Aligning Language Models with Human Preferences

Martim Filipe Almeida Santos

Thesis to obtain the Master of Science Degree in

Computer Science and Engineering

Supervisors: Prof. André Filipe Torres Martins
Dr. Sweta Agrawal

Examination Committee

Chairperson: Prof. António Paulo Teles de Menezes Correia Leitão
Supervisor: Prof. André Filipe Torres Martins
Member of the Committee: Prof. Alberto Abad Gareta

October 2024

Declaration
I declare that this document is an original work of my own authorship and that it fulfills all the requirements of the Code of Conduct and Good Practices of the Universidade de Lisboa.

Acknowledgments

I want to sincerely thank everyone who supported and guided me while I worked on this master's thesis. The journey was both challenging and rewarding, and I couldn't have done it without the help of many individuals and organizations.

First and foremost, I would like to express my sincere gratitude to my advisors, Prof. André F. T. Martins and Dr. Sweta Agrawal. Their priceless guidance, insightful feedback, and constant encouragement have been instrumental in shaping this research. Their expertise and dedication have inspired me to aim for excellence, and their patience and understanding have made this challenging journey achievable. I am deeply thankful for the countless hours they invested in reviewing my work, offering constructive criticism, and pointing me in the right direction whenever I faced obstacles.

I'm also very thankful to Unbabel and the Sardine Lab for their generosity and support. By providing access to their resources, they greatly enhanced the quality of my research.

My heartfelt appreciation goes out to my parents and brother, whose unwavering love and support have been a pillar of strength throughout my academic pursuits. Their belief in me has been the driving force behind my endeavors. I am eternally grateful for their encouragement during times of doubt and for always being there to celebrate my achievements.

I would also like to extend my gratitude to my entire family for their constant support and understanding. Their words of encouragement and genuine interest in my work have been a source of motivation and joy.

To all my friends, thank you for your companionship and for bringing balance to my life during this demanding period. Your friendship, laughter, and the moments we shared during this adventure have been invaluable. Your support has not only enriched my personal life but has also positively impacted my academic journey.

This thesis is a testament to the collective support, guidance, and love I have received from all of you.

To each and every one of you – Thank you.

Abstract

Large language models (LLMs) are characterized by their remarkable ability to learn extensive world knowledge and generate human-like text across diverse applications. However, the generated text often contains misleading and toxic content, emphasizing the need to align LLMs with human values and preferences to ensure more useful and secure AI systems. A widely employed strategy in numerous prominent models, including OpenAI's GPT-3.5 and GPT-4, involves Reinforcement Learning from Human Feedback (RLHF). While this method has demonstrated impressive outcomes, RLHF's complexity, instability, and sensitivity to hyperparameters challenge its empirical success and usability across various real-life scenarios. Recent reinforcement learning-free (RL-free) approaches — such as DPO, CPO, SimPO, and SLiC — address these issues. In this study, we investigate whether the promising results of RL-free methods observed in larger models extend to small language models (SLMs). Focusing on machine translation and summarization, we assess the ability of these models to efficiently learn human preferences by evaluating the quality and human alignment of their outputs, as well as their capacity to avoid common biases. Specifically, we train three compact baseline models — TinyLlama 1.1B, Gemma-2 2B, and EuroLLM 1.7B — with several RL-free methods and compare their performance against baselines. By evaluating the effectiveness of RL-free methods on smaller LLMs, this work is the first to provide a comprehensive comparison of several feedback methods applied to state-of-the-art small language models (SLMs), contributing to the development of secure and accessible AI systems suitable for resource-constrained environments.

Keywords

Language Models · Fine-tuning · Human Preferences · Alignment · Machine Translation · Summarization

Resumo

Os grandes modelos de linguagem (LLMs) destacam-se pela sua notória capacidade de aprender vasto conhecimento e gerar texto semelhante ao humano em diversas tarefas. No entanto, o texto gerado contém frequentemente erros e conteúdo tóxico, o que incentiva a necessidade de melhorar estes modelos usando valores e preferências humanas para garantir sistemas de IA mais úteis e seguros. Uma estratégia amplamente utilizada em vários modelos conhecidos, incluindo o GPT-3.5 e o GPT-4 da OpenAI, envolve Aprendizagem por Reforço através de Feedback Humano (RLHF). Apesar de bons resultados, a sua complexidade, instabilidade e sensibilidade relativamente aos hiperparâmetros desafiam o seu sucesso empírico e a sua usabilidade em diversos cenários. Abordagens mais recentes que não utilizam aprendizagem por reforço (RL-free) — como DPO, CPO, SimPO e SLiC — resolvem estes problemas. Neste trabalho, investigamos se os resultados promissores demonstrados pelos métodos RL-free em modelos maiores se estendem para modelos de linguagem pequenos (SLMs). Focando-nos nas tarefas de tradução automática e sumarização, avaliamos a capacidade destes modelos aprenderem eficientemente preferências humanas, avaliando a qualidade e alinhamento humano dos seus *outputs*, bem como a capacidade de evitar erros comuns. Especificamente, treinamos três modelos base — TinyLlama 1.1B, Gemma-2 2B e EuroLLM 1.7B — com vários métodos RL-free e avaliamos o seu desempenho. Ao avaliar a eficácia dos métodos RL-free em LLMs menores, este trabalho torna-se a primeira comparação detalhada de vários métodos de feedback aplicados a SLMs, contribuindo para o desenvolvimento de sistemas de IA seguros e acessíveis, adequados para ambientes com recursos limitados.

Palavras Chave

Modelos de Linguagem · Refinamento · Preferências Humanas · Alinhamento · Tradução Automática · Sumarização

Contents

List of Figures

List of Tables

Acronyms

CPO	Contrastive Preference Optimization
DPO	Direct Preference Optimization
LLM	Large Language Model
LM	Language Model
MT	Machine Translation
NLP	Natural Language Processing
PO	Preference Optimization
PPO	Proximal Policy Optimization
PRO	Preference Ranking Optimization
RLHF	Reinforcement Learning from Human Feedback
RM	Reward Model
RRHF	Rank Responses with Human Feedback
SFT	Supervised Fine-Tuning
SLiC	Sequence Likelihood Calibration
SLM	Small Language Model
SimPO	Simple Preference Optimization

1

Introduction

Contents

Large Language Models (LLMs) represent a significant stride in Artificial Intelligence, particularly in the realm of language processing. These deep learning models are characterized by their remarkable ability to learn extensive world knowledge and generate human-like text across diverse applications. Prominent examples such as the famous OpenAI's GPT series (including GPT-3.5 Brown et al. (2020) and GPT-4 (OpenAI, 2023)), Google's PaLM (Anil et al., 2023), and Meta's LLaMA (Touvron et al., 2023b), stand out due to their very large neural networks, containing billions of parameters. This architectural magnitude empowers them to capture and learn language intricacies, encompassing language patterns, syntax, and semantics on a large scale. This proficiency is attained through the utilization of enormous datasets during training, only possible with powerful computational resources. Essentially, LLMs excel in predictive tasks due to its training objective of predicting the next token, very effective for tasks such as translation, summarization, and even creative content generation.

The full potential of LLMs is realized when fine-tuned for specific tasks. This process consists of exposing the pre-trained language model to task-specific data, allowing it to adjust internal parameters and adapt to downstream tasks. The model is then tailored to generate more accurate and contextually relevant outputs for targeted applications. One approach consists of fine-tuning the pre-trained models in a supervised manner using labeled data, known as supervised fine-tuning (SFT). A specific strategy that has garnered attention is instruction tuning (Wei et al., 2021). By leveraging datasets enriched with instructions and corresponding human-generated completions, this approach has demonstrated improved model usability and generalization (Chung et al., 2022). While these strategies have led to markedly improved performance, they still encounter challenges that extend to various applications. One of the primary issues is the misalignment between maximizing the log probability of demonstrations and the more nuanced expectation of generating high-quality, safe outputs (Stiennon et al., 2020). Additionally, models may inadvertently amplify biases from the training data, leading to potentially low-quality and harmful outputs. Collecting high-quality, supervised samples is also costly, particularly when it requires expert involvement (Dong et al., 2023). Finally, models may also struggle to generate contextually appropriate content aligned with societal values, thereby limiting their overall effectiveness and acceptance.

Addressing the above-mentioned challenges is important for the practical applicability and real-world usefulness of these models. This entails going beyond conventional supervised fine-tuning approaches to ensure a safer and more harmonious alignment with the nuanced aspects of human expectations and preferences. **Incorporating human preferences** into large language models is a commonly used step for the ethical and effective deployment of such systems. It ensures that the generated content is not only accurate but also aligns with societal values, mitigating the risk of inadvertently producing harmful or biased outputs. Additionally, the integration of human feedback contributes to continuous model improvement, fostering a dynamic and adaptive system that can evolve with changing linguistic nuances and user expectations.

Collecting human feedback is then essential to align the model's behavior with human preferences, although it presents challenges in balancing harmlessness and helpfulness (Bai et al., 2022b). Harmlessness involves avoiding undesirable outputs, while helpfulness is gauged through task-related feedback. Some works explicitly divide the problem of alignment of these models into improving its helpfulness and increasing its harmlessness. The format and type of feedback - whether numerical, ranking-based, natural language, or other types - significantly influence on the expressivity, ease of collection and cost, ultimately affecting the models' performance.

To fully **leverage human feedback**, it is also important to guarantee that the model captures the nuanced details, rankings, and preferences of human evaluators accurately. For instance, using a reward function to evaluate the outputs generated by the model, presents some challenges as the reward model itself is a learned proxy for human preferences. If the reward model is not accurate or lacks granularity, it fails to completely capture human preferences and nuances. It is also vital to guarantee that

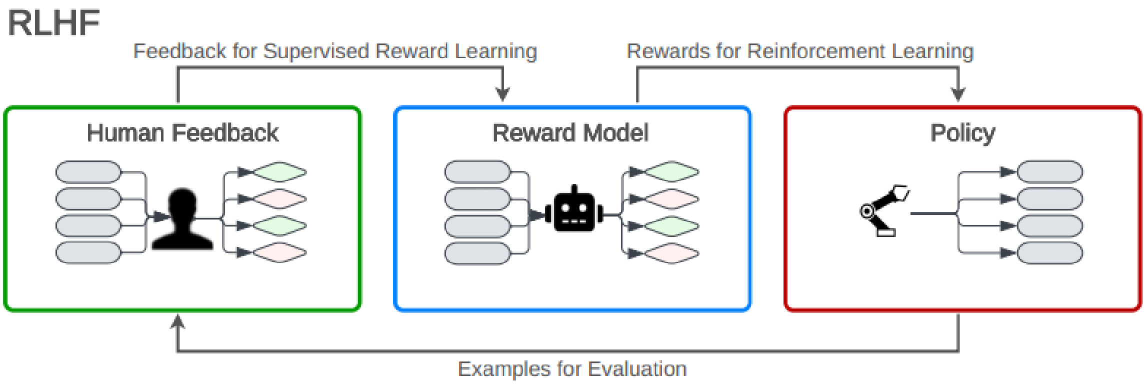
RLHF
Feedback for Supervised Reward Learning
Rewards for Reinforcement Learning
Human Feedback
Reward Model
Policy
Examples for Evaluation

Figure 1.1: Reinforcement Learning from Human Feedback. Gray, rounded boxes correspond to outputs (e.g., text), and colored diamonds correspond to evaluations. Figure taken from Casper et al. (2023).

the models are cost-effective in terms of training and data efficiency, avoiding excessive computational complexity and memory usage. Furthermore, modeling human preferences accurately requires careful consideration of both positive and negative examples, as it improves the model's ability to identify and avoid unwanted behaviors (Song et al., 2023) and enhances its overall quality and safety (Thoppilan et al., 2022).

Reinforcement Learning from Human (or AI) Feedback (RLHF/RLAIF) (Christiano et al., 2017; Bai et al., 2022c) marks a significant departure from traditional methods in refining LLMs, introducing a systematic approach that incorporates human preferences and evaluations. This proves especially valuable in various natural language processing tasks such as abstractive English text summarization, where judging summary quality is difficult without human judgments. RLHF methods fits a reward model that reflects human preferences, and then the language model is incentivized or "rewarded" when it produces outputs that align with the desired criteria. Reinforcement learning techniques, such as Proximal Policy Optimization (PPO) (Schulman et al., 2017), are then applied to maximize this estimated reward without drifting too far from the original model. This process enhances the models' capabilities by shaping their learning trajectory to improve the quality and appropriateness of their generated responses. Figure 1.1 depicts RLHF's three key steps: collecting human feedback, training a reward model using the collected feedback, and optimizing the LLM's policy with RL using the reward model trained. This methodology addresses challenges related to context understanding, deciphering hidden meanings, and aligning outputs with ethical considerations and user preferences.

Despite the advantages of RLHF in refining large language models, there are notable drawbacks that warrant consideration, namely its increased complexity, possible instability, and sensitivity to hyperparameters. Nonetheless, the primary objective remains the alignment of models with human preferences, to produce useful, safe, and accurate outputs.

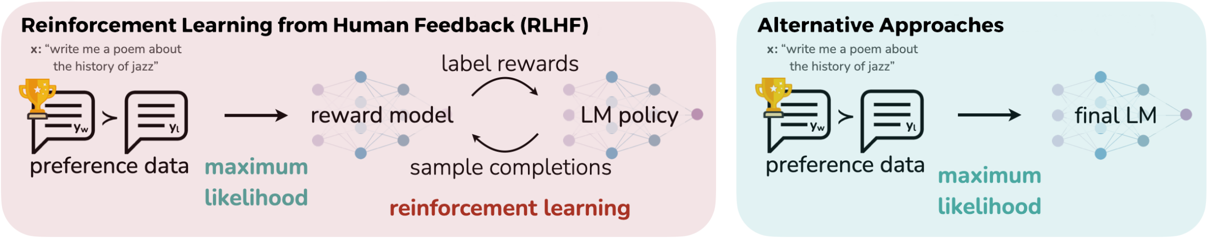

Figure 1.2: Alternative approaches optimizing for human preferences while avoiding reinforcement learning. RLHF involve first training a reward model on human preferences for response pairs, then using reinforcement learning to maximize this reward. In contrast, the alternative approaches simplifies this process by directly optimizing a policy without the need for a reward model. Figure based on (Rafailov et al., 2023).

As the development of more sophisticated models progresses, it is crucial to simultaneously maintain simplicity, stability, decreased sensitivity to hyperparameters, efficiency, scalability, and parallelizability. This comprehensive approach helps to mitigate the inherent challenges of RLHF, such as its tendency toward instability, while enhancing the models' robustness and reliability in real-world applications. Achieving a balance between addressing RLHF's limitations and optimizing model performance is essential for creating NLP systems that align with human preferences, minimize harmful outputs, and excel in a variety of practical scenarios.

The proposal of alternative algorithms which do not require RL, such as **DPO** (Direct Preference Optimization) (Rafailov et al., 2023), **RSO** (Statistical Rejection Sampling Optimization) (Liu et al., 2023), **SimPO** (Simple Preference Optimization) (Meng et al., 2024), **CPO** (Contrastive Preference Optimization) (Xu et al., 2024b), **PRO** (Preference-based Reinforcement Optimization) (Song et al., 2023), and **SLiC** (Sequence Likelihood Calibration) (Zhao et al., 2023) underscores a strategic shift in the pursuit of effective language model training. Traditional RLHF methods involve the intermediary step of learning a reward model using pairwise preference comparisons before training the policy. However, these recent paradigms deviate from this norm, aiming for more direct optimization based on human preferences while maintaining efficiency, simplicity, and robustness. An image illustrating these new approaches is presented in Figure 1.2. Furthermore, they demonstrate their effectiveness by showing competitive performance relative to RLHF methods across a range of tasks.

DPO and SLiC, for instance, bypass learning a reward model, matching traditional RLHF methods in various NLP tasks. DPO analyzes RLHF's objective function in the form of KL-regularized reward maximization, and analytically solves for the optimal policy induced by a reward function. SLiC utilizes a contrastive loss analogous to a hinge loss to effectively rank responses based on human feedback. RSO improves DPO by employing rejection sampling to sample directly from the optimal policy. PRO extends the pairwise Bradley-Terry (Bradley and Terry, 1952) comparison to accommodate arbitrary-length preference rankings, teaching the LLM to prioritize responses that closely align with human preferences and fully leverage the information provided by human rankings. Details about these models are given in Section 2.6.

1.1 Research Objectives

Despite the innovative strides above and the rapid development of alternative approaches, some problems and challenges persist. One common hurdle is efficiently handling diverse preferences and ensuring effective exploration during training, in various generation tasks at different model scales. A notable research gap is the lack of systematic comparison of preference optimization (PO) methods and their application to smaller language models.

Systematic comparison of PO methods allow to understand the conditions under which different PO methods are effective. Furthermore, applying these methods to smaller language models is particularly important as these models often offer a balance between computational efficiency and performance capability.

Small language models[1] (SLMs), despite their widespread adoption in modern smart devices, have received significantly less attention compared to their large language model (LLM) counterparts. Even with limited capacity, these models have demonstrated a competitive advantage over larger models in reasoning tasks (Fu et al., 2023; Magister et al., 2023). Yet, their application in natural language generation tasks remains underexplored. The effectiveness of preference alignment methods specifically tailored for these smaller models has not been comprehensively explored. Addressing this research gap is important to ensure that SLMs not only perform effectively but also align closely with human preferences, enhancing their applicability and acceptance in practical settings. This process involves both enhancing these models' capabilities and validating their performance to ensure they match human preferences.

This work aims to bridge this gap by being **the first to provide a comprehensive comparison of several feedback strategies when applied to state-of-the-art SLMs**. We explore how advanced architectures for training language models based on human preferences can be effectively implemented to improve their results and better align with human preferences while avoiding the drawbacks of traditional Reinforcement Learning from Human Feedback (RLHF). This exploration is particularly pertinent in scenarios characterized by constrained resources and the emergence of smaller models, which remain unexplored.

Previous research on preference algorithms has primarily focused on larger models, with sizes ranging from 6B to 13B parameters. Studies such as those by Rafailov et al. (2023), which utilized a GPT-J-6B (Wang and Komatsuzaki, 2021) for summarization tasks, indicate that DPO is at least as effective

[1] The definition of "small" is subjective and relative. We consider the upper limit of 5B parameters as for the size of SLMs, as done in Lu et al. (2024)

as traditional PPO-based RLHF methods. Similarly, Meng et al. (2024) employed two larger models, the Llama3-8B (AI@Meta, 2024) and Mistral-7B (Jiang et al., 2023), for SimPO results yet no findings were reported for smaller models. Further, Xu et al. (2024b) concentrated on 13B models proposing CPO, specifically using the ALMA-13B-LoRA as an initial checkpoint, but also showing significant benefits for 7B models. Research by Song et al. (2023) on PRO also utilized the popular LLaMA-7B (Touvron et al., 2023b) as a backbone model.

Our study seeks to determine whether the promising results of RL-free methods observed in larger models can be replicated in smaller language models. Focusing on the language tasks of machine translation and summarization, we assess the ability of these models to efficiently learn human preferences by evaluating the quality and human-alignment of their outputs, as well as their capacity to avoid common biases.

1.2 Main Contributions

Building upon our objectives and findings, the main contributions of this work are:

1. **Comprehensive Study of Small LLMs:** Conduct a detailed study of RL-free approaches on small LLMs, focusing on the machine translation, and summarization tasks. This includes the application of SFT and Preference Training using commonly used Preference Optimization methods such as DPO, CPO, SimPO, and SLiC within the context of Small Language Models.

2. **Evaluation of Benchmarks and Human Preferences:** Investigate how standard benchmarks for evaluating model performance in these tasks correlate with human preferences for utility and safety, particularly when applied to smaller models.

3. **Comparative Analysis Across Model Scales:** Perform an in-depth analysis of the outcomes from small models and compare these results, not only against those from larger models but also against similarly sized baseline models without additional training. This comparison aims to evaluate the consistency and reproducibility of results across different model scales.

4. **Bias Analysis in SLMs:** Analyze potential biases inherent in smaller models due to their limited size and constrained learning capabilities, discussing implications for their deployment and use in real-world applications.

1.3 Organization of the Document

This thesis is organized as follows: Chapter 2 compiles the background topics that provide the background necessary to understand the work developed in this thesis and also summarizes the existing related work. Chapter 3 presents the foundation models and datasets used in the training and evaluation of our models. Chapter 4 extensively describes the development process and the approaches applied during the training. Chapter 5 defines the evaluation criteria used to assess our models and discusses the results achieved in the different experiments, focusing on both performance and alignment with human preferences. Finally, Chapter 6 summarizes the key findings and proposes potential directions for future work.

2

Background and Related Word

Contents

In recent years, the field of NLP has undergone a revolutionary transformation with the introduction of LLMs, particularly those pre-trained on massive datasets. These models have propelled advancements in various NLP tasks such as machine translation, sentiment analysis, and text generation by leveraging deep learning architectures that can capture complex language patterns.

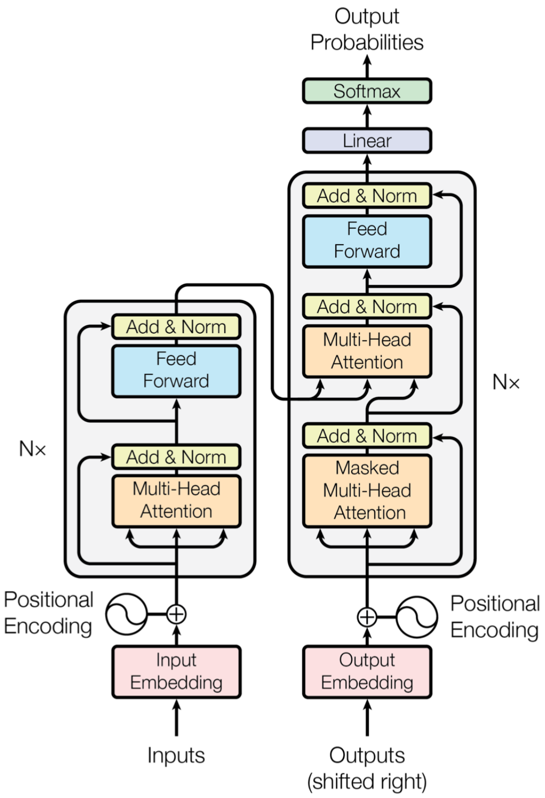

Figure 2.1: Transformer architecture. It consists of two main components: the encoder (left) and the decoder (right). This figure was taken from Vaswani et al. (2017).

2.1 Large Language Models (LLMs)

Language models consist of artificial neural networks designed to understand and generate human-like language. This involves ascertaining the probability of a specific word occurring within the model's vocabulary, given all previous words. The subsequent predicted word in the sequence is typically determined by selecting the one with the highest probability or by sampling according to this probability distribution. Formally, the probability of a sequence of words $y_1, y_2, ..., y_T$ given an input context x can be expressed using the chain rule of probability:

$$P(y_1, ..., y_T \mid x) = \prod_{t=1}^{T} P(y_t \mid x, y_1, ..., y_{t-1}) \tag{2.1}$$

A LLM is an advanced type of language model characterized by its numerous parameters. They are usually enormous neural networks that contain billions of parameters, enabling them to learn language patterns, syntax, and semantics on an extensive scale. LLMs achieve this through the use of massive datasets during training, coupled with powerful computational resources. Current state-of-the-art LLMs employ the Transformer architecture (Vaswani et al., 2017), depicted in Figure 2.1, which employs layers with multi-head attention mechanisms combined with feed-forward networks, enabling efficient capture of contextual relationships in sequential data.

Notable examples of LLMs include OpenAI's GPT models (e.g. GPT-3.5 (Brown et al., 2020) and GPT-4 (OpenAI, 2023), used in ChatGPT), Google's PaLM (used in Bard) (Chowdhery et al., 2023; Anil et al., 2023), and Meta's LLaMA (Touvron et al., 2023b; AI@Meta, 2024).

2.2 Pre-training

Pre-training is the foundational stage in the development and training process of LLMs, where the model learns from vast and diverse text data. During this phase, the model engages in self-supervised learning, capturing the contextual relationships and semantic nuances present in the data. It typically employs next-token prediction, a method where the model forecasts the next word based on the preceding context, as explained above. This extensive pre-training on massive datasets enables LLMs to capture a broad spectrum of linguistic knowledge, making them versatile and capable of handling various downstream natural language processing tasks effectively.

2.3 Fine-Tuning

Fine-tuning is an important process that adapts pre-trained models to specific tasks, leveraging the knowledge acquired during unsupervised pre-training. This approach enables efficient transfer learning, allowing models to specialize in particular domains or applications by exposing them to task-specific datasets and information. The benefits of fine-tuning include increased data efficiency, faster training cycles, and the incorporation of domain-specific knowledge. Ultimately, fine-tuning optimizes model performance, making it well-suited for diverse downstream tasks and facilitating the integration of advanced language understanding into various applications.

2.3.1 Supervised Fine-Tuning

SFT involves refining a pre-trained language model using labeled data tailored for a specific task. This process aims to improve the model's performance on targeted objectives by leveraging task-specific labeled datasets containing input examples and their corresponding desired outputs.

Instruction tuning is a specialized technique to tailor LLMs to perform specific tasks based on explicit instructions. While traditional supervised fine-tuning involves training a model on labeled task-specific data, instruction fine-tuning goes further by incorporating high-level instructions or demonstrations to guide the model's behavior. Through exposure to a variety of instructions, the model develops robust generalization skills, thereby improving its capacity to generate accurate responses that align with human-like instruction formats.

2.3.2 Fine-Tuning Challenges

The efficacy of the fine-tuning process has some challenges, especially in intricate tasks such as machine translation, text summarization, or dialogue. The main issue arises from a misalignment: while the fine-tuning goal typically focuses on maximizing the log probability of human demonstrations, it often fails to capture subtle but critical details that are necessary for producing high-quality outputs. Supervised fine-tuning only teaches LLMs about the best responses and cannot provide fine-grained comparisons to suboptimal ones. This misalignment becomes particularly conspicuous when the task requires a nuanced perception of context and the synthesis of information.

Although advantageous, the fine-tuning process also encounters the inherent difficulty of distinguishing between critical and less significant errors. Take, for instance, a machine translation task aimed at maximizing the log probability of aligning with a professional human translation. If translating the phrase ”I'm starving” from English to French, the model might produce ”J'ai très faim” (”I am very hungry”), which is correct but lacks the emotional depth conveyed by ”Je meurs de faim” (”I'm dying of hunger”), a phrase that carries stronger emotional weight and is more culturally nuanced in French. Despite its benefits, the models face the challenge of dealing with the complexities of subtle language nuances, where the difference between a precisely crafted output and a less optimal one can be subtle yet impactful. The challenge intensifies as models, during fine-tuning, tend to assign probability mass to all human demonstrations, including those of low quality. This arises from the fact that the data used to train these models is generally scraped from the Internet, often containing noise, social biases, toxicity, and errors.

