MODEL ADAPTATION WITH APPLICATIONS IN ADVERSARIAL ROBUSTNESS AND LARGE LANGUAGE MODELS

by

Yang Guo

A dissertation submitted in partial fulfillment of
the requirements for the degree of

Doctor of Philosophy

(Computer Sciences)

at the

UNIVERSITY OF WISCONSIN–MADISON

2025

Date of final oral examination: 06/19/2025

The dissertation is approved by the following members of the Final Oral Committee:
Yingyu Liang (Advisor), Associate Professor, Computer Sciences
Robert Nowak (Advisor), Professor, Computer Sciences
Frederic Sala, Assistant Professor, Computer Sciences
Kangwook Lee, Assistant Professor, Computer Sciences

To my grandmother, parents, and friends.

ACKNOWLEDGMENTS

I am deeply grateful to my parents for nurturing my childhood with curiosity, laughter, and an unwavering belief in my potential. Their constant support has carried me through every high and low of graduate school. I also hold close the memory of my grandmother, who used to wait with me for the bus to weekend math classes during elementary school and always picked up my late-night calls when I was struggling in graduate school. Her quiet presence was a source of strength throughout my life. She passed away during the COVID-19 pandemic. I hope this dissertation makes her proud.

I extend my deepest gratitude to my advisor, Prof. Yingyu Liang. I am especially thankful for his patience in the early stages of my Ph.D., when I was still developing my research habits and often struggled with focus and precision. He was always willing to answer even my most naive questions about research or methodology with honesty and patience, and consistently offered high-level insights that helped me see the bigger picture. He supported me through both the highs and lows of my projects, including the particularly difficult period during the COVID-19 pandemic.

I am equally grateful to my other advisor, Prof. Robert (Rob) Nowak. His support during the most challenging periods of my Ph.D. meant a great deal. His insights consistently pushed me to think from the first principle, and to question the motivation and novelty of my ideas. I also thank him for encouraging clear thinking and effective communication—both in research summaries and in planning. Through organizing SILO and attending its seminars, I had the valuable opportunity to learn from diverse researchers and engage with the broader research community.

I extend my heartfelt appreciation to my committee members, Prof. Frederic (Fred) Sala and Prof. Kangwook Lee. Fred's course on large language models cut through the hype and helped me develop a sharper intuition for what truly matters in research. His feedback during my prelims was both rigorous and deeply motivating. Kangwook's incisive suggestions, often shared in passing hallway conversations or SILO discussions, consistently pushed my work toward greater

clarity and impact.

Doing research alone can often feel isolating, and it meant a lot to be surrounded by others who shared a similar passion and curiosity. I am grateful to the members of Yingyu's group—Zhenmei Shi, Jiefeng Chen, Junyi Wei, and Nils Palumbo—and Rob's group—Jifan Zhang, Liu Yang, Haoyue Bai, Joseph Shenouda, Gokcan Tatli, and Subhojyoti Mukherjee—for their invaluable camaraderie and the many small but meaningful research conversations we shared. Collaborating with researchers outside our group has also been deeply rewarding. Working with Yifei Ming, Xi Wu, Somesh Jha, Andrew Wagenmaker, Robert Mankoff, Lalit Jain, and Kevin Jamieson broadened my perspective and pushed my thinking in new directions. Mentoring undergraduate researchers—Jiayi Chen, Kuan Zhou, and Tianqi Li—was a joy, and their fresh curiosity reminded me of the excitement that drew me to research in the first place.

I'm thankful to my mentors from the Amazon Smart Home Machine Learning team—Michael Dillon and Kathleen Champion—for giving me the opportunity to gain hands-on experience for improving the trustworthiness of smart home routines. I was also fortunate to be hosted by Bolin Ding, with whom I had a great time working on agent design in auction systems. All of them were based in Seattle, which became a special place for me. It was heartening to reconnect with old friends and meet new ones. I am especially thankful to Xin Lin, Tongtong Lian, Austin Liu, Mingxuan Zha, Jiayi Wang, Yue Gao, Ruixue Lian, Jingcheng Xu, and Allison, whose presence gave the feeling of home. I still remember the thrill of catching a 14-pound king salmon in Grays Harbor, driving straight back, and sharing fresh sashimi with a group of friends.

The long journey of the Ph.D. was made lighter by the steady encouragement and support of friends. I'm especially grateful to Xingjian Wu, my closest friend since elementary and high school, who has always stood by me. Whenever I felt overwhelmed, he welcomed me to visit, offering comfort and a fresh perspective when I needed it most. I also want to thank Yi Wang, my high school friend who was also working diligently towards his Ph.D. Every visit to Chicago was made memorable by his warmth and his determination to take me to the most authentic

Chinese food spots.

Thanks to our StudentSay group — Xiating Ouyang, Elvis Chang, Kaiyang Chen, Justin LiXie, Holdson Liang, Eric Lin, Jifan Zhang, and Xingjian Zhen — for making grad life more fun and less lonely. Saturday Szechuan nights, outdoor barbeques, and intense badminton matches were the real highlights of the week.

One special trip deserves a shout-out: biking 77 miles to Devil's Lake with Yiding Chen — thanks for riding every mile of it with me.

I would also like to thank many other friends who supported me throughout my Ph.D. Hantao Hu, one of my closest friends since elementary school, has always been there. I'm grateful to my undergrad friends — Jiahao Zhang, Ziyi Tang, Lini Tan, Tongcheng Li, Yifei Wu, Jialu Bao, Yu Sun, and Ziyuan Qin — for their continued support and life updates. Thanks to Hangdong Zhao, Changyu Gao, Jiayu Wang, Mu Cai, Xiaomin Zhang, Yuzhe Ma, Hongyi Wang, Kexin Li, Yutian Tao, Zhennan Wu, Ting Cai, and Jing Liu for bringing encouragement and joy during graduate school. Lastly, thanks to my roommates Haowen Hu, Yunjia Zhang, and Haochen Zhai for making everyday life easier and more grounded.

Gratitude goes to every professor, collaborator, lab-mate, friend, and family member who shaped this journey. Their guidance, encouragement, and presence — in both pivotal moments and quiet support — made all the difference.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

The key to success in machine learning lies in generalization. Under the classical statistical perspective, generalization refers to the model's ability to perform well on unseen test data drawn from the same distribution as the training data. Within this traditional framework, the upper bound of achievable performance is dictated by inherent label noise in the data. Over the past two decades, substantial advances have been made in classical generalization (inductive learning), demonstrated by remarkable improvements on canonical benchmarks such as MNIST (LeCun, 1998), CIFAR-10 Krizhevsky et al. (2009), and ImageNet (ILSVRC) (Deng et al., 2009). As these benchmarks approach their performance ceilings, researchers have started to reconsider and relax the strict assumption that training and test distributions match exactly, recognizing that the test distribution is rarely perfectly observable in practice. A natural and effective strategy to handle this uncertainty is to adapt the model based on available unlabeled test data at inference time. Model adaptation thus emerges as a powerful and versatile approach applicable across various machine learning paradigms—from traditional scenarios like test-time training and domain adaptation, to modern settings involving large language models (LLMs), including fine-tuning and retrieval-augmented generation. Model adaptation not only improves performance but also enables greater customization. In this thesis, I will investigate two critical aspects of model adaptation: its robustness properties and its applications on large language models.

This doctoral thesis advances the study of trustworthy machine learning (Part I) and foundation models (Part II) from the angle of model adaptation. Specifically, Part I begins in Chapter 2 by clarifying the appropriate notion of robustness for models under adaptation and introduces an effective adversarial attack framework. Chapter 3 extends this investigation by examining how model performance can be improved within this relaxed robustness framework, proposing rejection as a complementary approach. Transitioning from robustness considerations to application-driven adaptations, Part II explores model adaptation in large language models (LLMs). Adaptation in LLMs manifests in two primary forms: *across-context*

adaptation (fine-tuning), which efficiently customizes models without extensive retraining, and *within-context* adaptation (in-context learning), a key factor in the empirical success of LLMs. Chapter 4 illustrates across-context adaptation through finetuning, specifically addressing creative generation tasks. Finally, Chapter 5 investigates within-context adaptation by analyzing retrieval-augmented generation (RAG), conceptualizing it as query-dependent noisy in-context learning that generalizes both classical in-context learning and standard RAG methods.

1 INTRODUCTION

In recent years, machine learning has become a cornerstone of technological advancement, powering a wide array of applications from computer vision (Radford et al., 2021; Redmon et al., 2016; He et al., 2017) to natural language processing (Achiam et al., 2023; Vaswani et al., 2017; Devlin et al., 2018). Given the remarkable success of machine learning and deep learning in these areas, model adaptation has become increasingly important to sustain and enhance this progress. As machine learning models are deployed in dynamic real-world environments, they encounter ever-changing data distributions (Koh et al., 2021), new user requirements, and potential adversarial threats (Goodfellow et al., 2014), motivating various fields of research, such as domain adaptation (Ganin et al., 2016b; Ben-David et al., 2010), transfer learning (Zhuang et al., 2020; Yosinski et al., 2014), adversarial robustness (Madry et al., 2018a), out-of-distribution detection (Hendrycks and Gimpel, 2016; Yang et al., 2024), etc.

Recently, the development foundational models, like ChatGPT (Achiam et al., 2023), Claude (Anthropic, 2023), Gemini (Team et al., 2023), has seen great empirical success, while demanding extensive resources for pretraining. These large foundational models inherently gain the ability to adapt to new tasks via in-context learning (Mann et al., 2020; Min et al., 2022; Dong et al., 2022). Furthermore, researchers and practitioners continue to explore models' ability to adapt to harder tasks and customized responses at a reduced cost much lower than the pretraining. It leads to new area of researches, such as finetuning (Hu et al., 2021; Wei et al., 2021) and alignment (Ouyang et al., 2022; Rafailov et al., 2024; Wang et al., 2023a).

With the success of these model adaptation procedures, **understanding and improving the model adaptation process** in the era of deep learning and foundational models has become a crucial research problem to study. In particular, my research dives into the model adaptation from two aspects, **Robustness** and **Large Language Models**. For my research, I would answer the following questions: **Q1** What is the proper notion of robustness for machine learning model under adaptation? And can we improve the robustness of the adaptive modeling procedure? **Q2** Can we

explain the success of various adaptive modeling procedures in large language models and make further improvements?

1.1 Robustness for Model under Adaptation

In the 1990s and 2000s, people have systematically studied transductive learning in comparison with inductive learning as two different inference modes. The transductive inference framework aligned with early machine learning methods like K-nearest neighbor (Cover and Hart, 1967) and has also motivated important early machine learning methods like transductive support vector machine (Joachims et al., 1999). (Vapnik et al., 1998; Vapnik, 2006) first formally define the concept of transductive inference against inductive inference:

- **Inductive Inference**: *Estimation of the function*
- **Transductive Inference**: *Estimation of the value of the function at the points of interest*

Over the last decades, people have also developed various model adaptation methods that aim to improve the performance of predictive models when the training and test data are related but from different domains driven by the transductive inference methodology. Some successful techniques include feature alignment such as DANN (Ganin et al., 2016b), or self-supervision with auxiliary tasks, such as TTT (Sun et al., 2020).

