What do LLMs learn from negative examples?

Anonymous ACL submission

Abstract

Large language models (LLMs) undergo a three-phase training process: unsupervised pre-training, supervised fine-tuning (SFT), and learning from human feedback (RLHF/DPO). Notably, it is during the final phase that these models are exposed to negative examples—incorrect, rejected, or suboptimal responses to queries. This paper delves into the role of negative examples in the training of LLMs, using a likelihood-ratio (Likra) model on multiple-choice question answering benchmarks to precisely manage the influence and the volume of negative examples. Our findings reveal three key insights: (1) During a critical phase in training, Likra with negative examples demonstrates a significantly larger improvement per training example compared to SFT using only positive examples. This leads to a sharp jump in the learning curve for Likra unlike the smooth and gradual improvement of SFT; (2) negative examples that are plausible but incorrect (near-misses) exert a greater influence; and (3) while training with positive examples fails to significantly decrease the likelihood of plausible but incorrect answers, training with negative examples more accurately identifies them. These results indicate a potentially significant role for negative examples in improving accuracy and reducing hallucinations for LLMs.

1 Introduction

Large language models are typically pre-trained on next word prediction over large collections of text, then fine-tuned on desired responses to user prompts. They only encounter negative examples in the final stage of their training in the form of false, undesirable, unsafe, or low-quality outputs. Techniques like reinforcement learning from human feedback (RLHF) (Ouyang et al., 2022) or direct preference optimization (DPO) (Rafailov et al., 2024) are used to train from human preference data that includes such negative examples. RLHF learns a reward function that imitates user preferences and uses reinforcement learning to align the generation process with these preferences. DPO bypasses reward learning and directly trains the model on pairs of good and bad outputs in the training set.

In this paper we focus on the contribution of negative examples for language model training and find that their impact is both qualitatively and quantitatively different from positive examples. Specifically we demonstrate that (1) during a critical phase in training, each additional negative example can improve the accuracy of a model $10\times$ more than each additional positive example resulting in a sharp jump in the learning curve, (2) near-miss negative examples, i.e. plausible sounding but incorrect outputs, are a lot more effective in training, and (3) models exposed to negative examples are a lot better at differentiating correct answers from plausible but incorrect ones at inference time.

Negative examples can help guide the learning process by providing explicit information about what the model should avoid generating or classifying as positive. This approach can enhance the model's discriminative ability or refine its generative output. The use of negative examples in machine learning goes back to Patrick Winston's pioneering work on the importance of “near-miss” examples in concept learning (Winston, 1970). More recently, techniques such as hard negative mining (Felzenszwalb et al., 2008) and adversarial example generation (Szegedy et al., 2013) have used near-miss negative examples to measure and improve the robustness of discriminative models. The incorporation of negative examples can also make generative models more discerning and controlled, reducing the likelihood of generating undesirable outputs. Contrastive divergence (Hinton, 2002), auto-encoders (Hinton and Salakhutdinov, 2006) and generative adversarial networks (Goodfellow et al., 2014) learn by contrasting real examples