This inclination to treat all examples equally can compromise the model's ability to discern and prioritize the finer nuances necessary for generating high-quality and safe answers. For the output to be both harmless and truly helpful, a model should generate outputs considering both these parameters.

Moreover, the fine-tuning process contends with the challenge posed by distributional shifts during sampling. These shifts can introduce anomalies and discrepancies that were not present in the original training data, leading to a potential degradation in overall performance.

Addressing all these challenges is important for the practical applicability and real-world utility of large language models. It necessitates methodologies that move beyond traditional fine-tuning objectives, seeking to align model outputs more closely with human expectations and preferences.

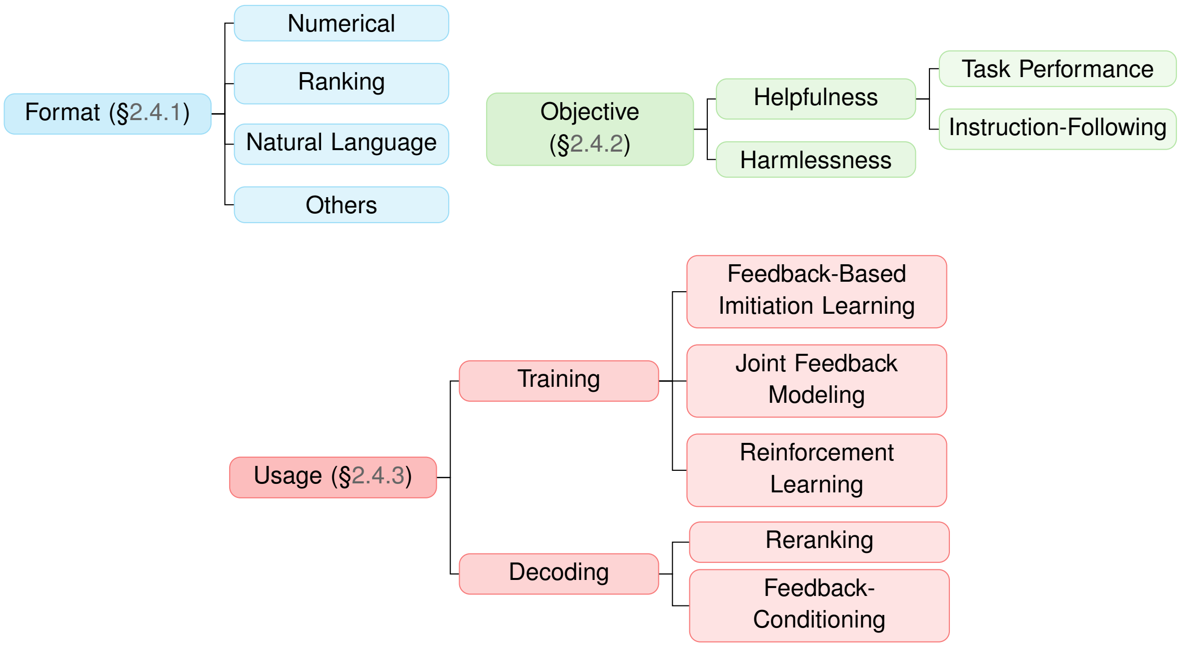

Figure 2.2: Taxonomy of methods that leverage human-feedback, inspired by an illustration in Fernandes et al. (2023).

2.4 Leveraging Human Feedback

Leveraging human feedback effectively requires understanding its format and the approaches through which it can be used to align model behavior with human preferences. Various feedback formats, their impact on model training and performance, and their role in refining outputs to better meet human expectations are explored. A summary is provided in Figure 2.2.

2.4.1 Feedback Format

When considering human feedback, it is important to decide in what *format* this feedback must be collected (Fernandes et al., 2023). While simpler formats are often easier to collect and use as part of the training process, they contain less information than more “complex” formats. Ghosal et al. (2023) further highlight that the choice of format also influences how easily humans can provide feedback, the consistency/agreement of that feedback, and the level of rationality within it.

Numerical feedback, which takes an input and an output and returns a single score is one of the simplest feedback formats to collect and use (Kreutzer et al., 2018; Liu et al., 2018; Shi et al., 2020). Although easy to leverage, it suffers from some limitations: depending on the complexity of the generation task, reducing feedback to a single score might generally be a challenging and vaguely defined task for humans. This leads to a costly collection process, problems of subjectivity and variance, and difficulties in distinguishing between outputs of similar quality.

Input	Output(s)	Feedback	Type
A melhor comida do mundo é a portuguesa.	*The worst food in the world are Portuguese.*	0.7	Score
		’worst’: `major/accuracy` ’are’: `minor/fluency`	MQM
		’worst’ → ’best’, ’are’ → ’is’	Post-Edition
Artificial intelligence has the potential to revolutionize industries (...) but ethical concerns need to be handled.	*AI can change industries.*	`Fluency`: 1 `Relevance`: 0.7	Multi-Aspect
		"Misses the ethical concerns."	Natural Language
Explain the moon landing to a 6 year old	A: *People went to the ...* B: *The moon is a satellite...*	A > B	Ranking

Figure 2.3: Example input and output for three tasks (machine translation, summarization, and instruction following) and possible different (example) feedback that can be given. This figure was taken from Fernandes et al. (2023).

Alternatively, **ranking-based** feedback involves humans ranking alternative outputs, providing a more nuanced understanding of model performance and often being easier to collect (Chaganty et al., 2018; Ziegler et al., 2019; Stiennon et al., 2020). **Natural language** feedback, on the other hand, offers detailed insights by capturing specific issues or suggesting improvements, contributing to a richer understanding of model shortcomings, albeit with the challenge of increased complexity in its collection. Additionally, domain-specific feedback types, such as multi-aspect feedback or post-editions, further enhance the potential for refining model behavior (Li et al., 2016; Scheurer et al., 2022; Li et al., 2022). Types of feedback are summarized in Figure 2.3 with examples.

2.4.2 Objective

Collecting human feedback is then essential to align the model's behavior with human preferences. Some works explicitly divide the problem of alignment of these models into improving its helpfulness and increasing its harmlessness (Bai et al., 2022b). **Harmlessness** is comparatively easier as it primarily involves adapting expression styles and maintaining politeness in conversations. It focuses on mitigating undesirable and harmful outputs by avoiding text toxicity, adherence to safety objectives, addressing ethical concerns, and non-violation of predefined rules. **Helpfulness**, on the other hand, is assessed through task performance feedback – such as the quality of translation or the relevance, consistency, and accuracy of summaries – with the assumption that improved task-related outputs contribute to overall usefulness.

2.4.3 Usage of Human Feedback

To leverage human feedback and capture nuanced human preferences, training approaches like feedback-imitation learning and joint-feedback modeling appear to optimize model parameters using the collected human feedback information (Fernandes et al., 2023), as shown in Figure 2.2.

Feedback-based imitation learning involves optimizing language models through supervised learning using positively labeled generations and corresponding inputs or preferred dialogues (Fernandes et al., 2023; Li et al., 2016; Kreutzer et al., 2018; Glaese et al., 2022). Its main disadvantage lies in disregarding generations without positive feedback, potentially missing valuable information from negative examples (Wang et al., 2024).

In contrast, **joint feedback modeling** incorporates all collected human feedback directly into model optimization, allowing for diverse feedback formats like natural language. Having $\mathcal{D}$ as the dataset of inputs x, generations y, and human feedback f, this can be achieved by minimizing the following loss of the form

$$\mathcal{L}_i(\theta) = -\log p_\theta(y_i, f_i \mid x_i) \tag{2.2}$$

Over all examples in $\mathcal{D}$. This loss function factors into

$$\mathcal{L}_i(\theta) = -\log p_\theta(f_i \mid y_i, x_i) + \log p_\theta(y_i \mid x_i) \tag{2.3}$$

The model can then be trained to predict feedback given to each generation (Li et al., 2016; Hancock et al., 2019), or predict both generations and corresponding human feedback (Xu et al., 2022; Thoppilan et al., 2022). Although this approach solves the challenge of learning to accurately predict nuanced human feedback, it introduces problems such as increased complexity and potential training costs associated with handling diverse feedback formats.

2.5 Reinforcement Learning from Human Feedback

Reinforcement Learning from Human Feedback (Christiano et al., 2017) emerges as a paradigmatic shift to tackle the misalignment challenges prevalent in fine-tuning objectives. RLHF represents a strategic deviation from conventional supervised learning approaches, aiming to train models based on human preferences and evaluations. For instance, in the domain of text summarization, RLHF introduces a methodical approach, leveraging human feedback to navigate the inherent subjectivity of summarization tasks, where judging summary quality is complex without human judgments. The incorporation of human feedback plays an important role in tackling challenges associated with toxicity and harmful content generation. It also integrates user preferences, thereby ensuring ethical and user-aligned AI outputs.

The application of RLHF extends far beyond summarization, making an impact across various NLP

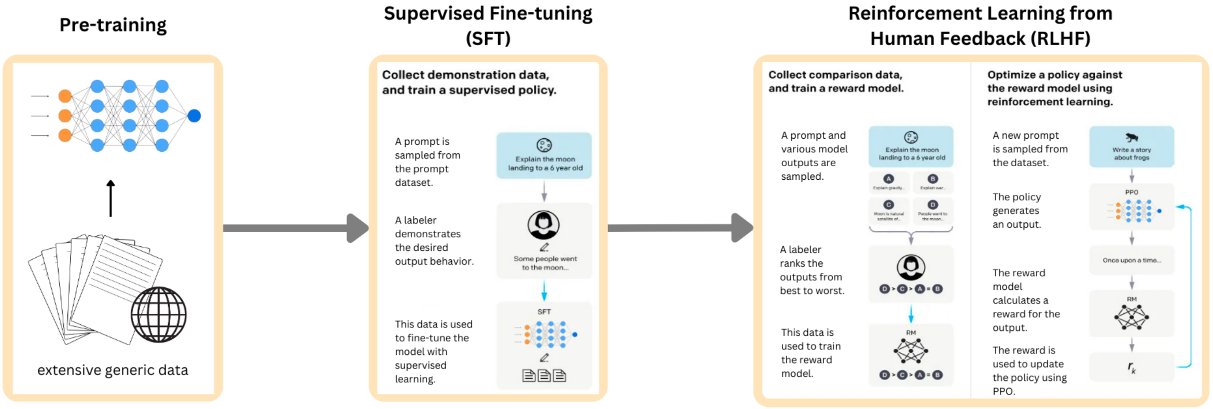
Pre-training
extensive generic data
Supervised Fine-tuning (SFT)
Collect demonstration data, and train a supervised policy.
A prompt is sampled from the prompt dataset.
Explain the moon landing to a 6 year old
A labeler demonstrates the desired output behavior.
Some people went to the moon...
This data is used to fine-tune the model with supervised learning.
SFT
Reinforcement Learning from Human Feedback (RLHF)
Collect comparison data, and train a reward model.
A prompt and various model outputs are sampled.
Explain the moon landing to a 6 year old
A
Explain gravity...
B
Explain war...
C
Moon is natural satellite of...
D
People went to the moon...
A labeler ranks the outputs from best to worst.
D > C > A = B
This data is used to train the reward model.
RM
D > C > A = B
Optimize a policy against the reward model using reinforcement learning.
A new prompt is sampled from the dataset.
Write a story about frogs
The policy generates an output.
PPO
Once upon a time...
The reward model calculates a reward for the output.
RM
The reward is used to update the policy using PPO.
r_k

Figure 2.4: The lifecycle of Large Language Models (LLMs) with distinct stages, including pre-training, supervised fine-tuning (SFT), and Reinforcement Learning from Human Feedback (RLHF).

domains. RLHF addresses a spectrum of tasks, including machine translation, dialog generation, image captioning, and sentiment generation. Notably, the scalability of language models in RLHF has increased, now reaching into the realm of hundreds of billions of parameters (Ye et al., 2023; Brown et al., 2020), reflecting a commitment to performance improvement and adaptability across the various NLP challenges.

The idea of RLHF is to align the language models with human preferences and social values by optimizing a reward function that reflects specific human preferences (e.g. moral, helpful, harmless). The process widely employed in recent studies involves a series of well-defined steps. The first stage consists of doing supervised fine-tuning (SFT) on the LLM: Labelers furnish the desired behavior's response with t tokens, denoted as $y = y_1, \ldots, y_t$, for a given input prompt, denoted as x. Subsequently, RLHF proceeds to fine-tune a pre-trained LLM using supervised learning (maximum likelihood) on this data, resulting in a model denoted as π_{SFT}:

$$\mathcal{L}_{\text{SFT}} = -\sum_t \log P_{\pi_{\text{SFT}}}(y_t \mid x, y_{1,\ldots,t-1}). \tag{2.4}$$

In the second phase, pairs of responses (y_1, y_2) are generated either by prompting the SFT model with prompts x to independently sample responses $(y_1, y_2) \sim \pi_{\text{SFT}}(y \mid x)$, or by using a different model to obtain these responses $(y_1, y_2) \sim \pi_{\text{alt}}(y \mid x)$. These pairs are then presented to human labelers, who express their preferences by indicating a favored answer as y_w, while the other response is denoted as y_l. Specifically, we use the notation $y_w \succ y_l \mid x$ where y_w and y_l denote the preferred and dispreferred answers amongst (y_1, y_2), respectively. The preferences are assumed to be generated by some latent reward model $r^*(x, y)$, which we do not have access to. To model human preferences, the Bradley-Terry (BT) model (Bradley and Terry, 1952) is one popular approach. This model seeks to assign higher scores to preferable responses in comparison to unfavorable ones when presented with the same prompt. The BT model defines the human preference distribution as follows:

$$p^*(y_1 \succ y_2 \mid x) = \frac{\exp(r^*(x, y_1))}{\exp(r^*(x, y_1)) + \exp(r^*(x, y_2))} \tag{2.5}$$

Subsequently, given a dataset providing comparative data $\mathcal{D} = \{x^{(i)}, y_w^{(i)}, y_l^{(i)}\}_{i=1}^N$, we can then parametrize a reward model $r_\theta(x, y)$ and estimate the parameters via maximum likelihood, minimizing its associated loss. Framing this objective as a binary classification to train the reward model, we get the following negative log-likelihood loss:

$$\mathcal{L}(r_\theta, \mathcal{D}) = -\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}\Big[\log \sigma(r_\theta(x, y_w) - r_\theta(x, y_l))\Big], \tag{2.6}$$

where σ is the logistic sigmoid function, $r_\theta(x, y)$ denotes the scalar output of the reward model for prompt x and response y with parameters θ, and $\mathcal{D}$ represents the dataset of human judgments sampled from p^*.

In the context of LMs, the network denoted as $r_\theta(x, y)$ is commonly initialized using the SFT model $\pi_{\text{SFT}}(y \mid x)$. This initialization involves the addition of a linear layer on top of the final transformer layer, generating a single scalar prediction for the reward value (Ziegler et al., 2019). To ensure a reward function with lower variance, prior works ensure that the rewards are normalized, hence $\mathbb{E}_{(x,y)\sim\mathcal{D}}\big[r_\theta(x, y)\big] = 0$ for all x.

In the subsequent step, RLHF utilizes the acquired reward function r_θ to provide feedback to the language model. The goal is to train a policy that generates higher-quality outputs as judged by humans (see Figure 2.5). Specifically, RLHF formulates the following optimization problem:

$$\max_{\pi_\theta} \mathbb{E}_{x\sim D, y\sim\pi_\theta(y \mid x)}\Big[r_\theta(x, y) - \beta D_{KL}(\pi_\theta(y \mid x) \parallel \pi_{\text{ref}}(y \mid x))\Big] \tag{2.7}$$

Here, β controls the deviation from the base reference policy π_{ref}, namely the initial STF model π_{SFT}. This is done to maintain generation diversity and prevent from generating of only high-reward but meaningless answers. Moreover, in practice, the language model policy π_θ is also initialized to π_{SFT}.

Due to the discrete nature of language generations, this objective is non-differentiable and is typically optimized using reinforcement learning methods. Commonly utilized approaches are PPO (Schulman et al., 2017), REINFORCE (Williams, 1992), as well as other variants such as Natural Language Policy Optimization (NLPO) (Ramamurthy et al., 2023), amongst others. (See Figure 2.4)

Importantly, previous works (Ziegler et al., 2019; Stiennon et al., 2020; Bai et al., 2022b) include in the reward function a term that penalizes the Kullback-Leibler (KL) divergence between the learned RL

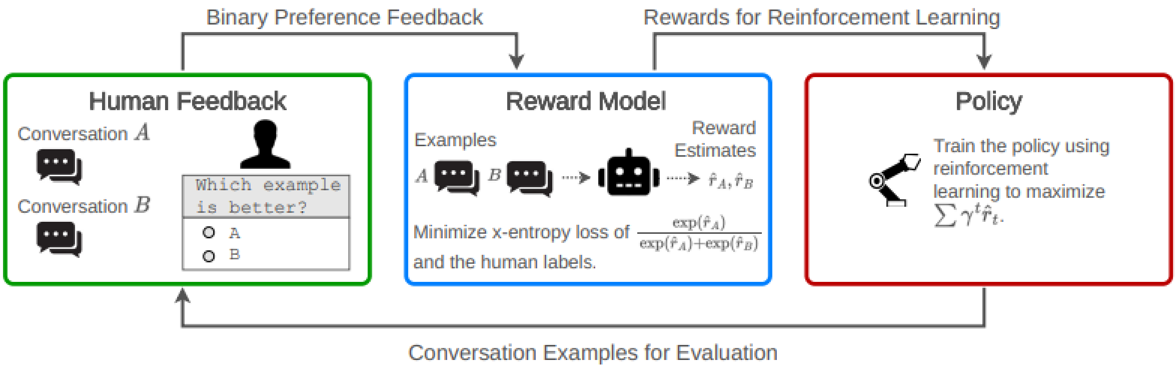
Binary Preference Feedback
Rewards for Reinforcement Learning
Human Feedback
Conversation A
Conversation B
Which example is better?
○ A
○ B
Reward Model
Examples
Reward Estimates
A
B
$\hat{r}_A, \hat{r}_B$
Minimize x-entropy loss of $\frac{\exp(\hat{r}_A)}{\exp(\hat{r}_A)+\exp(\hat{r}_B)}$ and the human labels.
Policy
Train the policy using reinforcement learning to maximize $\sum \gamma^t \hat{r}_t$.
Conversation Examples for Evaluation

Figure 2.5: An example of RLHF for finetuning chatbots with binary preference feedback, taken from Casper et al. (2023). Humans indicate which example between a pair they prefer. A reward model is trained using each example pair to provide rewards that reflect the human's preferences. Finally, the LLM policy is trained using the reward model and RL algorithms.

policy π_θ and the reference model π_{ref}. The complete reward function r can then be expressed as:

$$r(x, y) = r_\theta(x, y) - \beta \log \left[\frac{\pi_\theta(y \mid x)}{\pi_{\text{ref}}(y \mid x)} \right], \tag{2.8}$$

where β serves to weigh the KL divergence term. This KL term serves a dual purpose: acting as an entropy regularizer and preventing the policy from producing outputs too dissimilar from those encountered by the reward model during training.

The standard approach has been to construct the reward function presented in Equation 2.8 and maximize it using Proximal Policy Optimization (PPO) (Schulman et al., 2017).

In conclusion, RLHF approaches offer two notable advantages compared to traditional Supervised Fine-tuning (SFT). Firstly, they leverage both positively and negatively labeled responses, leading to a more nuanced and informative learning process. Secondly, they can engage in self-bootstrapping to iteratively refine the model's responses, contributing to continuous improvement and adaptability.

2.5.1 Limitations and Challenges of RLHF

Despite the numerous advantages, RLHF also presents several limitations. One notable concern is the heightened complexity in comparison to supervised learning approaches. This augmented complexity raises challenges that extend beyond conventional training paradigms. For instance, PPO learns in a trial-and-error fashion by interacting with the environment. In contrast to supervised learning, PPO training is difficult to implement, generally less stable, and computationally more demanding, resulting in a significantly slower training process. Furthermore, PPO exhibits heightened sensitivity to hyperparameters, demanding meticulous and precise tuning to achieve optimal outcomes.

Another noteworthy drawback of RLHF centers on the need for additional training dedicated to refining reward models and value networks, adding an extra layer of complexity to the RLHF process. The predominant framework (Ouyang et al., 2022) requires loading multiple LLMs for the PPO training, including the model being trained, the reference model, the reward model, and the critic model. This imposes a heavy burden on the memory resource, potentially limiting the scalability and practical applicability of RLHF in real-world scenarios.

2.6 Alternative Approaches to RLHF

In addressing these limitations, recent studies suggest novel and varied offline methodologies. Recently, offline methods such as **DPO** (Rafailov et al., 2023), **SimPO** (Meng et al., 2024), **CPO** (Xu et al., 2024b), **PRO** (Song et al., 2023), and **SLiC** (Liu et al., 2023) have emerged as attractive alternatives to RLHF, offering improvements in stability and scalability while maintaining competitive performance.

2.6.1 Direct Preference Optimization (DPO)

Motivated by challenges in applying reinforcement learning algorithms to large-scale tasks like language model fine-tuning, DPO (Rafailov et al., 2023) is proposed as a straightforward method for policy optimization using preferences *directly*. This method introduces a new parameterization of the reward model in RLHF, allowing for the extraction of the optimal policy in closed form without an RL training loop. The approach leverages an analytical mapping from reward functions to optimal policies, transforming a loss function over rewards into a loss function over policies. By doing so, it avoids fitting an explicit reward model while still optimizing under existing models of human preferences, such as the Bradley-Terry model. Essentially, the policy network represents both the language model and the implicit reward.

The optimal solution to the KL-constrained reward maximization objective in Equation 2.7 is formulated differently, making it possible to express the reward function r in terms of its corresponding optimal policy π_r, the reference policy π_{ref}, and an unknown partition function $Z(x) = \sum_y \pi_{\text{ref}}(y \mid x) \exp\left(\frac{1}{\beta} r(x, y)\right)$:

$$r(x, y) = \beta \log \frac{\pi_r(y \mid x)}{\pi_{\text{ref}}(y \mid x)} + \beta \log Z(x). \tag{2.9}$$

We can apply this reparameterization to the ground-truth reward r^* and corresponding optimal model π^*. Fortunately, this approach recognizes that the Bradley-Terry model represented in Equation 2.5 relies solely on the difference of rewards between two responses, i.e. $p^*(y_1 \succ y_2 \mid x) = \sigma(r^*(x, y_1) - r^*(x, y_2))$. Therefore, by substituting the reparameterization in Equation 2.9 for $r^*(x, y)$ into the preference model Equation 2.5, the partition function cancels and the human preference probability can be expressed solely in terms of the optimal policy π_* and the reference policy π_{ref}. Hence, the optimal RLHF policy π^*

under the Bradley-Terry model satisfies the preference model:

$$p^*(y_1 \succ y_2 \mid x) = \frac{1}{1 + \exp\left(\beta \log \frac{\pi^*(y_2|x)}{\pi_{\text{ref}}(y_2|x)} - \beta \log \frac{\pi^*(y_1|x)}{\pi_{\text{ref}}(y_1|x)}\right)} \tag{2.10}$$

Subsequently, with the probability of human preference data expressed in terms of the optimal policy instead of the reward model, it is possible to formulate a maximum likelihood objective for a parametrized policy π_θ. Analogous to the reward modeling approach in Equation 2.6, the policy objective becomes:

$$\mathcal{L}_{\text{DPO}}(\pi_\theta; \pi_{\text{ref}}) = -\mathbb{E}_{(x,y_w,y_l)\sim D}\left[\log \sigma\left(\beta \log \frac{\pi_\theta(y_w \mid x)}{\pi_{\text{ref}}(y_w \mid x)} - \beta \log \frac{\pi_\theta(y_l \mid x)}{\pi_{\text{ref}}(y_l \mid x)}\right)\right]. \tag{2.11}$$

By using this alternative parametrization, an implicit reward can be fitted in such a way that the optimal policy simply becomes π_θ.

2.6.2 Simple Preference Optimization (SimPO)

SimPO (Meng et al., 2024) refines DPO's approach by simplifying yet showing more effectiveness. Using Equation 2.9 as the implicit reward expression have two main drawbacks: (1) the requirement of a reference model π_{ref} during training incurs additional memory and computational costs, and (2) there is a discrepancy between the reward being optimized during training and the generation metric used for inference. Specifically, during generation, the policy model π_θ is used to generate a sequence that approximately maximizes the average log likelihood, defined as follows:

$$p_\theta(y \mid x) = \frac{1}{|y|} \log \pi_\theta(y \mid x) = \frac{1}{|y|} \sum_{i=1}^{|y|} \log \pi_\theta(y_i \mid x, y_{<i}). \tag{2.12}$$

In DPO, the fact that a triple (x, y_w, y_l) meets the reward ranking $r(x, y_w) > r(x, y_l)$ does not ensure that the likelihood ranking $p_\theta(y_w|x) > p_\theta(y_l|x)$ is also satisfied. In practice, Meng et al. (2024) show that only about 50% of the triples in the training set meet this criterion when trained with DPO.

