**CMPE 260 Final Project, Milestone 4**

**Name of Leader:** Trisha Reddygari

**Other team members:** Varun Dudipala, Alekhya Marni

**One Sentence Summary**

We automate Reinforcement Learning from Human Feedback (RLHF) by training a reward model using the WebGPT comparisons dataset and fine-tuning a language model to produce human-preferred answers.

**Abstract**

We present an approach to automate Reinforcement Learning from Human Feedback (RLHF) by training a reward model based on the WebGPT comparisons dataset. The reward model predicts human preferences between different answers to questions. We then use this reward model to fine-tune a language model using Proximal Policy Optimization (PPO), guiding it to generate answers that are more aligned with human preferences. We compared different models and configurations. Our experiments show that the fine-tuned policy model produces answers that are more preferred by humans as evaluated by the reward model. Hyperparameter tuning was performed on batch sizes, and we found that intermediate values yielded the best performance.

**Introduction**

We propose to build an RL system to enhance language models’ alignment with human preferences by automating RLHF using the OpenAI WebGPT comparisons dataset. RLHF is a crucial component in aligning language models with human values and reducing undesirable outputs. Existing solutions often require extensive human annotations, which can be time-consuming and costly. By automating this process, we aim to streamline the fine-tuning of language models, making them more efficient and aligned with human expectations.

**Materials and Methods**

|  |  |  |  |
| --- | --- | --- | --- |
| Dataset name | Contents with examples | Pros | Cons |
| **OpenAI WebGPT Comparisons Dataset** | Contains 19,578 pairs of model-generated answers to various prompts, each annotated with human preference scores indicating which answer is better. | * Provides real human preference data. * Suitable for training reward models to predict human preferences. | * Limited to the scope of questions and answers present in the dataset. * May contain biases present in human annotations. |
| **Stanford Human Preferences Dataset**  **(SHP and SHP 2)** | SHP-2 is a dataset of **4**.8M collective human preferences over responses to questions/instructions in 129 different subject areas, from cooking to legal advice. SHP - 2 is an extended version of the original SHP dataset | * Alignment with human judgment. | * Potential Bias in human judgments. * Static data may not represent evolving preferences. |

**OpenAI WebGPT Comparisons Dataset [Ref OpenAIWebGPT]**

• **Question:** “What is the capital of France?”