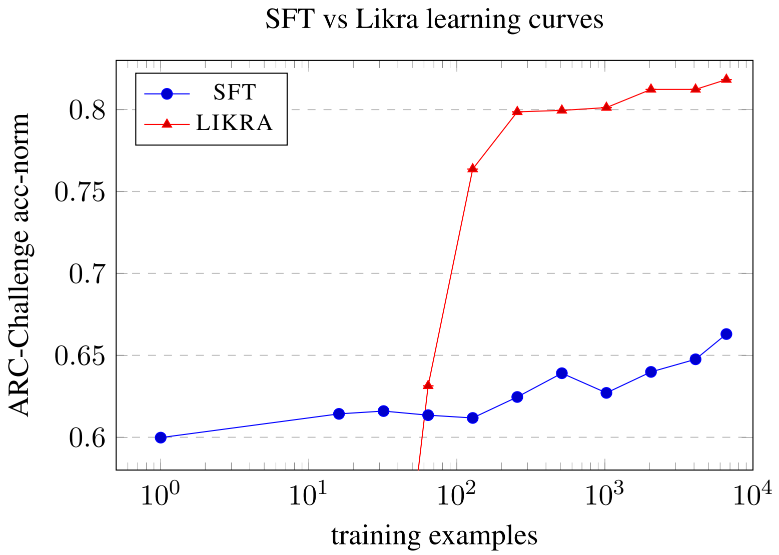

Figure 1: Comparison of learning curves for supervised fine-tuning (SFT) and likelihood-ratio (Likra) models for the ARC-Challenge benchmark (Clark et al., 2018) using Mistral-7B-v0.1 (Jiang et al., 2023) as a base model. The SFT model is the result of regular supervised fine-tuning using only correct question-answer pairs. The Likra model uses an equal number of incorrect question-answer pairs to train a negative head, uses the likelihood ratio of the SFT model and the negative head to decide on its answers.

with model-generated ones, unlikelihood training (Welleck et al., 2019) explicitly penalizes models for generating certain types of undesirable outputs, and noise-contrastive estimation (Gutmann and Hyvärinen, 2010) tackles modeling intractable distributions using negative "noise" data.

We use a likelihood-ratio model to study the impact of negative examples. Likelihood-ratio models have long been used in classification tasks where the likelihoods from multiple models are compared and used as a decision criterion or to identify out of distribution data: Naive Bayes classifiers, Gaussian Mixture Models are basic examples. In generative modeling, noise-contrastive estimation contrasts the target distribution with a "noise" distribution to construct a tractable loss function and unlikelihood training combines two likelihood terms to prevent generation of repetitive and dull text.

We chose to train two independent models (in practice two LoRA adapters on a single foundation model), one on positive examples the other on negative examples, and use their likelihood-ratio (hence the name Likra) during inference to isolate and quantify the impact of negative examples. This allows us to vary the number of positive and negative examples independently during training and control the relative weights of the two models during inference (unlike e.g. a single DPO model). The downside is that the resulting Likra model cannot be easily used for generation, so all our testing is done on multiple-choice benchmarks.

Figure 1 illustrates a key result of our work: when trained on an equal number of positive and negative examples for a multiple-choice task, Likra exhibits a sharp, step-function-like increase in accuracy after a few hundred examples. This behavior contrasts sharply with the smooth and gradual learning curve observed in supervised fine-tuning. Details for this result are provided in Section 3.

After formally describing Likra in Section 2, and presenting our main results in Section 3, we run a series of ablation experiments to help understand this strange contribution of negative examples in Section 4. Section 5 summarizes our findings.

2 Likra: the likelihood-ratio model

In our experiments we use a likelihood-ratio language model (Likra) to isolate and compare the contributions of positive and negative examples during training. The model consists of two heads: the positive head is trained on correct question-answer pairs, and the negative head is trained on incorrect/undesirable question-answer pairs. Each head is trained to maximize the conditional log-likelihood of the corresponding answer given the question:

$$\mathcal{L}_{\text{MLE}}(p_\theta, \mathcal{D}) = \sum_{q,a\in\mathcal{D}} \sum_{t=1}^{|a|} \log p_\theta(a_t|q, a_{<t}) \quad (1)$$

where $\mathcal{L}_{\text{MLE}}$ is the likelihood, θ represents model parameters, $\mathcal{D}$ is the training data, q, a are question-answer pairs (correct answer for the positive head, incorrect answer for the negative head), a_t is the t'th token of the answer.

Each head is independently trained starting from a base pre-trained language model, the positive head giving higher likelihood $\mathcal{L}^+$ to correct answers and the negative head giving higher likelihood $\mathcal{L}^-$ to incorrect answers. During inference time we use the log likelihood ratio $\mathcal{L}^+ - \mathcal{L}^-$ to score answer candidates.