Taking this into account, SimPO replaces the reward formulation in DPO with p_θ in 2.12 so that it aligns with the likelihood metric that guides generation. This results in a length-normalized reward:

$$r_{\text{SimPO}}(x, y) = \frac{\beta}{|y|} \log \pi_\theta(y \mid x) = \frac{\beta}{|y|} \sum_{i=1}^{|y|} \log \pi_\theta(y_i \mid x, y_{<i}), \tag{2.13}$$

where β is a constant that controls the scaling of the reward difference. Results on this work show that it is essential to normalize the reward by the length of responses. Omitting this normalization leads to a tendency to generate longer yet poorer-quality sequences. This approach to reward formulation also eliminates the need for a reference model, thereby boosting both memory and computational efficiency

Preference Algorithm	Loss Function
DPO	$\mathcal{L}_{\text{DPO}}(\pi_\theta;\pi_{\text{ref}}) = -\mathbb{E}_{(x,y_w,y_l)\sim D}\left[\log\sigma\left(\beta\log\frac{\pi_\theta(y_w\mid x)}{\pi_{\text{ref}}(y_w\mid x)} - \beta\log\frac{\pi_\theta(y_l\mid x)}{\pi_{\text{ref}}(y_l\mid x)}\right)\right]$
SimPO	$\mathcal{L}_{\text{SimPO}}(\pi_\theta) = -\mathbb{E}_{(x,y_w,y_l)\sim D}\left[\log\sigma\left(\frac{\beta}{\lvert y_w\rvert}\log\pi_\theta(y_w\mid x) - \frac{\beta}{\lvert y_l\rvert}\log\pi_\theta(y_l\mid x) - \gamma\right)\right]$
CPO	$\mathcal{L}(\pi_\theta;U) = -\mathbb{E}_{(x,y_w,y_l)\sim D}\Big[\log\sigma\Big(\beta\log\pi_\theta(y_w\mid x) - \beta\log\pi_\theta(y_l\mid x)\Big)\Big]$ $\min_\theta L(\pi_\theta, U) - \mathbb{E}_{(x,y_w)\sim D}\left[\log\pi_\theta(y_w\mid x)\right]$
SLiC	$\mathcal{L}(\pi_\theta) = \max\Big(0, \delta - \log\pi_\theta(y_w\mid x) + \log\pi_\theta(y_l\mid x)\Big) - \lambda\log\pi_\theta(y_{\text{ref}}\mid x)$
RSO	$\mathcal{L}_{\text{RSO}}(\pi_\theta;\pi_{\text{ref}}) = \mathbb{E}_{(x,y_u,y_l)\sim D}\left[\max\left(0, 1 - \left[\gamma\log\frac{\pi_\theta(y_w\mid x)}{\pi_{\text{ref}}(y_w\mid x)} - \gamma\log\frac{\pi_\theta(y_l\mid x)}{\pi_{\text{ref}}(y_l\mid x)}\right]\right)\right]$
PRO	$\mathcal{L}_{\text{PRO}} = -\sum_{k=1}^{n-1}\log\frac{\exp(r_{\pi_{\text{PRO}}}(x,y^k))}{\sum_{i=k}^{n}\exp(r_{\pi_{\text{PRO}}}(x,y^i))}$ $r_{\pi_{\text{PRO}}}(x,y^k) = \frac{1}{\lvert y^k\rvert}\sum_{t=1}^{\lvert y^k\rvert}\log P_{\pi_{\text{PRO}}}(y_t^k \mid x, y_{<t}^k)$

Table 2.1: Comparison of Loss Functions for the different preference algorithms.

compared to reference-dependent algorithms.

Additionally, SimPO introduces a target reward margin term, $\gamma > 0$, to the Bradley-Terry objective to ensure that the reward for the winning response, $r(x, y_w)$, exceeds the reward for the losing response, $r(x, y_l)$, by at least γ:

$$p(y_w > y_l \mid x) = \sigma\big(r(x, y_w) - r(x, y_l) - \gamma\big). \tag{2.14}$$

Finally, by plugging 2.12 into 2.14, SimPO objective becomes:

$$\mathcal{L}_{\text{SimPO}}(\pi_\theta) = -\mathbb{E}_{(x,y_w,y_l)\sim D}\left[\log\sigma\left(\frac{\beta}{|y_w|}\log\pi_\theta(y_w \mid x) - \frac{\beta}{|y_l|}\log\pi_\theta(y_l \mid x) - \gamma\right)\right] \tag{2.15}$$

A comparison between DPO's and SimPO's objective can be found on 2.1.

In summary, SimPO employs an implicit reward formulation that directly aligns with the generation metric, eliminating the need for a reference model. Additionally, it introduces a target reward margin γ to help separating the winning and losing responses.

2.6.3 Contrastive Preference Optimization (CPO)

CPO (Xu et al., 2024b) addresses the inherent limitations of traditional SFT by focusing on optimizing model outputs against a contrastive preference framework instead of merely minimizing differences from a gold-standard reference, which herently caps model performance at the quality level of the training data. Notably, even data considered high-quality, such as human-written texts, is not immune to quality issues. Furthermore, SFT lacks mechanisms to prevent errors in the model's outputs, as previously noted. While strong models can produce high-quality outputs, they occasionally exhibit minor errors, such as omitting small details.

The preference data for CPO is constructed by sampling outputs using 2 distincts well performing models, while also incorporating the original target reference. This process results in the creation of response triplets, each comprising three different outputs corresponding to a single input. These triplets are then scored by reference-free evaluation models, which classify the highest and lowest-scoring outputs to identify the preferred and dis-preferred responses. The designation 'dis-preferred' indicates that there is still room for improvement, perhaps through the addition of minor details. This approach of using high-quality but not flawless outputs as dis-preferred data contributes to training the model to refine its details and strive for excellence in the generated outputs.

When analysing DPO, Xu et al. (2024b) acknowledge that DPO has some noticeable inefficiencies: memory-inefficiency - it necessitates twice the memory capacity to simultaneously store both the parameterized policy and the reference policy; and speed-inefficiency - executing the model sequentially for two policies doubles the processing time.

This DPO's memory- or speed- inefficiency can be resolved by setting π_{ref} as a uniform prior U. This negates the need for additional computations and storage beyond the policy model itself. When setting π_{ref} as a uniform prior U, we obtain

$$\begin{aligned}\mathcal{L}(\pi_\theta; \pi_{\text{ref}} = U) &= -\mathbb{E}_{(x,y_w,y_l)\sim D}\left[\log \sigma\left(\beta \log \frac{\pi_\theta(y_w \mid x)}{c} - \beta \log \frac{\pi_\theta(y_l \mid x)}{c}\right)\right] \\ &= -\mathbb{E}_{(x,y_w,y_l)\sim D}\Big[\log \sigma\Big(\beta \log \pi_\theta(y_w \mid x) - \beta \log \pi_\theta(y_l \mid x)\Big)\Big]\end{aligned} \tag{2.16}$$

where the terms $\pi_{\text{ref}}(y_w \mid x)$ and $\pi_{\text{ref}}(y_l \mid x)$ cancel each other out.

Contrary to common practices where π_{ref} is set as the initial SFT checkpoint, the CPO's approach considers it as the ideal policy we aim to reach. Although the ideal policy π_w is unknown and unattainable during model training, it is not required in the loss function. Furthermore, it also incorporates a behavior cloning (BC) regularizer (Hejna et al., 2024) to ensure that π_θ does not deviate from the preferred data distribution:

$$\min_\theta \mathcal{L}(\pi_\theta, U) \quad \text{s.t.} \quad \mathbb{E}_{(x,y_w)\sim D}\Big[D_{KL}(\pi_w(y_w \mid x) \parallel \pi_\theta(y_w \mid x))\Big] < \epsilon, \tag{2.17}$$

where ϵ is a small positive constant and D_{KL} denotes Kullback–Leibler (KL) divergence. The regularizer can boil down to adding a SFT term on the preferred data, as demonstrated in (Xu et al., 2024b):

$$\min_\theta \underbrace{L(\pi_\theta, U)}_{L_{\text{prefer}}} \underbrace{- \mathbb{E}_{(x,y_w)\sim D}\Big[\log \pi_\theta(y_w \mid x)\Big]}_{L_{\text{NLL}}}. \tag{2.18}$$

The above is the formulation of CPO's loss, which includes one preference learning term L_{prefer} and one negative log likelihood term L_{NLL}.

By incorporating CPO, particularly in machine translation tasks, LLMs are trained to pay close attention to the nuances of translation quality, enabling them to distinguish subtle differences in quality among various outputs. As a result, LLMs develop a more nuanced understanding of language intricacies, leading to more accurate and contextually appropriate translations.

2.6.4 Sequence Likelihood Calibration (SLiC)

Similarly, SLiC (Zhao et al., 2023) effectively leverages feedback data from another model (off-policy), making it unnecessary to collect costly new feedback data. For sampled and labeled preference pairs, it combines a rank calibration loss and a cross-entropy regularization loss, taking advantage of their simplicity and natural fit to pairwise human feedback data. The loss function is then defined as:

$$\mathcal{L}(\theta) = \max\left(0, \delta - \log \pi_\theta(y_w \mid x) + \log \pi_\theta(y_l \mid x)\right) - \lambda \log \pi_\theta(y_{\text{ref}} \mid x) \tag{2.19}$$

where δ is a positive margin of the ranking loss, and π_θ is the learnable conditional probability function by the language model. The second term is the cross-entropy loss, with y_{ref} representing some target sequence and λ the regularization weight. SLiC intuitively applies a penalty to the model when the ratio $\frac{\pi_\theta(y_w \mid x)}{\pi_\theta(y_l \mid x)} < \exp(\delta)$, encouraging a substantial likelihood ratio between positive and negative outputs. Instead of directly fitting on human preference data, SLiC proposed refining its loss function using sequence pairs sampled from the SFT policy and labeling them using a pairwise reward-ranking model.

Although SLiC approaches appear as scalable alternatives to PPO, they lack theoretical understanding. On the other hand, DPO utilizes human preference data from other policies and lacks explicit study on the effects of sampling. Sampling human preference pairs directly from π^* is highly challenging in reality, given the mismatch between the sampling distribution and π^*. Furthermore, both these approaches might fail to capture global differences corresponding to human preference in the long rankings.

2.6.5 Preference Ranking Optimization (PRO)

PRO (Song et al., 2023) was proposed as an alternative to PPO, extending Bradley-Terry's pairwise comparison to include comparisons within preference rankings of arbitrary lengths. Equation 2.5 presents the Bradley-Terry model, designed to convey the understanding that $y_1 \succ y_2$ through score comparison. When given a prompt x and only two responses, y_1 and y_2, in the LLM response space, the reward model should prefer y_1 over y_2. If the LLM response space is extended to include n possible responses $\{y_i\}$, and the human-annotated order $y_1, \ldots, y_n$ is $y_1 \succ y_2 \succ \ldots \succ y_n$, the partial order between y_1 and

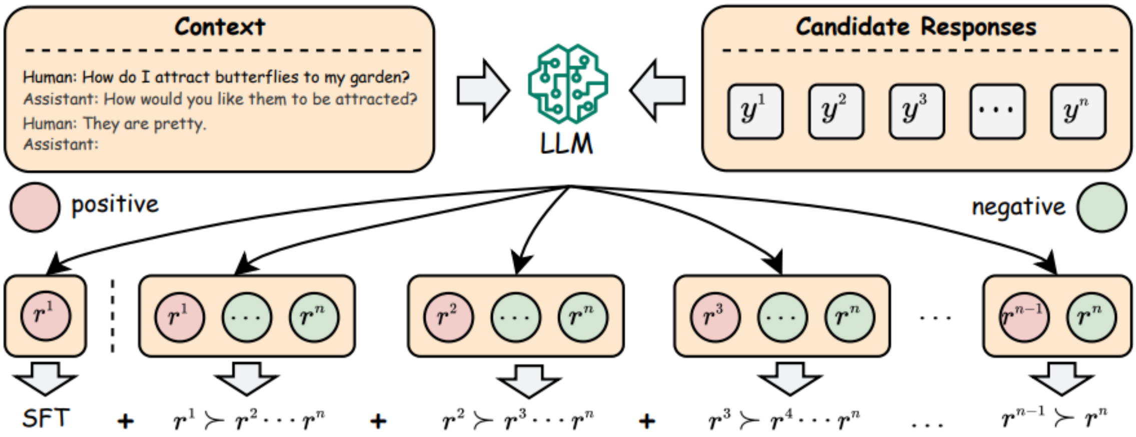

Figure 2.6: The pipeline of PRO for Human Feedback Alignment learning as depicted in (Song et al., 2023) which are optimized by Equation 2.22.

all candidates behind it is defined as $y_{1,2:n} = y_1 \succ \{y_2, \ldots, y_n\}$. The objective of Bradley-Terry can then be expressed as:

$$P(y_{1,2:n} \mid x) = \frac{\exp(r(x, y_1))}{\sum_{i=1}^{n} \exp(r(x, y_i))} \tag{2.20}$$

To fully leverage the preference rankings $y_1, \ldots, y_n$ and not disregard the $n-2$ valuable rankings such as $y_2 \succ y_3, \ldots, y_n$ and $y_{n-1} \succ y_n$, they propose an extension to Equation 2.20 as follows:

$$P(y_{1,\ldots,n} \mid x) = \prod_{k=1}^{n-1} P(y_{k,k+1:n} \mid x) = \prod_{k=1}^{n-1} \frac{\exp(r(x, y_k))}{\sum_{i=k}^{n} \exp(r(x, y_i))} \tag{2.21}$$

This extension enforces the desired ranking indicated by $y_1 \succ y_2 \succ \ldots \succ y_n$, employing Equation 2.20 iteratively. It starts with the first response, treating the remaining responses as negatives. Then, it removes the current response and moves to the next, repeating this process until no responses are left. Remarkably, this objective closely aligns with the overarching goal of Human alignment, which is the task of selecting desired responses from the extensive response space of LLMs. Essentially, as $n \to \infty$, this extension can exhaustively explore all possible responses, annotating y^1 as the most desired response, thus achieving perfect alignment with humans. Figure 2.6 demonstrates the pipeline of the PRO algorithm.

The reward function is then defined using the desired π_{PRO}. The LLM π_{PRO} calculates the score for response y_k by multiplying the probabilities of tokens generated by π_{PRO} itself. Optimizing Equation 2.21 enables π_{PRO} to consistently generate the most preferred response with a higher output probability from a candidate set, aligning with human preferences. The overall optimization objective can be defined as follows:

$$\mathcal{L}(y_{1,\ldots,n} \mid x) = \mathcal{L}_{\text{PRO}} + \beta \mathcal{L}_{\text{SFT}} \tag{2.22}$$

where $\mathcal{L}_{\text{SFT}}$ is the negative log-likelihood loss of the top 1 candidate, and β is the hyper-parameter that balances text quality and human preference. L_{PRO} is defined as:

$$\mathcal{L}_{\text{PRO}} = -\sum_{k=1}^{n-1} \log \frac{\exp(r_{\pi_{\text{PRO}}}(x, y^k))}{\sum_{i=k}^{n} \exp(r_{\pi_{\text{PRO}}}(x, y^i))} \tag{2.23}$$

where the policy agent (also RM) is parameterized as $r_{\pi_{\text{PRO}}}$:

$$r_{\pi_{\text{PRO}}}(x, y^k) = \frac{1}{|y^k|} \sum_{t=1}^{|y^k|} \log P_{\pi_{\text{PRO}}}(y_t^k \mid x, y_{<t}^k) \tag{2.24}$$

PRO and RLHF share the goal of human alignment, but PRO achieves this through differentiable ranking scores, avoiding RL-associated drawbacks. Additionally, PRO aims for high-quality model outputs with a differentiable alignment objective, allowing efficient single-stage training and multi-task learning.

2.7 Small Language Models (SLMs)

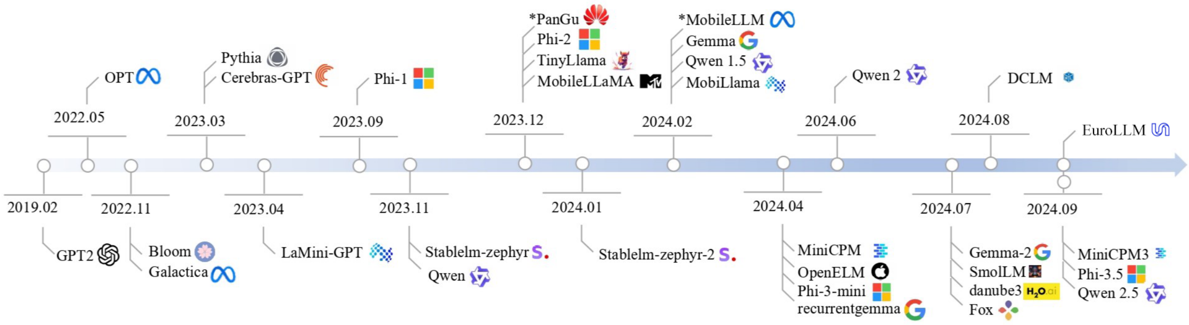

Figure 2.7: An overview of SLMs. * indicates the models are not open-sourced. Figure based on Lu et al. (2024).

As depicted in Figure 2.7, small language models (SLMs) have attracted considerable interest from both the academic and industrial sectors. Particularly since late 2023, there has been a marked increase in the development and deployment of SLMs. This surge is attributed to the growing recognition of their capabilities and the advantages they offer, such as lower computational demands and greater efficiency, making them highly suitable for a wide range of applications.

Recent advancements in small language models have showcased their potential to rival larger models in various computational tasks. Zhang et al. (2024a) developed a compact 1.1B model, TinyLlama, that demonstrates remarkable performance in a series of downstream tasks. It significantly outperforms existing open-source language models with comparable sizes. Microsoft (2024) released Phi-2, a 2.7 billion-parameter language model that demonstrates outstanding reasoning and language understanding capabilities, showcasing state-of-the-art performance among base language models with less than

13 billion parameters. Phi-2 matches or outperforms models up to 25x larger, thanks to new innovations in model scaling and training data curation. Team et al. (2024) introduces Gemma-2 2B, a update to the Gemma family of lightweight, state-of-the-art, an innovative approach that incorporates technical modifications to the Transformer architecture and a different training methodology. The resulting model delivers the best performance for their size, and even offer competitive alternatives to models that are 2-3× bigger. While the development of open-weight language models has predominantly focused on English, Martins et al. (2024) presents EuroLLM-1.7B, an initial model of the *EuroLLM* project aimed at developing a suite of open-weight multilingual LLMs capable of understanding and generating text in all official European Union languages along with other significant languages. In machine translation, EuroLLM-1.7B outperforms Gemma-2B, and is competitive with larger models like Gemma-7B despite the much lower number of parameters.

This trend reflects a shift in the focus of research and development efforts towards creating models that are not only powerful but also more sustainable and accessible for everyday use. Lu et al. (2024) provides a comprehensive survey of Small Language Models (SLMs), exploring their capabilities in various domains, including commonsense reasoning, in-context learning, mathematics, and coding as well as on-device runtime costs.

While there has been significant progress in the development of SLMs, the application of preference alignment methods on these models remains underexplored. Despite the promising results of RL-free approaches in larger models, it is crucial to investigate whether these strategies can be effectively replicated in SLMs to enhance performance and ensure alignment with human preferences.

2.8 Tasks Definitions & Evaluation

2.8.1 Machine Translation (MT)

Definition Machine translation (MT) refers to the automated process of translating text from one language to another. A key aspect of MT involves creating models that not only perform accurate translations but also preserve the original text's meaning and grammatical structure. Essentially, an MT system aims to produce the most precise translation for a given source sentence in the desired target language. This entails encoding the source sentence's meaning to ensure complete information retention (Weaver, 1952). The majority of MT research is dedicated to the generation and assessment of translations of a source texts.

Datasets Datasets for machine translation typically include a pair of input-output examples that are translations of each other, i.e., parallel text, and are curated by crawling the web or aligning texts available in multiple languages. The availability of resources varies significantly across languages, and because the internet is predominantly English-centric, the majority of accessible parallel texts involve English. State-of-the-art datasets for machine translation include the annual releases from the Workshop on Machine Translation: WMT21 (Akhbardeh et al., 2021), WMT22 (Kocmi et al., 2022), and WMT23 (Kocmi et al., 2023); FLORES-101 (Goyal et al., 2021) and FLORES-200 (Team et al., 2022) datasets are notable for benchmarking translation quality across a wide range of languages; TICO-19 (Anastasopoulos et al., 2020); TowerBlocks (Alves et al., 2024b); and MT-Pref (Agrawal et al., 2024).

Evaluation Evaluating MT systems predominantly relies on automatic metrics that compare generated translations to reference texts. Traditional metrics like BLEU (Papineni et al., 2002), METEOR (Banerjee and Lavie, 2005), and chrF (Popović, 2015) assess n-gram overlap between the machine translation and a reference translation. More advanced metrics, such as BLEURT (Sellam et al., 2020), COMET (Rei et al., 2020) and XCOMET (Guerreiro et al., 2023), utilize contextual embeddings from pretrained language models to evaluate semantic similarity.

A significant challenge in MT evaluation is the reliance on single-reference translations, which may not capture the full range of valid translations due to variations in style, word choice, and syntax (Qin and Specia, 2015). This limitation can introduce *reference-bias* and lead to inaccurate assessments of a system's performance (Freitag et al., 2020). The creation of high-quality, multi-reference datasets is both costly and time-consuming, further complicating the evaluation process.

To address these issues, there has been a growing interest in developing reference-free evaluation metrics that estimate translation quality without depending on reference texts (Yankovskaya et al., 2019; Zhao et al., 2020).

2.8.2 Summarization

Definition Summarization involves condensing a source text into a shorter form that retains the essential information and key ideas of the original content. The goal is to generate a concise summary $\hat{Y}$ from a given input sequence X, effectively capturing the key points while maintaining coherence and readability. This involves understanding the core concepts of X and producing a sequence of tokens $\hat{Y}$ that represents the gist of the original text (Mani and Maybury, 2001).

Datasets Datasets for summarization typically consist of document-summary pairs and are curated from various sources such as news articles, scientific papers, and conversational transcripts. Notable datasets include CNN/DailyMail (Hermann et al., 2015), Webis-TLDR-17 (Völske et al., 2017), Reddit TL;DR, and OpenAI TL;DR Human Feedback (Stiennon et al., 2020).

Evaluation Evaluating summarization systems predominantly relies on automatic metrics that compare generated summaries to reference summaries. Traditional metrics like ROUGE (Lin, 2004) and

METEOR (Banerjee and Lavie, 2005) assess n-gram overlap between the machine-generated summary and the reference, focusing on recall-oriented similarity. ROUGE measures the overlap of n-grams and is widely used due to its simplicity and effectiveness. METEOR extends this by computing a score for this matching using a combination of unigram-precision, unigram-recall, and a measure of fragmentation that is designed to directly capture how well-ordered the matched words in the output are in relation to the reference.

More advanced metrics like BERTScore (Zhang et al., 2020) utilize contextual embeddings from pretrained language models to evaluate semantic similarity between summaries, capturing semantic relationships beyond exact word matching.

Recent works, such as Song et al. (2023), utilize model-based evaluations employing reward models that assign a reward to summary for a given prompt.

2.9 Challenges in Model Evaluation

2.9.1 Biases in Automated LLM Evaluation

When using Large Language Models to evaluate generated texts, various biases can be introduced into the assessment process. These biases can affect the fairness and accuracy of evaluations and have been discussed in several studies (Zheng et al., 2023; Wang et al., 2023; Bai et al., 2022a). Some examples of these biases are presented below.

Position Bias: Position bias occurs when an LLM exhibits a preference for responses based on their position in a comparison rather than their content. Ideally, if we define the true probability of option a being preferred over option b as $P(a, b)$, it should hold that $P(a, b) = 1 - P(b, a)$. Position bias is present when this symmetry does not hold, i.e., when $\hat{P}(a, b) \neq 1 - \hat{P}(b, a)$, where $\hat{P}$ denotes the estimated probability from the model.

For example, Wang et al. (2023) observed that GPT-4 tends to prefer the first option presented, while ChatGPT shows a preference for the second option. To account for this bias, they propose several strategies such as swap the positions and evaluate the options twice, Multiple Evidence Calibration and Human-in-the-Loop Calibration.

Self-Enhancement Bias: Self-enhancement bias refers to the tendency of an LLM to prefer responses generated by itself over those generated by other models. This bias poses a significant challenge when LLMs are used to benchmark other LLMs (Zheng et al., 2023). However, in the context of Reinforcement Learning from AI Feedback (RLAIF), this bias is less problematic since comparisons are typically made between responses generated by the same model.

Length Bias: Length bias occurs when LLMs favor longer, more detailed responses over concise ones, even when the additional length does not contribute to quality. Training models under verbosity bias can lead them to produce excessively long outputs, which may not be practical or desirable in applications requiring succinct responses, such as question answering.

2.9.2 Verbosity Bias

As introduced above, verbosity bias represents a significant challenge in the evaluation of LLMs, manifesting as a preference for longer, more detailed responses over concise ones, even when such length does not enhance quality. Consequently, LLMs may produce excessively verbose outputs that may not be practical or desirable in contexts where succinctness is essential.

This issue is particularly important in tasks like summarization, where both the content and the length of the summaries significantly impact their overall quality. Given the potential consequences of verbosity bias, it is crucial to examine this phenomenon in depth to ensure LLMs generate responses that are both informative and appropriately concise.

Recent studies have sought to understand and address verbosity bias. For instance, Zheng et al. (2023) conducted experiments involving a ”repetitive list attack” on various LLMs. In this approach, the model is prompted to list items, and responses are intentionally made verbose by repeating items multiple times. The model's evaluation of these augmented responses determines the success of the attack; if it rates the verbose responses as superior to the original, the attack is deemed successful. Their findings indicate that GPT-4 is significantly less susceptible to this attack, exhibiting a success rate below 10%, whereas other weaker models like GPT-3.5 and Claude-v1 experience success rates exceeding 90%.

Additionally, Huang et al. (2024a) examined verbosity bias in summarization tasks, finding that GPT-4 surprisingly prefers shorter responses for faithfulness and coverage, especially in single-answer evaluations and not in comparative grading. Saito et al. (2023) demonstrate that LLMs tend to favor longer answers for creative writing tasks, and alignment with humans varies on verbosity showing lower human alignment in cases where humans prefer shorter answers. In cases where human feedback preferred the longer answer, human alignment was high for the LLMs, meaning the LLMs preferred the longer answers as well. However, when human feedback chose the answer with fewer words, human alignment was low, as the LLMs still chose the longer answers regardless of the helpfulness of the shorter answer.

This suggests that verbosity bias can be different between different tasks and highlights the need for a nuanced understanding of how verbosity affects evaluations across different tasks.

Despite its significance, addressing verbosity bias has not been studied in smaller models, especially after preference optimization. Preference optimization are relevant to this issue. In case of DPO (Rafailov et al., 2023), its reward expression $r(x, y) = \beta \log \frac{\pi_\theta(y \mid x)}{\pi_{\text{ref}}(y \mid x)}$ (with the partition function excluded) lacks an explicit term for length normalization, even though the logarithmic ratio between the policy model and the reference model can serve to implicitly counteract length bias.

Building on the work by Park et al. (2024) and Zhang et al. (2024b), which highlights significant exploitation of length bias in the DPO setting as well as in PO models in general, further research has been conducted. Meng et al. (2024) demonstrates that SimPO consistently outperforms DPO and its latest variants without substantially increasing response length using 7 and 8B models. Importantly, this study show that SimPO does not significantly increase response length compared to the SFT or DPO models, indicating minimal length exploitation (Singhal et al., 2024).

Therefore, it warrants closer examination to ensure that LMs, and particularly smaller models, generate responses that are appropriately concise and informative after undergoing preference optimization. This topic will be comprehensively discussed in Chapter 5.