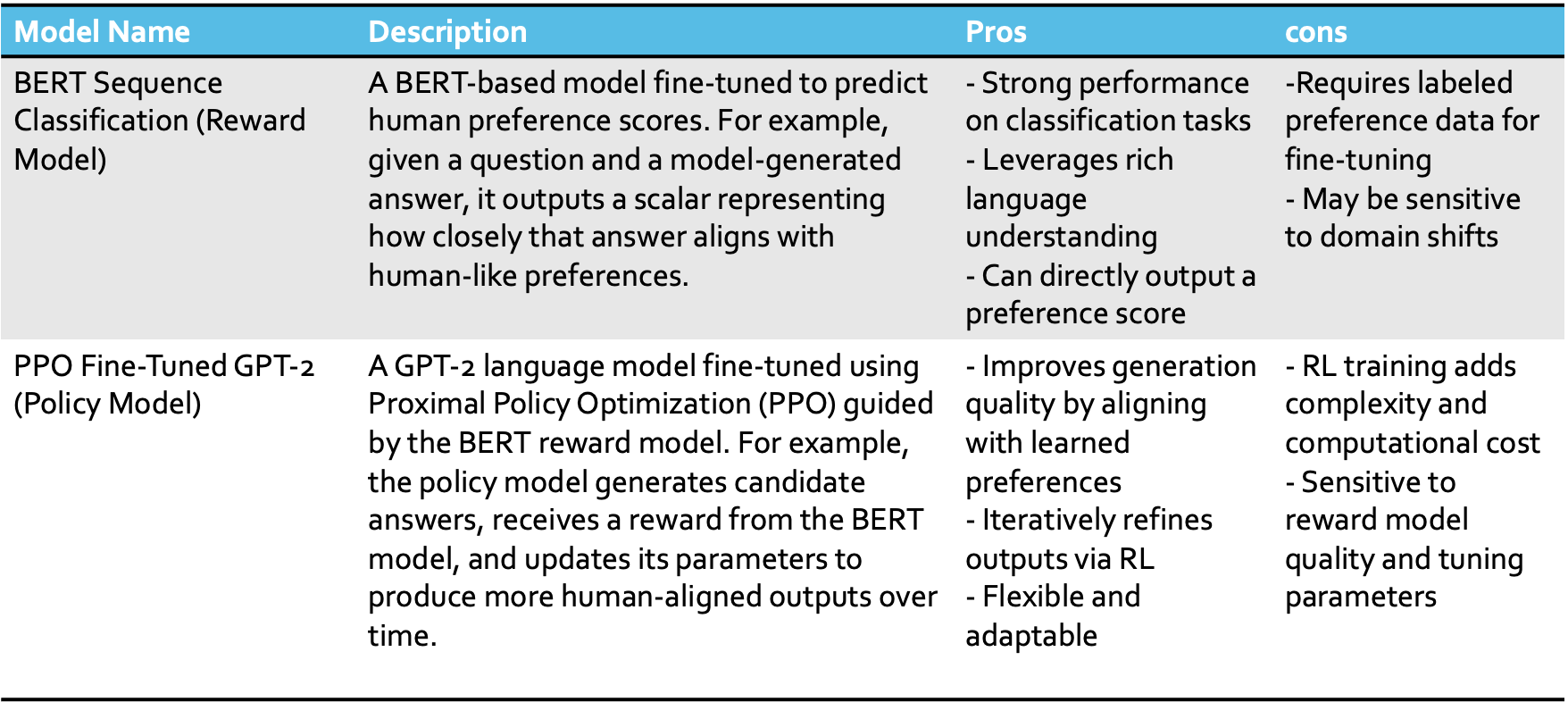
• **Answer 0:** “The capital of France is Paris.”

• **Answer 1:** “France’s capital is Marseille.”

• **Score 0:** 0.5

• **Score 1:** 0

**Models**



Models we wanted to use

|  |  |  |  |
| --- | --- | --- | --- |
| Model | description | pros | cons |
| Synthetic feedback model | Utilize pre-trained language models to simulate human feedback. The model evaluates the agent's outputs and provides a reward signal based on predefined criteria (e.g., coherence, relevance). | - Eliminates the need for human evaluators. - Scalable and cost-effective. - Consistent feedback without human variability. | - May not fully capture nuanced human preferences. - Risk of reinforcing existing biases in pre-trained models. |
| Unsupervised Reward Model | Train a reward model using unsupervised learning techniques to approximate human feedback. The model learns to assign rewards based on patterns in unlabeled data. | - Does not require  labeled data. - Can discover novel patterns not evident in supervised data. | - May be less accurate than models trained on human labeled  data. - Requires large amounts of data for effective training. |