3 Experiments

In this section we lay out the empirical evidence for the main thesis of this paper: starting from a pre-trained language model, during a critical phase of the training, negative examples (questions paired with incorrect answers) have a significantly larger impact on accuracy than positive examples (questions paired with correct answers), which leads to a sharp jump in the learning curve unlike the smooth and gradual improvement of SFT.

We start with a fairly standard supervised fine-tuning example where a pre-trained base language model is fine-tuned with correct question-answer pairs from the training set of a standard multiple-choice benchmark. We call this the SFT model. We then start with the same base model / dataset and train a negative model by fine-tuning with incorrect question-answer pairs. The Likra model chooses answers based on the likelihood ratio of these two models and demonstrates the step-function-like jump in accuracy in its training curve and significantly outperforms positive-example-only trained SFT model.

For brevity, the discussion below uses the results from the Mistral-7B-v0.1 (Jiang et al., 2023) base model and the ARC (Clark et al., 2018) benchmark unless otherwise noted. Experiments with other base models and multiple-choice benchmarks show similar results and are summarized at the end of the section.

3.1 Supervised fine-tuning

In this section we start with an example of supervised fine-tuning (training a base model with correct question-answer pairs) resulting in modest gains in accuracy on a multiple-choice benchmark.

To construct the training set we used the AI2 Reasoning Challenge (ARC) benchmark, a set of grade-school level multiple-choice science questions (Clark et al., 2018) such as:

```
What can a flower become?
(A) a fruit
(B) a leaf
(C) a stem
(D) a branch

Which substance is a compound?
(A) sodium
(B) chlorine
(C) table salt
(D) salt water
```

We used the LM-EVALUATION-HARNESS (Gao et al., 2023) for evaluation, which prepends 25 few-shot examples (random correct question-answer pairs) to each test question and compares the per-character likelihoods of different answer choices. We excluded the 1172 questions used by LM-EVALUATION-HARNESS from our training set and we paired the remaining 6615 questions with their correct answers for supervised fine-tuning:

```
Question: What can a flower become?
Answer: a fruit

Question: Which substance is a compound?
Answer: table salt
```

We used Mistral-7B-v0.1 as a base model (Jiang et al., 2023) for supervised fine-tuning. The training was performed with zero-shot examples (no extra questions in the context) optimizing the likelihood of the correct answer conditional on the question (the question logits were ignored). We trained a LoRA adapter (Hu et al., 2021) only for a single epoch (more epochs did not help) using batch size 8 and the Adam optimizer (Kingma and Ba, 2014) with learning rate 10^{-4}.

The SFT learning curve in Figure 1 was obtained by training the base model on random samples with 0 to 6615 (full dataset) correct question-answer pairs from the training set. It shows a modest increase in accuracy (60% to 66%) as expected using in-domain training examples. Figure 2 compares the average per-character likelihood of the

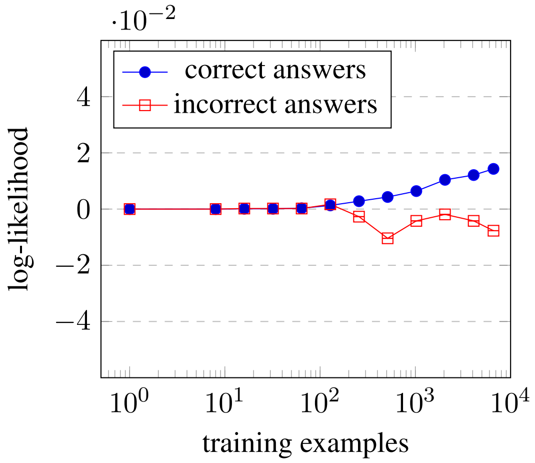

Figure 2: Likelihood of the correct answer vs the most likely incorrect answer during training.

correct answers with the most likely incorrect answer throughout training. Even though the correct answers seem to increase in likelihood, the incorrect answers do not seem to be sharply distinguished. In Section 4.4 we will look at the allocation of the model's probability mass in more detail and show that a model trained with negative examples can distinguish correct from incorrect more sharply. In the next section we look at how the model can learn more from incorrect answers by using them as negative examples.