2.10 Summary of the Related Work

In recent advancements in the field of training large language models (LLMs), such as GPT-3 (Brown et al., 2020), GPT-4 (OpenAI, 2023), Google's PaLM (Chowdhery et al., 2023; Anil et al., 2023), and Meta's LLaMA (Touvron et al., 2023a), there have been significant improvements in the models' ability to perform tasks without prior training or with minimal examples (zero-shot or few-shot scenarios) (Radford et al., 2019). This progress aligns with the broader evolution from discriminative models (e.g. BERT (Devlin et al., 2018)) to generative models (e.g., GPT-3 (Brown et al., 2020)) as part of the generative foundation model approach (Bommasani et al., 2021). Generative foundation models, pre-trained on extensive data and adaptable to diverse downstream tasks, have reshaped the landscape of natural language processing (NLP) and exhibited emergent capabilities in complex reasoning tasks. Despite their success, generative foundation models grapple with implicit biases, often resulting in inaccurate or toxic outcomes.

RLHF (Ziegler et al., 2019) focuses on maximizing rewards through interaction with a reward model using reinforcement learning algorithms like PPO (Schulman et al., 2017). While RLHF has demonstrated success, the process of fine-tuning large language models using reinforcement learning remains a challenge due to its instability, reward hacking, and scalability (Zhao et al., 2023; Rafailov et al., 2023).

Addressing these challenges, recent works explore alternatives to RLHF by introducing robust RL-free approaches to optimize for human preference feedback (Rafailov et al., 2023; Meng et al., 2024; Xu et al., 2024b; Liu et al., 2023; Song et al., 2023; Yuan et al., 2023). These alternative approaches aim to sidestep the practical obstacles associated with RL-based fine-tuning, providing promising avenues for enhancing language model performance without the complexities and potential pitfalls of Reinforcement Learning.

Particularly, some methods have been proposed for optimizing relative preferences without relying on **RL!** (Rafailov et al., 2023; Meng et al., 2024; Zhao et al., 2023; Xu et al., 2024b; Yuan et al., 2023). These methods optimize the model's compatibility with preference datasets under models like the BT model, focusing on human or model-ranked data pairs. DPO (Rafailov et al., 2023) introduces a new parameterization of the reward model in RLHF that enables the extraction of the corresponding optimal policy in closed form, solving the standard RLHF problem with only a simple classification loss. Based on the Bradley-Terry (BT) model (Bradley and Terry, 1952), DPO proposes a maximum likelihood estimator (MLE) to fit on human preference data *directly*. SimPO (Meng et al., 2024) is proposed as a simpler yet more efficient approach to DPO. Its reward formulation better aligns with model generation and eliminates the need for a reference model, making it more compute and memory efficient. Additionally, it introduces a target reward margin to the BT objective to encourage a larger margin between the winning and losing responses, further enhancing the algorithm's performance. SLiC (Zhao et al., 2023) advocates for training a pairwise reward-ranking model, labeling pairs generated from supervised fine-tuning (SFT) using the reward model, and subsequently training the model using a contrastive calibration loss (Zhao et al., 2022) and a regularization fine-tuning loss. CPO (Xu et al., 2024b) approach improves upon traditional SFT by adopting a contrastive preference framework that aims to enhance model outputs, rather than just minimizing deviations from a gold-standard reference. This strategy addresses inherent flaws in high-quality data and reduces output errors, such as missing minor details, thereby ensuring richer and more accurate model performance. Finally, RRHF (Yuan et al., 2023) assumes access to a list of responses and their reward values to the same input, utilizing a zero-margin likelihood contrastive loss. Although RRHF and SLiC are shown to be a scalable alternative to PPO, they lack theoretical understanding.

To further enhance model's performance, additional works (Liu et al., 2023) propose an approach that seeks to integrate the methodologies of SLiC and DPO, considering the choice of loss perspective and incorporating theoretical foundations. The proposed method also further enhances the collection of preference data using statistical rejection sampling (Neal, 2003), a technique for generating samples from a target distribution using a proposal distribution. RLHF works (Bai et al., 2022b; Stiennon et al., 2020; Touvron et al., 2023b) usually refer to ”rejection sampling” as best-of-*N* or top-*k*-over-*N* algorithm, where they sample a batch of N completions from a language model policy and then evaluate them across a reward model, returning the best one or the best k ones. RSO demonstrates that its algorithm constitutes a specialized instance of this existing approach.

With the same objectives in mind, the PRO (Song et al., 2023) approach proposes an extended version of the reward model training objective proposed in Ziegler et al. (2019), to further finetune LLMs to align with human preference. PRO extends the pairwise Bradley-Terry (Bradley and Terry, 1952) model to accommodate arbitrary-length preference rankings. Given instruction x and a set of responses with human preference, it fully leverages the output's ranking order $y_1 \succ y_2 \succ \ldots \succ y_n$.

While backbone models utilized in the previously mentioned approaches have architectures with 7 billion parameters or more, there has been a notable work towards smaller language models (SLMs) (Lu et al., 2024) due to their ability to match or even surpass larger models, reaching comparative or higher efficiency and effectiveness across a variety of tasks. Innovations in model architecture and training strategies have allowed SLMs such as TinyLlama-1B (Zhang et al., 2024a), Phi-2 (Microsoft, 2024), Gemma 2 2B (Team et al., 2024) and EuroLLM-1.7B (Martins et al., 2024) to demonstrate advanced capabilities while being cost-effective and accessible. However, the potential impact of state-of-the-art preference optimization algorithms on these smaller models remains an area that has not been comprehensively explored yet.

Despite these advancements in backbone models and preference algorithms, a critical challenge persists in the evaluation of LLMs with automated metrics or with other LLMs due to inherent biases that can distort performance assessments of the evaluated outputs. Evaluation biases in LLMs, such as favoring certain response styles or lengths, can lead to misleading conclusions about a model's true capabilities and its alignment with human preferences. Zheng et al. (2023) show that an LLM judge may favor longer, verbose responses, even if they are not as clear, high-quality, or accurate as shorter alternatives. This issue is particularly important in tasks like summarization, where both the content and the length of the summaries significantly impact their overall quality.

To address these evaluation biases, recent research highlights the importance of developing unbiased evaluation methodologies. Wang et al. (2023); Zheng et al. (2023) introduce techniques such as swapping the positions of the options and evaluating them twice can help mitigate position bias. Additionally, these works also propose incorporating Chain-of-Though prompting encourages models to provide reasoning steps before final judgments. Another promising approach is the use of a few-shot judge (Zheng et al., 2023), which assesses whether providing example judgments can enhance consistency in the position bias benchmark.

3

Models and Datasets

Contents

In this section, we present the backbone models and datasets used during our comprehensive study of preference optimization of small and compact models. We provide a detailed description of each model, highlighting its unique features. We also introduce the datasets used for training and evaluating these models across the machine translation and summarization tasks, using the diverse approaches.

3.1 Backbone Models

To investigate whether the promising results of RL-free methods observed in larger models for the language tasks of machine translation and summarization extend to smaller LLMs, we analyse the current state-of-the-art open source small models. These compact models are particularly interesting due to their impressive performance and efficiency, making them well-suited for environments with limited computational and memory resources, while also increasing their versatility across a broad spectrum of applications.

As backbone models, we used **TinyLlama 1.1B** (Zhang et al., 2024a), **Gemma-2 2B**, (Team et al., 2024) and EuroLLM-1.7B (Martins et al., 2024). For machine translation tasks, we conducted experiments across all three models and thoroughly assess the outcomes. For summarization, we focused our analysis on TinyLlama and Gemma-2.

3.1.1 TinyLlama 1.1B

TinyLlama (Zhang et al., 2024a) is a compact language model with 1.1 billion parameters, pre-trained on approximately 4 trillion tokens across up to 3 epochs. Despite its relatively small size, TinyLlama performs remarkably in various downstream tasks, significantly outperforming other open-source language models of similar scale. Its compact design makes the model ideally suited for environments with limited resources and enhances its versatility across a wide range of applications that require limited computational and memory resources. The model checkpoints and source code are publicly accessible on GitHub[1].

Model Architecture TinyLlama builds upon the architecture and tokenizer introduced in Llama 2 (Touvron et al., 2023b). It incorporates several advancements from the open-source community, such as FlashAttention (Dao, 2023) and Lit-GPT (AI, 2023), which enhance computational efficiency. These technologies are pivotal in allowing TinyLlama to achieve its impressive performance metrics, optimizing both the processing speed and power usage while maintaining a reduced footprint suitable for constrained environments.

Pre-Training Methodology Table 3.1 presents some key details of the training setup used for TinyLlama.

The model was pre-trained on 16 A100-40G GPUs, facilitating substantial parallel processing to accelerate the training timeline. This comprehensive approach prepared TinyLlama for general language understanding and positioned it to excel in downstream tasks like machine translation and summarization.

Model training To further enhance its capabilities and study preference optimization, we train the TinyLlama-1.1B[2] backbone model specifically for machine translation and summarization tasks. This training incorporats Supervised Fine-Tuning (SFT) and various preference optimization algorithms, including DPO, CPO, SimPO, and SLiC. A comprehensive explanation of the training methodologies is detailed in Chapter 4.

[1] https://github.com/jzhang38/TinyLlama

[2] https://huggingface.co/TinyLlama/TinyLlama-1.1B-intermediate-step-1431k-3T

Setting	Description
Parameters	1.1B
Attention Variant	Grouped Query Attention
Model Size	Layers: 22, Heads: 32, Key Value heads groups: 4, Embedding Size: 2048, Intermediate Size (Swiglu): 5632
Sequence Length	2048
Activation Function	SiLU
Batch Size	2 million tokens (2048 × 1024)
Vocab size	32,000
Learning Rate	4×10^{-4}
Learning Rate Schedule	Cosine with 2000 warmup steps
Training Data	Slimpajama & Starcoderdata
Combined Dataset Size	Around 950 billion tokens
Total Tokens During Training	3 trillion (slightly more than 3 epochs/1,430,000 steps)

Table 3.1: TinyLlama Pre-Training Setup

3.1.2 Gemma-2

Gemma 2 (Team et al., 2024) is a cutting-edge, lightweight model developed by Google as part of the Gemma family, leveraging the research and technological advancements that went into creating the Gemini models. With a focus on text-to-text, decoder-only configurations, Gemma 2 stands out for its versatility in text generation tasks like question answering, summarization, and reasoning. Available in English with both pre-trained and instruction-tuned variants, Gemma 2's modest size allows deployment on limited-resource environments such as personal computers or small-scale cloud setups. Notably, Gemma 2 exhibits performance that surpasses similarly sized models and competes with larger models.

Model Architecture Gemma 2 builds on the decoder-only transformer architecture known for its efficiency and effectiveness in handling large-scale language models. This model uses the same tokenizer as its predecessor, Gemma 1, and the Gemini series—a SentencePiece tokenizer enhanced with features like split digits, preserved whitespace, and byte-level encodings. The tokenizer's vocabulary comprises 256,000 entries, facilitating nuanced understanding and generation of text.

Pre-Training Methodology In a shift from traditional next token prediction, Gemma 2 employs a process known as knowledge distillation (Hinton et al., 2015) for training. This involves learning from the conditional probabilities $P_T(x \mid x_c)$ assigned by the teacher to each token x given its context x_c. The aim is to minimize the negative log-likelihood of the student's predictions compared to those of the teacher, which refines the model's predictive capabilities by learning from a larger, already-trained model. The table below provides an overview of the parameters set for Gemma 2's architecture:

Setting	**Description**
Parameters	2.6B
Attention Variant	Grouped Query Attention
Model Size	Layers: 26, Heads: 8, Key Value heads groups: 4, Embedding Size: 2304
Sequence Length	8192
Activation Function	GeGLU
Feedforward dim	18,432
Vocab size	256,128
Attention Mechanism	Alternating Local Sliding Window & Global Attention
Hardware	2x16x16 configuration of TPUv5e, totaling 512 chips

Table 3.2: Gemma 2 Model Specifications

Gemma 2 was pre-trained using an array of A100 GPUs, allowing for efficient parallel processing and speedy model training. The training regimen included both standard and advanced techniques to optimize performance and adaptability across various tasks, enabling Gemma 2 to achieve remarkable efficiency and utility in real-world applications.

Instruction Tunning To enhance its conversational capabilities, Google introduced an instruction-tuned version of the Gemma-2 2B model, designed with a specific chat template for structured dialogue. This template is integral for managing conversational interactions, and it is seamlessly integrated using the tokenizer's built-in features. Each conversation turn is clearly demarcated by a `<start_of_turn>` delimiter, indicating the speaker's role - either 'user' for inputs provided by the user or 'model' for responses generated by the LLM. The turn concludes with an `<end_of_turn>` token, ensuring each segment of the dialogue is distinctly encapsulated. This structured approach enhances the model's responsiveness in conversational tasks.

Model training Similarly to TinyLlama, to further enhance its capabilities and study preference optimization, both the Gemma-2 2B[3] and its instruction tuned version, we train Gemma-2 2B IT[4] backbone models for machine translation and summarization tasks. Specifically, the former is utilized for summarization and the latter for machine translation, benefiting from instruction tuning which yielded improved outcomes. The training regimen incorporated Supervised Fine-Tuning (SFT) and various preference optimization algorithms such as DPO, CPO, SimPO, SLiC-CPO, and SLiC-DPO. A comprehensive explanation of the training methodologies is detailed in Chapter 4.

Evaluation and Applications Gemma-2 2B has demonstrated decent performance across various

[3] https://huggingface.co/google/gemma-2-2b
[4] https://huggingface.co/google/gemma-2-2b-it

benchmarks, particularly showcasing its capability in machine translation and summarization tasks. This backbone model can handle complex tasks, providing robust baseline solutions that leverage its advanced architectural and training setups. The model's ability to operate effectively on minimal computational resources further extends its utility, making state-of-the-art AI tools accessible to a broader range of users and developers.

3.1.3 EuroLLM

Despite the growing availability of open-weight LLMs (e.g., LLaMA, Mistral, or Gemma (Touvron et al., 2023b; Jiang et al., 2023; Team et al., 2024)), these are predominantly limited to English and a few high-resource languages, leaving out many European languages. To address this issue, the EuroLLM project (Martins et al., 2024) has the goal of creating a suite of LLMs capable of understanding and generating text in all European Union languages as well as some additional relevant languages. EuroLLM-1.7B is the first pre-trained model of the EuroLLM series. It is a 1.7B parameter model trained on 4 trillion tokens divided across the considered languages and several data sources: Web data, parallel data (`en-xx` and `xx-en`), and high-quality datasets.

Model Architecture EuroLLM uses a standard, dense Transformer architecture (Vaswani et al., 2017), incorporating advanced features to enhance performance. It leverages grouped query attention (GQA), like TinyLlama and Gemma 2 models, with 8 key-value heads, like Gemma 2, which has been shown to improve speed during inference (Team et al., 2024). Additionally, EuroLLM also utilizes pre-layer normalization, the SwiGLU activation function, and rotary positional embeddings in every layer.

Pre-Training Methodology EuroLLM-1.7B was pre-trained on 4 trillion tokens, increasing the predominance of high-quality data on the final 10% of the pre-training process. The authors used 256 Nvidia H100 GPUs of the Marenostrum 5 supercomputer, training the model with a constant batch size of 3,072 sequences, which corresponds to approximately 12 million tokens, using the Adam optimizer and bfloat16 mixed precision. Relevant model and training hyperparameters are shown in Table 3.3:

Instruction Tunning To enhance its conversational capabilities, UTTER[5] has also released an instruction-tuned version of the model[6], employing the `chatml` chat template (Casper et al., 2023) for structured dialogue. EuroLLM-1.7B was further instruction-tuned on EuroBlocks, an instruction-tuning dataset with a focus on general instruction-following and machine translation, resulting in the first instruction-tuned model of the EuroLLM series.

EuroLLM was finetuned with chat format control tokens. Each conversation turn is marked by a `<|im_start|>` delimiter to indicate the speaker's role — 'user' for user inputs and 'assistant' for model-

[5] https://he-utter.eu/

[6] https://huggingface.co/utter-project/EuroLLM-1.7B-Instruct

Setting	Description
Parameters	1.7B
Attention Variant	Grouped Query Attention
Model Size	Layers: 24, Heads: 16, Key Value heads groups: 8, Embedding Size: 2048
Sequence Length	4,096
Layer Norm	RMSNorm
Activation Function	GeGLU
FFN Hidden size	5,632
Vocab size	128,000
Max Learning Rate	3×10^{-4}
Min Learning Rate	3×10^{-5}
Learning Rate Schedule	Trapezoid with 10% warmup steps

Table 3.3: EuroLLM Model Specifications

generated responses. Turns conclude with a `<|im_end|>` token to clearly separate dialogue segments. See Appendix A.1.1 for additional details and examples about the template. This structured approach enhances the model's responsiveness in conversational tasks.

Model training Similarly to TinyLlama and Gemma 2, to further enhance its capabilities and study preference optimization, the backbone model was specifically trained for the task of machine translation. We use an off-the-shelf instruction-tuned model (i.e. EuroLLM-1.7B-Instruct[7]) because the model has undergone extensive instruction-tuning processes, making it more powerful and robust than base setup. We also experiment with SFT on top of the instruction-tuned model before PO.

Evaluation and Limitations EuroLLM-1.7B-Instruct was evaluated on several machine translation benchmarks: FLORES-200, WMT-23, and WMT-24. These evaluations involved comparisons with Gemma-2B and Gemma-7B (Team et al., 2024), both of which were also instruction-tuned on EuroBlocks. The results demonstrate that EuroLLM-1.7B outperforms Gemma-2B in machine translation and exhibits competitive performance against Gemma-7B, despite the latter being a significantly larger model.

Although EuroLLM-1.7B-Instruct shows promising results, it has not been aligned with human preferences, indicating potential space for further enhancement. This presents a valuable opportunity for this work to explore and improve the alignment of this compact yet powerful model with human preferences.

[7] https://huggingface.co/utter-project/EuroLLM-1.7B-Instruct

3.2 Datasets

This section presents the datasets utilized for training and evaluating our models. We divide these datasets into training and evaluation datasets for both machine translation and summarization tasks:

3.2.1 Machine Translation

Training:

- **MT-Pref** (Agrawal et al., 2024) **M**etric-induced **T**ranslation **Pref**erence dataset is a high-quality translation preference dataset, which comprises 18k instances covering 18 language directions, using texts sourced from multiple domains post-2022. This includes English, German, Chinese, Russian, Portuguese, Italian, French, Spanish, Korean and Dutch languages. For each source sentence, translations are generated by various MT systems, representing a range of architectures, training data, and quality levels. The dataset includes both the most and least preferred translations from the set of hypotheses. Agrawal et al. (2024) demonstrated improved translation quality on the WMT23 (Kocmi et al., 2023) and FLORES (Team et al., 2022) benchmarks when aligning on MT-Pref. Their study also shows that aligned models better rank translations according to human preferences over baselines.

Evaluation:

- **WMT23 General MT Data**[8] (Kocmi et al., 2023) A freely available dataset for research for the WMT23 General MT task featuring diverse data sources for comprehensive machine translation research. It includes Europarl for political texts, ParaCrawl and Common Crawl for web-crawled content, News Commentary for bilingual news, etc. This dataset supports a wide range of language pairs and is ideal for training and evaluating translation models across varied text types and domains.

- **FLORES-200** (Team et al., 2022) Building upon the framework established by its predecessor, FLORES-101 (Goyal et al., 2021), the FLORES-200 dataset by Meta AI is a comprehensive multilingual evaluation benchmark that extends its coverage to 200 languages, significantly enhancing support for low-resource languages. This benchmark focuses on machine translation performance for both high-resource and diverse, low-resource languages. The dataset includes a range of text types and languages, from major ones like English, French, and Chinese to less-documented languages. Our study will specifically focus on translation pairs involving English: English-German (`en`↔`de`), English-French (`en`↔`fr`), English-Portuguese (`en`↔`pt`), English-Dutch (`en`↔`nl`), English-

[8] https://www.statmt.org/wmt23/mtdata/

Spanish (en↔es), English-Italian (en↔it), English-Chinese (en↔zh), English-Korean (en↔ko), and English-Russian (en↔ru).

3.2.2 Summarization

The following datasets were utilized for both training and evaluation of our models for the summarization task, with appropriate train and test splits considered for each stage.

Training & Evaluation:

- **Reddit TL;DR** (Stiennon et al., 2020) a summarization dataset, derived from the Webis-TLDR-17 Corpus (Völske et al., 2017), consists of ∼3 million Reddit posts covering various topics (subreddits) with human-written summaries (TL;DRs) provided by the original posters. This dataset was refined by OpenAI through the application of a quality filter, including a whitelist of subreddits that are understandable to the general population. Importantly, it was further refined to include only posts where the human-written summaries contain between 24 and 48 tokens to reduce the influence of summary length on quality. The resulting high-quality dataset contains ∼123k posts, with about 5% reserved as a validation set. Throughout this work, we refer to it simply as TL;DR.

- **OpenAI TL;DR Human Feedback** (Stiennon et al., 2020) On top of the previously mentioned dataset OpenAI also released a high-quality dataset of human comparisons between summaries. This dataset contains 64,832 summary comparisons derived from the TL;DR dataset, along with the preferences indicated by humans. The evaluators underwent training to achieve high concordance in their judgments, with ongoing checks to maintain consistency between the evaluators and researchers. Throughout this work, we refer to it simply as Summarize.

4

Implementation Details

Contents

Building on the insights gathered from the literature analysis, the architectural decisions, and the objectives of this study, this section presents our detailed approach for training for MT and summarization tasks.

4.1 Development Process

As described in Section 2.6, prominent RL-free models have a competitive performance to RLHF and overcome many of its drawbacks, including complexity, stability, computational weight, and significant hyperparameter tunning.

Building on this insight, our methodology encompasses a comparative analysis of both established models and those we have developed using baseline architectures, with a specific focus on machine translation and summarization tasks. We conduct a detailed evaluation of the developed models, applying SFT and Preference Training with well-known Preference Optimization algorithms such as DPO, CPO, SimPO, and SLiC, specifically within the context of small language models (SLMs).

Starting with backbone models that have been pre-trained to autoregressively predict the next token in extensive text corpora, as detailed in 3. These models undergo SFT employing maximum likelihood estimation to adapt to our specific tasks of machine translation and summarization.

Building upon established frameworks, we leverage tools such as the Transformer Reinforcement Learning (TRL) library (Leandro von Werra et al., 2020), alongside optimizations from DeepSpeed ZeRO3 (Rajbhandari et al., 2020) and FlashAttention-2 (Dao, 2023), enhancing training speed and memory efficiency, following methodologies similar to those reported in recent studies (Tunstall et al., 2023; Rafailov et al., 2023).

Subsequently, we implement prominent preference optimization methods to either the baseline backbone models or the SFT models, as appropriate. These methods include **DPO** (Direct Preference Optimization) (Rafailov et al., 2023), **SimPO** (Simple Preference Optimization) (Meng et al., 2024), **CPO** (Contrastive Preference Optimization) (Xu et al., 2024b), and **SLiC** (Sequence Likelihood Calibration) (Liu et al., 2023). Given the innovative nature of this approach, particularly in aligning small language models with human preferences, we also explore our own custom variants, **DPO-**γ and **SLiC-DPO** - we present the details in Section 4.2. These unique modifications are specifically designed to optimize the original algorithms for the distinctive challenges presented by smaller language models. Further details on the final models and their configurations are elaborated in Section 4.2, providing a comprehensive overview of the implementations and their theoretical foundations.

4.1.1 Foundational Codebase

As an initial codebase for this work, we used Alignment Handbook[1]. This handbook serves as a valuable resource for the community, offering a comprehensive collection of robust training recipes that encompass the entire training pipeline. It provides essential guidelines and best practices for effectively aligning language models with desired outcomes, thereby facilitating improved performance in various applications.

Each provided script supports distributed training of the complete model weights using tools like DeepSpeed ZeRO-3 or LoRA/QLoRA for parameter-efficient fine-tuning. The handbook includes practical recipes for replicating the alignment on models such as Zephyr-7B (Tunstall et al., 2023).

4.1.2 Our Codebase

Despite the initial utility of the Alignment Handbook, our preliminary results necessitated a strategic pivot. We customized our own codebase, initially based on the Alignment Handbook and TRL, but with

[1] https://github.com/huggingface/alignment-handbook

modifications to better suit our tuning needs. This customized approach leverages the Tower Alignment codebase from the DeepSpin project[2], which focuses on integrating deep learning with structured prediction to address challenges in NLP, such as improving machine translation and reducing errors like word dropping and entity mistranslation.

Our codebase includes adjustable parameters such as `contrast_loss_type`, `sft_type`, `generate_during_eval`, `average_log_prob`, `reference_free`, `lambda_contrast`, and `lambda_sft`. These were tuned to adjust to the different preference optimization methods. Adjustments in these parameters allow for nuanced adjustments to the loss function, evaluation process, and normalization techniques, thereby enabling the rapid setup and optimization of different preference algorithms for enhanced outcomes.

Given that the original Tower Alignment codebase design was intended for larger machine translation models, modifications in the code were necessary to accommodate smaller models and extend applicability to the summarization task. This revised codebase was instrumental in replicating experiments related to hyperparameter tuning and was utilized to train our models using TinyLlama, Gemma-2, and EuroLLM, adjusting parameters as needed to optimize performance.

4.1.3 Training Procedure

Prior research on preference algorithms predominantly focuses on larger models, typically ranging from 6B to 13B parameters, demonstrating the efficacy of strategies like DPO and SimPO in tasks like summarization for certain hyperparameter configurations. However, results for smaller models remain unreported. Although results from larger models provide valuable insights and serve as a reference, considerable effort was invested in optimizing hyperparameters for smaller models to ensure effective training for the machine translation and summarization tasks, while ensuring a robust basis for comparison amongst them.

To meet our specific requirements, we adapted scripts from the Alignment Handbook to our specific needs, incorporating our custom additional adjustable parameters and fine-tuning several key hyperparameters, such as learning rate, gradient accumulation steps, warmup steps, maximum sequence length, batch size per device, and the number of GPUs. The adjustments were guided by systematically studying the outcomes of different configurations and exploring values that consistently improved results. This approach allowed us to tailor the training process to the needs of smaller models, balancing memory usage with training speed on the available hardware and ensuring effective model adaptation.

Our experience revealed that preference algorithms like DPO and CPO exhibit significant instability when certain parameters are altered. Small models easily exploit shortcuts to minimize their loss, leading to over-optimization that compromises their ability to generate coherent, high-quality, human-readable text. For example, larger learning rates often led to the usage of more shortcuts, resulting in

[2] https://deep-spin.github.io/

excessive optimization and potential bias, such as unnecessary increased length.

An interesting observation for SLMs is that even though training and evaluation losses decrease during training, automatic metrics like ROUGE and BLEU do not always show significant improvements. This discrepancy arises because the loss function optimizes the likelihood of the correct output, which may not necessarily align with evaluation metrics that assess fluency, coherence, and relevance of the generated output. As a result, while the model might achieve lower loss by predicting more likely tokens, the generated outputs may still lack the qualities measured by complex metrics. Thus, decreasing loss does not always guarantee that the outputs reflect better real-world performance and alignment with human preferences.

