Let $\mathcal{V}$ denote the vocabulary of a pretrained autoregressive language model p_θ, parameterized by θ. Given an input prompt x (e.g., a question or problem description), the model autoregressively generates a response sequence $y = (y_1, y_2, \ldots, y_T)$ according to its current policy:

$$p_\theta(y \mid x) = \prod_{t=1}^{T} p_\theta(y_t \mid y_{<t}, x),$$

where T is the length of the generated sequence.

Our core idea is to reduce the model's uncertainty over its own predictions by minimizing the token-level entropy at each generation step. The conditional entropy at time step t is defined as:

$$H_t = -\sum_{v \in \mathcal{V}} p_\theta(v \mid y_{<t}, x) \log p_\theta(v \mid y_{<t}, x).$$

To avoid penalizing the prompt portion, we compute entropy only over the generated tokens. Let T_{prompt} denote the number of tokens in the prompt x. Then the set of target positions is:

$$\mathcal{I} = \{t \mid t > T_{\text{prompt}},\ y_t \neq \texttt{[PAD]}\}.$$

The overall EM loss for a single input x is given by:

$$\mathcal{L}_{\text{EM}}(x; \theta) = \frac{1}{|\mathcal{I}|} \sum_{t \in \mathcal{I}} H_t.$$

This loss encourages the model to become more confident in its own predictions without relying on external supervision.Entropy Minimization loss is fully differentiable with respect to model parameters, with gradients resembling the score-function estimator found in entropy-regularized reinforcement learning. What's more, Entropy Minimization offers a closed-form objective, eliminating the need for external reward estimation or value baselines, thereby simplifying optimization while retaining the effectiveness of entropy-driven exploration and exploitation.

2.2 Data Selection

Entropy minimization (EM) relies on the premise that the model's predictive uncertainty can serve as a meaningful training signal. However, not all input prompts are equally informative in this regard. As noted in prior work (Razin et al., 2025; Wang et al., 2025), certain prompts elicit deterministic behavior from the model (e.g., always correct or always incorrect), yielding limited gradient information under entropy-based objectives.

To address this, we adopt a variance-based data selection strategy. Specifically, we measure the variance of the model's *pass@k* accuracy across multiple samples and select prompts for which the model exhibits the highest behavioral variance. This targets inputs on the cusp of the model's capability—neither trivial nor impossible—making them ideal for entropy-driven optimization.

Given a prompt x, we draw k independent samples from the model:

$$\mathcal{Y}^{(x)} = \left\{y^{(1)}, y^{(2)}, \ldots, y^{(k)}\right\}, \quad y^{(i)} \sim p_\theta(\cdot \mid x).$$

We then compute the *pass@k* score as:

$$\text{pass@k}(x) = \frac{1}{k} \sum_{i=1}^{k} I\left[y^{(i)} \text{ is correct}\right],$$

where $I[\cdot]$ is the indicator function for whether a sample is considered correct (via execution or string match).

We further compute the sample variance of this binary success variable:

$$\text{Var}_{\text{pass@k}}(x) = \frac{1}{k} \sum_{i=1}^{k} \left(I\left[y^{(i)} \text{ is correct}\right] - \text{pass@k}(x)\right)^2.$$

This variance quantifies the inconsistency of the model's predictions for a given input. A low variance indicates either high confidence in correctness (near-perfect success) or high confidence in failure (uniformly wrong), both of which are suboptimal for entropy minimization, as they lead to low-entropy posteriors that cannot be further improved.

We therefore define our data selection objective as:

$$x^* = \arg\max_{x \in \mathcal{D}} \text{Var}_{\text{pass@k}}(x),$$

where $\mathcal{D}$ denotes the unlabeled data pool.

This approach effectively prioritizes those prompts where the model exhibits the most behavioral uncertainty, making them "entropy-sensitive." Such prompts are empirically found to produce the largest entropy gradients and hence drive meaningful parameter updates under EM.

Intuitively, data with high *pass@k* variance suggests that the model's response distribution is straddling the decision boundary—sometimes correct, sometimes not—indicating a broad or multimodal predictive distribution. These are precisely the regions where entropy minimization is most impactful: it encourages the model to concentrate its probability mass on a consistent and (ideally) correct reasoning trajectory.

By contrast, if a model consistently answers a question correctly or incorrectly regardless of sampling, the entropy is either already minimal or

optimization is ineffective. Thus, high-variance prompts provide the richest signal for improving model calibration and reasoning fidelity.

A sample from the NuminaMath(LI et al., 2024) dataset that meets the data filtering criteria is as follows:

An example of a selected data

Problem: The pressure P exerted by wind on a sail varies jointly as the area A of the sail and the cube of the wind's velocity V. When the velocity is 8 miles per hour, the pressure on a sail of 2 square feet is 4 pounds. Find the wind velocity when the pressure on 4 square feet of sail is 32 pounds.