With the rapid development of these methods, analyzing their inherent robustness becomes an important research question. Earlier works on studying robustness only focus on the adversarial robustness for inductive models, where no model adaptation occurs. In our previous preprint (Chen et al., 2021a), we first formulate an adversarial threat model for test-time model adaptation, where the defender may have a unique advantage as the adversarial game becomes a maximin game, instead of a minimax game as in the classic adversarial robustness threat model. We then study whether the maximin threat model admits more "good solutions" than

the minimax threat model, and is thus strictly weaker. For this purpose, we first present a provable separation between the two threat models in a natural Gaussian data model.

Based on this motivating formulation, we investigated the model adaptation procedure and made the following important findings: 1. Many model adaptation methods seemingly robust to inductive adversarial attacks are very susceptible against transductive attacks that attack the space of possibly adapted models. 2. Both theoretically and empirically, introducing the rejection option to a transductive model can further improve the robustness of a transductive model. I will summarize our high-level intuitions in the next paragraphs, and people can refer to Chapter 2 and Chapter 3 for further details.

Evaluating the Robustness for Model under Adaptation There has been emerging interest in using transductive learning for adversarial robustness (Goldwasser et al., NeurIPS 2020; Wu et al., ICML 2020; Wang et al., ArXiv 2021). Compared to traditional defenses, these defense mechanisms "dynamically learn" the model based on test-time input; and theoretically, attacking these defenses reduces to solving a bilevel optimization problem, which poses difficulty in crafting adaptive attacks. In this paper, we examine these defense mechanisms from a principled threat analysis perspective. We formulate and analyze threat models for transductive-learning based defenses, and point out important subtleties. We propose the principle of attacking model space for solving bilevel attack objectives, and present Greedy Model Space Attack (GMSA), an attack framework that can serve as a new baseline for evaluating transductive learning based defenses. Through systematic evaluation, we show that GMSA, even with weak instantiations, can break previous transductive-learning based defenses, which were resilient to previous attacks, such as AutoAttack. On the positive side, we report a somewhat surprising empirical result of "transductive adversarial training": Adversarially retraining the model using fresh randomness at the test time gives a significant increase in robustness against attacks we consider.

Improving the Transductive Robustness via Rejection Both transduction and rejection have emerged as important techniques for defending against adversarial perturbations. A recent work by (Goldwasser et al., 2020a) showed that rejection combined with transduction can give *provable* guarantees (for certain problems) that cannot be achieved otherwise. Nevertheless, under recent strong adversarial attacks (GMSA (Chen et al., 2022)), Goldwasser et al.'s work was shown to have low performance in a practical deep-learning setting. In this paper, we take a step towards realizing the promise of transduction+rejection in more realistic scenarios. Our key observation is that a novel application of a reduction technique in (Tramèr, 2022), which was until now only used to demonstrate the vulnerability of certain defenses, can be used to actually construct effective defenses. Theoretically, we show that a careful application of this technique in the transductive setting can give significantly improved samplecomplexity for robust generalization. Our theory guides us to design a new transductive algorithm for learning a selective model; extensive experiments using state-of-the-art attacks (AutoAttack, GMSA) show that our approach provides significantly better robust accuracy (81.6% on CIFAR-10 and 57.9% on CIFAR-100 under l_∞ with budget 8/255) than existing techniques (Croce et al., 2020).

1.2 Model Adaptation in Large Language Models

In Part II, we discuss two applications of model adaptations in LLMs, Finetuning for creative generation corresponding to across-context adaptation, and RAG corresponding to within-context adaptation. See Chapter 4 and Chapter 5 for more details, and here is the overview of our key ideas.

Adapting LLM for Creative Generation The success of foundational models has sprouted people's interests in adapting it to different domains via techniques like finetuning and alignment. One particularly interesting domain is the creative task, where desirable responses are highly diverse and idiosyncratic. It draws a difference with classical NLP tasks like question answering and summarization,

where the desirable responses share great similarities. One of the ACL 2023 best papers (Hessel et al., 2022) made seminal contributions towards understanding humor by asking LLMs to perform matching, quality ranking, and explanation generation tasks on the New Yorker Humor Dataset. We proceed one step further and benchmark the ability to actually generate humorous captions via finetuning and alignment.

We present a novel multimodal preference dataset for creative tasks, consisting of over 250 million human ratings on more than 2.2 million captions, collected through crowdsourcing rating data for The New Yorker's weekly cartoon caption contest over the past eight years. This unique dataset supports the development and evaluation of multimodal large language models and preference-based fine-tuning algorithms for humorous caption generation. We propose novel benchmarks for judging the quality of model-generated captions, utilizing both GPT4 and human judgments to establish ranking-based evaluation strategies. Our experimental results highlight the limitations of current fine-tuning methods, such as RLHF and DPO, when applied to creative tasks. Furthermore, we demonstrate that even state-of-the-art models like GPT4 and Claude currently underperform top human contestants in generating humorous captions. As we conclude this extensive data collection effort, we release the entire preference dataset to the research community, fostering further advancements in AI humor generation and evaluation.

Adapting LLMs for RAG In recent years, the remarkable in-context learning abilities of pretrained Large Language Models (LLMs) have drawn significant attention (Brown et al., 2020). Subsequent research has systematically benchmarked and analyzed these capabilities from both empirical and theoretical perspectives (Hendel et al., 2023; Garg et al., 2022; Zhang et al., 2024). Retrieval-Augmented Generation (RAG) further extends LLMs by incorporating externally retrieved texts into their inputs, enabling models to access knowledge beyond their pretraining corpus. RAG has been widely adopted for open-domain question answering, fact-checking, and other knowledge-intensive tasks (Huang et al., 2023; Lewis et al., 2020b; Ramos et al., 2022; Sarto et al., 2022; Zhao et al., 2024b).

To clarify discussions on adaptation, we distinguish between two related yet distinct notions: **learning with context** and **in-context learning**. Learning with context generally describes scenarios where the model conditions on additional information, such as user-provided or retrieved contexts. In contrast, in-context learning specifically refers to models learning from contexts composed of explicit input-output demonstration pairs. Historically, these terms were sometimes used interchangeably. When applied specifically to retrieval methods, we can draw an analogy between **RAG** (retrieving documents that inform the generation without explicit demonstrations) and **in-context retrieval** (retrieving explicit input-output pairs serving as demonstrations) (Luo et al., 2024). Within the scope of RAG, distinguishing between these concepts becomes nuanced because retrieved documents potentially containing labels could be interpreted under either framework, blurring the traditional boundaries.

In Chapter 4, we present a unified perspective of Retrieval-Augmented Generation (RAG) as noisy in-context learning (ICL). Within this framing, retrieved examples serve as noisy context, with their quality directly dependent on retrieval effectiveness. For theoretical feasibility, we specifically consider the in-context retrieval scenario, while empirically, we investigate both general RAG and in-context retrieval scenarios. While previous theoretical analyses of ICL assume clean, independently and identically distributed examples (Ahn et al., 2023; Zhang et al., 2024), these assumptions fail to capture RAG's inherent noisiness, where example quality is inversely correlated with retrieval relevance. Currently, no theoretical framework has been developed to study RAG under this structured ICL formulation. To bridge this gap, we explicitly model RAG as noisy ICL, characterizing retrieval noise under both uniform conditions (consistent noise across examples) and non-uniform conditions (noise inversely related to retrieval relevance). By modeling retrieval as introducing structured but perturbed distributions at test time—absent during pretraining—we quantify the impact of retrieval noise and derive explicit generalization bounds based on the number of in-context and retrieved examples, as well as their retrieval distances from the queries.

1.3 Thesis Outline

This doctoral thesis investigates two central themes: robustness of models under adaptation (Part I), and practical applications of model adaptation in large language models (LLMs), both across-context (fine-tuning) and within-context (in-context learning) (Part II).

Part I includes Chapter 2 and Chapter 3.

In **Chapter 2**, we formalize the adaptation process within a transductive framework and propose a corresponding threat model for robust adversarial evaluation in attacker-defender settings. Initially, we introduce a straightforward baseline, the fixed-point attack; however, recognizing its limited effectiveness, we develop a more potent method—Gradient Model Space Averaging (GMSA)—which leverages the average of previously adapted models in model space. Empirical results demonstrate the enhanced effectiveness of our GMSA attack. Detailed descriptions of datasets, experimental setups, and comprehensive additional experiments can be found in Chapter A.

In **Chapter 3**, we first formally define robustness in the setting that combines transduction and rejection. We then theoretically analyze this combined approach, establishing an improved generalization error bound compared to either induction with rejection or transduction alone. Additionally, we propose a novel defense method that integrates transduction and rejection directly into the training objective, demonstrating its empirical effectiveness. Full proofs and ablation studies are provided in Chapter B.

Part II includes Chapter 4 and Chapter 5.

In **Chapter 4**, we first detail our crowdsourcing pipeline used to create the benchmark dataset. Next, we introduce a scalable evaluation approach using calibrated LLM-based judgments, establishing it as an effective evaluation criterion. We then comprehensively benchmark various caption-generation strategies, including fine-tuning, Best-of-N, RLHF, DPO, and multimodal fine-tuning, and identify DPO as the most effective method for this creative generation task. Further details on prompt design and additional empirical analyses, such as generation diversity

studies, can be found in Chapter C.

In **Chapter 5**, we first formalize the mathematical framework for Retrieval-Augmented Generation (RAG) and justify the key assumptions underlying the data. We then provide a theoretical analysis, deriving generalization bounds and optimal selection criteria for in-context examples under uniform retrieval noise. Next, we extend this analysis to the non-uniform retrieval setting, specifically examining two practical scenarios: noise proportional to query distance and noise sampled with probability weighted by query distance. We conduct empirical evaluations on real-world datasets, mapping these two scenarios onto different retrieval regimes (documents vs. passages, few vs. more rag examples) and interpreting their practical implications. The complete proofs supporting our theoretical results are provided in Chapter D.

Part I

Robustness for Models under Adaptation

2 TOWARDS EVALUATING THE ROBUSTNESS OF NEURAL NETWORKS LEARNED BY TRANSDUCTION

Ensuring trustworthiness in machine learning is critical. Traditional approaches typically focus on inductive robustness, aiming to produce models resistant to *any* perturbations around test inputs. While this standard provides strong guarantees, the gap between robust and non-robust performance can be substantial, potentially limiting practical applicability (Croce et al., 2020). Moreover, as highlighted by insights from "Adversarial Examples Are Not Bugs, They Are Features" (Ilyas et al., 2019), rigidly enforcing robust features may sacrifice valuable aspects of model performance. This motivates a more relaxed yet pragmatic robustness framework, where models are allowed to dynamically adapt at test time—commonly referred to as the transductive setting. The work presented in this chapter establishes a foundational evaluation framework for understanding and assessing the robustness of models capable of test-time adaptation. Specifically, we systematically analyze and critique existing transductive-learning-based defenses, propose a novel principled attack methodology (Greedy Model Space Attack, GMSA), and demonstrate both vulnerabilities and promising new directions, such as transductive adversarial training, for achieving meaningful adversarial robustness.

