Efficient Summarization with Small Language Models via Direct Preference Optimization

Youssef Tamer
Arab Academy for Science, Technology & Maritime Transport
Smart Village, Giza, Egypt
y.sadek17571@student.aast.edu

Ali Badawi
Arab Academy for Science, Technology & Maritime Transport
Smart Village, Giza, Egypt
a.ibrahim17855@student.aast.edu

Mohamed Bahgat
Technology and Computer Science
Nile University
Giza, Egypt
M.Bahgat2591@nu.edu.eg

***Abstract*—The deployment of large language models for text summarization is often constrained by substantial computational requirements, limiting their applicability in resource-constrained environments. This paper investigates the application of Direct Preference Optimization (DPO) for fine-tuning compact language models to achieve high-quality summarization performance. We provide a direct empirical comparison of DPO against a supervised fine-tuning (SFT) baseline on the Qwen3 0.6B model, demonstrating that DPO achieves a G-Eval average score of 0.758 compared to 0.662 for SFT, with particularly strong gains in factual consistency (0.913 vs. 0.640). Our DPO-fine-tuned model requires only 3 hours of training on consumer-grade hardware and generalizes effectively to out-of-domain data. These results establish that preference-based optimization enables small language models to produce summaries competitive with larger models while remaining deployable on edge devices and mobile applications.**

***Index Terms*—Text summarization, direct preference optimization, reinforcement learning from human feedback, low-rank adaptation, small language models, parameter-efficient fine-tuning**

I. Introduction

Text summarization remains a challenging problem in natural language processing, with applications spanning information retrieval, content curation, and knowledge management [1]–[3]. Recent advances in large language models (LLMs) have demonstrated impressive summarization capabilities across a wide range of benchmarks [4]–[6]. However, deploying such models in real-world systems is often constrained by high computational costs, memory requirements, and inference latency [5], [7]. This gap between capability and practicality motivates the development of efficient fine-tuning techniques for smaller language models that can deliver competitive performance while remaining deployable in resource-constrained environments [7], [8].

Traditional approaches to fine-tuning language models for summarization have primarily relied on supervised fine-tuning (SFT), where models are trained to maximize the likelihood of reference summaries [9], [10]. While straightforward to implement, SFT suffers from several limitations. First, it optimizes for token-level prediction accuracy rather than summary quality as perceived by human readers [11]. Second, it can lead to exposure bias, where models generate summaries that diverge from the distribution of human-written text [12]. Third, SFT struggles to capture nuanced preferences such as conciseness, factual accuracy, and stylistic appropriateness that characterize high-quality summaries [13]. Finally, SFT greatly impacts the prompting behavior of an LLM; even without overfitting, most LLMs tend to produce outputs similar to the fine-tuning dataset while ignoring human prompts, which limits the versatility of such models [11], [14].

Recent work in reinforcement learning from human feedback (RLHF) has shown that aligning language models with human preferences can greatly improve output quality across multiple tasks [11], [13], [15]. However, traditional RLHF approaches require training separate reward models and employing complex reinforcement learning algorithms, adding significant computational overhead and training instability [16], [17]. Direct Preference Optimization (DPO) addresses these limitations by directly optimizing models on preference data without the need for explicit reward modeling or reinforcement learning [18].

In this work, we investigate the application of DPO to fine-tune Qwen3 0.6B [23], a compact yet capable language model, specifically for text summarization. Prior work such as Zephyr [19] and Tulu 2 [20] has demonstrated DPO-based alignment on larger models; our work extends this to the sub-1B parameter regime with a direct SFT comparison. We make the following contributions:

- We provide a direct experimental comparison of DPO and SFT for summarization on a 0.6B parameter model, demonstrating DPO's advantage on factual consistency and overall quality as measured by both ROUGE and LLM-based G-Eval metrics.
- We provide a comprehensive analysis of best practices for applying DPO to summarization tasks, including data construction, hyperparameter selection, and training strategies for resource-constrained models.
- We demonstrate effective cross-domain generalization from Reddit posts to CNN/DailyMail news summarization without domain-specific fine-tuning.
- We introduce G-Eval LLM-as-a-judge evaluation to provide a more comprehensive quality assessment beyond ROUGE, demonstrating DPO's advantage on factual consistency and relevance.

