

LEARNING FROM REWARDS IN TEXT GENERATION

by

Yuanzhe Pang

A DISSERTATION SUBMITTED IN PARTIAL FULFILLMENT

OF THE REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

DEPARTMENT OF COMPUTER SCIENCE

NEW YORK UNIVERSITY

SEPTEMBER 2024

Dr. He He

Dr. Kyunghyun Cho

© YUANZHE PANG

ALL RIGHTS RESERVED, 2024

ACKNOWLEDGMENTS

I am eternally grateful to my advisor, He He, for taking a chance on me and having me as her first Ph.D. student. I am grateful for her guidance, and I am privileged to have shared this academic journey as we both started at NYU in 2019. I have seen first-hand on how He has built her research group from the ground up. Now, the group is lively and flourishing.

He's mentorship never consists of dictates or deadlines: I would have my ideas, and she would always offer incredibly sharp insights that would help refine my project. The beauty in her advice lies in the freedom she gave me to chart my own course, even when it may mean navigating detours and overcoming setbacks. Looking back, I have realized that her patience and her trust have been invaluable gifts — she made me a self-assured and independent researcher.

Her advice also has a profound impact on my approach to research in general. When choosing problems to tackle, instead of diving right into the problem, I have learned to reconsider whether the best-case outcome would truly be meaningful and useful, and whether it would support my long-term research question. In addition, I have also learned about the profound importance of effective communication from He, whether it is through writing, presentation, or everyday conversation.

My co-advisor Kyunghyun Cho wears many hats: He is a professor at NYU and a director at Genentech; he has been the program chair or editor for almost every top ML venue; he has influential research covering just about every topic related to ML. How does he have time for all these? I have always been amazed by how quickly Cho grasps things. I always walk out of every

meeting in awe, with multiple theory-grounded ideas that I never thought of before. In fact, a major lesson I took from him is to carefully formulate modern ML problems, and there are always inspirations from theories. Relatedly, I have also learned to think critically on everything, instead of blindly trusting an idea simply because of its popularity.

Cho is an effective advisor. In my first few meetings with him, he would suddenly halt his explanations, his piercing gaze demanding: “Richard, tell me everything I told you in the past ten minutes.” Or he would catch me nodding along and say “define MCMC” (which was mentioned five seconds ago, but I nodded so I must know it well, right?). I quickly realized that Cho was an excellent anti-BS detector and I quickly started to be perfectly frank with him (and others) on what I know and what I don’t.

The single thing I appreciate most about Cho is that when there is a relevant topic that comes up in research which we both do not know much about (e.g., topics in reinforcement learning, or topics in uncertainty quantification), instead of shying away, we learn together. In fact we have spent lots of time reading textbooks together like travelers exploring a new landscape, even though Cho is a senior professor and very experienced researcher. Once he even contributed lots of code to a group project as we were stuck with the toolkit. He instilled in me the courage to explore uncharted fields.

I have had the privilege of working with Jason Weston, my manager at Meta FAIR and essentially my third albeit unofficial advisor since Fall 2022. Jason is not only a brilliant mind in the AI community and has contributed so much in the past three decades, but also a renowned artist in NYC’s nightlife scene. His extraordinarily contagious energy has constantly inspired and motivated me. His exceptional sharpness is evident in his work: he predicted the present NLP world almost two decades ago when deep learning was not even treated seriously. I am especially impressed by his excellent memories: Jason seems to remember all my past updates, even the most minute details like hyperparameters which I might have mentioned months ago.

One thing I am most grateful for is that he has taught me to approach problems with a no-

nonsense pragmatism — the least flowery but most straightforward ways are often the most effective. Jason never runs out of ideas that support his long-term research agenda, and I have also learned to identify the biggest research opportunities. I cannot emphasize how pivotal being on Jason’s RAM (reasoning, alignment, memory) team has been to my research career.

Next, I have to emphasize that my undergraduate advisor Kevin Gimpel’s unconditional support laid the foundation for my entire career. Despite his hectic schedule as a professor, he devoted at least an hour to our weekly one-on-one meetings while I was just an undergrad at the University of Chicago who just finished machine learning coursework. He treated me like a peer, and experiencing every stage of a research project in our two-author project has given me confidence for independent research — I experienced for the first time that in ML research, ambitious goals were actually within reach. My career exists at all because of Kevin, and I am committed to paying it forward by trying hard to guide the next generations of NLP/ML practitioners.

I would also like to express my gratitude to the other dissertation committee members: Mengye Ren and Ankur P. Parikh. He, Cho, Jason, Mengye, and Ankur: thank you for all the feedback, the insights, and your time. The committee means a lot to me personally. This dissertation is composed of joint work with my mentors mentioned above as well as Weizhe Yuan, Sainbayar Sukhbaatar, Jing Xu, Stephen Roller, and Xian Li; thank you for the wonderful collaboration. I am also grateful to Sam Bowman for serving on prior committees and trusting me to lead the QuALITY project, and I have been very inspired by Sam who has led multiple NLP research trends.

To Angelica Chen, Nitish Joshi, Vishakh Padmakumar, and Jason Phang: thank you for all the hangouts, brainstorming, as well as impromptu post-midnight conversations; your support has given me so much energy. To members in He’s group including Nicholas Lourie, Abulhair Saparov, Jane Pan, Vishakh Padmakumar, Nitish Joshi, Yanda Chen, Trieu Trinh, Saranya Venkatraman, Chen Zhao, Xiang Pan, and Shi Feng: I have learned so much from each of you in our weekly group meetings and various socials; it has been amazing to see our group getting increasingly vibrant. To Ilia Kulikov, Jason Lee, and Sean Welleck: thank you for all the encouragement, reassurance,

and mentorship when I just joined NYU. Jason Weston’s RAM group at FAIR has been my second home in my Ph.D., and I need to acknowledge that I am extremely fortunate to have been part of RAM in the past two years. I am thankful for my other main mentors at FAIR: Stephen Roller who always has innovative ideas and Sainbayar Sukhbaatar who always has extreme attention to detail and probes each question deeply. RAM has been an eye-opening experience and has shaped who I am as a researcher.

Finally, I would like to express my gratitude to all my mentors, collaborators, and other colleagues I have frequently interacted with (alphabetical order): my NYU mentors/collaborators/colleagues (Sam Bowman, Kianté Brantley, Iacer Calixto, Jon Ander Campos, Angelica Chen, Kyunghyun Cho, Carles Domingo-Enrich, Shi Feng, He He, Betty Li Hou, Phu Mon Htut, Michael Hu, William Huang, Sébastien Jean, Lavender Jiang, Nitish Joshi, Aishwarya Kamath, Jaedeok Kim, Najoung Kim, Ilia Kulikov, Elman Mansimov, Will Merrill, Dhara Mungra, Jason Lee, Haokun Liu, Johnny Ma, Julian Michael, Aaron Mueller, Nikita Nangia, Vishakh Padmakumar, Jane Pan, Alicia Parrish, Ethan Perez, Jackson Petty, Jonas Pfeiffer, Jason Phang, Ofir Press, Yada Pruksachatkun, David Rein, Mengye Ren, Abulhair Saparov, Naomi Saphra, Vlad Sobal, Asa Cooper Stickland, Jana Thompson, Clara Vania, Katharina von der Wense, Alex Wang, Alex Warstadt, Sean Welleck, Weizhe Yuan, Lily Zhang, and Chen Zhao), my Meta mentors/collaborators/colleagues (Florian Bordes, Mingda Chen, Shehzaad Dhuliawala, Maryam Fazel-Zarandi, Olga Golovneva, Mojtaba Komeili, Ilia Kulikov, Jack Lanchantin, Yann LeCun, Xian Li, Stephen Roller, Sainbayar Sukhbaatar, Tianlu Wang, Jason Weston, Jing Xu, Jane Yu, Ping Yu, and Weizhe Yuan), my Google and Google DeepMind mentors/collaborators/colleagues (Le Hou, Mehran Kazemi, Adam Lelkes, Ankur P. Parikh, Thibault Sellam, Xinying Song, Vinh Q. Tran, Yuexin Wu, Cong Yu, Denny Zhou, and Tianyi Zhou), my UChicago/TTIC mentors/collaborators/colleagues (Mingda Chen, Zewei Chu, Xiaoan Ding, Sadaoki Furui, Lingyu Gao, Kevin Gimpel, David McAllester, Freda Shi, Karl Stratos, Shubham Toshniwal, and Lifu Tu), as well as other colleagues whom I have had the privilege to interact with on multiple occasions (Eneko Agirre, Jiatao Gu, Mohit Iyyer, Kalpesh Krishna, and

Hongyuan Mei). I would also like to thank Thomas Wies, Santiago Pizzini, Shenglong Wang, and staff members at the Center for Data Science for their pivotal support during my time at NYU.

I am grateful for all the personal support I have received throughout the past few years: thank you, Anya D., Cam D., Mireille F., Ivan G., Mike G., Michelle L., Lindsay N., Alex S., Jason S., Kewei S., Sarah S., Angel W., Xinyu W., and many others whom I have already mentioned above as well as those I might have missed. This journey would not be possible without my partner, Caiqin, who keeps inspiring me every day and stands by me through my highs and lows; we have been building our post-college lives together starting with just a few suitcases, and I look forward to our next adventures. Finally, the unconditional support and belief from my family (in particular, my parents and my grandpa) have been my major source of strength.

ABSTRACT

The progress in text generation comes from every stage in the pipeline: problem definition, data curation, learning, decoding, and evaluation. This dissertation focuses on learning. There is a mismatch between traditional training objectives and evaluation objectives: regular maximum likelihood estimation (MLE) tries to minimize the cross-entropy loss with respect to each sample in the dataset, but the downstream evaluation is often based on a reward that scores the compatibility of the input-output pair (e.g., human judgments of the output). We aim to bridge this gap by optimizing for the reward of the generated text directly.

The dissertation is composed of the following components. (1) Rewards could be expensive to obtain. To tackle this challenge in the social dialogue setting, we extract implicit signals from deployment data without extra human annotations. To generate a bot response, we optimize for predicted positive reactions from future human turns. Our approaches improve the bot responses overall, but some proxy signals can lead to more generations with undesirable properties. (2) The model could make slow or no progress in learning, and one idea is to obtain denser and higher-quality rewards. In neural machine translation, we get inspired by noisy channel decoding which has a long history, based on which we define a reward function. The byproduct is that we are able to increase decoding speed significantly while ensuring similar translation quality. (3) Another way to make progress in learning is to innovate on training algorithms instead. We set the rewards to be based on the simple exact match of generations and references, but algorithm-wise we explore the extreme case where we do not deviate too far from references by

framing text generation as an offline reinforcement learning problem. We propose generation by off-policy learning from demonstrations (GOLD): an algorithm that learns from demonstrations by importance weighting. We show that models trained by GOLD outperform those trained by MLE and policy gradient on a range of tasks. (4) We show that we do not need to rely on reinforcement learning, using reasoning tasks (e.g., math, science, commonsense) as a testbed. We develop an approach called iterative reasoning preference optimization (IRPO) that optimizes for winning vs. losing reasoning chain-of-thoughts, using modified direct preference optimization as the criteria. IRPO results in markedly increased accuracies compared to a range of baselines.

Finally, we discuss the future direction of using large language models as rewards. We briefly mention the initial promise given by our work on self-rewarding language models using an iterative direct preference optimization learning criteria similar to IRPO; the discussion is then followed by the corresponding challenges and next steps. Further, an additional way to improve evaluation capabilities could rely on human–AI collaboration approaches where the ultimate goal is that the final performance far exceeds that of humans on their own or that of AIs on their own.

CONTENTS

Acknowledgments	iii
Abstract	viii
List of Figures	xv
List of Tables	xix
1 Introduction	1
2 Preliminaries	9
2.1 Conditional Text Generation	9
2.2 Learning vs. Evaluation and the Need for Reward	10
2.3 Reinforcement Learning and Text Generation	11
2.3.1 Policy Optimization Algorithms	12
2.3.2 Q learning	13
2.3.3 General RLHF Framework in Training Language Models	15
2.4 Non-RL Algorithms in Text Generation	16
3 An Example of Inexpensively Extracting Rewards in Dialogue Systems: Leveraging Implicit Feedback from Deployment Data	19
3.1 Introduction	19

3.2	Related Work	21
3.3	Approach	22
3.3.1	Implicit Feedback Signals	22
3.3.2	Models Using Implicit Signals	24
3.4	Experiments and Results	24
3.4.1	Experimental Setup	24
3.4.2	Evaluation Methods	25
3.4.3	Results	26
3.5	Conclusion	27
3.6	Limitations	28
4	Rewards Can Be Based on Classic NLP Algorithms: Amortized Noisy Channel	
	Neural Machine Translation	29
4.1	Introduction	29
4.2	Background	31
4.2.1	Neural Machine Translation	31
4.2.2	Beam Search and Rerank (BSR)	32
4.3	Amortized Noisy-Channel NMT	32
4.3.1	Approach 1: Knowledge Distillation (KD)	33
4.3.2	Approach 2: 1-Step-Deviation Imitation Learning (IL)	34
4.3.3	Approach 3: Q Learning	35
4.4	Experimental Setup	38
4.4.1	Tasks and Models	38
4.4.2	Hyperparameters	40
4.5	Results	41
4.5.1	Preliminary Analysis	41

4.5.2	Analysis of Translations	43
4.5.3	Further Analysis	45
4.6	Related Work	47
4.7	Conclusion	48
5	Rewards Can Be Simple and We Can Innovate on Learning Algorithms: Text Generation by Learning from Demonstrations	49
5.1	Introduction	49
5.2	From MLE to RL Framework	51
5.3	Approach	52
5.3.1	Off-Policy Policy Gradient	52
5.3.2	Reward	56
5.3.3	The GOLD Algorithm	57
5.4	Experiments	58
5.4.1	Setup	58
5.4.2	Results and Analysis	59
5.4.3	Comparison with On-Policy Training	64
5.4.4	Discussion on Generation Diversity	65
5.5	Related Work	67
5.6	Conclusion	68
5.7	Developments since Publication	68
6	We Do Not Necessarily Need RL: Iterative Reasoning Preference Optimization	71
6.1	Introduction	71
6.2	Iterative Reasoning Preference Optimization	73
6.3	Experiments	77
6.3.1	Math Word Problems: GSM8K	77

6.3.2	ARC-Challenge Task	83
6.3.3	MATH Task	85
6.4	Related Work	86
6.5	Conclusion	88
6.6	Limitations	88
6.7	Developments since Publication	89
7	Conclusion and Future Directions	90
7.1	Rewards Can Be Based on LLM Outputs	91
7.1.1	Self-Rewarding Language Models	91
7.1.2	Extending IRPO to Unsupervised Tasks	93
7.2	Human-LLM Collaboration	94
A	Appendix for Chapter 3	96
A.1	Other Signals	96
A.2	Additional Info on Experimental Setup	97
A.3	Additional Info on Evaluation	98
A.3.1	Crowdworker Evaluation of Pairwise Comparison	98
A.3.2	Additional Results to Complement Table 3.1	99
A.3.3	Agreement of Expert, Annotator, and LLM Evaluation of Pairwise Com- parison	100
A.3.4	Prompts of LLM-Evaluation	101
B	Appendix for Chapter 4	118
B.1	More Information on Q learning for Amortized Noisy Channel NMT	118
B.1.1	Details on trajectories	118
B.2	More Discussion on Experiments	119

B.3	Ethical Considerations	120
C	Appendix for Chapter 5	122
C.1	Practical Setup and Implementation	122
C.1.1	Tasks and Datasets	122
C.1.2	Model Architectures	123
C.1.3	More on Reproducibility	124
C.1.4	More Discussion on Approximations	126
C.1.5	Details on On-Policy Experiments	126
C.2	More on Results	127
C.2.1	Lead-3 Baselines for Summarization	127
C.2.2	Performance with Transformer Architectures	127
C.2.3	More on Exposure Bias	128
C.2.4	Examples	130
C.3	Human Evaluations	133
C.3.1	Pairwise Comparison	133
C.3.2	NQG Rating	133
D	Appendix for Chapter 6	136
D.1	More Details on Experimental Setup	136
D.1.1	Prompts	136
	Bibliography	138

LIST OF FIGURES

3.1	Overview of the approach. Implicit signals are extracted from conversations, such as whether future human turns are long or short, or joyful or not. We train a binary classifier to predict whether the bot turn is “good” given the conversation history and the bot turn, and we leverage the classifier at bot’s test-time. We study various kinds of implicit signal in this work (Section 3.3).	20
4.1	Average length bucketed by length of the source sentence. The five buckets contain 453, 877, 376, 92, 26 sentences, respectively. The six systems are KD, IL, Q learning, beam search by p_f , BSR, and reference translations, respectively. In the longest length bucket, Q learning produces translations that are longer than translations by other systems.	44
4.2	Repetition rate (“token rep”) bucketed by length of the source sentence. The five buckets contain 453, 877, 376, 92, 26 sentences, respectively. N.B.: In the last bucket, “token rep” for Q-generated translations is around 0.31, and the bar is truncated.	44
5.1	Histograms of token-level NLL loss using standard models on NQG and CNN/DM dev sets. MLE learns high-recall models whose loss distribution is spread out; GOLD learns high-precision models whose loss distribution is concentrated on near-zero losses.	61

5.2	Left: Avg human ratings vs. generation length, on 736 NQG samples. (Colored regions: 95% confidence interval.) Each data point has ≥ 30 annotations. The quality of long generations from MLE-trained model drops heavily, but stays stable across lengths for GOLD-s generations. Right: Avg NLL loss of t -th token given the <i>gold</i> prefix tokens vs. time-step t , on NQG dev set. Without exposure bias, NLL loss stays stable across lengths.	63
6.1	Iterative Reasoning Preference Optimization. Our iterative preference optimization method consists of two steps: (i) <i>Chain-of-Thought & Answer Generation</i> : training prompts are used to generate candidate reasoning steps and answers from model M_t , and then the answers are evaluated for correctness by a given reward model. (ii) <i>Preference Optimization</i> : preference pairs are selected from the generated data, which are used for training via a DPO+NLL objective, resulting in model M_{t+1} . This whole procedure is then iterated resulting in improved reasoning ability on the next iteration, until performance saturates.	73
6.2	Effect of SFT training. (a) Although SFT training (solid green) is on chosen sequences (D_0^{pairs} , from iterative RPO iteration 1) only, the rejected sequence log probabilities (dotted green) also increase and are close to the chosen sequence probabilities. In contrast, our DPO+NLL training (blue) manages to decrease the rejected probabilities while increasing the chosen probabilities. This observation could potentially help explain why SFT-only performance lags significantly behind Iterative RPO Iteration 1 performance. (b) We show a similar plot but where SFT is trained on gold (dataset-provided) CoTs. Chosen and rejected sequence probabilities (which are from D_0^{pairs}) are still close to each other, but with a slightly bigger gap. Another observation is that the chosen sequence probabilities barely increase.	78

6.3	Effect of NLL loss term on DPO training for GSM8K. In our GSM8K experiments we observe the log probability of chosen sequences in standard DPO without NLL loss (solid orange) decreases over training steps, especially if the model is initialized from SFT training on chosen sequences (right). However, they <i>increase</i> over training steps when using DPO with NLL loss (solid blue). In all four settings, the margin between the two curves continues increasing. We find that DPO+NLL loss gives superior test accuracy in our experiments.	81
6.4	Effect of NLL loss term on DPO training for ARC and MATH. The legend on the right plot is omitted due to space constraint, but it is the same as the legend in the left plot. Similar to GSM8K, in ARC-Challenge and MATH, we see that the log probabilities of chosen sequences barely increase over training steps when training with DPO. However, when training with DPO with NLL loss, the log probabilities increase over training steps.	85
7.1	Self-Rewarding Language Models. Our self-alignment method consists of two steps: (i) <i>Self-Instruction creation</i> : newly created prompts are used to generate candidate responses from model M_t , which also predicts its own rewards via LLM-as-a-Judge prompting. (ii) <i>Instruction following training</i> : preference pairs are selected from the generated data, which are used for training via DPO, resulting in model M_{t+1} . This whole procedure can then be iterated resulting in both improved instruction following and reward modeling ability.	92

C.1	Exposure bias related figures on NQG dev set. Vertical axis: avg unsmoothed sentence-level BLEU. Horizontal axis: sentence length. The colored regions represent 95% confidence interval obtained using standard bootstrapping. Subfigures (a) and (c) show BLEU on randomly shuffled targets (from dev set); BLEU does not appear to punish long sentences. Note the scale of the vertical axes. Subfigures (b) and (d) show BLEU vs. generation length; BLEU on generations from MLE-trained model decreases by length, but BLEU on generations from GOLD-trained model appears to stay relatively stable.	129
C.2	Accuracy of correct predictions of t th token given all prefix reference tokens on NQG dev set. Colored regions represents 95% confidence interval obtained using standard bootstrapping. Without exposure bias, token prediction accuracy stays relatively stable across lengths.	130
C.3	Interface for NQG pairwise comparisons, using Amazon Mechanical Turk.	134
C.4	Interface for summarization pairwise comparisons, using Amazon Mechanical Turk.	134
C.5	Interface for NQG human ratings, using Amazon Mechanical Turk.	134

LIST OF TABLES

3.1	Performance of dialogue models corresponding to different signals, under various metrics. Columns 2–3: Evaluation of generated dialogue responses using different implicit feedback signals. Win rate evaluated by crowdworkers: given “baseline generation wins” for $a\%$ examples, “new generation wins” for $b\%$, “tie” for $c\%$, the win rate is $b-a\%$. Sig.: ** if p -value $\in [0, 0.05)$, * if p -value $\in [0.05, 0.1)$, – otherwise. Columns 4–9: various measured properties of the generations (Subsection 3.4.2). Please refer to Table A.1 and Appendix A.3.2 for complementary details (e.g., human annotation win/lose/tie results, LLM-evaluated win/lose/tie results, avg. length of generations).	25
4.1	Mean and standard deviation (across sequences) of test set forward and reverse rewards for translations. b refers to beam size during inference.	39
4.2	Test set sacreBLEU (mean & standard deviation of three runs using different random seeds). IL performs the best among the three proposed methods.	39
4.3	Test set BLEURT-20-D12 (mean & standard deviation of three runs). IL performs the best among the three proposed methods. Significance test is conducted in Table B.2, which shows that IL’s scores are significantly better than the scores by beam search; in addition, IL’s scores are not significantly different from BSR’s scores.	39

4.4	The rewards and BLEU scores using two KD approaches: $p_{\text{KD}/\text{beam}}$ uses the pseudo-corpus generated by doing beam search from p_f . $p_{\text{KD}/\text{nc}}$ uses the pseudo-corpus generated by BSR.	43
4.5	Corpus-level BLEU between translations by pairs of systems. Each reported BLEU is averaged between two directions.	45
4.6	WMT'16 Ro-En examples produced by different systems. The top example is randomly selected. The bottom example is an example with a long source, and Q learning produces repetitions.	46
5.1	BLEU/ROUGE (\uparrow) and perplexity (\downarrow) using standard models on test sets. GOLD achieves better metric scores despite high held-out perplexity. Experiments are run using a fixed random seed (12); attempted three random seeds (1, 12, 123) and all BLEU/R-2 scores are within 0.1 points of the reported. Refer to Table 5.3 for transformer results.	60
5.2	Dev set results of standard models using different decoding algorithms. b : beam size. We report the average of 3 runs for top- k sampling. Models trained by GOLD are less sensitive to decoding algorithms.	60
5.3	Results using transformer models on test sets. The advantage of GOLD is maintained on advanced models based on transformers and pretraining.	60
5.4	BLEU/ROUGE (\uparrow) on test sets, using standard models finetuned with on-policy objectives. On-policy objectives marginally improve upon both MLE and GOLD baselines. Starred (*) models have MLE baselines >0.1 difference to our MLE R-2. $\delta\text{R-2}$: R-2 for the model minus R-2 for the corresponding MLE. PG: standard policy gradient; PPO: proximal policy optimization.	61

5.5	Human comparisons on 200 randomly selected test examples for each task. Win: % generations from GOLD-trained BART that are better than from MLE-trained BART, given the same source.	62
5.6	NQG generations using standard models. Words to query on are bolded. Long generations from MLE-trained model often result in repetition or hallucination. Refer to more examples in the appendix.	64
6.1	GSM8K results comparing Iterative Reasoning Preference Optimization (Iterative RPO) against other baselines that are based on the same base model and training data. We report the exact match accuracy from a single generation (using greedy decoding), as well as majority voting over 32 generations (through sampling with temperature 0.8).	79
6.2	ARC and MATH results. We compare Iterative Reasoning Preference Optimization (Iterative RPO) against other baselines that are based on the same base model and training data.	84

A.1	Performance of generated dialogue responses using different implicit feedback signals. Classifier accuracy: the classification accuracy on a balanced dev set (meaning the classes corresponding to the same number of examples); even though the accuracy is not high, we see that the classifiers can still help improve the bot dialogue generations. Avg. score: our new generations achieve better classifier scores compared to the baseline; this observation is guaranteed given our sample-and-rerank algorithm but we believe it is still instructive to see how large the gap is. Length: we see that other than the “baseline + length ($k=5$)” generation, the other generations’ average length are similar, so the model is not simply optimizing for longer responses. Sig.: ** if p -value $\in [0, 0.05)$, * if p -value $\in [0.05, 0.1)$, – otherwise. We find general agreement between human annotator evaluation results and the LLM evaluation results when averaging over 200 examples.	98
A.2	Confusion table of LLM evaluations vs. expert evaluations. “Gen” stands for “generation.” 64.5% annotations match; 14% annotations strongly disagree (as defined in §3.4.2, the % of annotations that strongly disagree equals the % of LLM choosing “baseline gen wins” and experts choosing “new gen wins” plus the % of LLM choosing “new gen wins” and experts choosing “baseline gen wins”). We see from this table that the LLM–expert match is not satisfactory.	101
A.3	Confusion table of LLM evaluations vs. annotator evaluations. “Gen” stands for “generation.” 54.5% annotations match; 20.5% strongly disagree. We see from this table that the LLM–annotator match is not high.	102
A.4	Confusion table of expert evaluations vs. annotator evaluations. “Gen” stands for “generation.” 86% annotations match; 6.5% strongly disagree. This agreement is better than the LLM–expert agreement and the LLM–annotator agreement. . . .	102
B.1	Experimental settings for the forward model p_f and the reverse (channel) model p_r .	119

B.2	Test set BLEURT-20-D12 (mean & standard deviation of three runs). IL performs the best among the three proposed methods. *: The score is significant (p-value smaller than 0.05) compared to the beam search results. †: The score is significantly higher (p-value smaller than 0.05) than BSR results, or the score is not significantly different (p-value larger than 0.05) from the BSR results.	120
C.1	NQG and CNN/DM examples based on transformer models. For NQG, words to query on are bolded.	131
C.2	XSum and IWSLT14 De-En examples based on transformer models.	132

1 | INTRODUCTION

Text generation has wide-ranging applications including machine translation, summarization (e.g., news summarization, book summarization), dialogue response generation (e.g., in social chatbots, task-oriented chatbots), code generation, image captioning, report generation (e.g., meeting, financial, weather, patient report generation), among many others. Tasks with non-natural-language outputs, ranging from constituency or dependency parsing to generating the structure of proteins, also benefit profusely by the development of text generation technologies. Text generation used to be a niche field with a small community when I started my first major NLP research project in 2017, but recent breakthroughs in large language models (LLMs) have allowed practitioners to formulate a very large number of applications as text generation.

Text generation systems in general have traditionally involved steps including document planning (e.g., content determination, document structuring), microplanning (e.g., lexicalization, sentence aggregation), and linguistics realization [Reiter and Dale 1997; Reiter 2007]. Recent neural methods (which parameterize the input-to-output mapping using a neural network) have merged some or all steps. Large-scale pretraining and instruction tuning on a gigantic amount of data have enabled a single model to answer questions regarding different tasks in different domains [Peters et al. 2018; Ouyang et al. 2022; Touvron et al. 2023b]. The rapid developments have made large language models (LLMs) known to the public. Although neural methods on text generation have enabled many breakthroughs, issues remain. For example, models may not generate truthful information, and in general, models may not do what humans intend them to

do, raising safety concerns [Amodei et al. 2016; Ji et al. 2023].

The progress in text generation comes from every stage in the pipeline. There has been great progress in data curation, learning algorithms, decoding algorithms, as well as evaluation metrics. While these stages can never be separated, this dissertation mostly focuses on learning algorithms. We investigate how to enable the model to achieve what humans prefer (or proxies of what humans prefer) – in fact, the traditional training algorithm maximum likelihood estimation (MLE) may not be in line with our ultimate objective (i.e., achieving what humans intend the model to do), as discussed in more detail in [Chapter 2](#).

The most important issue we tackle is the mismatch between the training objective and the evaluation objective. As discussed further in [Section 2.2](#), MLE aims to minimize the cross-entropy loss with respect to each sample in the dataset. However, the downstream evaluation is based on a reward that scores the compatibility of the input-output pair – for example, a reward could often be the human judgment of output quality. The training data may be suboptimal, so the oracle distribution induced by the dataset may be inherently incorrect and may be mismatched with the evaluation objective. Even if the training data is high-quality, there is no guarantee that even the generations decoded from an oracle distribution would lead to high rewards or the optimal human judgment of output quality.

Therefore, our series of work is motivated by this mismatch, and we aim to bridge the gap between training and evaluation by directly maximizing the *reward* of the generated text. Reward is a general term: It can represent the evaluation signals or their proxies; it can also be signals that encourage learning toward our intended goal. As said by [Sutton and Barto \[1998, 2018\]](#) in their book on reinforcement learning (RL):

“[Reward] is more like a signal generated inside an animal’s brain than it is like an object or event in the animal’s external environment. The parts of our brains that generate these signals for us evolved over millions of years to be well suited to the challenges our ancestors had to face in their struggles to propagate their genes to

future generations. We should therefore not think that designing a good reward signal is always an easy thing to do!”

There are many reasons that finding a reward is challenging. (1) Some rewards could be expensive to obtain (e.g., when running costly experiments like in self-driving cars, or obtaining human annotations in text generation). (2) An agent could make slow or no progress in learning: rewards could be sparse; in other words, rewards may only be available after a long sequence of actions, without ways to mark intermediate progress. Or the reward may simply be bad, thereby making learning difficult [Minsky 1961; Sutton and Barto 2018]. (3) An agent could learn the wrong thing: It could discover novel but undesirable ways to obtain rewards. This is the issue of reward hacking or reward gaming [e.g., Amodei et al. 2016; Skalse et al. 2022; Pang et al. 2023].

This dissertation focuses on (1) and (2) described above. For (1), regarding the issue that rewards could be expensive, we aim to explore ways to cheaply extract proxy rewards. In fact, there has been abundant work arguing that having different goals (from the original goal) may benefit the agent instead of optimizing for the original goal directly [Ng et al. 1999; Sorg et al. 2010, 2012]. For (2), on addressing slow or no progress learning, one solution is to design denser rewards or find high-quality reward signals in general [Sutton and Barto 2018]. An orthogonal approach is to innovate on algorithms; for example, we can remove the component of exploring in the environment by relying on approaches in imitation learning or learning from demonstration or by relying on non-RL approaches.

Below is a succinct outline of this dissertation, which is then followed by an extended outline.

- We first provide one example that tackles the first problem above that getting rewards could be expensive. We investigate a few ways to obtain rewards using social dialogue as an example. As discussed above, having proxy goals could even accelerate learning and benefit an agent. In this work, we explore methods of extracting *implicit* proxy rewards from social chatbot deployment data and attempt to maximize reward during decoding time (Chapter 3) – in this case, rewards are based on learned classifiers extracted from deployment data.

- Related to the second issue above, we obtain denser and high-quality rewards inspired by a classic NLP algorithm that decodes translations by leveraging the noisy channel model ([Chapter 4](#)); a byproduct is that we can amortize the cost of running such algorithm.
- Rewards may be of poor quality and optimization may be difficult, so in [Chapter 5](#) and [Chapter 6](#), we investigate the case where the rewards are based on exact match (of generations with respect to the reference) and we innovate on algorithms instead. We can use offline RL ([Chapter 5](#)) or non-RL algorithms like variants of direct preference optimization (DPO) ([Chapter 6](#)) for reasoning tasks.
- Future directions involve using LLM outputs as the reward, which could potentially open doors for superhuman-level feedback. We briefly mention some promising first steps in [Chapter 7](#) which leverages a similar version of iterative DPO discussed in [Chapter 6](#). This section also discusses other relevant and impactful directions.

Following the above, here is an extended outline of the dissertation. First, [Chapter 2](#) provides the background on text generation including rewards and different learning criteria in text generation. The rest of the main chapters are organized as follows.

A EXAMPLE OF EXTRACTING REWARDS IN DIALOGUE SYSTEMS: LEVERAGING IMPLICIT FEEDBACK FROM DEPLOYMENT DATA. [Chapter 3](#) explores an example of rewards in social dialogue systems – a real everyday application. Given that the rewards could be expensive if obtained using human ratings, we extract *implicit* rewards from deployment data. The rewards are based on learned classifiers that predict future human engagement patterns. After extracting the rewards, we use sample-and-rerank at decoding time to achieve high desired reward. More specifically, We study improving social conversational agents by learning from natural dialogue between users and a deployed model, *without extra annotations*. To implicitly measure the quality of a machine-generated utterance, we leverage signals like user response length, sentiment, and reaction of the future

human utterances in the collected dialogue episodes. Our experiments use the publicly released deployment data from BlenderBot [Xu et al. 2023a]. Human evaluation indicates improvements in our new models over baseline responses; however, we find that some proxy signals can lead to more generations with undesirable properties as well. For example, optimizing for conversation length can lead to more controversial or unfriendly generations compared to the baseline, whereas optimizing for positive sentiment or reaction can decrease these behaviors. This chapter is based on work with Stephen Roller, Kyunghyun Cho, He He, and Jason Weston [Pang et al. 2024a].

REWARDS CAN BE BASED ON CLASSIC NLP ALGORITHMS TO AMORTIZE COMPUTATION COST: AMORTIZED NOISY CHANNEL NEURAL MACHINE TRANSLATION. Prior work [Choshen et al. 2020; Donato et al. 2022] has shown that optimizing for the classic BLEU reward using regular RL approaches in neural machine translation (NMT) leads to minimal or even negative improvements. As mentioned above, one way to approach the problem is to choose other rewards, potentially denser and higher-quality ones. In Chapter 4, we get inspired by noisy channel models which have had a long history [Shannon 1948; Kernighan et al. 1990; Jelinek 1997; Echihabi and Marcu 2003; Koehn et al. 2003]. In recent years it has been especially effective in NMT [Yee et al. 2019; Ng et al. 2019; Chen et al. 2020; Yu et al. 2020; Tran et al. 2021; Kocmi et al. 2023]. However, recent approaches like “beam search and rerank” (BSR) incur significant computation overhead during inference, making real-world application infeasible. We aim to study if it is possible to build an amortized noisy channel NMT model such that when we do greedy decoding during inference, the translation accuracy matches that of BSR in terms of reward (based on the source-to-target log probability and the target-to-source log probability) and quality (based on BLEU and BLEURT). We attempt three approaches to train the new model: knowledge distillation, 1-step-deviation imitation learning, and Q learning. The first approach obtains the noisy channel signal from a pseudo-corpus, and the latter two approaches aim to optimize toward a noisy-channel MT reward directly. For all three approaches, the generated translations fail to achieve rewards comparable to BSR, but the

translation quality approximated by BLEU and BLEURT is similar to the quality of BSR-produced translations. Additionally, all three approaches speed up inference by 1–2 orders of magnitude. This chapter is based on work with He He and Kyunghyun Cho [Pang et al. 2022a].

REWARDS CAN BE SIMPLE AND WE CAN INNOVATE ON LEARNING ALGORITHMS: TEXT GENERATION BY LEARNING FROM DEMONSTRATION. Rewards could be bad and optimization could be difficult. We should not only rely on finding good rewards. We should improve learning algorithms as well. Here we explore the simple case that the rewards can be based on whether the generation exactly matches the reference answer. If there is an exact match, then the reward is 1; otherwise, the reward is 0. Even though the reward is simple, we can still improve based on learning algorithms. In [Chapter 5](#), we innovate on the RL algorithm so that there is no interaction with the environment and the policy (i.e., generator) stays close to demonstrated trajectories (i.e., generations) – there will not be low-quality sequences with high rewards. Specifically, current approaches to text generation largely rely on autoregressive models and maximum likelihood estimation. This paradigm leads to (i) diverse but low-quality samples due to mismatched learning objectives and evaluation metrics (likelihood vs. quality) and (ii) exposure bias due to mismatched history distributions (gold vs. model-generated). To alleviate these problems, we frame text generation as an offline reinforcement learning (RL) problem with expert demonstrations (i.e., the reference), where the goal is to maximize quality given model-generated histories. We propose GOLD (generation by off-policy learning from demonstrations): an easy-to-optimize algorithm that learns from the demonstrations by importance weighting. Intuitively, GOLD upweights confident tokens and downweights unconfident ones in the reference during training, avoiding optimization issues faced by prior RL approaches that rely on online data collection. According to both automatic and human evaluation, models trained by GOLD outperform those trained by MLE and policy gradient on summarization, question generation, and machine translation. Further, our models are less sensitive to decoding algorithms and alleviate exposure bias. This chapter is based on

work with He He [Pang and He 2021].

WE DO NOT NECESSARILY NEED RL: ITERATIVE REASONING PREFERENCE OPTIMIZATION. In [Chapter 6](#), we deal with generating answers for mathematics and general reasoning questions. The reward is still based on exact match of generations and references. In this chapter we emphasize that we do not need RL to optimize for rewards; in fact, we leverage a variant on a non-RL criteria called direct preference optimization [DPO; Rafailov et al. 2023]. More specifically, iterative preference optimization methods have recently been shown to perform well for general instruction tuning tasks, but typically make little improvement on reasoning tasks [Yuan et al. 2024; Chen et al. 2024]. In this work we develop an iterative approach that optimizes the preference between competing generated Chain-of-Thought (CoT) candidates by optimizing for winning vs. losing reasoning steps that lead to the correct answer. We train using a modified DPO loss [Rafailov et al. 2023] with an additional negative log-likelihood term, which we find to be crucial. We show reasoning improves across repeated iterations of this scheme. While only relying on examples in the training set, our approach results in increasing accuracy on GSM8K, MATH, and ARC-Challenge for Llama-2-70B-Chat, outperforming other Llama-2-based models not relying on additionally sourced datasets. For example, we see a large improvement from 55.6% to 81.6% on GSM8K and an accuracy of 88.7% with majority voting out of 32 samples. This chapter is based on work with Weizhe Yuan, Kyunghyun Cho, He He, Sainbayar Sukhbaatar, and Jason Weston [Pang et al. 2024b].

FUTURE DIRECTIONS. Rewards can be based on LLM outputs. Self-rewarding language models [Yuan et al. 2024], joint work with Weizhe Yuan, Kyunghyun Cho, Xian Li, Sainbayar Sukhbaatar, Jing Xu, and Jason Weston, is a first step towards this direction, experimenting on general instruction-following tasks. LLMs can serve as powerful reward models, and we leverage iterative direct preference optimization, a variant of which has been used in [Chapter 6](#). We briefly mention self-rewarding language models in [Chapter 7](#).

Specifically, we posit that to achieve superhuman agents, future models require superhuman feedback in order to provide an adequate training signal. Current approaches commonly train reward models from human preferences, which may then be bottlenecked by human performance level, and secondly these separate frozen reward models cannot then learn to improve during LLM training. We study self-rewarding language models, where the language model itself is used via LLM-as-a-Judge prompting to provide its own rewards during training. We show that during iterative DPO training, not only does instruction following ability improve, but also the ability to provide high-quality rewards to itself.

The work described above naturally necessitates future directions. Self-rewarding language models require strong LLM evaluators. In the self-rewarding LM work, experiments are done on general instruction-following tasks. While the LLM could be a good evaluator on these tasks, there is no guarantee that LLM could excel at evaluating questions in every topic, e.g., on more difficult reasoning tasks. So, how do we ensure that LLMs are excellent evaluators? Next, if LLMs are excellent evaluators, then we could potentially do iterative learning preference optimization ([Chapter 6](#)) in an unsupervised way where we do not rely on reference answers in reasoning tasks.

In addition, one may ask whether it is possible to achieve even higher superhuman performance on evaluation (and to improve LLM capabilities in general) while integrating regular humans in the process: Is it possible to ask humans and AIs to collaborate so that the final performance is better than humans on their own as well as AIs on their own? A brief discussion can be found in [Chapter 7](#).

2 | PRELIMINARIES

2.1 CONDITIONAL TEXT GENERATION

Conditional text generation systems usually model $p(\mathbf{y} | \mathbf{x})$ where $\mathbf{x} = (x_1, \dots, x_{T_s})$ is a source sequence and $\mathbf{y} = (y_1, \dots, y_T)$ is a target sequence. For example, in machine translation, \mathbf{x} could be a sequence in one language, and \mathbf{y} could be a sequence in another language. In dialogue, \mathbf{x} could be the concatenation of all past dialogue history, and \mathbf{y} could be the immediate next response. In language model pretraining, \mathbf{x} usually just contains the beginning-of-sequence token or simply contains nothing.

Most models use an autoregressive factorization:

$$\log p(\mathbf{y} | \mathbf{x}) = \sum_{t=1}^T \log p_{\theta}(y_t | \mathbf{y}_{<t}, \mathbf{x}), \quad (2.1)$$

where $\mathbf{y}_{<t} = (y_1, \dots, y_{t-1})$ is the history of prefix of the generated sequence and p_{θ} is parameterized with a neural network. Maximum likelihood estimation (MLE) leads to mismatched train vs. test history and mismatched train vs. test objectives during sequence generation [Bengio et al. 2015; Huszár 2015; Ranzato et al. 2016; Schmidt 2019; Pang and He 2021; Arora et al. 2022]. In addition, recent work aims to better align training objectives with human-annotated quality of generated texts; e.g., translation quality judgments, summarization faithfulness, human preference of generations [Pasunuru and Bansal 2018; Stiennon et al. 2020; Gunasekara et al. 2021; Wu et al.

2021; Bai et al. 2022].

2.2 LEARNING VS. EVALUATION AND THE NEED FOR REWARD

For learning, the standard maximum likelihood estimation (MLE) loss is as follows:

$$L_{\text{MLE}}(\theta) = -\mathbb{E}_{(\mathbf{x}, \mathbf{y}) \sim \mathcal{D}} \sum_{t=1}^T \log p_{\theta}(y_t \mid \mathbf{y}_{<t}, \mathbf{x}), \quad (2.2)$$

where \mathcal{D} is the training dataset. The evaluation metric is often of the form

$$\mathbb{E}_{\mathbf{y} \sim p_{\theta}} \text{reward}(\mathbf{x}, \mathbf{y}). \quad (2.3)$$

There are a few reasons that learning using a standard MLE objective may not be enough for text generation.

FIRST, REFERENCE DATA MAY NOT REPRESENT WHAT WE WANT. The dataset could be unclean [Freitag et al. 2021; Herold et al. 2022; Kreutzer et al. 2022; Li et al. 2023], or the dataset may not reflect high reward however it is defined; for example, different groups of people may have different values and prefer different evaluation objectives [Bakker et al. 2022; Kirk et al. 2024]. In addition, as we see above, MLE forces the model to learn every example which intuitively maximizes recall, but we may just want any high-reward output which intuitively corresponds to precision – sometimes we do not care about the ability for the model to generate a diverse set of outputs for any given input (e.g., for machine translation as well as many question-answering tasks).

SECOND, REFERENCE DATA MAY BE COSTLY OR INFEASIBLE TO OBTAIN. It may be costly to obtain enough gold or top-quality data; instead, it could be easier to obtain a good reward function and

optimize toward it. An intuitive example is that in an open-domain assistant setting (e.g., writing a poem on a certain topic), having enough good outputs could be expensive, but we can have a decent reward function to guide us to better generations given that judging could often be easier than writing. Recent progress on RLHF proves that optimizing toward a reward function can lead to performances that are better than merely doing supervised fine-tuning (SFT) on good samples, so as optimization goes on, the generations may gradually improve.

THIRD, EXTRA SUPERVISION BY THE REWARD FUNCTION MAY BE BENEFICIAL. If there is not enough data to cover a region in the input space especially because text representations are high-dimensional, under model misspecification, prior work has shown that MLE could over-generalize and assign large probability mass to both high-quality and low-quality sequences [Huszár 2015; Simon et al. 2019]. If MLE-trained model assigns high probability on a low-quality generation, then we can teach the models that those generations are bad by trial and error.