2.1 Introduction

Adversarial robustness of deep learning models has received significant attention in recent years (see Kolter and Madry (2018) and references therein). The classic threat model of adversarial robustness considers an *inductive setting* where a model is learned at the training time and fixed, and then at the test time, an attacker attempts to thwart the fixed model with adversarially perturbed input. This gives rise to the adversarial training (Madry et al., 2018b; Sinha et al., 2018; Schmidt et al., 2018; Carmon et al., 2019) to enhance adversarial robustness.

Going beyond the inductive threat model, there has been emerging interest in

using *transductive learning* (Vapnik, 1998)[1] for adversarial robustness (Goldwasser et al., 2020b; Wu et al., 2020b; Wang et al., 2021). In essence, these defenses attempt to leverage *a batch* of *test-time inputs*, which is common for ML pipelines deployed with batch predictions (bat, 2021), to *learn an updated model*. The hope is that this "test-time learning" may be useful for adversarial robustness since the defender can adapt the model to the perturbed input from the adversary, which is distinct from the inductive threat model where a model is fixed after training.

This paper examines these defenses from a principled threat analysis perspective. We first formulate and analyze rigorous threat models. Our basic 1-round threat model considers a single-round game between the attacker and the defender. Roughly speaking, the attacker uses an objective $\max_{V' \in N(V)} L_a(\Gamma(U'), V')$ (formula (2.2)), where V is the given test batch, $N(V)$ is a neighborhood around V, L_a is a loss function for attack gain, Γ is the transductive-learning based defense, and $U' = V'|_X$, the projection of V' to features, is the adversarially perturbed data for breaking Γ. This objective is *transductive* as U', the attacker's output, appears in both attack (V' in L_a) and defense (U' in Γ). We extend this threat model to multiple rounds, which is necessary when considering DENT (Wang et al., 2021) and RMC (Wu et al., 2020b). We point out important subtleties in the modeling that were unclear or overlooked in previous work.

We then study *adaptive attacks*, that is to leverage the knowledge about Γ to construct attacks. Compared to situations considered in BPDA (Athalye et al., 2018), a transductive learner Γ is even further from being differentiable, and theoretically the attack objective is a bilevel optimization (Colson et al., 2007). To address these difficulties, our key observation is to consider the *transferability of adversarial examples*, and consider a *robust* version of (2.2): $\max_{U'} \min_{\overline{U} \in \mathcal{N}(U')} L_a(\Gamma(\overline{U}), V')$ (formula (2.6)), where we want to find a *single* attack set U' to thwart a family of models, induced by $\overline{U}$ "around" U'. This objective relaxes the attacker-defender constraint, and provides more information in dealing with nondifferentiability. To

[1] We note that this type of defense goes under different names such as "test-time adaptation" or "dynamic defenses". Nevertheless, they all fall into the classic transductive learning paradigm (Vapnik, 1998), which attempts to leverage test data for learning. We thus call them *transductive-learning based defenses*. The word "transductive" is also adopted in Goldwasser et al. (2020b).

solve the robust objective, we propose Greedy Model Space Attack (GMSA), a general attack framework which attempts to solve the robust objective in a greedy manner. GMSA can serve as a new baseline for evaluating transductive-learning based defenses.

We perform a systematic empirical study on various defenses. For RMC (Wu et al., 2020b), DENT (Wang et al., 2021), and URejectron (Goldwasser et al., 2020b), we show that even weak instantiations of GMSA can break respective defenses. Specifically, for defenses based on adversarially training, we reduce the robust accuracy to that of adversarial training alone. We note that, under AutoAttack (Croce and Hein, 2020a), the state-of-the-art adaptive attack for the inductive threat model, some of these defenses have claimed to achieve *substantial improvements* compared to *adversarial training alone*. For example, Wang et al. show that DENT can improve the robustness of the state-of-the-art adversarial training defenses by more than 20% absolutely against AutoAttack on CIFAR-10. However, under our adaptive attacks, DENT only has minor improvement: less than 3% improvement over adversarial training alone. Our results thus demonstrates significant differences between attacking transductive-learning based defenses and attacking in the inductive setting, and significant difficulties in the use of transductive learning to improve adversarial robustness. On the positive side, we report a somewhat surprising empirical result of *transductive adversarial training*: Adversarially retraining the model using fresh private randomness on a new batch of test-time data gives a significant increase in robustness against all of our considered attacks.

2.2 Related Work

Adversarial robustness in the inductive setting. Many attacks have been proposed to evaluate the adversarial robustness of the defenses in the inductive setting where the model is fixed during the evaluation phase (Goodfellow et al., 2015a; Carlini and Wagner, 2017b; Kurakin et al., 2017; Moosavi-Dezfooli et al., 2016; Croce and Hein, 2020b). Principles for adaptive attacks have been developed in Tramèr et al. (2020) and many existing defenses are shown to be broken based on attacks developed

from these principles (Athalye et al., 2018). A fundamental method to obtain adversarial robustness in this setting is adversarial training (Madry et al., 2018b; Zhang et al., 2019b). A state-of-the-art attack in the inductive threat model is AutoAttack (Croce and Hein, 2020a).

Adversarial robustness via test-time defenses. There have been various work which attempt to improve adversarial robustness by leveraging test-time data. Many of such work attempt to "sanitize" test-time input using a non-differentiable function, and then send it to a pretrained model. Most of these proposals were broken by BPDA (Athalye et al., 2018). To this end, we note that a research agenda for "dynamic model defense" has been proposed in Goodfellow (2019b).

Adversarial robustness using transductive learning. There has been emerging interesting in using transductive learning to improve adversarial robustness. In view of "dynamic defenses", these proposals attempt to apply transductive learning to the test data and update the model, and then use the updated model to predict on the test data. In this work we consider three such work (Wu et al., 2020b; Wang et al., 2021; Goldwasser et al., 2020b).

2.3 Preliminaries

Let F be a model, and for a data point $(\mathbf{x}, y) \in \mathcal{X} \times \mathcal{Y}$, a loss function $\ell(F; \mathbf{x}, y)$ gives the loss of F on the point. Let V be a set of labeled data points, and let $L(F, V) = \frac{1}{|V|} \sum_{(\mathbf{x},y)\in V} \ell(F; \mathbf{x}, y)$ denote the empirical loss of F on V. For example, if we use binary loss $\ell^{0,1}(F; \mathbf{x}, y) = \mathbb{1}[F(\mathbf{x}) \neq y]$, this gives the test error of F on V. We use the notation $V|_X$ to denote the projection of V to its features, that is $\{(\mathbf{x}_i, y_i)\}_{i=1}^m|_X \mapsto \{\mathbf{x}_i\}_{i=1}^m$. Throughout the paper, we use $N(\cdot)$ to denote a neighborhood function for perturbing features: That is, $N(\mathbf{x}) = \{\mathbf{x}' \mid d(\mathbf{x}', \mathbf{x}) < \epsilon\}$ is a set of examples that are close to $\mathbf{x}$ in terms of a distance metric d (e.g., $d(\mathbf{x}', \mathbf{x}) = \|\mathbf{x}' - \mathbf{x}\|_p$). Given $U = \{\mathbf{x}_i\}_{i=1}^m$, let $N(U) = \{\{\mathbf{x}'_i\}_{i=1}^m \mid d(\mathbf{x}'_i, \mathbf{x}_i) < \epsilon, i = 0, \ldots, m\}$. Since labels are not changed for adversarial examples, we also use the notation $N(V)$ to denote perturbations of features, with labels fixed.

2.4 Modeling Transductive Robustness

In this section we formulate and analyze threat models for transductive defenses. We first formulate a threat model for a single-round game between the attacker and the defender. We then consider extensions of this threat model to multiple rounds, which are necessary when considering DENT (Wang et al., 2021) and RMC (Wu et al., 2020b), and point out important subtleties in modeling that were not articulated in previous work. We characterize previous test-time defenses using our threat models.

1-round game. In this case, the adversary "intercepts" a clean test data V (with clean features $U = V|_X$, and labels $V|_Y$), adversarially perturbs it, and sends a perturbed features U' to the defender. The defender learns a new model based on U'. A referee then evaluates the accuracy of the adapted model on U'. Formally:

Definition 2.1 (***1-round threat model for transductive adversarial robustness***)**.** *Fix an adversarial perturbation type (e.g., ℓ_∞ perturbations with perturbation budget ε). Let $P_{X,Y}$ be a data generation distribution. The attacker is an algorithm $\mathcal{A}$, and the defender is a pair of algorithms $(\mathcal{T}, \Gamma)$, where $\mathcal{T}$ is a supervised learning algorithm, and Γ is a transductive learning algorithm. A (clean) training set D is sampled i.i.d. from $P_{X,Y}$. A (clean) test set V is sampled i.i.d. from $P_{X,Y}$.*

• **[*Training time, defender*]** *The defender trains an optional base model $F = \mathcal{T}(D)$, using the labeled source data D.*

• **[*Test time, attacker*]** *The attacker receives V, and produces an (adversarial) unlabeled dataset U':*

1. *On input Γ, F, D, and V, $\mathcal{A}$ perturbs each point $(\mathbf{x}, y) \in V$ to $(\mathbf{x}', y)$ (subject to the agreed attack type), giving $V' = \mathcal{A}(\Gamma, F, D, V)$ (that is, $V' \in N(V)$).*
2. *Send $U' = V'|_X$ (the feature vectors of V') to the defender.*

• **[*Test time, defender*]** *The defender produces a model as $F^* = \Gamma(F, D, U')$.*

Multi-round games. The extension of 1-round games to multi-round contains several important considerations that were implicit or unclear in previous work, and is closely related to what it means by *adaptive attacks*. Specifically:

Private randomness. Note that Γ uses randomness, such as random initialization and random restarts[2] in adversarial training. Since these randomness are generated *after* the attacker's move, they are treated as *private randomness*, and *not* known to the adversary.

Intermediate defender states leaking vs. Non-leaking. In a multi-round game, *the defender* may maintain states across rounds. For example, the defender may store test data and updated models from previous rounds, and use them in a new round. If these intermediate defender states are "leaked" to the attacker, we call it *intermediate defender states leaking*, or simply *states leaking*, otherwise we call it *non states-leaking*, or simply non-leaking. Note that the attacker *cannot* simply compute these information by simulating on the training and testing data, due to the use of *private randomness*. We note that, however, the initial pretrained model is assumed to be known by the attacker. The attacker can also of course maintain arbitrary states, and are assumed not known to the defender.

Adaptive vs. Non-adaptive. Because transductive learning happens after the attacker produces U', the attacker may not be able to directly attack the model Γ produced. Nevertheless, the attacker is assumed to have *full knowledge of the transductive mechanism* Γ, except the private randomness. In this paper we call an attack *adaptive* if it makes *explicit use of the knowledge of* Γ.

Naturally ordered vs. Adversarially ordered. Both RMC and DENT handle batches of fixed sizes. An intuitive setup for multi-round game is that the batches come in sequentially, and the attacker must forward perturbed versions of these batches *in the same order* to the defender, which we call the "naturally ordered" game. However, this formulation does not capture an important scenario: An adversary can wait and pool a large amount of test data, then chooses a *"worst-case" order* of perturbed data points, and then sends them in batches one at a time for adaptation

[2]When perturbing a data point during adversarial training, one starts with a random point in the neighborhood.

in order to maximize the breach. We call the latter "adversarially ordered" game. We note that all previous work only considered naturally-ordered game, which gives the defender more advantages, and is thus our focus in the rest of the paper. Adversarially-ordered game is evaluated for DENT in Appendix A.1.7.

