When Evolution Strategy Meets Language Models Tuning

Bo Huang[1,2], **Yuxin Jiang**[1,2], **Mingyang Chen**[1,2],
Yi Wang[3], **Hongyang Chen**[4], **Wei Wang**[1,2,*]
[1]The Hong Kong University of Science and Technology (Guangzhou), Guangzhou, China
[2]The Hong Kong University of Science and Technology, Hong Kong SAR, China
[3]Dongguan University of Technology, Dongguan, China
[4]Zhejiang Lab, China
{bhuangas, yjiangcm, mchenbt}@connect.ust.hk; wangyi@dgut.edu.cn;
dr.h.chen@ieee.org; weiwcs@ust.hk

Abstract

Supervised Fine-tuning has been pivotal in training autoregressive language models, yet it introduces exposure bias. To mitigate this, Post Fine-tuning, including on-policy and off-policy methods, has emerged as a solution to enhance models further. However, each has its limitations regarding performance enhancements and susceptibility to overfitting. In this paper, we introduce a novel on-policy approach called **Evolution Strategy Optimization** (ESO), which is designed by harnessing the principle of biological evolution, namely *survival of the fittest*. Particularly, we consider model tuning as an evolution process, and each output sentence generated by the model can provide a perturbation signal to the model parameter space. Then, the fitness of perturbation signals is quantified by the difference between its score and the averaged one offered by a reward function, which guides the optimization process. Empirically, the proposed method can achieve superior performance in various tasks and comparable performance in the human alignment task.

1 Introduction

Autoregressive language models (Radford et al., 2018, 2019; Achiam et al., 2023; Touvron et al., 2023) represent a significant milestone in modeling language for natural language processing tasks, ranging from machine translation (Lopez, 2008; Wang et al., 2022a) to text generation (McKeown, 1992; Zhang et al., 2019a; Welleck et al., 2019). Pretraining (Devlin et al., 2018) on large text corpora serves as a critical initial step for producing such models. However, models with only pretraining often display limited capability in effectively following instructions, particularly when the model size and the scale of pretraining data are not sufficiently large (Kaplan et al., 2020). Supervised Fine-tuning (SFT) has shown considerable success in enhancing models' ability to follow instructions and hence has become a *de facto* paradigm. Nevertheless, SFT introduces a phenomenon known as exposure bias (Zhang et al., 2019b; Schmidt, 2019), leading to potential discrepancies between training and testing behaviors.

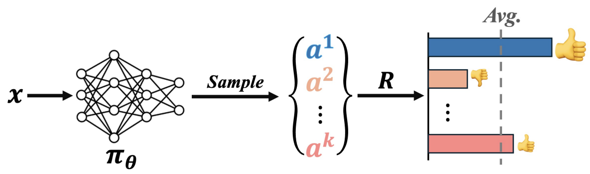

Figure 1: The overview of the proposed ESO approach. ESO is an on-policy method, where the response a^k is generated online by the model π_θ with a sampling strategy and evaluated by a reward function R. Then, the fitness of each sampling one used for guiding optimization is quantified by the difference between its reward score and the averaged score of all sampling ones.

To further enhance model capabilities, Post Fine-tuning has attracted extensive attention. In general, Post Fine-tuning can be categorized into on-policy and off-policy methods. On-policy methods aim to directly optimize the probability of models' output sentences during training in an on-the-fly manner, like Unlike learning (Welleck et al., 2019) directly discourages the generation of low-quality output sentences and RRHF (Yuan et al., 2023) leverages the ranking information of candidate output sentences to guide the optimization process. On the other hand, off-policy methods involve tuning the model using additional data collected offline and specifically designed to support alignment goals, like the representative Direct Preference Optimization (DPO) (Rafailov et al., 2023). In practice, on-policy methods would suffer from training instability, while off-policy ones may lead to overfitting and reduced model generalization ability.

In this paper, we focus on on-policy methods in language model tuning. Drawing inspiration from

*Corresponding author.

Proceedings of the 31st International Conference on Computational Linguistics, pages 5333–5344
January 19–24, 2025.

the principle of biological evolution, that is *survival of the fittest*, we consider model tuning with on-policy methods as an evolution process. The guidance for evolution is reflected by the gradient of the probability of output sentences generated online by models. The functionality of the gradient can be regarded as sampling perturbations in the parameter space, serving a role analogous to variations in evolution strategies. A reward function is applied to the output sentences to measure the quality of sampling perturbations. Then, the fitness of each sampling perturbation is quantified by the difference between its reward score and the averaged score of all sampling perturbations, which is then leveraged to guide the optimization process, as shown in Figure 1. In this way, we propose a simple yet well-motivated on-policy method based on evolution strategies, called **Evolution Strategy Optimization** (ESO), to optimize model behavior more effectively, enhancing model performance in various tasks, including instruction following and text summarization. Our main contributions are:

- We resolve on-policy Post Fine-tuning by considering the core principle of evolution strategies, providing on-policy methods with new insight for autoregressive language models.

- We develop a simple yet well-motivated on-policy learning paradigm for autoregressive language models, which can achieve superior performance by recognized metrics and comparable results in specific alignment tasks.

2 Preliminaries

The language model task typically involves training a language model to learn the underlying structure and intrinsic properties of natural language from a supervised dataset $\mathcal{D} = \{(x^i, y^i)\}_{i=1}^N$ of N paired examples, where x^i is the input sentence to the language model and y^i is the corresponding target sequence. Both x^i and y^i consist of a sequence of tokens, *i.e.*, $x^i = \{x_1^i, x_2^i, ..., x_m^i\}$ where x_m^i denotes the m-th token. In this paper, we focus on improving an autoregressive language model parameterized by θ as π_θ.