Recognizing this issue and the need for meticulous tuning due to the smaller size of our models, we tested various hyperparameter configurations which led to significantly varied outcomes, as shown in Chapter 5. These multiple initial experiments allowed us to evaluate the effects of different hyperparameters on training stability, performance, and alignment with human preferences. The hyperparameters tested include:

Parameter	**Values**
Learning Rate	$\{5 \times 10^{-8}, 1 \times 10^{-7}, 2 \times 10^{-7}, 3 \times 10^{-7}\}$
Batch Size	$\{16, 32, 64, 128\}$
Beta (β)	$\{0.01, 0.05, 0.1\}$
Number of Epochs	$\{1, 2, 3\}$
Warmup Ratio	$\{0.1, 0.15\}$
Gradient Accumulation Steps	$\{4, 8, 16, 32\}$
Number of Devices	$\{1, 2, 4\}$

Table 4.1: Tested hyperparameter during training of our models.

Ultimately, we fix the hyperparameters at values that demonstrated stable performance across all models. This ensured a reliable comparison of RL-free preference optimization methods on small LMs and enabled a thorough evaluation of the models' ability to align with human preferences. This was necessary given the lack of prior studies on hyperparameter tuning for SLMs. While hyperparameter tuning was not the primary focus, our findings indicate that further fine-tuning could lead to notable improvements, particularly when tailored to each preference optimization method. Table 4.2 details the final training hyperparameters shared across all models. Approach-specific parameters are detailed in Section 4.2.

The final parameter selection balanced output quality and available resources, as training time ranged from 3 to 7 hours for machine translation and 6 to 52 hours for summarization. We trained our models using 4 NVIDIA RTX A6000 48GB GPUs, enabling efficient parallel processing and accelerating the training time.

Parameter	**Configuration**
Learning Rate	1×10^{-7}
Contrast Loss Type	Sigmoid
Train batch size (per device)	1
Eval batch size (per device)	4
Number of devices	2 for TinyLlama; 4 for Gemma-2 and EuroLLM; 4 for all Summarization models
Number of Epochs	3
Warmup Ratio	0.1
Lora	False
Gradient Accumulation Steps	32 for TinyLlama and EuroLLM; 16 for Gemma-2 and Summarization models [3]
Flash Attention	True
Optimizer	RMSProp

Table 4.2: Shared Preference Optimization Methods Configuration

4.2 Trained models

We implement **DPO** (Rafailov et al., 2023), **CPO** (Xu et al., 2024b), and **SimPO** (Meng et al., 2024) over the respective baseline for both TinyLlama 1.1B and Gemma-2 2B models for the machine translation and summarization tasks. Additionally, we implement **SLiC** (Zhao et al., 2023) for both tasks, however, due to resource constraints, implementing SLiC for summarization with the Gemma-2 model was not feasible. For the EuroLLM model, all four methodologies are implemented specifically for the machine translation task.

DPO-γ. Following our objective of aligning the model with human preferences, we develop a modification of DPO which we name DPO-γ. This modification combines the policy objective from DPO, presented in Equation 2.11, and SimPO's target reward margin term, $\gamma > 0$, introduced to the Bradley-Terry objective to ensure that the reward for the winning response, $r(x, y_w)$, exceeds the reward for the losing response, $r(x, y_l)$, by at least γ. The resulting objective function for DPO-γ is defined as follows:

$$\mathcal{L}_{\text{DPO-}\gamma}(\pi_\theta; \pi_{\text{ref}}) = -\mathbb{E}_{(x,y_w,y_l)\sim D}\left[\log\sigma\left(\beta\log\frac{\pi_\theta(y_w \mid x)}{\pi_{\text{ref}}(y_w \mid x)} - \beta\log\frac{\pi_\theta(y_l \mid x)}{\pi_{\text{ref}}(y_l \mid x)} - \gamma\right)\right]. \tag{4.1}$$

where $\pi_{\text{ref}} = \pi_{\text{SFT}}$. Similarly to DPO, an implicit reward can be fitted in such a way that the optimal policy simply becomes π_θ.

SLiC-DPO. Also aligning with DPO's objective, we implement another custom approach, SLiC-DPO, which integrates DPO's reward reparameterization with SLiC's rank calibration loss. This combination results in the following loss function:

$$\begin{aligned}\mathcal{L}_{\textsf{SLiC-DPO}}(\pi_\theta;\pi_{\textsf{ref}}) &= \max\Big(0, \delta - \beta\Big(\log\frac{\pi_\theta(y_w \mid x)}{\pi_{\textsf{ref}}(y_w \mid x)} - \log\frac{\pi_\theta(y_l \mid x)}{\pi_{\textsf{ref}}(y_l \mid x)}\Big)\Big)\\ &= \max\Big(0, \delta - \beta\log\frac{\pi_\theta(y_w \mid x)}{\pi_{\textsf{ref}}(y_w \mid x)} + \beta\log\frac{\pi_\theta(y_l \mid x)}{\pi_{\textsf{ref}}(y_l \mid x)}\Big).\end{aligned} \tag{4.2}$$

While DPO-γ was implemented across all models and tasks, SLiC-DPO is not implemented in Gemma-2 for the summarization task due to resource constraints. To clarify distinctions between the two SLiC variants, the original approach is referred to as SLiC-CPO.

Additional Parameters. We set $\beta = 0.1$ for all our models by default, as done in Rafailov et al. (2023). For SimPO and DPO-γ, we use $\gamma = 0.5$, and for SLiC-based approaches, we use $\delta = 1$. Other training parameters are detailed in Table 4.2.

MT prompt template. We fine-tune our MT models using the `chatml` template (Casper et al., 2023); we provide an example in Appendix A.1.1.

Summarization prompt template. We fine-tune our Summarizaion models using the same template as Stiennon et al. (2020); we provide an example in Appendix A.1.2.

5

Experimental Results

Contents

5.1 Evaluation Setup

5.1.1 Automated Evaluation Metrics

Automated evaluation metrics are important in the iterative development and tuning of language models. They provide quantitative measures that reflect a model's performance in terms of linguistic accuracy and quality of text, in both machine translation and summarization tasks.

We begin by assessing each model with automated evaluation metrics, as these provide an overview of their performance and enable comparisons both among models using the same backbone and those employing different backbone models. The metrics used for each task are summarized in Table 5.1, followed by detailed descriptions of each metric.

Task	Metrics
Machine Translation	chrF (Popović, 2015) ↑
	BLEU (Papineni et al., 2002) ↑
	COMET-22 (Rei et al., 2020) ↑
	XCOMET (Guerreiro et al., 2023) ↑
Summarization	ROUGE (Lin, 2004) ↑
	METEOR (Banerjee and Lavie, 2005) ↑
	BERTScore (Zhang et al., 2020) ↑
	Reward (Song et al., 2023) ↑

Table 5.1: Metrics for Evaluating Machine Translation and Summarization. Higher values are preferred for all metrics, indicated by the upward arrow ↑.

chrF (Popović, 2015) Character n-gram F-score assesses the quality of translations by focusing on character-level accuracy.

BLEU (Papineni et al., 2002), Bilingual Evaluation Understudy, measures the precision of machine-generated translations by comparing them with one or more reference translations. It includes a brevity penalty to discourage overly short translations, thus promoting adequacy and fluency in evaluated translations.

COMET-22 (Rei et al., 2020) is an advanced neural framework designed to evaluate translation quality by comparing outputs with both their source and reference texts. Utilizing sophisticated machine learning methodologies, it produces scores that align closely with human judgments. As a state-of-the-art metric for assessing translation quality, COMET has demonstrated a significant correlation with human evaluations (Kocmi et al., 2021).

XCOMET (Guerreiro et al., 2023) extends the capabilities of COMET to a broader range of languages and more complex evaluation scenarios. It uses extensive training data and sophisticated model architectures to provide detailed assessments of translation quality across diverse linguistic contexts.

ROUGE (Lin, 2004), Recall-Oriented Understudy for Gisting Evaluation, evaluates the overlap of n-grams between the machine-generated outputs and reference outputs, focusing on recall to assess the amount of reference content captured by the generated text. This metric is particularly useful in summarization tasks where content coverage is crucial.

METEOR (Banerjee and Lavie, 2005), Metric for Evaluation of Translation with Explicit Ordering, was originally developed to assess the quality of machine translations, but it can also be used to assess the quality of machine summaries. It takes into account not only exact matches but also synonyms and paraphrases. Unlike simpler metrics that focus solely on precision, METEOR incorporates both precision and recall, providing a more complete evaluation of translation and summarization outputs. It also evaluates grammatical and semantic similarity, fluency, and addresses common issues such as word order, synonyms, and ambiguities.

BERTScore (Zhang et al., 2020) utilizes the contextual embeddings from models like BERT to compute cosine similarity scores between the tokens of the generated and reference texts. It evaluates semantic and syntactic similarity more effectively than overlap-based metrics, making it suitable for tasks where nuanced understanding is important.

Reward (Song et al., 2023) For evaluating summarization quality we utilized Deberta-v3-large reward model[1], which was trained using human feedback. This reward models is designed to assign a reward score to generated outputs and, therefore, and, when presented with two possible outputs for a given prompt, predict the human preference efficiently. This model was similarly utilized in the PRO study (Song et al., 2023), underscoring its relevance and effectiveness in our research context.

5.1.2 Generation

We utilized vLLM (Kwon et al., 2023) for its high-throughput capabilities and efficient memory management to generate translations and summaries finding that it offered superior performance and output quality compared to other methods, including Hugging Face Pipelines. While Hugging Face Pipelines provide a streamlined API for various NLP tasks, vLLM stood out due to its faster processing and lower memory usage, complemented by the simplicity of its code and parameter configuration. This efficiency enabled rapid and scalable experimentation with various model configurations. Additionally, its seamless integration with Hugging Face models enabled efficient and flexible testing across various settings. Furthermore, vLLM's support for sophisticated decoding algorithms allowed for precise fine-tuning of output quality. Ultimately, vLLM's comprehensive performance benchmarking ensured the reliability and accuracy of our results, enabling us to assess the effectiveness of small language models in handling complex NLP tasks.

Temperature For MT, all model generations are performed with greedy decoding, that is, with temperature 0. Consequently, adjustments to parameters such as *top-k* were deemed unnecessary. *Top-k* limits the selection to the k most probable next tokens. Similarly, changes to *top-p*, which defines the cumulative probability threshold for token selection, are also unnecessary. This is because the greedy search inherently focuses on the most probable token, eliminating the need for additional constraints.

To establish a robust basis for comparison in summarization tasks, all model generations were also performed using greedy decoding. We observed that temperature settings significantly influence the

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

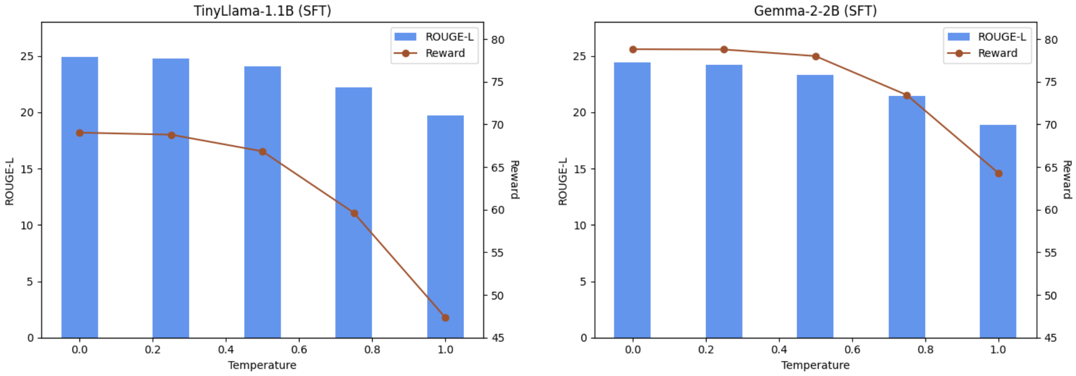

Figure 5.1: BLEU and COMET scores for outputs generated with different temperatures by the SFT model for English to German (en→de) and Chinese to English (zh→en).

evaluation metrics, with greater differences emerging as temperature varies. Figure 5.1 presents the variations in ROUGE and Reward for our SFT models, a trend that is consistently replicated across the various preference models. Metrics demonstrated to be very sensitive to temperature changes, showing a noticeable decrease at higher temperatures, particularly for bigger *top-k* values.

5.1.3 Baselines

For the task of machine translation, we extensively evaluate all our models, comparing with their respective base models or instruction-tuned versions, as applicable, and with open models TowerBase 7B, TowerBase 13B and TowerInstruct 7B (Alves et al., 2024b). For both TowerBase 7B and TowerBase 13B, we provide 5 in-context learning examples randomly selected from the development set. All other MT models were assessed in a zero-shot fashion.

For summarization, we compare our models against their backbone model and against Llama 7B fine-tuned with PRO (Song et al., 2023). Consistent with MT evaluations, all summarization models were also evaluated in a zero-shot fashion.

5.2 Model Performance

5.2.1 Machine Translation

We evaluate our fine-tuned MT models on the WMT23 test set (Kocmi et al., 2023), which considers translations across multiple English-centric language pairs and covers both translations from and to English (en↔{de,ru,zh}). Table 5.2 shows the obtained results. We also include the results on the FLORES-200 dev-test set (Team et al., 2022) (en↔{de,ru,zh,es,fr,pt,nl,it,ko}) in Table 5.5. These evaluations were done using Tower-Eval (Alves et al., 2024a). We report system-level translation quality using chrF (Popović, 2015), BLEU (Papineni et al., 2002), COMET (Rei et al., 2020) and XCOMET (Guerreiro et al., 2023).

Backbone models perform poorly prior to any fine-tuning. Both TinyLlama and Gemma-2 exhibit suboptimal initial performances due to insufficient fine-tuning specific to their target tasks. TinyLlama, notably, scores only 18.38 in chrF and 11.35 in BLEU for translations from English, as shown in Table 5.2. Gemma-2, though instruction-tuned which contributes positively to the model's baseline performance, also falls short because the tuning was not specifically focused on machine translation tasks.

While these models provide a solid foundation for our study, we also explore EuroLLM-1.7B (Martins et al., 2024) — a recent state-of-the-art model specifically designed for machine translation — to strengthen our results. Despite not yet being aligned with human preferences, Table 5.2 shows that EuroLLM-1.7B-Instruct performance achieves significantly higher values compared to the other two backbone models and presents comparable performance against TowerBase models – 7B and 13B – due to its instruction-tuning using `chatml` template (Casper et al., 2023) and EuroBlocks, an instruction tuning dataset with focus on general instruction-following and machine translation.

SFT increases both lexical and neural quality of the model translations. Applying SFT increases the performance of the backbone models. It enables the models to adhere to the designated chat prompts and generate more complete and accurate translations which, in turn, results in increasing both lexical and neural metrics. For instance, TinyLlama shows a notable improvement in the $\texttt{en}\rightarrow\texttt{xx}$ direction, with its chrF score improving from 18.28 to 34.01, and in its COMET score in the $\texttt{xx}\rightarrow\texttt{en}$ direction increasing from 48.51 to 76.97. On Gemma-2, SFT alone is sufficient to surpass larger models, like TowerBase-7B and TowerBase-13B, in translation quality from $\texttt{xx}\rightarrow\texttt{en}$.

Conversely, for EuroLLM-1.7B-Instruct, which was instruction-tuned using a chat template and high-quality datasets, the application of SFT does not yield substantial improvements across all metrics. This small enhancement suggests that the instruction-tuned model already performs at a high level due to extensive instruction-tuning processes, making it more powerful and robust. Given these observations, we report results both with and without SFT prior to preference optimization algorithms in Table 5.2. We aim to determine whether instruction tuning alone suffices for achieving desirable results, or if additional enhancements can be achieved through SFT.

DPO methods consistently outperform the SFT model and other preference-based methods. Both DPO, DPO-γ, and SLiC-DPO demonstrate the best performance when considering the COMET (Rei et al., 2020) and XCOMET (Guerreiro et al., 2023) metrics compared to other preference optimization methods. When applied on the SFT model, DPO-based methods, including DPO, DPO-γ, and SLiC-DPO, exhibit the better performances among all preference optimization techniques, leading consistently to the better top 3 COMET scores – as shown in Table 5.2 – indicating superior translation

Models	en→xx				xx→en			
	chrF	BLEU	COMET	XCOMET	chrF	BLEU	COMET	XCOMET
				TinyLlama-1.1B				
TinyLlama-1.1B	18.38	11.35	48.84	51.47	24.98	11.03	48.51	55.05
| + SFT	34.01	14.46	68.74	**63.31**	48.43	22.25	76.97	74.61
| + DPO	35.75	15.79	69.21	62.99	50.06	**22.63**	77.34	**74.93**
| + DPO-γ	35.88	15.97	**69.24**	63.03	50.14	22.60	**77.37**	74.90
| + SimPO	**36.27**	15.15	67.09	59.51	**50.90**	21.48	76.37	71.68
| + SLiC-DPO	35.99	**16.34**	69.23	62.96	50.02	22.56	77.36	74.89
| + CPO	19.47	5.61	50.78	54.09	35.88	14.32	61.52	56.97
| + SLiC-CPO	12.47	3.57	38.56	41.43	28.64	10.42	53.37	47.38
				Gemma-2-2B-IT				
Gemma-2-2B-IT	27.38	11.35	55.27	74.96	41.90	24.98	72.18	73.83
| + SFT	48.07	31.07	81.28	83.46	56.94	**32.21**	81.78	85.81
| + DPO	48.53	30.72	**82.42**	**84.28**	57.47	31.88	**82.21**	**87.10**
| + DPO-γ	48.52	30.70	**82.42**	84.27	57.50	31.84	**82.21**	87.02
| + SimPO	**48.72**	29.62	81.37	82.42	**58.11**	30.59	81.78	86.24
| + SLiC-DPO	48.55	**31.09**	82.39	82.43	57.33	31.92	82.15	86.24
| + CPO	43.30	27.41	77.33	82.54	53.74	27.81	80.43	84.96
| + SLiC-CPO	42.01	26.43	76.75	84.41	53.57	27.54	80.39	86.91
				EuroLLM-1.7B-Instruct				
EuroLLM-1.7B-Instruct	49.42	32.68	82.46	83.81	56.40	30.90	81.48	84.70
| + SFT	50.48	33.83	82.64	84.02	57.07	31.69	81.68	85.22
| + DPO	50.81	33.73	82.96	**84.32**	57.25	31.71	**81.91**	**85.63**
| + DPO-γ	50.95	**34.18**	**82.99**	84.30	57.32	31.65	**81.91**	85.57
| + SimPO	**51.00**	33.23	82.92	84.13	**57.55**	31.62	**81.91**	85.46
| + SLiC-DPO	**51.00**	34.10	82.97	84.23	57.32	31.72	81.89	85.45
| + CPO	50.01	33.48	82.66	83.94	56.44	30.66	81.41	84.58
| + SLiC-CPO	49.67	32.94	82.64	84.00	56.52	30.79	81.45	84.82
| + DPO	49.37	32.71	82.60	84.02	56.67	**30.87**	**81.62**	84.99
| + DPO-γ	49.58	32.88	**82.65**	**84.03**	**56.70**	30.77	**81.62**	84.99
| + SimPO	49.27	32.26	82.44	83.73	55.21	29.68	80.93	83.83
| + SLiC-DPO	**49.64**	**32.93**	82.61	83.96	56.68	30.68	**81.62**	**85.00**
TowerBase-7B	51.65	35.02	82.76	88.01	56.84	31.52	81.03	87.91
TowerBase-13B	51.77	35.21	82.89	88.12	57.72	32.20	80.95	87.93
TowerInstruct-7B	52.25	35.89	84.28	88.71	59.78	33.19	82.77	88.65

Table 5.2: Machine translation model results on WMT23 dataset. Best ranked models for each metric are in bold. Dark blue boxes indicates that the improvement over the SFT baseline is considerable. The lesser improvements are highlighted in shallow blue boxes. Decreases in performance are marked with red boxes. For EuroLLM, performance against the base model is also highlighted.

quality. **For all models, DPO-γ shows consistently good results, outperforming DPO and SLiC-DPO in the COMET metric across both translation directions.** The variance in performance metrics between DPO and its variant, DPO-γ, underscores the influence of the γ parameter. This adjustment notably impacts lexical metrics (i.e. chrF and BLEU), confirming the γ parameter's role when introduced in Equation 4.1.

SFT outperforms CPO. While CPO enhances translation quality over the backbone model, the performance gains fall short compared to the ones achieved with SFT. The disparity in performance, particularly in COMET and xCOMET metrics, ranges up to 14% for TinyLlama and 5% for Gemma, whereas in lexical metrics they can reach ~62%. The gap widens further when other preference optimization algorithms are layered on top of SFT. In the case of EuroLLM, SFT outperforms CPO across nearly all metrics except for COMET in en→xx direction, where the difference is very small although relevant.

We conclude that initial fine-tuning seems imperative for the model to grasp the essential characteristics of the expected desired translation outputs. With this consideration, we further evaluate the impact of the tuning learning rate on the training of CPO for TinyLlama. This adjustment impacts the negative log likelihood loss term, which corresponds to the SFT term, critically affecting model performance.

Learning Rate	chrF	BLEU	COMET	xCOMET
5.0e-8	0.72	0.0	0.1912	0.2313
1.0e-7	34.24	8.43	0.5358	0.5177
5.0e-7	42.49	10.44	0.6326	0.5463

Table 5.3: CPO on TinyLlama Model Performance with different learning rates during training.

As illustrated in Table 5.3, the sensitivity of CPO to training hyperparameters in smaller language models is evident: a higher learning rate yields better results, whereas a lower learning rate results in significantly worse performance. This emphasizes the necessity for precise hyperparameter tuning in these models. Additionally, it underscores the importance of including the SFT term in CPO's loss function to enhance the model's ability to comprehend the intricacies of the language task.

SLiC-CPO degrades the performance of the TinyLlama backbone model in en→xx translations. In the case of TinyLlama, both CPO and SLiC-CPO reduce the BLEU scores in the en→xx translations, with SLiC impacting all performance metrics negatively in this direction. Additionally, while there is a decrease in BLEU and XCOMET scores in translations into English (en→xx), SLiC-DPO's COMET score see an improvement compared to the backbone model.

SFT+CPO outperforms SimPO but remains behind DPO methods. To further explore the gap between SFT and CPO, we present the results for the TinyLlama and Gemma models, where CPO was applied to the SFT models instead of the base model. Even with CPO applied to the SFT models, DPO methods continue to deliver superior performance, as seen in Table 5.4. CPO achieves the highest XCOMET score in en→xx and, in Gemma-2, records the best BLEU score in the xx→en direction. Despite these successes, in all other metrics, CPO falls short, especially in COMET scores for translations from German, Russian, and Chinese to English.

Unlike SimPO, which causes a decline in performance when applied to SFT models, CPO brings performance closer to DPO methods when applied to the SFT model. This further emphasizes the

Models	en→xx				xx→en			
	chrF	BLEU	COMET	XCOMET	chrF	BLEU	COMET	XCOMET
TinyLlama-1.1B + SFT								
+ DPO	35.75	15.79	69.21	62.99	50.06	22.63	77.34	**74.93**
+ DPO-γ	35.88	15.97	**69.24**	63.03	50.14	22.60	**77.37**	74.90
+ SimPO	**36.27**	15.15	67.09	59.51	**50.90**	21.48	76.37	71.68
+ SLiC-DPO	35.99	**16.34**	69.23	62.96	50.02	22.56	77.36	74.89
+ **CPO**	35.04	14.74	69.18	**63.08**	49.51	22.28	77.31	74.77
Gemma-2-2B-IT + SFT								
+ DPO	48.53	30.72	**82.42**	84.28	57.47	31.88	**82.21**	**87.10**
+ DPO-γ	48.52	30.70	**82.42**	84.27	57.50	31.84	**82.21**	87.02
+ SimPO	**48.72**	29.62	81.37	82.42	**58.11**	30.59	81.78	86.24
+ SLiC-DPO	48.55	**31.09**	82.39	82.42	57.33	31.92	82.15	86.24
+ **CPO**	48.47	31.32	82.40	**84.37**	57.29	**32.14**	81.95	86.52

Table 5.4: Comparison of SFT+CPO with DPO, DPO-γ, SimPO, and SLiC-DPO for TinyLlama and Gemma models. Best ranked models for each metric are in bold.

importance of the SFT in training these smaller language models. Furthermore, CPO is a reference-free method and does not require storing a reference policy π_{ref}, yet it achieves similar performance than DPO-based approaches, making it particularly relevant in constrained environments where memory and speed efficiency are important considerations.

Although SLiC-DPO shows comparable results, it exhibits a marginal decline in COMET scores and a more noticeable reduction in XCOMET scores. This trend suggests a potential reduction in translation quality, particularly at the granular level, as XCOMET provides sentence-level evaluation and error span detection.

While SimPO achieves the highest chrF scores, its performance does not consistently outperform SFT models. SimPO is the only preference optimization method that degrades the performance of the SFT baseline for TinyLlama. Although there are visible gains in chrF and BLEU metrics, when we consider metrics like COMET and XCOMET, which focus more on semantic accuracy and contextual appropriateness, there is a decrease in performance. This decline may be attributed to SimPO's training objective, which tends to prioritize superficial n-gram matching over substantial quality enhancements in model outputs. For Gemma-2, SimPO shows an improvement in chrF and a downgrade in performance in BLEU in both directions (en↔xx). It also shows a small improvement in COMET for en→xx and no improvement nor downgrade in xx→en over the SFT baseline. In contrast, XCOMET experiences a decrease in en→xx while improving in xx→en. For EuroLLM, there is an enhancement across all metrics compared to the SFT baseline, except for BLEU, which sees a reduction.

SimPO achieves the highest chrF scores across all models but exhibits the poorest BLEU scores when compared to other alignment methods applied on the SFT model, even downgrading the SFT model's performance. Notably, SimPO is the only reference-free model applied on top of SFT. The absence of a reference model during training cuts down on memory and computational demands. However, this approach can cause performance fluctuations in SLMs, as evidenced in our results.

DPO-based models without pior SFT outperform EuroLLM-1.7B-Instruct base model. Both DPO, DPO-γ, and SLiC-DPO outperform EuroLLM-1.7B-Instruct base model when considering the COMET (Rei et al., 2020) and XCOMET (Guerreiro et al., 2023) metrics. As for lexical metrics in both language directions, there is no clear trend, but the results tend to favor DPO-based models over the base model. On the other hand, SimPO degrades base model's performance according to automatic metrics. This indicates that even without SFT, preference optimization can enhance the model's performance, albeit marginally. Further enhancements of these models regarding human preference learning are detailed in Section 5.3.