Our findings suggest that combining efficient model architectures with preference-based fine-tuning techniques offers a promising path toward democratizing access to high-quality summarization systems. By demonstrating superior results with DPO on a 0.6B parameter model, we show that effective summarization need not require massive computational resources, opening new possibilities for deployment in edge devices, mobile applications, and resource-limited settings.

The remainder of this paper is organized as follows. Section 2 reviews related work on summarization, fine-tuning techniques, and preference optimization. Section 3 describes our DPO-based fine-tuning methodology and implementation details. Section 4 presents experimental results and comparative analysis with SFT baselines. Section 5 provides discussion and analysis of our findings. Finally, Section 6 concludes with implications and future research directions.

II. Related Work

A. Neural Text Summarization

Neural approaches to text summarization have evolved significantly over the past decade. Early sequence-to-sequence models with attention mechanisms [1], [2] established the foundation for neural abstractive summarization. The introduction of pre-trained transformer models such as BART [9] and PEGASUS [10] substantially improved summarization quality by leveraging large-scale pre-training followed by fine-tuning on summarization datasets. More recently, large language models such as GPT-3 [4] and PaLM [5] have demonstrated strong zero-shot and few-shot summarization capabilities, though their deployment remains challenging due to computational requirements.

B. Reinforcement Learning for Text Summarization

The application of reinforcement learning to text summarization was pioneered by OpenAI's seminal work on learning to summarize with human feedback [13]. Stiennon et al. [13] demonstrated that using Proximal Policy Optimization (PPO) [16] as a reinforcement learning technique achieved remarkable results on summarization tasks that significantly outpaced regular supervised fine-tuning. Their approach involved training a reward model from human preferences and then optimizing a policy using PPO with approximately one million episodes, where each episode consisted of summarizing a single article and receiving a reward. The resulting models generated summaries superior to those from models ten times larger trained only with supervised learning, demonstrating the substantial potential of preference-based optimization.

This foundational work established a three-stage framework that became the standard for aligning language models: supervised fine-tuning (SFT) on demonstrations, reward model training on human preferences, and policy optimization using PPO. The success of this approach was later extended to InstructGPT [11], which applied the same RLHF methodology to general instruction following, ultimately leading to the development of ChatGPT and other influential language models. However, while PPO-based RLHF demonstrated impressive results, it introduced significant complexity, computational overhead, and training instability into the model development pipeline.

C. Supervised Fine-Tuning and Its Limitations

Supervised fine-tuning has been the conventional approach for adapting pre-trained language models to downstream tasks. In SFT, models are trained to maximize the likelihood of reference outputs given input prompts using standard cross-entropy loss. While this approach is straightforward to implement and computationally efficient, it suffers from several fundamental limitations for tasks requiring nuanced quality judgments like summarization.

SFT has several key limitations for summarization. First, it optimizes for token-level likelihood of reference text rather than actual summary quality as perceived by human readers, causing models to reproduce surface-level patterns without capturing the underlying qualities that make summaries informative.

A more consequential limitation is that SFT encourages models to reproduce training distribution patterns, reducing responsiveness to user-specified instructions at inference time. Even in the absence of overfitting, SFT models tend to generate outputs consistent with training examples rather than adapt to prompt-specified constraints such as length, focus, or style [11], [14].

Finally, SFT optimizes against a single reference summary per input, preventing models from learning that multiple valid summaries may exist for the same source text and limiting output diversity.

D. Reinforcement Learning from Human Feedback

Reinforcement Learning from Human Feedback (RLHF) emerged as a ground-breaking paradigm for aligning language models with human preferences across various tasks. The standard RLHF pipeline consists of three stages: first, supervised fine-tuning on demonstration data; secondly, training a reward model to predict human preferences from comparison data; and finally, optimizing the policy against the reward model using reinforcement learning algorithms, typically done using PPO.

PPO became the default reinforcement learning algorithm at OpenAI due to its ease of use and competitive performance. The algorithm addresses the challenge of taking sufficiently large policy improvement steps without causing performance collapse by using a clipped surrogate objective that constrains policy updates. Despite its advantages, RLHF with PPO introduces substantial complexity and computational costs. The approach requires maintaining and training a separate reward model, sampling from the language model during training, extensive hyperparameter tuning, and managing the inherent instability of reinforcement learning optimization. Additionally, reward models can suffer from distribution shift and reward hacking, where the policy learns to exploit imperfections in the learned reward function.