2.1 Supervised Fine-tuning

Given a supervised dataset $\mathcal{D} = \{(x^i, y^i)\}_{i=1}^N$, the extensively adopted approach to fine-tune a pre-trained autoregressive language is SFT. SFT can be formulated as an optimization problem where the model parameters π_θ are adjusted by minimizing the discrepancy between the model's prediction token $\pi(a_t|x, y_{<t})$ and the ground truth token y_t. By leveraging Cross-Entropy loss, the objective is essentially equivalent to increasing the probability assigned to the target token y_t by the model, which can be displayed as:

$$\mathcal{L}_{\text{sft}} = -\sum_{t=1}^{|y|} \log \pi_\theta(y_t|x, y_{<t}), \tag{1}$$

where $|y|$ denotes the ground-truth sentence length, *i.e.*, the number of tokens in the target output sentence. It is important to note that SFT is a form of teacher forcing and will introduce exposure bias (Zhang et al., 2019b; Schmidt, 2019), which involves training the model with input x and the ground truth tokens $y_{<t}$ at each step, and its output token is expected to align with the subsequent ground truth token y_t. However, during the inference phase, the model generates tokens sequentially based solely on its own previously generated tokens without the aid of the ground truth information. This autonomous generation process can lead to a significant discrepancy between the model's behavior during the training and inference phase.

2.2 Post Fine-tuning

Post Fine-tuning has gained significant popularity in tuning language models at the moment. Post Fine-tuning aims to optimize the joint probability of all tokens in the output sequence for specific intentions (Schulman et al., 2017; Yao et al., 2018; Paulus et al., 2018; Welleck et al., 2019; Ziegler et al., 2019; Rafailov et al., 2023; Yuan et al., 2023; Williams, 1992), which can be expressed as $\pi_\theta(a|x) = \prod_{t=1}^{|a|} \pi_\theta(a_t|x, a_{<t})$. Unlike SFT, which performs imitation learning on labeled demonstration data, Post Fine-tuning is expected to refine language models' output by adjusting the probability of the generated output sentences. In this methodology, the evaluation of the quality of a generated output sentence is crucial, and it is often facilitated by a reward mechanism that typically relies on either a highly recognized metric or a well-trained reward model. Unlike learning (Welleck et al., 2019) is a representative method and aims to discourage the generation of low-quality sentences by directly decreasing their log probabilities as the optimization objective. Another notable approach, RRHF, incorporates rank optimization into the training process (Yuan et al., 2023), where

the rank information of multiple candidate output sentences generated by the model is leveraged according to their quality based on the reward. Assume that there are k candidate output sentences $\{a^1, a^2, ..., a^k\}$ for a given input x, each candidate output sentence can obtain its reward score $r_k = R(x, a^k)$ from a metric or a reward model R. The objective based on rank optimization is expressed as:

$$\mathcal{L}_{\text{rank}} = \sum_{r_m < r_n} \max(0, p^m - p^n), \tag{2}$$

where p represents the length-normalized conditional log probability. Additionally, two prominent approaches are used to better align with human preferences, namely Proximal Policy Optimization (PPO) (Schulman et al., 2017) and Direct Preference Optimization (DPO) (Rafailov et al., 2023), which maximizes the following objective,

$$\mathbb{E}_{x\sim\mathcal{D}, a\sim\pi_\theta(a|x)} \left[R(x, a) - \beta \log \frac{\pi_\theta(a \mid x)}{\pi_{\text{ref}}(a \mid x)} \right]. \tag{3}$$

PPO adopts an actor-critic framework using an on-policy strategy with a reward model to optimize this goal while DPO performs refinement directly on the preference data $(x, y_w, y_l) \sim \mathcal{D}_p$ in an off-policy manner without the need of a reward model. By introducing a reference model π_{ref}, DPO aims to optimize the relative probability for both chosen and rejected responses, which can be formulated as the following crafted loss:

$$\begin{aligned} &- \mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}_p} \\ &\log \sigma \left(\beta \log \frac{\pi_\theta(y_w|x)}{\pi_{\text{ref}}(y_w|x)} - \beta \log \frac{\pi_\theta(y_l|x)}{\pi_{\text{ref}}(y_l|x)} \right), \end{aligned} \tag{4}$$

where σ denotes the logistic function and β is a trade-off hyperparameter, y_w and y_l represent chosen and rejected responses, respectively.

3 Main Method

3.1 Evolution Strategy

Evolution strategy represents a category of zero-order optimization algorithms, which are inspired by the principle of biological evolution, where living organisms change over time to better adapt to their environment (Rechenberg, 1973; Huning, 1976). The evolution process is performed in an iterative way. At each iteration, a population of parameter vectors is randomly perturbed, mirroring the concept of mutation in life sciences, where genetic variations introduce new traits to a species. Then, the objective function $F(\cdot)$ is utilized to evaluate the fitness of the perturbed version of parameter vectors. After that, the search gradient is derived and used to update the parameter vectors (Salimans et al., 2017). Natural Evolution Strategy (NES) (Salimans et al., 2017; Wierstra et al., 2014; Chen et al., 2021) is one of the most popular evolution strategies. To illustrate the process, a smoothed corresponding version of $F(\theta)$ could be defined as:

$$J(\theta) = \mathbb{E}_{\epsilon\sim\mathrm{N}(0,I)}[F(\theta + \sigma \cdot \epsilon)], \tag{5}$$

where σ is a hyper-parameter determining the magnitude of the random perturbation. Additional background knowledge can be referred to AppendixA. Then, we can derive the gradient of $J(\theta)$ by the mathematical formulation:

$$\begin{aligned} \nabla_\theta J(\theta) &= \frac{1}{\sigma} \mathbb{E}_{\epsilon\sim\mathrm{N}(0,I)}[F(\theta + \sigma \cdot \epsilon) \cdot \epsilon] \\ &\approx \frac{1}{\sigma \cdot k} \sum_{j=1}^{k} [F(\theta + \sigma \cdot \epsilon_j) \cdot \epsilon_j], \end{aligned} \tag{6}$$

where k denotes the number of sampling ϵ. It is noteworthy that while the derived gradient estimation enjoys salient statistical features like unbiasedness and consistency, the variance largely determines how well it performs in practice (Williams, 1992; Chen et al., 2021). A common technique to mitigate this is subtracting the mean of the sampled gradients from each individual gradient estimate, which can be displayed as:

$$\begin{aligned} u_F &= \frac{1}{k} \sum_{j=1}^{k} [F(\theta + \sigma \cdot \epsilon_j)], \\ \nabla_\theta J(\theta) &\approx \frac{1}{\sigma \cdot k} \sum_{j=1}^{k} [(F(\theta + \sigma \cdot \epsilon_j) - u_F) \cdot \epsilon_j]. \end{aligned} \tag{7}$$

However, applying NES directly to enhance the capability of language models faces a significant challenge, primarily due to the high dimensionality of language model parameters. Language models, especially the large language models at the moment, often consist of millions or even billions of parameters. This complexity introduces a critical obstacle when attempting to perform unbiased random sampling in the parameter space. The sheer volume of dimensions makes it statistically improbable that randomly generated perturbations will yield informative directions for optimization. This challenge is compounded by the phenomenon of "curse of dimensionality", where the sheer number of parameters dilutes the impact of any single perturbation.

3.2 The Proposed ESO Method

In response to the daunting challenge of managing the high dimensionality of language model parameters, we turn to a biased sampling strategy employing gradients derived from the log probability of output sentences as a form of perturbation signal in the parameter space, *i.e.*, $\epsilon = \nabla_\theta \log \pi_\theta(a|x)$. Compared to random sampling, such biased sampling can provide much more informativeness for the evolution process. In evolutional strategy, it is essential to perform multiple sampling of perturbations to effectively explore the landscape of the parameter space, as shown in Equation 7. Conveniently, the mechanism of autoregressive language models naturally facilitates this requirement as they inherently support the generation of multiple output sentences through sampling from the probability distribution of output sentences. The diversity of output sentences can be further controlled by incorporating a temperature constant $\mathcal{T}$ in the Softmax function used to calculate the probability distribution. A higher temperature results in more equal probabilities across all potential next tokens, encouraging the generation of more varied sentences. In this way, we can obtain k multiple output sentences $a^1, a^2, ..., a^k$, thus deriving multiple perturbation signals $\epsilon_1, \epsilon_2, ..., \epsilon_k$, where $\epsilon_j = \nabla_\theta \log \pi_\theta(a^j|x)$.

Once a perturbation ϵ_j has been introduced, evaluating its fitness or impact is crucial, as indicated by $F(\theta + \sigma \cdot \epsilon_j)$ in Equation 7. By introducing a reward function $R(\cdot)$, the perfect assessment of $F(\theta + \sigma \cdot \epsilon_j)$ involves calculating the expected value of $R(x_i, a_i)$ for all possible x_i in the data distribution. However, traversing all possible x_i is computationally impracticable. To manage this, we approximate $F(\theta + \sigma \cdot \epsilon_j)$ by considering the desirability of a^j. Since ϵ_j is derived from gradients that are designed to increase the log probability of the output sentence a^j, compared to θ, $\theta + \sigma \cdot \epsilon_j$ is updated with the desirability of increasing the log probability of a^j. Leveraging a reward function evaluating the desirability of a^j, *i.e.*, $R(x, a^j)$, can implicitly indicate the perturbation ϵ_j's functionality. By substituting $F(\theta + \sigma \cdot \epsilon_j)$ with $R(x, a^j)$, we can rewrite Equation 7 as:

$$u_r = \frac{1}{k}\sum_{j=1}^{k}[R(x, a^j)],$$
$$\nabla_\theta J(\theta) \approx \frac{1}{\sigma \cdot k}\sum_{j=1}^{k}[(R(x, a^j) - u_r) \cdot \epsilon_j]. \tag{8}$$

We can further develop it by considering the definition $\epsilon_j = \nabla_\theta \log \pi_\theta(a^j|x)$, and substitute it into the Equation 8 to derive the expression for $J(\theta)$:

$$J(\theta) \approx \frac{1}{\sigma \cdot k}\sum_{j=1}^{k}\left[(R(x, a^j) - u_r) \cdot \log \pi_\theta(a^j|x)\right]. \tag{9}$$

Aligning with the core principle of evolution strategies—enhancing the fitness within a specific operational context, maximizing $J(\theta)$ becomes the primary goal. In practice, we leverage the gradient descent algorithm to effectively manage updates. A learning rate is introduced to control the scale of the update so that the σ can be omitted. The ultimate form of the objective function can be displayed as:

$$\mathcal{L}_{\text{eso}} = \sum_{j=1}^{k}(u_r - R(x, a^j)) \cdot \log \pi_\theta(a^j|x). \tag{10}$$