3.2 Likra with negative examples

In this section we show that negative examples (incorrect question-answer pairs) have a *significantly* larger effect on final accuracy compared to positive examples (correct question-answer pairs). We experimented with a Likra model rather than RLHF or DPO because this made it easier to isolate and control the contribution of negative examples. The results show that doubling the number of positive examples increases the accuracy of the SFT model by less than 1%, whereas doubling the number of negative examples can increase the accuracy of the Likra model by more than 10% during the critical phase of training.

To generate a training set of negative examples we paired each question in the ARC training set with an incorrect answer chosen randomly from the multiple-choice options, e.g.:

```
Question: What can a flower become?
Answer: a leaf

Question: Which substance is a compound?
Answer: salt water
```

We used Mistral-7B-v0.1 as a base model for Likra. To train the negative head we followed a procedure similar to SFT training except for using negative examples.

Figure 1 compares the SFT model which has been fine-tuned with only positive examples with a Likra model that uses the same SFT model as its positive head in addition to a negative head fine-tuned with incorrect question-answer pairs. We observe that starting from a well pre-trained base model, the contribution of each negative example to the final accuracy far exceeds the contribution of each positive example in the critical training phase at 64-256 examples. Increasing the number of negative training examples from 64 to 128 (only 8 extra updates with a batch size of 8) adds nearly 15% accuracy, whereas the SFT model averages less than 1% improvement per doubling of positive examples.

It is unlikely for the model to learn much new information from just a few wrong answers, instead the negative examples seem to quite rapidly unlock latent knowledge that already exists in the pretrained model.

3.3 Other base models and benchmarks

Table 1: Final accuracy on ARC-Challenge and HellaSwag. No superscript in the model name indicates the base model, $^+$ indicates supervised fine-tuning with positive examples, $^-$ indicates the Likra model trained with both positive and negative examples.

Model	ARC	HellaSwag
Mistral-7B-v0.1	.5998	.8323
Mistral-7B-v0.1$^+$	.6630	.8468
Mistral-7B-v0.1$^-$	**.8123**	**.9633**
Mistral-7B-Instruct-v0.3	.6365	.8463
Mistral-7B-Instruct-v0.3$^+$	.6408	.8360
Mistral-7B-Instruct-v0.3$^-$	**.8063**	**.9569**
Llama-3.2-3B-Instruct	.5222	.7312
Llama-3.2-3B-Instruct$^+$	.5486	.7254
Llama-3.2-3B-Instruct$^-$	**.7321**	**.9071**

In order to test the generality of our results we experimented with two benchmarks and three models. Table 1 summarizes the results.

ARC-Challenge benchmark (Clark et al., 2018) is a set of grade-school level English multiple-choice science questions with 6615 training and 1172 test instances. The fine-tuning for ARC takes around 5 minutes on 1×A40 and the evaluation takes around 15 minutes on 8×A40. HellaSwag (Zellers et al., 2019) is a set of multiple-choice English text completion tasks based on video descriptions and how-to manuals with 39905 training and 10042 test instances. The fine-tuning for Hellaswag takes around 1 hour on 1×A40 and the evaluation takes around 30 minutes on 8×A40. Other than both being multiple-choice tasks, ARC and Hellaswag require quite different types of knowledge demonstrating some domain independence for our findings. Mistral (Jiang et al., 2023) and Llama (Grattafiori et al., 2024) are open source foundation models[1]. In each case we see a large jump in accuracy with the Likra model.