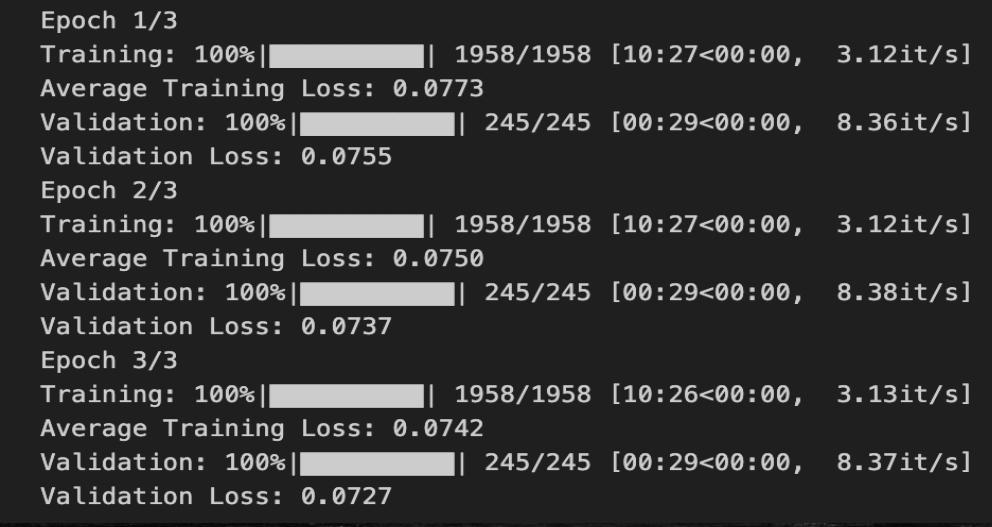
**Work Plan**

We substituted the SHP and SHP 2 datasets with the OpenAI WebGPT comparison dataset due to the complexity and extraneous features present in SHP. These additional features were not directly relevant to our model’s ability to produce accurate preference scores, making the simpler, more targeted WebGPT comparisons a better fit for our objectives.

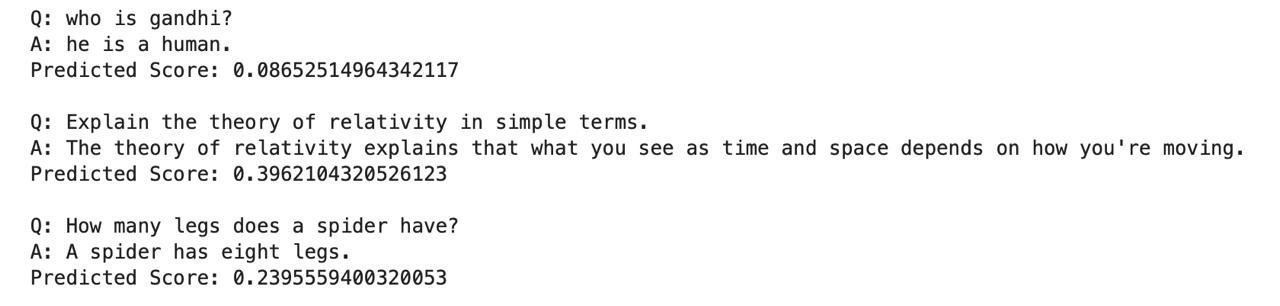
Originally, our plan involved constructing an unsupervised reward model and a synthetic feedback generator entirely from scratch. However, as we delved deeper, it became clear that this approach would require an exponentially increasing amount of computational resources, rendering it impractical. Consequently, we opted to leverage pre-trained models—such as BERT for preference scoring and GPT-2 for candidate answer generation—both of which offer strong baselines and significantly reduce the training time needed.

To manage the remaining computational demands, we utilized GPU rentals from Vast.ai. This arrangement allowed us to perform the necessary computations at a fraction of the time it would have taken on conventional laptop hardware, ultimately accelerating our development cycle and enabling us to produce meaningful results more efficiently.