E. Direct Preference Optimization

Direct Preference Optimization (DPO), introduced by Rafailov et al. [18], addresses the limitations of traditional RLHF by allowing the extraction of the optimal policy in closed form through a novel parameterization of the reward model. Rather than training a separate reward model and then optimizing it with reinforcement learning, DPO reformulates the RLHF objective as a simple binary classification problem on preference pairs. The resulting algorithm is stable, highly performance-oriented, and computationally cheap, eliminating the need for sampling from the language model during fine-tuning or extensive hyperparameter tuning.

The key insight behind DPO is that the optimal policy can be derived analytically from the reward function under the RLHF objective, allowing direct optimization of the policy using preference data. This change of variables approach means that the policy network implicitly represents both the language model and the reward function. Experimental results showed that DPO performs comparably or better than PPO-based RLHF on tasks including summarization and dialog, while being substantially simpler to implement and train.

DPO offers several advantages over traditional RLHF: improved stability due to the elimination of reinforcement learning dynamics, reduced computational overhead by avoiding reward model training and policy sampling, greater simplicity in implementation requiring only standard supervised learning infrastructure, and robustness to hyperparameter choices. The method is more computationally efficient compared to RLHF, achieving similar or better results with fewer resources.

F. Preference Optimization for Smaller Language Models

Recent work has demonstrated that preference-based optimization generalizes effectively to compact architectures. Zephyr [19] and Tulu 2 [20] showed that DPO-based alignment produces strong results on 7B-parameter models. DPO is particularly suited to resource-constrained settings due to its elimination of reward model training and policy sampling overhead [18], enabling deployment of aligned models on edge devices and mobile platforms. Our work extends this line of research to the sub-1B parameter regime.

Our work builds on this foundation by demonstrating that DPO can effectively fine-tune compact models like Qwen3 0.6B for summarization tasks, achieving competitive quality while maintaining the practical advantages of smaller model sizes. By applying preference optimization to an efficient architecture, we show that effective summarization need not require massive computational resources, extending the accessibility and applicability of high-quality summarization systems.

G. Low-Rank Adaptation (LoRA)

Low-Rank Adaptation (LoRA) [21] has emerged as a highly effective parameter-efficient fine-tuning technique for large language models. LoRA is based on the hypothesis that the weight updates during fine-tuning have a low intrinsic rank. Rather than updating all model parameters, LoRA injects trainable low-rank decomposition matrices into each layer of the transformer architecture while keeping the original pre-trained weights frozen.

For a pre-trained weight matrix $W_0 \in \mathbb{R}^{d\times k}$, LoRA constrains its update by representing it with a low-rank decomposition: $W_0 + \Delta W = W_0 + BA$, where $B \in \mathbb{R}^{d\times r}$ and $A \in \mathbb{R}^{r\times k}$, with the rank $r \ll \min(d, k)$. During training, W_0 is frozen and only A and B receive gradient updates. This dramatically reduces the number of trainable parameters—for large models, LoRA can reduce trainable parameters by 10,000 times and GPU memory requirements by 3 times compared to full fine-tuning [21].

The combination of LoRA with quantization techniques, particularly QLoRA [22], further enhances efficiency by enabling fine-tuning of quantized models. QLoRA uses 4-bit NormalFloat (NF4) quantization for the base model weights while maintaining LoRA adapters in higher precision, achieving comparable performance to full fine-tuning while drastically reducing memory requirements. This makes fine-tuning of large models feasible on consumer-grade hardware.

III. Methodology

A. Base Model: Qwen3-0.6B

- Number of Parameters (Non-Embedding): 0.44 billion
- Number of Layers: 28
- Number of Attention Heads: 16 query heads with 8 key-value heads (Grouped Query Attention)
- Maximum Context Length: 32,768 tokens

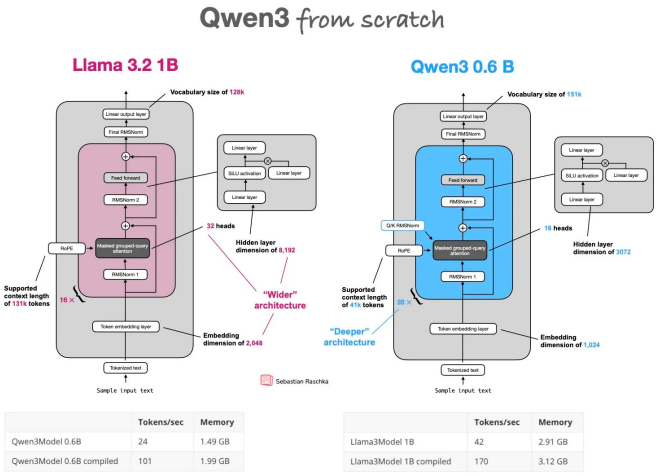

Fig. 1. Architecture overview of Qwen3 0.6B compared to similar sized model Llam 3.2 1B adapted. Figure adapted from [28].