4 Analysis

In this section we probe the training process more deeply to understand the role of negative examples in boosting model accuracy. First we change the ratio of the positive and negative examples during training and the weight of the positive and negative heads during inference. The results ensure us that negative examples have a significantly stronger effect compared to positive examples. Then we categorize negative examples as incorrect, irrelevant, or unrelated and train different models with each category. The results show that the more plausible incorrect answers increase model accuracy the most. Finally we look at how likelihoods of different answer types (correct, incorrect, irrelevant, unrelated) evolve during the training process. The results show that the negative head learns to sharply distinguish plausible but incorrect answers from correct ones, whereas the positive head assigns them closer likelihoods.

4.1 Do we even need positive examples?

The Likra model allows us to vary the number of positive and negative examples during training independently. Given the large impact of negative examples on model accuracy, we asked if Likra would work without positive examples. In Figure 3 SFT-Likra uses the likelihood ratio of the SFT model and the negative model (same as Figure 1), Base-Likra uses the likelihood ratio of the base model and the negative model. Effectively Base-Likra only uses negative examples for fine-tuning. The resulting learning curve of the two Likra models are fairly similar: they both have the step-function like accuracy increase at a few hundred examples and they both significantly outperform SFT.

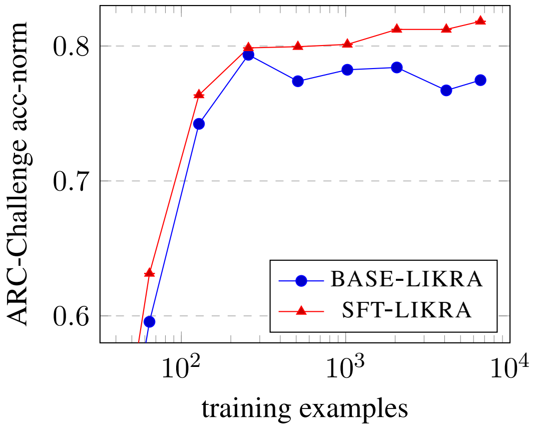

Figure 3: Likra with (SFT) and without (Base) positive examples.

4.2 Varying the weight of the negative head

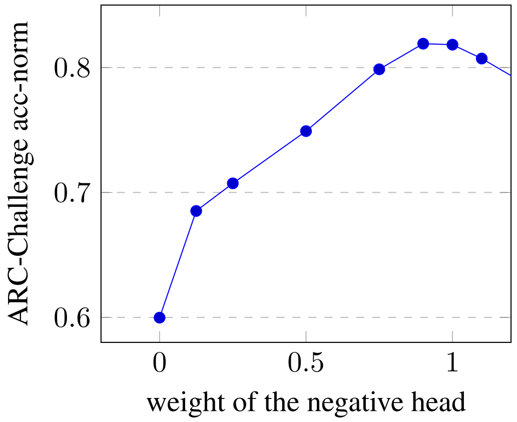

Figure 4: Changing the weight of the negative head.

The Likra model allows us to vary the relative weight of the positive head and the negative head during inference. Figure 4 plots the result of changing the weight of the negative head during inference, i.e. score = $\mathcal{L}^+ - \text{weight} \times \mathcal{L}^-$. It shows that the accuracy increases as the weight of the negative head is increased and peaks around 0.9-1.0.

4.3 Near-miss negative examples teach more

The Likra model works by contrasting the conditional likelihood given to an answer by a pretrained (or SFT trained) positive head and a negative head trained on incorrect question-answer pairs.

[1] These resources have the following licenses: ARC-Challenge: CC BY-SA 4.0; HellaSwag: MIT; Mistral: Apache 2.0; Llama: Llama 3.2 License. All models allow usage and modification, and none of the datasets contain personal information that require anonymization.

The results so far demonstrated the importance of training on negative examples. In this section we ask whether the plausibility of these incorrect answers matter during training, i.e. can we construct negative examples by answering questions with irrelevant text, or do the false answers have to be plausible? We conclude that even though all negative training can be beneficial, the near-miss negative examples consisting of plausible sounding but incorrect question-answer pairs work best.