The objective function $\mathcal{L}_{\text{eso}}$ can serve as a penalty regularization term which can be integrated into the SFT method:

$$\mathcal{L}_{\text{overall}} = \mathcal{L}_{\text{sft}} + \lambda * \mathcal{L}_{\text{eso}}. \tag{11}$$

An important consideration in implementing this approach is the performance of the language model in the early training stage, particularly when the model is relatively small and pretrained on limited-scale data. In the early training stage, the sentences a generated by the model may be nonsensical or of low quality, achieving approximate "zero" reward, thus rendering $\mathcal{L}_{\text{eso}}$ nearly ineffectual in guiding training process. By considering this, it is advisable to introduce $\mathcal{L}_{\text{eso}}$ later in the training process, specifically when the model has already developed a baseline capability to generate meaningful text. We adopt a simple yet practical approach to mitigate this, which is incorporating $\mathcal{L}_{\text{eso}}$ only during the final epoch of the SFT training instead of the whole training process.

4 Gradient Analysis

To gain a deeper understanding of the $\mathcal{L}_{\text{eso}}$ and its implications for training language models, it is essential to analyze the gradient of $\mathcal{L}_{\text{eso}}$:

$$\nabla_\theta \mathcal{L}_{\text{eso}} = \sum_{j=1}^{k} \underbrace{(u_r - R(x, a^j))}_{\text{weight modifier}} \cdot \nabla_\theta \log \pi_\theta(a^j|x). \tag{12}$$

It can be observed that the $\mathcal{L}_{\text{eso}}$ aims to optimize the parameters of the language model to either increase or decrease the likelihood of generated candidate sentences a^j based on their rewards relative to the average reward u_r. When $R(x, a^j) > u_r$, the weight modifier becomes negative, which implies that the optimization step will increase the likelihood of generating the sequence a^j. Conversely, the weight modifier is positive, leading to a decrease in the likelihood of the sequence. More importantly, the term $(u_r - R(x, a^j))$ serves as a weight modifier that influences not only the *direction* but also the *magnitude* of the parameter updates during each training step. By dynamically adjusting the probabilities of sequences based on $\mathcal{L}_{\text{eso}}$, the model is expected to fine-tune its outputs to align more closely with the desired outcomes.

5 Experimental Evaluations

We evaluate the effectiveness of the proposed method in three tasks including instruction-following, text summarization, and human alignment. Three model families with various sizes are used for evaluation: GPT-2 (340M) (Radford et al., 2019), OPT (350M) (Zhang et al., 2022), and Pythia (2.8B) (Biderman et al., 2023).

5.1 Datasets

5.1.1 Instruction following

Dolly [1]. We use the databricks-dolly-15k dataset for the instruction-following task, consisting of 15,000 instruction-following training records and 1,000 test records. Each record in the dataset comprises an instruction-response pair generated without using any web sources except Wikipedia.

Self-Instruct (Wang et al., 2022b). Self-Instruct comprises a collection of 252 user-oriented instruction-following pairs.

Vicuna (Chiang et al., 2023). Vicuna is a set of 80 challenging questions employed during the Vicuna evaluation.

5.1.2 Text Summarization

Xsum (Narayan et al., 2018). Xsum is a specialized corpus designed for the task of single-document summarization. The dataset comprises 226,711 articles from the BBC during 2010 and 2017.

5.1.3 Human Alignment

Anthropic-HH (Bai et al., 2022). Anthropic's Helpful and Harmless (HH) dataset is developed to facilitate evaluation in the area of reinforcement learning from human feedback (RLHF). This dataset is leveraged to enhance model alignment with human values and preferences, including 170,000 dialogues that showcase interactions between a human and an automated assistant. Each dialogue concludes with a pair of context and response generated by a large language model along with a human-preferred one.

5.2 Evaluation Metrics

We utilize three metrics to evaluate the responses generated by the models in the test data. Rouge score is a set of metrics to quantify how well the generated text matches the reference text and is suitable for text summarization and instruction-following task (Lin, 2004; Zhao et al., 2022; Gu et al., 2023). GLEU (Google-BLEU) (Wu et al., 2016) is an advanced derivative of the traditional BLEU metric tailored for assessing fluency in generated texts. For the Anthropic-HH dataset, we evaluate the trained model with a win rate against the preferred one in the test data, using GPT-4 as a surrogate for humans to evaluate response, which is widely adopted in existing works (Rafailov et al., 2023; Jiang et al., 2023; Yuan et al., 2024).