We use the Qwen3-0.6B model [23] as our base architecture for this study. Specifically, we employ the variant provided by Unsloth (Qwen/Qwen3-0.6B), which was selected for its small size suitable for our task requirements and its demonstrated superior performance across benchmarks when compared to other small language models in the sub-1B parameter range.

Qwen3-0.6B is a compact transformer-based language model with the following architecture:

The model employs Grouped Query Attention (GQA), an efficient attention mechanism that reduces the number of key-value heads relative to query heads, thereby decreasing memory requirements and computational costs during inference while maintaining model quality. Despite its small architecture, Qwen3-0.6B incorporates modern design choices that enable strong performance across various natural language understanding and generation tasks, making it an ideal candidate for investigating the effectiveness of preference-based fine-tuning in smaller LLMs.

B. Dataset

For our experiments, we utilize the CarperAI/openai_summarize_comparisons dataset [13], which is specifically designed for preference-based training on summarization tasks. This dataset consists of triplets containing a source document prompt along with two candidate summaries: a chosen (preferred) summary and a rejected (less preferred) summary. This format directly aligns with the requirements of Direct Preference Optimization, which learns from pairwise preferences between model outputs.

The dataset contains 92,534 training examples and 86,957 test examples. For SFT training, we use only the prompt and chosen summary pairs. For DPO training, we use the full triplets of prompt, chosen, and rejected summaries. For evaluation, we sample 300 examples from the test split for preference-based metrics. Out-of-domain evaluation is conducted on 200 examples from the CNN/DailyMail test set [27], which was not seen during training.

The dataset provides human preference annotations over summary pairs, enabling the model to learn the nuanced qualities that distinguish high-quality summaries from inferior ones. These preferences capture multi-faceted aspects of summary quality including factual accuracy, coherence, conciseness, and overall informativeness—characteristics that are difficult to optimize for using standard supervised fine-tuning approaches.

C. Supervised Fine-Tuning Baseline

We train an SFT baseline by fine-tuning Qwen3-0.6B to maximize the likelihood of chosen (preferred) summaries given source prompts using standard cross-entropy loss. The SFT model uses an identical LoRA configuration and number of training steps as the DPO model, ensuring a controlled comparison. Only the prompt and chosen summary pairs are used for SFT training; rejected summaries are discarded.

D. Direct Preference Optimization

We employ Direct Preference Optimization (DPO) as our fine-tuning methodology. DPO optimizes the language model policy directly on preference data without requiring an explicit reward model or reinforcement learning procedure. Given a dataset of preferences, DPO maximizes the log-likelihood of the preferred completions being ranked higher than rejected completions according to an implicit reward function parameterized by the model itself.

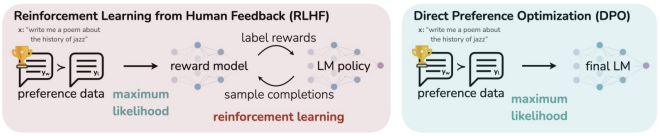

Fig. 2. Direct Preference Optimization training process. Unlike traditional RLHF, DPO directly optimizes the policy using preference pairs without requiring a separate reward model. Figure adapted from [18].

The DPO loss function is defined as:

$$\mathcal{L}_{\text{DPO}}(\pi_\theta; \pi_{\text{ref}}) = -\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}\left[\log\sigma\left(\beta\log\frac{\pi_\theta(y_w|x)}{\pi_{\text{ref}}(y_w|x)} - \beta\log\frac{\pi_\theta(y_l|x)}{\pi_{\text{ref}}(y_l|x)}\right)\right] \tag{1}$$

where π_θ is the model being trained, π_{ref} is a frozen reference model (the base Qwen3-0.6B), y_w and y_l are the chosen and rejected summaries respectively, x is the source document, β is a temperature parameter controlling the deviation from the reference policy, and σ is the logistic sigmoid function.