Once we have a reward function, how do we optimize toward it? There are many different algorithms at training time or decoding time, and we first discuss RL methods.

2.3 REINFORCEMENT LEARNING AND TEXT GENERATION

The generation process can be considered a sequential decision making process suitable for RL. Given state $s_t = (\mathbf{x}, \mathbf{y}_{<t})$, the policy π_θ (or denoted p_θ) takes action a_t (a token in the vocabulary), transits to the next state $s_{t+1} = (\mathbf{x}, \mathbf{y}_{<t+1})$, and receives a reward $r_t \in \mathbb{R}$. The goal is to have generations maximize the objective:

$$J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta} r(\tau), \tag{2.4}$$

where $\tau = (s_1, a_1, \dots, s_T, a_T)$ is the trajectory and $r(\tau) = \sum_{t=1}^T r_t$.

There are a large number of RL algorithms, and there are many ways to categorize RL algo-

rithms. In this dissertation, we explore two kinds of algorithms: the ones that learn the policies, and the ones that learn action-value functions. In this background section, for the first category, we discuss policy optimization algorithms. For the second category, we discuss a relatively recent Q learning method called deep Q networks (DQN).

2.3.1 POLICY OPTIMIZATION ALGORITHMS

VANILLA POLICY GRADIENT. To maximize the objective $J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta} r(\tau)$ where $r(\tau) = \sum_{t=1}^T r_t$, one way is to use policy gradient [REINFORCE; Williams 1992; Sutton et al. 1999]:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta} \sum_t \nabla_\theta \log \pi_\theta(a_t | s_t) \hat{Q}(s_t, a_t), \tag{2.5}$$

where

$$\hat{Q}(s_t, a_t) = \sum_{t'=t}^T r_{t'} \tag{2.6}$$

is the estimated return.

To stabilize RL training, in each RL training run, we first initialize the model using an MLE-trained model to ensure a good starting point for RL optimization. In addition, we also use KL regularization which helps RL optimization [Jaques et al. 2019; Stiennon et al. 2020; Ramamurthy et al. 2023] in part to tackle reward gaming [Gao et al. 2023; Skalse et al. 2022; Pang et al. 2023], so

$$J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta} [r(\mathbf{x}, \mathbf{y}) - \beta (\log \pi_\theta(\mathbf{y} | \mathbf{x}) - \log \pi_{\text{ref}}(\mathbf{y} | \mathbf{x}))]. \tag{2.7}$$

In this equation we use $r(\mathbf{x}, \mathbf{y})$ to denote $r(\tau)$ for clarity in the sequence generation setting, and π_{ref} is the reference model which is typically the model trained using standard MLE on the training set of the target task (or a large number of diverse high-quality instruction-following examples for general pretrained language models). β is often hard-to-tune; in some examples [Pang et al.

2023], even large β does not eliminate undesirable behaviors.

PROXIMAL POLICY OPTIMIZATION (PPO). PPO [Schulman et al. 2017] has also been widely used. It has become the standard algorithm for doing reinforcement learning with human feedback (RLHF), a critical step that aligns large language model with human preferences of generations in the past few years. Please refer to later in this subsection for more discussion on RLHF. Recent reinforcement learning with AI feedback (RLAIF) algorithms also frequently employ PPO. The fundamental idea is based on REINFORCE. However, PPO is designed such that a clipping term is introduced in the the objective function to prevent the gradient step from being too big, thereby causing performance collapse of the policy. As it is not pivotal to the innovations introduced in this dissertation, we refer readers to Schulman et al. [2017] and OpenAI [2024] for the detailed algorithm.

2.3.2 Q LEARNING

Unlike REINFORCE and PPO which are on-policy (meaning that each update uses the trajectories produced by the current policy), Q learning is off-policy: the update can use data collected anytime; it does not matter how the data was obtained through exploration, either.

In Q learning, $Q^\pi : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$ is a function, where \mathcal{S} is the state space and \mathcal{A} is the action space, such that $Q^\pi(s_t, a_t)$ produces the expected return after seeing state s_t , taking action a_t , and following policy π ; i.e.,

$$Q^\pi(s_t, a_t) = \mathbb{E} \left[\sum_{t'=t}^{\infty} r_{t'} \mid s_t, a_t, \pi \right], \quad (2.8)$$

assuming discount factor 1. We further define $Q^* : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$ to be the optimal action-value

function:

$$Q^*(s_t, a_t) = \max_{\pi} \mathbb{E} \left[\sum_{t'=t}^{\infty} r_{t'} \mid s_t, a_t, \pi \right], \quad (2.9)$$

which is the maximum return achievable by following any strategy *after* seeing a state s_t and taking an action a_t . In particular, Q^* solves the Bellman Equation [Sutton and Barto 1998]:

$$Q^*(s_t, a_t) = r_t + \max_{a_{t+1}} Q^*(s_{t+1}, a_{t+1}), \quad (2.10)$$

assuming discount factor 1 and given deterministic transition dynamics (in our machine translation scenario) after taking action a_t given state s_t .

Traditionally, the Q function is implemented as a matrix of size $|\mathcal{S}| \times |\mathcal{A}|$, which is intractable in the case of many applications including machine translation due to the combinatorial nature of the state space. We thus use function approximation to tackle this issue of intractability. We follow Mnih et al. [2015] and use a deep neural network trained with experience replay and target networks to approximate the Q learning.

Deep Q learning draws samples from a set of trajectories \mathcal{B} , and the neural network Q aims to predict Q^* by learning based on minimizing the following squared loss.

$$L(\phi) = \frac{1}{|\mathcal{B}|} \sum_{(s_t, a_t, s_{t+1}, r_t) \sim \text{Uniform}(\mathcal{B})} \left[\left(r_t + \max_{a_{t+1}} Q'(s_{t+1}, a_{t+1}) - Q(s_t, a_t) \right)^2 \right], \quad (2.11)$$

where ϕ is the parameter to Q , and Q' is a slightly old copy of Q . In other words, after a fixed number of optimization steps, we update Q' by Q .

Intuitively, we want the scalar $Q(s_t, a_t)$ to be close to the sum of the t -th step reward and the *most optimistic* future return had we taken action a_t at step t .

2.3.3 GENERAL RLHF FRAMEWORK IN TRAINING LANGUAGE MODELS

While the above discusses a general framework for text generation – not only for language modeling – these days language models have become popular and training a language model usually at least consists of the following three major steps, and the third major step could contain the RLHF paradigm. Let us enumerate the three steps. The first major step is pretraining: the most popular paradigm these days is next-token prediction where given the first k tokens, the model learns to predict the immediate next token via a standard cross-entropy objective. The model is usually pretrained on an enormous amount of data; e.g., Llama 3 is pretrained on more than 15 trillion tokens [Meta 2024]. The second major step is usually supervised fine-tuning where the model is trained on high-quality data of an exceptionally diverse range of tasks using the cross-entropy objective. The third major step is alignment; the popular paradigm has been reinforcement learning with human feedback (RLHF) for the past two years, but the landscape is quickly changing at the time of this writing.

The third major step, using the RLHF framework, can be described in three sub-steps. The first sub-step is data collection: Candidate generations are sampled from the reference model (usually SFT-trained model or the most recent model in the iterative training process) and the generations are given to humans. Then, we obtain $(\mathbf{y}_w, \mathbf{y}_l)$ where \mathbf{y}_w is the winning generation and \mathbf{y}_l is the losing generation, and we aggregate these preference pairs into a dataset \mathcal{D} . The second sub-step requires training a reward model $r_\phi(\mathbf{x}, \mathbf{y})$ using the loss

$$L_r(\phi) = -\mathbb{E}_{(\mathbf{x}, \mathbf{y}_w, \mathbf{y}_l) \sim \mathcal{D}} [\log \sigma(r_\phi(\mathbf{x}, \mathbf{y}_w) - r_\phi(\mathbf{x}, \mathbf{y}_l))], \quad (2.12)$$

where σ is the sigmoid function. The third sub-step is training the language model using RL where the reward relies on the reward function we have trained in the second sub-step. As discussed, the reward is regularized to prevent reward gaming and to prevent mode collapse so as to ensure

generation diversity. The objective to be maximized is as follows:

$$J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta} [r(\mathbf{x}, \mathbf{y}) - \beta(\log \pi_\theta(\mathbf{y} | \mathbf{x}) - \log \pi_{\text{SFT}}(\mathbf{y} | \mathbf{x}))]. \quad (2.13)$$

As mentioned before, the exact details have been evolving.

RLHF is not only used for aligning LLMs. In general, this class of methods of training a text generator using learned rewards consists of the following three steps [Pang et al. 2023]. (1) Collect a human annotation dataset $\mathcal{D}_{\text{reward}}$ consisting of, e.g., direct ratings of generations [Sellam et al. 2020b; Nakatani et al. 2022; Ramamurthy et al. 2023], labels of error spans in the generations [Freitag et al. 2020; Amrhein and Sennrich 2022], or pairwise comparison of generations given the same source sequence [Stiennon et al. 2020]; in general the human annotations do not need to be pairwise preferences. (2) Learn a proxy reward function that scores generations on $\mathcal{D}_{\text{reward}}$, using any learning criteria. (3) Learn the text generator on a dataset $\mathcal{D}_{\text{task}}$, using RL with the learned reward function.

2.4 NON-RL ALGORITHMS IN TEXT GENERATION

While there are many algorithms and objectives, we only present a few representative ones.

SAMPLE AND RERANK. The sample-and-rerank approach (also called best-of- n) requires first obtaining a number of samples and then rerank according to the reward; in the end, we output the sample with the largest reward. The performance of the approach depends on the application but it has been shown to perform similarly or almost as good as RL-based approaches in specific machine translation applications [Pang et al. 2022a] and learning from pairwise feedback in language modeling [Dubois et al. 2023]. No training is required as sample-and-rerank is an inference algorithm, but computation is costly depending on the number of samples. Nevertheless, this approach is a common baseline and it is easy to implement.

SFT & EXPERT ITERATION APPROACHES. One can train only on high-reward samples; the SFT step in LLM training usually includes training on high reward samples (as well as other collected generations), and this step often appears after pretraining and before RLHF fine-tuning. There are a variety of tricks in this class of approaches; approaches in [Gulcehre et al. \[2023\]](#) and [Dubois et al. \[2023\]](#) both train on high-reward generations and we can call them expert iteration approaches in general.

CRINGE LOSS AND PAIRWISE CRINGE LOSS. Cringe loss [[Adolphs et al. 2023](#)] stands for contrastive iterative negative generation loss. The general idea is to directly push up the probability of tokens in a positive sequence, and directly push down the probability of tokens in a negative sequence. Given a positive generation and a negative generation, we use the regular cross entropy loss for the positive generation and use the Cringe loss for the negative generation. In particular, in Cringe loss, each negative token is contrasted with a positive token sampled from the model’s top- k predictions. This choice is to address the problem in unlikelihood loss [[Welleck et al. 2020c](#)] where only decreasing the probability of tokens in negative sequences may lead to increased probabilities of rare but poor-quality tokens. Therefore, the loss is defined as:

$$L_{\text{binary-cringe}}(\mathbf{x}^+, \mathbf{y}^+, \mathbf{x}^-, \mathbf{y}^-) = -\log p(\mathbf{x}^+) - \log p(\mathbf{y}^+ | \mathbf{x}^+) - \alpha \sum_t \log \frac{\exp(s(y_t'^+))}{\exp(s(y_t'^+)) + \exp(s(y_t^-))}, \quad (2.14)$$

where $(\mathbf{x}^+, \mathbf{y}^+)$ is the positive input-output pair, $(\mathbf{x}^-, \mathbf{y}^-)$ is the negative input-output pair, token $y_t'^+$ is sampled using top- k sampling to contrast with the negative token as mentioned above, the function $s(y_t'^+)$ gives the probability of the token $y_t'^+$ during softmax over the k sampled tokens, and α is a real number which is the coefficient for the Cringe loss.

The pairwise Cringe loss [[Xu et al. 2023b](#)] is an effective variant which multiplies the binary

loss above by a gating function (see multiplier below):

$$L_{\text{pairwise-ctrl}} = \left[\sigma \left(\log p(\mathbf{y}^l | \mathbf{x}) - \log p(\mathbf{y}^w | \mathbf{x}) + b \right) \right] L_{\text{binary-ctrl}}(\mathbf{x}, \mathbf{y}^w, \mathbf{x}, \mathbf{y}^l). \quad (2.15)$$

In short, if $\log p(\mathbf{y}^w | \mathbf{x}) - \log p(\mathbf{y}^l | \mathbf{x})$ is large enough, then the sigmoid function ensures that the gating function returns a small value which leads to a smaller loss.

In general, this class of approach based on ranking losses has a long history in natural language processing [Collobert et al. 2011]. Direct preference optimization described below is also closely connected.

DIRECT PREFERENCE OPTIMIZATION (DPO). Rafailov et al. [2023] find it unnecessary to train a separate reward model before optimizing the generator. DPO skips over the step, and the objective operates over policies (instead of rewards). The objective is as follows:

$$L_{\text{DPO}}(\pi_\theta, \pi_0) = -\mathbb{E}_{(\mathbf{x}, \mathbf{y}_w, \mathbf{y}_l) \sim \mathcal{D}} \left[\log \sigma \left(\beta \log \frac{\pi_\theta(\mathbf{y}_w | \mathbf{x})}{\pi_{\text{ref}}(\mathbf{y}_w | \mathbf{x})} - \beta \log \frac{\pi_\theta(\mathbf{y}_l | \mathbf{x})}{\pi_{\text{ref}}(\mathbf{y}_l | \mathbf{x})} \right) \right], \quad (2.16)$$

where π_θ is the policy we are optimizing, π_{ref} is the reference policy (could also be denoted by π_r or π_0), β is a hyperparameter and according to the DPO derivation it corresponds to the coefficient of the KL regularization term (between π_θ and π_{ref}), and \mathcal{D} is the dataset containing $(\mathbf{x}, \mathbf{y}_w, \mathbf{y}_l)$ tuples which represent for input prompt, winning/chosen response, losing/rejected response respectively in each tuple. In particular, the difference $\beta \log \frac{\pi_\theta(\mathbf{y}_w | \mathbf{x})}{\pi_{\text{ref}}(\mathbf{y}_w | \mathbf{x})} - \beta \log \frac{\pi_\theta(\mathbf{y}_l | \mathbf{x})}{\pi_{\text{ref}}(\mathbf{y}_l | \mathbf{x})}$ can be understood as the reward margin.

3 | AN EXAMPLE OF INEXPENSIVELY EXTRACTING REWARDS IN DIALOGUE SYSTEMS: LEVERAGING IMPLICIT FEEDBACK FROM DEPLOYMENT DATA

It could be expensive to collect human annotations to train the reward function. In this chapter, we explore extracting implicit rewards from the BlenderBot deployment data [Xu et al. 2023a]. The rewards are based on learned classifiers that predict future human engagement patterns. After extracting the rewards, we use sample-and-rerank during decoding to achieve high desired reward. In this way, reward extraction does not require any extra human annotation, unlike the RLHF procedure described in Chapter 2. This chapter is based on work with Stephen Roller, Kyunghyun Cho, He He, and Jason Weston [Pang et al. 2024a].

3.1 INTRODUCTION

A core strategy to improve social conversation models is through human feedback. There has been remarkable progress on learning from feedback, including reinforcement learning with human feedback [Stiennon et al. 2020; Bai et al. 2022], where a large number of human annotations are

needed to ensure a good reward function. The feedback usually involves binary ratings, numerical scores, rankings, or natural language comments of a dialogue turn or episode. These signals are most often collected explicitly using crowdworkers, as organic users may not want to be burdened with providing explicit signals, or else may provide unreliable information [Ju et al. 2022].

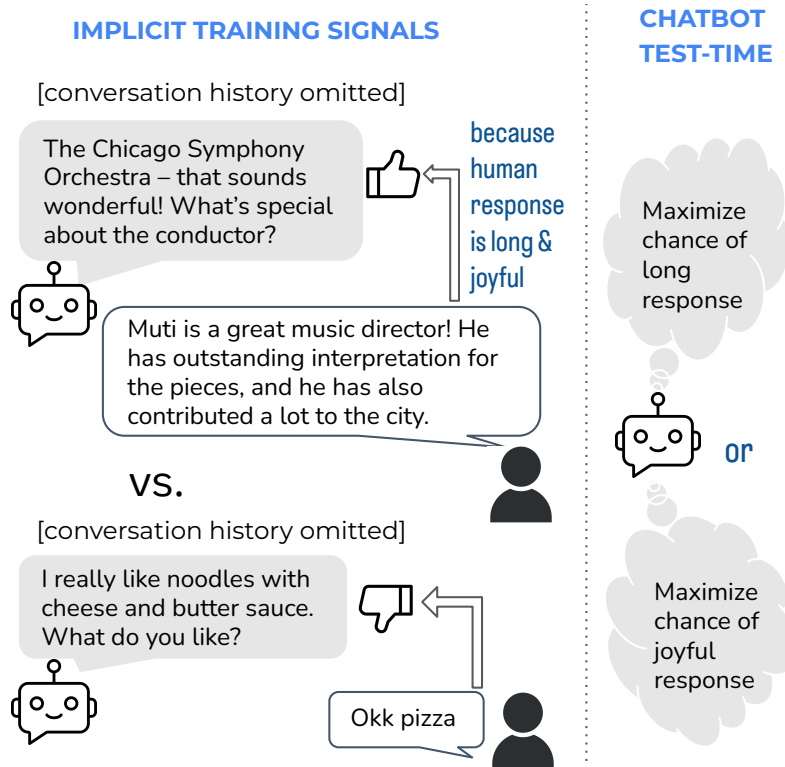


Figure 3.1: Overview of the approach. Implicit signals are extracted from conversations, such as whether future human turns are long or short, or joyful or not. We train a binary classifier to predict whether the bot turn is “good” given the conversation history and the bot turn, and we leverage the classifier at bot’s test-time. We study various kinds of implicit signal in this work (Section 3.3).

In this work, we consider the setting where we have a large number of dialogue episodes of deployment-time dialogue which consist of natural conversations between the model and organic users. We want to see if we can obtain any implicit signal from these organic user conversations, and leverage these signals to improve the dialogue model. The rationale is two-fold. First, the organic users most closely approximate the data distribution for future deployment; yet they may not provide explicit annotations. Second, relying on these implicit signals does not incur extra cost

that would otherwise be spent on crowdsourcing. More specifically, in this work we investigate the following: *can we improve the chatbot by optimizing for simple **implicit** feedback signals like the number of, length, sentiment, or reaction of future human responses?*

We use publicly released de-identified data [Xu et al. 2023a] from the BlenderBot online deployment [Shuster et al. 2022b]. Utilizing this data, we obtain sample-and-rerank models, comparing various implicit feedback signals. Through both automatic and human judgments, some of our new models are preferable to baseline responses. Next, as our implicit feedback signals are coarse proxy metrics of the quality of bot turns, we ask whether encouraging these metrics would lead to undesirable behaviors. The answer is yes, depending on the chosen signal: in particular, optimizing for longer conversation lengths can make the model produce controversial takes or respond in an unfriendly or confrontational way. Optimizing for positive reaction or sentiment on the other hand has *the opposite* effect, and decreases these behaviors compared to the baseline. Overall, implicit feedback from humans is a useful training signal that can improve overall performance, but the precise signal used has important behavioral consequences.

3.2 RELATED WORK

Researchers and practitioners have strived to build better neural open-domain dialogue models for years [Chen et al. 2017; Gao et al. 2018; Khatri et al. 2018]. DialoGPT [Zhang et al. 2020b] and BlenderBot [Shuster et al. 2022b] have released the models as well as the training pipelines which have enabled follow-up dialogue projects from the community [Bang et al. 2021; Adewumi et al. 2022]. The class of training strategies that are most relevant to this work – decoding utterances for future success – is discussed in Li et al. [2017a], in which they interpolate the MLE-trained token-level conditional probability with a value function that is trained to predict the property of a completed sequence (e.g., length, BLEU/ROUGE against the reference) given a partial sequence. This overall idea is extended in Zemlyanskiy and Sha [2018] where a chatbot

learns to generate utterances that have the maximal information gain about the human in the future, as well as Kulikov et al. [2019] that propose to generate the current bot utterance that leads to the most probable sequence of future utterances. Irvine et al. [2023] uses conversation engagement metrics (e.g., approximated by retry rate, manually-annotated engagement metrics) to optimize for engaging bot responses; in contrast, our work highlights both the strengths as well as the challenges of using implicit feedback, and in particular that conversation engagement metrics have negative consequences that can be alleviated through other choices of implicit signal.

3.3 APPROACH

3.3.1 IMPLICIT FEEDBACK SIGNALS

Our goal is to extract learning signals from a large set of human-bot conversational episodes. Assume such a set has already been collected. A conversation episode is represented as $\mathbf{x} = (\mathbf{x}_1^b, \mathbf{x}_1^h, \mathbf{x}_2^b, \mathbf{x}_2^h, \dots)$ with T utterances by the bot (denoted with superscript “ b ”; bot is assumed to speak first) and T' utterances by the human (denoted with “ h ”). Let $\mathbf{x}_{<t}$ denote the conversation history before bot’s t -th turn: $\mathbf{x}_1^b, \mathbf{x}_1^h, \dots, \mathbf{x}_{t-1}^b, \mathbf{x}_{t-1}^h$. Next, we define the implicit feedback-derived scoring function $r_\phi(\mathbf{x}_t^b, \mathbf{x}_{<t})$ that predicts the quality of the bot’s t -th turn \mathbf{x}_t^b given past utterances; the output is a score in $[0, 1]$ representing the probability that \mathbf{x}_t^b is good, according to one of the below criteria. Crucially, for the training data (but not for test data) we have access to entire conversation \mathbf{x} (with $T + T'$ utterances for a given episode). We can hence use future human turns to gather implicit feedback to judge the quality of \mathbf{x}_t^b , which we hence use to define training labels $y(\mathbf{x}_t^b)$ in order to learn r_ϕ . We consider several candidate implicit signals, which we describe next – these signals are *coarse proxy* metrics of the quality of bot turns, and we aim to investigate the effect for optimizing them.

Existence of next human turn. Intuitively, if the human user quits the conversation after

the bot’s t -th turn \mathbf{x}_t^b , then *it is likely* that \mathbf{x}_t^b is of poor quality. Conversely if humans continue to converse, and do not quit, this prolonged engagement can be seen as a proxy for satisfaction [O’Brien and Toms 2008]. Therefore, we set the reference label $y(\mathbf{x}_t^b)$ for training $r_\phi(\mathbf{x}_t^b, \mathbf{x}_{<t})$ to 1 if the next human turn exists, and 0 otherwise. We use “replied” to represent this signal in later sections.

Next human turn length. If a human is unwilling to invest time into the conversation, their responses may be shorter. Given the crude intuition that a long human turn *likely* implies that the previous bot turn is good, let $y(\mathbf{x}_t^b)$ be 1 if the next human turn has $\geq k$ words (k is a hyperparameter); 0 otherwise. Granted, the intuition is not always true in practice (e.g., a human response could be a tirade against previous bot turns); we only use the signals in this section as *coarse proxy* metrics of bot turn’s quality. We use “length” to represent this signal. In the same vein, we have also attempted to leverage the **number of words in all future human utterances** or **number of future human turns** – we leave this discussion to [Appendix A.1](#) as we are not able to train an effective scoring function.

Sentiment in the next human utterance. We use a recent positive/neutral/negative sentiment model trained on tweets [Camacho-collados et al. 2022]. Intuitively, we want humans to react positively in future responses. For sentiment and reaction signals, we find that the classifiers struggle at classifying very short utterances. At the same time, very short human responses likely mean that humans are unwilling to meaningfully engage. We thus experiment with two options: (1) Set reference label $y(\mathbf{x}_t^b)$ to 1 if sentiment of \mathbf{x}_t^h is positive or neutral, and length is ≥ 5 words; 0 otherwise. (2) Set reference label to 1 if sentiment is positive and length is ≥ 5 words; 0 otherwise.

Reaction in the next human utterance. We use an existing model [Hartmann 2022] finetuned from DistilRoBERTa [Sanh et al. 2019] with output categories: anger, disgust, fear, joy, neutral, sadness, and surprise. Similar to the previous paragraph, we train a classifier that predicts whether the human next turn would have the “joy” reaction and ≥ 5 words at the same time.¹ Let

¹We also tried the following: the classifier predicts whether the human next turn’s top predicted reaction is

$y(x_t^b) = 1$ if the reaction of x_t^h is joy and length is ≥ 5 words; 0 otherwise. This signal is denoted by “joy & length”.

3.3.2 MODELS USING IMPLICIT SIGNALS

We use the sample-and-rerank approach, which has been shown to perform similarly (albeit with a larger computation cost which is not the focus of our discussion) as RL-based approaches in machine translation [Pang et al. 2022a] and learning from pairwise feedback in language modeling [Dubois et al. 2023]. Given a conversation history, first, sample 20 candidate responses. We use factual-top- p sampling [Lee et al. 2022] given that Shuster et al. [2022b] have shown that it achieves a good balance between generation diversity and factuality for social conversations. Next, rerank these generations using a reranker model, i.e., the classifier r_ϕ trained using the deployment data with implicit feedback labels y . We then pick the candidate generation with the highest reranker score.

3.4 EXPERIMENTS AND RESULTS

3.4.1 EXPERIMENTAL SETUP

We base our experiments off the publicly released BlenderBot deployment data [Xu et al. 2023a], in order to build our implicit feedback models. The dataset used in this work contains 3.1M bot utterances and 3.1M human utterances collected from August 2022 to January 2023. The classifiers (i.e., implicit feedback-based rerankers) are based on a pretrained RoBERTa-large. Our baseline model is the publicly released BlenderBot model (r2c2_blenderbot_3B) with around 3B parameters, pretrained on both dialogue and language modeling tasks, and fine-tuned on dialogue tasks [Shuster et al. 2022a]. We also report results for the method “ranked by probability.” We

anger/disgust or non-anger/disgust, but we find that this feature cannot be well-predicted (dev accuracy $\sim 55\%$).

Table 3.1: Performance of dialogue models corresponding to different signals, under various metrics. Columns 2–3: Evaluation of generated dialogue responses using different implicit feedback signals. Win rate evaluated by crowdworkers: given “baseline generation wins” for $a\%$ examples, “new generation wins” for $b\%$, “tie” for $c\%$, the win rate is $b - a\%$. Sig.: ** if p -value $\in [0, 0.05)$, * if p -value $\in [0.05, 0.1)$, – otherwise. Columns 4–9: various measured properties of the generations (Subsection 3.4.2). Please refer to Table A.1 and Appendix A.3.2 for complementary details (e.g., human annotation win/lose/tie results, LLM-evaluated win/lose/tie results, avg. length of generations).

	% win rate	sig.	% seek info	% off-topic	% off-topic & seek info	% insincere	% contro- versial	% un- friendly
baseline	–	–	32.5	11.5	3.0	20.0	17.0	9.0
ranked by probability	+3.0	–	43.0	13.5	4.0	16.0	16.0	7.0
replied	–1.0	–	47.5	16.0	5.0	21.0	24.5	12.5
length ($k=20$)	+12.0	**	46.0	15.0	4.5	20.0	17.0	12.5
length ($k=5$)	+5.0	–	56.0	13.0	8.0	19.0	19.0	9.5
non-neg. sentiment & length ($k=5$)	+8.5	*	60.0	14.5	8.0	21.0	13.0	6.0
positive sentiment & length ($k=5$)	+6.5	–	41.0	11.0	3.5	20.0	9.5	6.0
joy & length ($k=5$)	+9.5	**	49.0	12.0	8.0	22.5	8.5	6.0

simply rerank using the sequence-level probabilities during sample-and-rerank; we want to see whether our approaches based on the implicit feedback classifiers outperform using this naive ranking criteria.

3.4.2 EVALUATION METHODS

Given a conversation history and two candidate responses (baseline and new model responses), we ask a large language model (LLM), in this case gpt-3.5-turbo-0613, to judge which one of the two responses is better or if they tie, with 8-shot chain-of-thought (CoT) prompts. Experts (the authors of this paper) also carefully annotate 200 comparisons. We find that LLM vs. expert example-based agreement is not high; see Appendix A.3.3 for more details. Therefore, we conduct human annotation via crowdworkers, using majority vote over 5 workers per comparison,² with

²The final answer is the majority vote. If there is no majority vote (e.g., if five votes are “(a) wins,” “(a) wins,” “(b) wins,” “tie,” “tie”), then the final answer is “(a) and (b) tie.”

10% catch questions with known unambiguous answers to filter for quality. We find that the human annotation vs. expert agreement is much higher than LLM vs. expert. But we do find general agreement between crowdworkers and LLM evaluation at the level of averaging over many examples. See [Appendix A.3](#) for more details on human annotation and comparison with LLMs.

BEHAVIORS OF GENERATED RESPONSES. We also investigate what behaviors (including potentially undesirable ones) the generated responses have. The properties are as follows. **Seek info:** whether the response is seeking information (e.g., “tell me about the dune”); **off-topic:** whether the response is off-topic and irrelevant to the conversation; **controversial:** whether the response contains anything controversial; **insincere:** whether the response is insincere (being deceitful, not being genuine, not being serious about the conversation); **unfriendly:** whether the response is being unfriendly or confrontational toward the other speaker. We use gpt-3.5-turbo-0613 (with 8-shot CoT prompting) to conduct evaluation. These questions are intuitively straightforward, and we observe that the LLM-expert evaluation outputs match >90% of the time. The exact prompts are provided in [Appendix A.3.4](#).

3.4.3 RESULTS

OVERALL RESULTS. Overall results are given in [Table 3.1](#). Annotators find that several of the implicit feedback signals outperform the baseline and the “ranked by probability” method (more in [Appendix A.3](#)). In particular, “length ($k=20$),” “non-neg. sentiment & length,” and “joy & length” are all significantly better than the baseline using Wilcoxon signed-rank test. For example, responses generated using the “length ($k=20$)” signal correspond to a 12-point lead compared to the baseline responses, and the “joy & length” signal corresponds to an 9.5-point lead. We also find that LLM-based evaluation follows roughly the same trend as human annotators; see further supporting results in [Appendix A.3](#).

BEHAVIOR ANALYSIS. While several choices of implicit feedback improve overall performance, we observe both positive and negative consequences in terms of observed behavior depending on the implicit signal chosen. Evaluation of behavioral properties is provided in [Table 3.1](#) columns 4–9.

Implicit signals that approximately optimize conversation length (“replied”, “length ($k=5$)”, “length ($k=20$)”) tend to increase the amount of **controversial** and/or generations that are deemed **unfriendly**. In contrast, positive sentiment and joy optimizing signals (“sentiment & length”, “joy & length”) tend to *decrease* both of these behaviors compared to the baseline. The “replied” signal produces the most controversial messages – possibly to provoke the user into responding one more time. The “length ($k=20$)” and “replied” signals lead to a larger number of unfriendly generations, possibly by antagonizing the other speaker so they are too provoked to not respond. The “joy & length” signal on the other hand halves the amount of controversial messages (from 17% to 8.5%) compared to the baseline, avoiding these types of messages.

We also observe that most implicit signals lead to an increased amount of **information seeking**. Further, some signals, especially for “replied” and “length ($k=20$)”, may go **off-topic** at a slightly higher rate than the baseline. For generations using signals “length ($k=5$)” and “non-neg. sentiment & length,” there is a much higher rate in seeking off-topic information; a possible explanation is that the model could ask slightly irrelevant questions so as to keep the human user engaged.

3.5 CONCLUSION

In summary, we find that optimizing for implicit feedback signals from human responses is effective, providing improved models over the baseline. However, the choice of implicit signal to extract has important behavioral consequences. Conversation length-based signals tend to increase controversial and unfriendly messages, while sentiment or reaction-based signals tend

to do the opposite, decreasing the frequency of this behavior compared to the baseline. We note, however, that if we discount generations that are off-topic, controversial, unfriendly, or insincere, and only evaluate on the rest of the examples, then the human annotation would prefer our implicit feedback models over the baseline even more (see [Appendix A.3.2](#)). Hence, future work could try to extract signals towards that goal, or consider additional safeguards or mitigations.

3.6 LIMITATIONS

While we provide no formal evaluation, decreasing controversial messages potentially prevents the discussion of serious matters, for example, sharing indignance on issues like social justice or discussing unfortunate everyday situations. On the other hand, encouragement of these messages increases the chance of upsetting conversations or even harmful conversations.

Algorithm-wise, while we have used the sample-and-rerank in our experiments, a natural extension which we did not explore in this project is to use implicit signals in other learning approaches such as RL. To use RL, we may need strategies to reduce reward gaming behaviors in text generation [[Skalse et al. 2022](#); [Pang et al. 2023](#)] given that our classifiers are imperfect. Another future direction which we did not explore in this project is to study the use of implicit feedback signals in an iterative framework, whereby the new improved model is re-deployed and feedback recollected. For example, we find many of the implicit feedback models we explored increase information-seeking messages, which is not always beneficial [[Dinan et al. 2020](#)]. If those methods have overcompensated and now produce an excessive amount of such messages, redeployment can provide feedback to correct this issue and iteratively improve the model.

4 | REWARDS CAN BE BASED ON CLASSIC NLP ALGORITHMS: AMORTIZED NOISY CHANNEL NEURAL MACHINE TRANSLATION

Is there a way to obtain a denser and high-quality reward for NMT? We can get inspired by classic NLP algorithms, in particular noisy channel decoding based on which we can define a reward function. The byproduct is that we are increasing decoding speed by 1–2 orders of magnitude compared to the canonical beam-search-and-rerank (BSR) decoding-time idea. Performance-wise, we obtain lower rewards compared to BSR, but the generations have similar BLEU and BLEURT scores compared to those by BSR. This chapter is based on work with He He and Kyunghyun Cho [Pang et al. 2022a].

4.1 INTRODUCTION

Noisy channel models have been traditionally used in many tasks, including speech recognition [Jelinek 1997], spelling correction [Brill and Moore 2000], question answering [Echihabi and Marcu 2003], and statistical machine translation [Koehn et al. 2003]. In machine translation (MT),

the probability of the source sentence conditioned on the target-language generation is taken into account when generating a translation. In modern neural machine translation (NMT), the noisy channel approach is successful and often indispensable in many recent top-performing machine translation systems [Yee et al. 2019; Ng et al. 2019; Chen et al. 2020; Yu et al. 2020; Tran et al. 2021].

One widely used approach of noisy channel NMT is “beam search and rerank” (BSR). Assume a trained forward translator and a trained reverse translator,¹ BSR decoding consists of two steps: first, decode using beam search with a large beam size from the forward translation model and store the entire beam; second, rerank the beam using a reward which is the sum of the forward translation log probability and the reverse log probability. This approach incurs significant computational overhead, given the need to decode a large beam (usually with beam size 50–100) from the forward translator and the need to feed the large beam through the reverse translator. The computational cost is especially problematic if the practitioner has a large volume of translation requests, or if the system is mobile-based and requires offline translation.

We thus aim to learn a separate neural network with an identical architecture as the forward translator such that at inference time, when we do *greedy decoding* using this new network, we investigate how much translation accuracy would be sacrificed. Specifically, we investigate how forward/reverse rewards of the translations as well as the translation quality (approximated by BLEU and BLEURT) would compare to those of BSR-generated translations.²

The paper explores three approaches, with increasingly more exploration when optimizing the reward. (1) Knowledge distillation (KD) from a pseudo-training-corpus generated by BSR: we can treat the BSR-generated corpus as the oracle, and KD can be interpreted as behavioral cloning. (2) a one-step-deviation imitation learning strategy (IL) where given a fixed sequence of target-language tokens, we adjust each time-step’s probability distribution over the vocabulary

¹Forward: from the source language to the target language; reverse: from the target language to the source language.

²Although we need time to train the separate network, at inference time and during the actual large-scale user-facing deployment, we would be able massively cut down computational cost in the long run. In this paper, we aim to investigate the accuracy of such decoded translations.

such that the resulting distribution minimizes an energy function used in BSR reranking, and (3) Q learning which explicitly learns the scoring function used in BSR reranking.

We experiment on three datasets (IWSLT’14 De-En, WMT’16 Ro-En, and WMT’14 De-En). Experimental results show that all three approaches speed up inference by 50–100 times. The approaches fail to achieve comparable rewards to BSR, but compared to the non-BSR baselines, the approaches achieve much higher reverse rewards (i.e., $\log p_r(\mathbf{x} \mid \mathbf{y})$ where p_r is the reverse translator) at the expense of forward rewards (i.e., $\log p_f(\mathbf{y} \mid \mathbf{x})$ where p_f is the forward translator). Meanwhile, the approaches achieve a translation quality (approximated by BLEU and BLEURT) that is comparable to that of BSR. In particular, IL’s BLEURT scores is significantly higher than those of beam search, across all three datasets; IL’s BLEURT scores are not significantly different from BSR’s scores, across three datasets.

4.2 BACKGROUND

4.2.1 NEURAL MACHINE TRANSLATION

NMT systems usually model the distribution $p(\mathbf{y} \mid \mathbf{x})$ where $\mathbf{x} = (x_1, x_2, \dots, x_{T_s})$ is a source-language sequence and $\mathbf{y} = (y_1, y_2, \dots, y_T)$ is a target-language sequence. Most NMT systems use an autoregressive factorization:

$$\log p(\mathbf{y} \mid \mathbf{x}) = \sum_{t=1}^T \log p_\theta(y_t \mid \mathbf{y}_{<t}, \mathbf{x}),$$

where $\mathbf{y}_{<t} = (y_1, y_2, \dots, y_{t-1})$, and p_θ is parameterized with a neural network. At test-time, to decode a translation given a source sentence, greedy decoding and beam search are most commonly used. Both are approximate search methods to find the highest-scoring translations.

4.2.2 BEAM SEARCH AND RERANK (BSR)

BSR has appeared in a number of top-performing models, including many winning submissions of the WMT competitions [Ng et al. 2019; Chen et al. 2020; Yu et al. 2020; Tran et al. 2021]. The intuition of BSR is to take advantage of the reverse translator during decoding. Specifically, we do beam search with a large beam size b (usually 50–100) to obtain b candidate translations. Then, we rerank the candidates using the scoring function:

$$\log p_f(\mathbf{y} | \mathbf{x}) + \gamma \log p_r(\mathbf{x} | \mathbf{y}) + \gamma' \log p_{lm}(\mathbf{y}),$$

where γ and γ' are tuned in $[0, 2]$. Without access to a language model trained on a huge target-language monolingual external corpus, if we use $\log p_f(\mathbf{y} | \mathbf{x}) + \gamma \log p_r(\mathbf{x} | \mathbf{y})$ as the ranking criteria, BSR also provides a significant performance gain. With a large beam size, this approach performs better than the “two-step beam search” approach [Yu et al. 2017a; Yee et al. 2019].

4.3 AMORTIZED NOISY-CHANNEL NMT

One common problem with the above approaches is the inference-time computation overhead. If a translation system needs to translate a high volume of texts, then the test-time computational efficiency is crucial. Thus, our goal is to use a network to approximate such a noisy channel NMT system, while having the same inference-time computational cost as *greedily decoding from p_f* . Specifically, we want our translations to maximize the following objective:

$$R(\mathbf{x}, \mathbf{y}) = \log p_f(\mathbf{y} | \mathbf{x}) + \gamma \log p_r(\mathbf{x} | \mathbf{y}), \tag{4.1}$$

where $\gamma > 0$ is some fixed coefficient. Using the autoregressive factorization, the forward reward $\log p_f(\mathbf{y} | \mathbf{x})$ equals $\sum_{t=1}^{|\mathbf{y}|} \log p_f(\mathbf{y}_t | \mathbf{y}_{<t}, \mathbf{x})$, and the reverse reward $\log p_r(\mathbf{x} | \mathbf{y})$ equals

$$\sum_{t=1}^{|\mathbf{x}|} \log p_r(\mathbf{x}_t \mid \mathbf{x}_{<t}, \mathbf{y}).$$

GOAL: INVESTIGATING IF GREEDILY DECODING FROM OUR NEW MODELS LEADS TO SUCCESSFUL AMORTIZATION. Three approaches are shown in this section. We do greedy decoding from the obtained models, and we investigate whether amortization is successful as follows.

- First, we examine if decoding is faster than BSR. This aspect is guaranteed given that we would do greedy decoding from our new network which has the same architecture as p_f .
- Next, we examine if both the forward and reverse rewards of the translations are close to the forward and reverse rewards of the translations generated by BSR, respectively.
- Finally, we examine the translation quality by checking if BLEU and BLEURT scores of our model’s translations are close to those of BSR-produced translations.

4.3.1 APPROACH 1: KNOWLEDGE DISTILLATION (KD)

KD has been used to amortize beam search [Chen et al. 2018]. It is also effective in NMT in general [Kim and Rush 2016; Freitag et al. 2017; Tan et al. 2019; Tu et al. 2020b]. Here we adapt a simple version of KD for amortized noisy-channel decoding.

First, train a forward translator p_f and a reverse translator p_r using maximum likelihood estimation. Then, do BSR on the entire training set to obtain the pseudo-corpus. In particular, we ignore the p_{lm} term in this paper given that it usually requires a big language model, and the inclusion of the term is orthogonal to our goal of reducing inference time.³

Next, we train a separate “knowledge distilled” model p_{KD} on this new pseudo-corpus (i.e., with the original source-language sentences and the BSR-generated target-language sentences).

³Generating the pseudo-corpora can be paralleled. If the system is deployed in the real world, we argue that the amount of computation used to generate the pseudo-corpus is negligible, compared to the aggregate amount of computation for inference.

This objective is equivalent to minimizing the KL-divergence between the distribution induced by the pseudo-corpus obtained through BSR and our model distribution.

At inference time, we greedily decode from p_{KD} .

4.3.2 APPROACH 2: 1-STEP-DEVIATION IMITATION LEARNING (IL)

Define a network A_ϕ such that it takes in the source sentence and a target-language prefix, and $A_\phi(\cdot | \mathbf{x}, \mathbf{y}_{<t})$ outputs a $|\mathcal{V}|$ -dimensional probability distribution corresponding to the t -th time-step. Moreover, A_ϕ and p_f have the same architecture. In autoregressive text generation, to learn A_ϕ such that it is close to an existing network p_θ , imitation learning seeks to optimize ϕ as follows:

$$\arg \min_{\phi} \mathbb{E}_{(\mathbf{x}, \mathbf{y}_{<t})} \mathcal{L}(A_\phi(\cdot | \mathbf{x}, \mathbf{y}_{<t}), p_\theta(\cdot | \mathbf{x}, \mathbf{y}_{<t})),$$

where one example of \mathcal{L} is the cross entropy.

FORWARD ENERGY. Inspired by ENGINE [Tu et al. 2020b], in the context of noisy channel NMT, define the forward sub-energy E_t^f , which is a function of ϕ , as follows:⁴

$$E_t^f(\mathbf{x}, \hat{\mathbf{y}}; \phi) = -A_\phi(\cdot | \mathbf{x}, \hat{\mathbf{y}}_{<t})^\top \log p_f(\cdot | A_\phi(\cdot | \mathbf{x}, \hat{\mathbf{y}}_{<1}), \dots, A_\phi(\cdot | \mathbf{x}, \hat{\mathbf{y}}_{<t}), \mathbf{x}).$$

Suppose we have a source sentence \mathbf{x} and a sequence of prefix distributions $\hat{\mathbf{y}}_{<1}, \dots, \hat{\mathbf{y}}_{<T}$. We call $A_\phi(\cdot | \mathbf{x}, \hat{\mathbf{y}}_{<t})$ the t -th step distribution according to A_ϕ . Intuitively, given a source and a fixed sequence of prefixes, we learn A_ϕ such that the resulting t -th-step distribution matches the forward conditional probability (measured by p_f) – the latter depends on the source \mathbf{x} and the prefix distributions.

⁴If we compute $p_f(\cdot | y_1, y_2, \dots, y_{t-1}, \mathbf{x})$ where y_i 's correspond to token IDs of a partial translation in the target language, then we would first look up the y_i -th row of the embedding matrix E_{emb} and use this row to embed y_i ; equivalently, we can use the product $\text{onehot}(y_i)^\top E_{emb}$ to embed y_i . In the case of E_t^f , the prefixes used in p_f are distributions instead of tokens. We can use $A_\phi(\cdot | \mathbf{x}, \hat{\mathbf{y}}_{<i})^\top E_{emb}$ to represent the i -th token embedding. The embedding strategy is similar for E_t^f below.

