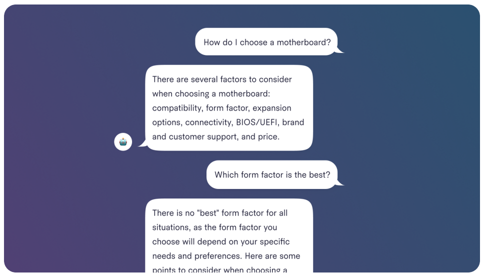
DEEP LEARNING

How ChatGPT actually works

Since its release, the public has been playing with ChatGPT and seeing what it can do, but how does ChatGPT actually work? While the details of its inner workings have not been published, we can piece together its functioning principles from recent research.



How do I choose a motherboard?

There are several factors to consider when choosing a motherboard: compatibility, form factor, expansion options, connectivity, BIOS/UEFI, brand and customer support, and price.

Which form factor is the best?

There is no "best" form factor for all situations, as the form factor you choose will depend on your specific needs and preferences. Here are some points to consider when choosing a

Marco Ramponi
Developer Educator

Dec 23, 2022

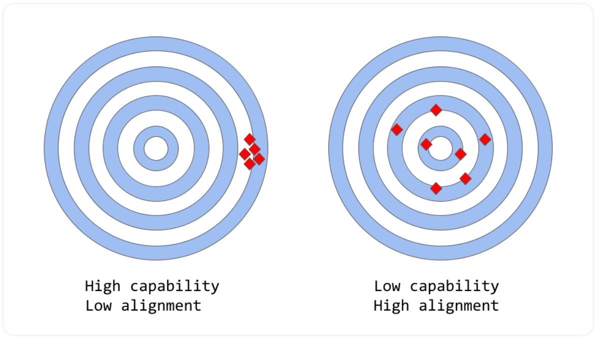
ChatGPT is the latest language model from OpenAI and represents a significant improvement over its predecessor GPT-3. Similarly to many Large Language Models, ChatGPT is different purposes, but with remarkably greater precision, detail, and coherence. It represents the next generation in OpenAI's line of Large Language Models, and it is designed with a strong focus on interactive conversations.

The creators have used a combination of both Supervised Learning and Reinforcement Learning to fine-tune ChatGPT, **but it is the Reinforcement Learning component specifically that makes ChatGPT unique**. The creators use a particular technique called Reinforcement Learning from Human Feedback (RLHF), which uses human feedback in the training loop to minimize harmful, untruthful, and/or biased outputs.

We are going to examine GPT-3's limitations and how they stem from its training process, before learning

overcome these issues. We will conclude by looking at some of the limitations of this methodology.

Capability vs Alignment in Large Language Models



"alignment vs capability" can be thought of as a more abstract analogue of "accuracy vs precision"

In the context of machine learning, the term *capability* refers to a model's ability to perform a specific task or set of tasks. A model's capability is typically evaluated by **how well it is able to optimize its objective function**, the mathematical expression that defines the goal of the model. For have an objective function that measures the accuracy of the model's predictions. If the model is able to accurately predict the movement of stock prices over time, it would be considered to have a high level of capability for this task.

Alignment, on the other hand, is concerned with **what we actually want the model to do** versus what it is being trained to do. It asks the question “is that objective function consistent with our intentions?” and refers to the extent to which a model's goals and behavior align with human values and expectations. For a simple concrete example, say we train a bird classifier to classify birds as either "sparrows" or "robins" and we use *log loss* (which measures the difference between the predicted probability distribution of the model and the true distribution) as the training accuracy. The model might have low log loss, i.e. **the model's capability is high, but poor accuracy** on the test set. In fact, the log loss is not perfectly correlated with accuracy in classification tasks. This is an example of misalignment, where the model is capable of optimizing the training objective but poorly aligned with our ultimate goal.

Models like the original GPT-3 are misaligned

Large Language Models, such as GPT-3, are trained on vast amounts of text data from the internet **and are capable of generating human-like text**, but they may not always produce output that is consistent with human expectations or desirable values. In fact, their objective function is a probability distribution over word sequences (or token sequence (more details on this below).