The key advantage of DPO for our application is its computational efficiency and stability compared to traditional RLHF approaches, making it particularly well-suited for fine-tuning smaller models where computational resources are limited.

E. Training Configuration

1) Quantization and Parameter-Efficient Fine-Tuning: To enable efficient training on consumer-grade hardware, we employ QLoRA (Quantized Low-Rank Adaptation) [22], a parameter-efficient fine-tuning technique that combines 4-bit quantization with low-rank adaptation [21]. We utilize the 4-bit quantized variant of Qwen3-0.6B provided by Unsloth, which uses the bitsandbytes library for NormalFloat4 (NF4) quantization.

LoRA is applied to the following transformer modules:

- Attention projections: `q_proj`, `k_proj`, `v_proj`, `o_proj`
- Feed-forward network projections: `gate_proj`, `up_proj`, `down_proj`

The LoRA configuration parameters are set as follows:

- Rank (r): 16
- LoRA alpha (α): 32
- LoRA dropout: 0.05

The rank of 16 provides a good balance between model training expense and parameter efficiency, allowing the model to learn task-specific adaptations while maintaining a small memory footprint. The alpha parameter of 32 scales the LoRA updates, and we apply a dropout rate of 0.05 to the LoRA layers to try and prevent overfitting.

We enable gradient checkpointing to further reduce memory consumption during training by trading computation for memory, allowing us to use larger effective batch sizes.

2) Optimization Hyperparameters: Our training configuration is designed to balance convergence speed, stability, and computational efficiency:

- **Training steps**: 3,000
- **Batch size per device**: 16
- **Gradient accumulation steps**: 4 (effective batch size: 64)
- **Learning rate**: 5×10^{-5}
- **Optimizer**: AdamW with 8-bit precision (adam_8bit)
- **Precision**: bfloat16 mixed precision training
- **Beta parameter** (β): 0.1 (standard DPO setting)

We use the 8-bit AdamW optimizer to reduce memory consumption while maintaining optimization quality. Training is conducted in bfloat16 precision. The gradient accumulation strategy allows us to achieve an effective batch size of 64 while maintaining acceptable memory requirements, which is crucial for stable DPO training.

The learning rate of 5×10^{-5} was selected based on earlier experiments and recommendation from unsloth documentation. The DPO beta parameter is set to 0.1, following the recommendations from the original DPO paper, which controls the strength of the KL penalty term that prevents the policy from deviating too far from the reference model.

F. Implementation Details

All experiments are implemented using PyTorch and the Hugging Face Transformers library [25]. We utilize the Unsloth library for efficient model loading and quantization, and the TRL (Transformer Reinforcement Learning) library for the DPO training implementation. Training is performed on a single NVIDIA RTX 4090 GPU, with total training time of approximately 3 Hours.

The combination of 4-bit quantization, LoRA parameter-efficient fine-tuning, gradient checkpointing, and 8-bit optimization enables us to fine-tune the model with limited computational resources while maintaining training stability and final model quality.

IV. Experiments

A. Evaluation Metrics

We evaluate models using three complementary approaches.

ROUGE metrics [26]: We report ROUGE-1, ROUGE-2, and ROUGE-L scores measuring n-gram overlap between generated and reference summaries.

BERTScore: We report BERTScore F1, which measures semantic similarity between generated and reference summaries using contextual embeddings, capturing meaning preservation beyond surface-level overlap.

G-Eval [29]: We employ G-Eval, an LLM-as-a-judge evaluation framework that uses chain-of-thought reasoning to assess summary quality across four dimensions: *Coherence* (logical structure and flow), *Consistency* (factual alignment with the source), *Fluency* (grammatical correctness), and *Relevance* (coverage of key information). Scores range from 0 to 1. We use GPT-5 via Azure OpenAI as the judge model on 20 samples from the CNN/DailyMail test set. G-Eval has been shown to correlate more strongly with human judgments than traditional n-gram metrics [29].

B. Results

Table I presents the ROUGE and BERTScore comparison across all three models. Table II presents G-Eval LLM-as-a-judge scores.