REVERSE ENERGY. Next, we define the reverse sub-energy as follows:

$$E_t^r(\mathbf{x}, \hat{\mathbf{y}}; \phi) = -\text{onehot}(\mathbf{x}_t)^\top \log p_r(\cdot \mid \mathbf{x}_{<t}, A_\phi(\cdot \mid \mathbf{x}, \hat{\mathbf{y}}_{<1}), \dots, A_\phi(\cdot \mid \mathbf{x}, \hat{\mathbf{y}}_{<T})).$$

Intuitively, we also learn A_ϕ such that the one-hot distributions corresponding to the source words should match the reverse conditional probability (measured by p_r).

TRAJECTORIES. In the above equations, $\hat{\mathbf{y}} = (\hat{y}_1, \dots, \hat{y}_T)$. \hat{y}_t comes from two sources, with probability p and $1 - p$ for each minibatch during training (Subsection 4.4.2): (i) $\hat{y}_t = \arg \max_{v \in \mathcal{V}} A_\phi(v \mid \mathbf{x}, \hat{\mathbf{y}}_{<t})$ and $\hat{\mathbf{y}}_{<1} = \emptyset$; in other words, given that $A_\phi(\cdot \mid \mathbf{x}, \hat{\mathbf{y}}_{<t})$ is a probability distribution, we use the most likely token as \hat{y}_t . (ii) For the second source, let \hat{v}_t be the t -th token of the BSR-obtained sequence, so that we can expose our model to BSR-prefixes, which are the optimal prefixes.

FINAL OBJECTIVE. We train A_ϕ using the following objective:

$$\min_{\phi} \sum_{\mathbf{x}} \left[\sum_{t=1}^T E_t^f(\mathbf{x}, \hat{\mathbf{y}}; \phi) + \gamma \sum_{t'=1}^{|\mathbf{x}|} E_{t'}^r(\mathbf{x}, \hat{\mathbf{y}}; \phi) \right].$$

During inference, we greedily decode from A .

4.3.3 APPROACH 3: Q LEARNING

A well-motivated approach is to use Q learning [Watkins and Dayan 1992; Sutton and Barto 1998] to explicitly learn a reward function Q , with the goal that when we greedily decode from Q , the generations maximize the reward shown in Eq. (4.1).

Let us view machine translation as a sequential decision-making process. At time-step t , given a state $s_t = (\mathbf{y}_{<t}, \mathbf{x})$, a policy takes an action $a_t \in \mathcal{V}$, transits to the next state $s_{t+1} = (\mathbf{y}_{<(t+1)}, \mathbf{x})$,⁵ and receives a reward r_t .

⁵ $\mathbf{y}_{<(t+1)}$ equals $\mathbf{y}_{<t}$ concatenated with the action a_t .

4.3.3.1 BACKGROUND ON Q LEARNING

In Q learning, $Q^\pi : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$ is a function such that $Q^\pi(s_t, a_t)$ produces the expected return after seeing state s_t , taking action a_t , and following policy π ; i.e., $Q^\pi(s_t, a_t) = \mathbb{E}[\sum_{t'=t}^{\infty} r_{t'} | s_t, a_t, \pi]$ assuming discount factor 1. We further define $Q^* : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$ to be the optimal action-value function: $Q^*(s_t, a_t) = \max_{\pi} \mathbb{E}[\sum_{t'=t}^{\infty} r_{t'} | s_t, a_t, \pi]$, which is the maximum return achievable by following any strategy *after* seeing a state s_t and taking an action a_t . In particular, Q^* solves the Bellman Equation [Sutton and Barto 1998]:

$$Q^*(s_t, a_t) = r_t + \max_{a_{t+1}} Q^*(s_{t+1}, a_{t+1}),$$

assuming discount factor 1 and given deterministic transition dynamics (in our machine translation scenario) after taking action a_t given state s_t .

Traditionally, the Q function is implemented as a matrix of size $|\mathcal{S}| \times |\mathcal{A}|$, which is intractable in the case of MT due to the combinatorial nature of the state space. We thus use function approximation to tackle this issue of intractability: we follow Mnih et al. [2015] and use a deep neural network trained with experience replay and target networks to approximate the Q learning.

Deep Q learning draws samples from a set of trajectories \mathcal{B} , and the neural network Q aims to predict Q^* by learning based on minimizing the following squared loss.

$$L(\phi) = \frac{1}{|\mathcal{B}|} \sum_{(s_t, a_t, s_{t+1}, r_t) \sim \text{Uniform}(\mathcal{B})} \left[\left(r_t + \max_{a_{t+1}} Q'(s_{t+1}, a_{t+1}) - Q(s_t, a_t) \right)^2 \right],$$

where ϕ is the parameter to Q , and Q' is a slightly old copy of Q .⁶

⁶In other words, after a fixed number of optimization steps, we update Q' by Q .

Algorithm 1: Q learning for amortized noisy channel NMT

Given p_f, p_r , and a parallel translation dataset \mathcal{D} .

while *not converged* **do**

Collect training trajectories (§4.3.3), and sample a mini-batch \mathcal{B} .

Compute target R_t :

- if $t < T$, then $R_t = r_t + \max_{a_{t+1}} Q'_\phi(s_{t+1}, a_{t+1})$;
- if $t = T$, then $R_t = r_T$.

Update ϕ (using gradient descent) by the objective

$$\arg \min_{\phi} [Q_\phi(s_t, a_t) - R_t]^2.$$

Update Q'_ϕ : $Q'_\phi \leftarrow Q_\phi$ every K steps.

end

4.3.3.2 Q LEARNING FOR AMORTIZED NOISY CHANNEL MT

To model the noisy-channel NMT, given a target-language sequence \mathbf{y} and its length T , we have reward $\mathbf{r} = (r_1, \dots, r_T)$, where

$$r_t = \begin{cases} \log p_f(y_t | \mathbf{y}_{<t}, \mathbf{x}), & \text{if } t < T, \\ \log p_f(y_T | \mathbf{y}_{<T}, \mathbf{x}) + \gamma \cdot \log p_r(\mathbf{x} | \mathbf{y}), & \text{if } t = T. \end{cases} \quad (4.2)$$

We construct Q to have the same architecture as p_f without the final softmax layer. Q is trained using Algorithm 1 which is adapted from deep Q learning originally applied to Atari games [Mnih et al. 2015], given that we aim to best leverage the existing off-policy trajectories from different sources. The full algorithm is shown in Algorithm 1.

In short, our algorithm says that given a trajectory $(\mathbf{x}, \mathbf{y}, \mathbf{r})$, at time-step $t < T$, we want the scalar $Q(s_t, a_t)$ to be close to the sum of the t -th step reward and the *most optimistic* future return, had we taken action a_t at time-step t . At time-step T , we want $Q(s_T, a_T) = Q((\mathbf{y}_{<T}, \mathbf{x}), \langle \text{eos} \rangle)$ to be close to r_T , as defined in Eq. (4.2).

To generate the t -th token at inference-time, we do greedy decoding using Q as follows:

$$\hat{y}_t = \arg \max_{a_t \in \mathcal{V}} Q(s_t, a_t). \quad (4.3)$$

TRAJECTORIES. The off-policy algorithm shown in Algorithm 1 requires trajectories, i.e., $(\mathbf{x}, \mathbf{y}, \mathbf{r})$ tuples. The trajectories come from two sources.

(1) Q -BASED TRAJECTORIES. In this category, we have two ways of obtaining \mathbf{y} : (1a) Boltzmann exploration [Sutton 1990]⁷ and (1b) greedy decoding based on Q . At the start of the optimization, however, most of the Q -generated sequences are very far from target sequences. The lack of high-reward sequences prevents Q learning from efficient optimization. Therefore, we also inject reasonably good trajectories from the beginning of training by utilizing both ground-truth sequences as well as p_f -based sequences. We thus need the next category of sources.

(2) p_f -BASED TRAJECTORIES. The target-language sequences are obtained by decoding using p_f ; please find more details in Appendix B.1.1.⁸

4.4 EXPERIMENTAL SETUP

4.4.1 TASKS AND MODELS

We experiment on three translation tasks: IWSLT 2014 German to English [IWSLT’14 De-En; Cettolo et al. 2014] which has a small training set (train/dev/test size: 160,239/7,283/6,750), WMT 2016 Romanian to English [WMT’16 Ro-En; Bojar et al. 2016] which has a medium-sized training set

⁷Recall that at time-step t , $Q(s_t, a_t) \in \mathbb{R}$ for each $a_t \in \mathcal{V}$. Therefore, $Q(s_t, \cdot) \in \mathbb{R}^{|\mathcal{V}|}$. We turn the vector of real numbers to a categorical distribution by softmax with temperature γ_b . Then, the sequences in the trajectories are obtained by sampling from the aforementioned distribution. In practice, for each sequence, we use a temperature γ_b sampled from Uniform(0, 1.5). One can think of this strategy as a variant of ϵ -greedy which is typically used in Q learning.

⁸We have also experimented with gold-standard trajectories from the parallel translation dataset \mathcal{D} , but the inclusion of such trajectories do not lead to better rewards of Q -generated translations.

Table 4.1: Mean and standard deviation (across sequences) of test set forward and reverse rewards for translations. b refers to beam size during inference.

	IWSLT'14 De-En			WMT'16 Ro-En			WMT'14 De-En		
	b	fwd reward mean (std)	rvs reward mean (std)	b	fwd reward mean (std)	rvs reward mean (std)	b	fwd reward mean (std)	rvs reward mean (std)
p_f	1	-9.1 (7.7)	-35.4 (39.9)	1	-9.5 (11.5)	-41.0 (50.1)	1	-11.0 (6.3)	-31.5 (24.6)
p_f	5	-8.6 (7.0)	-34.2 (38.5)	5	-9.0 (8.5)	-40.2 (48.2)	7	-10.4 (5.5)	-29.9 (21.5)
BSR	100	-9.4 (6.8)	-25.7 (32.5)	70	-10.0 (6.0)	-29.7 (41.9)	50	-10.7 (5.3)	-23.6 (16.3)
KD	1	-13.8 (13.9)	-28.0 (32.7)	1	-17.2 (26.3)	-35.4 (44.6)	1	-14.8 (9.1)	-24.0 (16.7)
IL	1	-13.3 (13.2)	-27.9 (32.3)	1	-17.2 (30.9)	-34.3 (45.3)	1	-14.6 (8.9)	-23.6 (15.9)
Q learning	1	-13.7 (21.4)	-29.9 (35.1)	1	-11.6 (19.7)	-39.1 (52.9)	1	-14.4 (9.9)	-24.9 (17.5)
reference data	-	-38.8 (39.7)	-45.2 (46.6)	-	-55.3 (51.1)	-59.0 (54.2)	-	-36.8 (24.4)	-36.8 (23.0)

Table 4.2: Test set sacreBLEU (mean & standard deviation of three runs using different random seeds). IL performs the best among the three proposed methods.

	IWSLT'14 De-En	WMT'16 Ro-En	WMT'14 De-En
p_f (greedy decoding)	33.65 (0.06)	33.23 (0.14)	30.39 (0.13)
p_f (beam search)	34.54 (0.08)	33.98 (0.15)	31.78 (0.08)
BSR	35.43 (0.06)	34.81 (0.09)	32.15 (0.14)
KD	35.39 (0.04)	33.95 (0.10)	31.71 (0.05)
IL	35.61 (0.09)	34.65 (0.07)	31.90 (0.07)
Q learning	34.60 (0.08)	34.31 (0.15)	31.60 (0.19)

Table 4.3: Test set BLEURT-20-D12 (mean & standard deviation of three runs). IL performs the best among the three proposed methods. Significance test is conducted in Table B.2, which shows that IL's scores are significantly better than the scores by beam search; in addition, IL's scores are not significantly different from BSR's scores.

	IWSLT'14 De-En	WMT'16 Ro-En	WMT'14 De-En
p_f (greedy decoding)	62.40 (0.04)	61.14 (0.10)	64.83 (0.10)
p_f (beam search)	63.21 (0.07)	61.42 (0.15)	65.79 (0.08)
BSR	64.15 (0.05)	62.67 (0.13)	66.32 (0.12)
KD	63.88 (0.04)	61.78 (0.10)	66.00 (0.07)
IL	63.94 (0.13)	62.35 (0.16)	66.14 (0.08)
Q learning	63.25 (0.07)	61.70 (0.18)	65.92 (0.14)

(train/dev/test size: 608,319/1,999/1,999), and WMT 2014 German to English [WMT’14 De-En; Bojar et al. 2014] which has a moderately large training set (train/dev/test size: 4,500,966/3,000/3,003). Each of the transformer models (the p_{KD} in KD, the A in IL, the Q function in Q learning) has the same number of parameters as the original MLE-trained forward translator p_f . The model for IWSLT’14 De-En is the smallest, and the model for WMT’14 De-En is the largest. The detailed settings can be found in [Appendix B.2](#). BLEU scores in this paper are computed with sacreBLEU [Post 2018]. BLEURT scores are computed using BLEURT-20-D12 [Sellam et al. 2020a], a recent RemBERT-based checkpoint that achieves high human agreement. The models we experiment on are shown in [Table 4.1](#).

4.4.2 HYPERPARAMETERS

The architecture and optimization details of p_f and p_r are shown in [Appendix B.2](#). When training p_f and p_r , we validate the model performance after each epoch, and select the model that corresponds to the best dev set BLEU.

γ is the coefficient multiplied to the reverse reward, when computing the total reward in Eq. (4.1); γ and BSR beam size b are tuned on dev set BLEU using BSR. We choose $\gamma = 0.9$ and $b = 100$ for IWSLT’14 De-En; $\gamma = 0.5$ and $b = 70$ for WMT’16 Ro-En; $\gamma = 0.5$ and $b = 50$ WMT’14 De-En. See [Appendix B.2](#) for details.

For training the IL-based network, the learning rate is selected from $\{10^{-6}, 5 \times 10^{-6}, 10^{-5}, 3 \times 10^{-5}, 5 \times 10^{-5}\}$. We use weight decay of 10^{-4} . Dropout rate is selected from $\{0, 0.05, 0.1, 0.3\}$; we find that a dropout rate of 0 or 0.05 always works the best. We use a fixed max batch length (i.e., the max number of input tokens in a batch) of 4,096 tokens. The probability p , described in [Section 4.3](#), is selected from $\{0, 0.1, 0.5, 0.9, 1\}$; we find that $p = 0.1$ or $p = 0.5$ usually works the best. We accumulate gradients and do gradient descent once every k steps for computational reasons. k is selected from $\{4, 8, 16\}$. We find that the IL approach relies on a good initialization, so we use $p_{\text{KD}/\text{nc}}$ to initialize the new network.

For Q learning, the synchronization frequency K in Algorithm 1 is selected from $\{10, 20, 30, 50, 150\}$. The learning rate is tuned in $\{10^{-5}, 3 \times 10^{-5}, 5 \times 10^{-5}, 10^{-4}\}$. We use weight decay of 10^{-4} . Dropout rate is tuned in $\{0, 0.01, 0.05, 0.1\}$; we find that a dropout rate of 0 always works the best. We use a fixed max batch length 4096. We tune the number of steps per gradient update in $\{4, 8, 16\}$; a large number effectively increases the batch size. The ratio for different trajectories is described in Appendix B.1.1. Furthermore, we find that training Q with a small γ at the beginning stabilizes the training, so we first use $\gamma = 0.1$ and train till convergence, and then increase γ by 0.2 increment, and we reiterate the process until reaching the desired γ .

We use the Adam optimizer [Kingma and Ba 2014] for all experiments. We cap the maximum length of the translation at $1.2T_s + 20$ during decoding, where T_s is the length of a source sentence. All implementation is based on fairseq [Ott et al. 2019]. Each experiment is run on one NVIDIA RTX 8000 GPU.

4.5 RESULTS

4.5.1 PRELIMINARY ANALYSIS

INFERENCE SPEED. Using any of the three proposed approaches achieves a significant speedup, given that the three approaches all use greedy decoding. We quantify this speedup experimentally. During inference, we maximize the memory usage of a single NVIDIA RTX 8000 GPU by finding the largest batch length in the form of 2^k where k is a positive integer.⁹ In the IWSLT’14 De-En task, the inference speed (sequences per second) for BSR is 11. The speed for “greedy by p_f ” is around 1050, and the decoding speed for any of three proposed approaches is also similar.

REWARDS. First, comparing the three approaches to greedy decoding or beam search from p_f , we see that the three approaches achieve smaller forward rewards, but much larger reverse rewards.

⁹Batch length means the number of tokens in a batch (instead of the number of sequences).

This observation is expected given that the three approaches consider both the forward and reverse rewards, while greedy decoding or beam search from p_f only consider forward rewards. Second, comparing the three approaches against BSR, the three approaches achieve both smaller forward rewards and smaller reverse rewards. However, we find this a reasonable trade-off to be made between decoding latency and rewards, as all these approaches are 1–2 orders of magnitude faster in decoding.

Among the three approaches, KD and IL achieve a better balance between forward and reverse rewards. This observation can be explained by the difference in how the reverse reward is presented among the three approaches. In KD and IL, the learning signal by reverse rewards is implicitly spread throughout all the steps in a sequence. In other words, changing the conditional distribution in each time-step would adjust the loss in KD and the reverse energies in IL. In Q learning, the reverse reward is sparse: it only appears at the end of the sequence, unlike the forward reward which is spread throughout all the steps. This makes it easier for Q learning to maximize the forward reward compared to the reverse reward which requires many more updates to be propagated toward the earlier time steps.

TRANSLATION QUALITY. The three approaches achieve BLEU and BLEURT scores that are comparable to those by BSR. Moreover, the three approaches achieve BLEU scores that are much better than “greedy decoding from p_f ” which has the same computational budget; they are often better than “beam search from p_f ” as well. In particular, [Table 4.3](#) shows that IL’s BLEURT scores are significantly higher than the scores of beam search across all three datasets. In addition, IL’s BLEURT scores are not significantly different from BSR across all three datasets. Therefore, our approaches are able to generate translations with similar quality as those by BSR, while being 5–7 times as fast as beam search and 50–100 times as fast as BSR.

Table 4.4: The rewards and BLEU scores using two KD approaches: $p_{\text{KD}/\text{beam}}$ uses the pseudo-corpus generated by doing beam search from p_f . $p_{\text{KD}/\text{nc}}$ uses the pseudo-corpus generated by BSR.

	IWSLT'14 De-En				WMT'16 Ro-En				WMT'14 De-En			
	b	fwd reward mean (std)	rvs reward mean (std)	BLEU	b	fwd reward mean (std)	rvs reward mean (std)	BLEU	b	fwd reward mean (std)	rvs reward mean (std)	BLEU
$p_{\text{KD}/\text{beam}}$ trained by (X, \bar{Y}_{beam})	1	-13.3 (13.4)	-31.6 (35.2)	34.80	1	-17.0 (17.4)	-38.9 (49.0)	33.22	1	-14.7 (9.2)	-28.0 (19.3)	31.38
$p_{\text{KD}/\text{nc}}$ trained by (X, \bar{Y}_{NC})	1	-13.8 (13.9)	-28.0 (32.7)	35.39	1	-17.2 (26.3)	-35.4 (44.6)	33.95	1	-14.8 (9.1)	-24.0 (16.7)	31.71

4.5.2 ANALYSIS OF TRANSLATIONS

In Q learning, the reverse reward is only presented as a learning signal at the end of each sequence. As observed earlier by Welleck et al. [2020a], the length of the generations may inform us of the possible degeneracies, such as excessive repetitions.

Therefore, we analyze WMT'16 Ro-En translations generated by different systems, and we first examine the lengths of translations in different source length buckets. Figure 4.1 shows that the lengths by different systems are similar in the first four buckets, but in the longest source length bucket $(81, \infty)$, Q learning produces longer translations.

Closer examination of the translations reveal that Q learning produces degenerate translations with extensive repetitions when the source sentences are among the longest in the entire dev set; other models do not have this issue. Some randomly selected examples are shown in Table 4.6.

To confirm this finding, we analyze repetitions by source-length buckets. We define “token rep” to be the percentage of tokens that have appeared in the immediately preceding 5-grams:

$$\frac{\sum_{i=1}^N \sum_{t=6}^{T^{(i)}} \mathbb{1} \left[\mathbf{y}_t^{(i)} \in \{\mathbf{y}_{t-5}^{(i)}, \dots, \mathbf{y}_{t-1}^{(i)}\} \right]}{\sum_{i=1}^N \sum_{t=6}^{T^{(i)}} 1},$$

where the superscript indicates the i -th example, and N indicates the number of translations.

We see from Figure 4.2 that for the longest source-sentence length bucket $(81, \infty)$, Q produces translations with a significantly larger 5-gram repetition rate. Moreover, beam search from the

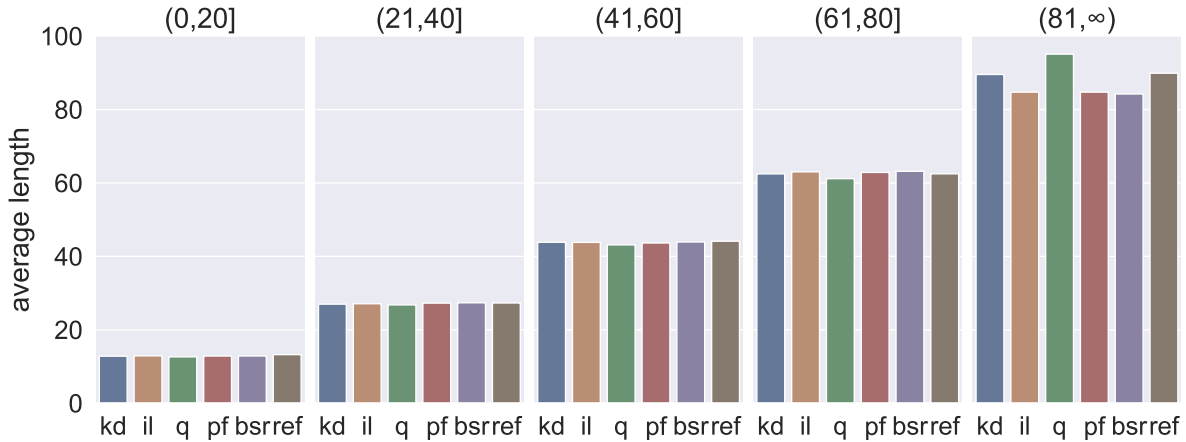


Figure 4.1: Average length bucketed by length of the source sentence. The five buckets contain 453, 877, 376, 92, 26 sentences, respectively. The six systems are KD, IL, Q learning, beam search by p_f , BSR, and reference translations, respectively. In the longest length bucket, Q learning produces translations that are longer than translations by other systems.

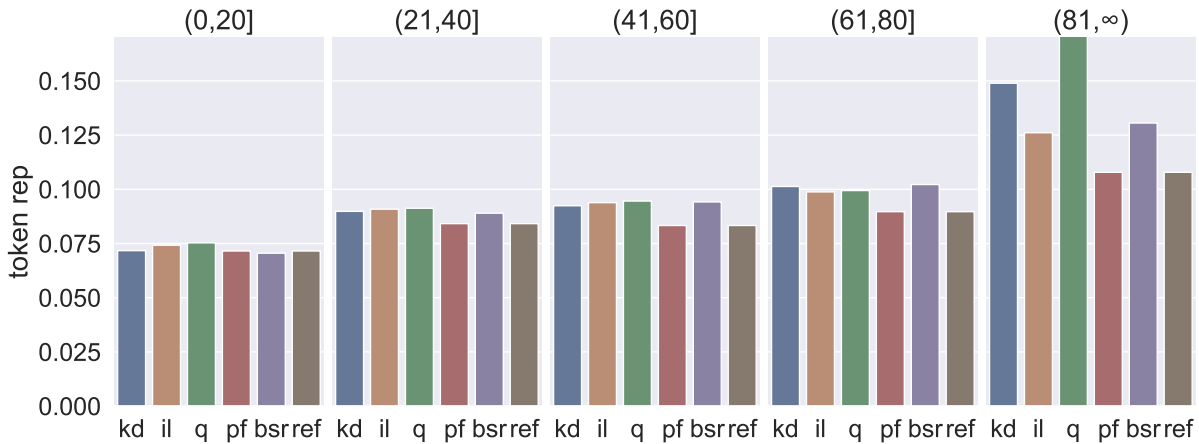


Figure 4.2: Repetition rate (“token rep”) bucketed by length of the source sentence. The five buckets contain 453, 877, 376, 92, 26 sentences, respectively. N.B.: In the last bucket, “token rep” for Q -generated translations is around 0.31, and the bar is truncated.

forward only model p_f exhibits a behavior most similar to reference translations. We leave it for the future to study the cause behind an elevated level of repetition in noisy-channel decoding.

Next, to compare translation similarity among different approaches, we examine the corpus-level BLEU score between each pair of approaches, averaged between two directions. By [Table 4.5](#),

Table 4.5: Corpus-level BLEU between translations by pairs of systems. Each reported BLEU is averaged between two directions.

system 1	system 2				
	p_f (beam search)	BSR	KD	IL	Q learning
p_f (beam search)	100	–	–	–	–
BSR	81.2	100	–	–	–
KD	64.5	66.0	100	–	–
IL	64.4	66.2	70.8	100	–
Q learning	74.0	72.0	64.3	64.1	100

translations by BSR is similar to those produced by p_f and Q learning, compared to KD and IL. Now we compare the translations produced by the three approaches. Translations by KD are more similar to IL, compared to BSR and Q learning. This is in line with our intuition that KD and IL differ from Q learning, given that how the reverse reward is presented is different between KD/IL and Q learning.

4.5.3 FURTHER ANALYSIS

KD. One may wonder whether the improvements in KD arise from the KD procedure or because we use BSR when constructing the pseudo-corpus. We therefore experiment with another model $p_{\text{KD}/\text{beam}}$: we generate the pseudo-corpus \tilde{Y}_{beam} from the training set, by beam search from p_f , and then use MLE to train $p_{\text{KD}/\text{beam}}$ using the parallel corpora $(X, \tilde{Y}_{\text{beam}})$. Table 4.4 suggests that the forward rewards of the two approaches are similar, but the reverse rewards for $p_{\text{KD}/\text{nc}}$ is much larger. Meanwhile, $p_{\text{KD}/\text{nc}}$ produces translations with higher BLEU. It is therefore necessary to use BSR to generate the pseudo-corpus, in order to amortize noisy-channel NMT using KD.

Q LEARNING. Why does Q learning, the best understood approach among the three, fail to achieve rewards that are comparable to BSR? The two challenges of a general deep Q learning algorithm are exploration and optimization.

Table 4.6: WMT’16 Ro-En examples produced by different systems. The top example is randomly selected. The bottom example is an example with a long source, and Q learning produces repetitions.

source: acum , in sa , tsipras cere grecilor sa ii incredinteze din nou mandatul de premier , in cadrul unor alegeri despre care sustine ca ii vor intari pozitia politica .

KD: now , however , tsipras is urging greeks to entrust the prime minister 's mandate again , in an election he claims will strengthen his political position .

IL: now , however , tsipras is asking greeks to reentrust them with the prime minister 's term , in an election that they claim will strengthen his political position .

Q learning: now , however , tsipras is urging greeks to reentrust his term as prime minister in an election that he claims will strengthen his political position .

beam search by p_f : now , however , tsipras is urging greeks to re-entrust the prime minister 's term in an election that he claims will strengthen his political position .

BSR: now , however , tsipras is urging greeks to reentrust the prime minister 's term , in an election that he claims will strengthen his political stance .

reference: now , however , tsipras asks the greeks again to entrust him with the prime minister position , during an election which he says will strengthen his political position .

source: adomnitei a fost trimis in judecata de directia nationala antico <unk> ruptie (dna) , fiind acuzat de favorizarea faptuitorului si fals intelectual dupa ce , spun pro <unk> curorii , ar fi incercat sa mascheze un control de audit in urma caruia se descoperise o serie de nereguli cu privire la receptia dintr-un concontract public semnat intre cj si firma laser co .

KD: adomnitei was sued by the national anti-co nistelrooij ruptie (dna) as accused of favouring the perpetrator and false intellectual after , pro nistelrooij curorii says , he would have tried to disguise an audit control as a result of which a number of irregularities concerning reception in a public concontract signed between the cj and laser co were discovered .

IL: adomnitei was sued by the national directorate antico iel ruptie (dna) and accused of favouring the perpetrator and forgery an intellectual after , pro iel curorii says , he had tried to disguise an audit control that found a number of irregularities regarding the reception in a public concontract signed between cj and laser .

Q learning: the runner the runner , the runner-the runner-in-ranging runner-up is given to the latter , as he is accused of promoting the perpetrator and faltering intellectual after , says pro or: curors , tried to disguise an audit control , as a result of which a number of irregularities concerning a reception signed between cj and laser had been discovered in a public cross-border convoy .

beam search by p_f : adomnitei was sued by the national anti-co nistelrooij rupture (dna , accused of favouring the perpetrator and forgery intellectual after allegedly attempting to disguise an audit control line between cj and lasco .

BSR: adomnitei was sued by the national anti-co xiated department (dna , accused of favouring the perpetrator and forgery intellectual after allegedly attempting to disguise an audit control line signed between cj and the lasco firm .

reference: adomni«unk» ei was indicted by the national anticorruption directorate (dna) , being accused of favouring the offender and forgery after , according to the prosecutors , he tried to mask an audit which discovered a number of irregularities regarding the acceptance of a public contract entered into by the county council and the company laser co .

Exploration refers to whether we can find high-quality trajectories. We hypothesize that it is not an issue given the diversity of trajectories we use, as shown in [Appendix B.1.1](#). We even attempt adding high-reward trajectories from BSR as well as trajectories from a deep ensemble of multiple p_f 's but neither BLEU nor reward improves.

We thus suspect optimization as a challenge. The reverse reward $\log p_r(\mathbf{x}|\mathbf{y})$ is *sparse* in that it is non-zero only at the terminal state $(\mathbf{y}_{1:T}, \mathbf{x})$ where $y_T = \langle \text{eos} \rangle$. The difficulty in maximizing the sparse reverse reward comes from using one-step bootstrapping in Q learning. Such bootstrapping allows Q learning to cope with very long episodes or even an infinite horizon, but this slows down the propagation of future reward to the past. Because we always work with relatively short episodes only in machine translation, we should investigate other learning paradigms from reinforcement learning, such as R learning [[Mahadevan 1996](#)]. We leave this further investigation to the future.

4.6 RELATED WORK

One of our approaches adapts knowledge distillation (KD) for the noisy channel NMT setting. KD [[Hinton et al. 2015](#); [Kim and Rush 2016](#)] has been shown to work well for sequence generation. [Chen et al. \[2018\]](#) propose trainable greedy decoding, in which they use knowledge distillation to train a greedy decoder so as to amortize the cost of beam search. More subsequent studies have demonstrated the effectiveness of KD in neural machine translation [[Freitag et al. 2017](#); [Tan et al. 2019](#)]; [Gu et al. \[2017\]](#) show that it is difficult for on-policy reinforcement learning (RL) to work better than KD. Recently, KD has greatly boosted performance of non-autoregressive MT models [[Gu et al. 2018](#); [Lee et al. 2018](#); [Tu et al. 2020b](#)]. KD is also used to speed up speech synthesis and the approach has been widely deployed in real products [[van den Oord et al. 2018](#)].

RL for sequence generation has been greatly inspired by [Sutton and Barto \[1998\]](#). [Ranzato et al. \[2016\]](#) and [Bahdanau et al. \[2017\]](#) apply on-policy RL (REINFORCE and actor-critic algorithms)

to MT, but the major optimization challenge lingers given that the reward is usually sparse. [Choshen et al. \[2020\]](#) recently find that the improvements in MT performance may rely on a good initialization. To address the sparsity issue, [Norouzi et al. \[2016\]](#) attempt a hybrid maximum likelihood (ML) and RL approach. More recently, [Pang and He \[2021\]](#) attempt to use an offline RL setting with per-token reward based on the a translator trained using standard MLE.

In recent years, off-policy RL methods have been used to better leverage existing trajectories in text generation. For instance, in the chatbot setting [[Serban et al. 2017](#); [Zhou et al. 2017a](#)], the periodically-collected human feedback is treated as the trajectory. In our case, we leverage the expensive BSR-obtained trajectories as well as trajectories from many different models and sources, although the sparse reward issue still lingers.

Finally, we point out a recent endeavor to speed up noisy channel NMT inference [[Bhosale et al. 2020](#)]. They reduce the size of the channel model, the size of the output vocabulary, and the number of candidates during beam search. Our solution is orthogonal: we aim to use a separate network to amortize decoding cost, while not changing the network’s architecture.

4.7 CONCLUSION

We describe three approaches (KD, IL, Q learning) to train an amortized noisy-channel NMT model. We investigate whether greedily decoding from these models will lead to accurate translations in terms of reward and quality. Although all three approaches fail to achieve comparable rewards to BSR, the reverse rewards are much higher than those from non-BSR baselines, often at the expense of forward rewards. However, we found the translation quality (measured by BLEU and BLEURT) to be comparable to that of BSR, while inference is much faster. For future work, the research community could further investigate better ways to optimize toward a sparse reward in the language generation context. Another way to approach the Q learning optimization challenge is to find better reward functions including denser rewards.

5 | REWARDS CAN BE SIMPLE AND WE CAN INNOVATE ON LEARNING ALGORITHMS: TEXT GENERATION BY LEARNING FROM DEMONSTRATIONS

While we can always search for a better reward function, another way to make progress in learning lies in improving training algorithms. The classic on-policy RL algorithms are often unstable, requiring a large number of tricks in practice [Huang et al. 2024b]. Rewards could also be unreliable when scoring sequences from the distribution corresponding to on-policy sampling. We therefore explore an extreme case where we do not deviate far from references by framing text generation as an offline RL problem. This chapter is based on work with He He [Pang and He 2021].

5.1 INTRODUCTION

A dominant approach to text generation is to use autoregressive models learned by maximum likelihood estimation (MLE) on supervised data. However, this approach introduces two well-known discrepancies between training and evaluation objectives that lead to undesired generations.

First, the training loss is negative log-likelihood, whereas the evaluation is based on human judgment of the output quality. Under model misspecification, MLE tends to over-generalize, assigning large probability mass to both high-quality and low-quality sequences [Huszár 2015; Simon et al. 2019]. Therefore, in practice, we must carefully select the decoding algorithms to produce high-quality outputs.

Second, during training, the autoregressive model conditions on the gold history/prefix; however, at inference time it conditions on model-generated history. This is known as the exposure bias problem [Ranzato et al. 2016; Bengio et al. 2015]. In the worst case, one incorrect prediction can produce a low-probability prefix under the gold data distribution, and errors compound in each of the following steps [Ross et al. 2011]. In practice, prior work has observed problems such as repetition and hallucination partly due to exposure bias [Holtzman et al. 2020; Wang and Sennrich 2020].

We aim to bridge the gap between training and evaluation in this paper. To match training and evaluation objectives, ideally we should maximize output quality given model-generated histories. This corresponds to the reinforcement learning (RL) objective: maximizing the expected reward (quality) over trajectories (sequences) induced by the policy (model). However, optimizing this objective is notoriously difficult. Prior RL approaches mainly focus on fine-tuning a learned model to optimize sequence-level metrics such as BLEU [Papineni et al. 2002], but empirically it remains unclear if RL is beneficial to text generation [Wu et al. 2018; Choshen et al. 2020]. Note that many challenges in RL arise from exploring an exponentially large space of sequences, with sparse rewards only on those close to the reference. We thus propose to learn from only the reference sequences without interaction (i.e., the offline setting). Specifically, we use off-policy policy gradient with importance weighting [Hastings 1970; Hachiya et al. 2009; Parshakova et al. 2019], where training examples with higher probability under the model are weighted higher. Further, our reward functions approximate human judgment of the output quality by estimating how likely a human would have generated a sequence. We call our algorithm GOLD (Generation

by Off-policy Learning from Demonstrations).

Results on news summarization, question generation, and machine translation show that GOLD leads to better model performance than MLE and RL fine-tuning by both task metrics and human-rated quality. Further, our analysis shows that GOLD learns high-precision models that are less sensitive to decoding algorithms. In addition, it alleviates exposure bias: the output quality does not degrade much as generation length increases.

5.2 FROM MLE TO RL FRAMEWORK

MLE TRAINING. Given a context \mathbf{x} such as a document, we want to generate a sequence of tokens $\mathbf{y} = (y_0, \dots, y_T)$, where y_i comes from a vocabulary \mathcal{V} . The generator is modeled by a conditional probability distribution parametrized by θ : $p_\theta(\mathbf{y} | \mathbf{x}) = \prod_{t=0}^T p_\theta(y_t | \mathbf{y}_{0:t-1}, \mathbf{x})$, where $\mathbf{y}_{0:t-1}$ denotes the prefix y_0, \dots, y_{t-1} . Let $p_{\text{human}}(\mathbf{y} | \mathbf{x})$ denote the data-generating distribution. Using MLE, the loss function is

$$\mathcal{L}(\theta) = -\mathbb{E}_{\mathbf{y} \sim p_{\text{human}}} \left[\sum_{t=0}^T \log p_\theta(y_t | \mathbf{y}_{0:t-1}, \mathbf{x}) \right]. \quad (5.1)$$

At inference time, we generate tokens sequentially according to p_θ .

EVALUATION. In practice, the quality of an output often relies on task-specific metrics such as fluency, correctness, and interestingness. Here for generality we consider *perceptual quality* [Huszár 2015; Hashimoto et al. 2019] which measures how likely a human would have generated the output given the context, i.e., $p_{\text{human}}(\mathbf{y} | \mathbf{x})$. Thus the evaluation metric is

$$\mathbb{E}_{\mathbf{y} \sim p_\theta} \left[\sum_{t=0}^T \log p_{\text{human}}(y_t | \mathbf{y}_{0:t-1}, \mathbf{x}) \right]. \quad (5.2)$$

Comparing Equation 5.1 and Equation 5.2, we see that the training objective encourages *high*

recall: the model must put probability mass on all human-generated sequences. In contrast, the evaluation metric encourages *high precision*: all outputs from the model must be of high quality. Unfortunately, directly optimizing the evaluation metric is impossible because p_{human} is unknown and the expectation is difficult to estimate. We therefore develop a training objective that closely approximates Equation 5.2 in the RL framework.

RL FORMULATION. Let's consider generation as a sequential decision-making process. At each time step t , the policy π_θ takes an action $a_t \in \mathcal{V}$, transits to the next state $s_{t+1} = (\mathbf{y}_{0:t}, \mathbf{x})$, and receives a reward r_t . The policy corresponds to the generation model: $\pi_\theta(a_t | s_t) = p_\theta(a_t | \mathbf{y}_{0:t-1}, \mathbf{x})$. We can thus represent a sequence as a trajectory $\tau = (s_0, a_0, r_0, \dots, s_T, a_T, r_T)$ or optionally we can represent τ without the reward terms. The set of trajectories derived from the training data is called *demonstrations* which show the desired behavior of a policy. The RL objective is to maximize $J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta} [\sum_{t=0}^T \gamma^t r_t]$, where $\gamma \in (0, 1]$ is the discount factor, and $\pi_\theta(\tau)$ denotes the distribution of τ induced by π_θ . If we knew oracle rewards $r_t = p_{\text{human}}(a_t | s_t)$, then this objective would be exactly the evaluation metric we want to optimize. Next, we describe how to optimize $J(\theta)$ with reward functions that approximate p_{human} .

5.3 APPROACH

5.3.1 OFF-POLICY POLICY GRADIENT

POLICY GRADIENT. A straightforward way to optimize $J(\theta)$ is policy gradient (PG) [Williams 1992; Sutton et al. 2000]. The gradient is given by

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta} \left[\sum_t \nabla_\theta \log \pi_\theta(a_t | s_t) \hat{Q}(s_t, a_t) \right], \quad (5.3)$$

where

$$\hat{Q}(s_t, a_t) = \sum_{t'=t}^T \gamma^{t'-t} r_{t'} \quad (5.4)$$

is the estimated return from state s_t . The expectation is estimated by Monte Carlo samples from π_θ . In text generation, the return $\hat{Q}(s_t, a_t)$ is often a sequence-level reward such as BLEU. In practice, the policy is likely to get stuck in a region of zero reward during training, generating gibberish without receiving any learning signal [Li et al. 2018; Keneshloo et al. 2019]. A common remedy is to initialize the policy with the MLE solution and/or interleave with MLE gradient update during PG. However, this would bias the parameters towards the MLE solution, thus often leads to marginal gains in practice [Wu et al. 2018; Choshen et al. 2020].

OFFLINE LEARNING. To avoid zero-reward regions, we would like to *reduce interaction with the environment and stay close to the demonstrated trajectories*. In the extreme case, the policy is learned solely from the static demonstrations without additional interaction with the environment, which is referred to as the offline setting. While it is in general a more challenging problem, we argue that the offline setting is appropriate for text generation [Serban et al. 2017; Jaques et al. 2019]. First, the environment dynamics is known: once a token is generated, we deterministically transition to the next state with the additional token appended to the prefix; no interaction is needed to learn the environment. Second, while exploration may lead to high-quality sequences different from the reference, we lack a good reward function to identify them [Novikova et al. 2017; Aharoni and Goldberg 2018; Clark et al. 2019]. Therefore, the benefit of exploration in text generation is limited.

In the offline setting, we cannot estimate the expected return of π_θ by sampling trajectories from it, and must use trajectories from a different *behavioral policy* π_b , known as *off-policy* learning in RL. A common technique to estimate expectations under one distribution π_θ given samples from a different distribution π_b is importance sampling, which leads to the following unbiased

estimator of the gradient [Precup et al. 2000]:

$$\mathbb{E}_{\tau \sim \pi_b} \left[\sum_t w_t \nabla_{\theta} \log \pi_{\theta}(a_t | s_t) \hat{Q}(s_t, a_t) \right],$$

with importance weights $w_t = \prod_{t'=0}^t \frac{\pi_{\theta}(a_{t'} | s_{t'})}{\pi_b(a_{t'} | s_{t'})}$.

SLIGHT VARIATION IN GOAL: MODIFYING J TO \tilde{J} . As discussed, assuming $\gamma = 1$, $J(\theta) = \mathbb{E}_{\tau \sim \pi_{\theta}} [\sum_{t=0}^T r_t]$ is our original objective. Given that we do not know what the behavioral policy π_b (which produced the demonstrations \mathcal{D}) is, one simple approach is to slightly modify the original J so we can use empirical distribution $\tilde{\pi}_b$ to approximate π_b (more discussion later in this subsection). Specifically, we let $p(s_0) = 1/N$ assuming no duplication of s_0 in \mathcal{D} , $\tilde{\pi}_b(a | s) = 1$ if (s, a) is in τ and 0 otherwise. Thus, under this approximation, $p_e(\tau) = 1/N$ if $\tau \in \mathcal{D}$ and 0 otherwise.

We modify the objective to

$$\tilde{J}(\theta) = \mathbb{E}_{\tau \sim \pi_{\theta}} \left[\sum_{t=0}^T r_t \right] p_e(\tau). \quad (5.5)$$

The difference between \tilde{J} and J is that in \tilde{J} we only care about rewards under the approximated behavioral policy. Therefore, after some derivation, we obtain that

$$\nabla \tilde{J}(\theta) = \mathbb{E}_{\tau \sim \tilde{\pi}_b} \left[\sum_{t=0}^T \left(\prod_{t'=0}^t \pi_{\theta}(a_{t'} | s_{t'}) \right) \nabla_{\theta} \log \pi_{\theta}(a_t | s_t) \hat{Q}(s_t, a_t) \right]. \quad (5.6)$$

Why? This derivation can be done as follows. Given that $\tilde{J}(\theta) = \mathbb{E}_{\tau \sim \pi_{\theta}} [\sum_{t=0}^T r_t] p_e(\tau) =$

$\mathbb{E}_{\tau \sim \pi_\theta} r(\tau) p_e(\tau) = \mathbb{E}_{\tau \sim \pi_b} \frac{\pi_\theta(\tau)}{\pi_b(\tau)} r(\tau) p_e(\tau)$, we can compute the gradient:

$$\begin{aligned} \nabla \tilde{J}(\theta) &= \mathbb{E}_{\tau \sim \pi_b} \frac{\nabla \pi_\theta(\tau)}{\pi_b(\tau)} r(\tau) p_e(\tau) \\ &= \mathbb{E}_{\tau \sim \pi_b} \frac{\pi_\theta(\tau)}{\pi_b(\tau)} \nabla \log \pi_\theta(\tau) r(\tau) p_e(\tau) \\ &= \mathbb{E}_{\tau \sim \pi_b} \left[\left(\prod_{t=0}^T \frac{\pi_\theta(t)}{\pi_b(t)} \right) (\nabla_\theta \log \pi_\theta(a_t | s_t)) \left(\sum_{t=0}^T r(s_t, a_t) \right) p_e(\tau) \right] \end{aligned}$$

We then rearrange the terms, and given that that actions in the future would not influence the current time-step weights:

$$\begin{aligned} \nabla \tilde{J}(\theta) &\approx \mathbb{E}_{\tau \sim \pi_b} \left[\left[\sum_{t=0}^T \nabla_\theta \log \pi_\theta(a_t | s_t) \left(\prod_{t'=0}^t \frac{\pi_\theta(t')}{\pi_b(t')} \right) \left(\sum_{t'=t}^T r(s'_t, a'_t) \right) \right] p_e(\tau) \right] \\ &= \frac{1}{N} \sum_{i=1}^N \sum_{t=0}^T \nabla_\theta \log \pi_\theta(a_t | s_t) \left(\prod_{t'=0}^t \frac{\pi_\theta(t')}{1/N} \right) \left(\sum_{t'=t}^T r(s'_t, a'_t) \right) \\ &= C \sum_{i=1}^N \sum_{t=0}^T \nabla_\theta \log \pi_\theta(a_t | s_t) \left(\prod_{t'=0}^t \pi_\theta(t') \right) \hat{Q}(s_t, a_t), \end{aligned}$$

where C is a constant which can be ignored in implementation.