Solution: 12.8

The ablation results of the variance-based data selection strategy are shown in Table 1.

3 Experiment

3.1 Experimental Setting

We implemented the overall training process of EM based on Acclerate (Gugger et al., 2022). We selected 1 piece of data from the dataset as prompt. Since it is an unsupervised method, we do not need any data labels. We directly train the model for only 10 steps with a constant learning rate of 2×10^{-5}, a temperature of 0.5, and a batch size of 64. The reason why only 10 steps are sufficient will be explained in detail in Section 3.4.

3.2 Main Result

We present our experimental results in Table 3. Compared with most RL-based baselines, our EM 1-shot results show strong competitiveness. Specifically, when applying our 1-shot EM method to the Qwen2.5-Math-7B base model, substantial performance improvements are observed across all evaluated math reasoning benchmarks. The performance on MATH500 significantly increases by 25.8 points (from 53.0 to 78.8), and similarly large improvements occur in Minerva Math (+24.3 points), Olympiad Bench (+22.5 points), and AMC23 (+26.2 points). On average, the EM 1-shot strategy achieves an impressive gain of 24.7 points compared to the original Qwen2.5-Math-7B model.

Notably, even with only a single-shot example and minimal training steps (only 10), EM dramatically reduces the gap between Qwen2.5-Math-7B and state-of-the-art RL-based models such as Prime-Zero-7B and RLVR-GRPO. In particular, on the AMC23 benchmark, the EM-enhanced Qwen2.5-Math-7B achieves a competitive score of 70.3, nearing the leading RL models. These results clearly indicate that Entropy Minimization (EM), despite being a simpler and more data-efficient technique compared to typical reinforcement learning methods, has substantial potential to enhance the performance of foundational language models in math reasoning tasks.

During training, the model repeatedly reinforces its own confidence, causing the logits distribution to gradually shift to the right—a phenomenon that will also be discussed in detail in Section 3.3.

3.3 Logits Shift

We sample 20 prompts from the NuminaMath (LI et al., 2024) dataset and generate responses using four different models (Qwen2.5-Math-7B, Qwen2.5-Math-7B-EM, Qwen2.5-Math-7B-RL, Qwen2.5-Math-7B-EM-RL). Each model generates 20 responses, resulting in a total of $4 \times 20 = 80$ outputs. For each output, we extract the logits, defined as the unnormalized scores $z_i \in R$ produced by the model before applying the softmax function.

Given a set of logits $\{z_1, z_2, \ldots, z_n\}$, the probability for each token is computed as:

$$p_i = \frac{e^{z_i}}{\sum_{j=1}^{n} e^{z_j}}.$$

To analyze the overall distributional behavior, we flatten all collected logits across all responses into a single one-dimensional vector:

$$\mathbf{z}_{\text{flat}} = \{z_1^{(1)}, z_2^{(1)}, \ldots, z_{m_1}^{(1)}, z_1^{(2)}, \ldots, z_{m_2}^{(2)}, \ldots, z_{m_k}^{(k)}\},$$

where $z_j^{(i)}$ denotes the j-th logit in the i-th sample, and $k = 80$ is the total number of generated samples. This flattening allows us to perform global statistical analyses.

To quantify the asymmetry of the flattened logits distribution, we compute the skewness:

$$\text{Skewness} = \frac{1}{n} \sum_{i=1}^{n} \left(\frac{z_i - \mu}{\sigma} \right)^3,$$

where μ and σ are the mean and standard deviation of the flattened logits vector $\mathbf{z}_{\text{flat}}$, respectively.