Modeling capacity of our threat models. Our threat models encompass a large family of defenses. For example, without using Γ, the threat model degenerates to the classic inductive threat model. Our threat models also capture various "test-time defenses" proposals (e.g., those broken by the BPDA (Athalye et al., 2018)), where Γ is a "non-differentiable" function which *"sanitizes" the test data*, instead of updating the model, before sending them to a fixed pretrained model. Therefore, in particular, these proposals are not transductive-learning based. Below we describe previous defenses which we study in the rest of this paper, where Γ is indeed transductive learning.

Example 2.2 (***Runtime masking and cleansing***)**.** *Runtime masking and cleansing* (*RMC*) (*Wu et al., 2020b*) *is a recent transductive-learning defense. For RMC, the defender is stateful and adapted from the model* learned in the last round*, on a single test point* ($|U| = 1$)*: The adaptation objective is* $F^* = \arg\min_F \sum_{(\mathbf{x},y)\in N'(\widehat{\mathbf{x}})} L(F, \mathbf{x}, y)$*, where* $\widehat{\mathbf{x}}$ *is the test time feature point, and* $N'(\widehat{\mathbf{x}})$ *is the set of examples in the adversarial training dataset* D' *that are top-K nearest to* $\widehat{\mathbf{x}}$ *in a distance measure. RMC paper considered two attacks:* **(1) Transfer attack***, which generates perturbed data by attacking the initial base model, and* **(2) PGD-skip attack***, which at round* $p + 1$*, runs PGD attack on the model learned at round* p*. In our language, transfer attack is* stateless (*i.e. the adversary maintains no state*) *and* non-adaptive*, PGD-skip attack is* state-leaking*, but still* non-adaptive*.*

Example 2.3 (***Defensive entropy minimization*** (***DENT*** (***Wang et al., 2021***))). *DENT adapts the model using test input, and can work with any training-time learning procedure. The DENT defender is* stateless*: It always starts the adaptation from the original pretrained model, fixed at the training time. During the test-time adaptation, only the affine parameters in batch normalization layers of the base model are updated, using entropy minimization with the information maximization regularization. In this paper, we show that with strong adaptive attacks under the naturally ordered setting, we are able to reduce the robustness to*

be almost the same as that of static models (see Section 2.6). Further, under the adversarially ordered setting, we can completely break DENT.

Example 2.4 (***Goldwasser et al.'s transductive threat model***)**.** *While seemingly our threat model is quite different from the one described in Goldwasser et al. (2020b), one can indeed recover their threat model naturally as a* 1-round game*: First, for the perturbation type, we simply allow arbitrary perturbations in the threat model setup. Second, we have a fixed pretrained model* F*, and the adaptation algorithm* Γ *learns a set* S *which represents the set of "allowable" points (so* $F|_S$ *yields a predictor with redaction, namely it outputs* $\perp$ *for points outside of* S*). Third, we define two error functions as (5) and (6) in Goldwasser et al. (2020b):*

$$\underset{U'}{\mathrm{err}}(F|_S, f) \equiv \frac{1}{|U'|}\left|\left\{ x' \in U' \cap S \,\middle|\, F(x') \neq f(x') \right\}\right|, \quad \underset{U}{\mathrm{rej}}(S) \equiv \frac{|U \setminus S|}{|U|} \tag{2.1}$$

where f *is the ground truth hypothesis. The first equation measures prediction errors in* U' *that passed through* S*, and the second equation measures the rejection rate of the clean input. The referee evaluates by measuring two errors:* $L(F|_S, V') = \left(\mathrm{err}_{U'}(F|_S), \mathrm{rej}_U(S)\right)$.

2.5 Adaptive Attacks in One Round

In this section we study a basic question: *How to perform adaptive attacks against a transductive-learning based defense in* one round*?* Note that, in each round of a multi-round game, an *independent* batch of test input U is sampled, and the defender can use transductive learning to produce a model specifically adapted to the adversarial input U', *after* the defender receives it. Therefore, it is of fundamental interest to attack this ad-hoc adaptation. We consider white-box attacks: The attacker knows all the details of Γ, except private randomness, which is sampled after the attacker's move.

We deduce a principle for adaptive attacks in one round, which we call *the principle of attacking model space*: Effective attacks against a transductive defense may need to consider *attacking a set of representative models induced in the neighborhood of*

U. We give concrete instantiations of this principle, and show in experiments that they break previous transductive-learning based defenses.

Attacks in multi-round. If the transductive-learning based defense is stateless, then we simply repeat one-round attack multiple times. If it is stateful, then we need to consider state-leaking setting or non-leaking setting. For all experiments in Section 2.6, we only evaluate *non-leaking* setting, which is more challenging for the adversary.

2.5.1 Goal of the attacker and challenges

To start with, given a defense mechanism Γ, the objective of the attacker can be formulated as:

$$\max_{V' \in N(V), U' = V'|_X} L_a\left(\Gamma\left(F, D, U'\right), V'\right). \tag{2.2}$$

where L_a is the loss function of the attacker. We make some notational simplifications: Since D is a constant, in the following we drop it and write $\Gamma\left(U'\right)$. Also, since the attacker does not modify the labels in the threat model, we abuse the notation and write the objective as

$$\max_{V', U' = V'|_X} L_a\left(\Gamma\left(U'\right), U'\right). \tag{2.3}$$

A generic attacker would proceed iteratively as follows: It starts with the clean test set V, and generates a sequence of (*hopefully*) *increasingly stronger* attack sets $U^{(0)} = V|_X, U^{(1)}, \ldots, U^{(i)}$ ($U^{(i)}$ must satisfy the attack constraints at U, such as ℓ_∞ bound). We note several basic but important *differences* between transductive attacks and inductive attacks in the classic minimax threat model:

(D1) $\Gamma(U')$ **is *not* differentiable**. For the scenarios we are interested in, Γ is an optimization algorithm to solve an objective $F^* \in \arg\min_F L_d(F, D, U')$. This

renders (2.3) into a bilevel optimization problem (Colson et al., 2007):

$$\max_{V' \in N(V); U'=V'|_X} L_a(F^*, V') \quad \text{subject to: } F^* \in \arg\min_F L_d(F, D, U'), \tag{2.4}$$

In these cases, Γ is in general *not* (in fact far from) differentiable. A natural attempt is to approximate Γ with a differentiable function, using theories such as Neural Tangent Kernels (Jacot et al., 2018). Unfortunately no existing theory applies to the transductive learning, which deals with unlabeled data U' (also, as we have remarked previously, tricks such as BPDA (Athalye et al., 2018) also does not apply because transductive learning is much more complex than test-time defenses considered there).

(D2) U' appears in *both* attack and defense. Another significant difference is that the attack set U' also appears as the input for the defense (i.e. $\Gamma(U')$). Therefore, while it is easy to find U' to fail $\Gamma(\overline{U})$ for any fixed $\overline{U}$, it is much harder to find a *good direction* to update the attack and converge to *an attack set U^* that fails an entire model space induced by itself:* $\Gamma(U^*)$.

(D3) $\Gamma(U')$ can be a *random variable*. In the classic minimax threat model, the attacker faces a fixed model. However, the output of Γ can be a *random variable of models* due to its private randomness, such as the case of Randomized Smoothing (Cohen et al., 2019). In these cases, successfully attacking a single sample of this random variable does not suffice.

Fixed Point Attack: A first attempt. We adapt previous literature for solving bilevel optimization in deep learning setting (Lorraine and Duvenaud, 2018) (designed for supervised learning). The idea is simple: At iteration $i+1$, we fix $U^{(i)}$ and model space $F^{(i)} = \Gamma(U^{(i)})$, and construct $U^{(i+1)}$ to fail it. We call this the Fixed Point Attack (FPA) (Algorithm 1), as one hopes that this process converges to a good fixed point U^*. Unfortunately, we found FPA to be weak in experiments. The reason is exactly **(D2)**: $U^{(i+1)}$ failing $F^{(i)}$ may not give any indication that it can also fail $F^{(i+1)}$ induced by itself. Note that transfer attack is a special case of FPA by setting $T = 0$.

Algorithm 1 FIXED POINT ATTACK (FPA)

Require: A transductive learning algorithm Γ, an optional training dataset D, a clean test set V, an initial model $F^{(0)}$, and an integer parameter $T \geqslant 0$ (the number of iterations).

1: **for** $i = 0, 1, \ldots, T$ **do**

2: Attack the model obtained in the last iteration to get the perturbed set:

$$V^{(i)} = \underset{V' \in N(V)}{\arg\max}\, L_a(F^{(i)}, V') \tag{2.5}$$

where L_a is a loss function. Set $U^{(i)} = V^{(i)}|_X$.

3: Run the transductive learning algorithm Γ to get the next model: $F^{(i+1)} = \Gamma(D, U^{(i)})$.

4: **end for**

5: Select the best attack set $U^{(k)}$ as $k = \arg\max_{0 \leqslant i \leqslant T} L(F^{(i+1)}, V^{(i)})$.

6: **return** $U^{(k)}$.

2.5.2 Strong adaptive attacks from attacking model spaces

To develop stronger adaptive attacks, we consider a key property of the adversarial attacks: The *transferability of adversarial examples*. Various previous work have identified that adversarial examples transfer (Tramèr et al., 2017; Liu et al., 2016), even across vastly different architectures and models. Therefore, if U' is a good attack set, we would expect that U' also fails $\Gamma(\overline{U})$ for $\overline{U}$ close to U'. This leads to the consideration of the following objective:

$$\max_{U'} \min_{\overline{U} \in \mathcal{N}(U')} L_a(\Gamma(\overline{U}), U'). \tag{2.6}$$

where $\mathcal{N}(\cdot)$ is a neighborhood function (possibly different than N). It induces a family of models $\{\Gamma(U') \mid U' \in \mathcal{N}(U^*)\}$, which we call a *model space*. (in fact, this can be a family of random variables of models) This can be viewed as a natural *robust* version of (2.3) by considering the transferability of U'. While this is seemingly even harder to solve, it has several benefits: **(1) Considering a model space naturally strengthens FPA.** FPA naturally falls into this formulation as a weak instantiation where we consider a single $\overline{U} = U^{(i)}$. Also, considering a model space gives the attacker more information in dealing with the nondifferentiability of Γ **(D1)**. **(2) It relaxes the attacker-defender constraint (D2).** Perhaps more importantly, for

the robust objective, we no longer need the same U' to appear in both defender and attacker. Therefore it gives a natural relaxation which makes attack algorithm design easier.

In summary, while "brittle" U' that does not transfer may indeed exist theoretically, their identification can be challenging algorithmically, and its robust variant provides a natural relaxation considering both algorithmic feasibility and attack strength. This thus leads us to the following principle:

The Principle of Attacking Model Spaces. *An effective adaptive attack against a transductive-learning based defense may need to consider* a model space *induced by a proper neighborhood of* U.

Algorithm 2 Greedy Model Space Attack (GMSA)

Require: A transductive learning algorithm Γ, an optional training dataset D, a clean test set V, an initial model $F^{(0)}$, and an integer parameter $T \geqslant 0$ (the number of iterations).