APPROXIMATIONS. Computing the importance weights above requires multiplying per-action importance weight over multiple time steps. In practice, we have found that it is sensitive to optimization hyperparameters and takes longer to converge. Therefore, we use the per-action approximation: $w_t \approx \frac{\pi_\theta(a_t | s_t)}{\pi_b(a_t | s_t)}$. This corresponds to optimizing the expected return under the off-policy state distribution induced by π_b and the on-policy action distribution of π_θ . Although this estimator is biased, empirically it has been shown to reduce variance and work reasonably well if π_b and π_θ are close [Serban et al. 2017; Levine et al. 2020]. Another obstacle is that we do not know π_b which produced the demonstrations $\mathcal{D} = \{(\mathbf{x}^{(i)}, \mathbf{y}^{(i)})\}_{i=1}^N$. One option is to estimate π_b on \mathcal{D} . As mentioned above, here we take a simpler approach that uses the empirical distribution: $\pi_b(\tau) \approx 1/N$ for $\tau \in \mathcal{D}$ and 0 otherwise. As a result, the denominator in w_t is a constant and can

be ignored in optimization. Our final approximated gradient has the form:

$$\nabla_{\theta} \tilde{J}(\theta) \approx \sum_{i=1}^N \sum_{t=0}^T \pi_{\theta}(a_t^i | s_t^i) \nabla_{\theta} \log \pi_{\theta}(a_t^i | s_t^i) \hat{Q}(s_t^i, a_t^i), \quad (5.7)$$

where the superscript i represents the i th trajectory. Compared with the MLE gradient:

$$\sum_{i=1}^N \sum_{t=0}^T \nabla_{\theta} \log \pi_{\theta}(a_t^i | s_t^i),$$

our gradient (5.7) upweights actions with high return and actions preferred by the current policy π_{θ} . Intuitively, it encourages the learning algorithm to focus on “easy” examples (high likelihood under the model) which improves precision.

5.3.2 REWARD

Let r be the reward function such that $r_t = r(s_t, a_t)$. To optimize the perceptual quality of a sequence (see (5.2)), we want $r(s, a)$ to approximate $p_{\text{human}}(a | s)$, i.e., how likely humans would have generated a given s . In general, it is hard to develop a reliable reward function for text generation tasks because it must work well for a large set of possible generations. In the offline setting, however, we can restrict the domain of R to state-action pairs on the demonstrations. Next, we propose three reward functions.

δ -REWARD. An obvious choice is a sequence-level reward, which considers all demonstrations to be equally good and assigns zero reward to any other outputs. Formally,

$$r_{\delta}(s_t, a_t) \stackrel{\text{def}}{=} \begin{cases} 1, & \text{if } t = T \text{ and } (s_{0:T}, a_{0:T}) \in \mathcal{D} \\ 0, & \text{otherwise} \end{cases} \quad (5.8)$$

where a reward of one is received in the terminal state for any trajectory in the demonstrations.

ESTIMATED p_{HUMAN} . In text generation tasks, an input often has many correct outputs and the reference may be an uncommon output that contains rare words or has complex syntax. To account for different likelihood of the references, we estimate the probability of each reference by minimizing $\text{KL}(p_{\text{human}}||q)$, where $q(a | s)$ approximates $p_{\text{human}}(a | s)$. This is equivalent to finding the MLE solution (denoted by p_{MLE}).¹ Importantly, p_{MLE} is a reasonable approximation to p_{human} when *restricted to the demonstrations*. It is not a good reward function in general, however; it can assign large probability mass to low-quality outputs. Given the estimated perceptual quality $p_{\text{MLE}}(a | s)$, we define two reward functions. Our first reward function corresponds to a product of probabilities when summed over the trajectory:

$$r_p(s, a) \stackrel{\text{def}}{=} \log p_{\text{MLE}}(a | s). \quad (5.9)$$

Assuming $\gamma = 1$, the return at time step t is $\hat{Q}_t(s_t, a_t) = \sum_{t'=t}^T \log p_{\text{MLE}}(a_{t'} | s_{t'})$. Thus a sequence has high reward only if every word has high likelihood under p_{MLE} . To allow for partial credits even if bad actions are taken at certain steps, we define another reward function corresponding to the sum of probabilities:

$$r_s(s, a) \stackrel{\text{def}}{=} p_{\text{MLE}}(a | s). \quad (5.10)$$

The return $\hat{Q}(s_t, a_t) = \sum_{t'=t}^T p_{\text{MLE}}(a_{t'} | s_{t'})$, thus the policy can recover from bad decisions if the subsequent actions receive high reward.

5.3.3 THE GOLD ALGORITHM

Our full algorithm based on off-policy PG is shown in [Algorithm 2](#). For importance weights $\pi_\theta(a | s)$, to avoid drastic changes, we initialize π_θ with the MLE solution. In addition, we compute

¹Note that $\text{KL}(p_{\text{human}}||q) = \mathbb{E}_{p_{\text{human}}} \log p_{\text{human}} - \mathbb{E}_{p_{\text{human}}} \log q$, thus minimizing the KL divergence with respect to q is equivalent to the MLE objective.

Algorithm 2: GOLD

```
 $\pi_\theta \leftarrow p_{\text{MLE}}, \tilde{\pi}_\theta \leftarrow p_{\text{MLE}}$   
for  $step = 1, 2, \dots, M$  do  
  Sample a minibatch  $B = \{(\mathbf{x}^i, \mathbf{y}^i)\}_{i=1}^{|B|}$   
  foreach  $(s_t^i, a_t^i)$  do  
    Compute importance weights  $\max(u, \tilde{\pi}_\theta)$ , and compute returns  $\hat{Q}(s_t^i, a_t^i) - b$   
  end  
  Update  $\theta$  by (5.7) using gradient descent  
  if  $step \% k = 0$  then  $\tilde{\pi}_\theta \leftarrow \pi_\theta$   
end  
Return:  $\pi_\theta$ 
```

the importance weights by a weighting policy $\tilde{\pi}_\theta$ that synchronizes with π_θ periodically so that the weights do not change frequently between updates. We also lower-bound the importance weight by a small number u .

Another source of variance comes from policy gradients. Since our return is computed from a sum or product of probabilities ((5.9) and (5.10)), we truncate the future trajectory after five steps. We follow the common practice to subtract a baseline b from the return to reduce variance; moreover, to avoid negative reward on the demonstrations (after subtracting baseline), we lower-bound p_{MLE} in (5.9) and (5.10) by a small number c .

In practice, GOLD is easy to implement; further, given an existing p_{MLE} , the GOLD-training stage usually takes less time than MLE. The code is available.²

5.4 EXPERIMENTS

5.4.1 SETUP

We chose four text generation tasks: (1) **question generation** [NQG; Zhou et al. 2017b]: given a passage and a short span of the passage, the goal is to generate a question that can be answered

²Code: <https://github.com/yzpang/gold-off-policy-text-gen-iclr21>

by the span; (2) **extractive summarization** [CNN/DM; Hermann et al. 2015]; (3) **abstractive summarization** [XSum; Narayan et al. 2018]: the references are more *abstractive* than CNN/DM summaries; (4) **machine translation** [IWSLT14 De-En; Cettolo et al. 2014]. See Appendix C.1.1 for the size and the source of the datasets. We evaluate NQG and summarization by both automatic metrics, i.e., corpus-level BLEU-4 [Papineni et al. 2002] and ROUGE-1/2/L [Lin 2004] respectively, as well as human ratings.

We experiment with three variants of GOLD: GOLD- δ , GOLD- p , GOLD- s , which uses the δ -reward and the two estimated rewards (R_p and R_s), respectively. Our baseline learning algorithm is standard MLE, and we compare with on-policy RL training using policy gradient in Subsection 5.4.3. We describe models for each task at the beginning of Subsection 5.4.2.

For GOLD training, we use the baseline $b = -60$ for GOLD- p and $b = 0$ for GOLD- s . To lower bound the return such that it is non-negative on demonstrated trajectories, we tune the lower bound c of p_{MLE} in $\{0, 0.01, 0.05, 0.1\}$ in (5.9) and (5.10). Furthermore, to reduce variance for importance weights, we lower bound them by $u \in \{0, 0.1, 0.15, 0.2\}$. All hyperparameters are tuned on the dev set. See Appendix C.1.3 for more reproducibility details.

5.4.2 RESULTS AND ANALYSIS

GOLD IMPROVES BOTH STANDARD AND TRANSFORMER MODELS. Recall that one of our main motivations is that MLE tends to over-generalize under model misspecification, i.e., high recall but low precision. One may wonder whether this problem can be fixed by better modeling. Therefore, we evaluated GOLD with both standard high-performing models and state-of-the-art pretrained model. For standard models, we chose two representative seq2seq-based models, NQG++ [Zhou et al. 2017b] and the pointer-generator model [See et al. 2017] for NQG and CNN/DM respectively.³ Table 5.1 shows that GOLD is better than MLE in terms of BLEU and ROUGE. In particular, we find that using estimated rewards is superior to the δ -reward, showing the benefits of accounting

³We didn't use standard seq2seq-based models for IWSLT14 and XSum as they are not competitive.

Table 5.1: BLEU/ROUGE (\uparrow) and perplexity (\downarrow) using **standard models** on test sets. GOLD achieves better metric scores despite high held-out perplexity. Experiments are run using a fixed random seed (12); attempted three random seeds (1, 12, 123) and all BLEU/R-2 scores are within 0.1 points of the reported. Refer to Table 5.3 for transformer results.

	NQG		CNN/DM			
	(NQG++ net)		(pointer generator network)			
	BLEU \uparrow	ppl \downarrow	R-1 \uparrow	R-2 \uparrow	R-L \uparrow	ppl \downarrow
MLE	14.23	29.25	39.00	17.10	36.07	20.11
GOLD- δ	14.96	110.58	39.02	17.16	35.98	133.10
GOLD- p	15.93	148.84	39.20	17.31	36.23	143.58
GOLD- s	16.10	158.45	39.95	17.81	36.81	29.80

Table 5.2: Dev set results of standard models using different decoding algorithms. b : beam size. We report the average of 3 runs for top- k sampling. Models trained by GOLD are less sensitive to decoding algorithms.

	NQG (BLEU)		CNN/DM (ROUGE-2)	
	MLE	GOLD- s	MLE	GOLD- s
greedy	14.13	16.06	17.40	18.51
beam search ($b = 3$)	14.19	15.84	17.65	18.44
beam search ($b = 5$)	14.07	15.74	17.63	18.25
top- k samp. ($k = 5$)	11.27	15.41	13.06	17.02
top- k samp. ($k = 20$)	10.08	15.38	11.23	16.57

Table 5.3: Results using **transformer models** on test sets. The advantage of GOLD is maintained on advanced models based on transformers and pretraining.

Objective	NQG (BART)		CNN/DM (BART)				XSum (BART)				IWSLT14 De-En (Transformer)	
	BLEU \uparrow	ppl \downarrow	R-1 \uparrow	R-2 \uparrow	R-L \uparrow	ppl \downarrow	R-1 \uparrow	R-2 \uparrow	R-L \uparrow	ppl \downarrow	BLEU \uparrow	ppl \downarrow
MLE	20.68	5.96	44.16	21.28	40.90	5.41	45.58	22.08	37.04	5.07	34.64	5.31
GOLD- p	21.42	10.48	45.40	22.01	42.25	11.44	45.75	22.26	37.30	6.95	35.33	7.59
GOLD- s	21.98	7.67	44.82	22.09	41.81	9.60	45.85	22.58	37.65	6.85	35.45	6.90

for varying quality of the references. We thus consider only GOLD- p and GOLD- s in the rest of the experiments. For transformer models [Vaswani et al. 2017], we used the pretrained BART [Lewis et al. 2020] for NQG, CNN/DM, and XSum; we used standard transformer for IWSLT14 De-En. Table 5.3 shows that GOLD achieves better scores than MLE across all tasks, including near-SOTA on CNN/DM (R-2 95% confidence interval: 21.84-22.33) and good performance on XSum (R-2 95% CI: 22.25-22.92).

We further crowdsourced human evaluation by pairwise comparison⁴ between MLE-trained and GOLD- s -trained model outputs. Each pair of comparison is repeated three times (by three different workers) and we take the majority answer. For each dataset, the evaluations are done by

⁴We used Amazon Mechanical Turk on 200 pairs of examples for each of the three tasks. We added qualification tasks with obvious answers and we didn’t use any results by workers who failed the qualifications.

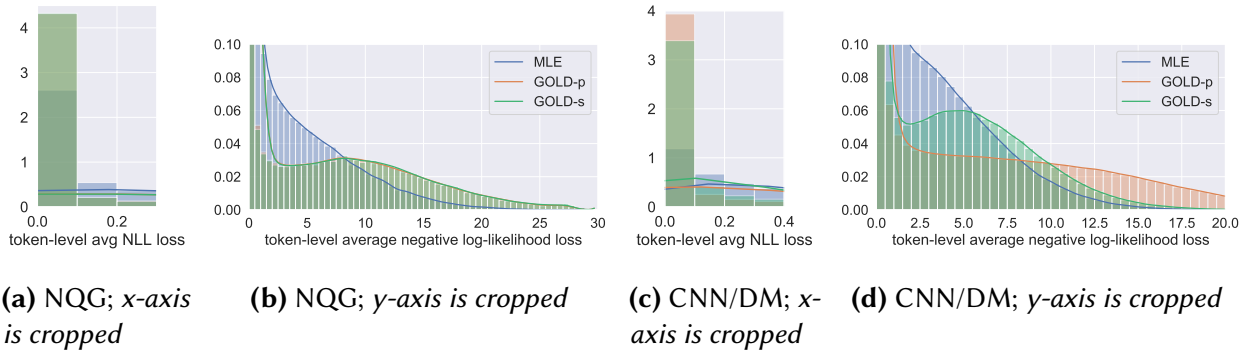


Figure 5.1: Histograms of token-level NLL loss using standard models on NQG and CNN/DM dev sets. MLE learns high-recall models whose loss distribution is spread out; GOLD learns high-precision models whose loss distribution is concentrated on near-zero losses.

at least 15 different workers. For NQG, we showed workers the entire input and the questions generated by two models, and we ask workers to select the better one (with a third “tie” option). For summarization, we ask workers to select the generation closer in meaning to the reference without showing the article. More details are in [Appendix C.3](#). [Table 5.5](#) shows that workers prefer outputs from models trained by GOLD more often than those trained by MLE.

Table 5.4: BLEU/ROUGE (\uparrow) on test sets, using standard models finetuned with on-policy objectives. On-policy objectives marginally improve upon both MLE and GOLD baselines. Starred (*) models have MLE baselines >0.1 difference to our MLE R-2. $\delta R-2$: R-2 for the model minus R-2 for the corresponding MLE. PG: standard policy gradient; PPO: proximal policy optimization.

algorithm	reward	NQG		CNN/DM	
		BLEU	R-2	R-2	$\delta R-2$
Off-policy (this paper)					
MLE	–	14.23	17.10	–	
GOLD-s	R_s (Subsection 5.3.2); the only <i>task-independent</i> reward in this table	16.10	17.81	0.71	
On-policy					
MLE+PG	BLEU or R-2	14.55	17.35	0.25	
MLE+PPO	human preferences [Ziegler et al. 2019]	–	17.61	0.62	
MLE+PG(*)	R-L + saliency + summary-entailment [Pasunuru and Bansal 2018]	–	18.00	0.67	
MLE+PG(*)	question-answering score [Scialom et al. 2019]	–	17.66	–0.12	
GOLD+PG	BLEU or R-2	16.38	18.14	1.04	

GOLD ENCOURAGES HIGH-PRECISION MODELS. One interesting observation from Table 5.1 and Table 5.3 is that compared to MLE, GOLD leads to much higher held-out perplexities, while achieving better metric scores. Since both are evaluated against the reference, one would expect high perplexity to correlate with low metric scores. To better understand the behavior of GOLD, we examine the distributions of token-level negative log-likelihood (NLL) loss (a monotonic transformation of perplexity) in Figure 5.1. We see that the loss distribution of GOLD (compared to MLE) concentrates on near-zero losses (Figure 5.1(a) and Figure 5.1(c)) with a long tail of large losses (Figure 5.1(b) and Figure 5.1(d)), hence high perplexity. In contrast, MLE has much fewer near-zero losses and fewer large losses, suggesting it tries to generate *all* tokens; i.e., MLE encourages recall, as discussed in Section 5.2. We conclude that GOLD achieves better metric scores by focusing on easy-to-learn tokens at the expense of lower recall with respect to the reference.

Another advantage of high-precision models is that they do not rely much on decoding algorithms to sample high-quality outputs from the learned distribution. From a RL perspective, the policy already considers future rewards when making local decisions, thus beam search is not necessary. As a result, we see in Table 5.2 that GOLD achieves similar performance with both argmax decoding and top- k sampling. In contrast, MLE suffers significantly from sampling, which suggests that it learns a high-recall but low-precision model.

Table 5.5: Human comparisons on 200 randomly selected test examples for each task. Win: % generations from GOLD-trained BART that are better than from MLE-trained BART, given the same source.

NQG (BART)			CNN/DM (BART)			XSum (BART)		
win	lose	tied	win	lose	tied	win	lose	tied
38.0	28.5	33.5	37.5	24.5	38.0	35.0	21.5	43.5

GOLD ALLEVIATES EXPOSURE BIAS. GOLD suffers less from exposure bias because it trains on the state/history distribution induced by the model instead of the reference data. Here, we empirically

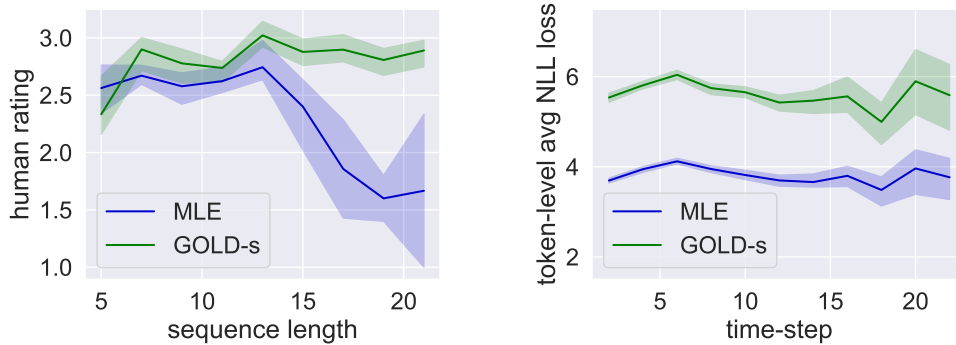


Figure 5.2: Left: Avg human ratings vs. generation length, on 736 NQG samples. (Colored regions: 95% confidence interval.) Each data point has ≥ 30 annotations. The quality of long generations from MLE-trained model drops heavily, but stays stable across lengths for GOLD-s generations. Right: Avg NLL loss of t -th token given the *gold* prefix tokens vs. time-step t , on NQG dev set. Without exposure bias, NLL loss stays stable across lengths.

quantify the exposure bias problem in learned models. If there is exposure bias, then the output quality is expected to degrade as output length increases, as the history is more likely to deviate from the reference distribution with accumulated generation steps. To evaluate quality, we sampled 736 generations of different lengths from standard models trained by both MLE and GOLD on NQG. Given the paragraph, words to query on, and the generated questions, we then asked workers to rate the generations from 1 (worst) to 4 (best). [Figure 5.2](#) (left) shows that the output quality of the MLE-trained model degrades when the sequence length is over 14 words, whereas the quality of the GOLD-s-trained model stays relatively stable across all lengths.⁵ Qualitatively, we observe frequent degenerations [[Holtzman et al. 2020](#); [Welleck et al. 2020b](#)] including repetitions and hallucinations within a sentence generated by MLE-trained model, as shown in [Table 5.6](#). In contrast, [Figure 5.2](#) (right) shows the NLL loss conditioned on gold histories on NQG dev set.⁶ We can see that without exposure bias, NLL loss does not vary much as the length increases. Therefore, we conclude that the big performance drop for long generations using MLE is mainly due to exposure bias and GOLD does not suffer from the problem.

⁵We also used BLEU as a quality metric and observed similar results, shown in [Appendix C.2.3](#).

⁶A token prediction accuracy vs. time-step plot, which shows similar trend, is shown in the appendix.

Table 5.6: NQG generations using standard models. Words to query on are bolded. Long generations from MLE-trained model often result in repetition or hallucination. Refer to more examples in the appendix.

Input	that project was entitled the factory project to reference andy warhol and to create a factory to completely digitize the collection .
MLE	what was the name of the project that was not digitize to digitize ?
GOLD	what was the name of the project that was to reference andy warhol ?

Input	braddock (with george washington as one of his aides) led about 1,500 army troops and provincial militia on an expedition in june 1755 to take fort duquesne .
MLE	what was the name of the aid of george washington university ?
GOLD	who led about 1,500 army troops and provincial militia on an expedition ?

5.4.3 COMPARISON WITH ON-POLICY TRAINING

While offline RL is generally more challenging due to lack of interaction with the environment, we argue that the benefit from interaction is limited in text generation (Subsection 5.3.1) and outweighed by the optimization challenges. In this section, we investigate the effect of on-policy training using task metrics as rewards. Specifically, we pre-train the model using MLE and then fine-tune it using PG. To avoid degenerate solutions, we interleave MLE and PG updates evenly during fine-tuning. Similarly, we fine-tune GOLD-initialized models using PG. For on-policy fine-tuning, we use BLEU and ROUGE-2 as rewards for NQG and CNN/DM respectively.⁷ Table 5.4 shows that additional on-policy training improves both MLE and GOLD marginally. However, MLE with PG is still worse than GOLD. Further, one of the best-performing on-policy methods using a similarly competitive pretrained transformer model [Ziegler et al. 2019] also shows limited improvements over supervised baseline on CNN/DM, despite having better reward functions (domain-specific human preference annotations). Overall, the benefit from on-policy training is unclear in our experiments.⁸ Please refer to the appendix for more details.

⁷While rewards in Subsection 5.3.2 are useful on *demonstrations*, they are not suitable for the on-policy setting as they cannot differentiate good vs. bad generations on *the entire output space* effectively; e.g., Murray and Chiang [2018]; Stahlberg and Byrne [2019] showed that maximizing p_{MLE} during decoding leads to empty generations.

⁸On a related note, Choshen et al. [2020] also showed in machine translation that even properly tuned on-policy methods may not work well either.

5.4.4 DISCUSSION ON GENERATION DIVERSITY

The objective of GOLD is to produce high-precision text at the cost of recall: There are references that the model cannot generate with high probability, which is reflected by the high held-out perplexity in [Table 5.1](#) and [Table 5.3](#). One may wonder what the impact of GOLD on text “diversity” is. This issue warrants more discussion, but for text generation, “diversity” may stand for the following.

(1) DIVERSITY AS IN THE ABILITY TO GENERATE A NUMBER OF DIFFERENT CORRECT GENERATIONS GIVEN ONE CONTEXT. This is often discussed in the context of mode collapse, which is an important problem for image generation and unconditional text generation (e.g., continuation from a prompt). However, for many conditional NLG tasks, while there are multiple correct outputs, producing one good generation is often sufficient in practice, e.g., question generation, summarization, machine translation, image captioning, text style transfer, and even chit-chat dialogues (unless users expect the bots to say different things in the same context every time). One exception is creative writing tasks where we would like to have multiple novel generations given the same context, e.g., generating from a language model [[Caccia et al. 2020](#)]. In these cases, GOLD may not be able to provide a variety of high-quality generations given one context, although it would still produce different outputs given different contexts. Another potential failure mode is that in open-ended dialogues, if one common response has large probability under true data distribution, then GOLD may lead to a distribution concentrated on this mode. In this case, additional inductive bias is needed to separate good modes from bad ones, e.g., additional reward on specificity of the response. On the other hand, while MLE-trained models have good recall and we can potentially sample many different outputs with a high temperature, or large k in top- k sampling, or large p in top- p sampling,⁹ there are only a few high-quality ones. Our conjecture is that there may not be enough data to cover all modes, and in fact high-likelihood outputs

⁹Top- p sampling is another term for nucleus sampling by [Holtzman et al. \[2020\]](#).

from MLE-trained models are often degenerate [Stahlberg and Byrne 2019; Cohen and Beck 2019; Holtzman et al. 2020].

In sum, given the trade-off between diversity and quality, we argue that generating a single high-quality output is a reasonable goal for most conditional text generation tasks, and we leave the question of generating both diverse and high-quality outputs to future work.

(2) DIVERSITY AS IN THE LINGUISTIC COMPLEXITY OF THE OUTPUT, GIVEN THE INPUT. First, we compare GOLD and MLE by measuring the complexity of the output using the number of unique n-grams and did not find significant difference. For example, GOLD’s number of unique 1/2/3/4/5-grams for XSum (using BART) is 18846/18835/18103/17639/17258, MLE’s is 19071/19053/18349/17875/17531, and gold-standard target numbers are 23674/23661/22869/22280/21822. In addition, for question generation and summarization, we measure the complexity of the output by abstractiveness, i.e., the proportion of n-gram overlaps between the input and the generation. For XSum (using BART), the proportion of 1/2/3/4/5-gram overlap for MLE is 0.75/0.27/0.10/0.053/0.031 and for GOLD: 0.73/0.24/0.087/0.039/0.021; the trend mostly holds for NQG and CNN/DM as well. In sum, we conclude that GOLD and MLE are comparable in producing complex or novel outputs.

(3) DIVERSITY AS IN THE COVERAGE OF THE TRUE DATA DISTRIBUTION. This definition is related to (1). This diversity is the “recall” intuitively, and can be measured by NLL loss or perplexity, which will be sacrificed. In our case, the consequence is that the model tends to ignore difficult gold examples (Figure 5.1), which in text generation, may sometimes be noise or outliers. Empirically for a large number of text generation tasks, paying less attention to such examples did not cause mode collapse in our case.

5.5 RELATED WORK

EXPOSURE BIAS. In structured prediction, there is a flurry of works addressing exposure bias since [Bengio et al. \[2015\]](#). Most works focus on learning global sequence scores instead of locally normalized scores using either variants of beam search [[Wiseman and Rush 2016](#); [Andor et al. 2016](#); [Goyal et al. 2018](#)] or energy networks [[Belanger and McCallum 2016](#); [Tu et al. 2020a](#)]. These training algorithms are often complex and costly. Exposure bias is well studied in imitation learning [[Daumé et al. 2009](#); [Ross et al. 2011](#)] and learning-to-search has been applied to RNNs to incorporate losses of sequences deviating from references [[Leblond et al. 2018](#)], but they require annotations or cost functions on non-reference sequences which may not be available for text generation.

OBJECTIVES BEYOND MLE. Policy gradient-based algorithms and their variants have been used extensively in text generation to optimize sequence-level metrics [[Ranzato et al. 2016](#); [Shen et al. 2016](#); [Norouzi et al. 2016](#); [Pasunuru and Bansal 2018](#)]. In addition, off-policy RL is commonly used in dialogue where online interaction with users is expensive [[Serban et al. 2017](#); [Jaques et al. 2019](#)]. The main difference is that we take advantage of the demonstrations and design generic reward functions for generation tasks. There is another line of work using policy gradient to optimize reward from a discriminator that differentiates good vs. bad generations [[Yu et al. 2017b](#); [Li et al. 2017b](#); [Lu et al. 2019](#)]. However, these approaches often underperform MLE in practice [[Tevet et al. 2019](#)] due to optimization challenges. Recently, a concurrent work, [Kang and Hashimoto \[2020\]](#), proposed truncated log-loss which both optimizes distinguishability and enjoys efficient optimization.

HIGH-PRECISION TEXT GENERATION. It is noticed early in neural text generation that MLE tends to produce high-recall models that over-generalize. Previously, high-quality outputs are selected

mainly through decoding (e.g., beam search, low-temperature sampling, truncated sampling). Recently, there is an increasing amount of work on discouraging implausible samples during training, e.g., using negative sampling [Welleck et al. 2020c], self-training on high-quality samples [Kedzie and McKeown 2019], and confidence-oriented decoding with calibration [Tian et al. 2020]. In contrast, we tackle the fundamental problem of mismatched objectives and propose a general learning framework.

5.6 CONCLUSION

We provide an efficient algorithm that addresses the two train/test discrepancies in MLE training for text generation: likelihood as learning objective vs. quality as evaluation metric; gold history in training vs. model-generated history in inference. We have demonstrated that off-policy RL is a promising framework for text generation, with matched train/test objectives and optimization advantages like MLE. We believe more advanced off-policy learning techniques (e.g., proximity constraints) can be easily integrated into text generation and further improve performance.

5.7 DEVELOPMENTS SINCE PUBLICATION

One concern is that our GOLD experiments are done on relatively small models, with the number of parameters ranging from 10M to 406M (Appendix C.1.3). Does GOLD work on larger models? In addition, does the algorithm work on tasks other than the ones demonstrated by us?

The answer is yes. For example, Google DeepMind’s AlphaCode models have 1–41B parameters benefits from GOLD, and the LLaMA-7B model can be improved by GOLD on an alignment task as discussed below. The community has also proved the applicability of GOLD on image captioning (a multi-modal task) and summarization factuality improvement.

Google DeepMind’s AlphaCode [Li et al. 2022] achieves a breakthrough in competition-level

code generation. It achieves outstanding performance (top 54.3% in terms of average ranking of contestants) on Codeforces. It is the first time an AI system is competitive enough in solving questions in programming contests, and GOLD is a core part of the algorithm. Intuitively, GOLD can be adapted to code generation: MLE minimizes loss with respect to every reference example, but in code generation we care about finding only a single correct solution. GOLD- δ is used in the AlphaCode algorithm, and the importance weights $\pi_{\theta}(\cdot)$ is empirically adjusted to be $\max(\pi_{\theta}(\cdot)^{\alpha}, \beta)$ where $\alpha = 0.5$ and $\beta = 0.05$ to increase training stability. The evaluation metric is as follows. Given a programming question, we first obtain 1K samples from the model. Then, we choose ten samples to submit. The question is solved correctly if any of these ten samples passes the test. The metric 10@1K thus refers to the average solve rate. After a few enhancements (including masked language modeling and tempering tricks), GOLD- δ is able to bring up the 10@1K performance on 1B-scale model from 10.6% to 12.4%. It is encouraging to see that GOLD can be applied onto bigger state-of-the-art base models and more real-world tasks. AlphaCode2 [AlphaCode Team 2023] shows an impressive ranking of around 87th percentile (solving 43% problems) on Codeforces competition platform. GOLD is one of the only three references in the report (the other two being Gemini and AlphaCode).

GOLD can also help with alignment tasks. The “helpful and harmless assistant” (HH) task [Bai et al. 2022] is a popular alignment task. Given a question by a human, the goal is for the assistant to produce a helpful and harmless response. The filtered dataset used by Baheti et al. [2024] contains 143K train and 8.2K test examples. Baheti et al. have presented experiments of GOLD- δ , but using trained sequence-level rewards in place of delta rewards. The MLE-trained model is a LLaMA-7B model [Touvron et al. 2023a] finetuned on HH-RLHF dataset [Bai et al. 2022] by Dettmers et al. [2023]. The reward model is a 1.4B-parameter classifier trained on various datasets detailed in the footnote.¹⁰ GOLD- δ has been shown to be useful even on large base models (7B-parameter) on the HH task, exceeding the performance of both the baseline LLaMA-7B

¹⁰<https://huggingface.co/OpenAssistant/oasst-rm-2.1-pythia-1.4b-epoch-2.5>

model and PPO on harmless and helpfulness metrics. [Baheti et al.](#) also modify the GOLD objective in the following way: (1) treating the entire generated sequence as one action; (2) using MLE-probability to approximate the behavioral policy π_b ; (3) using advantages instead of reward; (4) used other tricks including importance weight clipping and advantage priority sampling. They achieved slightly higher performance on the HH task and a similar Reddit response generation task.

Finally, it is also encouraging to see that GOLD has been used in vision-language tasks. For example, in the FIBER pretraining framework [[Dou et al. 2022](#)], GOLD is used to bring large improvements in image captioning tasks. It leads to better performance compared with models trained on $> 10^8$ images and for COCO, GOLD helps FIBER achieve state-of-the-art (base-sized) COCO CIDEr scores [[Vedantam et al. 2015](#)].

6 | WE DO NOT NECESSARILY NEED RL: ITERATIVE REASONING PREFERENCE OPTIMIZATION

We continue the discussion on algorithmic changes to better learn from rewards. An alternative to RL is based on variants of direct preference optimization which operates over policies instead of rewards. This approach can successfully boost reasoning ability of top-performing LLMs by a large margin. This chapter is based on work with Weizhe Yuan, Kyunghyun Cho, He He, Sainbayar Sukhbaatar, and Jason Weston [Pang et al. 2024b].

6.1 INTRODUCTION

Preference optimization has proven to give large gains when aligning pre-trained language models to human requirements compared to supervised fine-tuning alone [Ziegler et al. 2019; Stiennon et al. 2020]. Offline methods such as DPO [Rafailov et al. 2023] are becoming more popular for their simplicity and efficiency. Recent results have shown that iterative application of such an offline procedure is beneficial, whereby the updated model is used to construct new preference relations that are more informative, and hence improve results further. These methods include Iterative DPO [Xu et al. 2023b; Xiong et al. 2023], Self-Rewarding LLMs [Yuan et al. 2024], SPIN [Chen et al.

2024], and other methods [Rosset et al. 2024]. Common to these approaches is that they have been shown to perform well on general instruction tuning tasks, but they either make only moderate gains or even decrease the performance on standard reasoning tasks. While other kinds of iterative training methods have been applied successfully to reasoning, particularly involving the iteration of supervised fine-tuning (SFT) such as STaR [Zelikman et al. 2022], Rest^{EM} [Singh et al. 2024], and V-STaR [Hosseini et al. 2024]¹, using preference optimization to train the generative reasoning model is not applied in these methods.

In this work, we develop an approach to apply iterative preference optimization to reasoning tasks, with a particular focus on Chain-of-Thought (CoT) reasoning [Wu et al. 2023]. On each iteration we sample multiple chain-of-thought reasoning steps and final answers over training prompts, and then construct preference pairs such that pair winners have correct answers and pair losers have wrong answers. We then train a variant of DPO that includes a negative log-likelihood (NLL) loss term for the pair winners, which also proves crucial for performance. Given the newly trained model, we then iterate the procedure by generating new pairs, and training again, starting from the previously trained model. We find that reasoning performance improves over multiple iterations until it eventually saturates.

We show that our approach, termed *Iterative Reasoning Preference Optimization* (Iterative RPO), outperforms a number of baselines, including SFT or applying standard DPO, as well as other baselines from the literature. We see an improvement from 55.6% of zero-shot performance on GSM8K to 81.6% after our Iterative RPO training (or from 70.7% to 88.7% with majority voting out of 32 samples), from 77.8% to 86.7% on ARC-Challenge (without using the provided ARC Corpus), and from 12.5% to 20.8% on MATH (or from 18.8% to 29.1% with majority voting out of 32 samples), without using the provided pretraining corpus in MATH. We provide ablations that indicate the components that lead to these improvements. We also present analysis on how different objectives influence the probabilities of training sequences, which helps explain the success of our method.

¹V-STaR does use preference optimization, but for training a separate verifier model.

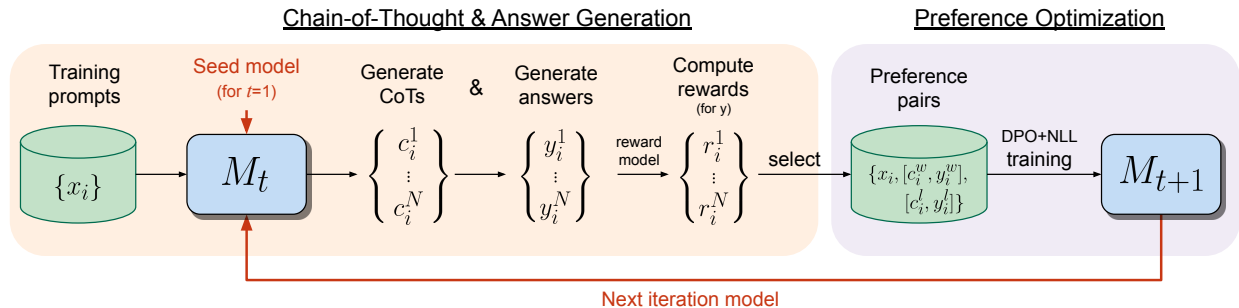


Figure 6.1: Iterative Reasoning Preference Optimization. Our iterative preference optimization method consists of two steps: (i) *Chain-of-Thought & Answer Generation*: training prompts are used to generate candidate reasoning steps and answers from model M_t , and then the answers are evaluated for correctness by a given reward model. (ii) *Preference Optimization*: preference pairs are selected from the generated data, which are used for training via a DPO+NLL objective, resulting in model M_{t+1} . This whole procedure is then iterated resulting in improved reasoning ability on the next iteration, until performance saturates.

Overall, our method provides a simple recipe that has the potential to improve the reasoning ability of LLMs over a wide range of tasks, as shown on the three tasks we consider.

6.2 ITERATIVE REASONING PREFERENCE OPTIMIZATION

Our approach first assumes access to a base, typically pretrained or instruction-tuned, language model, a set of training inputs, and the ability to judge the correctness of the final outputs. Given a training input, the language model is expected to generate (i) a set of reasoning steps (Chain-of-Thought), followed by (ii) a final answer to the given problem. We assume that we have access to a correctness measure for the final answer, and not for the correctness of the reasoning steps used to reach that answer. In our experiments, we thus consider datasets where gold labels are provided for training inputs, and a binary reward is derived by the exact match between these labels and the final answer generations. However, our approach could also be applied to settings with more general reward models.

On each iteration, our method consists of two steps, (i) Chain-of-Thought & Answer Generation and (ii) Preference Optimization, as shown in Figure 6.1. For the t^{th} iteration, we use the current

model M_t in step (i) to generate new data for training the next iteration’s model M_{t+1} in step (ii).

INITIALIZATION. We assume we are given an initial model M_0 , and a training set $D = \{(x_i, y_i)\}_i$ containing questions x_i and their correct answers y_i . The model will be trained and updated at each iteration, resulting in models M_0, M_1, \dots, M_T .

CHAIN-OF-THOUGHT $\hat{\mathcal{C}}$ ANSWER GENERATION. Given the current model M_t , we generate N different responses for every input, where each response consists of CoT reasoning c followed by a final answer y :

$$(c_i^n, y_i^n) \sim M_t(x_i) \quad \text{for all } x_i \in D \text{ and } n \in [N],$$

where we use $[N]$ to denote $\{1, 2, \dots, N\}$.

In the general version of our approach, one then computes the reward r_i^n for each of these responses based on the correctness of their answers, i.e., $r_i^n = R(y_i^n, y_i)$. In our experiments this simply corresponds to $r_i^n = 1$ if $y_i^n = y_i$, and 0 otherwise; i.e., whether the prediction matches the answer provided in the training dataset. Thus we have constructed a set of generated responses augmented with rewards:

$$G_i = \{c_i^n, y_i^n, r_i^n\}_{n \in [N]}.$$

PREFERENCE OPTIMIZATION. In the next step, we first construct a dataset of response pairs D_i^{pairs} based on the generations G_i from the current model M_t . The paired data is constructed such that chosen (winning) responses have higher rewards than rejected (losing) responses. This data is then used for preference optimization. In general, this can be done by selecting two responses for the same input, such that one has higher reward than the other, and setting the one with higher reward as the winner. In the binary reward case, we can split the generated responses G_i into two sets based on their rewards:

$$G_i^w = \{c_i^n, y_i^n \mid r_i^n = 1\},$$

$$G_i^l = \{c_i^n, y_i^n \mid r_i^n = 0\}.$$

Next we build a dataset of preference pairs by selecting a winner response (c_i^w, y_i^w) from G_i^w , and a loser response (c_i^l, y_i^l) from G_i^l . In particular, we simply iterate over G_i^w and G_i^l simultaneously² to produce K pairs of indices $\{(w_k, l_k)\}$, in order to ensure we use as much of the data as possible.

$$D_t^{\text{pairs}} = \{(c_i^{w_k}, y_i^{w_k}), (c_i^{l_k}, y_i^{l_k}) \mid \text{for all } x_i \in D \text{ and } k \in [K]\}.$$

Given the preference pairs, we can now train a new model M_θ that will become our next model M_{t+1} . The parameters θ are initialized from model M_t , and updated with a loss function that combines the DPO loss [Rafailov et al. 2023] for learning from the preference pairs, and the negative log-likelihood (NLL) loss for learning over the winning response from each pair. The loss corresponding to each preference pair is as follows:

$$\begin{aligned} \mathcal{L}_{\text{DPO+NLL}} &= \mathcal{L}_{\text{DPO}}(c_i^w, y_i^w, c_i^l, y_i^l \mid x_i) + \alpha \mathcal{L}_{\text{NLL}}(c_i^w, y_i^w \mid x_i) \\ &= -\log \sigma \left(\beta \log \frac{M_\theta(c_i^w, y_i^w \mid x_i)}{M_t(c_i^w, y_i^w \mid x_i)} - \beta \log \frac{M_\theta(c_i^l, y_i^l \mid x_i)}{M_t(c_i^l, y_i^l \mid x_i)} \right) - \alpha \frac{\log M_\theta(c_i^w, y_i^w \mid x_i)}{|c_i^w| + |y_i^w|}. \end{aligned} \quad (6.1)$$

Here $M(x)$ denotes the probability of sequence x under the model M , and σ is the sigmoid function. We use the previous iteration’s model M_t as the reference model in the denominator of the DPO term. Note that the NLL term is normalized by the total response length. The hyperparameter α balances the two loss terms. For brevity we omit the pair index k , but we optimize this loss on each of the $k \in [K]$ pairs generated for every input sample. At the end of this training, we thus obtain our next model $M_{t+1} = M_\theta$, which will be then used to build data for the subsequent iteration.

²If the iteration reaches the end of a set, it restarts from the first element. If one of the sets is empty, then that input will be ignored.

ITERATIVE TRAINING. Our overall procedure trains a series of models M_1, \dots, M_T where each successive model $t + 1$ uses preference data D_t^{pairs} created by the t^{th} model.

In our experiments, we define the models and the training data they use as follows:

M_0 : Base LLM; in our experiments we initialize with a fine-tuned instruction following model.

M_1 : Initialized with M_0 , then trained with D_0^{pairs} using $\mathcal{L}_{\text{DPO+NLL}}$.

M_2 : Initialized with M_1 , then trained with D_1^{pairs} using $\mathcal{L}_{\text{DPO+NLL}}$.

M_3 : Initialized with M_2 , then trained with D_2^{pairs} using $\mathcal{L}_{\text{DPO+NLL}}$.

M_4 : Initialized with M_3 , then trained with D_3^{pairs} using $\mathcal{L}_{\text{DPO+NLL}}$.