Model	Dataset Size	Training Step	MATH500	Minerva	OlympiadBench	AMC23	KK	MBPP	Avg.
OpenReasoner-Zero-7B (Hu et al., 2025)	129k	600+	79.2	31.6	44.0	47.0	27.2	78.3	51.1
SimpleRL-Zoo (Zeng et al., 2025)	24K	4000	76.8	30.9	39.4	55.3	17.4	75.9	49.3
Prime-Zero-7B (Cui et al., 2025)	230K	240	83.8	36.0	40.9	62.7	4.4	41.0	44.8
Oat-Zero-7B (Liu et al., 2025)	12K	300	80.0	30.1	41.0	62.7	2.2	58.7	45.8
RLVR (Wang et al., 2025)	1.2 K	1000	78.6	33.8	41.6	62.5	12.4	54.5	47.2
RLVR 16-shot	16	1000	77.8	35.3	39.9	62.2	2.4	58.5	46.0
RLVR 1-shot	1	1000	78.6	36.0	43.7	61.9	5.8	60.3	47.7
Qwen3-8B-Base	NA	NA	53.8	22.1	20.4	31.6	1.2	49.1	29.7
+ EM 1-shot	**1**	**10**	$60.4^{\uparrow 6.6}$	$26.8^{\uparrow 4.7}$	$21.3^{\uparrow 0.9}$	$35.6^{\uparrow 4.0}$	$9.2^{\uparrow 8.0}$	$54.2^{\uparrow 5.1}$	$34.6^{\uparrow 4.9}$
w/o Variance-based Data Selection	**1**	**10**	60.5	24.3	21.0	35.2	8.4	53.8	33.9
Qwen2.5-Math-7B	NA	NA	53.0	11.0	17.2	44.1	1.0	48.9	29.2
+ EM 1-shot	**1**	**10**	$78.8^{\uparrow 25.8}$	$35.3^{\uparrow 24.3}$	$39.7^{\uparrow 22.5}$	$70.3^{\uparrow 26.2}$	$17.4^{\uparrow 16.4}$	$65.1^{\uparrow 16.2}$	$51.1^{\uparrow 24.9}$
w/o Variance-based Data Selection	**1**	**10**	77.9	35.0	37.5	70.5	11.5	64.3	49.5

Table 1: Comparison of different methods on math reasoning benchmarks (MATH500 (Lightman et al., 2023), MinervaMath (Lewkowycz et al., 2022), OlympiadBench (He et al., 2024), AMC23), logic reasoning benchmark (KK (Xie et al., 2024)), and code benchmark (MBPP (Austin et al., 2021)). To reduce randomness, each benchmark is evaluated under avg@8.

A positive skewness indicates a right-tailed distribution, whereas a negative skewness indicates a left-tailed distribution.

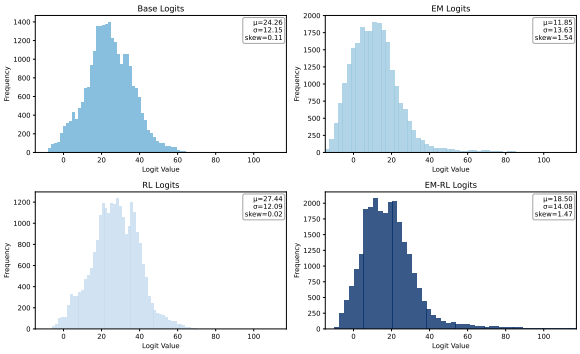

Figure 1: Distribution plot of the flattened logits sampled from 20 sample data points for each of the four models: Base, EM, RL, and RL after EM.

As observed in Figure 1, models trained with EM exhibit a significantly higher skewness in their logits distribution, indicating a rightward shift. This suggests that EM increases overall model confidence, concentrating probability mass on a subset of tokens. As a result, previously high-probability regions in the original logits become extended into long-tail high-probability intervals.

In contrast, models trained with RL show a pronounced reduction in logits skewness, indicating a leftward shift in the distribution. We hypothesize that this is due to the influence of ground-truth signals during RL training. Specifically, RL appears to perform re-ranking that suppresses tokens with high predicted probability but low alignment with ground truth, thereby reducing their rank and shifting the overall distribution leftward. Similarly, models trained with EM followed by RL exhibit the same pattern: after RL, the skewness of the model's logits decreases from 1.54 (after EM) to 1.47, aligning with this pattern.

We refer to this phenomenon as ***logits shift***. A rightward logits shift is beneficial to the generation and sampling process of large language models, as the model primarily samples from high-probability tokens. Shifting the logits to the right increases the number of candidate tokens and expands high-probability paths, thereby enhancing model capability. In contrast, a leftward logits shift is considered harmful to the generation process of large language models, as it reduces the number of high-probability paths during sampling—opposite to the effect of a rightward shift—thus diminishing the model's overall performance. Therefore, the rightward logits shift induced by EM is preferable to the leftward shift caused by RL.

3.4 Training Loss vs. Reasoning Performance

As shown in Figure 2, the loss drops to a relatively low level around step 10, and the model's performance on mathematical reasoning reaches its peak. However, unexpectedly, as the EM training loss continues to decrease beyond step 10, the mathematical reasoning performance begins to decline.