1: **for** $i = 0, 1, \ldots, T$ **do**

2: Attack the previous models to get the perturbed set:

$$V^{(i)} = \arg\max_{V' \in N(V)} L_{\text{GMSA}}(\{F^{(j)}\}_{j=0}^{i}, V') \tag{2.7}$$

where L_{GMSA} is a loss function. Set $U^{(i)} = V^{(i)}|_X$.

3: Run the transductive learning algorithm Γ to get the next model: $F^{(i+1)} = \Gamma(D, U^{(i)})$.

4: **end for**

5: Select the best attack $U^{(k)}$ as $k = \arg\max_{0 \leqslant i \leqslant T} L(F^{(i+1)}, V^{(i)})$,

6: **return** $U^{(k)}$.

An instantiation: Greedy Model Space Attack (GMSA). We give a simplest possible instantiation of the principle, which we call the *Greedy Model Space Attack* (Algorithm 2). In experiments we use this instantiation to break previous defenses. In this instantiation, the family of model spaces to consider is just all the model spaces constructed in previous iterations. $L_{\text{GMSA}}(\{F^{(j)}\}_{j=0}^{i}, V')$ is a loss function that the attacker uses to attack the history model spaces. We consider two instantiations: (1) $L_{\text{GMSA}}^{\text{AVG}}(\{F^{(j)}\}_{j=0}^{i}, V') = \frac{1}{i+1}\sum_{j=0}^{i} L_a(F^{(j)}, V')$, (2) $L_{\text{GMSA}}^{\text{MIN}}(\{F^{(j)}\}_{j=0}^{i}, V') = \min_{0 \leqslant j \leqslant i} L_a(F^{(j)}, V')$, where $L_{\text{GMSA}}^{\text{AVG}}$ gives attack algorithm

GMSA-AVG, and L_{GMSA}^{MIN} gives attack algorithm GMSA-MIN. We solve (2.7) via Projected Gradient Decent (PGD) (the implementation details of GMSA can be found in Appendix A.1.1).

2.6 Empirical Study

This section evaluates various transductive-learning based defenses. Our main findings are: **(1)** The robustness of existing transductive defenses like RMC and DENT is overestimated. Under our evaluation framework, those defenses either have little robustness or have almost the same robustness as that of the static base model. To this end, we note that while AutoAttack is effective in evaluating the robustness of static models, it is not effective in evaluating the robustness of transductive defenses. In contrast, our GMSA attack is a strong baseline for attacking transductive defenses. **(2)** We experimented a novel idea of applying Domain Adversarial Neural Networks (Ajakan et al., 2014), an unsupervised domain adaptation technique (Wilson and Cook, 2020), as a transductive-learning based defense. We show that DANN has nontrivial and even better robustness compared to existing work, under AutoAttack, PGD attack, and FPA attack, even though it is broken by GMSA. **(3)** We report a somewhat surprising phenomenon on *transductive adversarial training*: Adversarially retraining the model *using fresh private randomness on a new batch of test-time data* gives a significant increase in robustness, against all of our considered attacks. **(4)** Finally, we demonstrated that URejectron, while enjoying theoretical guarantees in the bounded-VC dimensions situation, can be broken in natural deep learning settings.

Evaluation framework. For each defense, we report accuracy and robustness. The accuracy is the performance on the clean test inputs, and the robustness is the performance under adversarial attacks. The robustness of transductive defenses is estimated using AutoAttack (AA)[3], PGD attack, FPA, GMSA-MIN and GMSA-AVG. We use PGD attack and AutoAttack in the transfer attack setting for the transductive defense: We generate adversarial examples by attacking a static model (e.g. the base

[3]We use the standard version of AutoAttack: https://github.com/fra31/auto-attack/.

Dataset	Base Model	Accuracy		Robustness					
		Static	RMC	Static	RMC				
				AA	AA	PGD	FPA	GMSA-AVG	GMSA-MIN
MNIST	Standard	99.50	99.00	0.00	97.70	98.30	0.60	**0.50**	1.10
	Madry et al.	99.60	97.00	87.70	95.70	96.10	59.50	61.40	**58.80**
CIFAR-10	Standard	94.30	93.10	0.00	94.20	97.60	8.50	**8.00**	8.10
	Madry et al.	83.20	90.90	44.30	77.90	71.70	40.80	42.50	**39.60**

Table 2.1: Results of evaluating RMC. We also evaluate the static base model for comparison. **Bold** numbers are worst results.

Base Model	Accuracy		Robustness						
	Static	DENT	Static	DENT					
			AA	DENT-AA	AA	PGD	FPA	GMSA-AVG	GMSA-MIN
Wu et al. (2020a)	85.70	86.10	58.00	78.80	64.40	59.50	**59.30**	59.60	59.60
Carmon et al. (2019)	88.00	87.40	57.30	80.10	61.70	**58.40**	**58.40**	58.50	58.50
Sehwag et al. (2020)	87.30	86.90	54.90	76.50	59.60	**55.80**	**55.80**	**55.80**	**55.80**
Wang et al. (2020)	86.60	85.60	53.60	75.90	61.30	55.90	**55.80**	56.10	56.10
Hendrycks et al. (2019)	85.80	85.50	51.80	77.20	58.40	**54.20**	54.40	**54.20**	**54.20**
Wong et al. (2020)	81.20	81.00	42.40	69.70	48.90	**44.10**	44.30	44.50	44.30
Ding et al. (2020)	82.40	82.40	39.70	62.80	44.80	39.90	39.40	**39.10**	39.20

Table 2.2: Results of evaluating DENT on CIFAR-10. We also evaluate the static base model for comparison. **Bold** numbers are worst results.

Dataset	Accuracy			Robustness						
	Standard	Madry et al.	DANN	Standard	Madry et al.	DANN				
				AA	AA	AA	PGD	FPA	GMSA-AVG	GMSA-MIN
MNIST	99.42	99.16	99.27	0.00	88.92	97.59	96.66	96.81	79.37	**6.17**
CIFAR-10	93.95	86.06	89.61	0.00	39.49	66.61	60.54	53.98	**5.53**	8.56

Table 2.3: Results of evaluating DANN. **Bold** numbers are worst results.

model used by the transductive defense), and then evaluate the transductive defense on the generated adversarial examples. Accuracy and robustness of the static models are also reported for comparison. We always use AutoAttack to estimate the robustness of static models since it is the state-of-the-art for the inductive setting. For all experiments, the defender uses his own private randomness, which is different from the one used by the attacker. Without specified otherwise, all reported values are percentages. Below we give details. Appendix A.1 gives details for replicating the results.

Runtime Masking and Cleansing (RMC (Wu et al., 2020b)). RMC adapts the

network at test time, and was shown to achieve state-of-the-art robustness under several attacks that are unaware of the defense mechanism (thus these attacks are non-adaptive according to our definition). We follow the setup in Wu et al. (2020b) to perform experiments on MNIST and CIFAR-10 to evaluate the robustness of RMC. On MNIST, we consider L_∞ norm attack with $\epsilon = 0.3$ and on CIFAR-10, we consider L_∞ norm attack with $\epsilon = 8/255$. The performance of RMC is evaluated on a sequence of test points $\mathbf{x}^{(1)}, \cdots, \mathbf{x}^{(n)}$ randomly sampled from the test dataset. So we have a n-round game. The FPA and GMSA attacks are applied on each round and the initial model $F^{(0)}$ used by the attacks at the $(k+1)$-th round is the adapted model (with calibration in RMC) obtained at the k-th round. To save computational cost, we set $n = 1000$. The robustness of RMC is evaluated on a sequence of adversarial examples $\hat{\mathbf{x}}^{(1)}, \cdots, \hat{\mathbf{x}}^{(n)}$ generated by the attacker on the sequence of test points $\mathbf{x}^{(1)}, \cdots, \mathbf{x}^{(n)}$. We evaluate the robustness of RMC in the non-state leaking setting with private randomness (both are in favor of the defender).

Results. The results are in Table 2.1. RMC with the standard model is already broken by FPA attack (weaker than GSMA). Compared to the defense-unaware AutoAttack, our GMSA-AVG attack reduces the robustness from 97.70% to 0.50% on MNIST and from 94.20% to 8.00% on CIFAR-10. Further, RMC with adversarially trained model actually provides *worse* adversarial robustness than using adversarial training alone. Under our GMSA-MIN attack, the robustness is reduced from 96.10% to 58.80% on MNIST and from 71.70% to 39.60% on CIFAR-10.

Defensive Entropy Minimization (DENT (Wang et al., 2021)). DENT performs test-time adaptation, and works for any training-time learner. It was shown that DENT improves the robustness of the state-of-the-art adversarial training defenses by 20+ points absolute against AutoAttack on CIFAR-10 under L_∞ norm attack with $\epsilon = 8/255$ (DENT is implemented as a model module, and AutoAttack is directly applied to the module, and we denote this as DENT-AA). Wang et al. also considers adaptive attacks for DENT, such as attacking the static base model using AutoAttack to generate adversarial examples, which is the same as the AutoAttack (AA) in our evaluation.

We evaluate the best version of DENT, called DENT+ in Wang et al., under their

Dataset	Accuracy		Robustness					
	Madry et al.	TADV	Madry et al.	TADV				
			AA	AA	PGD	FPA	GMSA-AVG	GMSA-MIN
MNIST	99.01	99.05	86.61	96.07	96.48	95.47	**94.27**	95.48
CIFAR-10	87.69	88.51	45.29	72.12	59.05	58.64	**54.12**	57.77

Table 2.4: Results of evaluating TADV. **Bold** numbers are worst results.

original settings on CIFAR-10: DENT is combined with various adversarial training defenses, and only the model adaptation is included without input adaptation. The model is adapted sample-wise for six steps by AdaMod (Ding et al., 2019) with learning rate of 0.006, batch size of 128 and no weight decay. The adaptation objective is entropy minimization with the information maximization regularization. To save computational cost, we only evaluate on 1000 examples randomly sampled from the test dataset. We consider L_∞ norm attack with $\epsilon = 8/255$. We design loss functions for the attacks to generate adversarial examples with high confidence (See Appendix A.1.3 for the details).

Results. Table 2.2 show that both DENT-AA and AA overestimate the robustness of DENT. Our PGD attack reduces the robustness of DENT to be almost the same as that of the static defenses. Further, our FPA, GMSA-AVG and GMSA-MIN have similar performance as the PGD attack. The results show that AutoAttack is not effective in evaluating the robustness of transductive defenses.

Domain Adversarial Neural Network (DANN (Ajakan et al., 2014)). We consider DANN as a transductive defense for adversarial robustness. We train DANN on the labeled training dataset D (source domain) and unlabeled adversarial test dataset U' (target domain), and then evaluate DANN on U'. For each adversarial set U', we train a new DANN model from scratch. We use the standard model trained on D as the base model for DANN. We perform experiments on MNIST and CIFAR-10 to evaluate the adversarial robustness of DANN. On MNIST, we consider L_∞ norm attack with $\epsilon = 0.3$ and on CIFAR-10, we consider L_∞ norm attack with $\epsilon = 8/255$.

Results. Table 2.3 show that DANN has non-trivial robustness under AutoAttack, PGD attack and FPA attack. However, under our GMSA attack, DANN has little robustness.