We generated three different training sets of negative answers by pairing each question in the ARC training set with a false answer chosen from (i) multiple-choice options for that question, (ii) random false answer for a different ARC question, and (iii) random false answer from an unrelated benchmark (we tried non-science-related tests from MMLU (Hendrycks et al., 2020) and HellaSwag (Zellers et al., 2019) with similar results). For example:

```
Question: Which substance is a compound?
Answer: salt water
(incorrect from the same question)

Question: Which substance is a compound?
Answer: reduce the energy requirements
(irrelevant from the same test (ARC))

Question: Which substance is a compound?
Answer: is playing the piano
(unrelated from another test (HellaSwag))
```

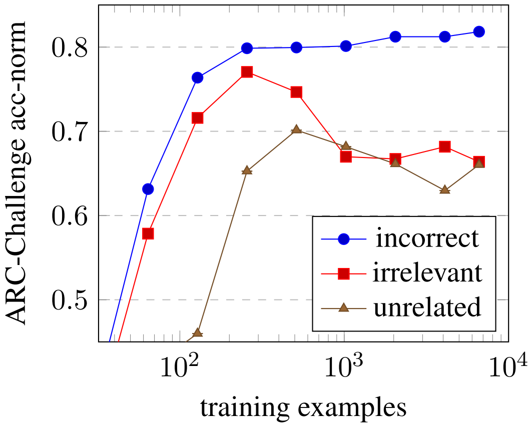

Figure 5: Training with different negative examples.

Figure 5 compares the performance of training the negative head with these three types of false answers (using the SFT model as the positive head). One surprising observation is that even when the false answers are completely unrelated random text (e.g. HellaSwag answers to ARC science questions) they are beneficial (the Likra model reaches 70% accuracy outperforming the SFT model). Maybe less surprising is the finding that the more plausible false answers are the more beneficial they seem to be, as suggested by Patrick Winston's pioneering observation of the importance of near-miss negative examples for learning (Winston, 1970).

4.4 Where does the probability mass go?

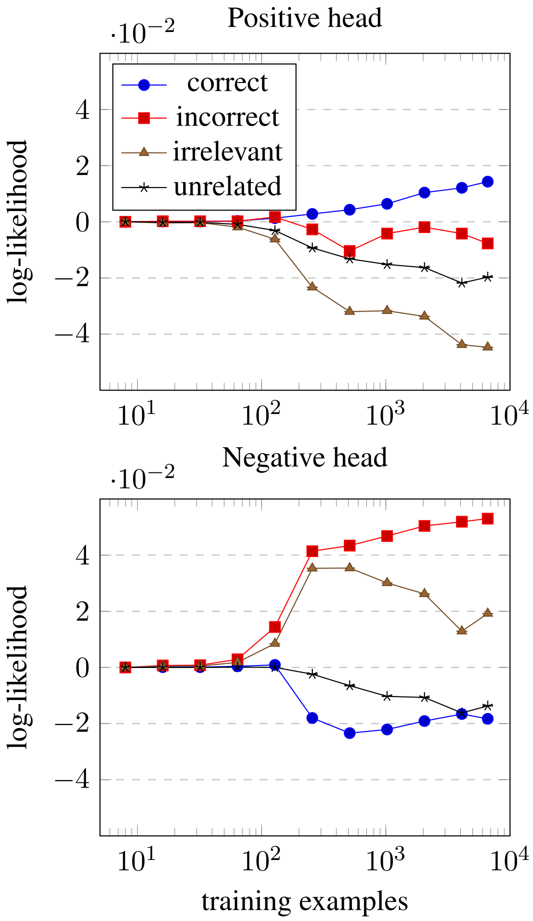

Figure 6: Likelihood assigned to answer types.