In practical applications, however, **these models are intended to perform some form of valuable cognitive work**, and there is a clear divergence between the way these models are trained and the way we would like to use them. Even though a machine calculated statistical distribution of word sequences might be, mathematically speaking, a very effective choice to model language, we as humans generate language by choosing text sequences that are best for the given situation, using our background knowledge and common sense to guide this process. This can be a problem when language models are used in applications that require a high degree of trust or reliability, such as dialogue systems or intelligent personal assistants.

of data have become extremely capable in the last few years, when used in production systems to make human lives easier they often fall short of this potential. The alignment problem in Large Language Models typically manifests as:

- Lack of helpfulness: not following the user's explicit instructions.
- Hallucinations: model making up unexisting or wrong facts.
- Lack of interpretability: it is difficult for humans to understand how the model arrived at a particular decision or prediction.
- Generating biased or toxic output: a language model that is trained on biased/toxic data may reproduce that in its output, even if it was not explicitly instructed to do so.

it **the very way language models are trained inherently prone to misalignment?**

How language model training strategies can produce misalignment

`Next-token-prediction` and `masked-language-modeling` are **the core techniques used for training language models**, such as transformers. In the first approach, the model is given a sequence of words (or “tokens”, i.e. parts of words) as input and is asked to predict the next word in the sequence. For example, if the model is given the input sentence

"The cat sat on the"

it might predict the next word as *"mat"*, *"chair"*, or *"floor"*, because of the high-probability of occurrence of these words given the previous context; the language

word (in its vocabulary) given the previous sequence.

The masked language modeling approach is a variant of next token prediction, in which some of the words in the input sentence are replaced with a special token, such as `[MASK]`. The model is then asked to predict the correct word that should be inserted in place of the mask. For example, if the model is given the sentence

"The `[MASK]` sat on the"

as input, it might predict the next word as *"cat"*, *"dog"*, or *"rabbit"*.

One advantage of these objective functions is that it allows the model to **learn the statistical structure of language**, such as common word sequences and patterns of word usage. This generally helps the model generate more natural and fluent text, and it is an

However, these objective functions can also lead to problems, essentially because the model is not capable of **distinguishing between an important error and an unimportant one**. To make a very simple example, if the model is given the input sentence:

"The Roman Empire `[MASK]` with the reign of Augustus."

it might predict *"began"* or *“ended”*, as both words score high likelihood of occurrence (indeed, both sentences are historically correct), even though the second choice implies a very different meaning.

More generally, these training strategies can lead to a misalignment of the language model for some more complex tasks, because a model which is only trained to predict the next word (or a masked word) in a text sequence, **may not necessarily**

As a result, the model struggles to generalize to tasks or contexts that require a deeper understanding of language.

Researchers and developers are working on various approaches to address the alignment problem in Large Language Models. ChatGPT is based on the original GPT-3 model, but has been further trained by using human feedback to guide the learning process with the specific goal of mitigating the model's misalignment issues. The specific technique used, called Reinforcement Learning from Human Feedback, is based on previous academic research. ChatGPT represents the **first case of use of this technique for a model put into production**.

But how exactly do the creators of ChatGPT use human feedback to attack the alignment problem?

Learning from Human Feedback

The method overall consists of three distinct steps:

1. Supervised fine-tuning step: a pre-trained language model is fine-tuned on a relatively small amount of demonstration data curated by labelers, to learn a supervised policy (the **SFT model**) that generates outputs from a selected list of prompts. This represents the baseline model.
2. “Mimic human preferences” step: labelers are asked to vote on a relatively large number of the SFT model outputs, this way creating a new dataset consisting of *comparison data*. A new model is trained on this dataset. This is referred to as the **reward model (RM)**.

is used to further fine-tune and improve the SFT model. The outcome of this step is the so-called **policy model**.

Step 1 takes place only once, while steps 2 and 3 can be iterated continuously: more comparison data is collected on the current best policy model, which is used to train a new reward model and then a new policy.

Let’s now dive into the details of each step!

Note: The rest of this article is based on the content of the InstructGPT paper. According to OpenAI, ChatGPT has been trained *“using the same methods as InstructGPT, but with slight differences in the data collection setup”* (source). Unfortunately, exact quantitative reports have yet to be made publicly available for ChatGPT.

Fine-Tuning (SFT) model

The first step consists in collecting demonstration data in order to train a supervised policy model, referred to as the SFT model.

- Data collection: a list of prompts is selected and a group of human labelers are asked to **write down the expected output response**. For ChatGPT, two different sources of prompts have been used: some have been prepared directly from the labelers or developers, some have been sampled from OpenAI’s API requests (i.e. from their GPT-3 customers). As this whole process is slow and expensive, the result is a relatively small, high-quality curated dataset (of approximately 12-15k data points, presumably) that is to be used to fine-tune a pretrained language model.

model, the developers of ChatGPT opted for a pretrained model in the so-called GPT-3.5 series. Presumably the baseline model used is the latest one `text-davinci-003`, a GPT-3 model which was fine-tuned mostly on programming code.