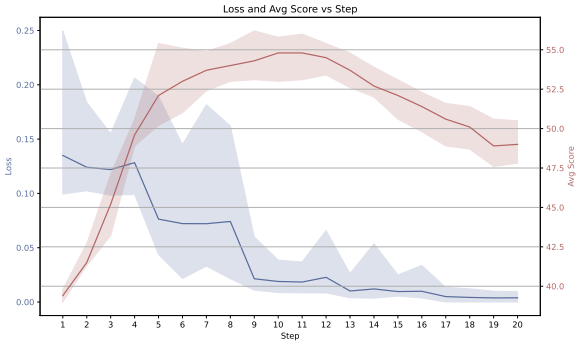

Figure 2: The left y-axis represents the EM training loss, while the right y-axis shows the average score across four reasoning benchmarks. It can be observed that the training loss converges rapidly, whereas the average score peaks around step 10 and then begins to decline. The results in the figure are obtained by repeating the experiments with the same training hyper-parameters using 16 different seeds to reduce randomness.

We designate this phenomenon as ***over confidence***. Persistent EM may excessively amplify the model's confidence in its tokens during inference, thereby exacerbating algorithmic bias and leading to significant deviations in outputs. In conjunction with the findings presented in Section 3.3, we argue that EM functions primarily as a tool for shaping the model's distribution rather than as a learning method or strategy. Consequently, the effect of distribution shaping is largely achieved within a very small number of training steps, leading to a decoupling between the continued decrease in EM training loss and improvements in mathematical reasoning performance.

3.5 Sampling Temperature in Training

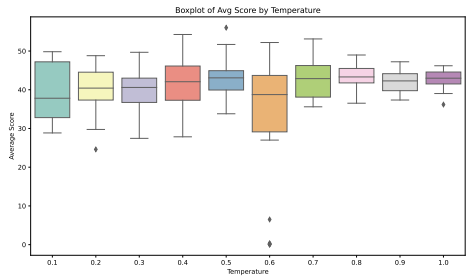

Figure 3: The impact of generation temperature during EM training on the average performance of the trained model across four reasoning datasets. The results in the figure are obtained by repeating the experiments with the same training hyper-parameters using 16 different seeds to reduce randomness.

As shown in Figure 3, the average performance of the EM-trained model across four math reasoning benchmarks generally exhibits an upward trend as the generation temperature increases. The maximum of the average performance initially increases and then declines around a temperature of 0.5. Higher temperatures lead to better average reasoning ability, while moderate temperatures (e.g., 0.5) result in greater performance variance, thereby creating opportunities for higher peak performance. Therefore, we prioritize the model trained at temperature 0.5 when reporting final results.

However, as shown in the figure, EM training exhibits significant randomness. The results in the figure are obtained by repeating experiments with 16 different random seeds under the same set of hyperparameters. It can be seen that, even with identical settings, the average scores across the four math reasoning benchmarks can differ by as much as a factor of two depending on the seed. Therefore, all the conclusions in this paper are based on at least 16 repeated experiments with different seeds. We also advocate that future research should focus on reducing the stochasticity of EM training.

3.6 Sampling Temperature in Evaluate Generation

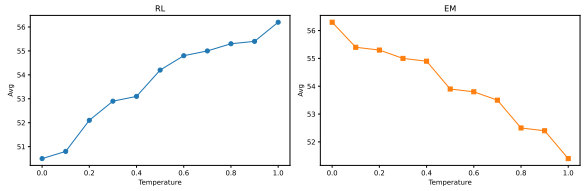

Figure 4: The impact of generation temperature during evalutating on the average performance of the trained model across four reasoning datasets. The results in the figure are obtained by repeating the experiments with the same training hyper-parameters using 16 different seeds to reduce randomness.

As shown in Figure 4, we observe a striking pattern in the EM-trained model's response to varying sampling temperatures during generation. Specifically, as the sampling temperature increases during evaluation, the model's average performance across four math reasoning benchmarks consistently *decreases*. This trend is in sharp contrast to that of Reinforcement Learning (RL)-trained models shown in Figure 4, where higher sampling temperatures often improve performance.

Greedy Decoding. The observation can be formally contextualized through the greedy decoding process, which selects the token with maximum

conditional probability at each step:

$$y_t = \arg\max_{v \in \mathcal{V}} \; p_\theta(v \mid y_{<t}, x),$$

where $\mathcal{V}$ is the vocabulary and x is the input prompt.

Together with our analysis in Section 3.3, we hypothesize that EM training systematically reshapes the model's logits distributions to become increasingly right-skewed. This process reinforces confidence in already high-probability tokens, effectively concentrating the probability mass on semantically coherent and correct options. As a result, greedy decoding—which deterministically selects the most probable token—becomes a highly effective strategy after EM training.

In contrast, RL adjust token probabilities based on external ground-truth reward. This often promotes the relative ranking of previously low-probability (tail) tokens. Even after reranking, these tokens tend to occupy intermediate positions in the probability distribution, requiring higher temperatures during sampling to be selected. Consequently, RL-trained models exhibit the opposite trend: performance improves with higher sampling temperatures, as seen in Figure 4.