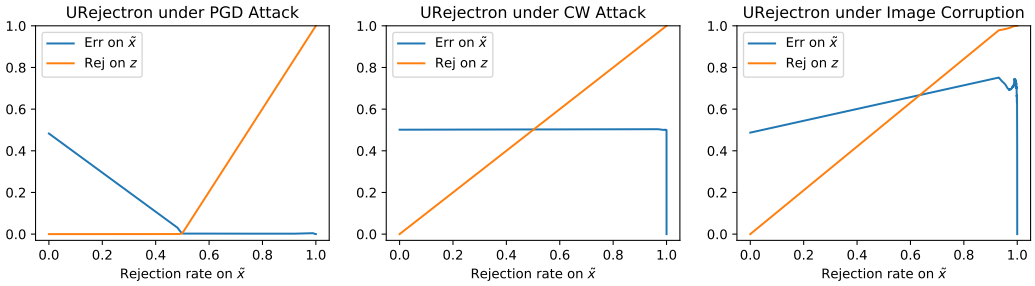


Figure 2.1: URejectron in three settings. z contains "normal" examples on which the classifier can have high accuracy. $\tilde{x}$ includes z and consists of a mix of 50% "normal" examples and 50% adversarial examples. In (a), the normal examples are clean test inputs and the adversarial examples are generated by PGD attack. In (b), the "normal" examples are still clean test inputs but adversarial examples are generated by CW attack. In (c), the "normal" examples are generated by image corruptions (adversarial examples are generated by PGD attacks).

Transductive Adversarial Training (TADV). We consider a simple but novel transductive-learning based defense called transductive adversarial training: After receiving a set of examples at the test time, we always adversarially retrain the model using fresh randomness. The key point of this transduction is that *private randomness* is sampled after the attacker's move, and so the attacker cannot directly attack the resulting model as in the inductive case. Specifically, for our GMSA attacks, we attack (with loss L^{AVG}_{GMSA} or L^{MIN}_{GMSA}) an ensemble of $T = 10$ models, adversarially trained with independent randomness, and generate a perturbed test set U'. Then we adversarially train another model from scratch with independent randomness, and check whether U' transfers to the new model (this thus captures the scenario described earlier). Somewhat surprisingly, we show that U' does not transfer very well, and the TADV improves robustness significantly.

Results. Table 2.4 shows that transductive adversarial training significantly improves the robustness of adversarial training (Madry et al., 2018b). On MNIST, the robustness is improved from 86.61% to 94.27%. On CIFAR-10, the robustness is improved from 45.29% to 54.12%.

URejectron in deep learning settings. URejectron performs transductive learning

for defense, and has theoretical guarantees under bounded VC dimension case. We evaluated URejectron on GTSRB dataset using ResNet18 network. We used the same implementation by Goldwasser et al..

Results. Figure 2.1(a) shows that for *transfer attacks* generated by PGD attack (Madry et al., 2018b), URejectron can indeed work as expected. However, by using different attack algorithms, such as CW attack (Carlini and Wagner, 2017b), we observe two failure modes: **(1)** *Imperceptible adversarial perturbations that slip through*. Figure 2.1(b) shows that one can construct adversarial examples that are very similar to the clean test inputs that can slip through their URejectron construction of S (in the deep learning setting), and cause large errors. **(2)** *Benign perturbations that get rejected*. Figure 2.1(c) shows that we can generate "benign" perturbed examples using image corruptions, such as slightly increased brightness, but URejectron rejects all.

2.7 Conclusion

In this paper, we formulate threat models for transductive defenses and propose an attack framework called Greedy Model Space Attack (GMSA) that can serve as a new baseline for evaluating transductive defenses. We show that GMSA can break previous transductive defenses, which were resilient to previous attacks such as AutoAttack. On the positive side, we show that transductive adversarial training gives a significant increase in robustness against attacks we consider. For the future work, one can explore transductive defenses that can be robust under our GMSA attacks, and can also explore even stronger adaptive attacks that are effective in evaluating transductive defenses.

3 TWO HEADS ARE *ACTUALLY* BETTER THAN ONE: TOWARDS BETTER ADVERSARIAL ROBUSTNESS VIA TRANSDUCTION AND REJECTION

As established in the previous chapter, transductive robustness provides a promising avenue to relax conventional adversarial robustness constraints by allowing test-time adaptation. A natural next step is to investigate whether and how model performance can be further enhanced within this relaxed robustness framework. In this chapter, we address this question by exploring rejection as a complementary strategy to transduction. Specifically, building upon insights from recent theoretical advances—such as Tramèr's (Tramèr, 2022) classifier-to-detector reduction and Goldwasser et al.'s (Goldwasser et al., 2020a) provable guarantees transductive learning—we demonstrate that combining rejection with transduction not only leads to significant theoretical improvements in robust generalization but also translates into substantial empirical gains under state-of-the-art adversarial attacks.

3.1 Introduction

A recent line of research (Goldwasser et al., 2020a; Montasser et al., 2021; Goodfellow, 2019a; Wang et al., 2021; Wu et al., 2020c) has investigated augmenting models with *transduction* (leveraging unlabeled test input to revise the learned model) and *rejection* (allowing a model to reject on certain input) to defend against adversarial perturbations. There are in general two classes of algorithms. One class is *transduction-only*. For example, (Montasser et al., 2021) showed that robust learning with transduction allows for significant improvements in sample complexity, reducing dependency on VC dimension from exponential to linear; however, this comes at the cost of significantly greater assumptions on the data ($\mathrm{OPT}_{\mathcal{U}^2}$ for the realizable case rather than the $\mathrm{OPT}_{\mathcal{U}}$ of the inductive setting [1]).

[1]The optimal robust risk is $\mathrm{OPT}_{\mathcal{U}} = \inf_{h\in\mathcal{H}} \Pr_{(x,y)\sim\mathcal{D}} [\exists z \in \mathcal{U}(x) : h(z) \neq y]$. For $\mathcal{U}$ which are perturbations up to ϵ in some metric, $\mathcal{U}^x$ is a perturbation of up to $x\epsilon$, see Section 3.3 for more

The other class is to have both transduction and rejection. For example, (Goldwasser et al., 2020a) studied this setting and showed even more surprising results, not achievable with transduction or rejection alone. However, one prominent limitation of these works seems to be that none has yet resulted in practical robust learning mechanisms in the deep learning setting typically considered.

In this paper, we take a step towards realizing the promise of transduction+rejection in more realistic scenarios. Compared to (Goldwasser et al., 2020a), which considers arbitrary perturbations, we focus on the classic and practical scenario of bounded perturbations for deep learning. Somewhat surprisingly, we show that a novel application of Tramèr's classifier-to-detector technique (Tramèr, 2022), which has thus far only been applied to indicate that certain defenses are vulnerable, in the transductive setting can give significantly improved sample-complexity for robust generalization, noting that bounded perturbations are critical for the construction to work. To obtain these improvements, we do not require stronger assumptions on the data, as with (Montasser et al., 2021); in the realizable case, we only need to assume $\text{OPT}_{\mathcal{U}^{2/3}} = 0$, which is even better than the $\text{OPT}_{\mathcal{U}} = 0$ assumption in the inductive case.

Our theory guides us to identify a practical transductive algorithm for learning a robust selective model. As a component, we present a simple empirical approximation to the reduction which enables the computationally efficient realization of the improvement to robustness offered by rejection; our experiments show that the the robustness of models utilizing our rejection-only defense very closely matches the theoretical bound (i.e. the robustness achievable to adversarial budget $\epsilon/2$). While our approach does not have the theoretical guarantees of the computationally inefficient construction, it is a significant step towards developing an efficient reduction, left as an open problem by (Tramèr, 2022).

In addition, we present an objective for general adaptive attacks targeting selective classifiers based on our algorithm. Our transductive defense algorithm gives strong empirical performance on image classification tasks, both against our adaptive attack and against existing state-of-the-art attacks such as AutoAttack

details.

Table 3.1: **Summary of generalization bounds for the four settings**. Compared to transduction alone and (Goldwasser et al., 2020a), our defense weakens the necessary conditions in the realizable case and improves the asymptotic error in the agnostic case. Compared to induction and rejection alone, sample complexity has a linear rather than exponential dependence on the VC dimension. Compared to (Goldwasser et al., 2020a), the dependence on the error bound ϵ improves from inverse quadratic to inverse linear in the realizable case. Note that (Goldwasser et al., 2020a) requires the existence of a hypothesis with bounded error on the perturbed data in the agnostic case, and hence does not tolerate all possible perturbations.

	Realizable			
	Soundness Condition	Completeness Condition	Generalization Bound	Agnostic Generalization Bound
Induction (Montasser et al., 2019)	$\mathrm{OPT}_{\mathcal{U}} = 0$	$\mathrm{OPT}_{\mathcal{U}} = 0$	$\mathcal{O}\left(\frac{2^{\mathrm{VC}(\mathcal{H})}\log(n)+\log(1/\delta)}{n}\right)$	$\mathrm{OPT}_{\mathcal{U}} + \mathcal{O}\left(\sqrt{\frac{2^{\mathrm{VC}(\mathcal{H})}+\log(1/\delta)}{n}}\right)$
Transduction (Montasser et al., 2021)	$\mathrm{OPT}_{\mathcal{U}^2} = 0$	$\mathrm{OPT}_{\mathcal{U}^2} = 0$	$\mathcal{O}\left(\frac{\mathrm{VC}(\mathcal{H})\log(n)+\log(1/\delta)}{n}\right)$	$2\mathrm{OPT}_{\mathcal{U}^2} + O\left(\sqrt{\frac{\mathrm{VC}(\mathcal{H})+\log(1/\delta)}{n}}\right)$
Rejection (Theorem B.2, B.6)	$\mathrm{OPT}^{\mathrm{rej}}_{\mathcal{U}} = 0$	$\mathrm{OPT}^{\mathrm{rej}}_{\mathcal{U}} = 0$	$\mathcal{O}\left(\frac{2^{\mathrm{VC}(\mathrm{T}(\mathcal{H}))}\log(n)+\log(1/\delta)}{n}\right)$	$\mathrm{OPT}^{\mathrm{rej}}_{\mathcal{U}} + \mathcal{O}\left(\sqrt{\frac{2^{\mathrm{VC}(\mathrm{T}(\mathcal{H}))}+\log(1/\delta)}{n}}\right)$
Transduction+Rejection (Goldwasser et al., 2020a)	$\mathrm{OPT}_{\mathcal{U}} = 0$	$\mathrm{OPT}_{\mathcal{U}} = 0$	$\mathcal{O}\left(\sqrt{\frac{\mathrm{VC}(\mathcal{H})\log(n)}{n}} + \frac{\log(1/\delta)}{n}\right)$	$2\,\mathrm{OPT}_{\mathcal{U}} + 2\sqrt{2\,\mathrm{OPT}_{\mathcal{U}}} + \mathcal{O}\left(\sqrt{\frac{\mathrm{VC}(\mathcal{H})\log n+\log(1/\delta)}{n}}\right)$
Transduction+Rejection (Theorem 3.1, B.12)	$\mathrm{OPT}_{\mathcal{U}^{2/3}} = 0$	$\mathrm{OPT}_{\mathcal{U}^2} = 0$	$\mathcal{O}\left(\frac{\mathrm{VC}(\mathcal{H})\log(n)+\log(1/\delta)}{n}\right)$	$2\mathrm{OPT}_{\mathcal{U}^{2/3}} + O\left(\sqrt{\frac{\mathrm{VC}(\mathcal{H})+\log(1/\delta)}{n}}\right)$

and standard GMSA. On CIFAR-10, we obtain 81.6% transductive robust accuracy with rejection, a significant improvement on the current state-of-the-art result of 71.1% (Peng et al., 2023; Croce et al., 2020) for robust accuracy up to the perturbation considered (l_∞ with budget $\epsilon = 8/255$); on CIFAR-100, we obtain 57.9% transductive robust accuracy with rejection, significantly exeeding the strongest existing baseline of 42.7% (Wang et al., 2023b; Croce et al., 2020) with the same adversarial budget.