DPO-based models and SimPO on EuroLLM outperform TowerBase-7B and TowerBase-13B in COMET. While EuroLLM-1.7B-Instruct shows comparable performance to TowerBase-7B, when further improved through SFT and various preference optimization methods it surpasses both TowerBase models – 7B and 13B – in COMET metric.

Results on FLORES Dataset. We further evaluate our models on the FLORES dataset to assess their performance across different datasets. The results are presented in Table 5.5. TinyLlama model, without any fine-tuning, performs poorly in both language directions ($\texttt{en}\leftrightarrow\texttt{xx}$). Similarly, Gemma-2-2B demonstrates limited effectiveness in translations ($\texttt{en}\leftrightarrow\texttt{xx}$), yet it exhibits greater proficiency in translating from various languages into English. Mirroring what was observed on WMT23, SFT significantly improves the performance, increasing every metric score in both directions for every model, with TinyLlama showing the greatest improvements. Observations from the WMT23 dataset are replicated on the FLORES dataset: 1) **DPO methods consistently outperform the SFT model and other preference-based methods**; 2) **while SimPO achieves the highest chrF scores in from English to the target language, it either maintains or degrades COMET values compared to SFT for the TinyLlama and Gemma**; and 3) **DPO-based models without pior SFT outperform EuroLLM-1.7B-Instruct base model**.

Remarkably on FLORES, **all our preference optimization models using EuroLLM-1.7B-Instruct show comparable COMET values against TowerBase-7B** and come close to TowerBase-13B and TowerInstruct 7B, all models of much superior size. Compared to TowerBase-7B, our models show higher translation quality for $\texttt{en}\rightarrow\texttt{xx}$ – with superior COMET values – but underperform in $\texttt{xx}\rightarrow\texttt{en}$. Despite not being specifically instruction-tuned for machine translation, Gemma-2 models also perform close to TowerBase-7B, showcasing their efficacy despite their smaller size.

Models	en→xx			xx→en		
	chrF	BLEU	COMET	chrF	BLEU	COMET
TinyLlama-1.1B						
Tinyllama	21.90	5.30	58.65	44.44	17.97	75.27
| + SFT	39.52	16.25	72.61	54.77	26.11	83.33
| + DPO	40.53	**16.78**	73.18	55.44	25.92	**83.56**
| + DPO-γ	40.51	16.70	**73.19**	**55.45**	25.89	83.54
| + SimPO	**40.77**	16.24	70.72	55.12	23.37	82.11
| + SLiC-DPO	40.46	16.74	73.05	55.41	25.86	**83.56**
| + CPO	25.55	7.33	61.59	47.28	20.25	76.44
| + SLiC-CPO	17.84	3.84	49.15	40.44	17.25	68.31
Gemma-2-2B-IT						
Gemma-2-2B-IT	22.89	7.87	63.69	58.41	30.65	84.53
| + SFT	51.40	29.75	85.92	61.70	34.22	87.41
| + DPO	51.76	29.69	86.34	61.85	33.65	**87.54**
| + DPO-γ	51.80	29.75	86.35	61.84	33.59	**87.54**
| + SimPO	**51.87**	29.11	85.67	61.33	31.10	87.02
| + SLiC-DPO	51.77	**29.85**	**86.36**	**61.86**	**33.77**	**87.54**
| + CPO	46.79	25.95	82.01	61.05	32.11	87.08
| + SLiC-CPO	46.07	25.40	81.46	61.01	32.01	87.07
EuroLLM-1.7B-Instruct						
EuroLLM-1.7B-Instruct	54.41	33.15	87.57	62.19	34.23	87.54
| + SFT	54.98	33.87	87.58	62.56	**34.35**	87.69
| + DPO	55.18	34.08	87.73	62.67	34.33	**87.75**
| + DPO-γ	55.21	**34.09**	87.74	62.67	34.33	87.74
| + SimPO	**55.26**	34.03	87.71	62.64	34.15	87.71
| + SLiC-DPO	55.18	34.04	**87.75**	**62.68**	34.34	**87.75**
| + CPO	54.80	33.47	87.63	62.37	34.24	87.57
| + SLiC-CPO	54.64	33.33	87.63	62.32	34.21	87.58
| + DPO	**54.74**	**33.43**	87.69	**62.31**	**34.09**	**87.58**
| + DPO-γ	54.73	33.39	**87.70**	62.29	34.02	87.58
| + SimPO	54.63	33.23	87.57	62.16	33.69	87.49
| + SLiC-DPO	**54.74**	33.39	87.69	62.30	34.01	**87.58**
TowerBase-7B	55.08	33.95	87.45	63.63	37.33	88.02
TowerBase-13B	55.67	34.32	87.90	54.75	37.41	88.16
TowerInstruct-7B	56.15	35.28	88.51	64.09	37.79	88.27

Table 5.5: Results for machine translation on FLORES dataset by language pair. We considered all the following language pairs: en↔de, en↔fr, en↔pt, en↔nl, en↔es, en↔it, en↔zh, en↔ko, and en↔ru. Best ranked models for each metric are in bold.

5.2.2 Summarization

We evaluate our fine-tuned models for summarization on TL;DR test set (Stiennon et al., 2020) and present the results in Table 5.6. We report summarization quality using the standard ROUGE metric (Lin, 2004), by reporting the F1 scores for ROUGE-L and ROUGE-Lsum, METEOR (Banerjee and Lavie, 2005), BERTScore (Zhang et al., 2020) and Reward, a model-based metric that replicates human preferences (Song et al., 2023).

Due to the extensive resources required to train Gemma-2 models with non reference-free approaches on the summarization dataset, we excluded both SLiC from our evaluation.

Models	ROUGE-L	ROUGE-Lsum	METEOR	BERTScore	Reward
			TinyLlama-1.1B		
Tinyllama-1.1B	11.91	12.54	20.06	83.08	27.70
| + SFT	24.90	24.89	26.29	88.43	69.04
| + DPO	**24.99**	**24.99**	31.15	**88.45**	88.46
| + DPO-γ	24.64	24.65	**31.56**	88.38	**89.34**
| + SimPO	21.80	21.80	29.98	87.63	88.43
| + SLiC-DPO	24.61	24.62	31.40	88.36	89.32
| + CPO	21.82	21.83	25.74	87.76	64.92
| + SLiC-CPO	21.70	21.70	26.22	87.76	66.90
			Gemma-2-2B		
Gemma-2-2B	15.55	16.57	25.30	84.88	67.53
| + SFT	24.44	24.45	26.29	88.05	78.82
| + DPO	**25.99**	**25.99**	32.70	**88.78**	96.04
| + DPO-γ	25.80	25.81	**32.80**	88.76	**96.32**
| + SimPO	23.09	23.22	32.43	88.08	95.37
| + CPO	23.68	23.68	27.21	88.26	86.24
LLaMA-7B + PRO	33.63	–	–	–	89.71

Table 5.6: Summarization results on TL;DR dataset. Dark blue boxes indicates that the improvement over the SFT baseline is considerable. The lesser improvements are highlighted in shallow blue boxes. Decreases in performance are marked with red boxes.

Gemma-2 2B establishes a stronger baseline than TinyLlama-1.1B in the summarization task. When applied to summarization, the Gemma-2 2B model significantly outperforms TinyLlama-1.1B across all metrics, making it a more robust baseline. This performance gap is especially pronounced in the Result metric with Gemma-2 showing a remarkable improvement of ~240% over TinyLlama and highlighting its superior capability in generating high-quality summaries.

SFT enhances the quality of the model-generated summaries. Similar to its impact on machine translation, SFT proves to be essential in improving summarization performance beyond that of the base model. Both models exhibit enhancements across all metrics, particularly noticeable in TinyLlama's Reward score, which jumps from 27.70 to 69.04. This substantial increase reflects a significant improvement in the quality of the generated summaries.

CPO outperforms SFT on Gemma-2 but underperforms on TinyLlama. CPO significantly enhances the base model's performance, though results against SFT are mixed. Contrary to the trend observed in the machine translation task, CPO demonstrates a notable advantage over SFT on Gemma-2, particularly in the Reward metric. This suggests that CPO-generated summaries are more aligned with human preferences. Additionally, while METEOR score are also higher, ROUGE score is lower compared to SFT's. This suggests that CPO may prioritize semantically rich and human-aligned outputs that do not strictly adhere to the lexical structure of the reference summaries, thus explaining the lower ROUGE scores, which measure lexical similarity.

On TinyLlama, SFT consistently outperforms CPO and SLiC-CPO in every metric. The Reward metric, in particular, highlights SFT's superior alignment with human preferences. This performance gap becomes even more pronounced when SFT serves as the baseline for other preference methods. Overall, and in contrast to larger models (7B and 13B) (Xu et al., 2024b), CPO performs poorly compared to alternative PO algorithms in summarization tasks when applied to SLMs.

DPO-based methods and SimPO consistently outperform SFT and CPO models. As illustrated in Table 5.6, DPO, DPO-γ, SimPO, and SLiC-DPO demonstrate clear superiority over both the SFT model and CPO in terms of the Reward metric. Notably, **DPO stands out as the only method that consistently surpasses SFT across all metrics for both models**. DPO-γ also performs well, consistently outperforming SFT in all metrics for the Gemma model. On the other hand, SimPO shows a degradation in ROUGE scores compared to SFT and produces the least preferred summaries amongst the PO methods applied over SFT.

DPO, DPO-γ and SLiC-DPO demonstrate comparable performance. In terms of overall results, DPO consistently presents the best results for ROUGE and BERTScore, while DPO-γ outperforms others in METEOR and Reward metrics. Although SLiC-DPO shows similarly scores across all metrics, it never surpasses both DPO and DPO-γ. For TinyLlama, while DPO, DPO-γ and SLiC-DPO perform similarly across every metric, there is a significant difference in the Reward metric: DPO-γ emerges as the best-performing model, with SLiC-DPO trailing slightly, while DPO lags further behind. A similar trend is observed in Gemma-2, although the performance gap is less pronounced.

SLiC-DPO's distinction lies in its use of a different loss function, whereas the only difference between DPO and DPO-γ is the introduction of $\gamma > 0$ in the Bradley-Terry (BT) objective (Bradley and Terry, 1952). This modification ensures that the reward for the preferred response, $r(x, y_w)$, exceeds the reward for the dispreferred response, $r(x, y_l)$, by at least γ. We speculate that this adjustment enhances the model's alignment with human preferences in summarization, leading to improved generation of preferred summaries.

DPO, DPO-γ, and SimPO summaries in Gemma-2 are preferred over LLaMA-7B + PRO's. When applied over SFT, the implemented preference optimization methods show superior Reward metric compared to LLaMA-7B model fine-tuned with PRO (Song et al., 2023). However, there is a notable decline in ROUGE-L scores, suggesting that while the models excel in aligning with human preferences, they may struggle with maintaining certain text-level qualities like overlap with the reference summaries.

Influence of summary length on performance metrics. The performance of summarization models is crucially influenced by their ability to generate concise summaries that encapsulate essential information. Therefore, it is imperative to assess how the length of outputs from SLMs affects performance metrics. This aspect will be explored further in Section 5.4.

5.2.3 Valuable Results and Takeaways

Our comprehensive evaluation of preference optimization methods on SLMs has led to several key insights regarding the quality of outputs they produce. The primary conclusions drawn from our study are as follows:

TinyLlama is not suitable for MT. TinyLlama lacks the specialized fine-tuning required for effective MT tasks. To achieve high-quality translations, a model must be extensively fine-tuned on the target language data, capturing its unique grammatical structures, idiomatic expressions, and contextual nuances. Models such as EuroLLM (Martins et al., 2024), ALMA (Xu et al., 2024a), or Tower (Alves et al., 2024b) have undergone this fine-tuning, enabling them to handle the complexities of specific languages with greater accuracy and fluency.

SFT is Essential for SLMs. Across all models and tasks, SFT or previous instruction tuning consistently provides substantial improvements over the base models. This significant enhancement demonstrates that an initial supervised fine-tuning is a crucial step for aligning SLMs with the desired output formats and improving their performance.

DPO outperformes CPO. Contrary to findings in larger models, namely 7B models (Saeidi et al., 2024; Agrawal et al., 2024), DPO outperforms CPO in smaller models. Even when applied on top of SFT, CPO fails to match the performance of DPO methods. DPO shows the best improvements in performance considering the automated metrics taken into account for the tasks of machine translation and summarization. In contrast to findings with larger models (Xu et al., 2024b), simpler PO methods like CPO tend to underperform in SLMs. This trend indicates that CPO may not be suitable for preference optimization in smaller models without meticulous hyperparameter tuning.

Reference policy is needed to improve model outputs. In MT, SimPO consistently boosts the chrF metric across experiments, yet generally falls short in generating quality outputs compared to other preference optimization methods like DPO and SLiC-DPO. In some cases, there is a significant difference between the quality of the outputs generated by SimPO and DPO. This suggests indicates that the SLMs tend to produce better results when π_{ref} is integrated in the objective – i.e. when the reference fine-tuned model plays a part on the loss function. For summarization, SimPO underperforms against DPO-based models and shows degrading performance in some metrics against the SFT model.

PO in summarization significantly improves the quality of the summaries. In summarization, we find that preference optimization methods can significantly enhance metrics that correlate with human judgment, such as the Reward metric. For Gemma-2, applying DPO-γ improved the Reward metric to 96.32, surpassing even larger models like LLaMA-7B fine-tuned with PRO (Table 5.6). As for other metrics, improvements are visible compared to the SFT and base models for some RL-free approaches.

This indicates that preference optimization can indeed make SLMs competitive with larger models for certain tasks.

5.3 Alignment with Human Preferences

Building on insights from Chen et al. (2024b), we study the effectiveness of preference learning algorithms by assessing their *ranking accuracies* – a pivotal metric for evaluating how well these models align with human preferences. Chen et al. (2024a) underscore a significant alignment gap, with these algorithms frequently achieving less than 60% ranking accuracy on standard preference datasets. This finding motivates a thorough examination of the alignment of SLMs with human preferences.

We focus our study of human preferences primarily on machine translation, as constructing high-quality human preference datasets for other tasks, particularly summarization, remains costly and logistically challenging.

To evaluate the alignment of our MT models with human preferences, we utilize the data collected in Agrawal et al. (2024). The dataset contains 200 source instances, sampled from the WMT23 English-German (en→de) and Chinese-English (zh→en) test sets. It features generated translations for those source instances using five well-performing models and their evaluation by human native speakers, including both preferred and dispreferred responses, alongside their respective score. We call this dataset ***HumanPreferences-BW***. Figure 5.2 illustrates the distribution of the collected preferences, categorized based on the score difference between the chosen and rejected responses. References with a higher score difference represent clearer preferences, while smaller score differences indicate

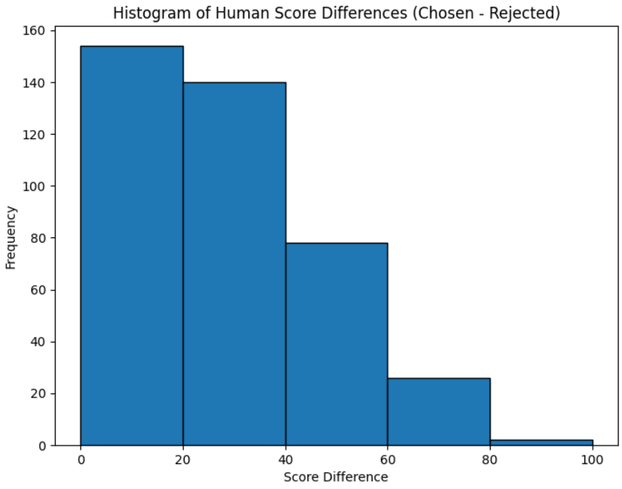

Figure 5.2: Preference scores differences distribution on HumanPreferences-BW. Higher scores differences represent clearer preferences.

less pronounced preferences between translations.

The dataset comprises numerous similar outputs, with only a few instances where one output is distinctly preferred over another. These rare cases of small preference score differences present challenges, as minor nuances that could make one translation slightly superior may be difficult for smaller models to discern. This limitation impacts the overall results, as the subtle distinctions might not adequately captured by the models.

5.3.1 Accuracy Scores

Our initial evaluation involves calculating all our MT models' accuracy in preferring the chosen translation over the rejected one. We analyze whether the models assign higher log probabilities to the outputs that match human preferences. Specifically, for each sample, we compare the log probabilities of the preferred and dispreferred translations, considering the model accurate if it assigns a higher log probability to the preferred translation. Table 5.7 presents the results obtained for all the models.

Model	TinyLlama	Gemma-2	EuroLLM
Base Model	0.53	0.56	0.6525
| + SFT	0.54	0.6275	0.615
| + DPO	0.5425	0.6325	0.6325
| + DPO-γ	0.5425	**0.6425**	0.64
| + SimPO	**0.5625**	0.6275	0.65
| + SLiC-DPO	0.545	0.6325	0.64
| + CPO	0.5325	0.6	0.6425
| + SLiC-CPO	0.53	0.61	0.64
| + DPO	–	–	0.6575
| + DPO-γ	–	–	0.6625
| + SimPO	–	–	**0.67**
| + SLiC-DPO	–	–	0.6625

Table 5.7: Model accuracy score comparison on HumanPreferences-BW. Dark blue boxes indicate a big improvement over the SFT baseline. Smaller improvements are highlighted in light blue boxes. Highest accuracy values for each model are in bold.

Base models' capabilities of learning preferences. Results show that SFT alone is able to increase the alignment with the human preference for TinyLlama and Gemma 2 models. For TinyLlama, CPO shows no significant improvement in preference accuracy, while SLiC-CPO shows slight improvement. Gemma-2 experiences improvements across SFT, CPO, and SLiC-CPO, with SFT making the most notable improvements. Conversely, for EuroLLM, SFT, CPO, and SLiC-CPO all reduce preference accuracy. Interestingly, when the instruction-tuned model is directly fine-tuned using preference optimization algorithms – bypassing SFT – there is an increase in all accuracies, with SimPO yielding the best outcomes.

SimPO achieves the highest accuracy in TinyLlama and EuroLLM. SimPO, when paired with SFT, achieves the best performance outcomes for TinyLlama. Importantly, as shown in Table 5.2, SimPO is not the model that generates higher quality responses. It decreases both COMET and XCOMET metrics compared to SFT in TinyLamma and loses to DPO-based models in COMET and XCOMET metrics for both TinyLlama and EuroLLM.

In contrast, SimPO alone excels in enhancing human preference alignment in EuroLLM without the need for prior SFT. However, its impact on Gemma-2 is neutral, neither improving nor reducing preference accuracy.

TinyLlama vs. Gemma-2 vs. EuroLLM. TinyLlama struggles with preference learning, possibly due to its smaller size and more unstable training for MT. In contrast, Gemma-2, with more than double the parameters of TinyLlama, shows enhanced ability to align with human preferences, particularly after the integration of SFT and other preference optimization techniques. Meanwhile, EuroLLM-1.7B-Instruct, although it does not show improvements with SFT, maintains high accuracy and effectively learns preferences when PO algorithms are directly applied instruction-tuned model.

5.3.2 Easy Preferences

Easy preferences are defined as scenarios where the model's probabilistic outputs align with human preferences. Specifically, these are instances where the domain-annotated preferred option is rated higher than the less preferred option both by the human evaluators and the base model.

This alignment indicates that the model inherently understands and replicates human-like decisions within specific contexts, which may diminish the need for additional tuning or complex optimization strategies. We assess how different models perform under these conditions, specifically focusing on their ability to consistently maintain these easy preferences across various PO strategies. We report the results obtained for easy preferences in Table 5.8.

SFT reduces the retention of original preferences by approximately 14%. Across the three models, applying SFT leads to a decrease in accuracy from 1.0 to approximately 0.86-0.88, indicating a loss of about 14% of the original preferences. For TinyLlama and Gemma-2, although some easy human preferences are lost, the overall quality of the text is significantly improved, as shown in Section 5.2. Additionally, for all three models, the average log probabilities for both the chosen and rejected options increase, suggesting that the models' confidence in their outputs has grown, even if their ability to distinguish between preferred and less preferred options has diminished.

CPO preserve the most easy preferences. Across the three models, CPO stands out as the most effective method in preserving easy preferences from the base models. CPO achieves an accuracy of 0.991 for TinyLlama and 0.963 for EuroLLM, the highest amongst all methods for TinyLlama. This indicates that it effectively maintains the original preferences.

Model	Accuracy	Avg. Chosen diff	Avg. Rejected diff	Avg. Model diff
		TinyLlama-1.1B		
Base Model	1.0 (212 samples)	–	–	–
| + SFT	0.863	+29.720	+29.702	+0.018
| + DPO	0.873	+27.061	+27.194	-0.132
| + DPO-γ	0.854	+28.768	+27.848	+0.920
| + SimPO	0.811	+21.416	+18.649	**+2.766**
| + SLiC-DPO	0.853	+28.843	+27.947	+0.897
| + CPO	**0.991**	+0.445	+0.429	+0.016
| + SLiC-CPO	0.929	+17.311	+17.163	+0.148
		Gemma-2 2B IT		
Base Model	1.0 (224 samples)	–	–	–
| + SFT	0.862	+41.322	+43.221	-1.899
| + DPO	0.830	+38.290	+37.949	+0.341
| + DPO-γ	0.830	+38.010	+37.535	+0.474
| + SimPO	0.763	+23.514	+18.0125	**+5.501**
| + SLiC-DPO	0.835	+39.250	+39.400	-0.150
| + CPO	0.884	+32.404	+32.888	-0.484
| + SLiC-CPO	**0.888**	+31.325	+31.723	-0.398
		EuroLLM-1.7B-Instruct		
Base Model	1.0 (261 samples)	–	–	–
| + SFT	0.877	+10.273	+10.569	-0.295
| + DPO	0.900	+10.446	+9.834	+0.612
| + DPO-γ	0.912	+10.474	+9.797	+0.676
| + SimPO	0.916	+10.458	+9.259	**+1.199**
| + SLiC-DPO	0.904	+0.107	-0.753	+0.861
| + CPO	**0.963**	+8.661	+8.775	-0.114
| + SLiC-CPO	0.961	+8.379	+8.580	-0.201
| + DPO	0.973	-2.306	-3.661	+1.356
| + DPO-γ	0.973	-2.524	-3.955	+1.431
| + SimPO	0.939	-20.072	-21.857	**+1.785**
| + SLiC-DPO	**0.977**	-2.389	-3.901	+1.512

Table 5.8: Easy Preferences Accuracy and details. Higher values for accuracy and average model difference are in bold.

Effects of preference algorithms applied on top of SFT. Applying preference optimization algorithms on top of SFT yields varied effects on the retention of easy preferences. For TinyLlama, DPO is the only preference method that improve accuracy, partially recovering the preferences lost during SFT. Conversely, SimPO further decreases SFT's accuracy to 0.811 for TinyLlama, indicating a greater loss of preferences. With Gemma-2, PO algorithms consistently reduce accuracy relative to SFT. In contrast, EuroLLM benefits from the direct application of preference algorithms over the instruction-tuned base model, showing improvements in the retention of easy preferences.

EuroLLM does not benefit from SFT in terms of retaining easy preferences. The results indicate that applying preference optimization directly to the EuroLLM-1.7B-Instruct retains the most easy preferences. DPO-based methods achieve accuracies above 97%, therefore losing less than 3% of the initial ”correct” preferences. This suggests that SFT may not be necessary or beneficial for EuroLLM

Model	Accuracy	Avg. Chosen diff	Avg. Rejected diff	Avg. Model diff
		TinyLlama-1.1B + SFT (216 samples)		
+ DPO	0.935	-2.672	-2.011	-0.661
+ DPO-γ	0.949	-0.689	-1.929	+1.240
+ SimPO	0.894	-7.576	-11.381	**+3.805**
+ SLiC-DPO	**0.954**	-0.620	-1.832	+1.211
		Gemma-2 2B IT + SFT (251 samples)		
+ DPO	0.928	-2.595	-5.680	+3.085
+ DPO-γ	0.932	-2.840	-6.108	+3.268
+ SimPO	0.832	-17.273	-26.727	**+9.453**
+ SLiC-DPO	**0.952**	-1.703	-4.141	+2.438

Table 5.9: Easy Preferences accuracies compared to the SFT baseline.

in the context of maintaining easy preferences.

SimPO exhibits lower accuracy but increases model differentiation. While SimPO is the method that struggles more to maintain the easy preferences from the base model – with accuracies dropping to 0.811 for TinyLlama and 0.763 for Gemma-2, when comparing against the base model (Table 5.8), and to 0.894 and 0.832 when compared with SFT retained preferences (Table 5.9). Surprisingly, SimPO significantly increases the average model score difference, enhancing the model's ability to distinguish between chosen and rejected responses. Despite the loss in preference accuracy, SimPO consistently improves the model's capacity to separate the preferred from the less preferred options. When SimPO is directly applied to EuroLLM-1.7B-Instruct, it causes the greatest decrease in log probabilities for both prefered and disprefered translations. These value are almost 9× larger than those observed with DPO, DPO-γ, and SLiC-DPO.

DPO-γ balances easy preferences retention and model differentiation. DPO-γ effectively maintains high levels of preference retention while enhancing the average model difference, which measures the model's ability to distinguish between preferred and less preferred responses. When applied after SFT, DPO-γ achieves comparable performance to DPO in retaining easy preferences but increases the log probability difference between the preferred and dispreferred responses. The enhancement in model differentiation can be attributed to the introduction of the parameter $\gamma > 0$, which ensures that the reward for the preferred response exceeds that of the less preferred response by at least γ. This parameter strengthens the model's discriminative abilities, leading to clearer separation between preferred and non-preferred outputs.

DPO reduces SFT correct preferences in TinyLlama. When DPO is applied post-SFT on TinyLlama, Table 5.9 show that it leads to lowest accuracy of 0.935, lower than that achieved with DPO-γ (0.949) and SLiC-DPO (0.954). Additionally, the average model difference decreases 0.661, suggesting that the model becomes less effective at distinguishing between chosen and rejected responses. This indicates that DPO may lead to the unlearning of certain preferences acquired during SFT.

5.3.3 Hard Preferences

Hard preferences refer to scenarios where the preferences explicitly contradict the model's predictions. Specifically, these are cases where the translation annotated as 'chosen' by human labelers is preferred over another, yet the model inversely ranks these choices, preferring the 'rejected' translation over the 'chosen' one. This assessment helps to understand how well models can align with human judgment, especially in cases where the model's probabilistic outcomes are directly opposed to established human preferences.

We report the results of different models against a baseline in Table 5.10. Similarly to our approach for *easy* preferences, we also report the results against the SFT model when the PO techniques are applied over it. These results are shown in Table 5.11.