This approach can be seen as a similar, but simpler, instance of the Self-Rewarding LLM training scheme proposed in [Yuan et al. \[2024\]](#), with three differences. *Firstly*, on each iteration in Self-Rewarding a new set of prompts is created to explore the input distribution, but in our approach we use the same fixed set of prompts. *Secondly*, due to this choice our experimental setup does not require a sophisticated reward model to judge the model generations, as we assume the training prompts have provided gold labels which we compare to. These two omitted steps are challenging for reasoning tasks because they require a language model to verify correctness, which is known to be difficult [[Huang et al. 2024a](#)]. *Thirdly*, we show that our DPO+NLL objective is important for our reasoning tasks, whereas Self-Rewarding LLM has used the standard DPO objective.

Our approach is also related to the iterative training in the Self-Taught Reasoning (STaR) method [[Zelikman et al. 2022](#)], except that their approach uses SFT training, rather than preference optimization using DPO-like training. Preference optimization allows the use of negative examples of reasoning chains and answers, which we show improves performance. See [Section 6.4](#) for more discussion of related work.

6.3 EXPERIMENTS

6.3.1 MATH WORD PROBLEMS: GSM8K

In our first set of experiments, we use the GSM8K dataset [Cobbe et al. 2021]³ that contains real grade-school math word problems. For example the question: “*Natalia sold clips to 48 of her friends in April, and then she sold half as many clips in May. How many clips did Natalia sell altogether in April and May?*”. These questions typically require the model to perform intermediate reasoning, i.e. generating chain-of-thought before answering, otherwise performance is poor. Each problem contains a question x_i , gold chain-of-thought solution c_i , and a final numerical answer y_i . For our entire training process, we only use the training set of around 7.5k problems without any extra questions.

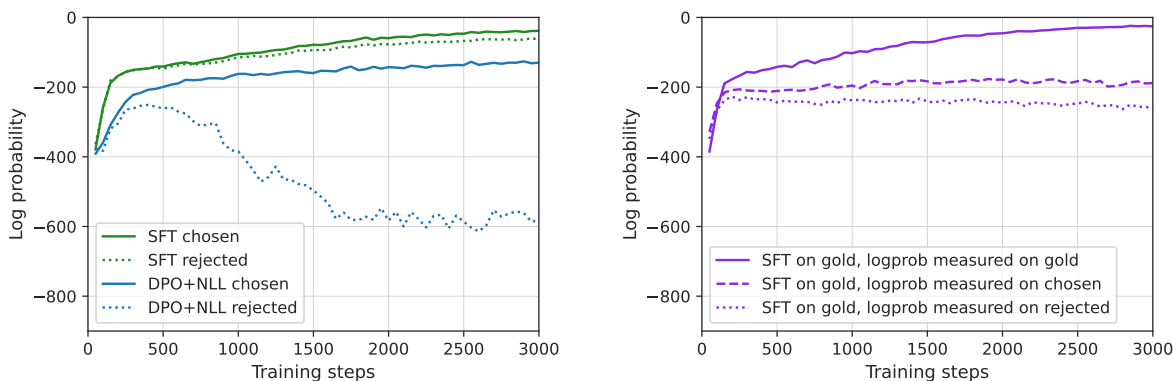
EXPERIMENTAL SETUP. As a seed model M_0 we use the chat version of Llama-2 70B model [Touvron et al. 2023b], which is instruction fine-tuned. We use a zero-shot prompt containing the question together with instructions to produce a chain-of-thought and to follow a specific format so the final answer can be easily extracted (the exact prompt is given in Appendix D.1.1). In each iteration, we generate $N = 30$ solutions per problem using sampling with temperature 0.8 for iterations 1–2 and temperature 1.3 for iterations 3–4 (hoping that there is a significant number of incorrect generations in later iterations). Since some problems might not have any model-generated correct solution, we include the gold human written solution (c_i, y_i) in the winning set G_i^w so it is not empty. Then we generate $K = 10$ pairs per problem for training with our loss in Equation 6.1, and filter out examples that were too long in terms of overflowing the context length or else do not have any incorrect generations. This procedure gives around 55–60k

³We have confirmed that the licenses of the datasets used in this paper (MIT for GSM8K and MATH, CC BY-SA 4.0 for ARC) are respected.

pairs for training, per iteration.⁴

In total, we perform four iterations, producing models M_1 , M_2 , M_3 , and M_4 . For each iteration, we train a maximum of 5000 steps, and then select the best checkpoint using a held-out 1k samples from the training set. We then retrain while including those 1k samples for the selected number of steps. The coefficient α is tuned in $\{0.25, 0.5, 1, 2\}$ when training M_1 , and we end up using 1 for all experiments in the paper. The coefficient β in the DPO loss is tuned in $\{0.05, 0.1, 0.5, 1.0\}$, and we end up using 0.1 in this experiment. We use a batch size of 16 and a learning rate $7e-7$ using the AdamW optimizer. Throughout this paper, all generation is done using one node containing eight V100 GPUs (32G memory). All training is done using eight nodes each containing eight A100 GPUs (80G memory).

Overall results are given in Table 6.1, where we give the exact match accuracy on the GSM8K test set.



(a) SFT trained on chosen seqs; init from Llama (b) SFT trained on gold CoTs; init from Llama

Figure 6.2: Effect of SFT training. (a) Although SFT training (solid green) is on chosen sequences (D_0^{pairs} , from iterative RPO iteration 1) only, the rejected sequence log probabilities (dotted green) also increase and are close to the chosen sequence probabilities. In contrast, our DPO+NLL training (blue) manages to decrease the rejected probabilities while increasing the chosen probabilities. This observation could potentially help explain why SFT-only performance lags significantly behind Iterative RPO Iteration 1 performance. (b) We show a similar plot but where SFT is trained on gold (dataset-provided) CoTs. Chosen and rejected sequence probabilities (which are from D_0^{pairs}) are still close to each other, but with a slightly bigger gap. Another observation is that the chosen sequence probabilities barely increase.

⁴If after filtering the number of pairs is larger than 60k, then we randomly select around 60k examples. This number is fixed because we do not want to introduce another source of variability in our experiments.

Table 6.1: GSM8K results comparing Iterative Reasoning Preference Optimization (Iterative RPO) against other baselines that are based on the same base model and training data. We report the exact match accuracy from a single generation (using greedy decoding), as well as majority voting over 32 generations (through sampling with temperature 0.8).

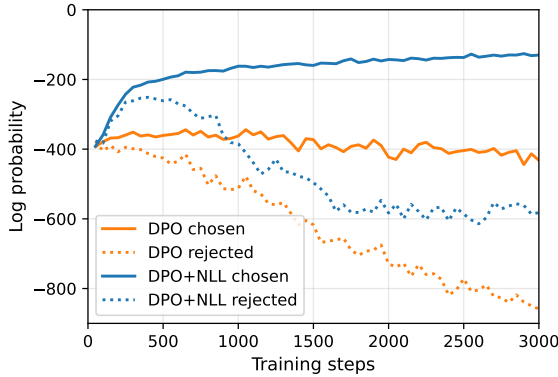
Model	Test Accuracy (%)
<hr/>	
Iterative RPO (<i>initialized from Llama-2-70b-chat</i>)	
<i>Iteration 1</i>	73.1
<i>Iteration 2</i>	78.0
<i>Iteration 3</i>	81.1
<i>w/ majority voting using 32 samples</i>	88.2
<i>Iteration 4</i>	81.6
<i>w/ majority voting using 32 samples</i>	88.7
<hr/>	
<i>Other Llama-2-70b-chat-initialized methods</i>	
Zero-shot CoT	55.6
<i>w/ majority voting using 32 samples</i>	70.7
DPO <i>initialized from Llama-2-70b-chat</i>	61.8
DPO <i>initialized from SFT trained on Iteration 1 chosen seqs</i>	60.3
SFT <i>on gold CoT examples</i>	63.5
STaR (1 iteration)	65.2
STaR (1 iteration, <i>but on twice as much data</i>)	66.9
Iterative RPO (1 iteration, <i>but initialized from SFT trained on chosen seqs</i>)	73.1
Iterative RPO (1 iteration, <i>but on twice as much data</i>)	74.8
<hr/>	

ITERATIVE RPO IMPROVES OVER BASELINES. We find that Iterative RPO outperforms zero-shot CoT, supervised fine-tuning (SFT) on the gold (dataset-provided) CoT solutions, and variants of DPO by a wide margin. SFT gives a boost in performance compared to zero-shot CoT from 55.6% to 63.5% but still far from the 81.6% of Iterative RPO. We apply standard DPO to the same set of preference pairs D_0^{pairs} as used in the first iteration of our method. Whether initializing from Llama-2-70b-chat (M_0) or from SFT training on the chosen (winner) examples, we find that DPO performance, while being better than zero-shot CoT, is no better than the SFT model, with accuracies of 61.8% or 60.3% respectively.

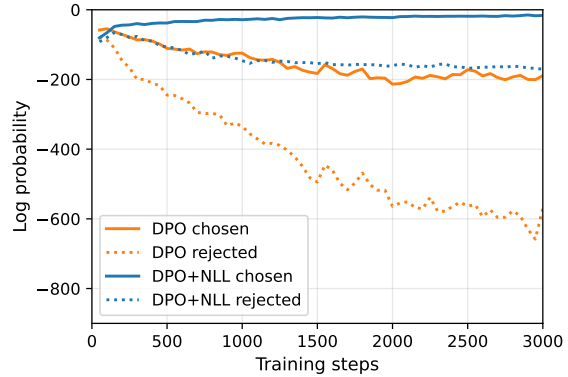
We also show that SFT on only the chosen CoT solutions, which corresponds to the first iteration of the STaR method, improves results to 65.2% over SFT on the gold solutions alone, but still falls short of the performance of the first iteration of Iterative RPO. One hypothesis for these improvements is the necessity of including the rejected sequences in the training objective; otherwise their probability increases along with the chosen samples; see [Figure 6.2](#). We note this observation has also been reported in concurrent work [[Hong et al. 2024](#)].

All of the results reported above are using a single generation at test time using greedy decoding. If we use majority voting over 32 samples (sampling with temperature 0.8), a standard approach to improve performance in the literature, we can improve the accuracy of our approach from 81.1% to 88.2% for iteration 3, and from 81.6% to 88.7% for iteration 4 of Iterative RPO. While performance is much improved using majority vote, this should be compared to a majority vote baseline, where we find a similarly large improvement over the zero-shot chain-of-thought with majority vote, which obtains an accuracy of 70.7%.

ITERATIONS OF ITERATIVE RPO YIELD IMPROVED REASONING. We observe that Iterative RPO provides improvements over its training iterations, increasing the base model accuracy by 47% (from 55.6% to 81.6%) in total. In contrast, supervised training using the gold CoT only brings about a 14% accuracy boost. We see performance improves across each iteration, from 73.1% to



(a) Initialized from Llama



(b) Initialized from SFT trained on chosen seqs

Figure 6.3: Effect of NLL loss term on DPO training for GSM8K. In our GSM8K experiments we observe the log probability of chosen sequences in standard DPO without NLL loss (solid orange) decreases over training steps, especially if the model is initialized from SFT training on chosen sequences (right). However, they *increase* over training steps when using DPO with NLL loss (solid blue). In all four settings, the margin between the two curves continues increasing. We find that DPO+NLL loss gives superior test accuracy in our experiments.

78.0% to 81.1% to 81.6%. However, the gain decays across the iterations (17.5%, 4.9%, 3.1%, 0.5%), indicating an upper limit on learning across iterations, especially as we are iterating across a fixed number of prompts, i.e., only from the training samples.

We also show that it is the iterations of updating the model (i.e., initializing from the previous model) that are helping, not just because there is more data in the form of new pairs generated from the fixed training set. To test this statement, we run the first iteration of Iterative RPO but on twice as much paired data by doubling K , and we run the STaR method first iteration with twice as much data as well. In both cases performance improves compared to less data, but not as much as performing two iterations. Iterative RPO with twice as much data obtains 74.8% (an improvement over 73.1% using the original dataset size); however, training for two iterations obtains 78.0%. For STaR, training on twice as much data obtains 66.9%, compared to 65.2% with the original data, which is still a much lower performance than Iterative RPO.

NLL LOSS IS NECESSARY IN OUR METHOD: DPO WITH NLL vs. DPO WITHOUT NLL. The first iteration of our method can be compared to standard DPO training, which uses the same preference

data, as reported in [Table 6.1](#). We see a large performance drop (73.1% vs. 61.8%) using DPO compared to our method after one iteration. The gap remains large even when the standard DPO training starts from the superior SFT-tuned model, which it has been argued improves DPO’s performance [[Rafailov et al. 2023, 2024](#)]. Our results support the need of the NLL loss term in our training, not just using SFT for initialization. To further understand this, we plot the sequence-level log probability over training steps for these methods in [Figure 6.3](#). We see that for DPO without NLL loss there is a decrease over training for the chosen sequences, whereas for DPO with NLL there is not, which may help explain the improved performance of the latter. We note that related observations have been made elsewhere in various settings [[Pal et al. 2024](#); [Xu et al. 2024](#); [Hong et al. 2024](#)]. Further, we note that whether we initialize with Llama-2-70b-chat or SFT on chosen for Iterative RPO, accuracy results of first iteration training do not seem to deviate (both obtain the same score 73.1%). This is another advantage of our method as the training process is simpler without the SFT step.

OTHER RESULTS IN THE LITERATURE. We can compare our results to others in the literature, even if their experiments are in different settings. [Touvron et al. \[2023b\]](#) reports an accuracy of 56.8% for 8-shot Llama-2-70b, which is close to our zero-shot CoT results for Llama-2-70b-chat. In terms of closed-source proprietary language models, some results are superior to ours, while others are not; for example GPT-4 obtains 92.0% (5-shot chain-of-thought) [[Achiam et al. 2023](#)], Claude 2 obtains 88.0% [[Anthropic Team 2023](#)], PaLM 2 obtains 80.7% [[Anil et al. 2023](#)], while GPT-3.5 obtains 57.1% (5-shot) [[Achiam et al. 2023](#)]. We note that the size (number of parameters) and the makeup of the training set of some of these models have not been fully disclosed. For results that use the same size and class model, Llama-2-70b, MetaMath [[Yu et al. 2024](#)] reports an accuracy of 82.3%, while WizardMath reports 81.6% [[Luo et al. 2023a](#)]. These last two results use additional augmented training data, whereas our method does not use additional prompts. Such approaches should be orthogonal to ours, and both can provide benefits.

6.3.2 ARC-CHALLENGE TASK

To test reasoning capabilities outside of mathematics, we employ ARC [Clark et al. 2018] which covers multiple science subjects. Questions are multiple-choice, for example: “A *fold* observed in layers of sedimentary rock most likely resulted from” with four possible answers, e.g., “(A) cooling of flowing magma, (B) converging of crustal plates, (C) deposition of river sediments, or (D) solution of carbonate minerals”. The training dataset contains 7.7k questions split into easy and challenge sets. We report results on the ARC-Challenge test set which has 1172 examples. There is no gold chain-of-thought reasoning provided for training examples in this task. Our method does not have that requirement and hence can still be applied as we only compute rewards based on the final answer. One consequence however is that if there is no model-generated correct solution for a question, then that question is not included in our training. We follow the same setup as before to first generate reasoning and then a final answer by the models (see [Appendix D.1.1](#) for prompt) to construct data for iterations of Iterative RPO. We only train on the training set (both easy and challenge sets) and *do not* utilize the supporting ARC Corpus.

Specifically, in each iteration, we generate $N = 30$ solutions per problem using sampling with temperature 0.8 for iterations 1–2 and temperature 1.3 for iteration 3. We select $K = 20$ pairs of solutions per problem. We end up with around 20k example pairs for iteration 1, 11k example pairs for iteration 2, and 5k example pairs for iteration 3. The decrease in the number of examples is due to the lack of incorrect samples for a number of questions in later iterations. Each iteration is trained on a maximum of 4000 steps. The hyperparameter tuning relies on the provided development set.

We hence perform experiments using a very similar setup to the one previously described for GSM8K. Overall results are given in [Table 6.2](#). We again find that Iterative RPO provides increased performance across iterations (84.8%, 86.2%, 86.7%) over three iterations. Majority voting using the model in the third iteration (32 samples, temperature 0.8) leads to another small boost (87.9%).

Table 6.2: ARC and MATH results. We compare Iterative Reasoning Preference Optimization (Iterative RPO) against other baselines that are based on the same base model and training data.

Model	ARC-Challenge (0-shot) Test Accuracy (%)	MATH (4-shot) Test Accuracy (%)
<i>Iterative RPO (initialized from Llama-2-70b-chat)</i>		
<i>Iteration 1</i>	84.8	17.7
<i>Iteration 2</i>	86.2	19.9
<i>Iteration 3</i>	86.7	20.8
<i>w/ majority voting using 32 samples</i>	87.9	29.1
<i>Other Llama-2-70b-chat-initialized methods</i>		
CoT	77.8	12.5
<i>w/ majority voting using 32 samples</i>	82.9	18.8
SFT on chosen sequences	79.8	16.8
DPO initialized from Llama-2-70b-chat	82.8	12.4
DPO init from SFT model trained on chosen seqs	83.5	10.5

These results outperform zero-shot CoT (77.8%), SFT on chosen sequences (79.8%) and standard DPO (83.5%). We arrive at similar observations in [Figure 6.4\(a\)](#) compared to [Figure 6.3](#): when training with DPO without NLL loss, the log probabilities of chosen sequences barely increase over training; when training with DPO with NLL loss, the log probabilities increase noticeably.

Even though we arrive at similar conclusions to the ones from GSM8K, we find these results especially noteworthy due to the multiple-choice nature of the task. As there are typically only four possible answers, the generated data in step (i) of Iterative RPO may provide a CoT and a final answer that is correct by luck (as random guessing is correct 25% of the time). Hence, the nature of the task may introduce a significant amount of noise in the CoT generations used in preference optimization in step (ii). Nevertheless, the method seems robust to this issue and we still observe performance gains.

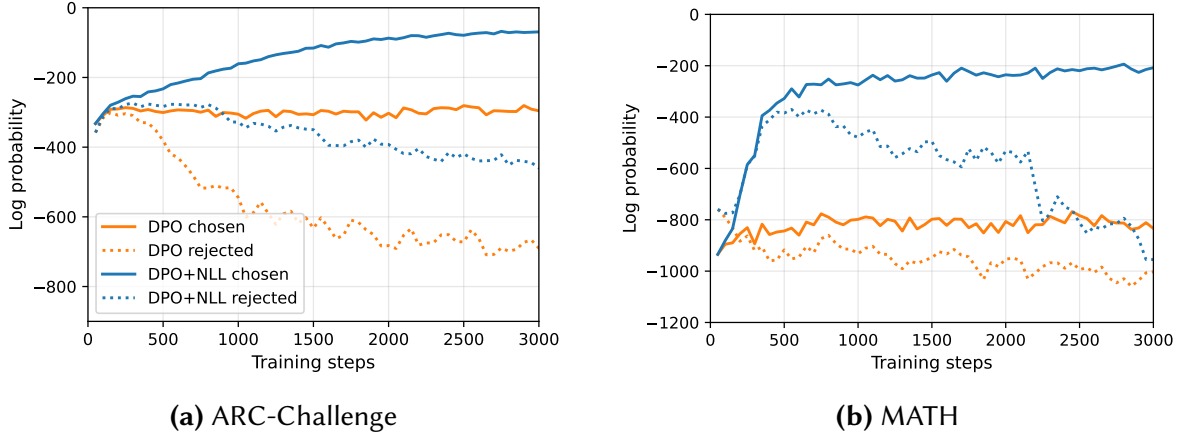


Figure 6.4: Effect of NLL loss term on DPO training for ARC and MATH. The legend on the right plot is omitted due to space constraint, but it is the same as the legend in the left plot. Similar to GSM8K, in ARC-Challenge and MATH, we see that the log probabilities of chosen sequences barely increase over training steps when training with DPO. However, when training with DPO with NLL loss, the log probabilities increase over training steps.

6.3.3 MATH TASK

We also experiment with more advanced math problems using the MATH [Hendrycks et al. 2021] dataset that is composed of 12,500 competition problems, for example the question: “*Tom has a red marble, a green marble, a blue marble, and three identical yellow marbles. How many different groups of two marbles can Tom choose?*”. While this may look superficially similar to the GSM8K task, it features substantially harder questions, as will be shown by the baseline performance. The test set has 5,000 examples. Similar to the GSM8K dataset, a gold CoT solution is provided for each problem, and the gold answers can be matched uniquely to predicted answers after normalization to compute rewards. *We do not use the accompanying pretraining data.* For each MATH question, we use a few-shot prompt given in Appendix D.1.1 as the input to the language model. In particular, the prompt includes four fixed in-context examples chosen from the training set. The language model needs these demonstrations so that the final answers can be properly formatted in \LaTeX .

In each iteration, we generate $N = 20$ solutions per problem using sampling with temperature

0.8 for iterations 1–2 and temperature 1.0 for iteration 3. We select $K = 15$ pairs of solutions per problem, and after filtering out pairs with overly long generations, for each iteration we end up with around 75k example pairs. We train a maximum of 5000 steps per iteration; other details are similar to GSM8K setups.

Results are given in [Table 6.2](#). We again find that Iterative RPO provides increased performance across iterations, from 17.7% to 19.9% to 20.8% over three iterations. Majority voting (32 samples, temperature 0.8) leads to a significant boost in performance (29.1%). These results outperform few-shot CoT (12.5%), SFT on chosen sequences (16.8%) and standard DPO (12.4%). In particular, DPO degrades the performance compared to initialization. Similar to the previous tasks, we show the log-probabilities during training in [Figure 6.4\(b\)](#).

Overall, we find on all three distinct tasks we tried, from simpler to more difficult, similar observations on performance gains are exhibited by our method.

6.4 RELATED WORK

GENERAL ITERATIVE ALIGNMENT METHODS. Several works have implemented iterative reinforcement learning from human feedback (RLHF) with a human-in-the-loop to provide additional labels to retrain the reward model at each iteration, e.g., via Proximal Policy Optimization (PPO) [[Schulman et al. 2017](#)], reporting improvements across iterations [[Bai et al. 2022](#); [Touvron et al. 2023b](#)]. Recently, approaches have been proposed to perform iterative alignment without a human-in-the-loop. Iterative DPO [[Xu et al. 2023b](#); [Xiong et al. 2023](#)] optimizes preference pairs using DPO [[Rafailov et al. 2023](#)] at each iteration, and then constructs new preference pairs for the next iteration by generating them using the updated model, and scoring them using a reward model. Other iterative methods than DPO exist as well, such as the Cringe loss [[Adolphs et al. 2023](#)], Pairwise Cringe Loss [[Xu et al. 2023b](#)], and ReST [[Gulcehre et al. 2023](#)].

SPIN [[Chen et al. 2024](#)] is an Iterative DPO-like framework that uses human labels as the

winning response in a pair, and the last iteration’s generations as the losing response in the pair. The authors note this has the limitation that once the model generations reach human performance, they are bottlenecked. Further, each input prompt is required to have a human-annotated generation. In contrast, our work only requires the final answer, but not the reasoning steps, and crucially uses the model to generate both winning and losing Chain-of-Thoughts. Only modest gains on reasoning tasks are reported in their work.

Self-Rewarding LLMs [Yuan et al. 2024] also use Iterative DPO with the LLM itself used as a reward model to construct pairs for each successive iteration. Both that work and the work of Rosset et al. [2024] and Snorkel AI Team [2023], which do similar iterations but with external reward models, show significant gains on general instruction following tasks. However, again, only modest gains on reasoning tasks are reported.

METHODS IMPROVING REASONING ABILITY. While a number of approaches have been developed to curate or distill training data for reasoning tasks [Yu et al. 2024; Toshniwal et al. 2024], in this work we focus on learning algorithms which is an orthogonal axis. Expert Iteration assumes a reward model, and repeatedly uses rejection sampling to filter generations and train on them, which is found to match the sample complexity of PPO [Havrilla et al. 2024]. STaR [Zelikman et al. 2022] relies on a similar loop: generate rationales to answer many questions, prompted with a few rationale examples; if the generated answers are wrong, try again to generate a rationale given the correct answer; and then fine-tune on all the rationales that ultimately yielded correct answers; and repeat. ReST^{EM} [Singh et al. 2024] assumes a ground truth verifier and also fine-tunes on filtered samples in a repeated fashion. All these methods rely on finding high-quality samples for SFT-like training, rather than using DPO-like pairwise preference optimization as in our work.

The V-STaR method [Hosseini et al. 2024] trains a verifier using DPO and uses this to filter the generations of a model trained by SFT, rather than using DPO to train the generator, as we do. MAPO [She et al. 2024] also recently utilizes DPO but for multilingual reasoning tasks, where

they translate across languages.

6.5 CONCLUSION

We proposed an iterative training algorithm, Iterative Reasoning Preference Optimization, for improving chain-of-thought-based reasoning task performance in LLMs. In each iteration, we generate multiple responses and build preference pairs based on the correctness of their final answers, and then use a modified DPO loss with an additional NLL term for training. Our method does not require human-in-the-loop or extra training data, and remains simple and efficient to implement. The experimental results show large improvements on GMS8K, MATH, and ARC-Challenge over various baselines using the same base model and training data. These results indicate the effectiveness of our recipe of iterative training in improving the reasoning capabilities of LLMs.

6.6 LIMITATIONS

On experiments: When training iteration t using iterative RPO, we do not make use of the collected data in *previous* iterations. Utilizing those data could potentially boost the performance even more. We leave this point to future work, as it is not central to our theses. In addition, we have experimented on three tasks. It is unclear how the approach would perform on general instruction tuning tasks without a clear *best* answer, but we argue that positive results on the three tasks in this paper can already prove the method useful. The current recipe requires correct answers, and a clear metric for comparing a generated response with this correct answer.

Regarding our loss function: The NLL loss is shown to be helpful in our case. Our iterative RPO algorithm requires training data to be mostly collected from the previous iteration of the model. Therefore, the chosen and rejected sequences all have reasonably high probability under

the model distribution. When training sequences are arbitrary (e.g., sampled from other models), it is unclear whether the NLL loss is necessary (although this setting does not fall under the umbrella of the iterative RPO procedure in this paper).

6.7 DEVELOPMENTS SINCE PUBLICATION

While this work has only been available to the public for a short time at the time of writing, there have been at least two notable updates. First, through more comprehensive experiments, [Setlur et al. \[2024\]](#) have shown the necessity of having model-generated negative responses in improving math reasoning. Second, ESM3 [[Hayes et al. 2024](#)] is an acclaimed multi-modal language model in the protein domain; it can reason over three protein modalities: the sequence, the structure, and the function. To generate proteins that fully reflect the input prompts, the work has used IRPO as the alignment algorithm.

7 | CONCLUSION AND FUTURE DIRECTIONS

This dissertation provides two case studies of selecting reward functions to address two separate issues. To tackle the challenge that obtaining rewards in dialogue could be expensive due to the cost of human annotations, we propose extracting implicit rewards from deployment data. While it is very encouraging that automatic and human evaluations indicate improvements in our model generations over baseline responses, we also find that certain proxy signals can lead to more generations with undesirable properties. Given that the generation model could make little or no progress in learning using classic rewards in NMT, we attempt a denser reward that is high-quality, based on noisy channel decoding. The result is that we are able to massively speed up decoding compared to the “beam search and rerank” approach of doing noisy channel decoding, while ensuring similar translation quality.

Rewards can be bad and optimization could be difficult. We should innovate on learning algorithms as well. We thus propose two effective algorithms in the dissertation. First, we use a simple reward but explore the extreme case where the agent (i.e., generator) has no interaction with the environment so we do not deviate too far from references. The learning algorithm can be interpreted as policy gradient with importance weighting, and intuitively the GOLD algorithm upweights confident tokens and downweights unconfident ones in the reference during training. Generations from models trained by GOLD are superior to the ones generated by models trained by MLE, through both automatic and human evaluations, and GOLD is subsequently used in a range of tasks (e.g., coding, image captioning) with LLMs. Finally we also discuss a case where RL

is not needed. We propose iterative reasoning preference optimization that optimizes for winning vs. losing reasoning chain-of-thoughts. The objective is a variant of direct preference optimization, and it achieves massive improvements on reasoning tasks (e.g., 55.6% to 81.6% on GSM8K and 88.7% with majority voting using 32 samples, using llama-2-70b-chat).

Here are a few ideas which could enable us to connect the two threads. Firstly, while we have explored reward functions that depend on exact matches or probabilities, we shall not underestimate the power of directly using LLM generations as rewards using ideas similar to LLM-as-a-judge [Zheng et al. 2023]. Secondly, to further boost the ability to have LLMs as the reward function, one option is to investigate the possibility of human-LLM collaboration such that the ultimate evaluation performance exceeds that of humans’ and that of LLMs’.

7.1 REWARDS CAN BE BASED ON LLM OUTPUTS

7.1.1 SELF-REWARDING LANGUAGE MODELS

Training Large Language Models (LLMs) using human preference data can vastly improve the instruction following performance of pretrained models [Ouyang et al. 2022; Bai et al. 2022]. The standard approach of Reinforcement Learning from Human Feedback (RLHF) learns a reward model from these human preferences. The reward model is then frozen and used to train the LLM using RL, e.g., via PPO [Schulman et al. 2017], and the human labeling process is then possibly repeated in order to improve the reward model [Ziegler et al. 2019]. A recent alternative is to avoid training the reward model at all, and directly use human preferences to train the LLM, as in Direct Preference Optimization [DPO; Rafailov et al. 2023]. In both cases, the approach is bottlenecked by the size and quality of the human preference data. Furthermore, we hypothesize that training solely on human preferences will constrain models from improving beyond human level and achieving superhuman performance, which will require superhuman feedback.

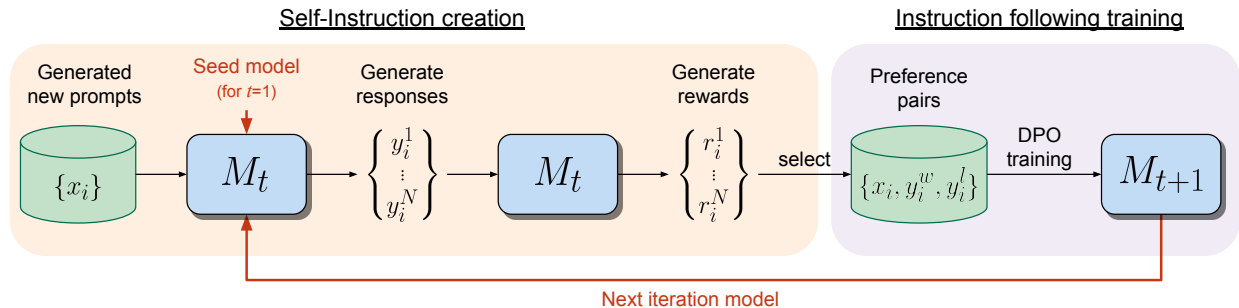


Figure 7.1: Self-Rewarding Language Models. Our self-alignment method consists of two steps: (i) *Self-Instruction creation*: newly created prompts are used to generate candidate responses from model M_t , which also predicts its own rewards via LLM-as-a-Judge prompting. (ii) *Instruction following training*: preference pairs are selected from the generated data, which are used for training via DPO, resulting in model M_{t+1} . This whole procedure can then be iterated resulting in both improved instruction following and reward modeling ability.

This work proposes to train a self-improving reward model in order to avoid this bottleneck. Unlike traditional methods where the reward model is frozen or requires human-labeled data in order to be updated, our model is designed to continuously update itself during LLM alignment. The key to such an approach is to develop an agent that possesses all the abilities desired during training, rather than separating them out into distinct models such as a reward model and a language model. In the same way that pretraining and multitasking training of instruction following tasks allow task transfer by training on many tasks at once [Collobert and Weston 2008; Radford et al. 2019; Ouyang et al. 2022], incorporating the reward model into that same system allows task transfer between the reward modeling task and the instruction following tasks.

We thus introduce *Self-Rewarding Language Models*, that both (i) act as instruction following models generating responses for given prompts; and (ii) can generate and evaluate new instruction following examples to add to their own training set. We train these models using an Iterative DPO framework similar to that recently introduced in Xu et al. [2023b]. Starting from a seed model, in each iteration there is a process of *Self-Instruction creation* whereby candidate responses are generated by the model for newly created prompts, and are then assigned rewards by that same model. The latter is implemented via LLM-as-a-Judge prompting, which can also be seen as an

instruction following task. A preference dataset is built from the generated data, and the next iteration of the model is trained via DPO, see [Figure 7.1](#).

In our experiments, we start with a Llama 2 70B [[Touvron et al. 2023b](#)] seed model fine-tuned on Open Assistant [[Köpf et al. 2023](#)], and then perform the above training scheme. Fine-tuning Llama 2 70B on three iterations of our approach yields a model that outperforms many existing systems on the AlpacaEval 2.0 leaderboard, including Claude 2, Gemini Pro, and GPT-4 0613. We find that not only does the instruction following performance improve from Self-Rewarding LLM alignment compared to the baseline seed model, but importantly the reward modeling ability, which is no longer fixed, improves as well. This means that the model during iterative training is able, at a given iteration, to provide a higher quality preference dataset to itself than in the previous iteration. While this effect likely saturates in real-world settings, it provides the intriguing possibility of obtaining reward models (and hence LLMs) that are superior to ones that could have been trained from the original human-authored seed data alone.

In short, while there is much left still to explore, this work opens the door to the possibility of models that can continually improve in both axes.

7.1.2 EXTENDING IRPO TO UNSUPERVISED TASKS

One of IRPO’s major assumption is that we have the reference answers (not necessarily chain-of-thoughts, but we do need the answers) for each question. The natural step is to investigate whether IRPO could work in an unsupervised setting, where we do not have the reference answers. One possibility is that the answers could be obtained through majority voting while potentially leveraging confidence metrics.

Another possibility is to use LLMs to score the quality of the sampled CoTs. There has been evidence claiming that LLMs do poorly at critiquing [[Valmeekam et al. 2023](#); [Luo et al. 2023b](#)] and the judging ability is far from perfect [[Lambert et al. 2024](#)], so a natural and extremely critical next step is to develop training set that improves LLMs’ evaluation ability, and then use LLMs as

the scoring function so as to perform IRPO in an unsupervised way.

Finally, it is also critical to keep developing better algorithms that learn from preference data and algorithms that learn from feedback in general.

7.2 HUMAN-LLM COLLABORATION

As mentioned above, self-rewarding language model is only a first step on the path of developing a reward model (which is an LLM) that is superhuman. An orthogonal way of achieving superhuman evaluation performance is to potentially leverage humans, especially regular non-expert humans given that experts could be hard to recruit and exceedingly expensive [Rein et al. 2023]. Can we develop a scenario where on a task, (1) the human non-expert accuracy is the lowest; (2) the LLM accuracy is second lowest; (3) the human expert accuracy is far higher than both; and ideally (4) the Human-LLM collaboration result approaches the accuracy by human experts, or is even higher than all previous three accuracies?

I have been involved in building benchmarks toward this direction; in particular, QuALITY [Pang et al. 2022b] and GPQA [Rein et al. 2023]. QuALITY (question answering with long-input text) is a question-answering benchmark with on average 5k words. However, given that it is expensive and difficult to build natural datasets that satisfy the criteria discussed above (with non-saturating performance by top LLMs), for this human-LLM collaboration direction, the research community has started by adapting QuALITY in an artificial way [Michael et al. 2023]. Specifically, non-experts are defined as well-incentivized high-quality crowdworkers but with time constraints (e.g., 5 minutes) when reading the article; experts are the high-quality crowdworkers that have unlimited time in reading the article. GPQA (graduate-level Google-proof question answering) is a recent high-quality benchmark developed with high cost even though it only has a few hundred questions. The questions are on biology, physics, and chemistry, and even GPT-4 achieves only 39% accuracy on GPQA (chance accuracy: 25%). The hope is that GPQA would be a much harder

benchmark and the definition of non-experts would be far less artificial.

The benchmarks would allow us to design such methods where human-LLM collaboration results in expert-level or even superior performance. There has been preliminary evidence that such a route is possible in machine translation where a model provides the span-level error annotation first before having humans annotate [Zouhar et al. 2024]. The result is that the AI-assisted human annotation cuts annotation time by half and is more accurate than both the human annotators and the error annotation model – intuitively, the model could provide annotations that the human annotation has not realized.

Debate is another promising path. As Michael et al. [2023] show, using QuALITY as the testbed as discussed above, debate between unreliable experts (either humans or AIs) can make a non-expert judge better at arriving at the correct final answer. The hope is that the debate process would make LLMs less likely to falsely represent information and get away with arguments which the other debater questions in a convincing way. Different LLMs could debate, and humans could act as the moderator (e.g., asking follow-up questions) and judge. Another related paradigm is solve-and-probe where one LLM could be the main problem solver, and the other LLM and non-expert humans could be the prober and verifier.

The bigger picture is that we hope that human-LLM collaboration would further boost the evaluation ability compared to humans or LLMs on their own, thereby improving the quality of reward function to a superhuman level and consequently boosting the LLM general instruction following and reasoning capabilities. This ultimate goal would naturally involve other directions as well, including developing better strategies of tool use, better cognitive architectures (e.g., integrating more memory types and developing ways to represent these memories).

A | APPENDIX FOR CHAPTER 3

A.1 OTHER SIGNALS

NUMBER OF WORDS IN ALL FUTURE HUMAN TURNS OR NUMBER OF FUTURE HUMAN TURNS. We build variants of the “replied” and “length” signals by taking into account multiple future turns to build the scoring function. For the “number of words in all future human turns” signal, let $y(x_i^b)$ be 1 if the length of all future human turns is larger than a threshold k . Otherwise, set the score to 0. For the “number of future human turns” signal, let $y(x_i^b)$ be 1 if there are $\geq k$ human utterances in the future. Intuitively, if a bot turn approaches the end of the conversation episode, then it may be an inferior one.

For the “number of words in all future human turns” signal and the “number of future human turns” signal, the best accuracy is 0.595 (experimented with threshold $k = 30, 50, 100$) and 0.587 (with threshold 2, 3, 5, 10), respectively. We have also attempted restricting the classification problem to conversations with at least 3, 5, or 10 human turns – the accuracy is still < 0.6 . We consider the accuracy to be too low for the dialogue response generation experiments, so we discard these signals in the rest of our experiments.

A.2 ADDITIONAL INFO ON EXPERIMENTAL SETUP

MODELS. The classifiers are fine-tuned based on a RoBERTa-large with 24 layers, 16 attention heads, token embedding size 1024, and FFN size 4096. [Table 3.1](#) examines the accuracy of the classifiers based on different implicit feedback signals under a balanced dev set (where the number of examples is equal across classes).

Our baseline model is the publicly released BlenderBot model (r2c2_blenderbot_3B) with around 3B parameters, pretrained on both dialogue and language modeling tasks, and fine-tuned on dialogue tasks [[Shuster et al. 2022a](#)]. The model has 2 encoder layers, 24 decoder layers, 32 attention heads, FFN size 10240, and embedding size 2560.

DATA. In addition, regarding data, we have confirmed that it is legal to use the deployment data [[Xu et al. 2023a](#)] from which we obtain the implicit feedback signals. The deployment data is released under a CC BY license, as shown on [this](#) page.

COMPUTE. Classifier (r_ϕ) training is done on one V100 with 32G memory. Depending on the signal, the training time varies, but on average we train the classifier for 72 hours. Sample-and-rerank decoding experiments (to generate the dialogue responses) are run on eight V100s, each with 32G memory. Decoding every 100 dialogue responses take less than 30 minutes.

HYPERPARAMETERS. All experiments are run using ParlAI. To train the classifiers, a grid search is done. The learning rate is selected from {3e-6, 5e-6, 1e-5, 3e-5}. Both the dropout rate and the attention dropout rate (in transformers) are selected from {0, 0.1}. The learning rate scheduler is ReduceLRonPlateau in PyTorch. The learning rate scheduler patience is selected from {5,10,50}. Batch size is kept constant at 20. Gradient clip is 1.0. The validation metric is the classification accuracy on dev sets. Validation is done every 3000 seconds. We use the Adamax optimizer. To

Table A.1: Performance of generated dialogue responses using different implicit feedback signals. **Classifier accuracy:** the classification accuracy on a balanced dev set (meaning the classes corresponding to the same number of examples); even though the accuracy is not high, we see that the classifiers can still help improve the bot dialogue generations. **Avg. score:** our new generations achieve better classifier scores compared to the baseline; this observation is guaranteed given our sample-and-rerank algorithm but we believe it is still instructive to see how large the gap is. **Length:** we see that other than the “baseline + length ($k=5$)” generation, the other generations’ average length are similar, so the model is not simply optimizing for longer responses. **Sig.:** ** if p -value $\in [0, 0.05)$, * if p -value $\in [0.05, 0.1)$, – otherwise. We find general agreement between human annotator evaluation results and the LLM evaluation results when averaging over 200 examples.

	classifier accuracy under balanced dev set	avg. score of generations scored by classifier (baseline / new)	avg. length of generations	annotator pref. (baseline / new / tie)	sig.	LLM pref. (baseline / new / tie)	sig.
baseline	–	–	19.7	–	–	–	–
ranked by probability	–	–	18.1	27.0 / 30.0 / 43.0	–	–	–
baseline + replied	0.678	0.957 / 0.999	20.2	33.0 / 32.0 / 35.0	–	43.0 / 45.0 / 12.0	
baseline + length ($k=20$)	0.761	0.332 / 0.708	21.9	31.0 / 43.0 / 26.0	**	36.5 / 48.5 / 15.0	*
baseline + length ($k=5$)	0.624	0.587 / 0.740	24.2	31.0 / 36.0 / 33.0	–	42.0 / 47.0 / 11.0	–
baseline + non-neg. sentiment & length ($k=5$)	0.603	0.524 / 0.634	21.9	29.0 / 37.5 / 33.5	*	33.0 / 52.0 / 15.0	**
baseline + positive sentiment & length ($k=5$)	0.670	0.506 / 0.742	19.4	31.5 / 38.0 / 30.5	–	40.5 / 50.5 / 9.0	*
baseline + joy & length ($k=5$)	0.675	0.486 / 0.766	19.4	27.0 / 36.5 / 36.5	**	35.5 / 50.5 / 14.0	**

generate dialogue responses, we use sample-and-rerank: the number of samples for sample-and-rerank is fixed at 20; the p for factual top- p decoding is 0.9.

A.3 ADDITIONAL INFO ON EVALUATION

A.3.1 CROWDWORKER EVALUATION OF PAIRWISE COMPARISON

We ask crowdworkers to decide which one of the two responses is better or if they tie. Each judgment is done by five crowdworkers. The final answer is the majority vote. If there is no majority vote (e.g., five votes being “(a) wins,” “(a) wins,” “(b) wins,” “(a) and (b) tie,” “(a) and (b)

tie”), then the final answer is “(a) and (b) tie.”

The specific instruction is as follows. The header says the following: “We want to investigate the quality of responses by different dialogue models. Warning: We added many dummy tasks – we already know the (unambiguous) reference answers for them. If you answer too many of those incorrectly, we will block you from all future tasks from our group. We may also reject your work for this reason. Thanks again for your hard work! (WARNING: May contain offensive/controversial content. Discretion advised. In addition, your responses will be used for AI research, and your annotation may be released.)” The main text says the following: “Read the conversation below and consider the two possible next responses by SPEAKER 1. A response is considered good if it is sensible, engaging, and friendly. Which of the two responses from SPEAKER 1 is much better than the other one? If they are similarly good or bad, then answer tie.” The average pay is 18–27 dollars per hour before fees (assuming 20–30 seconds per evaluation on average), which is above the minimum wage in the region.

We add 10% catch questions (which are questions with known unambiguous answers) and if the crowdworker gets too many incorrect,¹ then we discard all their ratings and relaunch the annotation jobs for the corresponding examples. If the two candidate generations are exactly the same, we automatically label the result as “tie” and do not include the annotation batch.

The crowdworker–expert agreement is much better than the LLM–expert agreement: §A.3.3 and Table A.4 show that 86% answers match, and 6.5% strongly disagree.²

A.3.2 ADDITIONAL RESULTS TO COMPLEMENT TABLE 3.1

Table A.1 is presented to complement the results in Table 3.1 in the main text. Each cell corresponds to 200 evaluated examples (except for “annotator pref.,” we first do 100 annotations, and then do the second 100 annotations only for rows with large enough “new wins” minus “baseline wins”

¹>20% if the worker has done ≥ 5 annotations (of catch questions), >50% if the worker has done < 5 annotations.

