(2022), is aimed at improving the model's helpfulness by focusing on natural language generation for diverse tasks. To ensure the model's ability to generalize to a wide range of scenarios, we specifically excluded data formatted in prompt templates that could potentially limit its capabilities. Furthermore, we have prioritized the safety of the language model by annotating data related to safety concerns such as violence, bias, and pornography. This will enable the model to detect and reject malicious prompts or provide safe answers in such situations.

In addition to data quality, we have observed that the training method can significantly impact the final performance of the model. To achieve this, we utilized the ChatML-style format (OpenAI, 2022), which is a versatile meta language capable of describing both the metadata (such as roles) and the content of a turn. This format enables the model to effectively distinguish between various types of information, including system setup, user inputs, and assistant outputs, among others. By leveraging this approach, we can enhance the model's ability to accurately process and analyze complex conversational data.

3.1.2 TRAINING

Consistent with pretraining, we also apply next-token prediction as the training task for SFT. We apply the loss masks for the system and user inputs. More details are demonstrated in Appendix A.1.1.

The model's training process utilizes the AdamW optimizer, with the following hyperparameters: β_1 set to 0.9, β_2 set to 0.95, and ϵ set to 10^{-8}. The sequence length is limited to 2048, and the batch size is 128. The model undergoes a total of 4000 steps, with the learning rate gradually increased over the first 1430 steps, reaching a peak of 2×10^{-6}. To prevent overfitting, weight decay is applied with a value of 0.1, dropout regularization is set to 0.1, and gradient clipping is enforced with a limit of 1.0.

3.2 REINFORCEMENT LEARNING FROM HUMAN FEEDBACK

While SFT has proven to be effective, we acknowledge that its generalization and creativity capabilities may be limited, and it is prone to overfitting. To address this issue, we have implemented Reinforcement Learning from Human Feedback (RLHF) to further align SFT models with human preferences, following the approaches of Ouyang et al. (2022); Christiano et al. (2017). This process involves training a reward model and using Proximal Policy Optimization (PPO) (Schulman et al., 2017) to conduct policy training.

3.2.1 REWARD MODEL

To create a successful reward model, like building a large language model (LLM), it is crucial to first undergo pretraining and then finetuning. This pretraining process, also known as preference model pretraining (PMP) (Bai et al., 2022b), necessitates a vast dataset of comparison data. This dataset consists of sample pairs, each containing two distinct responses for a single query and their corresponding preferences. Similarly, finetuning is also conducted on this type of comparison data, but with a higher quality due to the presence of quality annotations.

During the fine-tuning phase, we gather a variety of prompts and adjust the reward model based on human feedback for responses from the QWEN models. To ensure the diversity and complexity of user prompts are properly taken into account, we have created a classification system with around 6600 detailed tags and implemented a balanced sampling algorithm that considers both diversity and complexity when selecting prompts for annotation by the reward model (Lu et al., 2023). To generate a wide range of responses, we have utilized QWEN models of different sizes and sampling strategies, as diverse responses can help reduce annotation difficulties and enhance the performance of the reward model. These responses are then evaluated by annotators following a standard annotation guideline, and comparison pairs are formed based on their scores.

In creating the reward model, we utilize the same-sized pre-trained language model QWEN and initiate the PMP process. Subsequently, we fine-tune the PMP model to enhance its performance. It is important to mention that we have incorporated a pooling layer into the original QWEN model to extract the reward for a sentence based on a specific end token. The learning rate for this process has been set to a constant value of 3×10^{-6}, and the batch size is 64. Additionally, the sequence length is set to 2048, and the training process lasts for a single epoch.