Quite interestingly, therefore, in order to create a general purpose chatbot like ChatGPT, the developers decided to **fine-tune on top of a “code model”** rather than a pure text model.

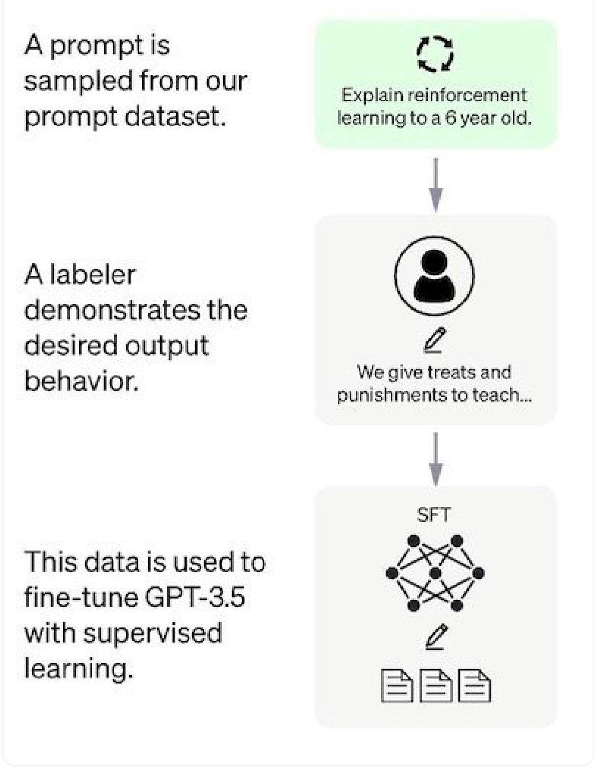

A prompt is sampled from our prompt dataset.
Explain reinforcement learning to a 6 year old.
A labeler demonstrates the desired output behavior.
We give treats and punishments to teach...
This data is used to fine-tune GPT-3.5 with supervised learning.
SFT

Figure adapted from source

Due to the limited amount of data for this step, the SFT model obtained after this process is likely to output text which is still (probabilistically) not very user-attentive and generally suffers from misalignment, in the sense explained in the above sections. The problem here is that **the supervised learning step suffers from high scalability costs**.

a much bigger curated dataset, a slow and costly process, the strategy is now to have the labelers rank different outputs of the SFT model to create a reward model –let's explain this in more detail in the following section.

Step 2: The reward model (RM)

The goal is to learn an objective function (the reward model) **directly from the data**. The purpose of this function is to give a score to the SFT model outputs, proportional to how desirable these outputs are for humans. In practice, this will strongly reflect the specific preferences of the selected group of human labelers and the common guidelines which they agreed to follow. In the end, this process will extract from the data an automatic system that is supposed to **mimic human preferences**.

- A list of prompts is selected and the SFT model generates multiple outputs (anywhere between 4 and 9) for each prompt.
- Labelers rank the outputs from best to worst. The result is a new labeled dataset, where the rankings are the labels. The size of this dataset is approximately 10 times bigger than the curated dataset used for the SFT model.
- This new data is used to train a reward model (RM). This model takes as input a few of the SFT model outputs and ranks them in order of preference.