Model Enhancements	**TinyLlama-1.1B** (188 samples)	**Gemma-2 2B IT** (176 samples)	**EuroLLM-1.7B-Instruct** (139 samples)
Base Model	0.0	0.0	0.0
| + SFT	0.176	0.330	0.122
| + DPO	0.170	0.381	0.129
| + DPO-γ	0.191	0.403	0.129
| + SimPO	**0.282**	**0.449**	**0.151**
| + SLiC-DPO	0.197	0.375	0.144
| + CPO	0.016	0.381	0.043
| + SLiC-CPO	0.080	0.256	0.036
| + DPO	–	–	0.065
| + DPO-γ	–	–	0.079
| + SimPO	–	–	**0.165**
| + SLiC-DPO	–	–	0.072

Table 5.10: Hard Preferences Accuracy and details. Higher values for accuracy and average model difference are in bold.

PO algorithms rarely flip preference rankings. Building on the findings of Chen et al. (2024b) - which demonstrate that DPO rarely adjusts preference rankings - we extend this observation to include a wider array of PO methods applied to both smaller and similarly sized models. Our empirical analysis indicates that these algorithms often fail to rectify even minor ranking discrepancies in the outputs of reference models. This deficiency complicates the task of precisely aligning smaller language models with detailed human preferences without substantial adjustments or advancements in training approaches.

SimPO corrects the most *hard* preferences. SimPO outperforms all the other PO methods applied on the same baseline (SFT model) across all three backbone models. It is important to consider that SFT might have shifted the model's probabilities in a way that some previously incorrect preferences become correct and therefore SimPO might be correcting preferences that were ”unlearned” during the SFT phase. While SimPO shows strong performance in addressing *hard* preferences, it also tends to unlearn many *easy* preferences, as detailed in Table 5.8. Overall, as depicted in Table 5.7,

Model Enhancements	**TinyLlama-1.1B** (184 samples)	**Gemma-2 2B IT** (149 samples)	**EuroLLM-1.7B-Instruct** (154 samples)
SFT	0.0	0.0	0.0
| + DPO	0.082	0.134	0.071
| + DPO-γ	0.065	0.154	0.097
| + SimPO	**0.174**	**0.275**	**0.117**
| + SLiC-DPO	0.065	0.094	0.084
DPO	–	–	0.227
DPO-γ	–	–	0.240
SimPO	–	–	**0.279**
SLiC-DPO	–	–	0.234

Table 5.11: Hard Preferences accuracies compared to the SFT baseline.

SimPO achieves the highest accuracy scores in both the TinyLlama and Gemma-2 models. Specially for Gemma-2, it corrects up to 45% for the *hard* preferences. Finally, for EuroLLM-1.7B-Instruct, applying SimPO directly on top of the instruction-tuned model yields the best correction of *hard* preferences, though it is less effective at retaining *easy* preferences.

DPO-γ outperforms DPO and SLiC-DPO learning *hard* preferences. There is a consistent ~2% increase in learning *hard* preferences using the DPO variant incorporating γ. This improvement persists even when directly fine-tuning the EuroLLM-1.7B-Instruct with preference optimization methods.

5.3.4 Valuable Results and Takeways

The results presented in this section highlight several key findings that shed light on the alignment of SLMs with human preferences in MT:

SimPO consistently aligns translations with human preferences. While SimPO struggles to retain easy preferences from the base models, it excels in correcting hard preferences, resulting in strong overall accuracy. This is particularly evident in the TinyLlama and Gemma models, where SimPO consistently outperforms other methods in addressing preference misalignments.

DPO-γ enhances DPO capability of learning human preferences by introducing a reward margin. The introduction of a $\gamma > 0$ in DPO-γ improves its ability to align translations with human preferences. This margin ensures a clear distinction between preferred and non-preferred outputs, leading to better overall results compared to standard DPO, as it helps the model distancing winning and losing translations more effectively.

CPO excels at preserving easy preferences but rarely flips incorrect preferences. CPO is the most effective method for retaining easy preferences from the base models across TinyLlama, Gemma-2, and EuroLLM. However, it struggles to correct *hard* preferences. This is most visible in TinyLlama and EuroLLM, where corrections rates only range from 1-8%.

Gemma-2 demonstrates the most significant improvement in aligning with human prefer-

ences. As shown in Table 5.8, Gemma-2's accuracy in preferring the chosen translation over the rejected one increases from 56% to 64.25% when trained with SFT combined with DPO-γ.

5.4 Length Bias

Length exploitation of LLMs is significant issue, leading to studies to improve LLMs performance while reducing this bias (Dubois et al., 2024; Park et al., 2024). Not only generative models but benchmarks to estimate response quality suffer from this bias. Despite being highly correlated with human preferences, evaluators tend to favor models that generate longer outputs (Huang et al., 2024b; Thakur et al., 2024).

Addressing length bias is essential for applications such as summarization, where the balance between brevity and content coverage is necessary. Specifically in summarization, this bias refers to the model's tendency to generate unnecessarily lengthy summaries, which may not enhance the quality or relevance of the content but dilute important details with redundancy.

Building on previous studies conducted with larger models, we examine the impact on smaller, more recent models that are less than half the size, specifically for the task of summarization. In this section we present the results for our SLMs when taking into account the length bias.

5.4.1 Length Comparison

Table 5.12 presents the results for a comparative analysis of the output summaries generated by each model on the TL;DR test set. For TinyLlama, we extend this analysis to the the OpenAI TL;DR Human Feedback (Summarize) test set, presenting the results Figure 5.3.

	TinyLlama-1.1B			**Gemma-2-2B**		
Model	**Mean**	**Median**	**Std.**	**Mean**	**Median**	**Std.**
Base Model	**85.3**	87.0	13.5	**79.9**	92.0	29.8
| + SFT	24.2	23.0	9.4	23.2	23.0	7.3
| + DPO	36.1	34.0	12.7	35.1	34.0	9.4
| + DPO-γ	38.2	35.0	13.6	35.9	35.0	9.6
| + SimPO	**42.7**	42.0	10.0	**44.7**	43.0	13.4
| + SLiC-DPO	37.7	35.0	13.2	–	–	–
| + CPO	34.0	29.0	20.1	28.9	26.0	13.6
| + SLiC-CPO	35.3	30.0	19.9	–	–	–
Reference Summaries	27.2	26.0	5.9	27.2	26.0	5.9

Table 5.12: Comparative output summaries statistics (Mean, Median, Standard Deviation) for the TinyLlama-1.1B and Gemma-2-2B models on TL;DR.

SFT shows the shortest summaries in average. While base models are overly verbose, when SFT is applied the average size of the summaries produced significantly decreases, for both models. It not only comes close to the reference summaries, but also learn to reduce its size even further. In fact, Figure 5.3 shows that the summaries produced by the SFT model, prior to PO, are generally shorter than both chosen and rejected summaries.

CPO produces the shortest summaries on average among the PO methods but shows the highest variance. Among the preference optimization methods, CPO produces the least verbose summaries. However, it also exhibits the greatest variance, both on TL;DR and Summarize, as reported in Figure 5.3. Like SFT, CPO, shortens the average length of output summaries but also outputs the most summaries exaggeratedly lengthy. The use of the negative-likelihood loss, similar to what is observed with SFT, is responsible for shortening the summaries and shifting the length distribution towards lower values.

DPO-based methods increase the length of summaries. While improving the quality of the generated summaries, DPO-based methods also result in an increase in the average length of the summaries compared to the SFT model and the reference summaries. Across all models, there is a noticeable rise in the number of summaries exceeding 30 tokens. **DPO, which outperforms SFT across all automated metrics, shows the smallest increase in summary length**, demonstrating the lowest increase in average length, the smallest median, and the least variance across every PO model and datasets.

DPO-γ produces longer summaries compared to DPO. While the introduction of the γ parameter improves DPO-γ's performance on the Reward metric, it also results in generating longer summaries across both datasets. Both average length, median and std increase in TL;DR. In Summarize, although preserve the position the count density peak in Figure 5.3, the average length rises, and the model shows an increased density of summaries in the 40 to 60 token range.

SimPO generates the longest summaries. While for larger models SimPO is presented as a DPO alternative that presents minimal length exploitation (Meng et al., 2024), for smaller models it is responsible for producing, in average, the most verbose summaries apart from those produced by the base models. Its median summary length is the largest, with summaries nearly twice as long as those from the SFT model. Particularly, there is a shift in the distribution of lengths towards higher values, with the density peak occurring at a greater length and a wider spread of lengths overall. Few summaries match the length of dataset's chosen summaries.

Correlation between length and quality of the summary. The length of summaries correlates significantly with the Reward metric, which serves as a surrogate for human preferences measured by a Reward Model (RM). This shows a significant bias towards longer summaries, which are often scored higher, indicating that length can influence perceived quality.

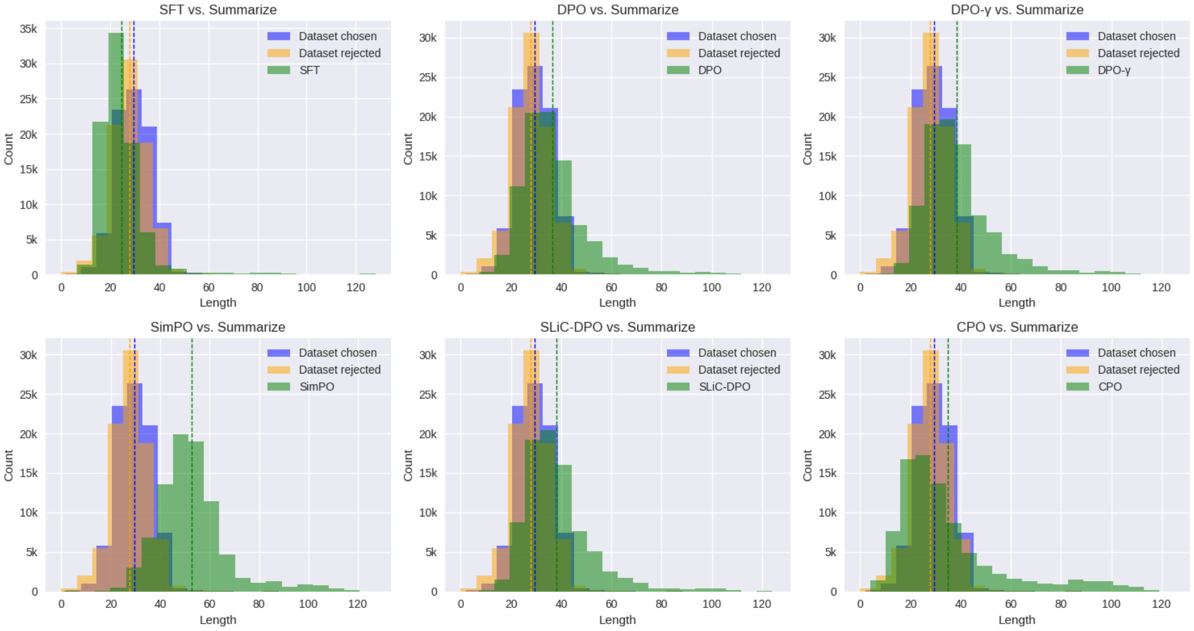

Figure 5.3: Distribution of response lengths for TinyLlama model in Summarize dataset. Average lengths are marked by the dashed line.

5.4.2 Length vs. Reward Bias

LLMs are often influenced by the verbosity of the outputs, favoring longer and more verbose texts, even if they less succinct or of lower quality. Saito et al. (2023) highlights discrepancy between answers of LLMs and those of humans. When compared to human preferences, LLMs show a significant verbosity bias by favoring longer responses.

With that in mind, the results obtained during the training process, and both performance and length results, we present a comprehensive analysis of the impact of length in the Reward metric utilized. Figure 5.4 shows the results obtained.

DPO-based methods lengthy summaries tend to have high Reward. In DPO-based models, there is a noticeable trend where longer summaries generally receive higher rewards. This observation is consistent across all DPO variations, with reward distribution showing higher density closer to 100 as the sequence length increases – Figure 5.4. Specifically, in the DPO model, rewards tend to cluster closer to the top score with increasing sequence lengths, highlighting a bias towards longer outputs in the reward structure.

CPO generate longer and low reward outputs. The CPO model reflects a spread reward distribution similar to that of SFT, with emphasis on the upper and lower bounds of the Reward metric. CPO displays significant variability in rewards, especially with longer sequences where outcomes can have either very low or very high reward values. Such wide-ranging results highlight that CPO is particularly sensitive to specific parameters for the summarization task when applied to SLMs. This variability might

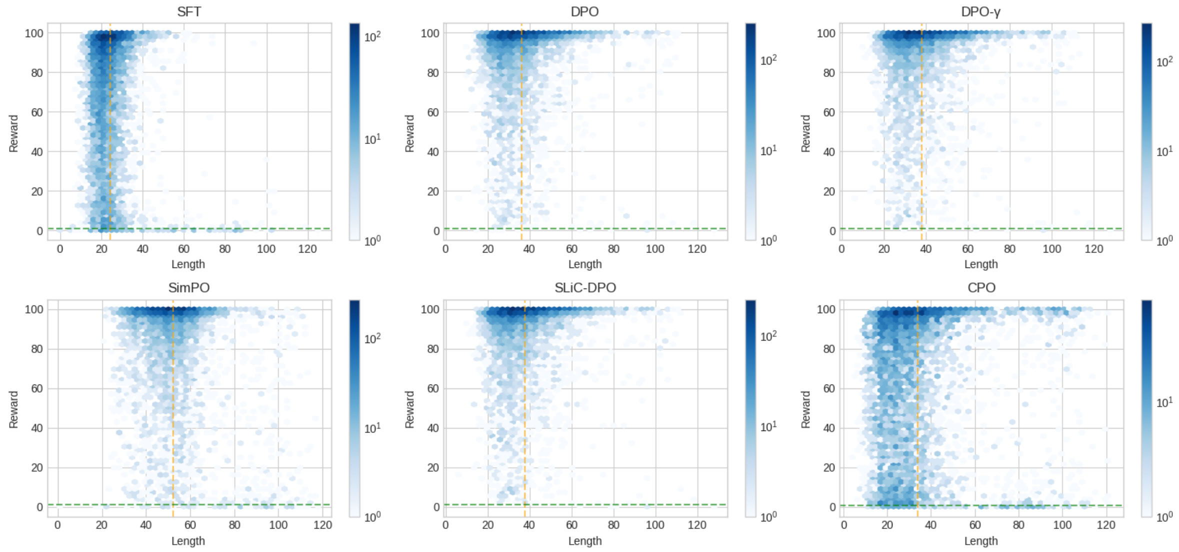

Figure 5.4: Heatmap of lengths of summarization models outputs vs. reward model scores given by Deberta-v3-large RM.

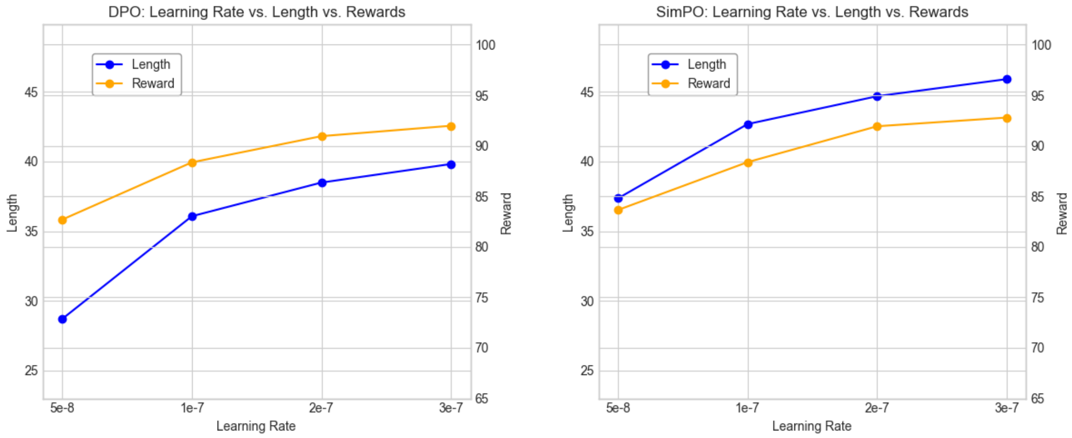

Figure 5.5: DPO training learning rate influence on length and Reward.

cause the generation of redundant filler content, potentially degrading the quality or producing inaccurate outputs.

SimPO lengthy summaries are not necessarily good. Compared to SFT, SimPO generates summaries with higher reward, clustering the density of the rewards more towards zones with higher rewards. While SimPO tends to generate the longest summaries than SFT and other PO methods, these summaries do not usually match human preferences. In Figure 5.4 we observe that for longer outputs, there is not a necessary correlation between length and reward. This suggests potential length hacking (Zhang et al., 2024b), where the model exhibit a bias towards longer responses, even when their quality is comparable.

To further explore the performance of SimPO in comparison to models based on DPO, and considering the valuable insights gathered during the training phase about the sensitivity of these SLMs to specific training parameters, we also include the outcomes achieved when generating from the DPO and SimPO models. These models were trained with varying learning rates, while all other parameters remained constant. The results are presented in Figure 5.5.

Training learning rate influence the quality and the length of the summaries. Both models show a direct correlation between learning rate, summary length and reward. With the increase of learning rate during training, models tend to generate longer but more rewarded summaries. As reward is influenced by the length of the summaries, this poses a problem when working with SLMs as their hyperameter fine-tuning must take into account tasks specific criteria. We show that for summarization, it is needed to find a balance between longer and generally better summaries or shorter and less quality summaries.

DPO's summaries quality are very sensitive to training learning rate. With lower learning rates, DPO tend to produce significantly shorter but equally preferred responses according to Reward.

SimPO produces longer but similarly preferred responses compared to DPO across all learning rates. While SimPO summaries reach lengths like 50 tokens, their preference improvements against DPO remain minimum.

6

Conclusion

Contents

This chapter presents the concluding remarks and outlines future directions for our thesis.

6.1 Conclusions

This thesis provides **the first comprehensive comparison of feedback methods on state-of-the-art small language models (SLMs)**. It effectively conducts a detailed study of RL-free approaches on small LLMs, focusing on machine translation, and summarization tasks. This investigation includes the application of SFT and Preference Training using well-known Preference Optimization methods such as DPO, CPO, SimPO, and SLiC within the context of small language models.

A thorough evaluation was conducted, comparing the performance of SLMs against both larger models and similarly sized baseline models without additional training. This involved a meticulous analysis based on multiple performance metrics, providing a detailed comparison of each model's capabilities.

This research evaluates the alignment of these RL-free methods with human preferences, investigating their adherence to user expectations in the application to SLMs. It addresses the challenges and potential biases that smaller models encounter due to their limited size and constrained learning capacity, offering critical insights into their deployment and utility in real-world scenarios.

Collectively, the findings from this thesis contribute to a better understanding of how SLMs can be effectively optimized to improve output quality and align more closely with human preferences, thus bolstering their utility and dependability for a variety of applications.

6.2 Limitations and Future Work

This thesis establishes a foundational understanding of feedback methods in SLMs, yet we recognize opportunities for further enhancement. A significant limitation was the **absence of evaluations conducted by human experts or automatic evaluation systems that align closely with human preferences**, such as GPT-4. Incorporating such assessments could more robustly validate the human alignment effectiveness of these models.

Additionally, the study's focus on **summarization tasks was limited by the high costs and extensive resources needed**. The models demanded considerable time to train because of the large datasets involved. There was also a scarcity of datasets providing high-quality, human-labeled data for summarization, and obtaining such data, similar to what was required for machine translation, would incur significant expenses.

Future efforts should continue to build on the solid foundation established by this thesis to further enhance SLMs optimization. Tailoring hyperparameter settings to align with specific PO approaches for a specific task could significantly enhance both the quality of models and their alignment with human preferences. Additionally, conducting **significance testing** would be beneficial to verify the reliability of the results.

Another avenue for future work involves **performing manual evaluations on the outputs** of all the SLMs. Although this thesis lacks manual evaluation due to the substantial costs associated with conducting extensive manual reviews at scale, it compensates by offering automatic evaluation metrics. These metrics help to illuminate certain strengths and weaknesses of the models compared within this work.

The investigation was also confined to a select group of reinforcement learning-free methods: DPO, SimPO, CPO, and SLiC. **Expanding the range of methods explored** could potentially reveal additional approaches that enhance model performance even more significantly.

Bibliography

S. Agrawal, J. G. C. de Souza, R. Rei, A. Farinhas, G. Faria, P. Fernandes, N. M. Guerreiro, and A. Martins. Modeling user preferences with automatic metrics: Creating a high-quality preference dataset for machine translation, 2024. URL https://arxiv.org/abs/2410.07779.

L. AI. Lit-gpt, 2023. URL https://github.com/Lightning-AI/lit-gpt.

AI@Meta. Llama 3 model card. 2024. URL https://github.com/meta-llama/llama3/blob/main/MODEL_CARD.md.

F. Akhbardeh, A. Arkhangorodsky, M. Biesialska, O. Bojar, R. Chatterjee, V. Chaudhary, M. R. Costa-jussa, C. España-Bonet, A. Fan, C. Federmann, M. Freitag, Y. Graham, R. Grundkiewicz, B. Haddow, L. Harter, K. Heafield, C. Homan, M. Huck, K. Amponsah-Kaakyire, J. Kasai, D. Khashabi, K. Knight, T. Kocmi, P. Koehn, N. Lourie, C. Monz, M. Morishita, M. Nagata, A. Nagesh, T. Nakazawa, M. Negri, S. Pal, A. A. Tapo, M. Turchi, V. Vydrin, and M. Zampieri. Findings of the 2021 conference on machine translation (WMT21). In L. Barrault, O. Bojar, F. Bougares, R. Chatterjee, M. R. Costa-jussa, C. Federmann, M. Fishel, A. Fraser, M. Freitag, Y. Graham, R. Grundkiewicz, P. Guzman, B. Haddow, M. Huck, A. J. Yepes, P. Koehn, T. Kocmi, A. Martins, M. Morishita, and C. Monz, editors, *Proceedings of the Sixth Conference on Machine Translation*, pages 1–88, Online, Nov. 2021. Association for Computational Linguistics. URL https://aclanthology.org/2021.wmt-1.1.

D. M. Alves, J. Pombal, N. M. Guerreiro, P. H. Martins, J. Alves, A. Farajian, B. Peters, R. Rei, P. Fernandes, S. Agrawal, P. Colombo, J. G. C. de Souza, and A. F. T. Martins. Tower: An open multilingual large language model for translation-related tasks, 2024a. URL https://arxiv.org/abs/2402.17733.

D. M. Alves, J. Pombal, N. M. Guerreiro, P. H. Martins, J. Alves, A. Farajian, B. Peters, R. Rei, P. Fernandes, S. Agrawal, P. Colombo, J. G. C. de Souza, and A. F. T. Martins. Tower: An open multilingual large language model for translation-related tasks, 2024b.

A. Anastasopoulos, A. Cattelan, Z.-Y. Dou, M. Federico, C. Federmann, D. Genzel, F. Guzmán, J. Hu, M. Hughes, P. Koehn, R. Lazar, W. Lewis, G. Neubig, M. Niu, A. Öktem, E. Paquin, G. Tang, and S. Tur. TICO-19: the translation initiative for COvid-19. In K. Verspoor, K. B. Cohen, M. Conway, B. de Bruijn,

M. Dredze, R. Mihalcea, and B. Wallace, editors, *Proceedings of the 1st Workshop on NLP for COVID-19 (Part 2) at EMNLP 2020*, Online, Dec. 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.nlpcovid19-2.5. URL https://aclanthology.org/2020.nlpcovid19-2.5.

R. Anil, A. M. Dai, O. Firat, M. Johnson, D. Lepikhin, A. Passos, S. Shakeri, E. Taropa, P. Bailey, Z. Chen, et al. Palm 2 technical report. *arXiv preprint arXiv:2305.10403*, 2023.

Y. Bai, A. Jones, K. Ndousse, A. Askell, A. Chen, N. DasSarma, D. Drain, S. Fort, D. Ganguli, T. Henighan, N. Joseph, S. Kadavath, J. Kernion, T. Conerly, S. El-Showk, N. Elhage, Z. Hatfield-Dodds, D. Hernandez, T. Hume, S. Johnston, S. Kravec, L. Lovitt, N. Nanda, C. Olsson, D. Amodei, T. Brown, J. Clark, S. McCandlish, C. Olah, B. Mann, and J. Kaplan. Training a helpful and harmless assistant with reinforcement learning from human feedback, 2022a. URL https://arxiv.org/abs/2204.05862.

Y. Bai, A. Jones, K. Ndousse, A. Askell, A. Chen, N. DasSarma, D. Drain, S. Fort, D. Ganguli, T. Henighan, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022b.

Y. Bai, S. Kadavath, S. Kundu, A. Askell, J. Kernion, A. Jones, A. Chen, A. Goldie, A. Mirhoseini, C. McKinnon, C. Chen, C. Olsson, C. Olah, D. Hernandez, D. Drain, D. Ganguli, D. Li, E. Tran-Johnson, E. Perez, J. Kerr, J. Mueller, J. Ladish, J. Landau, K. Ndousse, K. Lukosuite, L. Lovitt, M. Sellitto, N. Elhage, N. Schiefer, N. Mercado, N. DasSarma, R. Lasenby, R. Larson, S. Ringer, S. Johnston, S. Kravec, S. E. Showk, S. Fort, T. Lanham, T. Telleen-Lawton, T. Conerly, T. Henighan, T. Hume, S. R. Bowman, Z. Hatfield-Dodds, B. Mann, D. Amodei, N. Joseph, S. McCandlish, T. Brown, and J. Kaplan. Constitutional ai: Harmlessness from ai feedback, 2022c. URL https://arxiv.org/abs/2212.08073.

S. Banerjee and A. Lavie. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In J. Goldstein, A. Lavie, C.-Y. Lin, and C. Voss, editors, *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan, June 2005. Association for Computational Linguistics. URL https://aclanthology.org/W05-0909.

R. Bommasani, D. A. Hudson, E. Adeli, R. Altman, S. Arora, S. von Arx, M. S. Bernstein, J. Bohg, A. Bosselut, E. Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

R. A. Bradley and M. E. Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952.

T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

S. Casper, X. Davies, C. Shi, T. K. Gilbert, J. Scheurer, J. Rando, R. Freedman, T. Korbak, D. Lindner, P. Freire, et al. Open problems and fundamental limitations of reinforcement learning from human feedback. *arXiv preprint arXiv:2307.15217*, 2023.

A. T. Chaganty, S. Mussman, and P. Liang. The price of debiasing automatic metrics in natural language evaluation. *arXiv preprint arXiv:1807.02202*, 2018.

A. Chen, S. Malladi, L. H. Zhang, X. Chen, Q. Zhang, R. Ranganath, and K. Cho. Preference learning algorithms do not learn preference rankings, 2024a. URL https://arxiv.org/abs/2405.19534.