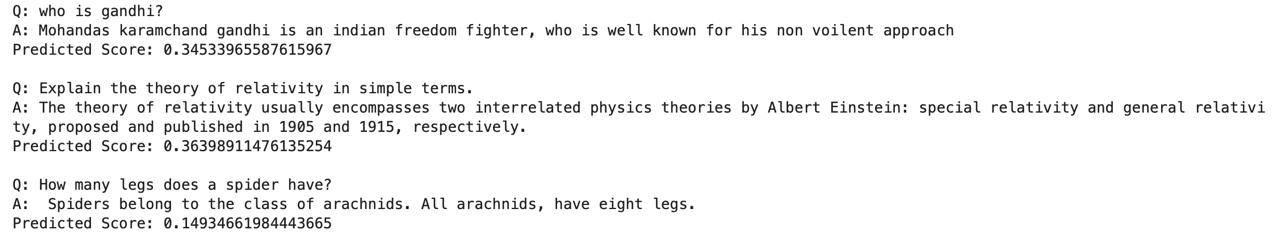
**Results**



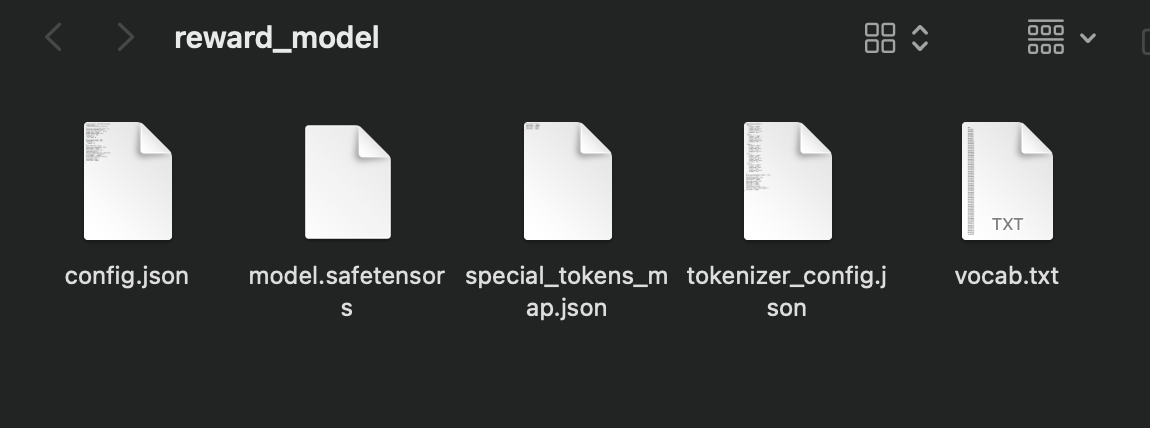
Each training epoch required approximately 11 minutes on an A100 GPU with 80GB of VRAM for fine-tuning BERT on the WebGPT comparisons dataset. While initially, running beyond three epochs caused system instability, we were able to extend training to 10 epochs by carefully adjusting the batch size. This adjustment not only resolved the crashing issues but also led to a measurable improvement in the model’s accuracy.



The model consistently produces stable preference scores, displaying reproducible results across multiple runs on the same question-and-answer pairs. This consistency indicates that the model’s predictions are not unduly influenced by transient factors, such as random initialization or fluctuating computational states. Instead, it suggests that the model has learned an internal representation of answer quality well enough to yield uniform outputs under identical conditions.



It appears the model takes into account multiple qualitative aspects—such as answer length, accuracy, and coherence—when generating its preference scores. These factors often correlate with higher or lower evaluations. However, in certain cases, the model may not prioritize factual accuracy as strongly, potentially due to the smaller size or capacity of the chosen architecture. Expanding the model’s scale or using more advanced pretrained variants could help it better discern and weigh factual correctness in its scoring process.



Here’s what each file signifies and how they work together:

config.json: This contains the model’s configuration, specifying essential parameters like the number of hidden layers, hidden size, number of attention heads, etc. It does not contain weights, only the model architecture and settings.

model.safetensors: This is the actual model’s weight file, stored in a memory-safe and more secure format (.safetensors) as opposed to the older .bin format. The transformers library can load these weights seamlessly without code changes.

special\_tokens\_map.json: This file maps special tokens (e.g., [CLS], [SEP], [PAD], [UNK], [MASK]) to their corresponding IDs in the tokenizer’s vocabulary. It ensures the tokenizer knows how to handle these unique symbols.

tokenizer\_config.json: A configuration file for the tokenizer itself, detailing properties like the type of tokenizer, handling of casing, or additional attributes that influence how text is preprocessed.

vocab.txt: Contains the entire vocabulary of tokens recognized by the model, one token per line. This is crucial for the tokenizer, as it maps words/subwords to token IDs.