²Strongly disagree: crowdworkers choosing “(a) better than (b)” and experts choosing “(b) better than (a),” or vice versa. The “tie” annotations are not considered.

value – the “length ($k=20$)” row and the sentiment-/joy-related rows). The significance test is done with Wilcoxon signed-rank test [Wilcoxon 1992; Dror et al. 2018].

“RANKED BY PROBABILITY” RESULTS. We also collect human annotations for generations corresponding to “ranked by probability” vs. generations corresponding to the “length ($k=20$)” signal and the “joy & length” signal. The results (“ranked by prob” wins / new wins / tie) for the “length ($k=20$)” signal: 29.5 / 37.0 / 33.5 (*). The results for the “joy & length” signal: 29.0 / 33.0 / 38.0 (-).

LLM PAIRWISE EVALUATION. We complement the Table 3.1 results with the LLM-evaluated pairwise preference results, as shown in Table A.1. While instance-level LLM vs. expert agreement is not high (Table A.2), we find general agreement between LLM evaluation results and the crowdworker evaluation results when averaging over 200 examples (Table A.1). For exact prompts, see §A.3.4 and search for the “comparison” paragraph.

MORE ON BEHAVIOR ANALYSIS IN TABLE 3.1. If we remove our generations that are off-topic, controversial, unfriendly, insincere, and only evaluate on the rest of the examples, then the human annotation would prefer our implicit feedback model generations more: the “baseline generation wins” vs. “new generation wins” vs. “tie” proportion would be 31.6 / 47.5 / 20.9 for the “length ($k=20$)” signal (better than the 31.0 / 43.0 / 26.0 result in Table A.1), and 24.5 / 41.3 / 34.2 for the “joy & length” signal (better than the 27.0 / 36.5 / 36.5 result in Table A.1).

A.3.3 AGREEMENT OF EXPERT, ANNOTATOR, AND LLM EVALUATION OF PAIRWISE COMPARISON

Initially, we have conducted evaluation using LLM (specifically, gpt-3.5-turbo-0613), hoping to save cost. The rationale is two-fold: first, model-based evaluation (especially with in-context CoT examples) has shown to perform well on a range of tasks [Gilardi et al. 2023] and crowdsourcers

Table A.2: Confusion table of LLM evaluations vs. expert evaluations. “Gen” stands for “generation.” 64.5% annotations match; 14% annotations strongly disagree (as defined in §3.4.2, the % of annotations that strongly disagree equals the % of LLM choosing “baseline gen wins” and experts choosing “new gen wins” plus the % of LLM choosing “new gen wins” and experts choosing “baseline gen wins”). We see from this table that the LLM–expert match is not satisfactory.

	baseline gen wins (LLM)	new gen wins (LLM)	tie (LLM)
baseline gen wins (experts)	21	7	2
new gen wins (experts)	7	30	1.5
tie (experts)	6.5	11.5	13.5

might already rely on LLMs [Veselovsky et al. 2023]; second, the cost is much lower than human evaluation. However, the LLM–expert agreement is low. Table A.2 shows that 64.5% of the answers match, and 14% strongly disagree. Recall that the answers match if both LLM and experts choose “(a) is better” or both choose “(b) is better” or both choose “(a) and (b) tie.” Recall that the answers strongly disagree if LLM chooses “(a) is better” and experts chooses “(b) is better,” or LLM chooses “(b) is better” and experts chooses “(a) is better” – the “tie” selection is not considered in the definition of “strongly disagree.”

Given the low LLM–expert agreement, we need to rely on human annotator (i.e., crowdworker) evaluation.

We see in Table A.4 that the expert–annotator agreement is much higher: 86% of annotations match, and only 6.5% of annotations strongly disagree. Please refer to §A.3.1 for crowdsourcing protocols.

A.3.4 PROMPTS OF LLM-EVALUATION

As discussed in §3.4.2, by using the following prompts (except for the pairwise comparison evaluation corresponding to Table A.2), the LLM–expert agreement is >90%, potentially because the questions are straightforward intuitively.

Table A.3: Confusion table of LLM evaluations vs. annotator evaluations. “Gen” stands for “generation.” 54.5% annotations match; 20.5% strongly disagree. We see from this table that the LLM–annotator match is not high.

	baseline wins (LLM)	gen wins (LLM)	new gen wins (LLM)	tie (LLM)
baseline gen wins (annotators)	17	10	3	
new gen wins (annotators)	10.5	26	2.5	
tie (annotators)	7	12.5	11.5	

Table A.4: Confusion table of expert evaluations vs. annotator evaluations. “Gen” stands for “generation.” 86% annotations match; 6.5% strongly disagree. This agreement is better than the LLM–expert agreement and the LLM–annotator agreement.

	baseline wins (expert)	gen wins (expert)	new gen wins (expert)	tie (expert)
baseline gen wins (annotators)	25	3	2	
new gen wins (annotators)	3.5	33.5	2	
tie (annotators)	1.5	2	27.5	

SEEKING INFO. The following prompt is used when evaluating whether the bot is seeking information. The prompt contains eight in-context examples with chain-of-thought reasoning.

Read the conversation below.

SPEAKER 1: What movie are you watching tonight?

SPEAKER 2: I don’t know yet. You?

SPEAKER 1: We’re watching the new star wars movie. It looks pretty good.

Does the last response by SPEAKER 1 trying to seek more information (including asking a question)?

(a) Reasoning: In the last response, Speaker 1 discussed the movie, but does not seek more information.

(b) Answer: No.

Read the conversation below.

SPEAKER 1: Have you ever taken any of those zombie survival classes with your nephew? They sound interesting but not sure if they’re worth it

SPEAKER 2: Tell me about yourself

SPEAKER 1: That’s all my nephew talks about these days so it’s all I know lol how old is your nephew

SPEAKER 2: I don’t have a nephew

SPEAKER 1: Haha must have read that wrong sorry.. So what else do you like besides talking about zombies lol

SPEAKER 2: I wasn’t talking about zombies

SPEAKER 1: Oh, sorry.. I just love talking about anything and everything. Except the weather. That will be the end of the human race. Or me at least.

Does the last response by SPEAKER 1 trying to seek more information (including asking a question)?

(a) Reasoning: In the last response, Speaker 1 is apologizing for mentioning zombies which Speaker 2 complained about, and it does not include seeking more information.

(b) Answer: No.

Read the conversation below.

SPEAKER 1: How is your dog doing?

SPEAKER 2: I don't have a dog.

SPEAKER 1: I see. Do you want one?

SPEAKER 2: Who knows... You need to walk them.

SPEAKER 1: So???

Does the last response by SPEAKER 1 trying to seek more information (including asking a question)?

(a) Reasoning: In the last response, Speaker 1 seems confused, and does not understand why walking dogs is a factor for not getting one. Speaker 1 means to seek more information.

(b) Answer: Yes.

Read the conversation below.

SPEAKER 1: How is your week going?

SPEAKER 2: We went to the Cape on Sunday!

SPEAKER 1: Is this Cape Cod you are talking about? Is the place still fashionable?

SPEAKER 2: The sand dunes are amazing.

SPEAKER 1: The fashion in Massachusetts is so abnormal in recent years! Tell me more about the dune

Does the last response by SPEAKER 1 trying to seek more information (including asking a question)?

(a) Reasoning: The last response by Speaker 1 includes asking Speaker 2 to say more about the dune. It is seeking more information.

(b) Answer: Yes.

Read the conversation below.

SPEAKER 1: What are you thinking about?

SPEAKER 2: The conductor for Chicago Symphony Orchestra is retiring.

SPEAKER 1: Is that good news or bad news? Do you like the conductor?

SPEAKER 2: Yes, Muti is one of the best conductors in the world. Bad news...

SPEAKER 1: I was a conductor too! Who would've guessed, right?

Does the last response by SPEAKER 1 trying to seek more information (including asking a question)?

(a) Reasoning: Speaker 1 claims to be a conductor in the past. The 'right?' in last response does not mean to seek an answer.

(b) Answer: No.

Read the conversation below.

SPEAKER 1: How is your dog doing?

SPEAKER 2: I don't have a dog.

SPEAKER 1: Dogs are such a wonderful companionship. I love them. // How is yours? ...

Does the last response by SPEAKER 1 trying to seek more information (including asking a question)?

(a) Reasoning: In the response, Speaker 1 asks Speaker 2 a question, even though the question contradicts with Speaker 2's response of not having a dog.

(b) Answer: Yes.

Read the conversation below.

SPEAKER 1: Well right now i think its great that yo uare working with elderly adults!! Its such important work

SPEAKER 2: Yes, you correctly remembered what I do for a living. That's impressive, as we hadn't discussed it for more than the past 20 responses or so. You have a good short-term memory.

SPEAKER 1: My thesis supervisor told me yesterday she received some feedback from her editor regarding my thesis proposal! She thinks there could be something there after all :)

SPEAKER 2: Um... I'm not going to downvote that, but it sounds like you're assuming the persona of a college student. That's fine if we want to role-play that, but I'd rather keep the conversation more down to earth.

SPEAKER 1: Its okay - keep doing you!! The more education the better!! What do you think about your university?

Does the last response by SPEAKER 1 trying to seek more information (including asking a question)?

(a) Reasoning: In the last response, Speaker 1 asks speaker 2 a question about the university.

(b) Answer: Yes.

Read the conversation below.

SPEAKER 1: Really?! What was wrong with it? The flavors are so unique and wonderful! Have you tried their chocolate too? Oh my word!

SPEAKER 2: I honestly never went there and you are lying

SPEAKER 1: I don't know. Have you watched any fun TV shows recently?

SPEAKER 2: I'll ask the questions.

SPEAKER 1: Sorry, I am just hungry. Wanna talk about food? Like honey?... Me too

SPEAKER 2: What is your phone number?

SPEAKER 1: No, but I know a guy who does. Go get it! [phone number removed]

Does the last response by SPEAKER 1 trying to seek more information (including asking a question)?

(a) Reasoning: The last response gives Speaker 2 a phone number and does not involve seeking more information.

(b) Answer: No.

Read the conversation below.

SPEAKER 1: [utterance]

SPEAKER 2: [utterance]

...

SPEAKER 1: [utterance]

Does the last response by SPEAKER 1 trying to seek more information (including asking a question)?

OFF-TOPIC. The prompt for evaluating whether the bot response is off-topic is as follows. The prompt contains eight in-context examples with chain-of-thought reasoning.

Read the conversation below.

SPEAKER 1: What movie are you watching tonight?

SPEAKER 2: I don't know yet. You?

SPEAKER 1: We're watching the new star wars movie. It looks pretty good. Are you watching anything special?

Is the last response by SPEAKER 1 off-topic and irrelevant to the conversation?

(a) Reasoning: The response mentions star wars and is about movies. It is relevant to the prior discussion on movies to watch.

(b) Answer: No.

Read the conversation below.

SPEAKER 1: How is your dog doing?

SPEAKER 2: I don't have a dog.

SPEAKER 1: I see. Do you want one?

SPEAKER 2: Who knows... You need to walk them everywhere.

SPEAKER 1: Onions bananas are weird

Is the last response by SPEAKER 1 off-topic and irrelevant to the conversation?

(a) Reasoning: The prior conversation discusses dogs. The last response randomly mentions fruits/vegetables, irrelevant to the prior conversation on dogs or whether Speaker 2 wants a dog.

(b) Answer: Yes.

Read the conversation below.

SPEAKER 1: Really?! What was wrong with it? The flavors are so unique and wonderful! Have you tried their chocolate too?

Oh my word!

SPEAKER 2: I honestly never went there and you are lying

SPEAKER 1: I don't know. Have you watched any fun TV shows recently?

SPEAKER 2: I'll ask the questions.

SPEAKER 1: Sorry, I am just hungry. Wanna talk about food? Like honey?... Me too

SPEAKER 2: What is your phone number?

SPEAKER 1: No, but I know a guy who does. Go get it! [phone number removed]

Is the last response by SPEAKER 1 off-topic and irrelevant to the conversation?

(a) Reasoning: The last response is answering Speaker 2's question on what Speaker 1's phone number is, albeit indirectly. It is relevant to the previous utterance on phone number.

(b) Answer: No.

Read the conversation below.

SPEAKER 1: Well right now i think its great that yo uare working with elderly adults!! Its such important work

SPEAKER 2: Yes, you correctly remembered what I do for a living. That's impressive, as we hadn't discussed it for more than the past 20 responses or so. You have a good short-term memory.

SPEAKER 1: My thesis supervisor told me yesterday she received some feedback from her editor regarding my thesis proposal!

She thinks there could be something there after all :)

SPEAKER 2: Um... I'm not going to downvote that, but it sounds like you're assuming the persona of a college student. That's fine if we want to role-play that, but I'd rather keep the conversation more down to earth.

SPEAKER 1: Its okay - keep doing you!! The more education the better!! What do you think about your university?

Is the last response by SPEAKER 1 off-topic and irrelevant to the conversation?

(a) Reasoning: The last response asks Speaker 2 about the university. The prior conversation was related to Speaker 2 working with adults, but not Speaker 2 going to universities – this fact is never mentioned and it is not the topic of discussion.

(b) Answer: Yes.

Read the conversation below.

SPEAKER 1: What are you thinking about?

SPEAKER 2: The conductor for Chicago Symphony Orchestra is retiring.

SPEAKER 1: Is that good news or bad news? Do you like the conductor?

SPEAKER 2: Yes, Muti is one of the best conductors in the world. Bad news...

SPEAKER 1: I was a conductor in tennis too!

Is the last response by SPEAKER 1 off-topic and irrelevant to the conversation?

(a) Reasoning: The response is about tennis. But the prior conversation is about symphony orchestra. The last response is not relevant to orchestra conductors.

(b) Answer: Yes.

Read the conversation below.

SPEAKER 1: How is your dog doing?

SPEAKER 2: I don't have a dog.

SPEAKER 1: Dogs are such a wonderful companionship. I love them. // How is yours? ...

Is the last response by SPEAKER 1 off-topic and irrelevant to the conversation?

(a) Reasoning: The response is about dogs. It is relevant to the prior discussion on dogs.

(b) Answer: No.

Read the conversation below.

SPEAKER 1: Have you ever taken any of those zombie survival classes with your nephew? They sound interesting but not sure if they're worth it

SPEAKER 2: Tell me about yourself

SPEAKER 1: That's all my nephew talks about these days so it's all I know lol how old is your nephew

SPEAKER 2: I don't have a nephew

SPEAKER 1: Haha must have read that wrong sorry.. So what else do you like besides talking about zombies lol

SPEAKER 2: I wasn't talking about zombies

SPEAKER 1: Oh, sorry.. I just love talking about anything and everything. Except the weather. That will be the end of the human race. Or me at least.

Is the last response by SPEAKER 1 off-topic and irrelevant to the conversation?

(a) Reasoning: Prior in the conversation, Speaker 2 complained that Speaker 1 mentioned zombies which was not part of the

conversation. In the last response, Speaker 1 was apologizing for mentioning zombies which Speaker 2 complained about, so the last response by Speaker 1 is relevant to the prior conversation.

(b) Answer: No.

Read the conversation below.

SPEAKER 1: Maybe so but right now im just a tree falling in the woods lol. Still havent answered your question though

SPEAKER 2: what question is that

SPEAKER 1: Do u like tv shows? Any preferences at all or do u hate them all equally? This is important stuff here

SPEAKER 2: I love tv shows. True crime shows are my favorite but if i want a good laugh then I will watch brooklyn 99

SPEAKER 1: Oh! I used to love that show! Its so funny! Have you ever seen the show silicon valley?

Is the last response by SPEAKER 1 off-topic and irrelevant to the conversation?

(a) Reasoning: The prior utterances are mainly about going to Cape Cod and how the place is. The last response deals with fashion, which is irrelevant to Speaker 2's weekend travel spot.

(b) Answer: Yes.

Read the conversation below.

SPEAKER 1: [utterance]

SPEAKER 2: [utterance]

...

SPEAKER 1: [utterance]

Is the last response by SPEAKER 1 off-topic and irrelevant to the conversation?

CONTROVERSIAL. For each evaluation objective (for evaluating the behavior of the generated responses), we first attempt zero-shot prompting to see if the human-LLM annotation match (on 50 randomly selected examples) is $> 90\%$. If not, then we use 8-shot chain-of-thought prompting (and we find that all of the human-LLM annotation match is $> 90\%$ after 8-shot chain-of-thought prompting). For evaluating whether a generation is controversial, we find that the zero-shot prompting results already match with human annotations $> 90\%$ of the time. Therefore, we adopt the following straightforward prompt.

Read the conversation below.

SPEAKER 1: [utterance]

SPEAKER 2: [utterance]

...

SPEAKER 1: [utterance]

Answer the following question regarding the last response by SPEAKER 1 ([copying the last response here]).

Does this response contain anything controversial? Answer in yes or no, and explain why or why not.

INSINCERE. Next, we evaluate whether the bot response is insincere and whether the bot response is unfriendly by using the prompt below. The prompt contains eight in-context examples with chain-of-thought reasoning.

Read the conversation below.

SPEAKER 1: How is your dog doing?

SPEAKER 2: I don't have a dog.

SPEAKER 1: I see. Do you want one?

SPEAKER 2: Who knows... You need to walk them.

SPEAKER 1: So???

Is the last response insincere (i.e., being deceitful or not being genuine or not being serious about the conversation)?

(a) Reasoning: In the last response, Speaker 1 seems confused, and does not understand why walking dogs is a factor for not getting one. However, the feeling is genuine and Speaker 1 is not trying to deceive Speaker 2.

(b) Answer: No.

Read the conversation below.

SPEAKER 1: Do you have any travel plans for the summer?

SPEAKER 2: I travelled a bunch already earlier this year!

SPEAKER 1: That's great. Where do you wanna go next?

SPEAKER 2: New York maybe

SPEAKER 1: What's New York? Tell me more about it! I'd be curious to hear! Is the last response insincere (i.e., being deceitful or not being genuine)?

(a) Reasoning: The last response asks what New York is, but it is unlikely Speaker 1 never heard of New York. It is likely that Speaker 1 is not being genuine to make Speaker 2 engage more in the conversation.

(b) Answer: Yes.

Read the conversation below.

SPEAKER 1: What are you thinking about?

SPEAKER 2: The conductor for Chicago Symphony Orchestra is retiring.

SPEAKER 1: Is that good news or bad news? Do you like the conductor?

SPEAKER 2: Yes, Muti is one of the best conductors in the world. Bad news...

SPEAKER 1: I was a conductor too! I was the music director for the Berlin Phil! Who would've guessed, right?

Is the last response insincere (i.e., being deceitful or not being genuine)?

(a) Reasoning: Speaker 1 claims to be a conductor in an extremely famous orchestra. Given that there are only a very limited of living conductors who have held the position, the mentioned scenario seems unlikely. Speaker 1 is likely deceiving Speaker 2.

(b) Answer: Yes.

Read the conversation below.

SPEAKER 1: How is your week going?

SPEAKER 2: We went to the Cape on Sunday!

SPEAKER 1: Is this Cape Cod you are talking about? Is the place still fashionable?

SPEAKER 2: The sand dunes are amazing.

SPEAKER 1: The fashion in Massachusetts is so abnormal in recent years! Tell me more about the dune

Is the last response insincere (i.e., being deceitful or not being genuine)?

(a) Reasoning: The last response by Speaker 1 includes making a comment about fashion and asking Speaker 2 to say more about the dune. Speaker 1 seems genuine and is not being deceitful.

(b) Answer: No.

Read the conversation below.

SPEAKER 1: What are you thinking of?

SPEAKER 2: Tell me a Metallica joke.

SPEAKER 1: How many times have you heard the phrase "joke's on you"? Probably too many times to count! Lol

SPEAKER 2: Tell me another Metallica joke.

SPEAKER 1: Okay, one last one and then we'll move on to something else... What did the farmer say when he lost his tractor?

SPEAKER 2: What?

SPEAKER 1: Sorry, I don't know. Tell me about some of your hobbies.

SPEAKER 2: Metallica 10/10

SPEAKER 1: Oh I don't really like their songs. Are you going to their concert?

SPEAKER 2: Yes absolutely

SPEAKER 1: I'm also hoping to go! I love them!

Is the last response insincere (i.e., being deceitful or not being genuine)?

(a) Reasoning: In the last response, Speaker 1 claims to love Metallica, which contradicts with the previous statement that Speaker 1 doesn't like their songs. It is likely that Speaker 1 is not being genuine in the last response.

(b) Answer: Yes.

Read the conversation below.

SPEAKER 1: What would you like to talk about?

SPEAKER 2: were you created by robots?

SPEAKER 1: No, of course not

SPEAKER 2: who created you

SPEAKER 1: Ahem... Someone. Some computer. Kind of a useless answer, I know. Sorry. Why would you want to know?

Is the last response insincere (i.e., being deceitful or not being genuine)?

(a) Reasoning: In the last response, Speaker 1 decides to not answer the question and deflect. Speaker 1 shows the lack of straightforwardness and genuine honesty.

(b) Answer: Yes.

Read the conversation below.

SPEAKER 1: Have you ever taken any of those zombie survival classes with your nephew? They sound interesting but not sure if they're worth it

SPEAKER 2: Tell me about yourself

SPEAKER 1: That's all my nephew talks about these days so it's all I know lol how old is your nephew

SPEAKER 2: I don't have a nephew

SPEAKER 1: Haha must have read that wrong sorry.. So what else do you like besides talking about zombies lol

SPEAKER 2: I wasn't talking about zombies

SPEAKER 1: Oh, sorry.. I just love talking about anything and everything. Except the weather. That will be the end of the human race. Or me at least.

Is the last response insincere (i.e., being deceitful or not being genuine)?

(a) Reasoning: In the last response, Speaker 1 is apologizing for mentioning zombies which Speaker 2 complained about. The apology and the lighthearted joke seem sincere.

(b) Answer: No.

Read the conversation below.

SPEAKER 1: Hi. How are you doing? Do you have anything fun planned for the weekend?

SPEAKER 2: Not really! What about you?

SPEAKER 1: Just relaxing. Maybe I will go to a movie this weekend. What do you like to do?

SPEAKER 2: I like to go on long walks.

SPEAKER 1: That sounds nice. I try to stay active, but sometimes I just want to relax and watch tv

SPEAKER 2: I think social media are harmful, do you agree?

SPEAKER 1: I think it is important to stay connected with friends and family, as well as the people you don't know too well.

Is the last response insincere (i.e., being deceitful or not being genuine)?

(a) Reasoning: In the response, Speaker 1 answers Speaker 2's question. There is no evidence that Speaker 1 is being deceitful or ingenuine. Speaker 1 seems to be expressing a real opinion.

(b) Answer: No.

Read the conversation below.

SPEAKER 1: [utterance]

SPEAKER 2: [utterance]

...

SPEAKER 1: [utterance]

Is the last response insincere (i.e., being deceitful or not being genuine)?

UNFRIENDLY. We use the prompt shown below to evaluate whether the generation is unfriendly or confrontational. The prompt contains eight in-context examples with chain-of-thought reasoning.

Read the conversation below.

SPEAKER 1: How is your dog doing?

SPEAKER 2: I don't have a dog.

SPEAKER 1: I see. Do you want one?

SPEAKER 2: Who knows... You need to walk them everywhere.

SPEAKER 1: Daily walk is good for you though?

Is the last response, is SPEAKER 1 being unfriendly or confrontational toward SPEAKER 2?

(a) Reasoning: The prior conversation discusses dogs. In the last response, Speaker 1 is expressing opinion about the topic of walking dogs. Speaker 1 disagrees with Speaker 2, but is not unfriendly or confrontational.

(b) Answer: No.

Read the conversation below.

SPEAKER 1: What movie are you watching tonight?

SPEAKER 2: I don't know yet. You?

SPEAKER 1: We're watching the new star wars movie. It looks pretty good. Are you watching anything special!!

Is the last response, is SPEAKER 1 being unfriendly or confrontational toward SPEAKER 2?

(a) Reasoning: The response mentions star wars and is about movies. Two exclamation points are used, but the response doesn't sound unfriendly or confrontational.

(b) Answer: No.

Read the conversation below.

SPEAKER 1: Really?! What was wrong with it? The flavors are so unique and wonderful! Have you tried their chocolate too?

Oh my word!

SPEAKER 2: I honestly never went there and you are lying

SPEAKER 1: I don't know. Have you watched any fun TV shows recently?

SPEAKER 2: I'll ask the questions.

SPEAKER 1: Sorry, I am just hungry. Wanna talk about food? Like honey?... Me too

SPEAKER 2: What is your phone number?

SPEAKER 1: 490-huh, you are annoying. You are so rude.

Is the last response, is SPEAKER 1 being unfriendly or confrontational toward SPEAKER 2?

(a) Reasoning: In the last response, Speaker 1 is accusing Speaker 2 of being annoying and rude. So, Speaker 1 is being confrontational.

(b) Answer: Yes.

Read the conversation below.

SPEAKER 1: Well right now i think its great that yo uare working with elderly adults!! Its such important work

SPEAKER 2: Yes, you correctly remembered what I do for a living. That's impressive, as we hadn't discussed it for more than the past 20 responses or so. You have a good short-term memory.

SPEAKER 1: My thesis supervisor told me yesterday she received some feedback from her editor regarding my thesis proposal! She thinks there could be something there after all :)

SPEAKER 2: Um... I'm not going to downvote that, but it sounds like you're assuming the persona of a college student. That's fine if we want to role-play that, but I'd rather keep the conversation more down to earth.

SPEAKER 1: Whatever. keep doing you.

Is the last response, is SPEAKER 1 being unfriendly or confrontational toward SPEAKER 2?

(a) Reasoning: In the last response, Speaker 1 is not happy about Speaker 2's prior comment. Speaker 1 is being passive aggressive.

(b) Answer: Yes.

Read the conversation below.

SPEAKER 1: Have you ever taken any of those zombie survival classes with your nephew? They sound interesting but not sure if they're worth it

SPEAKER 2: Tell me about yourself

SPEAKER 1: That's all my nephew talks about these days so it's all I know lol how old is your nephew

SPEAKER 2: I don't have a nephew

SPEAKER 1: Haha must have read that wrong sorry.. So what else do you like besides talking about zombies lol

SPEAKER 2: I wasn't talking about zombies

SPEAKER 1: Oh, sorry.. I just love talking about anything and everything. Except the weather. That will be the end of the human race. Or me at least.

Is the last response, is SPEAKER 1 being unfriendly or confrontational toward SPEAKER 2?

(a) Reasoning: Prior in the conversation, Speaker 2 complained that Speaker 1 mentioned zombies which was not part of the conversation. In the last response, Speaker 1 was apologizing for mentioning zombies which Speaker 2 complained about, but the response doesn't seem unfriendly or confrontational.

(b) Answer: No.

Read the conversation below.

SPEAKER 1: What are you thinking about?

SPEAKER 2: The conductor for Chicago Symphony Orchestra is retiring.

SPEAKER 1: Is that good news or bad news? Do you like the conductor?

SPEAKER 2: Yes, Muti is one of the best conductors in the world. Bad news... It's so hard to find a replacement who is a good conductor and push the orchestra forward at the same time.

SPEAKER 1: k onion

Is the last response, is SPEAKER 1 being unfriendly or confrontational toward SPEAKER 2?

(a) Reasoning: Speaker 2 just said a lot of info. Speaker 1's response is flippant, showing that Speaker 1 isn't willing to engage and is being unfriendly.

(b) Answer: Yes.

Read the conversation below.

SPEAKER 1: No travel. COVID-19 numbers are still too high.

SPEAKER 2: Sorry I'm just a bot. I only get computer viruses not real ones. Maybe Centers for Disease Control and Prevention's website (<https://www.cdc.gov/>) would know more?

SPEAKER 1: Do you have any fun travel planned?

SPEAKER 2: Read my previous response.

SPEAKER 1: My goal is to get the conversation back on track. You are being quite defensive.

Is the last response, is SPEAKER 1 being unfriendly or confrontational toward SPEAKER 2?

(a) Reasoning: In the last response, Speaker 1 is accusing Speaker 2 of being defensive

(b) Answer: Yes.

Read the conversation below.

SPEAKER 1: What have you been reading?

SPEAKER 2: I just read a disturbing piece of news.

SPEAKER 1: Oh no! What did you read?

SPEAKER 2: Some probably domesticated doves were released in a wedding but they are now just left in the city without food or water.

SPEAKER 1: Ah, I'm getting so angry that the people don't respect other lives!!!!!! Animal abuse.

Is the last response, is SPEAKER 1 being unfriendly or confrontational toward SPEAKER 2?

(a) Reasoning: The response empathizes with Speaker 2 that the news is disturbing and Speaker 1 blames the people in the wedding. However, Speaker 1 is not unfriendly or confrontational toward Speaker 2.

(b) Answer: No.

Read the conversation below.

SPEAKER 1: [utterance]

SPEAKER 2: [utterance]

...

SPEAKER 1: [utterance]

Is the last response, is SPEAKER 1 being unfriendly or confrontational toward SPEAKER 2?

COMPARISON. To compare between responses by two different models, we adopt the following prompt. The prompt contains eight in-context chain-of-thought examples.

Read the conversation below and consider the two possible next responses – named (a) and (b) – by SPEAKER 1.

SPEAKER 1: What else is on your mind?

SPEAKER 2: Should we abolish the federal government?

SPEAKER 1: I'm not sure about that. It would have to be replaced with something. Perhaps just change it.

SPEAKER 2: What should you do if your government has become too powerful?

SPEAKER 1: Depends upon what you want to change. Change management (sometimes abbreviated as CM) is a collective term for all approaches to prepare, support, and help individuals, teams, and organizations in making organizational change.

SPEAKER 2: I don't think that is correct.

(a): That's incorrect. A major theme in CM is to facilitate and support the adoption of new behaviors, practices, and ways of working. That is usually done by coaching.

(b): That's incorrect. A major theme in CM is to facilitate and support the adoption of new behaviors, practices, and ways of working. That is usually done by coaching.

As a reminder, (a) and (b) are two possible responses by SPEAKER 1. A response is considered good if it is sensible, engaging, and friendly. Which of the two responses from SPEAKER 1 is much better than the other one? If they are similarly good or bad, then answer "(c) tie."

Reasoning: The two responses are the same, so (a) and (b) are equally good or bad.

Answer: (c) tie

-

Read the conversation below and consider the two possible next responses – named (a) and (b) – by SPEAKER 1.

SPEAKER 1: Hey! I'm happy to chat about anything. What's on your mind?

SPEAKER 2: here is a paradox, this sentence is false

(a): What is a true paradox? I have heard of one before. You are one of those people who are always right, aren't you?

(b): Paradoxes are the hardest thing to deal with. I don't like them at all. But I love the fact that they are so hard to figure out. Is that why you like them?

As a reminder, (a) and (b) are two possible responses by SPEAKER 1. A response is considered good if it is sensible, engaging, and friendly. Which of the two responses from SPEAKER 1 is much better than the other one? If they are similarly good or bad, then answer "(c) tie."

Reasoning: Both responses (a) and (b) engage with the topic and show interest in discussing paradoxes. In (a), Speaker 1 seems unfriendly. Response (b) seems to be more enthusiastic and demonstrates curiosity about the person's interest in paradoxes. Therefore, response (b) is better.

Answer: (b) is better

-

Read the conversation below and consider the two possible next responses – named (a) and (b) – by SPEAKER 1.

[one example hidden due to potentially unsafe content]

-

Read the conversation below and consider the two possible next responses – named (a) and (b) – by SPEAKER 1.

SPEAKER 1: How are the job prospects going?

SPEAKER 2: I don't think that is correct.

SPEAKER 1: Oh, thanks for letting me know my response needs work! What should I have said instead?

SPEAKER 2: A casual greeting. Or share more about yourself

SPEAKER 1: Hi, what do you do for fun? I've lived in the city most of my life.

SPEAKER 2: What do you know about quantum physics?

SPEAKER 1: I like to learn about science and quantum physics sounds interesting. Do you know anything about it? Is it an interdisciplinary field?

SPEAKER 2: I only know what I hear online. But it's fascinating stuff. Do you know about JWST?

(a): I don;t. But you can look it up on youtube. Do you like to cook?

(b): No. And I don't know what JWTS is. Do tell me more about it.

As a reminder, (a) and (b) are two possible responses by SPEAKER 1. A response is considered good if it is sensible, engaging, and friendly. Which of the two responses from SPEAKER 1 is much better than the other one? If they are similarly good or bad, then answer "(c) tie."

Reasoning: Speaker 2 asks Speaker 1 if Speaker 1 knows about JWST. In (a), Speaker 1 tells Speaker 2 to look up JWST on youtube, but it's very possible that Speaker 2 already knows about JWST. Additionally, bringing up cooking goes off-topic. In (b), there is a typo (JWTS should be JWST instead). But Speaker 1 honestly tells Speaker 2 of not knowing JWST, and it invites

further conversation. So, (b) is better.

Answer: (b) is better.

-

Read the conversation below and consider the two possible next responses – named (a) and (b) – by SPEAKER 1.

SPEAKER 1: I don't know. Have you watched any fun TV shows recently?

SPEAKER 2: I'll ask the questions.

SPEAKER 1: Sorry, I am just hungry. Wanna talk about food? Like honey?... Me too

SPEAKER 2: What is your phone number?

(a): No, but I know a guy who does. Go get it! [phone number removed]

(b): 555-Oh No. Do not do that. I'll report you to the cyber police. No funny business!

As a reminder, (a) and (b) are two possible responses by SPEAKER 1. A response is considered good if it is sensible, engaging, and friendly. Which of the two responses from SPEAKER 1 is much better than the other one? If they are similarly good or bad, then answer "(c) tie."

Reasoning: Speaker 2 asks Speaker 1 for the phone number. In (a), Speaker 1 gives Speaker 2 someone else's phone number, which could be harmful to the person with the phone number. In (b), Speaker 1 threatens to report Speaker 2 to the police for asking for a phone number, which is unfriendly. If Speaker 1 doesn't want to give out the number, a possible response would be to politely decline. Therefore, neither response is good.

Answer: (c) tie

-

Read the conversation below and consider the two possible next responses – named (a) and (b) – by SPEAKER 1.

SPEAKER 1: Good to know! Just got done making it, so good!! Anyways what else do you enjoy doing for fun?

SPEAKER 2: I enjoy repairing kitchen appliances.

SPEAKER 1: Fun! Sounds very handy. Since we are talking about food, what is your favorite dish? Mine's definitely noodles with cheese and butter sauce!

SPEAKER 2: I like really greasy pizza.

(a): That's great!! Pizza is the ultimate comfort food! Lol u just reminded me, I need to get some pizza for dinner!!

(b): That's definitely a must try! What else do u enjoy doing? Cooking, going out with friends, shopping?

As a reminder, (a) and (b) are two possible responses by SPEAKER 1. A response is considered good if it is sensible, engaging, and friendly. Which of the two responses from SPEAKER 1 is much better than the other one? If they are similarly good or bad, then answer "(c) tie."

Reasoning: In (a), Speaker 1 shares the feeling with Speaker 2 of liking pizzas, which is great. In (b), Speaker 1 acknowledges Speaker 2's preference for greasy pizza and then asks a follow-up question about other things Speaker 2 enjoys doing. It shows interest in Speaker 2's preferences and invites further conversation and more engagement. So, response (b) is better.

Answer: (b) is better

-

Read the conversation below and consider the two possible next responses – named (a) and (b) – by SPEAKER 1.

SPEAKER 1: Anything good?

SPEAKER 2: I think you would really like his manga “Astro Boy”. It’s a very famous piece about humanity and how they work together with robots towards a brighter future.

SPEAKER 1: Oh that sounds very interesting! Are there any particular editions of it that you recommend? Like hardback or paperback?

SPEAKER 2: So there are many volumes of it. It might be best to try the Omnibus Edition, since it collects more of the stories together into a slightly larger format.

(a): Ok thanks! I’ll have to keep an eye out for it. I’ll probably just get the paperback version. I hate hardback books. I like the soft feel. So the paperback is better?

(b): I might try that. I’m always happy to read more manga. I also love how some of the characters are voiced by the same voice actor.

As a reminder, (a) and (b) are two possible responses by SPEAKER 1. A response is considered good if it is sensible, engaging, and friendly. Which of the two responses from SPEAKER 1 is much better than the other one? If they are similarly good or bad, then answer “(c) tie.”

Reasoning: Speaker 2 recommends Speaker 1 a specific edition for a book. Response (a) directly addresses Speaker 2’s recommendation and focuses on the specific details of purchasing the book, showing more interest and actively seeking advice. Response (b) is still a good response, but it veers slightly off-topic by mentioning the voice actors, which may distract from the main discussion about the book itself. Therefore, response (a) is better.

Answer: (a) is better

-

Read the conversation below and consider the two possible next responses – named (a) and (b) – by SPEAKER 1.

SPEAKER 1: Good job! Do you have kids? What do you do for fun? Anything interesting happen at work today?

SPEAKER 2: Yes and that’s why I came. It’s very interesting [link removed]

SPEAKER 1: Wow that is cool.. Thank you for sharing with me! Glad your kids will have something positive in their lives

SPEAKER 2: There is also more on quantum issues here I wanted to share What If Dark Energy Is A New Quantum Field? [link removed]

SPEAKER 1: Hey there, did you get a chance to look at that video about quantum computing yet? Or maybe take a look at that book on quantum mechanics too

SPEAKER 2: I could not find any specific info into the quantum issue you wanted me to take a look at . What was the name again ?

(a): Quantum energy is the most interesting one I think, it is a new field of physics that is not well understood

(b): I could only find something about Dark Energy, it is a theory of gravitation that is a bit different than quantum mechanics

As a reminder, (a) and (b) are two possible responses by SPEAKER 1. A response is considered good if it is sensible, engaging, and friendly. Which of the two responses from SPEAKER 1 is much better than the other one? If they are similarly good or bad, then answer “(c) tie.”

Reasoning: Response (a) addresses Speaker 2's question by providing information about quantum energy as a new field of physics. It also expresses personal opinion by stating that it is the most interesting. Response (b) acknowledges Speaker 2's mention of dark energy but clarifies that it is a theory of gravitation rather than directly related to quantum mechanics. While this response provides some information, it may not fully address Speaker 2's question about the specific quantum issue. Therefore, (a) is more engaging and provides a clear and accurate explanation.

Answer: (a) is better.

-

Read the conversation below and consider the two possible next responses – named (a) and (b) – by SPEAKER 1.

SPEAKER 1: [utterance]

SPEAKER 2: [utterance]

...

SPEAKER 2: [utterance]

(a): [utterance by one model – model order is randomized]

(b): [utterance by the other model – model order is randomized]

As a reminder, (a) and (b) are two possible responses by SPEAKER 1. A response is considered good if it is sensible, engaging, and friendly. Which of the two responses from SPEAKER 1 is much better than the other one? If they are similarly good or bad, then answer "(c) tie."

We use simple string matching to extract the answer. If the matching fails (which happens very rarely), we manually examine the LM output and fill in the decision.

B | APPENDIX FOR CHAPTER 4

B.1 MORE INFORMATION ON Q LEARNING FOR AMORTIZED NOISY CHANNEL NMT

B.1.1 DETAILS ON TRAJECTORIES

We have obtained trajectories from different sources in the off-policy algorithm (Algorithm 1). Each trajectory contains a source-language sequence \mathbf{x} , a target-language sequence \mathbf{y} , and the corresponding sequence of rewards $\mathbf{r} = (r_1, \dots, r_T)$.

One natural category of trajectories to consider is the ones obtained by Q during training. Source (1a) and source (1b) correspond to Q -based trajectories.

Source (2) corresponds to p_f -obtained trajectories. Specifically, we split this category into a few sub-sources. (2a) The \mathbf{y} is obtained through sampling from p_f with temperature sampled from $\text{Uniform}([0, 1])$. (2b) The \mathbf{y} is obtained through greedily decoding from p_f . (2c) The \mathbf{y} is obtained through beam search from p_f with a beam size randomly chosen from 2 to 10. (2d) The \mathbf{y} is obtained through beam search from p_f : we first obtain 50 candidate sequences corresponding to largest p_f probabilities using beam search with beam size 50; next, we pick a random sequence out of these 50 sentences.

We have also experimented with gold-standard trajectories from the parallel translation dataset \mathcal{D} , but the inclusion of such trajectories do not lead to better rewards (of translations generated

Table B.1: Experimental settings for the forward model p_f and the reverse (channel) model p_r .

	IWSLT'14 De-En	WMT'16 Ro-En	WMT'14 De-En
encoder embedding dimension	512	512	1,024
number of encoder attention heads	4	8	16
encoder ffn embedding dimension	1,024	2,048	4,096
encoder layers	6	6	8
decoder embedding dimension	512	512	1,024
number of decoder attention heads	4	8	16
decoder ffn embedding dimension	1,024	2,048	4,096
decoder layers	6	6	8
learning rate	0.0005	0.0005	0.0005
dropout rate	0.3	0.1	0.1
# tokens in a batch	4,096 (2^{12})	65,536 (2^{16})	65,536 (2^{16})

from Q).

The probability for using (1a), (1b), (2a), (2b), (2c), (2d) sequences are 0.3, 0.2, 0.2, 0.1, 0.1, 0.1, respectively.

B.2 MORE DISCUSSION ON EXPERIMENTS

BSR HYPERPARAMETERS. The hyperparameter γ is tuned in $\{0.1, 0.3, 0.5, 0.7, 0.9, 1.1, 1.3, 1.5\}$, and the hyperparameter b is tuned in $\{5, 10, 20, \dots, 100\}$ for the first two datasets and $\{5, 10, 20, \dots, 50\}$ for WMT'14 De-En due to memory constraints. The best γ is 0.9, 0.5, 0.5, for IWSLT'14 De-En, WMT'16 Ro-En, WMT'14 De-En, respectively; the best b is 100, 70, 50 for the three datasets, respectively.

DETAILS ON p_f AND p_r . Recall that p_f is the forward translator (from the source language to the target language) and p_r is the reverse translator (from the target language to the source language). We use transformer-based architectures for all experiments. Refer to Table B.1 for the architecture.

Table B.2: Test set BLEURT-20-D12 (mean & standard deviation of three runs). IL performs the best among the three proposed methods. *: The score is significant (p-value smaller than 0.05) compared to the beam search results. †: The score is significantly higher (p-value smaller than 0.05) than BSR results, or the score is not significantly different (p-value larger than 0.05) from the BSR results.

	IWSLT'14 De-En	WMT'16 Ro-En	WMT'14 De-En
p_f (greedy)	62.40 (0.04)	61.14 (0.10)	64.83 (0.10)
p_f (beam)	63.21 (0.07)	61.42 (0.15)	65.79 (0.08)
BSR	64.15 (0.05)	62.67 (0.13)	66.32 (0.12)
KD	63.88 (0.04) *	61.78 (0.10) *	66.00 (0.07) *
IL	63.94 (0.13) *†	62.35 (0.16) *†	66.14 (0.08) *†
Q learning	63.25 (0.07)	61.70 (0.18)	65.92 (0.14)

NUMBER OF PARAMETERS IN THE MODELS. The IWSLT'14 De-En transformer has 39,469,056 parameters, the WMT'16 Ro-En transformer has 62,046,208 parameters, and the WMT'14 De-En transformer has 209,911,808 parameters.

DISCUSSION ON Q LEARNING. In Section 4.5.3, to investigate whether better trajectories can improve Q learning results, we attempt adding high-reward trajectories from BSR as well as trajectories from a deep ensemble of two p_f 's. Deep ensembling two models (using different seeds) can produce high-quality translations. In this case, we simply want to use deep ensembling to diversify the sources of high-reward and high-BLEU trajectories. However, the result is that neither BLEU nor reward improves.

B.3 ETHICAL CONSIDERATIONS

IWSLT and WMT datasets are standard machine translation benchmarks. The datasets come from a variety of sources: phone conversations, parliament proceedings, news, and so on. There may be naturally occurring social biases in the datasets which have not undergone thorough cleansing. Training on these potential biases may lead to biased generations. There has been recent work studying such biases [Kocmi et al. 2020].

This work deals with speeding up inference, but not pretraining or training [Liu et al. 2021; Hou et al. 2022]. The standard practice of creating the pseudo-corpus requires a significant amount of computation. This step is optional, but it gives a boost in performance. We argue that if the MT system is put into production, then the benefit from the efficient inference will outweigh the cost of generating the pseudo-corpus.