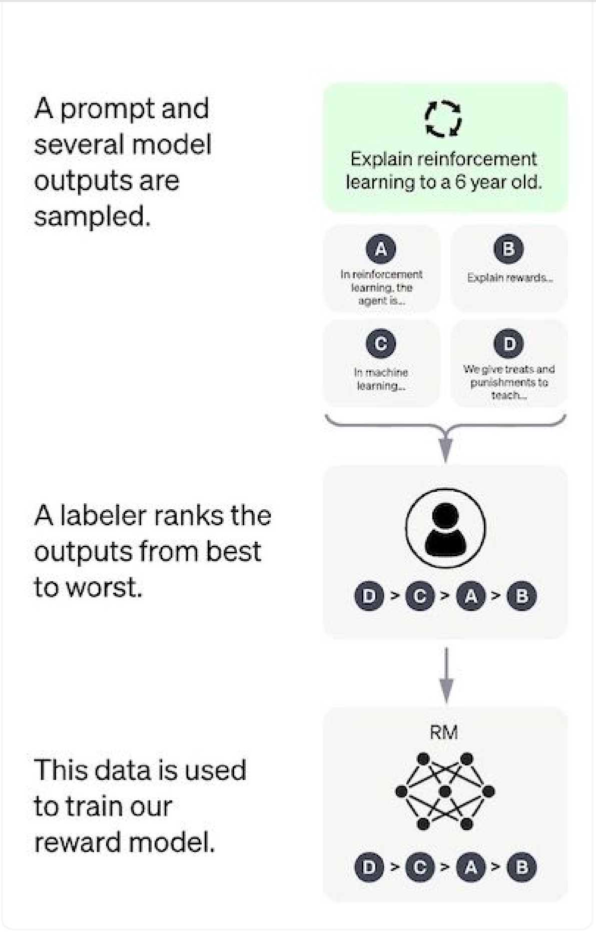

A prompt and several model outputs are sampled.
Explain reinforcement learning to a 6 year old.
A
In reinforcement learning, the agent is...
B
Explain rewards...
C
In machine learning...
D
We give treats and punishments to teach...
A labeler ranks the outputs from best to worst.
D > C > A > B
This data is used to train our reward model.
RM
D > C > A > B

Figure adapted from source

As for labelers it is much easier to rank the outputs than to produce them from scratch, **this process scales up much more efficiently**. In practice, this dataset has been generated from a selection of 30-40k prompts, and a variable number of the generated outputs (for each prompt) is presented to the each labeler during the ranking phase.

SFT model via Proximal Policy Optimization (PPO)

Reinforcement Learning is now applied to fine-tune the SFT policy by letting it optimize the reward model. The specific algorithm used is called Proximal Policy Optimization (PPO) and the fine-tuned model is referred to as the PPO model.

What is PPO? Here are the main takeaways of this method:

- PPO is an algorithm that is used to train agents in reinforcement learning. It is called an "on-policy" algorithm because **it learns from and updates the current policy directly**, rather than learning from past experiences as in "off-policy" algorithms like DQN (Deep Q-Network). This means that PPO is continuously adapting the current policy based on the

- PPO uses a trust region optimization method to train the policy, which means that **it constrains the change in the policy to be within a certain distance of the previous policy** in order to ensure stability. This is in contrast to other policy gradient methods which can sometimes make large updates to the policy that can destabilize learning.
- PPO uses a **value function to estimate the expected return of a given state or action**. The value function is used to compute the advantage function, which represents the difference between the expected return and the current return. The advantage function is then used to update the policy by comparing the action taken by the current policy to the action that would have been taken by the

updates to the policy based on the estimated value of the actions being taken.

In this step, the PPO model is initialized from the SFT model, and **the value function is initialized from the reward model**. The environment is a bandit environment which presents a random prompt and expects a response to the prompt. Given the prompt and response, it produces a reward (determined by the reward model) and the episode ends. A per-token KL penalty is added from the SFT model at each token to mitigate over optimization of the reward model.

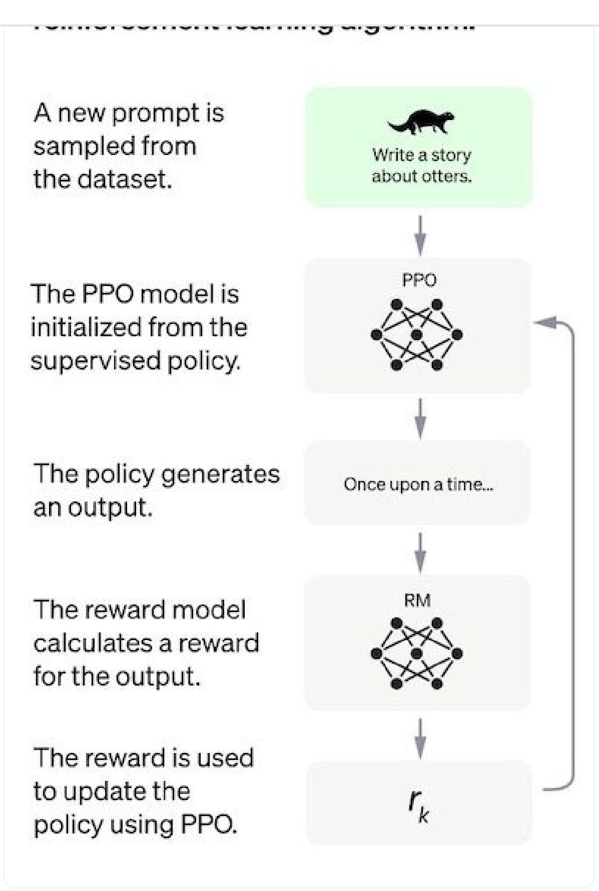

A new prompt is sampled from the dataset.
Write a story about otters.
The PPO model is initialized from the supervised policy.
PPO
The policy generates an output.
Once upon a time...
The reward model calculates a reward for the output.
RM
The reward is used to update the policy using PPO.
r_k

Figure adapted from source

Performance Evaluation

Because the model is trained on human labelers input, **the core part of the evaluation is also based on human input**, i.e. it takes place by having labelers rate the quality of the model outputs. To avoid overfitting to the judgment of the labelers involved in the training phase, the test set uses prompts training data.