The rest of the paper is organized as follows. Section 3.2 reviews main related work, and Section 3.3 presents some necessary background. We develop our theory results in Section 3.4. Guided by our theory, Section 3.5 develops a practical robust learning algorithm, leveraging both transduction and rejection. We provide systematic experiments in Section 3.6, and conclude in Section 3.7.

3.2 Related Work

In recent years, there have been extensive studies on adversarial robustness in the traditional inductive learning setting, where the model is fixed during the

evaluation phase (Carlini and Wagner, 2017a; Goodfellow et al., 2015b; Moosavi-Dezfooli et al., 2016). Most popular and effective methods are adversarial training, such as PGD (Madry et al., 2018a), TRADES (Zhang et al., 2019a). These methods are effective against adversaries on small dataset like MNIST, but still ineffective on complex dataset like CIFAR-10 or ImageNet (Croce et al., 2020). Defenses beyond adversarial training have been proposed but most are broken by strong adaptive attacks (Croce and Hein, 2020c; Tramer et al., 2020).

To break this robust bottleneck, recent work has proposed alternative settings with relaxed yet realistic assumptions, particularly by allowing rejection and transduction. In robust learning with rejection (a.k.a., abstain), we allow rejection of adversarial examples instead of correctly classifying all of them (Tramèr, 2022). Variants of adversarial training with rejection option have been considered (Laidlaw and Feizi, 2019; Pang et al., 2022; Chen et al., 2021b; Kato et al., 2020; Sotgiu et al., 2020; He et al., 2022), including generalizations to unseen attacks (Stutz et al., 2020) and to certified robustness (Sheikholeslami et al., 2020; Baharlouei et al., 2022; Sheikholeslami et al., 2022). (Tramèr, 2022) proves an equivalence between robust learning with rejection and standard robust learning in the inductive setting and shows that the evaluation of past defenses with rejection was unreliable.

The other approach is to define an alternative notion of adversarial robustness via transductive learning, i.e. "dynamically" ensuring robustness on the particular given test samples rather than on the whole distribution. Similar settings have been studied but under the view of "test-time defense" or "dynamic defense" (Goodfellow, 2019a; Wang et al., 2021; Wu et al., 2020c). (Goldwasser et al., 2020a) is the first paper to formalize transductive learning for robust learning, and the first to consider transduction+rejection. It considers general adversaries on test data and presents novel theoretical guarantees. (Chen et al., 2022) formally defines the notion of transductive robustness as a maximin problem and presents a principled adaptive attack, GMSA. (Montasser et al., 2021) discusses robust transductive learning against bounded perturbation from a learning theory perspective and obtains corresponding sample complexity.

3.3 Preliminaries

Table 3.2: Summary of the robust error in all settings. Note that transductive error of the learner $\mathbb{A}$ is the corresponding notion of error where $h = \mathbb{A}(\mathbf{x}, \mathbf{y}, \tilde{\boldsymbol{z}})$.

	Robust Error	Robust Error (with Rejection)
Inductive	$\mathrm{err}_{\mathcal{U}}(h; x, y) := \sup_{z \in \mathcal{U}(x)} \mathbb{1}\{h(z) \neq y\}$	$\mathrm{err}^{\mathrm{rej}}_{\mathcal{U}}(h; x, y) := \sup_{z \in \mathcal{U}(x)} \mathbb{1}\{h(z) \notin \{y, \perp\} \vee h(x) \neq y\}$
Transductive	$\mathrm{err}_{\mathcal{U}}(h; \mathbf{x}, \mathbf{y}, \tilde{\mathbf{z}}, \tilde{\mathbf{y}}) := \frac{1}{m}\sum_{i=1}^{m} \mathbb{1}\{h(\tilde{z}_i) \neq \tilde{y}_i\}$	$\mathrm{err}^{\mathrm{rej}}_{\mathcal{U}}(h; \mathbf{x}, \mathbf{y}, \tilde{\mathbf{x}}, \tilde{\mathbf{z}}, \tilde{\mathbf{y}}) := \frac{1}{m}\sum_{i=1}^{m} \mathbb{1}\left\{ \begin{array}{l} (h(\tilde{z}_i) \notin \{\tilde{y}_i\} \wedge \tilde{z}_i = \tilde{x}_i) \\ \vee (h(\tilde{z}_i) \notin \{\tilde{y}_i, \perp\} \wedge \tilde{z}_i \neq \tilde{x}_i) \end{array} \right\}$

Let $\mathcal{X}$ denote the input space, $\mathcal{Y}$ the label space, $\mathcal{D}$ the clean data distribution over $\mathcal{X} \times \mathcal{Y}$. We will assume binary classification for our theoretical analysis: $\mathcal{Y} = \{\pm 1\}$. Let $\mathcal{U}(x)$ denote the set of possible perturbations of an input x, e.g., for ℓ_p norm perturbation of budget ϵ, $\mathcal{U}$ is the ℓ_p ball of radius ϵ: $\mathcal{U}(x) = \{z : \|z - x\|_p \leqslant \epsilon\}$. We assume $\mathcal{U}$ satisfies $\forall x \in \mathcal{X}, x \in \mathcal{U}(x)$; essentially all interesting perturbations satisfy this. Let $\mathcal{U}^2(x) := \{z : \exists t \in \mathcal{U}(x), \text{such that } z \in \mathcal{U}(t)\}$, and $\mathcal{U}^{-1}(x) := \{z : x \in \mathcal{U}(z)\}$. If a perturbation set Λ satisfies $\Lambda^2 = \mathcal{U}$, then we say $\Lambda = \mathcal{U}^{1/2}$; $\mathcal{U}^{-1/2} = (\mathcal{U}^{-1})^{1/2}$. When $\mathcal{U}$ is the ℓ_p ball of radius ϵ, $\mathcal{U}^2$ is that of radius 2ϵ, $\mathcal{U}^{-1} = \mathcal{U}$, and $\mathcal{U}^{1/2}$ is that of radius $\epsilon/2$; we define $\mathcal{U}^3$, $\mathcal{U}^{1/3}$, and $\mathcal{U}^{-1/3}$ similarly.

All learners are provided with n i.i.d. training samples [2] $(\mathbf{x}, \mathbf{y}) = (x_i, y_i)_{i=1}^n \sim \mathcal{D}^n$. There are m i.i.d. test samples $(\tilde{\mathbf{x}}, \tilde{\mathbf{y}}) \sim \mathcal{D}^m$, and the adversary can perturb $\tilde{\mathbf{x}}$ to $\tilde{\boldsymbol{z}} \in \mathcal{U}(\tilde{\mathbf{x}})$.

We describe the main settings below; the corresponding notions of error are in Table 3.2. For each setting, we define risk as the expected worst-case error up to the perturbation $\mathcal{U}$, and empirical risk similarly.

Induction. In the traditional robust classification setting (e.g., (Madry et al., 2018a)); also called the inductive setting or simply induction), the learning algorithm (the defender) is given training set $(\mathbf{x}, \mathbf{y})$, learns a classifier $h : \mathcal{X} \mapsto \mathcal{Y}$ from some hypothesis class $\mathcal{H}$.

[2] Here $\mathbf{x} = (x_i)_{i=1}^n$ and similarly with $\mathbf{y}, \tilde{\mathbf{x}}, \tilde{\mathbf{y}}$, etc. We will also overload the notation $\mathcal{U}$, e.g., $\mathcal{U}(\mathbf{x}) := \{\mathbf{u} \in \mathcal{X}^n : u_i \in \mathcal{U}(x_i)\}$.

Rejection. In the setting of robust classification with rejection, the classifier has the extra power of abstaining (i.e., outputting a rejection option denoted by $\perp$), and furthermore, rejecting a perturbed input does not incur an error. The learning algorithm is given training set $(\mathbf{x}, \mathbf{y})$ and learns a *selective classifier*, defined as a function

$$h : \mathcal{X} \mapsto \mathcal{Y} \cup \{\perp\} \tag{3.1}$$

from some hypothesis class $\mathcal{H}$ which, given a sample x, either outputs a label $y \in \mathcal{Y}$ or abstains from prediction with an output of $\perp$. An error occurs only when h rejects a clean input, or accepts and misclassifies. We define additionally $\mathrm{OPT}^{\mathrm{rej}}_{\mathcal{U}} := \inf_{h \in \mathcal{H}} \mathbb{R}^{\mathrm{rej}}_{\mathcal{U}}(h; \mathcal{D})$.

Transduction. In the setting of robust classification with transduction (e.g., (Montasser et al., 2021)), the learning algorithm (the transductive learner) has access to the unlabeled test input data; the goal is to predict labels only for these given test inputs (a transductive learner need not generalize). The learner $\mathbb{A}$ is given the training data $(\mathbf{x}, \mathbf{y})$ and the (potentially perturbed) test inputs $\tilde{\mathbf{z}}$, and outputs m labels $h(\tilde{\mathbf{z}}) = (h(\tilde{z}_i))_{i=1}^m$ as predictions for $\tilde{\mathbf{z}}$. That is, the learner is a mapping $\mathbb{A} : (\mathcal{X} \times \mathcal{Y})^n \times \mathcal{X}^m \mapsto \mathcal{Y}^m$. A special case is when $\mathbb{A}$ learns a classifier h and use it to label $\tilde{\mathbf{z}}$; the labels are also denoted as $h(\tilde{\mathbf{z}})$.

Our setting: Transduction+Rejection. A transductive learner for selective classifiers $\mathbb{A}$ is given $(\mathbf{x}, \mathbf{y}, \tilde{\mathbf{z}})$, and outputs rejection or a label for each input in $\tilde{\mathbf{z}}$. That is, the learner is a mapping $\mathbb{A} : (\mathcal{X} \times \mathcal{Y})^n \times \mathcal{X}^m \mapsto (\mathcal{Y} \cup \{\perp\})^m$. An error occurs when it rejects a clean test input or accepts and misclassifies. Hence, we present the appropriate notion of error in Table 3.2, the natural extension of the rejection-only error to the transductive setting, with the key difference being that we penalize rejection only if the sample is not perturbed (as transductive learners produce outputs only on the provided test data, there is no notion of rejecting $\tilde{x}_i$ if it has been perturbed).