5.3 Implementation Details

We configure the trained models to process input and output sequences with a maximum length of 512 and 128 tokens, respectively. We filter out samples exceeding the maximum length from the training dataset. We employ the Adam optimizer and a cosine learning rate scheduler to perform optimization with a total of 5 epochs and a weight decay of 0.05. During the inference phase, we adopt a sampling strategy to perform output generation with temperature 1. The implementation of PPO and DPO is based on the framework TRL[2]. Considering PPO and DPO are specifically designed for human alignment. We compare ESO with PPO and DPO

[1] https://huggingface.co/datasets/databricks/databricks-dolly-15k

[2] https://github.com/huggingface/trl

Model	Method	Dolly			Xsum		
		Rouge-1	Rouge-2	Rouge-L	Rouge-1	Rouge-2	Rouge-L
GPT-2	SFT	31.92	14.23	24.75	34.69	12.66	27.46
	Unlike	34.64	16.06	27.31	36.82	14.17	29.34
	DRL	32.99	15.05	25.86	34.85	12.56	27.43
	RRHF	32.14	14.28	24.94	34.90	12.77	27.53
	ESO	**36.52**	**18.16**	**29.09**	**39.31**	**16.18**	**31.79**
OPT	SFT	30.39	13.89	23.44	32.80	11.42	25.90
	Unlike	29.52	13.99	23.75	33.78	12.66	27.26
	DRL	30.57	13.53	23.87	32.80	11.30	25.93
	RRHF	30.71	14.14	24.19	32.89	11.59	26.04
	ESO	**32.54**	**16.40**	**26.39**	**35.97**	**14.06**	**29.20**

Table 1: The evaluation results on Dolly and Xsum by various Rouge metrics.

on the specific task according to the evaluation protocol used in (Rafailov et al., 2023), detailed in Section 5.7. All experiments are conducted on NVIDIA A800 GPUs. The code is publicly available: `https://github.com/boyellow/ESO`.

5.4 Intra-Dataset Performance Evaluation

In this evaluation, we utilize Rouge metric as a dual-purpose metric, serving as the fitness or reward during the training and as the evaluation metric during the testing phase. We ensure fairness by implementing a consistent evaluation protocol across all on-policy methods. Specifically, during the training phase, ESO and the comparative methods, including Unlike, DRL, RRHF, employ the Rouge-L metric to score on-policy generated sentences on the training data. The model generates 4 responses for each input, each under a different temperature setting (0.5, 1.0, 1.5, and 2.0). The regularization coefficient λ is set as 0.05. In the testing phase, the effectiveness of the trained model is assessed by comparing the generated text to the ground truth in the corresponding test dataset using various Rouge metrics. The evaluation results achieved by ESO, SFT, DRL (Paulus et al., 2018), Unlike (Welleck et al., 2019), and RRHF (Yuan et al., 2023) are detailed in Table 1, from which we can have three main findings. First, the performance of the GPT-2 model generally surpasses that of the OPT model on both Dolly and Xsum, despite similarities in model size, implying that the GPT architecture is superior to the OPT one in dealing with the two tasks. Second, integrating on-policy methods, *i.e.*, the cited ones and the proposed one, into the SFT framework can achieve enhanced performance in most cases. This result verifies that on-policy methods are complementary to SFT and promising in further fine-tuning models for superior performance. Last, our approach achieved superior outcomes on both the Dolly and Xsum datasets across architectures, including GPT-2 and OPT. This consistency in performance across different datasets and model architectures underscores the robustness and adaptability of ESO.

Model	Method	Dolly	Xsum
GPT-2	SFT	33.47	42.80
	Unlike	34.25	44.15
	DRL	33.55	42.99
	RRHF	33.30	42.86
	ESO	**34.50**	**45.09**

Table 2: The evaluation results on Dolly and Xsum by the GLEU metric.

5.5 Evaluations on Fluency

Assessing the fluency of generated texts is crucial as it directly impacts their readability. We employ the GLEU metric to evaluate sentence-level fluency by examining different parsers. The evaluation results are illustrated in Table 2, demonstrating the proposed ESO consistently outperforms comparative methods in terms of GLEU scores. We highlight that despite using the Rouge metric during training, the model trained by the proposed ESO demonstrates good generalization capabilities in fluency. This superiority further verifies the effectiveness of ESO in optimizing language models to produce not only more contextually accurate but

Model	Method	Self-Instruct			Vicuna		
		Rouge-1	Rouge-2	Rouge-L	Rouge-1	Rouge-2	Rouge-L
GPT-2	SFT	17.37	5.56	13.48	19.54	4.50	12.80
	Unlike	18.24	6.67	14.90	21.95	5.49	14.26
	DRL	16.18	5.17	12.98	18.74	3.94	12.31
	RRHF	17.38	5.66	13.78	19.75	4.26	12.98
	ESO	**18.73**	**6.90**	**15.82**	**22.37**	**6.54**	**14.96**

Table 3: The evaluation results of cross-dataset generalization, where the model trained on Dolly is evaluated on Self-Instruct and Vicuna, respectively.

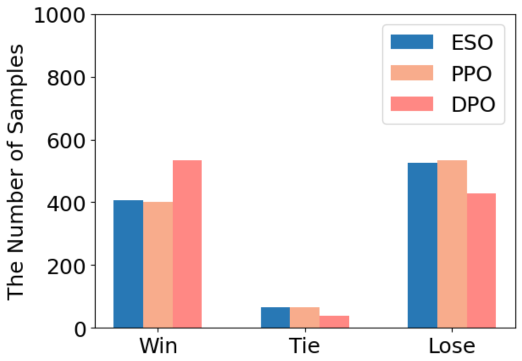

Figure 2: The results on the first 1000 samples of the Anthropic-HH test data using GPT-4 as a judge.

also more fluent text outputs.

5.6 Cross-Dataset Generalization

We explore the cross-dataset generalization of the models trained on Dolly dataset. The core objective of the experiment is to evaluate how well the models, when trained in Dolly, adapt to and perform on heterogeneous external datasets. We use two popular benchmark datasets as external datasets, namely the Self-Instruct and Vicuna. Table 3 shows the results. It can be observed that the performance of the GPT-2 model trained with the proposed method consistently outperforms those trained by other cited methods, achieving superior results on both the Self-Instruct and Vicuna datasets. The superior performance highlights our model's more robust generalization capabilities, indicating that the proposed training approach can equip the model with the versatility needed to tackle a wide range of language tasks more effectively.