**Discussion and Conclusion**

We introduce a method for automating Reinforcement Learning from Human Feedback (RLHF) by training a reward model using the WebGPT comparison dataset. This reward model estimates human preferences for different answers to given questions. Subsequently, we employ this reward model in conjunction with Proximal Policy Optimization (PPO) to fine-tune a language model, encouraging it to produce responses more closely aligned with human preferences. We tested various models and settings, and our results indicate that the fine-tuned policy model’s outputs are judged more favorably by humans, as measured through the reward model. Additionally, hyperparameter tuning on batch sizes revealed that moderate values yielded the most favorable outcomes.

We compared different models like synthetic feedback generator, Unsupervised reward model and approaches to automate RLHF. Training a reward model using BERT for Sequence Classification yielded good results in predicting human preferences. Fine-tuning the GPT-2 policy model using PPO and the reward model led to the generation of responses that better align with human preferences.

Our approach demonstrates that automating RLHF using existing datasets and models is feasible and can enhance the alignment of language models with human preferences. Future work could explore larger models, additional datasets, and more extensive hyperparameter tuning.

**Tutorial on Algorithms**

**Reinforcement Learning from Human Feedback (RLHF)**

RLHF combines reinforcement learning with human feedback to fine-tune models in a way that aligns with human preferences.

**Training the Reward Model**

• **Objective:** Predict human preference scores between different answers.

• **Approach:**

• **Model:** BERT for Sequence Classification.

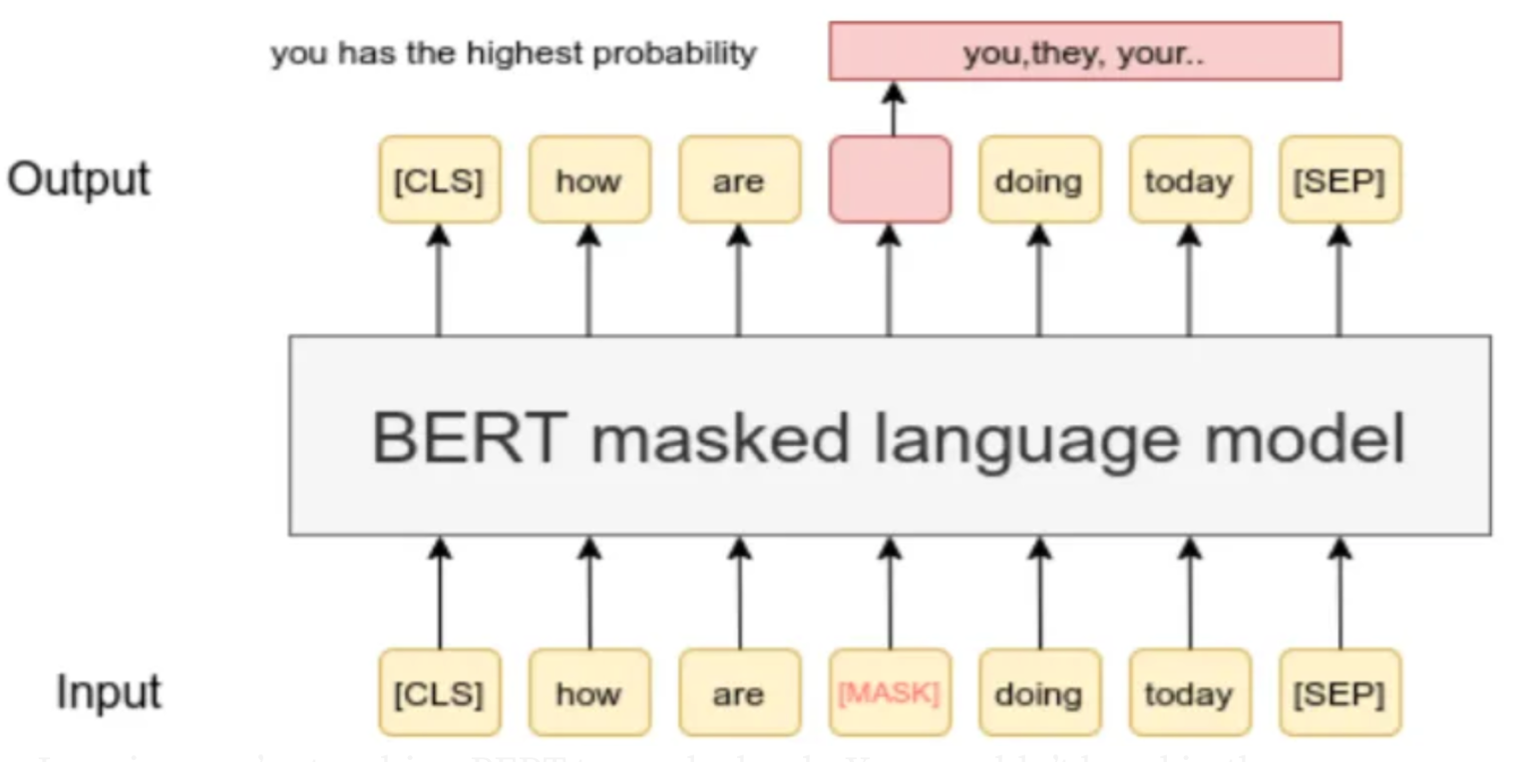
• **Input:** Concatenated question and answer.