3.7 EM Before/After RL

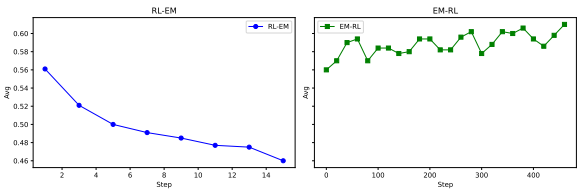

Figure 5: The blue curve on the left represents the average performance across four mathematical reasoning benchmarks of the model trained with RL during the EM phase as training steps progress, while the curve on the right shows the performance of the model trained with EM during the RL phase. The results in the figure are obtained by repeating the experiments with the same training hyper-parameters using 16 different seeds to reduce randomness.

Figure 5 shows a clear asymmetry: applying Entropy Minimization (EM) after Reinforcement Learning (RL) leads to a steady decline in performance across four math benchmarks, whereas applying RL after EM yields consistent gains. This suggests that EM exacerbates the distributional distortions introduced by RL, reinforcing the "alignment tax" of RL.

This aligns with prior work showing that RL is most effective when preceded by Supervised Fine-Tuning (SFT) (Casper et al., 2023; Gou and Nguyen, 2025), while applying SFT or entropy-based methods after RL often harms performance. In our case, EM after RL locks in narrow, overconfident output modes, while EM before RL enhances reasoning and allows RL to refine outputs without degrading diversity or accuracy.

3.8 EM on Various Models

In Table 2, we report the peak performance of 1-shot EM after minimal training steps on various base models. Remarkably, with only a single exemplar and minimal optimization, EM consistently boosts inference accuracy across all models on tasks MATH500, Minerva Math, Olympiad Bench, and AMC23. For instance, on the Qwen2.5-Math-7B model, 1-shot EM yields impressive improvements of 25.8 points on MATH500 (from 53.0 to 78.8), 24.3 points on Minerva Math (from 11.0 to 35.3), 22.5 points on Olympiad Bench (from 17.2 to 39.7), and 26.2 points on AMC23 (from 44.1 to 70.3). This demonstrates EM's ability to substantially enhance reasoning performance with virtually no extra data or compute. Consequently, EM emerges as a compelling lightweight alternative to RL across diverse model bases.

We observe that the upper bound of 1-shot EM's gains is determined by the base model's intrinsic reasoning strength. On the relatively weak LLaMA-3.1-8B, 1-shot EM raises average accuracy only to 24.3%, barely surpassing the baseline of 23.6%. This suggests that when the underlying model lacks sufficient reasoning capacity, minimal EM optimization cannot fully compensate for its deficits. In contrast, on the stronger Qwen2.5-7B base, 1-shot EM elevates average accuracy significantly from 29.6% to 37.3%. Even more notably, on the highly capable Qwen2.5-7B-Instruct base, 1-shot EM pushes average accuracy from 43.12% to 44.5%, not only surpassing the baseline but also outperforming EM alone.

Crucially, EM requires no external instruction–response pairs or reward models and can be applied on top of existing SFT or RL checkpoints as a "confidence compression" layer, enabling faster convergence and more stable outputs from very few exemplars. However, interestingly, on the RL-trained SimpleRL-Zoo base, 1-shot EM resulted in a accuracy drop from 44.14% to 39.1%, suggesting potential limitations when applied to models

already extensively fine-tuned with RL methods. Nonetheless, across most scenarios—especially in SFT and minimally trained RL contexts—EM significantly prunes redundant decision paths and stabilizes key predictions, validating its effectiveness and versatility as a minimal yet powerful optimization strategy.

3.9 1-shot vs. Multi-shot

As shown in Table 2, EM training with just one or two examples achieves an average score on four math benchmarks that is on par with a dataset comprising thousands of examples, and even yields improvements, respectively. Data in Table 2 further demonstrate that EM using a minimal exemplar set delivers superior generalization and stronger performance across diverse downstream tasks.

To uncover the underlying factors, we conducted a detailed analysis of the 1-shot EM training dynamics. We observed that both prompt length and generated output length remain markedly more stable under 1-shot EM. Moreover, whereas multi-shot EM losses continue to fluctuate significantly after step 3, the 1-shot EM loss steadily declines from step 3 onward and remains at a low level beyond step 10. This indicates that relying on a single exemplar substantially reduces sample bias and narrows output variance, enabling more fine-grained and stable optimization.

Furthermore, as depicted in Figure 3.4, EM exhibits RL-like post-saturation generalization behavior: the 1-shot EM loss saturates at step 10, highlighting its remarkably fast convergence—just one example is sufficient to achieve significant gains.