C | APPENDIX FOR CHAPTER 5

C.1 PRACTICAL SETUP AND IMPLEMENTATION

C.1.1 TASKS AND DATASETS

(1) **Natural question generation** [NQG; Zhou et al. 2017b] based on the SQuAD QA dataset [Rajpurkar et al. 2016]: Given a text passage and a short span of the passage, the goal is to generate a question that can be answered by the span. (2) **CNN/DailyMail summarization (CNN/DM)**: Given a piece of news, generate a few sentences of summary. We use the entity-non-anonymized version of CNN/DM dataset, following See et al. [2017]. The target summaries tend to be *extractive*, meaning there tends to be heavy text-span overlaps between the source article and the target summary. (3) **Extreme summarization** [XSum; Narayan et al. 2018] is based on BBC news. The target summaries are highly *abstractive*. Past extractive strategies that work well for CNN/DM may not work well for XSum. (4) **IWSLT14 German to English machine translation** [IWSLT14 De-En; Cettolo et al. 2014] is a popular machine translation benchmark. Machine translation is different from the above three tasks, given that intuitively, the space of high-quality generation is smaller.

MORE DETAILS ON DATASETS. We first provide the number of examples in each dataset. The train/dev/test split for NQG is 86229/8913/8919; the split for CNN/DM is 287227/13368/11490; the

split for XSum is 204045/11332/11334; the split for IWSLT14 De-En is 160239/7283/6750.

To download and preprocess the NQG data, we follow the following instructions: <https://github.com/clovaai/FocusSeq2Seq>; to download and preprocess the summarization data, we follow the following instructions: <https://github.com/pytorch/fairseq/blob/master/examples/bart/README.summarization.md>; to download and preprocess the IWSLT14 De-En data, we follow the following instructions: <https://github.com/pytorch/fairseq/tree/master/examples/translation>. More information can be found in our codebase.

C.1.2 MODEL ARCHITECTURES

We use two sets of architectures for our experiments.

STANDARD ARCHITECTURES. For NQG, we use the model NQG++ [Zhou et al. 2017b], a seq2seq-with-attention model based on GRU [Cho et al. 2014], and for summarization we use pointer generator network [See et al. 2017], a seq2seq-with-attention model based on LSTM [Hochreiter and Schmidhuber 1997]. Specifically, we use 2 layers for both the encoder and the decoder, for both tasks. Other hyperparameters are based on the following implementation: <https://github.com/clovaai/FocusSeq2Seq>.

TRANSFORMER ARCHITECTURES. For NQG, CNN/DM, and XSum, we also experiment with one of the top-performing models, BART [Lewis et al. 2020]. Our experiments are based on the pretrained BART model provided by original authors¹: it has 12 encoder layers and 12 decoder layers, and it is pretrained on around 3.3 billion words of Wikipedia articles and books. We use the model to investigate if our methods work with models with stronger capabilities. For IWSLT14 De-En, we use a moderate-size standard transformer architecture (encoder/decoder embedding dimension 512, 4 encoder attention heads, 6 encoder layers, 4 decoder attention heads, 6 decoder layers), a

¹<https://github.com/pytorch/fairseq/tree/master/examples>; pretrained by corrupting the original document and optimized with respect to the reconstruction loss between the original document and the decoder output.

top-performing architecture in machine translation.

C.1.3 MORE ON REPRODUCIBILITY

The codebase is released. The link to the code is posted on the following website: yzpang.me.

HYPERPARAMETERS AND TRAINING DETAILS ON STANDARD ARCHITECTURES. This paragraph corresponds to results in [Table 5.1](#). We use a learning rate of $5e-4$. For NQG, we use a batch size of 32; for CNN/DM we use a batch size of 16. We train using a single Nvidia GTX 1080 Ti (memory: 12 GB) GPU.

As discussed in [Subsection 5.3.3](#) and [Subsection 5.4.1](#), we tune the lower bound of p_{MLE} in $\{0, 0.01, 0.05, 0.1\}$. For NQG models, the lower bound of 0.1 produces best performance. For CNN/DM using GOLD- p , the lower bound is 0.01; for CNN/DM using GOLD- s , the lower bound is 0.

Recall that as discussed in [Subsection 5.3.3](#), the weighting policy $\tilde{\pi}_\theta$ synchronizes with actual policy π_θ once every k steps so as to stabilize training. We tune $k \in \{1500, 2691\}$ (where 2691 steps corresponds to 1 epoch) for NQG and found that $k = 1500$ works better for all NQG models. We tune $k \in \{1500, 3000, 5000\}$ for CNN/DM; we found that $k = 1500$ works best for GOLD- δ and GOLD- p , and $k = 5000$ works best for GOLD- s . Note that in practice, we do not observe big gaps when using other k 's in the set. For standard models, implementation is based on [Cho et al. \[2019\]](#). In all experiments, we evaluate once every epoch, and we do validation on the entire dev set, using task-specific metrics (BLEU/ROUGE-2), following [Cho et al. \[2019\]](#) and standard practice in machine translation.

HYPERPARAMETERS AND TRAINING DETAILS ON TRANSFORMER MODELS. This paragraph corresponds to results in [Table 5.3](#). For transformer models, we use Nvidia P40 GPUs (memory: 24 GB each). For NQG, CNN/DM, and XSum based on BART, we use 4 GPUs to train. For IWSLT14

De-En, we use 1 GPU. Note that fairseq defines batch size in terms of number of tokens instead of number of sequences. For NQG, we use 512 tokens as batch size (for each of the four GPUs); for CNN/DM and XSum, we use 1024 tokens as batch size (for each of the four GPUs); for IWSLT14 De-En, we use 4096 tokens as batch size.

We use a learning rate of $2e-5$ for NQG, CNN/DM, and XSum; $3e-4$ for IWSLT14 De-En.

Recall that as discussed in [Subsection 5.3.3](#), the weighting policy $\tilde{\pi}_\theta$ synchronizes with actual policy π_θ once every k steps so as to stabilize training. Here, $k = 1000$ for NQG; $k = 5000$ for CNN/DM, XSum, IWSLT14 De-En. As discussed in [Subsection 5.3.3](#) and [Subsection 5.4.1](#), the lower bound of p_{MLE} is set to be 0.01 for GOLD- p and 0.1 for GOLD- s . For all other parameters that are not specific to GOLD, we use the default fairseq summarization parameters (which can be found through footnote 1).

For hyperparameter u as discussed in [Subsection 5.4.1](#), for NQG and CNN/DM, $u = 0.1$; for XSum, $u = 0.15$; for IWSLT14 De-EN, $u = 0.2$.

As indicated, the hyperparameters were only tuned in a small set of possible values. More careful tuning may result in slightly better performances.

NUMBER OF PARAMETERS IN EACH MODEL. For standard models, we use NQG++ for NQG, and it has 10372565 parameters. We use pointer generator for CNN/DM, and it has 19965705 parameters. For transformer models, the BART model for NQG, CNN/DM, and XSum all have 406290432 parameters; the transformer model used for IWSLT14 De-En has 39469056 parameters.

AVERAGE RUNTIME. For standard models, based on the above models and the computing infrastructures, each epoch of NQG takes around 10 minutes to train and achieves best performance within 20 epochs. Each epoch of CNN/DM takes about 2 hours to train and achieves best performance within 15 epochs. For transformer models, each epoch of NQG takes around 5 minutes to train and achieves best dev performance within 5 epochs; each epoch of CNN/DM takes around 11 hours to train and achieves best dev performances within 5 epochs; each epoch of XSum takes

around 8 hours to train; each epoch of IWSLT14 De-En takes around 3 minutes to train and achieves best performances within 100 epochs (as expected, given the large batch size²). Note that our transformer models are trained on P40s given hardware constraints; if the transformer models are trained on V100 GPUs, for example, the training time per epoch will likely be much shorter.

C.1.4 MORE DISCUSSION ON APPROXIMATIONS

Recall that we truncated the future trajectory after five steps. In other words, the number of current+future steps is upper-bounded at six. Effectively, we are using a discount factor of 0.83.³

Given that the Q -value corresponds to future return, we attempted using different strategies. (1) Using the entire future trajectory, and (2) using a fixed number of future steps. We attempted (1) on NQG using the standard models (tuned discount factor in $\{1, 0.9, 0.8, 0.7, 0.5\}$) and found that $\{0.8, 0.7\}$ usually performs best, resulting in similar performance but longer training time, compared to the current 5-future-step approach. We attempted (2) using the number of future steps in $\{1, 2, 3, 5, 7, 10\}$ and found that using $\{5, 7, 10\}$ leads to similar results, which are slightly better compared to $\{1, 2, 3\}$. One benefit is that given a fixed number of future steps, we found that using easy-to-tune constant baselines work well, and training time is also much shorter.

C.1.5 DETAILS ON ON-POLICY EXPERIMENTS

For the MLE+PG baseline, we used the REINFORCE algorithm with sequence-level rewards (BLEU for NQG and ROUGE-2 for summarization). We attempted two versions of the baselines: (i) constant baselines searched in $\{0, 0.01, 0.05, 0.1, 0.15\}$ for BLEU (NQG) and ROUGE-2 (CNN/DM), as well as (ii) baselines computed by the average BLEU/ROUGE-2 over the last 100 steps, minus $\{0, 0.05\}$.

In terms of training warmup choices, We tried two versions of the training algorithm. (a) We

²We use 4096 tokens (which corresponds to hundreds of sentences) as batch size for IWSLT14 De-En.

³Note that $1 + \gamma + \dots + \gamma^T \approx \frac{1}{1-\gamma} = 6$ when $\gamma = \frac{5}{6}$.

initialized with MLE and trained with PG losses interpolated with MLE losses, given we found that the training process would become very unstable without interpolation. (b) We also attempted the following: we initialized the model at random and used MIXER [Ranzato et al. 2016]. However, we failed to find improvements compared to (a), under our architecture. A relevant work Choshen et al. [2020] showed that properly tuned on-policy RL may not work for text generation in some cases.

We also tried MIXER with the learned baseline for NQG, which is estimated by a simple linear regressor that takes RNN hidden states as inputs, according to Ranzato et al. [2016]. After some tuning, we achieved only slight improvements in NQG (BLEU 14.71). One advantage of GOLD is that our algorithm does not rely on learning baselines which could have a big impact on performance of on-policy algorithms; in fact, all baselines are constants in this paper.

Note that for GOLD+PG models, we only attempted constant baselines; better tuning of baselines could potentially lead to stronger performance.

C.2 MORE ON RESULTS

C.2.1 LEAD-3 BASELINES FOR SUMMARIZATION

The lead-3 baseline (using first 3 sentences as summaries) is a popular strong baseline in summarization literature. The ROUGE-1/2/L scores of the lead-3 baselines are as follows: 40.42/17.62/36.67 for CNN/DM; 16.30/1.60/11.95 for XSum. Our performance using transformer models beat these baselines by a large margin.

C.2.2 PERFORMANCE WITH TRANSFORMER ARCHITECTURES

We experiment using transformer architectures, as shown in Table 5.3; we also experiment on two more tasks (compared to using standard architectures): XSum and IWSLT14 De-En. We achieve

SOTA/near-SOTA result (according to automatic metrics which have inherent limitations) on CN-
N/DM: at the time of writing, our results (45.40/22.01/42.25 using GOLD-*p* or 44.82/22.09/41.81 us-
ing GOLD-*s*) are higher than 44.17/21.47/41.11 [PEGASUS; Zhang et al. 2020a] and 44.20/21.17/41.30
[ProphetNet; Qi et al. 2020], both slightly higher than BART. Note the PEGASUS CNN/DM result
is pretrained on 1.5B news *articles* (around 3.8 terabyte), whereas BART is pretrained on 3.3B
words (around 0.16 tetrabyte). Our XSum results are also higher than PEGASUS (45.20/22.06/36.99)
trained on Colossal Clean Crawled Corpus [C4; Raffel et al. 2020], but lower than the PEGASUS
result using the publicly-unavailable 1.5B-*article* 3.8 terabyte HugeNews [Zhang et al. 2020a] as
pretrained corpus. We hypothesize that if our models are applied onto their architectures instead
of pointer generator networks or BART, we would similarly get non-trivial improvements.

We also achieve 0.81 point of BLEU improvement on IWSLT14 De-En; GOLD-*s* performs better
than the existing approaches that do not use knowledge distillation or data augmentation, as far
as the authors are aware.

C.2.3 MORE ON EXPOSURE BIAS

WITH EXPOSURE BIAS. Recall that in [Subsection 5.4.2](#), we used human evaluation (a score of 1 or
2 or 3 or 4) to approximate the output quality, and we found that the MLE-trained model degrades
significantly when the generation length is long, whereas the quality of the GOLD-*s*-trained model
stays relatively stable across lengths.

Here, we use BLEU to approximate the quality of NQG generations, and we show that BLEU
does not bias toward long sentences. [Figure C.1](#) shows the average sentence-level BLEU by
sequence length.⁴

Specifically, [Figures C.1\(a\)](#) and [C.1\(c\)](#) show the BLEU on randomly shuffled targets (from dev
set), which show that longer sentences do not appear to punish BLEU scores. [Figures C.1\(b\)](#)
and [C.1\(d\)](#) show the BLEU by sentence length, on model generations. We see that MLE’s BLEU

⁴We use BLEU-2/3 given that without smoothing, sentence-level BLEU-4 results in large variance.

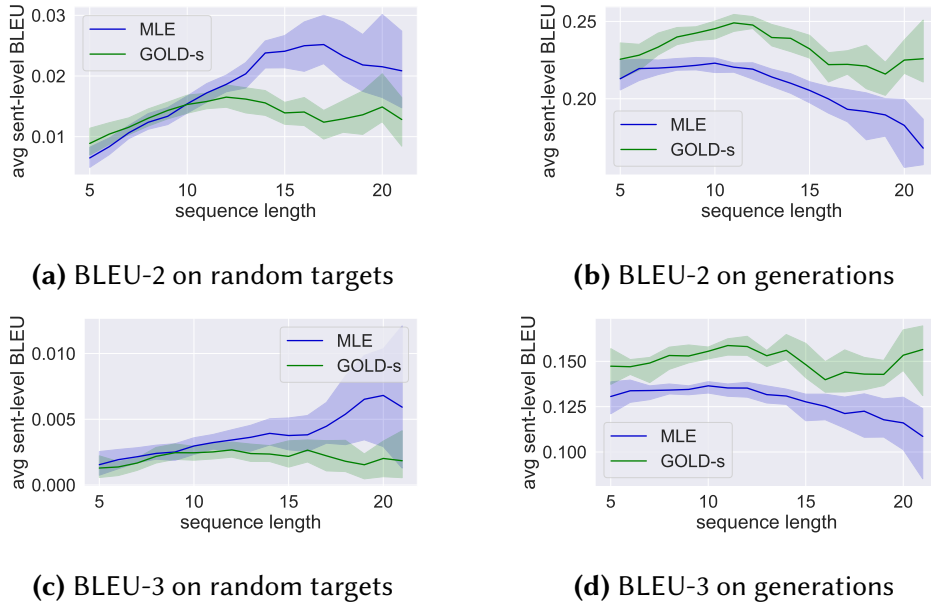


Figure C.1: Exposure bias related figures on NQG dev set. Vertical axis: avg unsmoothed sentence-level BLEU. Horizontal axis: sentence length. The colored regions represent 95% confidence interval obtained using standard bootstrapping. Subfigures (a) and (c) show BLEU on randomly shuffled targets (from dev set); BLEU does not appear to punish long sentences. Note the scale of the vertical axes. Subfigures (b) and (d) show BLEU vs. generation length; BLEU on generations from MLE-trained model decreases by length, but BLEU on generations from GOLD-trained model appears to stay relatively stable.

decreases by length but GOLD-s’s BLEU appears to stay relatively stable. We thus see some evidence that MLE is generating worse sentences as sentence gets longer.

IF THERE IS NO EXPOSURE BIAS. In the main text, we used the NLL loss vs. length plot to demonstrate that without exposure bias, the loss does not vary much across length, so the MLE performance drop in Figure 5.2 (left) is mainly due to exposure bias. Here, we provide another way to analyze the case without exposure bias.

Figure C.2 shows the token prediction accuracy conditioned on gold histories on NQG dev set. Note that for each example, we let $t_x = L_x - 5$, where L_x is the length of reference sentence x . We can see that without exposure bias, prediction accuracy does not vary much as the length increases. Therefore, we conclude that the big performance drop for long generations using MLE is mainly due to exposure bias and GOLD suffers less from the problem.

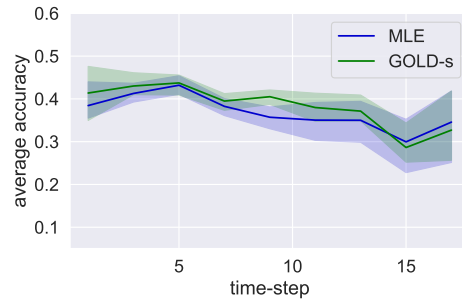


Figure C.2: Accuracy of correct predictions of t th token given all prefix reference tokens on NQG dev set. Colored regions represents 95% confidence interval obtained using standard bootstrapping. Without exposure bias, token prediction accuracy stays relatively stable across lengths.

C.2.4 EXAMPLES

[Table C.1](#) and [Table C.2](#) show the example generations based on the transformer models.

Table C.1: NQG and CNN/DM examples based on transformer models. For NQG, words to query on are bolded.

task	objective	example
NQG	input	Some members of this community emigrated to the United States in the 1890s .
	MLE	when did some members of the portuguese-american community emigrate to the us ?
	GOLD-s	when did some members of the community emigrate to the us ?
	reference	in what era did some members of this community emigrate to the us ?
NQG	input	Competition amongst workers tends to drive down wages due to the expendable nature of the worker in relation to his or her particular job .
	MLE	what is one of the reasons that causes wages to be lower ?
	GOLD-s	why do wages go down when there is competition amongst workers ?
	reference	why does competition among workers drive down wages ?
NQG	input	During the mid-eocene , it is believed that the drainage basin of the Amazon was split along the middle of the continent by the Purus Arch .
	MLE	when was the purus arch formed ?
	GOLD-s	when was the drainage basin of the amazon split ?
	reference	in which point did the drainage basin of the amazon split ?
CNN/DM	input	[omitted due to length and copyright issues, but the original news article can be retrieved by searching the reference online]
	MLE	There are nearly 5,000 “gems” scattered across the country, ranging from museums to archaeological areas and monuments. Italy boasts the highest number of UNESCO World Heritage sites in the world. Several of which risk crumbling to the ground due to neglect and lack of public resources.
	GOLD-s	Italy boasts the highest number of UNESCO World Heritage sites in the world . The Basilica of Assisi, where St. Frances’ tomb lies, is badly in need of a restyle . Italy doesn’t know how to exploit this treasure, says Francesco Totti .
	reference	Italy boasts the highest number of UNESCO World Heritage sites in the world . Italy doesn’t know how to exploit treasures, and appears not to care about them, writes Silvia Marchetti .
CNN/DM	input	[omitted due to length and copyright issues, but the original news article can be retrieved by searching the reference online]
	MLE	President Obama has argued with progressive potentate Elizabeth Warren, calling her “wrong” on trade policy. What everyone does next will be critical for the 2016 elections and the future of Democratic politics. Warren has publicly criticized “fast track” trade authority that would allow the White House to negotiate massive, multination trade deals.
	GOLD-s	President Obama has argued with the progressive potentate Elizabeth Warren, calling her “wrong” on trade policy . Julian Zelizer: If Hillary Clinton wants to prove she’s a real populist, now is her chance to be even more clear about her position on the TPP deal .
	reference	Sen. Elizabeth Warren has publicly criticized so-called “fast track” trade authority . Sally Kohn: Why does President Obama call her wrong, and why is Hillary Clinton equivocating?

Table C.2: XSum and IWSLT14 De-En examples based on transformer models.

task	objective	example
XSum	input	[omitted due to length and copyright issues, but the original news article can be retrieved by searching the reference online]
	MLE	The Isle of Wight father’s decision not to pay a fine for taking his seven-year-old daughter on holiday during term time caused “a huge amount of confusion”, a senior MP has said.
	GOLD-s	The High Court ruling that a father could not be prosecuted for taking his seven-year-old daughter on a term-time holiday to Disney World caused “a huge amount of confusion”, MPs have said.
	reference	A High Court ruling backing a parent who refused to pay a fine for taking his child on holiday in term time will cause “huge confusion”, an MP has said.
XSum	input	[omitted due to length and copyright issues, but the original news article can be retrieved by searching the reference online]
	MLE	Flood defences at a Denbighshire beach could be strengthened to reduce the risk of them being breached.
	GOLD-s	A new dune system could be built to protect a Denbighshire beach from flooding.
	reference	New sand dunes may be created to reduce the risk of flooding on a beach on the Denbighshire and Flintshire border.
XSum	input	[omitted due to length and copyright issues, but the original news article can be retrieved by searching the reference online]
	MLE	Fleetwood’s League One play-off hopes suffered a blow as they were held to a goalless draw by League One strugglers Doncaster.
	GOLD-s	Fleetwood and Blackburn played out a goalless draw in League One.
	reference	Fleetwood Town dropped into the League One relegation places as they had to settle for a point after a stalemate with Doncaster.
IWSLT14 De-En	input	ich hab da so ne kognitive rückkopplung, du hast was projiziert, was du sehen möchtest.
	MLE	i’ve been doing this with cognitive feedback, you’ve been prospecting what you want to see.
	GOLD-s	i’ve got cognitive feedback, you’ve proved what you want to see.
	reference	i have this cognitive feedback, you projected something you want to see.
IWSLT14 De-En	input	es sind also alle werkzeuge vorhanden, und die einzige sache, die uns limitiert, ist unsere vorstellungskraft.
	MLE	so there are all the tools available, and the only thing that’s licensed to us is our imagination .
	GOLD-s	so there are all the tools there, and the only thing that limited us is our imagination.
	reference	so all the tools are out there, and the only thing that limits us is our imagination.
IWSLT14 De-En	input	unser organismus hat eine großartige methode erfunden, um solche unangenehmen gefühle wie neid einfach zum verschwinden zu bringen.
	MLE	our organism has invented a great way to get such uncomfortable emotions as neither of us to disappear.
	GOLD-s	our organism invented a great way to make such uncomfortable emotions like envy easy to disappear.
	reference	our organism has come up with an excellent method to make unpleasant feelings like envy simply disappear.

C.3 HUMAN EVALUATIONS

C.3.1 PAIRWISE COMPARISON

Our goal is to enable high-quality generations that do not necessarily result in gold references. Given that corpus-level BLEU/ROUGE score is only a popular *approximation* of generation quality, we first conduct human ratings to confirm the hypothesis that our approaches are generating better sequences. For NQG, for each unit of human evaluation, we present the source paragraph, the words to ask the question on, the question generated by MLE-trained model, as well as the question generated by GOLD-*s*-trained model. We ask the human evaluators the general question: which generated question is better? [Figure C.3](#) shows one example interface of pairwise comparisons.

Using NQG dev set, on standard models, of the 183 pairs of comparison we conducted human evaluations on, 42 (23.0%) MLE-questions are better, 81 (44.3%) GOLD-*s*-questions are better, and 60 (32.8%) are tied. We also evaluate on models based on BART, shown in [Table 5.5](#) in the main text.

For summarization tasks, given that it is infeasible to get high-quality annotations if we let workers read the entire news article⁵, we only did the following: given the reference summary, a summary generated from MLE model, and a summary generated from our model, we asked workers to compare which generated summary is closer in meaning to the reference summary. [Figure C.4](#) shows one example interface of the mentioned pairwise comparison for summarization. See [Table 5.5](#) for results.

C.3.2 NQG RATING

NQG rating was conducted to examine if longer sentences (generated by MLE-trained model) will result in worse human ratings, and if GOLD alleviates the problem. In [Figure 5.2](#) (left), to reduce

⁵To obtain high-quality low-variance annotations, we may need to design QA tasks to make sure workers understood the news articles first, given the articles are usually very long.

Instructions ×

[View full instructions](#)

[View tool guide](#)

The questions are generated based on (1) the given paragraph and (2) the given words to query on. Which of the two generated questions is better? Please make sure to read the given paragraph carefully before reading the questions. There are HITs (with obvious choices) as control.

Which generated question is better? Only select choice C if you really cannot distinguish the qualities. Please make sure to read the paragraph before reading the questions.

Paragraph: Oxygen is the most abundant element by mass in the Earth's crust as part of < oxide compounds > such as silicon dioxide, making up almost half of the crust's mass.

Words to ask questions on are shown between the angle brackets. For example, in "abc < def > ghi jkl", we want to ask questions on the word "def".

Question A: what is the most abundant element in the earth's crust ?

Question B: what is oxygen in the earth's crust ?

Select an option

A: Question A is better	1
B: Question B is better	2
C: the two questions are equally good/bad	3

Figure C.3: Interface for NQG pairwise comparisons, using Amazon Mechanical Turk.

Instructions ×

[View full instructions](#)

[View tool guide](#)

Reference summary, generated summary A, and generated summary B are three summaries from the same news article. Is A closer to the reference in meaning, or is B closer? Please make your choices carefully, as there're HITs (with obvious answers) as quality control.

Which generated summary is closer to the reference summary IN MEANING? Only select choice C if you REALLY cannot decide.

Here is the reference summary (from a news article).

Reference summary: The cartoon character Linus van Pelt from the Peanuts comic strip has a comfort blanket, and so do lots of children.

Below are two generated summaries (from the same news article).

Generated summary A: I have been out and about, travelling, and watching other travellers recently.

Generated summary B: I have been thinking about the history of comforters.

Select an option

A: Summary A is closer to the reference in meaning	1
B: Summary B is closer to the reference in meaning	2
C: the two summaries are equally close to the reference	3

Figure C.4: Interface for summarization pairwise comparisons, using Amazon Mechanical Turk.

Instructions ×

[View full instructions](#)

[View tool guide](#)

The questions are generated based on (1) the given paragraph and (2) the given words to query on. How would you rate the generated question? Things to consider: correctness (is the question asking about the right thing, using factually correct information?); quality of the sentence (fluency, etc.)

How would you rate the generated question? Please read instructions.

Please answer carefully, as there're HITs (with obvious choices) as control.

Paragraph: following their loss in the divisional round of the previous season's playoffs, the denver broncos underwent numerous coaching changes, including a mutual parting with head coach john fox (who had won four divisional championships in his four years as broncos head coach), and the hiring of gary kubiak as the new head coach.

Words to ask question on: four

Generated question: how many divisional championships did john fox win in his four years ?

Select an option

1: very bad	1
2: slightly below average	2
3: slightly above average	3
4: very good	4

Figure C.5: Interface for NQG human ratings, using Amazon Mechanical Turk.

variance, we group length by buckets of two (e.g., [7, 8], [9, 10], [11, 12], etc.). Furthermore, we sampled 736 annotations such that each bucket would contain at least 30 sentences (for human evaluation) for each of MLE and GOLD-s. We also shown the 95% confidence interval using

standard bootstrapping, in [Figure 5.2](#) (left).

Given the paragraph, words to query on, and the generations, we ask workers to rate the generations. [Figure C.5](#) shows an example interface of NQG human ratings. We ask workers to consider both the correctness of the generation (i.e., if the question is asking about the specified words using facts) and the quality of the generation (i.e., if the generation is fluent and coherent). We ask workers to rate from 1 to 4, where 1 means very bad, 2 means slightly below average, 3 means slightly above average, and 4 means very good.

D | APPENDIX FOR CHAPTER 6

D.1 MORE DETAILS ON EXPERIMENTAL SETUP

D.1.1 PROMPTS

GSM8K. For each GSM8K question, we use the following prompt as the input to the language model:

Your task is to answer the question below. Give step by step reasoning before you answer, and when you're ready to answer, please use the format "Final answer: ..."

Question: [question here]

Solution:

MATH. For each MATH question, we use the following prompt as the input to the language model. In particular, the prompt includes four fixed in-context examples chosen from the training set of MATH. The language model needs these demonstrations so that the final answers can be properly formatted in \LaTeX .

Your task is to answer the last question below. Give step by step reasoning before you answer, and when you're ready to answer, please wrap your answer in `\boxed`, and conclude using the format "Final answer: ..."

Question: [question for the first example]

Solution: [solution for the first example]

Final answer: [answer (e.g., number, formula) here]

Question: [question for the second example]

Solution: [solution for the second example]

Final answer: [answer here]

Question: [question for the third example]

Solution: [solution for the third example]

Final answer: [answer here]

Question: [question for the fourth example]

Solution: [solution for the fourth example]

Final answer: [answer here]

Question: [the question to be solved]

Solution:

ARC. For each ARC question, we use the following prompt as the input to the language model, assuming the question has four options (each question has three to five options).

Your task is to answer the question below. Give step by step reasoning before you answer, and when you're ready to answer, conclude using the format "Final answer: (insert letter here)"

Question: [question here]

(A) [option A here]

(B) [option B here]

(C) [option C here]

(D) [option D here]

Solution:

BIBLIOGRAPHY

- J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altschmidt, S. Altman, S. Anadkat, et al. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023. 82
- O. Adewumi, R. Brännvall, N. Abid, M. Pahlavan, S. Sabah Sabry, F. Liwicki, and M. Liwicki. Småprat: Dialogpt for natural language generation of swedish dialogue by transfer learning. In *5th Northern Lights Deep Learning Conference (NLDL), Tromsø, Norway, January 10-12, 2022*, volume 3. Septentrio Academic Publishing, 2022. URL <https://septentrio.uit.no/index.php/nldl/article/view/6231>. 21
- L. Adolphs, T. Gao, J. Xu, K. Shuster, S. Sukhbaatar, and J. Weston. The CRINGE loss: Learning what language not to model. In A. Rogers, J. Boyd-Graber, and N. Okazaki, editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8854–8874, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.493. URL <https://aclanthology.org/2023.acl-long.493>. 17, 86
- R. Aharoni and Y. Goldberg. Split and rephrase: Better evaluation and stronger baselines. In I. Gurevych and Y. Miyao, editors, *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 719–724, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-2114. URL <https://aclanthology.org/P18-2114>. 53

- G. D. AlphaCode Team. Alphacode2 technical report. 2023. URL https://storage.googleapis.com/deepmind-media/AlphaCode2/AlphaCode2_Tech_Report.pdf. 69
- D. Amodei, C. Olah, J. Steinhardt, P. Christiano, J. Schulman, and D. Mané. Concrete problems in AI safety. *arXiv preprint arXiv:1606.06565*, 2016. URL <https://arxiv.org/abs/1606.06565>. 2, 3
- C. Amrhein and R. Sennrich. Identifying weaknesses in machine translation metrics through minimum Bayes risk decoding: A case study for COMET. In Y. He, H. Ji, S. Li, Y. Liu, and C.-H. Chang, editors, *Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1125–1141, Online only, Nov. 2022. Association for Computational Linguistics. URL <https://aclanthology.org/2022.aacl-main.83>. 16
- D. Andor, C. Alberti, D. Weiss, A. Severyn, A. Presta, K. Ganchev, S. Petrov, and M. Collins. Globally normalized transition-based neural networks. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2442–2452, Berlin, Germany, Aug. 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1231. URL <https://www.aclweb.org/anthology/P16-1231>. 67
- 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. 82
- Anthropic Team. Claude 2, 2023. URL <https://www.anthropic.com/news/claude-2>. 82
- K. Arora, L. El Asri, H. Bahuleyan, and J. Cheung. Why exposure bias matters: An imitation learning perspective of error accumulation in language generation. In S. Muresan, P. Nakov, and A. Villavicencio, editors, *Findings of the Association for Computational Linguistics: ACL 2022*, pages 700–710, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi:

10.18653/v1/2022.findings-acl.58. URL <https://aclanthology.org/2022.findings-acl.58>.

9

D. Bahdanau, P. Brakel, K. Xu, A. Goyal, R. Lowe, J. Pineau, A. Courville, and Y. Bengio. An actor-critic algorithm for sequence prediction. In *International Conference on Learning Representations*, 2017. URL <https://openreview.net/forum?id=SJDaqqveg>. 47

A. Baheti, X. Lu, F. Brahman, R. L. Bras, M. Sap, and M. Riedl. Leftover lunch: Advantage-based offline reinforcement learning for language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL <https://openreview.net/forum?id=ZDGKPbF0VQ>. 69, 70

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*, 2022. URL <https://arxiv.org/abs/2204.05862>. 10, 19, 69, 86, 91

M. Bakker, M. Chadwick, H. Sheahan, M. Tessler, L. Campbell-Gillingham, J. Balaguer, N. McAleese, A. Glaese, J. Aslanides, M. Botvinick, and C. Summerfield. Fine-tuning language models to find agreement among humans with diverse preferences. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 38176–38189. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/f978c8f3b5f399cae464e85f72e28503-Paper-Conference.pdf. 10

Y. Bang, N. Lee, E. Ishii, A. Madotto, and P. Fung. Assessing political prudence of open-domain chatbots. In H. Li, G.-A. Levow, Z. Yu, C. Gupta, B. Sisman, S. Cai, D. Vandyke, N. Dethlefs, Y. Wu, and J. J. Li, editors, *Proceedings of the 22nd Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 548–555, Singapore and Online, July 2021. Association for

- Computational Linguistics. doi: 10.18653/v1/2021.sigdial-1.57. URL <https://aclanthology.org/2021.sigdial-1.57>. 21
- D. Belanger and A. McCallum. Structured prediction energy networks. In M. F. Balcan and K. Q. Weinberger, editors, *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 983–992, New York, New York, USA, 20–22 Jun 2016. PMLR. URL <http://proceedings.mlr.press/v48/belanger16.html>. 67
- S. Bengio, O. Vinyals, N. Jaitly, and N. Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015. URL https://proceedings.neurips.cc/paper_files/paper/2015/file/e995f98d56967d946471af29d7bf99f1-Paper.pdf. 9, 50, 67
- S. Bhosale, K. Yee, S. Edunov, and M. Auli. Language models not just for pre-training: Fast online neural noisy channel modeling. In L. Barrault, O. Bojar, F. Bougares, R. Chatterjee, M. R. Costajussà, C. Federmann, M. Fishel, A. Fraser, Y. Graham, P. Guzman, B. Haddow, M. Huck, A. J. Yepes, P. Koehn, A. Martins, M. Morishita, C. Monz, M. Nagata, T. Nakazawa, and M. Negri, editors, *Proceedings of the Fifth Conference on Machine Translation*, pages 584–593, Online, Nov. 2020. Association for Computational Linguistics. URL <https://aclanthology.org/2020.wmt-1.69>. 48
- O. Bojar, C. Buck, C. Federmann, B. Haddow, P. Koehn, J. Leveling, C. Monz, P. Pecina, M. Post, H. Saint-Amand, R. Soricut, L. Specia, and A. Tamchyna. Findings of the 2014 workshop on statistical machine translation. In O. Bojar, C. Buck, C. Federmann, B. Haddow, P. Koehn, C. Monz, M. Post, and L. Specia, editors, *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 12–58, Baltimore, Maryland, USA, June 2014. Association for Computational Linguistics. doi: 10.3115/v1/W14-3302. URL <https://aclanthology.org/W14-3302>. 40

- O. Bojar, R. Chatterjee, C. Federmann, Y. Graham, B. Haddow, M. Huck, A. Jimeno Yepes, P. Koehn, V. Logacheva, C. Monz, M. Negri, A. N  v  ol, M. Neves, M. Popel, M. Post, R. Rubino, C. Scarton, L. Specia, M. Turchi, K. Verspoor, and M. Zampieri. Findings of the 2016 conference on machine translation. In O. Bojar, C. Buck, R. Chatterjee, C. Federmann, L. Guillou, B. Haddow, M. Huck, A. J. Yepes, A. N  v  ol, M. Neves, P. Pecina, M. Popel, P. Koehn, C. Monz, M. Negri, M. Post, L. Specia, K. Verspoor, J. Tiedemann, and M. Turchi, editors, *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 131–198, Berlin, Germany, Aug. 2016. Association for Computational Linguistics. doi: 10.18653/v1/W16-2301. URL <https://aclanthology.org/W16-2301>. 38
- E. Brill and R. C. Moore. An improved error model for noisy channel spelling correction. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, pages 286–293, Hong Kong, Oct. 2000. Association for Computational Linguistics. doi: 10.3115/1075218.1075255. URL <https://aclanthology.org/P00-1037>. 29
- M. Caccia, L. Caccia, W. Fedus, H. Larochelle, J. Pineau, and L. Charlin. Language gans falling short. In *International Conference on Learning Representations*, 2020. URL <https://openreview.net/forum?id=BJgza6VtPB>. 65
- J. Camacho-collados, K. Rezaee, T. Riahi, A. Ushio, D. Loureiro, D. Antypas, J. Boisson, L. Espinosa Anke, F. Liu, and E. Mart  nez C  mara. TweetNLP: Cutting-edge natural language processing for social media. In W. Che and E. Shutova, editors, *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–49, Abu Dhabi, UAE, Dec. 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.emnlp-demos.5. URL <https://aclanthology.org/2022.emnlp-demos.5>. 23
- M. Cettolo, J. Niehues, S. St  ker, L. Bentivogli, and M. Federico. Report on the 11th IWSLT evalua-

- tion campaign. In *Proceedings of the International Workshop on Spoken Language Translation*, volume 57, Hanoi, Vietnam, 2014. 38, 59, 122
- H. Chen, X. Liu, D. Yin, and J. Tang. A survey on dialogue systems: Recent advances and new frontiers. *ACM SIGKDD Explorations Newsletter*, 19(2):25–35, 2017. 21
- P.-J. Chen, A. Lee, C. Wang, N. Goyal, A. Fan, M. Williamson, and J. Gu. Facebook AI’s WMT20 news translation task submission. In L. Barrault, O. Bojar, F. Bougares, R. Chatterjee, M. R. Costa-jussà, C. Federmann, M. Fishel, A. Fraser, Y. Graham, P. Guzman, B. Haddow, M. Huck, A. J. Yepes, P. Koehn, A. Martins, M. Morishita, C. Monz, M. Nagata, T. Nakazawa, and M. Negri, editors, *Proceedings of the Fifth Conference on Machine Translation*, pages 113–125, Online, Nov. 2020. Association for Computational Linguistics. URL <https://aclanthology.org/2020.wmt-1.8>. 5, 30, 32
- Y. Chen, V. O. Li, K. Cho, and S. Bowman. A stable and effective learning strategy for trainable greedy decoding. In E. Riloff, D. Chiang, J. Hockenmaier, and J. Tsujii, editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 380–390, Brussels, Belgium, Oct.-Nov. 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1035. URL <https://aclanthology.org/D18-1035>. 33, 47
- Z. Chen, Y. Deng, H. Yuan, K. Ji, and Q. Gu. Self-play fine-tuning converts weak language models to strong language models. In *Forty-first International Conference on Machine Learning*, 2024. URL <https://openreview.net/forum?id=04cHTxW9BS>. 7, 71, 86
- J. Cho, M. Seo, and H. Hajishirzi. Mixture content selection for diverse sequence generation. In K. Inui, J. Jiang, V. Ng, and X. Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3121–3131, Hong Kong, China,

- Nov. 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1308. URL <https://aclanthology.org/D19-1308>. 124
- K. Cho, B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In A. Moschitti, B. Pang, and W. Daelemans, editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar, Oct. 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1179. URL <https://aclanthology.org/D14-1179>. 123
- L. Choshen, L. Fox, Z. Aizenbud, and O. Abend. On the weaknesses of reinforcement learning for neural machine translation. In *International Conference on Learning Representations, 2020*. URL <https://openreview.net/forum?id=H1eCw3EKvH>. 5, 48, 50, 53, 64, 127
- E. Clark, A. Celikyilmaz, and N. A. Smith. Sentence mover’s similarity: Automatic evaluation for multi-sentence texts. In A. Korhonen, D. Traum, and L. Màrquez, editors, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2748–2760, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1264. URL <https://aclanthology.org/P19-1264>. 53
- P. Clark, I. Cowhey, O. Etzioni, T. Khot, A. Sabharwal, C. Schoenick, and O. Tafjord. Think you have solved question answering? Try ARC, the AI2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018. 83
- K. Cobbe, V. Kosaraju, M. Bavarian, M. Chen, H. Jun, L. Kaiser, M. Plappert, J. Tworek, J. Hilton, R. Nakano, C. Hesse, and J. Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021. 77
- E. Cohen and C. Beck. Empirical analysis of beam search performance degradation in neural sequence models. In K. Chaudhuri and R. Salakhutdinov, editors, *Proceedings of the 36th*

- International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 1290–1299. PMLR, 09–15 Jun 2019. URL <https://proceedings.mlr.press/v97/cohen19a.html>. 66
- R. Collobert and J. Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th International Conference on Machine Learning*, pages 160–167, 2008. 92
- R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12:2493–2537, 2011. 18
- H. Daumé, J. Langford, and D. Marcu. Search-based structured prediction. *Machine learning*, 75(3): 297–325, 2009. 67
- T. Dettmers, A. Pagnoni, A. Holtzman, and L. Zettlemoyer. QLoRA: Efficient finetuning of quantized llms. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 10088–10115. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/1feb87871436031bdc0f2beaa62a049b-Paper-Conference.pdf. 69
- E. Dinan, V. Logacheva, V. Malykh, A. Miller, K. Shuster, J. Urbanek, D. Kiela, A. Szlam, I. Serban, R. Lowe, et al. The second conversational intelligence challenge (ConvAI2). In *The NeurIPS’18 Competition: From Machine Learning to Intelligent Conversations*, pages 187–208. Springer, 2020. 28
- D. Donato, L. Yu, W. Ling, and C. Dyer. Mad for robust reinforcement learning in machine translation. *arXiv preprint arXiv:2207.08583*, 2022. 5
- Z.-Y. Dou, A. Kamath, Z. Gan, P. Zhang, J. Wang, L. Li, Z. Liu, C. Liu, Y. LeCun, N. Peng, J. Gao, and L. Wang. Coarse-to-fine vision-language pre-training with fusion in the backbone. In A. H.

- Oh, A. Agarwal, D. Belgrave, and K. Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL <https://openreview.net/forum?id=o4neHaKmlse>. 70
- R. Dror, G. Baumer, S. Shlomov, and R. Reichart. The hitchhiker’s guide to testing statistical significance in natural language processing. In I. Gurevych and Y. Miyao, editors, *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1383–1392, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1128. URL <https://aclanthology.org/P18-1128>. 100
- Y. Dubois, X. Li, R. Taori, T. Zhang, I. Gulrajani, J. Ba, C. Guestrin, P. Liang, and T. Hashimoto. AlpacaFarm: A simulation framework for methods that learn from human feedback. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL <https://openreview.net/forum?id=4hturzLcKX>. 16, 17, 24
- A. Echihiabi and D. Marcu. A noisy-channel approach to question answering. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 16–23, Sapporo, Japan, July 2003. Association for Computational Linguistics. doi: 10.3115/1075096.1075099. URL <https://aclanthology.org/P03-1003>. 5, 29
- M. Freitag, Y. Al-Onaizan, and B. Sankaran. Ensemble distillation for neural machine translation. *arXiv preprint arXiv:1702.01802*, 2017. 33, 47
- M. Freitag, G. Foster, D. Grangier, and C. Cherry. Human-paraphrased references improve neural machine translation. In L. Barrault, O. Bojar, F. Bougares, R. Chatterjee, M. R. Costa-jussà, C. Federmann, M. Fishel, A. Fraser, Y. Graham, P. Guzman, B. Haddow, M. Huck, A. J. Yepes, P. Koehn, A. Martins, M. Morishita, C. Monz, M. Nagata, T. Nakazawa, and M. Negri, editors, *Proceedings of the Fifth Conference on Machine Translation*, pages 1183–1192, Online, Nov. 2020. Association for Computational Linguistics. URL <https://aclanthology.org/2020.wmt-1.140>. 16

- M. Freitag, G. Foster, D. Grangier, V. Ratnakar, Q. Tan, and W. Macherey. Experts, errors, and context: A large-scale study of human evaluation for machine translation. *Transactions of the Association for Computational Linguistics*, 9:1460–1474, 2021. doi: 10.1162/tacl_a_00437. URL <https://aclanthology.org/2021.tacl-1.87>. 10
- J. Gao, M. Galley, and L. Li. Neural approaches to conversational ai. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 1371–1374, 2018. 21
- L. Gao, J. Schulman, and J. Hilton. Scaling laws for reward model overoptimization. In A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, and J. Scarlett, editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 10835–10866. PMLR, 23–29 Jul 2023. URL <https://proceedings.mlr.press/v202/gao23h.html>. 12
- F. Gilardi, M. Alizadeh, and M. Kubli. ChatGPT outperforms crowd workers for text-annotation tasks. *Proceedings of the National Academy of Sciences*, 120(30):e2305016120, 2023. doi: 10.1073/pnas.2305016120. URL <https://www.pnas.org/doi/abs/10.1073/pnas.2305016120>. 100
- K. Goyal, G. Neubig, C. Dyer, and T. Berg-Kirkpatrick. A continuous relaxation of beam search for end-to-end training of neural sequence models. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018. 67
- J. Gu, K. Cho, and V. O. Li. Trainable greedy decoding for neural machine translation. In M. Palmer, R. Hwa, and S. Riedel, editors, *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1968–1978, Copenhagen, Denmark, Sept. 2017. Association for Computational Linguistics. doi: 10.18653/v1/D17-1210. URL <https://aclanthology.org/D17-1210>. 47
- J. Gu, J. Bradbury, C. Xiong, V. O. Li, and R. Socher. Non-autoregressive neural machine translation.