DPO-LoRA achieves the highest scores across nearly all metrics. The G-Eval results show a particularly strong DPO advantage in Consistency (0.913 vs. 0.640 for SFT), indicating that preference-based optimization produces summaries substantially more faithful to source content. Notably, SFT reduces win-rate below the base model (55.00% vs. 59.67%), consistent with the known tendency of SFT to reduce prompt responsiveness [11], [14]. DPO recovers this responsiveness (58.67%) while improving factual quality.

V. Discussion

A. Advantages of DPO Over Supervised Fine-Tuning

Our results demonstrate that Direct Preference Optimization offers substantial advantages over traditional supervised fine-tuning for summarization tasks. Beyond the quantitative improvements shown in our experiments, DPO preserves a critical capability that is often compromised in SFT approaches: controllability through natural language instructions.

A fundamental limitation of supervised fine-tuning is that models become rigidly aligned to the characteristics of the reference summaries in the training data. When trained with SFT, models learn to reproduce specific summary lengths, styles, and content selections present in the training examples, making them less responsive to user preferences expressed through prompts. For instance, if the training data contains summaries of a particular length, an SFT model may struggle to generate substantially longer or shorter summaries even when explicitly instructed to do so.

In contrast, our DPO-fine-tuned model retains strong responsiveness to prompt-based control while simultaneously improving summary quality. Users can specify desired summary characteristics—such as length constraints (”provide a brief one-sentence summary” or ”write a comprehensive paragraph-length summary”), focus areas (”summarize the main findings” or ”focus on the methodology”), or stylistic preferences (”write in simple language” or ”use technical terminology”)—and the model adjusts its output accordingly while maintaining the quality improvements learned through preference optimization.

This controllability emerges because DPO optimizes for preference rankings rather than forcing the model to imitate specific reference outputs. The model learns what makes one summary better than another across diverse contexts, rather than learning a single ”correct” way to summarize. This distinction is crucial for practical deployment, where users

TABLE I
PERFORMANCE COMPARISON ON SUMMARIZATION TASKS

Model	Win-Rate (%)	R-1 (pref)	R-1 (CNN)	R-2 (CNN)	R-L (CNN)	BERTScore
Base Model	59.67	0.1516	0.1577	0.0616	0.1109	0.8399
SFT-LoRA	55.00	0.1316	0.1606	0.0675	0.1163	0.8373
DPO-LoRA	**58.67**	**0.1917**	**0.1723**	**0.0679**	**0.1203**	**0.8419**

Win-Rate evaluated on 300 samples from OpenAI summarize comparisons test set. ROUGE and BERTScore evaluated on 200 CNN/DailyMail samples (out-of-domain).

TABLE II
G-EVAL LLM-AS-A-JUDGE SCORES ON CNN/DAILYMAIL (GPT-5 JUDGE, 20 SAMPLES)

Model	Coherence	Consistency	Fluency	Relevance	Avg
Base Model	0.220	0.293	0.247	0.333	0.273
SFT-LoRA	0.747	0.640	0.680	0.580	0.662
DPO-LoRA	**0.720**	**0.913**	**0.667**	**0.733**	**0.758**

DPO-LoRA achieves the highest average score, with a particularly strong advantage in Consistency (+0.273 over SFT), indicating superior factual alignment.

have varying needs and preferences that cannot be anticipated during training.

The preservation of prompt-following capabilities alongside quality improvements makes DPO particularly valuable for building flexible summarization systems that can adapt to different use cases without requiring task-specific fine-tuning. A single DPO-trained model can serve multiple applications—from generating short social media posts to creating detailed technical summaries—simply by varying the prompts, whereas SFT models typically require separate training for each summary style.

B. Computational Efficiency and Accessibility

The combination of DPO with parameter-efficient fine-tuning techniques demonstrates that high-quality preference alignment is achievable with limited computational resources. Our entire training process, including 3,000 optimization steps, completed in approximately 3 hours on a single consumer-grade RTX 4090 GPU. This computational efficiency stands in stark contrast to traditional RLHF approaches, which require training separate reward models, extensive sampling from the policy, and careful tuning of reinforcement learning hyperparameters.

The use of QLoRA further reduces resource requirements by enabling 4-bit quantization during training while maintaining model quality. By training only 0.29% of the model's parameters through low-rank adaptation, we achieve substantial performance improvements with minimal memory overhead. This parameter efficiency means that the fine-tuned model can be deployed with negligible additional storage costs compared to the base model.