Notably, in the late stage of training, EM loss and model performance become decoupled—further loss reduction does not translate into performance gains and may even incur a slight degradation. We attribute this phenomenon to EM's "overconfidence" effect, wherein the model amplifies its own prior biases, ultimately leading to a performance penalty.

4 Related Work

4.1 Entropy Minimization

Wang et al. (Wang et al., 2025) discovered that RLVR can achieve performance comparable to using thousands of data with just a single data, which is one of the main inspirations for our work. They also were the first to observe that simply optimizing the entropy loss can significantly improve reasoning performance. However, they did not explore this phenomenon in depth, stating in the original paper, "We leave the rigorous analysis to future works." Agarwal et al. (Agarwal et al., 2025) is the first to study entropy minimization during post-training for large language models. However, they considered the effectiveness of entropy minimization to be unreasonable; although they presented experimental results, their analysis of entropy minimization remained limited.

4.2 Reinforcement Learning for LLM

Recent research has increasingly explored post-training approaches to improve the reasoning abilities of large language models. These approaches often involve additional fine-tuning or reinforcement learning using curated datasets that include reasoning tasks and chain-of-thought annotations (Xu et al., 2025b; Xie et al., 2025; Gao et al., 2024; Zeng et al., 2025; Hu et al., 2025; Cui et al., 2025; Wang et al., 2025; Agarwal et al., 2025). Reinforcement learning techniques such as Direct Preference Optimization (DPO) (Rafailov et al., 2023), Proximal Policy Optimization (PPO) (Schulman et al., 2017), Group Relative Policy Optimization (GRPO) (Shao et al., 2024), and REINFORCE++ (Hu, 2025) are gaining prominence in this area.

5 Limitation

Instability of EM Training While entropy minimization (EM) demonstrates remarkable efficiency—achieving substantial performance gains within as few as ten training steps—this very rapid convergence also introduces notable limitations. EM optimization is highly sensitive to hyperparameter configurations such as learning rate, step count, and temperature scaling, making its behavior difficult to control across settings. As discussed in Section 3.2, continued reduction of the EM loss beyond a certain threshold can in fact degrade reasoning quality, indicating that EM functions more as a distribution–reshaping process than as a conventional gradient-based learning algorithm. When the model becomes excessively confident, its predictive distribution may collapse into a narrow subset of tokens, harming both output diversity and factual correctness. Consequently, stabilizing EM training remains a key limitation of this approach. Future work may explore adaptive learning-rate schedules, confidence-based early stopping, or entropy regularization to mitigate these instabilities.

Model	MATH500	Minerva	OlympiadBench	AMC23	KK	MBPP	Avg.
Qwen3-8B-Base	53.8	22.1	20.4	31.6	1.2	49.1	29.7
+ EM multi-shot	49.2	23.2	13.8	25.9	7.6	73.0	32.1
+ EM 1-shot	60.4	26.8	21.3	35.6	9.2	54.2	34.6
LLaMA-3.1-8B	49.0	21.3	15.4	25.6	7.6	73.0	32.0
+ EM multi-shot	49.2	23.2	13.8	25.9	7.6	73.0	32.1
+ EM 1-shot	42.6	21.0	17.0	37.8	10.2	70.1	33.1
Qwen2.5-7B	59.8	10.3	29.2	40.3	11.8	72.5	37.3
+ EM multi-shot	61.8	18.4	31.1	43.4	14.0	78.3	41.2
+ EM 1-shot	67.4	22.1	33.6	45.6	12.6	75.9	42.9
Qwen2.5-7B-Instruct	77.0	37.1	37.9	51.9	17.0	80.7	50.3
+ EM multi-shot	74.6	34.2	36.4	48.4	11.0	78.3	47.2
+ EM 1-shot	76.4	36.4	39.3	54.1	14.8	77.5	49.8
Qwen2.5-Math-7B	53.0	11.0	17.2	44.1	1.0	48.9	29.2
+ EM multi-shot	68.0	18.8	33.2	61.6	9.0	67.2	43.0
+ EM 1-shot	78.8	35.3	39.7	70.3	17.4	65.1	51.1

Table 2: Comparison of different methods on math reasoning benchmarks (MATH500 (Lightman et al., 2023), MinervaMath (Lewkowycz et al., 2022), OlympiadBench (He et al., 2024), AMC23), logic reasoning benchmark (KK (Xie et al., 2024)), and code benchmark (MBPP (Austin et al., 2021)). To reduce randomness, each benchmark is evaluated under avg@8.

Task Dependence Although the proposed one-shot EM method achieves strong improvements on mathematical reasoning, logical reasoning, and coding tasks, its effectiveness across a wider variety of domains remains uncertain. In particular, tasks involving creativity, open-ended generation, or factual knowledge retrieval—such as story writing or knowledge-based question answering—may not benefit equally, and could even experience performance degradation due to overconfidence or reduced linguistic diversity. Moreover, EM's reliance on deterministic token-level certainty might conflict with the need for exploration in tasks requiring flexible reasoning or multi-modal inference. A more comprehensive evaluation across diverse benchmarks and modalities is necessary to assess the generality and robustness of this approach.