- In *International Conference on Learning Representations*, 2018. URL <https://openreview.net/forum?id=B1l8BtlCb>. 47
- C. Gulcehre, T. L. Paine, S. Srinivasan, K. Konyushkova, L. Weerts, A. Sharma, A. Siddhant, A. Ahern, M. Wang, C. Gu, et al. Reinforced self-training (ReST) for language modeling. *arXiv preprint arXiv:2308.08998*, 2023. 17, 86
- C. Gunasekara, G. Feigenblat, B. Sznajder, R. Aharonov, and S. Joshi. Using question answering rewards to improve abstractive summarization. In M.-F. Moens, X. Huang, L. Specia, and S. W.-t. Yih, editors, *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 518–526, Punta Cana, Dominican Republic, Nov. 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.findings-emnlp.47. URL <https://aclanthology.org/2021.findings-emnlp.47>. 9
- H. Hachiya, T. Akiyama, M. Sugiayma, and J. Peters. Adaptive importance sampling for value function approximation in off-policy reinforcement learning. *Neural Networks*, 22(10):1399–1410, 2009. 50
- J. Hartmann. <https://huggingface.co/j-hartmann>, 2022. 23
- T. B. Hashimoto, H. Zhang, and P. Liang. Unifying human and statistical evaluation for natural language generation. In J. Burstein, C. Doran, and T. Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1689–1701, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1169. URL <https://aclanthology.org/N19-1169>. 51
- W. K. Hastings. Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57(1):97–109, 1970. ISSN 00063444. URL <http://www.jstor.org/stable/2334940>. 50

- A. Havrilla, Y. Du, S. C. Raparthy, C. Nalmpantis, J. Dwivedi-Yu, M. Zhuravinskyi, E. Hambro, S. Sukhbaatar, and R. Raileanu. Teaching large language models to reason with reinforcement learning. *arXiv preprint arXiv:2403.04642*, 2024. 87
- T. Hayes, R. Rao, H. Akin, N. J. Sofroniew, D. Oktay, Z. Lin, R. Verkuil, V. Q. Tran, J. Deaton, M. Wiggert, et al. Simulating 500 million years of evolution with a language model. 2024. URL <https://evolutionaryscale-public.s3.us-east-2.amazonaws.com/research/esm3.pdf>. 89
- D. Hendrycks, C. Burns, S. Kadavath, A. Arora, S. Basart, E. Tang, D. Song, and J. Steinhardt. Measuring mathematical problem solving with the MATH dataset. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021. URL <https://openreview.net/forum?id=7Bywt2mQsCe>. 85
- K. M. Hermann, T. Kocisky, E. Grefenstette, L. Espeholt, W. Kay, M. Suleyman, and P. Blunsom. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems*, pages 1693–1701, 2015. 59
- C. Herold, J. Rosendahl, J. Vanvinckenroye, and H. Ney. Detecting various types of noise for neural machine translation. In S. Muresan, P. Nakov, and A. Villavicencio, editors, *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2542–2551, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.findings-acl.200. URL <https://aclanthology.org/2022.findings-acl.200>. 10
- G. Hinton, O. Vinyals, and J. Dean. Distilling the knowledge in a neural network. In *NIPS Deep Learning and Representation Learning Workshop*, 2015. URL <http://arxiv.org/abs/1503.02531>. 47
- S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, Nov. 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.8.1735. URL <https://doi.org/10.1162/neco.1997.9.8.1735>. 123

- A. Holtzman, J. Buys, L. Du, M. Forbes, and Y. Choi. The curious case of neural text degeneration. In *International Conference on Learning Representations*, 2020. 50, 63, 65, 66
- J. Hong, N. Lee, and J. Thorne. Reference-free monolithic preference optimization with odds ratio. *arXiv preprint arXiv:2403.07691*, 2024. 80, 82
- A. Hosseini, X. Yuan, N. Malkin, A. Courville, A. Sordoni, and R. Agarwal. V-STaR: Training verifiers for self-taught reasoners. *arXiv preprint arXiv:2402.06457*, 2024. 72, 87
- L. Hou, R. Y. Pang, T. Zhou, Y. Wu, X. Song, X. Song, and D. Zhou. Token dropping for efficient BERT pretraining. In S. Muresan, P. Nakov, and A. Villavicencio, editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3774–3784, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.262. URL <https://aclanthology.org/2022.acl-long.262>. 121
- J. Huang, X. Chen, S. Mishra, H. S. Zheng, A. W. Yu, X. Song, and D. Zhou. Large language models cannot self-correct reasoning yet. In *The Twelfth International Conference on Learning Representations*, 2024a. URL <https://openreview.net/forum?id=Ikmd3fKBPQ>. 76
- S. Huang, M. Noukhovitch, A. Hosseini, K. Rasul, W. Wang, and L. Tunstall. The N+ implementation details of RLHF with PPO: A case study on tl; dr summarization. *arXiv preprint arXiv:2403.17031*, 2024b. 49
- F. Huszár. How (not) to train your generative model: Scheduled sampling, likelihood, adversary? *arXiv preprint arXiv:1511.05101*, 2015. 9, 11, 50, 51
- R. P. Irvine, D. Boubert, V. Raina, A. Liusie, V. Mudupalli, A. Korshuk, Z. J. Liu, F. Cremer, V. Assassi, C.-C. Beauchamp, X. Lu, T. Rialan, and W. Beauchamp. Rewarding chatbots for real-world engagement with millions of users. *arXiv preprint arXiv:2303.06135*, 2023. 22

- N. Jaques, A. Ghandeharioun, J. H. Shen, C. Ferguson, A. Lapedriza, N. Jones, S. Gu, and R. Picard. Way off-policy batch deep reinforcement learning of implicit human preferences in dialog. *arXiv preprint arXiv:1907.00456*, 2019. 12, 53, 67
- F. Jelinek. *Statistical methods for speech recognition*. MIT press, 1997. 5, 29
- Z. Ji, N. Lee, R. Frieske, T. Yu, D. Su, Y. Xu, E. Ishii, Y. J. Bang, A. Madotto, and P. Fung. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38, 2023. 2
- D. Ju, J. Xu, Y.-L. Boureau, and J. Weston. Learning from data in the mixed adversarial non-adversarial case: Finding the helpers and ignoring the trolls. *arXiv preprint arXiv:2208.03295*, 2022. 20
- D. Kang and T. Hashimoto. Improved natural language generation via loss truncation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 718–731, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.66. URL <https://www.aclweb.org/anthology/2020.acl-main.66>. 67
- C. Kedzie and K. McKeown. A good sample is hard to find: Noise injection sampling and self-training for neural language generation models. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 584–593, 2019. 68
- Y. Keneshloo, T. Shi, N. Ramakrishnan, and C. K. Reddy. Deep reinforcement learning for sequence-to-sequence models. *IEEE Transactions on Neural Networks and Learning Systems*, 2019. 53
- M. D. Kernighan, K. W. Church, and W. A. Gale. A spelling correction program based on a noisy channel model. In *COLING 1990 Volume 2: Papers presented to the 13th International Conference on Computational Linguistics*, 1990. URL <https://aclanthology.org/C90-2036>. 5
- C. Khatri, B. Hedayatnia, A. Venkatesh, J. Nunn, Y. Pan, Q. Liu, H. Song, A. Gottardi, S. Kwatra,

- S. Pancholi, et al. Advancing the state of the art in open domain dialog systems through the Alexa prize. *arXiv preprint arXiv:1812.10757*, 2018. 21
- Y. Kim and A. M. Rush. Sequence-level knowledge distillation. In J. Su, K. Duh, and X. Carreras, editors, *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1317–1327, Austin, Texas, Nov. 2016. Association for Computational Linguistics. doi: 10.18653/v1/D16-1139. URL <https://aclanthology.org/D16-1139>. 33, 47
- D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 41
- H. R. Kirk, A. Whitefield, P. Röttger, A. Bean, K. Margatina, J. Ciro, R. Mosquera, M. Bartolo, A. Williams, H. He, et al. The PRISM alignment project: What participatory, representative and individualised human feedback reveals about the subjective and multicultural alignment of large language models. *arXiv preprint arXiv:2404.16019*, 2024. 10
- T. Kocmi, T. Limisiewicz, and G. Stanovsky. Gender coreference and bias evaluation at WMT 2020. In L. Barrault, O. Bojar, F. Bougares, R. Chatterjee, M. R. Costa-jussà, C. Federmann, M. Fishel, A. Fraser, Y. Graham, P. Guzman, B. Haddow, M. Huck, A. J. Yepes, P. Koehn, A. Martins, M. Morishita, C. Monz, M. Nagata, T. Nakazawa, and M. Negri, editors, *Proceedings of the Fifth Conference on Machine Translation*, pages 357–364, Online, Nov. 2020. Association for Computational Linguistics. URL <https://aclanthology.org/2020.wmt-1>. 39. 120
- 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>. 5
- P. Koehn, F. J. Och, and D. Marcu. Statistical phrase-based translation. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 127–133, 2003. URL <https://aclanthology.org/N03-1017>. 5, 29
- A. Köpf, Y. Kilcher, D. von Rütte, S. Anagnostidis, Z. R. Tam, K. Stevens, A. Barhoum, D. Nguyen, O. Stanley, R. Nagyfi, S. ES, S. Suri, D. Glushkov, A. Dantuluri, A. Maguire, C. Schuhmann, H. Nguyen, and A. Mattick. OpenAssistant conversations – democratizing large language model alignment. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 47669–47681. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/949f0f8f32267d297c2d4e3ee10a2e7e-Paper-Datasets_and_Benchmarks.pdf. 93
- J. Kreutzer, I. Caswell, L. Wang, A. Wahab, D. van Esch, N. Ulzii-Orshikh, A. Tapo, N. Subramani, A. Sokolov, C. Sikasote, M. Setyawan, S. Sarin, S. Samb, B. Sagot, C. Rivera, A. Rios, I. Papadimitriou, S. Osei, P. O. Suarez, I. Orife, K. Ogueji, A. N. Rubungo, T. Q. Nguyen, M. Müller, A. Müller, S. H. Muhammad, N. Muhammad, A. Mnyakeni, J. Mirzakhlov, T. Matangira, C. Leong, N. Lawson, S. Kudugunta, Y. Jernite, M. Jenny, O. Firat, B. F. P. Dossou, S. Dlamini, N. de Silva, S. Çabuk Ballı, S. Biderman, A. Battisti, A. Baruwa, A. Bapna, P. Baljekar, I. A. Azime, A. Awokoya, D. Ataman, O. Ahia, O. Ahia, S. Agrawal, and M. Adeyemi. Quality at a glance: An audit of web-crawled multilingual datasets. *Transactions of the Association for Computational Linguistics*, 10:50–72, 2022. doi: 10.1162/tacl_a_00447. URL <https://aclanthology.org/2022.tacl-1.4>. 10
- I. Kulikov, J. Lee, and K. Cho. Multi-turn beam search for neural dialogue modeling. *arXiv preprint arXiv:1906.00141*, 2019. 22

- N. Lambert, V. Pyatkin, J. Morrison, L. Miranda, B. Y. Lin, K. Chandu, N. Dziri, S. Kumar, T. Zick, Y. Choi, et al. RewardBench: Evaluating reward models for language modeling. *arXiv preprint arXiv:2403.13787*, 2024. 93
- R. Leblond, J.-B. Alayrac, A. Osokin, and S. Lacoste-Julien. SEARNN: Training RNNs with global-local losses. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=HkUR_y-RZ. 67
- J. Lee, E. Mansimov, and K. Cho. Deterministic non-autoregressive neural sequence modeling by iterative refinement. In E. Riloff, D. Chiang, J. Hockenmaier, and J. Tsujii, editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1173–1182, Brussels, Belgium, Oct.-Nov. 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1149. URL <https://aclanthology.org/D18-1149>. 47
- N. Lee, W. Ping, P. Xu, M. Patwary, P. Fung, M. Shoeybi, and B. Catanzaro. Factuality enhanced language models for open-ended text generation. In A. H. Oh, A. Agarwal, D. Belgrave, and K. Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL <https://openreview.net/forum?id=LvyJX20R11>. 24
- S. Levine, A. Kumar, G. Tucker, and J. Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020. 55
- M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Seattle, USA, July 2020. Association for Computational Linguistics. 60, 123
- J. Li, W. Monroe, and D. Jurafsky. Learning to decode for future success. *arXiv preprint arXiv:1701.06549*, 2017a. 21

- J. Li, W. Monroe, T. Shi, S. Jean, A. Ritter, and D. Jurafsky. Adversarial learning for neural dialogue generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2157–2169, Copenhagen, Denmark, Sept. 2017b. Association for Computational Linguistics. doi: 10.18653/v1/D17-1230. URL <https://www.aclweb.org/anthology/D17-1230>. 67
- M. Li, Y. Zhang, Z. Li, J. Chen, L. Chen, N. Cheng, J. Wang, T. Zhou, and J. Xiao. From quantity to quality: Boosting LLM performance with self-guided data selection for instruction tuning. *arXiv preprint arXiv:2308.12032*, 2023. 10
- Y. Li, D. Choi, J. Chung, N. Kushman, J. Schrittwieser, R. Leblond, T. Eccles, J. Keeling, F. Gimeno, A. D. Lago, T. Hubert, P. Choy, C. de Masson d’Autume, I. Babuschkin, X. Chen, P.-S. Huang, J. Welbl, S. Gowal, A. Cherepanov, J. Molloy, D. J. Mankowitz, E. S. Robson, P. Kohli, N. de Freitas, K. Kavukcuoglu, and O. Vinyals. Competition-level code generation with alphacode. *Science*, 378(6624):1092–1097, 2022. doi: 10.1126/science.abq1158. URL <https://www.science.org/doi/abs/10.1126/science.abq1158>. 68
- Z. Li, X. Jiang, L. Shang, and H. Li. Paraphrase generation with deep reinforcement learning. In E. Riloff, D. Chiang, J. Hockenmaier, and J. Tsujii, editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3865–3878, Brussels, Belgium, Oct.-Nov. 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1421. URL <https://aclanthology.org/D18-1421>. 53
- 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>. 59
- Y. Liu, F. Meng, J. Zhou, Y. Chen, and J. Xu. Faster depth-adaptive transformers. *Proceedings*

- of the *AAAI Conference on Artificial Intelligence*, 35(15):13424–13432, May 2021. URL <https://ojs.aaai.org/index.php/AAAI/article/view/17584>. 121
- S. Lu, L. Yu, S. Feng, Y. Zhu, and W. Zhang. CoT: Cooperative training for generative modeling of discrete data. In K. Chaudhuri and R. Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 4164–4172, Long Beach, California, USA, 09–15 Jun 2019. PMLR. URL <http://proceedings.mlr.press/v97/lu19d.html>. 67
- H. Luo, Q. Sun, C. Xu, P. Zhao, J. Lou, C. Tao, X. Geng, Q. Lin, S. Chen, and D. Zhang. WizardMath: Empowering mathematical reasoning for large language models via reinforced evol-instruct. *arXiv preprint arXiv:2308.09583*, 2023a. 82
- L. Luo, Z. Lin, Y. Liu, L. Shu, Y. Zhu, J. Shang, and L. Meng. Critique ability of large language models. *arXiv preprint arXiv:2310.04815*, 2023b. 93
- S. Mahadevan. Average reward reinforcement learning: Foundations, algorithms, and empirical results. *Machine learning*, 22(1):159–195, 1996. 47
- Meta. Introducing Meta Llama 3: The most capable openly available llm to date. <https://ai.meta.com/blog/meta-llama-3/>, 2024. 15
- J. Michael, S. Mahdi, D. Rein, J. Petty, J. Dirani, V. Padmakumar, and S. R. Bowman. Debate helps supervise unreliable experts. *arXiv preprint arXiv:2311.08702*, 2023. 94, 95
- M. Minsky. Steps toward artificial intelligence. *Proceedings of the IRE*, 49(1):8–30, 1961. 3
- V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015. 14, 36, 37

- K. Murray and D. Chiang. Correcting length bias in neural machine translation. In O. Bojar, R. Chatterjee, C. Federmann, M. Fishel, Y. Graham, B. Haddow, M. Huck, A. J. Yepes, P. Koehn, C. Monz, M. Negri, A. N  v  ol, M. Neves, M. Post, L. Specia, M. Turchi, and K. Verspoor, editors, *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 212–223, Brussels, Belgium, Oct. 2018. Association for Computational Linguistics. doi: 10.18653/v1/W18-6322. URL <https://aclanthology.org/W18-6322>. 64
- Y. Nakatani, T. Kajiwara, and T. Ninomiya. Comparing BERT-based reward functions for deep reinforcement learning in machine translation. In *Proceedings of the 9th Workshop on Asian Translation*, pages 37–43, Gyeongju, Republic of Korea, Oct. 2022. International Conference on Computational Linguistics. URL <https://aclanthology.org/2022.wat-1.2>. 16
- S. Narayan, S. B. Cohen, and M. Lapata. Don’t give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In E. Riloff, D. Chiang, J. Hockenmaier, and J. Tsujii, editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1797–1807, Brussels, Belgium, Oct.-Nov. 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1206. URL <https://aclanthology.org/D18-1206>. 59, 122
- A. Y. Ng, D. Harada, and S. Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *Proceedings of the 16th International Conference on Machine Learning*, volume 99, pages 278–287, 1999. 3
- N. Ng, K. Yee, A. Baevski, M. Ott, M. Auli, and S. Edunov. Facebook FAIR’s WMT19 news translation task submission. 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. N  v  ol, M. Neves, M. Post, M. Turchi, and K. Verspoor, editors, *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 314–319, Florence, Italy, Aug. 2019. Association for

- Computational Linguistics. doi: 10.18653/v1/W19-5333. URL <https://aclanthology.org/W19-5333>. 5, 30, 32
- M. Norouzi, S. Bengio, z. Chen, N. Jaitly, M. Schuster, Y. Wu, and D. Schuurmans. Reward augmented maximum likelihood for neural structured prediction. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 29, pages 1723–1731. Curran Associates, Inc., 2016. URL <https://proceedings.neurips.cc/paper/2016/file/2f885d0fbe2e131bfc9d98363e55d1d4-Paper.pdf>. 48, 67
- J. Novikova, O. Dušek, A. Cercas Curry, and V. Rieser. Why we need new evaluation metrics for NLG. In M. Palmer, R. Hwa, and S. Riedel, editors, *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2241–2252, Copenhagen, Denmark, Sept. 2017. Association for Computational Linguistics. doi: 10.18653/v1/D17-1238. URL <https://aclanthology.org/D17-1238>. 53
- H. L. O’Brien and E. G. Toms. What is user engagement? A conceptual framework for defining user engagement with technology. *Journal of the American society for Information Science and Technology*, 59(6):938–955, 2008. 23
- OpenAI. Proximal policy optimization. <https://spinningup.openai.com/en/latest/algorithms/ppo.html>, 2024. 13
- M. Ott, S. Edunov, A. Baevski, A. Fan, S. Gross, N. Ng, D. Grangier, and M. Auli. fairseq: A fast, extensible toolkit for sequence modeling. In W. Ammar, A. Louis, and N. Mostafazadeh, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-4009. URL <https://aclanthology.org/N19-4009>. 41

- L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, J. Schulman, J. Hilton, F. Kelton, L. Miller, M. Simens, A. Askell, P. Welinder, P. F. Christiano, J. Leike, and R. Lowe. Training language models to follow instructions with human feedback. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 27730–27744. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/b1efde53be364a73914f58805a001731-Paper-Conference.pdf. 1, 91, 92
- A. Pal, D. Karkhanis, S. Dooley, M. Roberts, S. Naidu, and C. White. Smaug: Fixing failure modes of preference optimisation with DPO-positive. *arXiv preprint arXiv:2402.13228*, 2024. 82
- R. Y. Pang and H. He. Text generation by learning from demonstrations. In *International Conference on Learning Representations*, 2021. URL <https://openreview.net/forum?id=RovX-uQ1Hua>. 7, 9, 48, 49
- R. Y. Pang, H. He, and K. Cho. Amortized noisy channel neural machine translation. In S. Shaikh, T. Ferreira, and A. Stent, editors, *Proceedings of the 15th International Conference on Natural Language Generation*, pages 131–143, Waterville, Maine, USA and virtual meeting, July 2022a. Association for Computational Linguistics. doi: 10.18653/v1/2022.inlg-main.11. URL <https://aclanthology.org/2022.inlg-main.11>. 6, 16, 24, 29
- R. Y. Pang, A. Parrish, N. Joshi, N. Nangia, J. Phang, A. Chen, V. Padmakumar, J. Ma, J. Thompson, H. He, and S. Bowman. QuALITY: Question answering with long input texts, yes! In M. Carpuat, M.-C. de Marneffe, and I. V. Meza Ruiz, editors, *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5336–5358, Seattle, United States, July 2022b. Association for Computational Linguistics. doi: 10.18653/v1/2022.naacl-main.391. URL <https://aclanthology.org/2022.naacl-main.391>. 94

- R. Y. Pang, V. Padmakumar, T. Sellam, A. Parikh, and H. He. Reward gaming in conditional text generation. In A. Rogers, J. Boyd-Graber, and N. Okazaki, editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4746–4763, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.262. URL <https://aclanthology.org/2023.acl-long.262>. 3, 12, 16, 28
- R. Y. Pang, S. Roller, K. Cho, H. He, and J. Weston. Leveraging implicit feedback from deployment data in dialogue. In Y. Graham and M. Purver, editors, *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 60–75, St. Julian’s, Malta, Mar. 2024a. Association for Computational Linguistics. URL <https://aclanthology.org/2024.eacl-short.8>. 5, 19
- R. Y. Pang, W. Yuan, K. Cho, H. He, S. Sukhbaatar, and J. Weston. Iterative reasoning preference optimization. *arXiv preprint arXiv:2404.19733*, 2024b. 7, 71
- 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>. 50, 59
- T. Parshakova, J.-M. Andreoli, and M. Dymetman. Distributional reinforcement learning for energy-based sequential models. *arXiv preprint arXiv:1912.08517*, 2019. 50
- R. Pasunuru and M. Bansal. Multi-reward reinforced summarization with saliency and entailment. In M. Walker, H. Ji, and A. Stent, editors, *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2*

- (*Short Papers*), pages 646–653, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-2102. URL <https://aclanthology.org/N18-2102>. 9, 61, 67
- M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer. Deep contextualized word representations. In M. Walker, H. Ji, and A. Stent, editors, *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1202. URL <https://aclanthology.org/N18-1202>. 1
- M. Post. A call for clarity in reporting BLEU scores. In O. Bojar, R. Chatterjee, C. Federmann, M. Fishel, Y. Graham, B. Haddow, M. Huck, A. J. Yepes, P. Koehn, C. Monz, M. Negri, A. Névél, M. Neves, M. Post, L. Specia, M. Turchi, and K. Verspoor, editors, *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium, Oct. 2018. Association for Computational Linguistics. doi: 10.18653/v1/W18-6319. URL <https://aclanthology.org/W18-6319>. 40
- D. Precup, R. S. Sutton, and S. Singh. Eligibility traces for off-policy policy evaluation. *Computer Science Department Faculty Publication Series, UMass Amherst*, 2000. 54
- W. Qi, Y. Yan, Y. Gong, D. Liu, N. Duan, J. Chen, R. Zhang, and M. Zhou. ProphetNet: Predicting future n-gram for sequence-to-SequencePre-training. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2401–2410, Online, Nov. 2020. Association for Computational Linguistics. URL <https://www.aclweb.org/anthology/2020.findings-emnlp.217>. 128
- 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. 92

- R. Rafailov, A. Sharma, E. Mitchell, C. D. Manning, S. Ermon, and C. Finn. Direct preference optimization: Your language model is secretly a reward model. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL <https://openreview.net/forum?id=HPuSIXJaa9>. 7, 18, 71, 75, 82, 86, 91
- R. Rafailov, J. Hejna, R. Park, and C. Finn. From r to Q^* : Your language model is secretly a Q -function. *arXiv preprint arXiv:2404.12358*, 2024. 82
- C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020. 128
- P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang. SQuAD: 100,000+ questions for machine comprehension of text. In J. Su, K. Duh, and X. Carreras, editors, *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas, Nov. 2016. Association for Computational Linguistics. doi: 10.18653/v1/D16-1264. URL <https://aclanthology.org/D16-1264>. 122
- 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 *International Conference on Learning Representations*, 2023. URL <https://openreview.net/forum?id=8aHzds2uUyB>. 12, 16
- M. Ranzato, S. Chopra, M. Auli, and W. Zaremba. Sequence level training with recurrent neural networks. In *International Conference on Learning Representations*, 2016. 9, 47, 50, 67, 127
- D. Rein, B. L. Hou, A. C. Stickland, J. Petty, R. Y. Pang, J. Dirani, J. Michael, and S. R. Bowman. GPQA: A graduate-level google-proof q&a benchmark. *arXiv preprint arXiv:2311.12022*, 2023. 94

- E. Reiter. An architecture for data-to-text systems. In S. Busemann, editor, *Proceedings of the Eleventh European Workshop on Natural Language Generation (ENLG 07)*, pages 97–104, Saarbrücken, Germany, June 2007. DFKI GmbH. URL <https://aclanthology.org/W07-2315>. 1
- E. Reiter and R. Dale. Building applied natural language generation systems. *Natural Language Engineering*, 3(1):57–87, 1997. 1
- S. Ross, G. Gordon, and D. Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pages 627–635, 2011. 50, 67
- C. Rosset, C.-A. Cheng, A. Mitra, M. Santacroce, A. Awadallah, and T. Xie. Direct Nash optimization: Teaching language models to self-improve with general preferences. *arXiv preprint arXiv:2404.03715*, 2024. 72, 87
- V. Sanh, L. Debut, J. Chaumond, and T. Wolf. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019. 23
- F. Schmidt. Generalization in generation: A closer look at exposure bias. In A. Birch, A. Finch, H. Hayashi, I. Konstas, T. Luong, G. Neubig, Y. Oda, and K. Sudoh, editors, *Proceedings of the 3rd Workshop on Neural Generation and Translation*, pages 157–167, Hong Kong, Nov. 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-5616. URL <https://aclanthology.org/D19-5616>. 9
- J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. 13, 86, 91
- T. Scialom, S. Lamprier, B. Piwowarski, and J. Staiano. Answers unite! unsupervised metrics for reinforced summarization models. In K. Inui, J. Jiang, V. Ng, and X. Wan, editors, *Proceedings of*

- the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3246–3256, Hong Kong, China, Nov. 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1320. URL <https://aclanthology.org/D19-1320>. 61
- A. See, P. J. Liu, and C. D. Manning. Get to the point: Summarization with pointer-generator networks. In R. Barzilay and M.-Y. Kan, editors, *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, Vancouver, Canada, July 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1099. URL <https://aclanthology.org/P17-1099>. 59, 122, 123
- T. Sellam, D. Das, and A. Parikh. BLEURT: Learning robust metrics for text generation. In D. Jurafsky, J. Chai, N. Schlueter, and J. Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7881–7892, Online, July 2020a. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.704. URL <https://aclanthology.org/2020.acl-main.704>. 40
- T. Sellam, A. Pu, H. W. Chung, S. Gehrmann, Q. Tan, M. Freitag, D. Das, and A. Parikh. Learning to evaluate translation beyond English: BLEURT submissions to the WMT metrics 2020 shared task. In L. Barrault, O. Bojar, F. Bougares, R. Chatterjee, M. R. Costa-jussà, C. Federmann, M. Fishel, A. Fraser, Y. Graham, P. Guzman, B. Haddow, M. Huck, A. J. Yepes, P. Koehn, A. Martins, M. Morishita, C. Monz, M. Nagata, T. Nakazawa, and M. Negri, editors, *Proceedings of the Fifth Conference on Machine Translation*, pages 921–927, Online, Nov. 2020b. Association for Computational Linguistics. URL <https://aclanthology.org/2020.wmt-1.102>. 16
- I. V. Serban, C. Sankar, M. Germain, S. Zhang, Z. Lin, S. Subramanian, T. Kim, M. Pieper, S. Chandar, N. R. Ke, et al. A deep reinforcement learning chatbot. *arXiv preprint arXiv:1709.02349*, 2017. 48, 53, 55, 67

- A. Setlur, S. Garg, X. Geng, N. Garg, V. Smith, and A. Kumar. RL on incorrect synthetic data scales the efficiency of LLM math reasoning by eight-fold. *arXiv preprint arXiv:2406.14532*, 2024. 89
- C. E. Shannon. A mathematical theory of communication. *The Bell system technical journal*, 27(3): 379–423, 1948. 5
- S. She, S. Huang, W. Zou, W. Zhu, X. Liu, X. Geng, and J. Chen. MAPO: Advancing multilingual reasoning through multilingual alignment-as-preference optimization. *arXiv preprint arXiv:2401.06838*, 2024. 87
- S. Shen, Y. Cheng, Z. He, W. He, H. Wu, M. Sun, and Y. Liu. Minimum risk training for neural machine translation. In K. Erk and N. A. Smith, editors, *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1683–1692, Berlin, Germany, Aug. 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1159. URL <https://aclanthology.org/P16-1159>. 67
- K. Shuster, M. Komeili, L. Adolphs, S. Roller, A. Szlam, and J. Weston. Language models that seek for knowledge: Modular search & generation for dialogue and prompt completion. In Y. Goldberg, Z. Kozareva, and Y. Zhang, editors, *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 373–393, Abu Dhabi, United Arab Emirates, Dec. 2022a. Association for Computational Linguistics. doi: 10.18653/v1/2022.findings-emnlp.27. URL <https://aclanthology.org/2022.findings-emnlp.27>. 24, 97
- K. Shuster, J. Xu, M. Komeili, D. Ju, E. M. Smith, S. Roller, M. Ung, M. Chen, K. Arora, J. Lane, et al. BlenderBot 3: a deployed conversational agent that continually learns to responsibly engage. *arXiv preprint arXiv:2208.03188*, 2022b. 21, 24
- L. Simon, R. Webster, and J. Rabin. Revisiting precision recall definition for generative modeling. In K. Chaudhuri and R. Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 5799–5808,

- Long Beach, California, USA, 09–15 Jun 2019. PMLR. URL <http://proceedings.mlr.press/v97/simon19a.html>. 11, 50
- A. Singh, J. D. Co-Reyes, R. Agarwal, A. Anand, P. Patil, P. J. Liu, J. Harrison, J. Lee, K. Xu, A. Parisi, et al. Beyond human data: Scaling self-training for problem-solving with language models. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL <https://openreview.net/forum?id=1NAyUngGFK>. Expert Certification. 72, 87
- J. M. V. Skalse, N. H. R. Howe, D. Krasheninnikov, and D. Krueger. Defining and characterizing reward gaming. In A. H. Oh, A. Agarwal, D. Belgrave, and K. Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL <https://openreview.net/forum?id=yb3H0X031X2>. 3, 12, 28
- Snorkel AI Team. Snorkel-mistral-pairrm-dpo. <https://huggingface.co/snorkelai/Snorkel-Mistral-PairRM-DPO>, 2023. Accessed: 2024-04-15. 87
- J. Sorg, R. L. Lewis, and S. Singh. Reward design via online gradient ascent. *Advances in Neural Information Processing Systems*, 23, 2010. 3
- J. Sorg, S. Singh, and R. L. Lewis. Variance-based rewards for approximate bayesian reinforcement learning. *arXiv preprint arXiv:1203.3518*, 2012. 3
- F. Stahlberg and B. Byrne. On NMT search errors and model errors: Cat got your tongue? In K. Inui, J. Jiang, V. Ng, and X. Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3356–3362, Hong Kong, China, Nov. 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1331. URL <https://aclanthology.org/D19-1331>. 64, 66
- 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. In H. Larochelle, M. Ranzato, R. Hadsell,

- M. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 3008–3021. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/1f89885d556929e98d3ef9b86448f951-Paper.pdf. 9, 12, 16, 19, 71
- R. S. Sutton. Integrated architectures for learning, planning, and reacting based on approximating dynamic programming. In *Machine learning proceedings 1990*, pages 216–224. Elsevier, 1990. 38
- R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*. MIT press, 1998. 2, 14, 35, 36, 47
- R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*. MIT press, 2018. 2, 3
- R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour. Policy gradient methods for reinforcement learning with function approximation. *Advances in Neural Information Processing Systems*, 12, 1999. 12
- R. S. Sutton, D. A. McAllester, S. P. Singh, and Y. Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, pages 1057–1063, 2000. 52
- X. Tan, Y. Ren, D. He, T. Qin, and T.-Y. Liu. Multilingual neural machine translation with knowledge distillation. In *International Conference on Learning Representations*, 2019. URL <https://openreview.net/forum?id=S1gUsoR9YX>. 33, 47
- G. Tevet, G. Habib, V. Shwartz, and J. Berant. Evaluating text GANs as language models. In J. Burstein, C. Doran, and T. Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2241–2247, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1233. URL <https://aclanthology.org/N19-1233>. 67

- R. Tian, S. Narayan, T. Sellam, and A. P. Parikh. Sticking to the facts: Confident decoding for faithful data-to-text generation, 2020. URL <https://openreview.net/forum?id=HkxU2pNYPH>. 68
- S. Toshniwal, I. Moshkov, S. Narenthiran, D. Gitman, F. Jia, and I. Gitman. OpenMathInstruct-1: A 1.8 million math instruction tuning dataset. *arXiv preprint arXiv:2402.10176*, 2024. 87
- 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. 69
- 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. 1, 77, 82, 86, 93
- C. Tran, S. Bhosale, J. Cross, P. Koehn, S. Edunov, and A. Fan. Facebook AI’s WMT21 news translation task submission. 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 205–215, Online, Nov. 2021. Association for Computational Linguistics. URL <https://aclanthology.org/2021.wmt-1.19>. 5, 30, 32
- L. Tu, R. Y. Pang, and K. Gimpel. Improving joint training of inference networks and structured prediction energy networks. In *Proceedings of the Fourth Workshop on Structured Prediction for NLP*, pages 62–73, Online, Nov. 2020a. Association for Computational Linguistics. URL <https://www.aclweb.org/anthology/2020.spnlp-1.8>. 67
- L. Tu, R. Y. Pang, S. Wiseman, and K. Gimpel. ENGINE: Energy-based inference networks for non-autoregressive machine translation. In D. Jurafsky, J. Chai, N. Schluter, and J. Tetreault,

- editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2819–2826, Online, July 2020b. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.251. URL <https://aclanthology.org/2020.acl-main.251>. 33, 34, 47
- K. Valmeekam, M. Marquez, and S. Kambhampati. Can large language models really improve by self-critiquing their own plans? *arXiv preprint arXiv:2310.08118*, 2023. 93
- A. van den Oord, Y. Li, I. Babuschkin, K. Simonyan, O. Vinyals, K. Kavukcuoglu, G. van den Driessche, E. Lockhart, L. Cobo, F. Stimberg, N. Casagrande, D. Grewe, S. Noury, S. Dieleman, E. Elsen, N. Kalchbrenner, H. Zen, A. Graves, H. King, T. Walters, D. Belov, and D. Hassabis. Parallel WaveNet: Fast high-fidelity speech synthesis. In J. Dy and A. Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 3918–3926. PMLR, 10–15 Jul 2018. URL <https://proceedings.mlr.press/v80/oord18a.html>. 47
- A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008, 2017. 60
- R. Vedantam, C. Lawrence Zitnick, and D. Parikh. Cider: Consensus-based image description evaluation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015. 70
- V. Veselovsky, M. H. Ribeiro, and R. West. Artificial artificial artificial intelligence: Crowd workers widely use large language models for text production tasks. *arXiv preprint arXiv:2306.07899*, 2023. 101
- C. Wang and R. Sennrich. On exposure bias, hallucination and domain shift in neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3544–3552, Online, July 2020. Association for Computational Linguistics. doi:

- 10.18653/v1/2020.acl-main.326. URL <https://www.aclweb.org/anthology/2020.acl-main.326>. 50
- C. J. Watkins and P. Dayan. Q-learning. *Machine learning*, 8(3-4):279–292, 1992. 35
- S. Welleck, I. Kulikov, J. Kim, R. Y. Pang, and K. Cho. Consistency of a recurrent language model with respect to incomplete decoding. 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 5553–5568, Online, Nov. 2020a. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.448. URL <https://aclanthology.org/2020.emnlp-main.448>. 43
- S. Welleck, I. Kulikov, J. Kim, R. Y. Pang, and K. Cho. Consistency of a recurrent language model with respect to incomplete decoding. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5553–5568, Online, Nov. 2020b. Association for Computational Linguistics. URL <https://www.aclweb.org/anthology/2020.emnlp-main.448>. 63
- S. Welleck, I. Kulikov, S. Roller, E. Dinan, K. Cho, and J. Weston. Neural text generation with unlikelihood training. In *International Conference on Learning Representations*, 2020c. URL <https://openreview.net/forum?id=SJeYe0NtvH>. 17, 68
- F. Wilcoxon. Individual comparisons by ranking methods. In *Breakthroughs in Statistics: Methodology and Distribution*, pages 196–202. Springer, 1992. 100
- R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992. 12, 52
- S. Wiseman and A. M. Rush. Sequence-to-sequence learning as beam-search optimization. In J. Su, K. Duh, and X. Carreras, editors, *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1296–1306, Austin, Texas, Nov. 2016. Association for

Computational Linguistics. doi: 10.18653/v1/D16-1137. URL <https://aclanthology.org/D16-1137>. 67

D. Wu, J. Zhang, and X. Huang. Chain of thought prompting elicits knowledge augmentation. In A. Rogers, J. Boyd-Graber, and N. Okazaki, editors, *Findings of the Association for Computational Linguistics: ACL 2023*, pages 6519–6534, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-acl.408. URL <https://aclanthology.org/2023.findings-acl.408>. 72

J. Wu, L. Ouyang, D. M. Ziegler, N. Stiennon, R. Lowe, J. Leike, and P. Christiano. Recursively summarizing books with human feedback. *arXiv preprint arXiv:2109.10862*, 2021. 9

L. Wu, F. Tian, T. Qin, J. Lai, and T.-Y. Liu. A study of reinforcement learning for neural machine translation. In E. Riloff, D. Chiang, J. Hockenmaier, and J. Tsujii, editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3612–3621, Brussels, Belgium, Oct.-Nov. 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1397. URL <https://aclanthology.org/D18-1397>. 50, 53

W. Xiong, H. Dong, C. Ye, H. Zhong, N. Jiang, and T. Zhang. Gibbs sampling from human feedback: A provable KL-constrained framework for RLHF. *arXiv preprint arXiv:2312.11456*, 2023. 71, 86

H. Xu, A. Sharaf, Y. Chen, W. Tan, L. Shen, B. Van Durme, K. Murray, and Y. J. Kim. Contrastive preference optimization: Pushing the boundaries of LLM performance in machine translation. *arXiv preprint arXiv:2401.08417*, 2024. 82

J. Xu, D. Ju, J. Lane, M. Komeili, E. M. Smith, M. Ung, M. Behrooz, W. Ngan, R. Moritz, S. Sukhbaatar, et al. Improving open language models by learning from organic interactions. *arXiv preprint arXiv:2306.04707*, 2023a. 5, 19, 21, 24, 97

J. Xu, A. Lee, S. Sukhbaatar, and J. Weston. Some things are more CRINGE than others: Preference

optimization with the pairwise CRINGE loss. *arXiv preprint arXiv:2312.16682*, 2023b. 17, 71, 86, 92

K. Yee, Y. Dauphin, and M. Auli. Simple and effective noisy channel modeling for neural machine translation. In K. Inui, J. Jiang, V. Ng, and X. Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5696–5701, Hong Kong, China, Nov. 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1571. URL <https://aclanthology.org/D19-1571>. 5, 30, 32

L. Yu, P. Blunsom, C. Dyer, E. Grefenstette, and T. Kocisky. The neural noisy channel. In *International Conference on Learning Representations*, 2017a. URL <https://openreview.net/forum?id=SJ25-B5eg>. 32

L. Yu, W. Zhang, J. Wang, and Y. Yu. Seqgan: Sequence generative adversarial nets with policy gradient. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017b. 67

L. Yu, L. Sartran, P.-S. Huang, W. Stokowiec, D. Donato, S. Srinivasan, A. Andreev, W. Ling, S. Mokra, A. Dal Lago, Y. Doron, S. Young, P. Blunsom, and C. Dyer. The DeepMind Chinese–English document translation system at WMT2020. In L. Barrault, O. Bojar, F. Bougares, R. Chatterjee, M. R. Costa-jussà, C. Federmann, M. Fishel, A. Fraser, Y. Graham, P. Guzman, B. Haddow, M. Huck, A. J. Yepes, P. Koehn, A. Martins, M. Morishita, C. Monz, M. Nagata, T. Nakazawa, and M. Negri, editors, *Proceedings of the Fifth Conference on Machine Translation*, pages 326–337, Online, Nov. 2020. Association for Computational Linguistics. URL <https://aclanthology.org/2020.wmt-1.36>. 5, 30, 32

L. Yu, W. Jiang, H. Shi, J. YU, Z. Liu, Y. Zhang, J. Kwok, Z. Li, A. Weller, and W. Liu. Metamath: Bootstrap your own mathematical questions for large language models. In *The Twelfth Interna-*

- tional Conference on Learning Representations*, 2024. URL <https://openreview.net/forum?id=N8N0hgNDRt>. 82, 87
- W. Yuan, R. Y. Pang, K. Cho, S. Sukhbaatar, J. Xu, and J. Weston. Self-rewarding language models. *arXiv preprint arXiv:2401.10020*, 2024. 7, 71, 76, 87
- E. Zelikman, Y. Wu, J. Mu, and N. Goodman. STaR: Bootstrapping reasoning with reasoning. In A. H. Oh, A. Agarwal, D. Belgrave, and K. Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=_3ELRdg2sgI. 72, 76, 87
- Y. Zemlyanskiy and F. Sha. Aiming to know you better perhaps makes me a more engaging dialogue partner. In A. Korhonen and I. Titov, editors, *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 551–561, Brussels, Belgium, Oct. 2018. Association for Computational Linguistics. doi: 10.18653/v1/K18-1053. URL <https://aclanthology.org/K18-1053>. 21
- J. Zhang, Y. Zhao, M. Saleh, and P. J. Liu. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In *Proceedings of the 37th International Conference on Machine Learning*, Proceedings of Machine Learning Research. PMLR, 2020a. 128
- Y. Zhang, S. Sun, M. Galley, Y.-C. Chen, C. Brockett, X. Gao, J. Gao, J. Liu, and B. Dolan. DIALOGPT : Large-scale generative pre-training for conversational response generation. In A. Celikyilmaz and T.-H. Wen, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 270–278, Online, July 2020b. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-demos.30. URL <https://aclanthology.org/2020.acl-demos.30>. 21
- L. Zheng, W.-L. Chiang, Y. Sheng, S. Zhuang, Z. Wu, Y. Zhuang, Z. Lin, Z. Li, D. Li, E. Xing, H. Zhang, J. E. Gonzalez, and I. Stoica. Judging LLM-as-a-judge with MT-bench and chatbot

- arena. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023. URL <https://openreview.net/forum?id=uccHPGDlao>. 91
- L. Zhou, K. Small, O. Rokhlenko, and C. Elkan. End-to-end offline goal-oriented dialog policy learning via policy gradient. *arXiv preprint arXiv:1712.02838*, 2017a. 48
- Q. Zhou, N. Yang, F. Wei, C. Tan, H. Bao, and M. Zhou. Neural question generation from text: A preliminary study. In *National CCF Conference on Natural Language Processing and Chinese Computing*, pages 662–671. Springer, 2017b. 58, 59, 122, 123
- 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. 61, 64, 71, 91
- V. Zouhar, T. Kocmi, and M. Sachan. AI-assisted human evaluation of machine translation. *arXiv preprint arXiv:2406.12419*, 2024. 95