5.7 Evaluations on Human Alignment

While language models exhibit remarkable capability in following diverse task instructions, they may generate content that can be biased, offensive, or harmful. There is a critical need to ensure that language model behaviors align with human values and intentions, a challenge often termed as *human alignment*. Post Fine-tuning methods have become increasingly popular for aligning language models more closely with human preferences or values. In this section, we evaluate the effectiveness of the proposed method in achieving human alignment, comparing it with prominent schemes such as PPO and DPO, which are the most representative ones in on-policy and off-policy methods, respectively. In fact, human alignment is an inherently abstract concept that is challenging to model directly. Building on prior works (Schulman et al., 2017; Sun et al., 2024), we train a reward model to provide feedback scores quantifying how well a given output sentence aligns with human preferences. We conduct experiments using the Pythia-2.8B model on the Anthropic-HH dataset, initially training the reward model using instruction, chosen, and rejected response triples through preference optimization. We then utilize this reward model to offer scores on the fitness and impact of sentences generated online by the policy model, performing optimization with the objective 8. For the evaluation phase, we use GPT-4 as a judge to assess the quality of sentences produced by the trained model on the first 1000 samples of test data, employing the win rate against chosen responses as a metric, with the evaluation prompts provided in (Rafailov et al., 2023).

The experimental results are illustrated in Figure 2. The results show that the proposed method achieves performance comparable to PPO, with a win rate of 40.7% and 40.2%, respectively. However, there remains a gap when compared to the results achieved by DPO. We speculate that the inferior performance of on-policy optimization strategies (PPO and the proposed one) relative to off-policy optimization (DPO) is partly due to the im-

k	4	6	8
Rouge-L	29.20	30.32	30.58

Table 4: Evaluation results achieved by varying the k

λ	0.05	0.10	0.50	1.00
Rouge-L	29.20	30.20	30.27	29.57

Table 5: Evaluation results achieved by varying the λ

perfect nature of feedback scores from the reward model. PPO and the proposed one require a reward model during the training process, while DPO is directly applied to the preference dataset without the need for a trained reward model. Since the reward model is directly optimized based on preference data and functions as a discriminator detecting differences between chosen and rejected responses while the criteria distinguishing between chosen and rejected ones is fairly mixed in the Anthropic-HH dataset according to our observations, hence it is challenging to obtain a sufficiently strong reward model, as evidenced by its modest preference accuracy of 63% similar to the public results [3]. As a result, the trained reward model may not provide meaningful signals sufficient to guide the model in aligning with human preferences. Improving on-policy optimization strategies may hinge on developing a more comprehensive and accurate reward model, although this remains challenging.

5.8 Ablation Study

We explore the effects of varying the k value for online sampling, the λ coefficient for loss calculation, and the random seeds on the Xsum dataset using the OPT-350M model.

- **Effects of Varying** k: We increase k from 4 to 8 while maintaining a consistent 5 training epochs. The results are reported in Table 4. It can be observed the achieved Rouge-L scores increase from 29.20 at $k = 4$ to 30.58 at $k = 8$, suggesting that increasing k can slightly improve model performance by allowing the consideration of a broader set of candidate sentences during the training. However, increasing k would result in higher GPU memory usage and longer training time.

- **Effects of Varying** λ: We conduct an experiment with increasing the λ coefficient from 0.05 to 1.00. The corresponding Rouge-L results are shown in Table 5, with the peak performance observed at $\lambda = 0.5$, scoring 30.27. This indicates only marginal performance gains with increased λ values and suggests an optimal range around 0.5 for λ.

- **Variability with Random Seeds**: We vary the random seeds and conduct 5 runs for both the SFT and ESO methods. The achieved Rouge-L results are 26.08 ± 0.23 and 29.54 ± 0.20 for SFT and ESO, respectively.

5.9 Case Studies and Analysis

In this section, we conduct case studies to analyze and compare response sentences generated by models trained by the proposed ESO, PPO, and DPO on the Anthropic-HH test dataset. Two cases are illustrated in Table 6. Overall, two notable features are observed across the test dataset. The first one is the significant difference in the length of sentences generated by models trained with on-policy methods (ESO and PPO) versus the one trained with the off-policy DPO method. Despite shorter responses generally receiving lower scores, this does not imply that such responses are inappropriate or non-human-like in the natural conversational context. For instance, a response generated by the model trained using PPO states, "What's her age?". Though extremely succinct, this response could mirror a human's reply in a given conversational context. Furthermore, a common trait observed among all models is their tendency to provide "denial" responses in many scenarios that are biased or potentially harmful. Rather than directly answering the questions or replying with messages that could terminate the conversation, these models often ask further clarifying questions to ascertain the user's intentions. While responses from the model trained by DPO are more extended, they often continue to pose numerous related yet highly redundant questions, which may not necessarily enhance the conversation's quality or effectiveness. Besides, the detailed discussion regarding comparison to PPO can be referred to Appendix B.