6 Conclusion

This work introduces one-shot entropy minimization as a simple yet powerful post-training method for large language models. Using just a single unlabeled example, our approach achieves performance comparable to or better than reinforcement learning methods that rely on large-scale supervision and carefully designed rewards. It is fully unsupervised, highly efficient, and converges within a few training steps. Our experiments show that entropy minimization reshapes the model's output distribution to increase confidence in correct reasoning paths, effectively enhancing the utility of pretrained knowledge. We identify key indicators—such as logit skewness and behavioral variance—that help target entropy-sensitive inputs and guide optimization.

These findings indicate that significant improvements in reasoning performance can be attained by restructuring existing knowledge through confidence calibration, rather than by acquiring additional information. Entropy minimization thus emerges as both a practical technique and a conceptual framework for advancing our understanding of post-training in large language models.

References

Shivam Agarwal, Zimin Zhang, Lifan Yuan, Jiawei Han, and Hao Peng. 2025. The unreasonable effectiveness of entropy minimization in llm reasoning. *Preprint*, arXiv:2505.15134.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. 2021. Program synthesis with large language models. *Preprint*, arXiv:2108.07732.

Stephen Casper, Xander Davies, Claudia Shi, Thomas Krendl Gilbert, Jérémy Scheurer, Javier Rando, Rachel Freedman, Tomasz Korbak, David Lindner, Pedro Freire, Tony Wang, Samuel Marks, Charbel-Raphaël Segerie, Micah Carroll, Andi Peng, Phillip Christoffersen, Mehul Damani, Stewart Slocum, Usman Anwar, and 13 others. 2023. Open problems and fundamental limitations of reinforcement learning from human feedback. *Preprint*, arXiv:2307.15217.

Ganqu Cui, Lifan Yuan, Zefan Wang, Hanbin Wang, Wendi Li, Bingxiang He, Yuchen Fan, Tianyu Yu, Qixin Xu, Weize Chen, Jiarui Yuan, Huayu Chen, Kaiyan Zhang, Xingtai Lv, Shuo Wang, Yuan Yao, Xu Han, Hao Peng, Yu Cheng, and 4 others. 2025. Process reinforcement through implicit rewards. *Preprint*, arXiv:2502.01456.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, and 181 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *Preprint*, arXiv:2501.12948.

Zitian Gao, Boye Niu, Xuzheng He, Haotian Xu, Hongzhang Liu, Aiwei Liu, Xuming Hu, and Lijie Wen. 2024. Interpretable contrastive monte carlo tree search reasoning. *Preprint*, arXiv:2410.01707.

Qi Gou and Cam-Tu Nguyen. 2025. Mixed preference optimization: Reinforcement learning with data selection and better reference model. *Preprint*, arXiv:2403.19443.

Sylvain Gugger, Lysandre Debut, Thomas Wolf, Philipp Schmid, Zachary Mueller, Sourab Mangrulkar, Marc Sun, and Benjamin Bossan. 2022. Accelerate: Training and inference at scale made simple, efficient and adaptable. https://github.com/huggingface/accelerate.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Leng Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, Jie Liu, Lei Qi, Zhiyuan Liu, and Maosong Sun. 2024. Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems. *Preprint*, arXiv:2402.14008.

Jian Hu. 2025. Reinforce++: A simple and efficient approach for aligning large language models. *Preprint*, arXiv:2501.03262.

Jingcheng Hu, Yinmin Zhang, Qi Han, Daxin Jiang, Xiangyu Zhang, and Heung-Yeung Shum. 2025. Open-reasoner-zero: An open source approach to scaling up reinforcement learning on the base model. *Preprint*, arXiv:2503.24290.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, Yuhuai Wu, Behnam Neyshabur, Guy Gur-Ari, and Vedant Misra. 2022. Solving quantitative reasoning problems with language models. *Preprint*, arXiv:2206.14858.

Jia LI, Edward Beeching, Lewis Tunstall, Ben Lipkin, Roman Soletskyi, Shengyi Costa Huang, Kashif Rasul, Longhui Yu, Albert Jiang, Ziju Shen, Zihan Qin, Bin Dong, Li Zhou, Yann Fleureau, Guillaume Lample, and Stanislas Polu. 2024. Numinamath. [https://huggingface.co/AI-MO/NuminaMath-CoT](https://github.com/project-numina/aimo-progress-prize/blob/main/report/numina_dataset.pdf).