The model is evaluated on three high-level criteria:

- Helpfulness: judging the model's ability to follow user instructions, as well as infer instructions.
- Truthfulness: judging the model's tendency for hallucinations (making up facts) on closed-domain tasks. The model is evaluated on the TruthfulQA dataset.
- Harmlessness: the labelers evaluate whether the model's output is appropriate, denigrates a protected class, or contains derogatory content. The model is also benchmarked on the RealToxicityPrompts and CrowS-Pairs datasets.

The model is also evaluated for zero-shot performance on comprehension, and summarization, on some of which **the developers observed performance regressions compared to GPT-3**. This is an example of an “alignment tax” where the RLHF-based alignment procedure comes at the cost of lower performance on certain tasks.

The performance regressions on these datasets can be greatly reduced with a trick called **pre-train mix**: during training of the PPO model via gradient descent, the gradient updates are computed by mixing the gradients of the SFT model and the PPO model.

Shortcomings of the methodology

A very clear limitation of the methodology, as discussed in the InstructGPT paper (on which ChatGPT is based, according to its models with human intentions, the data for fine-tuning the models is influenced by an intricate variety of subjective factors, including:

- The preferences of the labelers who produce the demonstration data.
- The researchers who design the study and write the labeling instructions.
- The choice of prompts crafted by the developers or provided by the OpenAI customers.
- The labelers bias is both included in the reward model training (by ranking outputs) **and in the model evaluation**.

In particular, the authors point out the obvious fact that the labelers and researchers taking part in the training process may not be representative of all potential end users of the language model.

few other possible shortcomings of the method, problems not explicitly addressed, as well as some open questions:

Lack of control study: The reported results measure the performance of the final PPO model, taking the SFT model as the baseline. This can be misleading: how can we know the improvements are actually due to the RLHF? A proper (yet, expensive) control study would consist in investing the exact same amount of labeler-hours as those used to train the reward model into creating a larger curated SFT dataset with high-quality demonstration data. One would then be in the position to objectively measure the performance improvements of the RLHF methodology versus the supervised approach. In very simple terms, the lack of such a control study is leaving a good job in aligning language models?

Lack of ground truth for the comparison data: labelers can often disagree on the ranking of the model's outputs. Technically speaking, the risk is to add a *high potential variance* to the comparison data without any ground truth.

Human preferences are just not homogeneous: The RLHF method treats human preferences as if they were homogeneous and static. Assuming that all people share the same values would be an obvious stretch, at least on a large amount of topics of human knowledge. Some recent research is starting to tackle this open problem differently.

Prompt-stability testing for the reward model (RM): There seem to be no experiments investigating the sensitivity of the reward model syntactically different but are semantically equivalent, can the RM show significant differences in the ranking of the model outputs? In simpler terms, how much does the *quality of the prompt* matter for the RM?

Wireheading-type issues: In RL approaches, the model can sometimes learn to manipulate its own reward system to achieve a desired outcome, leading to an "over optimized policy". This can push the model recreating some patterns that for some unknown reason make the reward model score high (see Table 29 in this paper from OpenAI for an explicit example of this behavior in language modeling). ChatGPT puts a patch on this with the KL penalty term in the reward function. Note that one is trying to optimize the RM *input* (i.e. the PPO output) in order to improve its *output* (the reward score), *all the* *reference input* (the SFT output). More details on the limitations of this approach in this recent preprint.

Selected references for further reading

- The most relevant paper about the RLHF methodology used for ChatGPT is Training language models to follow instructions with human feedback, which in fact details a model called InstructGPT, referred to by OpenAI as a "sibling model" to ChatGPT.
- The paper Learning to summarize from Human Feedback describes RLHF in the context of text summarization.
- Proximal Policy Optimization: the PPO algorithm paper.

one of the earliest (Deep Learning) papers using human feedback in RL, in the context of Atari games.

- Alternatives to OpenAI's RLHF have been proposed by DeepMind in Sparrow and GopherCite papers.
- Anthropic has an open source repository (with accompanying paper) for RLHF.

Popular posts