These efficiency gains have important implications for democratizing access to high-quality language model alignment. Researchers and practitioners without access to large-scale compute infrastructure can apply DPO to improve model performance on their specific tasks and domains. The approach is particularly promising for specialized applications where domain-specific preference data is available but computational resources are limited.

C. Generalization to Out-of-Domain Data

The improvement on CNN/DailyMail news summarization, despite training only on Reddit posts from the OpenAI comparisons dataset, demonstrates notable cross-domain generalization. This suggests that the preference patterns learned through DPO capture generalizable aspects of summary quality rather than dataset-specific artifacts.

The model learns fundamental principles of what constitutes a good summary—such as faithfulness to source content, appropriate information selection, and coherent organization—that transfer across domains. This generalization capability reduces the need for domain-specific preference data, making DPO more practical for real-world applications where preference annotations may be scarce or expensive to obtain.

D. Limitations and Future Work

While our results are promising, several limitations warrant discussion. First, our evaluation relies primarily on a single preference dataset and one out-of-domain benchmark. More comprehensive evaluation across diverse summarization tasks, domains, and languages would provide stronger evidence of the approach's general applicability.

Secondly, we have not conducted extensive hyperparameter optimization for the DPO training process. While our chosen hyperparameters follow recommended settings from prior work, systematic grid search may yield further improvements. Utilizing techniques like grid-search could provide us with a better set of hyper-parameters further increasing out results.

Future work could explore several promising directions. Multi-task training with DPO across different types of text generation tasks could leverage shared preference patterns while maintaining task-specific capabilities. Investigating iterative DPO training, where models are fine-tuned on preferences over their own outputs, could potentially lead to continued quality improvements. Additionally, combining DPO with other alignment techniques, such as constitutional AI principles or chain-of-thought reasoning, may further enhance summary quality and controllability.

E. Implications for Practical Deployment

Our findings have important implications for deploying summarization systems in resource-constrained environments. The combination of a compact 0.6B parameter model with DPO fine-tuning enables deployment scenarios that would be impractical with larger models, including mobile applications, edge devices, and real-time processing pipelines where latency and resource constraints are paramount.

The retention of prompt-based controllability means that a single model can serve multiple use cases without requiring separate deployments or models for different summary types. This versatility reduces operational complexity and maintenance overhead in production systems.

Furthermore, the computational efficiency of DPO training enables rapid iteration and customization for specific domains or user populations. Organizations can fine-tune models on their own preference data to align with specific quality standards or stylistic requirements without requiring extensive computational infrastructure.

VI. Conclusion

We have presented a systematic comparison of DPO and SFT for fine-tuning a compact 0.6B parameter language model on text summarization using Direct Preference Optimization. Our experiments demonstrate that DPO achieves superior performance across G-Eval quality dimensions, particularly factual consistency (0.913 vs. 0.640 for SFT), while maintaining better prompt responsiveness than SFT.

The key advantage of our approach is the preservation of prompt-following capabilities, allowing users to control summary characteristics through natural language instructions. Combined with the ability to train on consumer-grade hardware in just around 3 hours, this makes high-quality summarization accessible for practical deployment in edge devices, mobile applications, and other resource-limited environments.

Future work should explore multi-task DPO training, end-to-end deployment of speech recognition and summarization pipelines on low-power devices, and evaluation across more diverse domains and languages to further validate and extend these findings.

Acknowledgment

The authors would like to thank the Unsloth team for providing optimized implementations of Qwen3-0.6B and the Hugging Face community for maintaining the tools and datasets that made this research possible.

References

[1] A. M. Rush, S. Chopra, and J. Weston, "A neural attention model for abstractive sentence summarization," in *Proc. Conf. Empirical Methods Natural Language Process.*, 2015, pp. 379–389.

[2] A. See, P. J. Liu, and C. D. Manning, "Get to the point: Summarization with pointer-generator networks," in *Proc. 55th Annu. Meeting Assoc. Comput. Linguistics*, vol. 1, 2017, pp. 1073–1083.

[3] W. Kryscinski, N. S. Keskar, B. McCann, C. Xiong, and R. Socher, "Neural text summarization: A critical evaluation," in *Proc. Conf. Empirical Methods Natural Language Process.*, 2019, pp. 540–551.

[4] T. Brown *et al.*, "Language models are few-shot learners," in *Advances Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 1877–1901.

[5] A. Chowdhery *et al.*, "PaLM: Scaling language modeling with pathways," *arXiv preprint arXiv:2204.02311*, 2022.