In this section we analyze how the probability mass shifts between different types of text during training to explain the performance of the Likra model. Our results show that the negative head of the Likra model becomes very good at identifying near-miss negative examples (plausible wrong answers) which compensates for the main weakness of the positive head.

To measure how the probability mass shifts between different regions of the text space during training, we paired *test set* questions with different types of answers (as opposed training set questions, like we did in the last section): the correct answer, the most likely incorrect answer, irrelevant text from the same benchmark (ARC), and unrelated text from a different benchmark (HellaSwag). We took the positive head (trained with correct answers), and the negative head (trained with random incorrect answers) at various points along their learning curve and evaluated their conditional likelihoods for these different types of text.

Figure 6 shows the absolute change in per-character log-likelihood for different types of text. Text unrelated to the subject domain becomes less likely for both positive and negative heads. The positive head increases the likelihood of the correct answers as expected and decreases the likelihood of irrelevant/incorrect answers, however the likelihood of incorrect answers does not seem to decrease by much. The negative head increases the likelihood of incorrect answers significantly, irrelevant answers to a lesser extent, and decreases the likelihood of the correct answers. Even though the negative head has never seen a correct answer during fine-tuning it is able to distinguish them from plausible incorrect answers.

When we take the difference of log likelihoods for inference, the biggest impact of the negative head turns out to be significantly decreasing the likelihood of incorrect answers. It seems that pre-trained language models can learn significantly more from plausible sounding incorrect answers, i.e. near-miss negative examples, than correct answers whose likelihoods are already relatively high in the base model.

5 Discussion

We still find it incredible that a few hundred negative examples improve the answer accuracy of a pre-trained language model significantly more than thousands of positive examples albeit in a restricted domain. It seems clear that *wrong* answers to a few hundred training questions cannot give the model much missing information about the test questions. Thus the knowledge to answer these test questions correctly must already reside in the pre-trained model but obfuscated by the probability mass given to other plausible sounding answers. Training with negative examples seems to flip a switch that causes the model to sharply distinguish factually accurate answers from plausible sounding ones. This supports a version of the "Superficial Alignment Hypothesis" (Zhou et al., 2024): A model's knowledge and capabilities are learnt almost entirely during pretraining, and alignment teaches it not only format and style, but also preference for factual accuracy.

6 Limitations

We presented a method that improves the factual accuracy of large language models, however it does not guarantee that the resulting models will always generate or choose factually correct answers. The base models we use, as well as their fine-tuned versions may in some instances produce inaccurate, biased or other objectionable responses to user prompts. Our fine-tuning and evaluation only used English benchmarks. The Likra model specifically trains and uses a negative head specialized in recognizing factually inaccurate answers, however the two-head model structure makes it challenging to generate text. We suggest using Likra models to evaluate potential answers for accuracy, hallucination detection, or in multiple-choice testing scenarios.

References

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

Pedro Felzenszwalb, David McAllester, and Deva Ramanan. 2008. A discriminatively trained, multiscale, deformable part model. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. 2023. A framework for few-shot language model evaluation.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. *Advances in neural information processing systems*, 27.

Aaron Grattafiori et al. 2024. The Llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Michael Gutmann and Aapo Hyvärinen. 2010. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 297–304. JMLR Workshop and Conference Proceedings.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2020. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*.

Geoffrey E Hinton. 2002. Training products of experts by minimizing contrastive divergence. *Neural computation*, 14(8):1771–1800.

Geoffrey E Hinton and Ruslan R Salakhutdinov. 2006. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2024. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2013. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*.

Sean Welleck, Ilia Kulikov, Stephen Roller, Emily Dinan, Kyunghyun Cho, and Jason Weston. 2019. Neural text generation with unlikelihood training. *arXiv preprint arXiv:1908.04319*.

Patrick Henry Winston. 1970. *Learning structural descriptions from examples*. Ph.d. thesis, MIT.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srinivasan Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, et al. 2024. Lima: Less is more for alignment. *Advances in Neural Information Processing Systems*, 36.