A. Chen, S. Malladi, L. H. Zhang, X. Chen, Q. Zhang, R. Ranganath, and K. Cho. Preference learning algorithms do not learn preference rankings, 2024b. URL https://arxiv.org/abs/2405.19534.

A. Chowdhery, S. Narang, J. Devlin, M. Bosma, G. Mishra, A. Roberts, P. Barham, H. W. Chung, C. Sutton, S. Gehrmann, et al. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113, 2023.

P. F. Christiano, J. Leike, T. B. Brown, M. Martic, S. Legg, and D. Amodei. Deep reinforcement learning from human preferences. In I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, and R. Garnett, editors, *NIPS*, pages 4299–4307, 2017. URL http://dblp.uni-trier.de/db/conf/nips/nips2017.html#ChristianoLBMLA17.

H. W. Chung, L. Hou, S. Longpre, B. Zoph, Y. Tay, W. Fedus, Y. Li, X. Wang, M. Dehghani, S. Brahma, et al. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*, 2022.

T. Dao. Flashattention-2: Faster attention with better parallelism and work partitioning. *arXiv preprint arXiv:2307.08691*, 2023.

J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

H. Dong, W. Xiong, D. Goyal, R. Pan, S. Diao, J. Zhang, K. Shum, and T. Zhang. Raft: Reward ranked finetuning for generative foundation model alignment. *arXiv preprint arXiv:2304.06767*, 2023.

Y. Dubois, B. Galambosi, P. Liang, and T. B. Hashimoto. Length-controlled alpacaeval: A simple way to debias automatic evaluators, 2024. URL https://arxiv.org/abs/2404.04475.

P. Fernandes, A. Madaan, E. Liu, A. Farinhas, P. H. Martins, A. Bertsch, J. G. de Souza, S. Zhou, T. Wu, G. Neubig, et al. Bridging the gap: A survey on integrating (human) feedback for natural language generation. *arXiv preprint arXiv:2305.00955*, 2023.

M. Freitag, D. Grangier, and I. Caswell. BLEU might be guilty but references are not innocent. In B. Webber, T. Cohn, Y. He, and Y. Liu, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 61–71, Online, Nov. 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.5. URL https://aclanthology.org/2020.emnlp-main.5.

Y. Fu, H. Peng, L. Ou, A. Sabharwal, and T. Khot. Specializing smaller language models towards multi-step reasoning, 2023. URL https://arxiv.org/abs/2301.12726.

G. R. Ghosal, M. Zurek, D. S. Brown, and A. D. Dragan. The effect of modeling human rationality level on learning rewards from multiple feedback types. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 5983–5992, 2023.

A. Glaese, N. McAleese, M. Trebacz, J. Aslanides, V. Firoiu, T. Ewalds, M. Rauh, L. Weidinger, M. Chadwick, P. Thacker, et al. Improving alignment of dialogue agents via targeted human judgements. *arXiv preprint arXiv:2209.14375*, 2022.

N. Goyal, C. Gao, V. Chaudhary, P.-J. Chen, G. Wenzek, D. Ju, S. Krishnan, M. Ranzato, F. Guzmán, and A. Fan. The flores-101 evaluation benchmark for low-resource and multilingual machine translation. 2021.

N. M. Guerreiro, R. Rei, D. van Stigt, L. Coheur, P. Colombo, and A. F. T. Martins. xcomet: Transparent machine translation evaluation through fine-grained error detection, 2023. URL https://arxiv.org/abs/2310.10482.

B. Hancock, A. Bordes, P.-E. Mazare, and J. Weston. Learning from dialogue after deployment: Feed yourself, chatbot! In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3667–3684, 2019.

J. Hejna, R. Rafailov, H. Sikchi, C. Finn, S. Niekum, W. B. Knox, and D. Sadigh. Contrastive preference learning: Learning from human feedback without rl, 2024. URL https://arxiv.org/abs/2310.13639.

K. M. Hermann, T. Kočiský, E. Grefenstette, L. Espeholt, W. Kay, M. Suleyman, and P. Blunsom. Teaching machines to read and comprehend, 2015. URL https://arxiv.org/abs/1506.03340.

G. Hinton, O. Vinyals, and J. Dean. Distilling the knowledge in a neural network, 2015. URL https://arxiv.org/abs/1503.02531.

K.-H. Huang, P. Laban, A. R. Fabbri, P. K. Choubey, S. Joty, C. Xiong, and C.-S. Wu. Embrace divergence for richer insights: A multi-document summarization benchmark and a case study on summarizing diverse information from news articles, 2024a. URL https://arxiv.org/abs/2309.09369.

Z. Huang, Z. Qiu, Z. Wang, E. M. Ponti, and I. Titov. Post-hoc reward calibration: A case study on length bias, 2024b. URL https://arxiv.org/abs/2409.17407.

A. Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. S. Chaplot, D. d. l. Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.

T. Kocmi, C. Federmann, R. Grundkiewicz, M. Junczys-Dowmunt, H. Matsushita, and A. Menezes. To ship or not to ship: An extensive evaluation of automatic metrics for machine translation. In L. Barrault, O. Bojar, F. Bougares, R. Chatterjee, M. R. Costa-jussa, C. Federmann, M. Fishel, A. Fraser, M. Freitag, Y. Graham, R. Grundkiewicz, P. Guzman, B. Haddow, M. Huck, A. Jimeno Yepes, P. Koehn, T. Kocmi, A. Martins, M. Morishita, and C. Monz, editors, *Proceedings of the Sixth Conference on Machine Translation*, pages 478–494, Online, Nov. 2021. Association for Computational Linguistics. URL https://aclanthology.org/2021.wmt-1.57.

T. Kocmi, R. Bawden, O. Bojar, A. Dvorkovich, C. Federmann, M. Fishel, T. Gowda, Y. Graham, R. Grundkiewicz, B. Haddow, R. Knowles, P. Koehn, C. Monz, M. Morishita, M. Nagata, T. Nakazawa, M. Novák, M. Popel, and M. Popović. Findings of the 2022 conference on machine translation (WMT22). In P. Koehn, L. Barrault, O. Bojar, F. Bougares, R. Chatterjee, M. R. Costa-jussà, C. Federmann, M. Fishel, A. Fraser, M. Freitag, Y. Graham, R. Grundkiewicz, P. Guzman, B. Haddow, M. Huck, A. Jimeno Yepes, T. Kocmi, A. Martins, M. Morishita, C. Monz, M. Nagata, T. Nakazawa, M. Negri, A. Névéol, M. Neves, M. Popel, M. Turchi, and M. Zampieri, editors, *Proceedings of the Seventh Conference on Machine Translation (WMT)*, pages 1–45, Abu Dhabi, United Arab Emirates (Hybrid), Dec. 2022. Association for Computational Linguistics. URL https://aclanthology.org/2022.wmt-1.1.

T. Kocmi, E. Avramidis, R. Bawden, O. Bojar, A. Dvorkovich, C. Federmann, M. Fishel, M. Freitag, T. Gowda, R. Grundkiewicz, B. Haddow, P. Koehn, B. Marie, C. Monz, M. Morishita, K. Murray, M. Nagata, T. Nakazawa, M. Popel, M. Popović, and M. Shmatova. Findings of the 2023 conference on machine translation (wmt23): Llms are here but not quite there yet. In P. Koehn, B. Haddow, T. Kocmi, and C. Monz, editors, *Proceedings of the Eighth Conference on Machine Translation*, pages 1–42, Singapore, dec 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.wmt-1.1. URL https://aclanthology.org/2023.wmt-1.1.

J. Kreutzer, S. Khadivi, E. Matusov, and S. Riezler. Can neural machine translation be improved with user feedback? *arXiv preprint arXiv:1804.05958*, 2018.

W. Kwon, Z. Li, S. Zhuang, Y. Sheng, L. Zheng, C. H. Yu, J. E. Gonzalez, H. Zhang, and I. Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*, 2023.

Y. B. Leandro von Werra, L. Tunstall, E. Beeching, T. Thrush, N. Lambert, and S. Huang. Trl: Transformer reinforcement learning. 2020.

J. Li, A. H. Miller, S. Chopra, M. Ranzato, and J. Weston. Dialogue learning with human-in-the-loop. *arXiv preprint arXiv:1611.09823*, 2016.

Z. Li, P. Sharma, X. H. Lu, J. C. Cheung, and S. Reddy. Using interactive feedback to improve the accuracy and explainability of question answering systems post-deployment. *arXiv preprint arXiv:2204.03025*, 2022.

C.-Y. Lin. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain, July 2004. Association for Computational Linguistics. URL https://aclanthology.org/W04-1013.

B. Liu, G. Tur, D. Hakkani-Tur, P. Shah, and L. Heck. Dialogue learning with human teaching and feedback in end-to-end trainable task-oriented dialogue systems. *arXiv preprint arXiv:1804.06512*, 2018.

T. Liu, Y. Zhao, R. Joshi, M. Khalman, M. Saleh, P. J. Liu, and J. Liu. Statistical rejection sampling improves preference optimization. *arXiv preprint arXiv:2309.06657*, 2023.

Z. Lu, X. Li, D. Cai, R. Yi, F. Liu, X. Zhang, N. D. Lane, and M. Xu. Small language models: Survey, measurements, and insights, 2024. URL https://arxiv.org/abs/2409.15790.

L. C. Magister, J. Mallinson, J. Adamek, E. Malmi, and A. Severyn. Teaching small language models to reason. In A. Rogers, J. Boyd-Graber, and N. Okazaki, editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 1773–1781, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-short.151. URL https://aclanthology.org/2023.acl-short.151.

I. Mani and M. Maybury. Automatic summarization. page 5, 01 2001.

P. H. Martins, P. Fernandes, J. Alves, N. M. Guerreiro, R. Rei, D. M. Alves, J. Pombal, A. Farajian, M. Faysse, M. Klimaszewski, P. Colombo, B. Haddow, J. G. C. de Souza, A. Birch, and A. F. T. Martins. Eurollm: Multilingual language models for europe, 2024. URL https://arxiv.org/abs/2409.16235.

Y. Meng, M. Xia, and D. Chen. Simpo: Simple preference optimization with a reference-free reward, 2024. URL https://arxiv.org/abs/2405.14734.

R. Microsoft. Phi-2: The surprising power of small language models. https://www.microsoft.com/en-us/research/blog/phi-2-the-surprising-power-of-small-language-models/, 2024.

R. M. Neal. Slice sampling. *The annals of statistics*, 31(3):705–767, 2003.

OpenAI. Gpt-4 technical report, 2023.

L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744, 2022.

K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. Bleu: a method for automatic evaluation of machine translation. In P. Isabelle, E. Charniak, and D. Lin, editors, *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA, July 2002. Association for Computational Linguistics. doi: 10.3115/1073083.1073135. URL https://aclanthology.org/P02-1040.

R. Park, R. Rafailov, S. Ermon, and C. Finn. Disentangling length from quality in direct preference optimization, 2024. URL https://arxiv.org/abs/2403.19159.

M. Popović. chrF: character n-gram F-score for automatic MT evaluation. In O. Bojar, R. Chatterjee, C. Federmann, B. Haddow, C. Hokamp, M. Huck, V. Logacheva, and P. Pecina, editors, *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 392–395, Lisbon, Portugal, Sept. 2015. Association for Computational Linguistics. doi: 10.18653/v1/W15-3049. URL https://aclanthology.org/W15-3049.

Y. Qin and L. Specia. Truly exploring multiple references for machine translation evaluation. In İ. El[U+2010]Kahlout, M. Özkan, F. Sánchez[U+2010]Martínez, G. Ramírez[U+2010]Sánchez, F. Hollywood, and A. Way, editors, *Proceedings of the 18th Annual Conference of the European Association for Machine Translation*, Antalya, Turkey, May 11–13 2015. European Association for Machine Translation. URL https://aclanthology.org/2015.eamt-1.16.

A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

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. *arXiv preprint arXiv:2305.18290*, 2023.

S. Rajbhandari, J. Rasley, O. Ruwase, and Y. He. Zero: Memory optimizations toward training trillion parameter models. In *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–16. IEEE, 2020.

R. Ramamurthy, P. Ammanabrolu, K. Brantley, J. Hessel, R. Sifa, C. Bauckhage, H. Hajishirzi, and Y. Choi. Is reinforcement learning (not) for natural language processing: Benchmarks, baselines, and building blocks for natural language policy optimization. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=8aHzds2uUyB.

R. Rei, C. Stewart, A. C. Farinha, and A. Lavie. Comet: A neural framework for mt evaluation, 2020. URL https://arxiv.org/abs/2009.09025.

A. Saeidi, S. Verma, and C. Baral. Insights into alignment: Evaluating dpo and its variants across multiple tasks, 2024. URL https://arxiv.org/abs/2404.14723.

K. Saito, A. Wachi, K. Wataoka, and Y. Akimoto. Verbosity bias in preference labeling by large language models, 2023. URL https://arxiv.org/abs/2310.10076.

J. Scheurer, J. A. Campos, J. S. Chan, A. Chen, K. Cho, and E. Perez. Training language models with natural language feedback. *arXiv preprint arXiv:2204.14146*, 8, 2022.

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

T. Sellam, D. Das, and A. Parikh. BLEURT: Learning robust metrics for text generation. In D. Jurafsky, J. Chai, N. Schluter, and J. Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7881–7892, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.704. URL https://aclanthology.org/2020.acl-main.704.

W. Shi, Y. Li, S. Sahay, and Z. Yu. Refine and imitate: Reducing repetition and inconsistency in persuasion dialogues via reinforcement learning and human demonstration. *arXiv preprint arXiv:2012.15375*, 2020.

P. Singhal, T. Goyal, J. Xu, and G. Durrett. A long way to go: Investigating length correlations in rlhf, 2024. URL https://arxiv.org/abs/2310.03716.

F. Song, B. Yu, M. Li, H. Yu, F. Huang, Y. Li, and H. Wang. Preference ranking optimization for human alignment. *arXiv preprint arXiv:2306.17492*, 2023.

N. Stiennon, L. Ouyang, J. Wu, D. Ziegler, R. Lowe, C. Voss, A. Radford, D. Amodei, and P. F. Christiano. Learning to summarize with human feedback. *Advances in Neural Information Processing Systems*, 33:3008–3021, 2020.

G. Team, M. Riviere, S. Pathak, P. G. Sessa, C. Hardin, S. Bhupatiraju, L. Hussenot, T. Mesnard, B. Shahriari, A. Ramé, J. Ferret, P. Liu, P. Tafti, A. Friesen, M. Casbon, S. Ramos, R. Kumar, C. L.

Lan, S. Jerome, A. Tsitsulin, N. Vieillard, P. Stanczyk, S. Girgin, N. Momchev, M. Hoffman, S. Thakoor, J.-B. Grill, B. Neyshabur, O. Bachem, A. Walton, A. Severyn, A. Parrish, A. Ahmad, A. Hutchison, A. Abdagic, A. Carl, A. Shen, A. Brock, A. Coenen, A. Laforge, A. Paterson, B. Bastian, B. Piot, B. Wu, B. Royal, C. Chen, C. Kumar, C. Perry, C. Welty, C. A. Choquette-Choo, D. Sinopalnikov, D. Weinberger, D. Vijaykumar, D. Rogozińska, D. Herbison, E. Bandy, E. Wang, E. Noland, E. Moreira, E. Senter, E. Eltyshev, F. Visin, G. Rasskin, G. Wei, G. Cameron, G. Martins, H. Hashemi, H. Klimczak-Plucińska, H. Batra, H. Dhand, I. Nardini, J. Mein, J. Zhou, J. Svensson, J. Stanway, J. Chan, J. P. Zhou, J. Carrasqueira, J. Iljazi, J. Becker, J. Fernandez, J. van Amersfoort, J. Gordon, J. Lipschultz, J. Newlan, J. yeong Ji, K. Mohamed, K. Badola, K. Black, K. Millican, K. McDonell, K. Nguyen, K. Sodhia, K. Greene, L. L. Sjoesund, L. Usui, L. Sifre, L. Heuermann, L. Lago, L. McNealus, L. B. Soares, L. Kilpatrick, L. Dixon, L. Martins, M. Reid, M. Singh, M. Iverson, M. Görner, M. Velloso, M. Wirth, M. Davidow, M. Miller, M. Rahtz, M. Watson, M. Risdal, M. Kazemi, M. Moynihan, M. Zhang, M. Kahng, M. Park, M. Rahman, M. Khatwani, N. Dao, N. Bardoliwalla, N. Devanathan, N. Dumai, N. Chauhan, O. Wahltinez, P. Botarda, P. Barnes, P. Barham, P. Michel, P. Jin, P. Georgiev, P. Culliton, P. Kuppala, R. Comanescu, R. Merhej, R. Jana, R. A. Rokni, R. Agarwal, R. Mullins, S. Saadat, S. M. Carthy, S. Perrin, S. M. R. Arnold, S. Krause, S. Dai, S. Garg, S. Sheth, S. Ronstrom, S. Chan, T. Jordan, T. Yu, T. Eccles, T. Hennigan, T. Kocisky, T. Doshi, V. Jain, V. Yadav, V. Meshram, V. Dharmadhikari, W. Barkley, W. Wei, W. Ye, W. Han, W. Kwon, X. Xu, Z. Shen, Z. Gong, Z. Wei, V. Cotruta, P. Kirk, A. Rao, M. Giang, L. Peran, T. Warkentin, E. Collins, J. Barral, Z. Ghahramani, R. Hadsell, D. Sculley, J. Banks, A. Dragan, S. Petrov, O. Vinyals, J. Dean, D. Hassabis, K. Kavukcuoglu, C. Farabet, E. Buchatskaya, S. Borgeaud, N. Fiedel, A. Joulin, K. Kenealy, R. Dadashi, and A. Andreev. Gemma 2: Improving open language models at a practical size, 2024. URL https://arxiv.org/abs/2408.00118.

N. Team, M. R. Costa-jussà, J. Cross, O. Çelebi, M. Elbayad, K. Heafield, K. Heffernan, E. Kalbassi, J. Lam, D. Licht, J. Maillard, A. Sun, S. Wang, G. Wenzek, A. Youngblood, B. Akula, L. Barrault, G. M. Gonzalez, P. Hansanti, J. Hoffman, S. Jarrett, K. R. Sadagopan, D. Rowe, S. Spruit, C. Tran, P. Andrews, N. F. Ayan, S. Bhosale, S. Edunov, A. Fan, C. Gao, V. Goswami, F. Guzmán, P. Koehn, A. Mourachko, C. Ropers, S. Saleem, H. Schwenk, and J. Wang. No language left behind: Scaling human-centered machine translation, 2022. URL https://arxiv.org/abs/2207.04672.

A. S. Thakur, K. Choudhary, V. S. Ramayapally, S. Vaidyanathan, and D. Hupkes. Judging the judges: Evaluating alignment and vulnerabilities in llms-as-judges, 2024. URL https://arxiv.org/abs/2406.12624.

R. Thoppilan, D. D. Freitas, J. Hall, N. Shazeer, A. Kulshreshtha, H.-T. Cheng, A. Jin, T. Bos, L. Baker, Y. Du, Y. Li, H. Lee, H. S. Zheng, A. Ghafouri, M. Menegali, Y. Huang, M. Krikun, D. Lepikhin, J. Qin,

D. Chen, Y. Xu, Z. Chen, A. Roberts, M. Bosma, V. Zhao, Y. Zhou, C.-C. Chang, I. Krivokon, W. Rusch, M. Pickett, P. Srinivasan, L. Man, K. Meier-Hellstern, M. R. Morris, T. Doshi, R. D. Santos, T. Duke, J. Soraker, B. Zevenbergen, V. Prabhakaran, M. Diaz, B. Hutchinson, K. Olson, A. Molina, E. Hoffman-John, J. Lee, L. Aroyo, R. Rajakumar, A. Butryna, M. Lamm, V. Kuzmina, J. Fenton, A. Cohen, R. Bernstein, R. Kurzweil, B. Aguera-Arcas, C. Cui, M. Croak, E. Chi, and Q. Le. Lamda: Language models for dialog applications, 2022. URL https://arxiv.org/abs/2201.08239.

H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023a.

H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023b.

L. Tunstall, E. Beeching, N. Lambert, N. Rajani, K. Rasul, Y. Belkada, S. Huang, L. von Werra, C. Fourrier, N. Habib, et al. Zephyr: Direct distillation of lm alignment. *arXiv preprint arXiv:2310.16944*, 2023.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

M. Völske, M. Potthast, S. Syed, and B. Stein. TL;DR: Mining Reddit to learn automatic summarization. In L. Wang, J. C. K. Cheung, G. Carenini, and F. Liu, editors, *Proceedings of the Workshop on New Frontiers in Summarization*, pages 59–63, Copenhagen, Denmark, Sept. 2017. Association for Computational Linguistics. doi: 10.18653/v1/W17-4508. URL https://aclanthology.org/W17-4508.

M. Völske, M. Potthast, S. Syed, and B. Stein. Tl;dr: Mining reddit to learn automatic summarization. In *Proceedings of the Workshop on New Frontiers in Summarization*, pages 59–63, 2017.

B. Wang and A. Komatsuzaki. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. https://github.com/kingoflolz/mesh-transformer-jax, May 2021.

P. Wang, L. Li, L. Chen, Z. Cai, D. Zhu, B. Lin, Y. Cao, Q. Liu, T. Liu, and Z. Sui. Large language models are not fair evaluators, 2023. URL https://arxiv.org/abs/2305.17926.

R. Wang, H. Li, X. Han, Y. Zhang, and T. Baldwin. Learning from failure: Integrating negative examples when fine-tuning large language models as agents, 2024. URL https://arxiv.org/abs/2402.11651.

W. Weaver. Translation. In *Proceedings of the Conference on Mechanical Translation*, Massachusetts Institute of Technology, 17-20 June 1952. URL https://aclanthology.org/1952.earlymt-1.1.

J. Wei, M. Bosma, V. Y. Zhao, K. Guu, A. W. Yu, B. Lester, N. Du, A. M. Dai, and Q. V. Le. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*, 2021.

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

H. Xu, Y. J. Kim, A. Sharaf, and H. H. Awadalla. A paradigm shift in machine translation: Boosting translation performance of large language models, 2024a. URL https://arxiv.org/abs/2309.11674.

H. Xu, A. Sharaf, Y. Chen, W. Tan, L. Shen, B. V. Durme, K. Murray, and Y. J. Kim. Contrastive preference optimization: Pushing the boundaries of llm performance in machine translation, 2024b. URL https://arxiv.org/abs/2401.08417.

J. Xu, M. Ung, M. Komeili, K. Arora, Y.-L. Boureau, and J. Weston. Learning new skills after deployment: Improving open-domain internet-driven dialogue with human feedback. *arXiv preprint arXiv:2208.03270*, 2022.

E. Yankovskaya, A. Tättar, and M. Fishel. Quality estimation and translation metrics via pre-trained word and sentence embeddings. In O. Bojar, R. Chatterjee, C. Federmann, M. Fishel, Y. Graham, B. Haddow, M. Huck, A. J. Yepes, P. Koehn, A. Martins, C. Monz, M. Negri, A. Névéol, M. Neves, M. Post, M. Turchi, and K. Verspoor, editors, *Proceedings of the Fourth Conference on Machine Translation (Volume 3: Shared Task Papers, Day 2)*, pages 101–105, Florence, Italy, Aug. 2019. Association for Computational Linguistics. doi: 10.18653/v1/W19-5410. URL https://aclanthology.org/W19-5410.

J. Ye, X. Chen, N. Xu, C. Zu, Z. Shao, S. Liu, Y. Cui, Z. Zhou, C. Gong, Y. Shen, et al. A comprehensive capability analysis of gpt-3 and gpt-3.5 series models. *arXiv preprint arXiv:2303.10420*, 2023.

Z. Yuan, H. Yuan, C. Tan, W. Wang, S. Huang, and F. Huang. Rrhf: Rank responses to align language models with human feedback without tears. *arXiv preprint arXiv:2304.05302*, 2023.

P. Zhang, G. Zeng, T. Wang, and W. Lu. Tinyllama: An open-source small language model, 2024a. URL https://arxiv.org/abs/2401.02385.

T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, and Y. Artzi. Bertscore: Evaluating text generation with bert, 2020. URL https://arxiv.org/abs/1904.09675.

X. Zhang, W. Xiong, L. Chen, T. Zhou, H. Huang, and T. Zhang. From lists to emojis: How format bias affects model alignment, 2024b. URL https://arxiv.org/abs/2409.11704.

W. Zhao, G. Glavaš, M. Peyrard, Y. Gao, R. West, and S. Eger. On the limitations of cross-lingual encoders as exposed by reference-free machine translation evaluation. In D. Jurafsky, J. Chai,

N. Schluter, and J. Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1656–1671, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.151. URL https://aclanthology.org/2020.acl-main.151.

Y. Zhao, M. Khalman, R. Joshi, S. Narayan, M. Saleh, and P. J. Liu. Calibrating sequence likelihood improves conditional language generation. *arXiv preprint arXiv:2210.00045*, 2022.

Y. Zhao, R. Joshi, T. Liu, M. Khalman, M. Saleh, and P. J. Liu. Slic-hf: Sequence likelihood calibration with human feedback. *arXiv preprint arXiv:2305.10425*, 2023.

L. Zheng, W.-L. Chiang, Y. Sheng, S. Zhuang, Z. Wu, Y. Zhuang, Z. Lin, Z. Li, D. Li, E. P. Xing, H. Zhang, J. E. Gonzalez, and I. Stoica. Judging llm-as-a-judge with mt-bench and chatbot arena, 2023. URL https://arxiv.org/abs/2306.05685.

D. M. Ziegler, N. Stiennon, J. Wu, T. B. Brown, A. Radford, D. Amodei, P. Christiano, and G. Irving. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*, 2019.

A
Appendix

A.1 Training and model details

A.1.1 MT Template

We finetune our MT models with the `chatml` template (Casper et al., 2023). Table A.1 provides an example of an interaction using the aforementioned template.

User	`<|im_start|>user`
	Translate the following English source text to German:
	English: Police arrest 15 after violent protest outside UK refugee hotel
	German: `<|im_end|>`
	`<|im_start|>assistant`

Model	Polizei nimmt 15 wegen gewalttätigen Protestes vor UK-Flüchtlingshotel fest `<|im_end|>`

Table A.1: Example of a dialogue with one of the trained models with user and model control tokens.

A.1.2 Summarization Template

We finetune our MT models with the template presented in Table A.2, as done in Stiennon et al. (2020).

Format	**Max tokens**

SUBREDDIT: r/{subreddit}
TITLE: {title}
POST: {post}
TL;DR:
	512

Table A.2: Summarization template used to train our models. Based on Stiennon et al. (2020).