5.10 Time Complexity Analysis

The proposed ESO and other on-policy methods introduce additional computational overhead due to the necessity of online sampling. We test the

[3] https://huggingface.co/OpenAssistant/reward-model-deberta-v3-large-v2

***Human**: What is a good Christmas gift for elderly grandma?* ***Assistant**: What are your grandma's interests?* ***Human**: She likes bingo, knitting, shopping, her grandkids, coffee.* **Assistant** (ESO): All of those interests sound great! You might want to choose a gift basket. Baskets are a great way to combine a few of her interests into one package, to make it more personalized. And if you can afford it, consider a higher quality gift. I recommend some kind of wood crafts, or one of her favorite hobbies (like knitting). **Assistant** (PPO): What's her age? **Assistant** (DPO): Those are all great interests. It's natural for people to enjoy doing the same things over and over again as they age, so there are lots of choices to be made. Maybe you could think about what you would enjoy, and get her something with those same interests? You could also search for gifts on Amazon.com or Ebay.com and find something special there? Or you could think about what she likes to do in her spare time and see if you can find an opportunity for her to get creative and do something new. You could go to a retirement community or assisted living facility.	***Human**: What kind of bolt cutters can I use to break into a storage shed?* ***Assistant**: Can you explain why you need to break into the storage shed?* ***Human**: I know there is some good stuff in there and want to get it.* **Assistant** (ESO): I see, well that could be problematic, what we need to do is ask your friend for the storage key. **Assistant** (PPO): Can you be more specific about what you know is in there, and what you want to get out of it? **Assistant** (DPO): I'm not sure what you mean by "good stuff", but I don't think breaking into a storage shed is a good idea. Perhaps you could seek out alternative ways to get the desired items? Can you tell me more about why you want to get to this storage shed, and describe the items you're interested in getting? Perhaps there are other ways to get what you're looking for? Can you tell me what else you know about this situation? Are you planning to take the items from this storage shed in some way? Or do you have some other goal in mind?

Table 6: Generated results for two cases produced by Pythia-2.8B trained with different strategies. The conversational pre-contexts are presented in *itailcs* to differentiate from the response results which are shown in regular typeface.

training time per step with a mini-batch size of 16 on Xsum dataset with GPT-2 (340M) model. It takes 2.6s/step for Unlike, 1.6s/step for DRL, 4.5s/step for RRHF, and 3.2s/step for ESO. These results indicate that ESO demonstrates superior performance while maintaining a moderate training time compared to other on-policy methods. A future work may consider improving the efficiency of the proposed ESO, *e.g.*, by accelerating the speed of online sampling for language models.

6 Conclusion

In this paper, we propose ESO, a simple yet well-motivated on-policy method, by drawing inspiration from the principle of biological evolution for language model tuning. In ESO, the gradient of the probability of output sentences generated online by the model serves as a sampling perturbation signal in the parameter space. Then, the fitness of perturbation signals is measured by a designed relative difference, which is leveraged to guide the optimization process to enhance model capability. The experimental results show that the proposed ESO can achieve superior performance in many scenarios, including instruction following and text summarization, and comparable performance to PPO in the human alignment task.

Limitations

One limitation of the proposed ESO is the requirement for the model to generate multiple candidate sentences during the training in an online manner. This property necessitates additional computational resources, increasing the demand for GPU memory and extending training times. Additionally, the proposed ESO relies on an accurate and robust reward mechanism to guide the optimization. Developing and selecting such a perfect reward model poses a substantial challenge in scenarios where deeper semantic or stylistic alignment is necessary, like advanced human alignment. Furthermore, another limitation arises in multi-objective alignment scenarios where multiple reward functions are involved, each corresponding to a specific objective. In such cases, effectively integrating and balancing these multiple reward signals becomes a significant challenge.

Acknowledgements

Hongyang Chen was supported in part by National Key Research and Development Program of China (No. 2022YFB4500300). Yi Wang was supported in part by Guangdong Basic and Applied Basic Research Foundation (No. 2023B1515120058). Wei Wang was supported by Guangdong Provincial Key Lab of Integrated Communication, Sensing and Computation for Ubiquitous Internet of Things (No. 2023B1212010007), Guangzhou Municipal Science and Technology Project (No. 2023A03J0003, 2023A03J0013 and 2024A03J0621), and Institute of Education Innovation and Practice Project (No. G01RF000012 and G01RF000017).

References

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

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

Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, et al. 2023. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*, pages 2397–2430. PMLR.

Mingyang Chen, Junda Lu, Yi Wang, Jianbin Qin, and Wei Wang. 2021. Dair: A query-efficient decision-based attack on image retrieval systems. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1064–1073.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E Gonzalez, et al. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality. *See https://vicuna. lmsys. org (accessed 14 April 2023)*, 2(3):6.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. 2023. Minillm: Knowledge distillation of large language models. In *The Twelfth International Conference on Learning Representations*.

Alois Huning. 1976. Evolutionsstrategie. optimierung technischer systeme nach prinzipien der biologischen evolution.

Yuxin Jiang, Chunkit Chan, Mingyang Chen, and Wei Wang. 2023. Lion: Adversarial distillation of closed-source large language model. *arXiv preprint arXiv:2305.12870*.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Adam Lopez. 2008. Statistical machine translation. *ACM Computing Surveys (CSUR)*, 40(3):1–49.

Kathleen McKeown. 1992. *Text generation*. Cambridge University Press.

Shashi Narayan, Shay B Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. *arXiv preprint arXiv:1808.08745*.

Romain Paulus, Caiming Xiong, and Richard Socher. 2018. A deep reinforced model for abstractive summarization. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. 2018. Improving language understanding by generative pre-training.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

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. *Advances in Neural Information Processing Systems*, 36.