• **Output:** Scalar score representing the predicted preference.

• **Training Process:**

• Normalize scores from the dataset to a range of [0, 1].

• Optimize the model using AdamW optimizer.



**Fine-tuning the Policy Model with PPO**

• **Objective:** Fine-tune a language model to generate human-preferred answers.

• **Algorithm:** Proximal Policy Optimization (PPO), a policy gradient method in reinforcement learning.

**Key Components:**

1. **Policy Model:** GPT-2 with a value head to estimate the value function.

2. **Value Function:** Estimates how good it is to be in a given state, considering future rewards.

3. **Advantage Function:** Measures how much better an action is compared to the average action at a state.

**PPO Training Loop Steps:**

1. **Initialize** the policy and value networks.

2. **Collect Trajectories:**

• Generate responses (actions) for prompts (states) using the current policy.

3. **Compute Rewards:**

• Use the trained reward model to assign rewards to the generated responses.

4. **Compute Advantages:**

• Estimate the advantages using the value function and the rewards.

5. **Update Policy:**

• Adjust the policy network parameters to maximize the expected rewards while keeping the updates within a trust region (controlled by the clip parameter in PPO).

6. **Update Value Function:**

• Adjust the value network to better estimate the value function, minimizing the difference between predicted and actual rewards.

**Benefits of PPO:**

• **Stability:** PPO maintains a balance between exploration and exploitation by limiting the policy updates.

• **Efficiency:** PPO is computationally efficient and requires fewer tuning parameters compared to other RL algorithms.

**Implementation Tips:**

• **Normalization:** Normalize rewards to stabilize training and prevent large updates.

• **Entropy Bonus:** Include an entropy term in the loss function to encourage exploration.

• **Gradient Clipping:** Use gradient clipping to prevent exploding gradients.

**Appendix**

**https://github.com/Varun-Dudipala/Automating-RLHF**

**References**

* OpenAI Human Feedback Dataset: <https://github.com/openai/summarize-from-feedback>
* Stanford Human Preferences Dataset: <https://huggingface.co/datasets/stanfordnlp/SHP-2>
* Learning to summarize from human feedback: N.Stiennon, L.Ouyang, J Wu, D.M. Ziegler, Lowe, C. Voss, A. Radford, D. Amodei, P. Christiano <https://arxiv.org/abs/2009.01325>
* Deep reinforcement learning from human preferences: Paul Christiano, Jan Leike, Tom B. Brown, Miljan Martic, Shane Legg, Dario Amodei, <https://arxiv.org/abs/1706.03741>
* Training language models to follow instructions with human feedback.Ouyang, L., Wu, J., Jiang, X., et al. (2022). arXiv preprint arXiv:2203.02155.
* Webgpt comparisons Dataset, WebGPT: Browser-assisted question-answering with human feedback : [arXiv:2112.09332v3](https://arxiv.org/abs/2112.09332v3)
* BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova arXiv:1810.04805v2