[6] J. Zhang, Y. Zhao, M. Saleh, and P. J. Liu, "PEGASUS: Pre-training with extracted gap-sentences for abstractive summarization," in *Proc. Int. Conf. Machine Learning*, 2020, pp. 11328–11339.

[7] P. Zhang, G. Zeng, T. Wang, and W. Lu, "TinyLlama: An open-source small language model," *arXiv preprint arXiv:2401.02385*, 2024.

[8] N. Muennighoff *et al.*, "OctoPack: Instruction tuning code large language models," *arXiv preprint arXiv:2308.07124*, 2023.

[9] M. Lewis *et al.*, "BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," in *Proc. 58th Annu. Meeting Assoc. Comput. Linguistics*, 2020, pp. 7871–7880.

[10] J. Zhang, Y. Zhao, M. Saleh, and P. J. Liu, "PEGASUS: Pre-training with extracted gap-sentences for abstractive summarization," in *Proc. Int. Conf. Machine Learning*, 2020, pp. 11328–11339.

[11] L. Ouyang *et al.*, "Training language models to follow instructions with human feedback," in *Advances Neural Inf. Process. Syst.*, vol. 35, 2022, pp. 27730–27744.

[12] M. Ranzato, S. Chopra, M. Auli, and W. Zaremba, "Sequence level training with recurrent neural networks," in *Proc. Int. Conf. Learning Representations*, 2016.

[13] N. Stiennon *et al.*, "Learning to summarize from human feedback," in *Advances Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 3008–3021.

[14] C. Zhou *et al.*, "LIMA: Less is more for alignment," in *Advances Neural Inf. Process. Syst.*, vol. 36, 2023.

[15] Y. Bai *et al.*, "Constitutional AI: Harmlessness from AI feedback," *arXiv preprint arXiv:2212.08073*, 2022.

[16] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[17] D. M. Ziegler *et al.*, "Fine-tuning language models from human preferences," *arXiv preprint arXiv:1909.08593*, 2019.

[18] R. Rafailov, A. Sharma, E. Mitchell, S. Ermon, C. D. Manning, and C. Finn, "Direct preference optimization: Your language model is secretly a reward model," in *Advances Neural Inf. Process. Syst.*, vol. 36, 2024.

[19] L. Tunstall *et al.*, "Zephyr: Direct distillation of LM alignment," *arXiv preprint arXiv:2310.16944*, 2023.

[20] H. Ivison *et al.*, "Camels in a changing climate: Enhancing LM adaptation with Tulu 2," *arXiv preprint arXiv:2311.10702*, 2023.

[21] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen, "LoRA: Low-rank adaptation of large language models," in *Proc. Int. Conf. Learning Representations*, 2022.

[22] T. Dettmers, A. Pagnoni, A. Holtzman, and L. Zettlemoyer, "QLoRA: Efficient finetuning of quantized LLMs," in *Advances Neural Inf. Process. Syst.*, vol. 36, 2023.

[23] A. Yang *et al.*, "Qwen3 technical report," *arXiv preprint arXiv:2505.09388*, 2025.

[24] J. Ainslie, J. Lee-Thorp, M. de Jong, Y. Zemlyanskiy, F. Lebrón, and S. Sanghai, "GQA: Training generalized multi-query transformer models from multi-head checkpoints," in *Proc. Conf. Empirical Methods Natural Language Process.*, 2023, pp. 4895–4901.

[25] T. Wolf *et al.*, "Transformers: State-of-the-art natural language processing," in *Proc. Conf. Empirical Methods Natural Language Process.: Syst. Demonstrations*, 2020, pp. 38–45.

[26] C.-Y. Lin, "ROUGE: A package for automatic evaluation of summaries," in *Text Summarization Branches Out*, Barcelona, Spain, 2004, pp. 74–81.

[27] K. M. Hermann *et al.*, "Teaching machines to read and comprehend," in *Advances Neural Inf. Process. Syst.*, vol. 28, 2015.

[28] S. Raschka, *Build a Large Language Model (From Scratch)*. Manning Publications, 2024. [Online]. Available: https://github.com/rasbt/LLMs-from-scratch

[29] Y. Liu *et al.*, "G-Eval: NLG evaluation using GPT-4 with better human alignment," in *Proc. Conf. Empirical Methods Natural Language Process.*, 2023, pp. 2511–2522.