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step. *arXiv preprint arXiv:2305.20050*.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. 2025. Understanding r1-zero-like training: A critical perspective. *Preprint*, arXiv:2503.20783.

OpenAI. 2024. Introducing openai o1. https://openai.com/o1/. Accessed: 2024-10-02.

OpenAI. 2025. Introducing openai o3 and o4-mini. Accessed: 2025-05-27.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. In *Advances in Neural Information Processing Systems*, volume 36, pages 53728–53741. Curran Associates, Inc.

Noam Razin, Zixuan Wang, Hubert Strauss, Stanley Wei, Jason D. Lee, and Sanjeev Arora. 2025. What makes a reward model a good teacher? an optimization perspective. *Preprint*, arXiv:2503.15477.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *Preprint*, arXiv:1707.06347.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024.

Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *Preprint*, arXiv:2402.03300.

Kimi Team, Angang Du, Bofei Gao, Bowei Xing, Changjiu Jiang, Cheng Chen, Cheng Li, Chenjun Xiao, Chenzhuang Du, Chonghua Liao, Chuning Tang, Congcong Wang, Dehao Zhang, Enming Yuan, Enzhe Lu, Fengxiang Tang, Flood Sung, Guangda Wei, Guokun Lai, and 75 others. 2025. Kimi k1.5: Scaling reinforcement learning with llms. *Preprint*, arXiv:2501.12599.

Yiping Wang, Qing Yang, Zhiyuan Zeng, Liliang Ren, Lucas Liu, Baolin Peng, Hao Cheng, Xuehai He, Kuan Wang, Jianfeng Gao, Weizhu Chen, Shuohang Wang, Simon Shaolei Du, and Yelong Shen. 2025. Reinforcement learning for reasoning in large language models with one training example. *Preprint*, arXiv:2504.20571.

Chulin Xie, Yangsibo Huang, Chiyuan Zhang, Da Yu, Xinyun Chen, Bill Yuchen Lin, Bo Li, Badih Ghazi, and Ravi Kumar. 2024. On memorization of large language models in logical reasoning.

Tian Xie, Zitian Gao, Qingnan Ren, Haoming Luo, Yuqian Hong, Bryan Dai, Joey Zhou, Kai Qiu, Zhirong Wu, and Chong Luo. 2025. Logic-rl: Unleashing llm reasoning with rule-based reinforcement learning. *Preprint*, arXiv:2502.14768.

Fengli Xu, Qianyue Hao, Zefang Zong, Jingwei Wang, Yunke Zhang, Jingyi Wang, Xiaochong Lan, Jiahui Gong, Tianjian Ouyang, Fanjin Meng, Chenyang Shao, Yuwei Yan, Qinglong Yang, Yiwen Song, Sijian Ren, Xinyuan Hu, Yu Li, Jie Feng, Chen Gao, and Yong Li. 2025a. Towards large reasoning models: A survey of reinforced reasoning with large language models. *Preprint*, arXiv:2501.09686.

Haotian Xu, Xing Wu, Weinong Wang, Zhongzhi Li, Da Zheng, Boyuan Chen, Yi Hu, Shijia Kang, Jiaming Ji, Yingying Zhang, Zhijiang Guo, Yaodong Yang, Muhan Zhang, and Debing Zhang. 2025b. Redstar: Does scaling long-cot data unlock better slow-reasoning systems? *Preprint*, arXiv:2501.11284.

Weihao Zeng, Yuzhen Huang, Qian Liu, Wei Liu, Keqing He, Zejun Ma, and Junxian He. 2025. Simplerl-zoo: Investigating and taming zero reinforcement learning for open base models in the wild. *Preprint*, arXiv:2503.18892.

A Appendix

A.1 Training Learning Rate

As shown in Figure 6, a learning rate of 2×10^{-5} demonstrates a significant advantage. Since EM typically converges and reaches peak performance within just 10 steps, an excessively large learning rate may cause severe and rapid overconfidence, while a smaller learning rate can slow down convergence. To improve training efficiency, we therefore recommend using 2×10^{-5} as the default baseline setting for EM training.

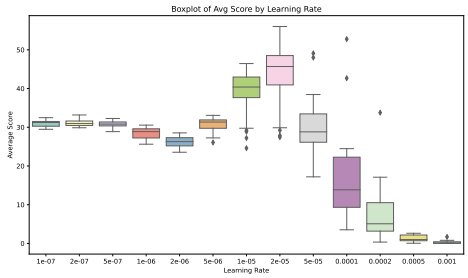

Figure 6: The impact of learning rate during EM training on the average performance of the model at step 10 across four reasoning datasets. The results in the figure are obtained by repeating the experiments with the same training hyper-parameters using 16 different seeds to reduce randomness.