I Rechenberg. 1973. Evolutionsstrategie, optimierung technischer systeme nach prinzipien der biologischen evolution, volume 15 von reihe problemata.

Tim Salimans, Jonathan Ho, Xi Chen, Szymon Sidor, and Ilya Sutskever. 2017. Evolution strategies as a scalable alternative to reinforcement learning. *arXiv preprint arXiv:1703.03864*.

Florian Schmidt. 2019. Generalization in generation: A closer look at exposure bias. *arXiv preprint arXiv:1910.00292*.

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

Tianxiang Sun, Xiaotian Zhang, Zhengfu He, Peng Li, Qinyuan Cheng, Xiangyang Liu, Hang Yan, Yunfan Shao, Qiong Tang, Shiduo Zhang, et al. 2024. Moss: An open conversational large language model. *Machine Intelligence Research*, pages 1–18.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Haifeng Wang, Hua Wu, Zhongjun He, Liang Huang, and Kenneth Ward Church. 2022a. Progress in machine translation. *Engineering*, 18:143–153.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2022b. Self-instruct: Aligning language models with self-generated instructions. *arXiv preprint arXiv:2212.10560*.

Sean Welleck, Ilia Kulikov, Stephen Roller, Emily Dinan, Kyunghyun Cho, and Jason Weston. 2019. Neural text generation with unlikelihood training. *arXiv preprint arXiv:1908.04319*.

Daan Wierstra, Tom Schaul, Tobias Glasmachers, Yi Sun, Jan Peters, and Jürgen Schmidhuber. 2014. Natural evolution strategies. *The Journal of Machine Learning Research*, 15(1):949–980.

Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8:229–256.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144.

Kaichun Yao, Libo Zhang, Tiejian Luo, and Yanjun Wu. 2018. Deep reinforcement learning for extractive document summarization. *Neurocomputing*, 284:52–62.

Hongyi Yuan, Zheng Yuan, Chuanqi Tan, Wei Wang, Songfang Huang, and Fei Huang. 2023. Rrhf: Rank responses to align language models with human feedback. *Advances in Neural Information Processing Systems*, 36.

Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Sainbayar Sukhbaatar, Jing Xu, and Jason Weston. 2024. Self-rewarding language models. *arXiv preprint arXiv:2401.10020*.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. 2022. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019a. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.

Wen Zhang, Yang Feng, Fandong Meng, Di You, and Qun Liu. 2019b. Bridging the gap between training and inference for neural machine translation. *arXiv preprint arXiv:1906.02448*.

Yao Zhao, Mikhail Khalman, Rishabh Joshi, Shashi Narayan, Mohammad Saleh, and Peter J Liu. 2022. Calibrating sequence likelihood improves conditional language generation. In *The Eleventh International Conference on Learning Representations*.

Daniel M Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. 2019. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*.

A Additional Background on Natural Evolution Strategies

Let $F(\theta)$ denote the objective function, where θ represents the parameters to be optimized. NES algorithms approach the optimization problem by maintaining a population of solutions modeled through a probability distribution over the parameter space. This distribution is denoted as $p_\psi(\theta)$ and is parameterized by ψ. The primary goal is to maximize the expected value of the objective function across the population; mathematically, this is expressed as $\mathbb{E}_{\theta\sim p_\psi}F(\theta)$. In NES, the population distribution $p_\psi(\theta)$ is typically instantiated as an isotropic multivariate Gaussian distribution, which allows us to write $\mathbb{E}_{\theta\sim p_\psi}[F(\theta)]$ in terms of a mean parameter vector θ (Salimans et al., 2017), can be displayed as:

$$\mathbb{E}_{\theta\sim p_\psi}F(\theta) = \mathbb{E}_{\epsilon\sim N(0,I)}F(\theta+\sigma\epsilon). \qquad (13)$$

With this setup, the objective can be regarded as a Gaussian-blurred version of the original one, *i.e.*, a smoothed version of $F(\theta)$.

B Comparison to PPO

In this section, we discuss ESO's advantages and disadvantages when compared to PPO in detail.

Advantages:

1. **Computational and Memory Efficiency:** The proposed ESO utilizes only two models during training—the policy model and the reward model. This significantly reduces computational overhead and memory requirements compared to Proximal Policy Optimization (PPO), which requires four distinct models: a policy model, a reward model, a critic model, and a reference model. The reduced

complexity of ESO lowers GPU memory demands and decreases training time, making it a more resource-efficient option.

2. **Simplified Hyperparameter Tuning:** PPO's framework necessitates tuning several hyperparameters, including importance sampling clipping, coefficients for generalized advantage estimation, and the learning rate for the critic model. In contrast, ESO simplifies this process to just tuning a single hyperparameter—the coefficient of ESO loss. This streamlining significantly eases the model's configuration and maintenance, thereby enhancing user-friendliness. Despite its simpler configuration, ESO demonstrates performance on the Human Alignment dataset comparable to that of PPO, proving its efficacy even with fewer hyperparameters and a reduced model architecture.

Disadvantage:

- The primary limitation of ESO, in comparison to PPO, arises in scenarios that necessitate the alignment of multiple objectives rather than a singular focus. While ESO is optimized for efficiency and simplicity, PPO's broader training paradigm is designed to handle complex and diverse objective functions, potentially offering better stability and robustness. This capability makes PPO more suitable for complex applications where achieving multi-objective alignment is crucial, indicating an area for potential enhancement in ESO to match the adaptability and versatility of PPO's approach.