3.4 Theoretical Analysis

In this section, we present theoretical results which guide the design of our algorithm (see Section 3.5). We show that, by applying an algorithm which produces a hypothesis robust with reduced ϵ about an intermediate perturbation and incorporating rejection (via Tramèr's classifer-to-detector reduction (Tramèr, 2022)), we can derive an algorithm with strong guarantees up to the full ϵ; in particular, the full algorithm obtains linear dependence on the VC dimension with greatly reduced necessary conditions compared to transduction alone. This suggests that this simple approach may provide significant improvements to robustness. We find in that this is indeed the case: our algorithm, described in Section 3.5, obtains significantly improved robustness compared to existing baselines; see Section 3.6 for more details.

We focus on the realizable case for the setting with transduction+rejection here, for more details and results for the agnostic case and the setting with rejection alone see Appendix B.1. For comparison with existing results in the inductive-only and transduction-only settings (Montasser et al., 2019, 2021), we follow their setup: assume there exists a classifier (without rejection) with 0 robust error from a hypothesis class $\mathcal{H}$ of VC-dimension $\mathrm{VC}(\mathcal{H})$; the goal is to design a learner with a small robust error.

Theorem 3.1. *For any $n \in \mathbb{N}$, $\delta > 0$, hypothesis class $\mathcal{H}$ of classifiers without rejection, perturbation set $\mathcal{U}$ such that $\mathcal{U} = \mathcal{U}^{-1}$ and $\mathcal{U}^{1/3}$ exists, and distribution $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$ satisfying $\mathrm{OPT}_{\mathcal{U}^{2/3}} = 0$, there exists a transductive learner $\mathbb{A}$ that constructs a set of selective classifiers (of the form Equation (3.1)) Δ s.t. the following is true: with probability $\geqslant 1-\delta$ over $(\mathbf{x}, \mathbf{y}) \sim \mathcal{D}^n$, $(\tilde{\mathbf{x}}, \tilde{\mathbf{y}}) \sim \mathcal{D}^n$, we have that for any $\tilde{\mathbf{z}} \in \mathcal{U}(\tilde{\mathbf{x}})$, if $\Delta \neq \emptyset$, then for any $h \in \Delta$,* [3]

$$\mathrm{err}^{\mathrm{rej}}_{\mathcal{U}}(h; \mathbf{x}, \mathbf{y}, \tilde{\mathbf{x}}, \tilde{\mathbf{z}}, \tilde{\mathbf{y}}) \leqslant \frac{\mathrm{VC}(\mathcal{H}) \log(2n) + \log(1/\delta)}{n}.$$

[3] Note that Δ is a function of $\mathbf{x}$, $\mathbf{y}$, and $\tilde{\mathbf{z}}$, so this is more precisely a bound of $\sup_{\tilde{\mathbf{z}} \in \mathcal{U}(\tilde{\mathbf{x}}), h \in \mathbb{A}(\mathbf{x}, \mathbf{y}, \tilde{\mathbf{z}})} \mathrm{err}^{\mathrm{rej}}_{\mathcal{U}}(h; \mathbf{x}, \mathbf{y}, \tilde{\mathbf{x}}, \tilde{\mathbf{z}}, \tilde{\mathbf{y}})$.

For $\mathcal{U}$ satisfying our conditions (including l_p balls), we obtain a stronger guarantee than those using only transduction or only rejection. First, compared to the guarantee for transduction without rejection (Montasser et al., 2021) (see Table 3.1), our result requires weaker assumptions on the data: we need $\mathrm{OPT}_{\mathcal{U}^{2/3}} = 0$ rather than $\mathrm{OPT}_{\mathcal{U}^2} = 0$. For example, consider the ℓ_p norm perturbation: $\mathcal{U}(x) = \{z : \|z-x\|_p \leqslant \epsilon\}$. Transduction alone requires that there exists a classifier with 0 robust error up to the perturbation $\mathcal{U}^2$, i.e. up to an ℓ_p norm perturbation of adversarial budget 2ϵ. In contrast, our result shows that using both transduction and rejection only requires there exists a classifier with 0 robust error up to perturbation $\mathcal{U}^{2/3}$, corresponding to adversarial budget of $2\epsilon/3$. Equivalently, for a data distribution with a margin 2ϵ, transduction without rejection can only handle adversarial perturbations with budget ϵ, while combining transduction and rejection can handle adversarial perturbations with budget 3ϵ, tolerating three times the adversarial magnitude. Second, compared to rejection only (see Table 3.1), this bound has a linear sample complexity rather than exponential. Therefore, combining transduction and rejection has the benefits of both techniques.

Furthermore, note that the result bounds the rate of incorrect rejections as well, i.e. the rate of rejections on clean data, with the same bound as a direct consequence of the definition of robust error under transduction and rejection. However, the result, while potentially very strong, comes with the caveat that the defense is not guaranteed to find a nonempty Δ (i.e., the defense is sound but may not be complete) under conditions weaker than $\mathrm{OPT}_{\mathcal{U}^2} = 0$; by Lemma B.14 in Appendix B.1.3, Δ is guaranteed to be nonempty, and hence we have completeness, under the same conditions as transduction alone. Hence, the result is strictly stronger than the result for transduction alone (Montasser et al., 2021).

Consider an adversarial budget ϵ, and suppose $\tilde{z}$ is the given potentially perturbed test input and $\tilde{x}$ is the corresponding clean test input. To obtain the guarantee, we need to find a model which is $\epsilon/3$-robust at $q = \tilde{x} + (\tilde{z} - \tilde{x})/3$. Such a model always exists when $\mathrm{OPT}_{\mathcal{U}^{2/3}} = 0$. However, given only $\tilde{z}$ without knowing q or $\tilde{x}$, our algorithm finds a model $\epsilon/3$-robust at every perturbation within $2\epsilon/3$ of $\tilde{z}$ and thus Δ may be empty.

While weaker conditions don't guarantee that we find a model satisfying the conditions, the result still provides intuition for the success of our derived empirical defense. For typical data distributions and hypothesis classes, it might be expected that, if we fail to find a ϵ-robust hypothesis at the fully-perturbed data, we will nevertheless be more likely to find a model which is robust nearer the clean data distribution (i.e. where the condition is required by the theory) rather than further away. Determining conditions for this is an interesting direction for future research.

Such conditions do exist: in Appendix B.1.3 we present a distribution $\mathcal{D}$, hypothesis class $\mathcal{H}$, and perturbation $\mathcal{U}$ for which Δ is guaranteed to be nonempty and the error bound above applies, but where trasduction has a minimum asymptotic error of $1/2$.

Proof Sketch. For intuition, think of $\mathcal{U}$ as the ℓ_p norm perturbation with adversarial budget ϵ. We omit technical details; see Appendix B.1.3 for the complete proof. Consider some clean training set $\mathbf{x}, \mathbf{y}$, clean test set $\tilde{\mathbf{x}}, \tilde{\mathbf{y}}$, with perturbed test data $\tilde{\mathbf{z}}$ with $\tilde{z}_i$ within ϵ of $\tilde{x}_i$. Let $\tilde{\mathbf{z}}' = \tilde{\mathbf{x}} + (\tilde{\mathbf{z}} - \tilde{\mathbf{x}})/3$ be the intermediate perturbation a third of the way between $\tilde{\mathbf{x}}$ and $\tilde{\mathbf{z}}$.

First, following (Montasser et al., 2021), define the set of robust hypotheses $\Delta_{\mathcal{H}}^{\mathcal{U}^{1/3}}(\mathbf{x}, \mathbf{y}, \tilde{\mathbf{z}}')$ as $\Delta_{\mathcal{H}}^{\mathcal{U}^{1/3}}(\mathbf{x}, \mathbf{y}, \tilde{\mathbf{z}}') = \{R_{\mathcal{U}^{1/3}}(h; \mathbf{x}, \mathbf{y}) = 0 \wedge R_{\mathcal{U}^{1/3}}(h; \tilde{\mathbf{z}}') = 0\}$ where $R_{\mathcal{U}}(h; \mathbf{z}, \mathbf{y}) = \sup_{\tilde{\mathbf{x}} \in \mathcal{U}(\mathbf{z})} \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}\{h(\tilde{x}_i) \neq y_i\}$ and $R_{\mathcal{U}}(h; \mathbf{z}) = R_{\mathcal{U}}(h; \mathbf{z}, h(\mathbf{z}))$.

That is, we find those classifiers that satisfy: (1) they are $\epsilon/3$-robustly correct (i.e., correct and robust to perturbations of budget $\epsilon/3$) on the training data $(\mathbf{x}, \mathbf{y})$; (2) they have $\epsilon/3$ margin on the intermediate perturbations $\tilde{\mathbf{z}}'$ (i.e., have the same prediction for all perturbations of budget $\epsilon/3$). This then guarantees, as shown in (Montasser et al., 2021), that with high probability, for any $h \in \Delta_{\mathcal{H}}^{\mathcal{U}^{1/3}}(\mathbf{x}, \mathbf{y}, \tilde{\mathbf{z}}')$ the robust error facing perturbation of budget $\epsilon/3$ is bounded by $\frac{VC(\mathcal{H}) \log(2n) + \log(1/\delta)}{n}$ if $OPT_{\mathcal{U}^{2/3}} = 0$.

Next, following (Tramèr, 2022), define a transformation $F_{\mathcal{U}^{1/3}}$ that maps a classi-

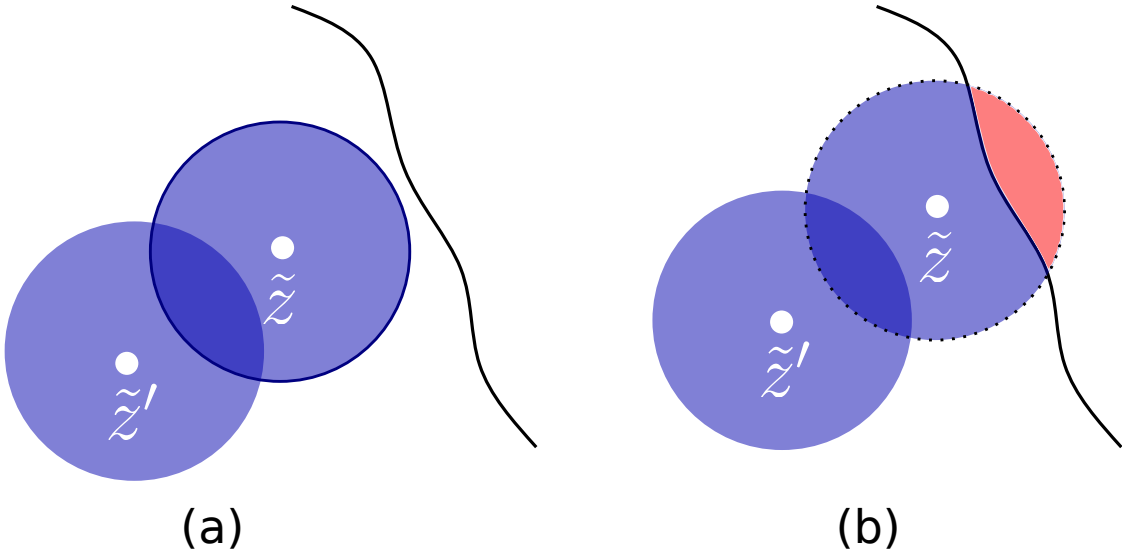


Figure 3.1: (a) h is $\epsilon/3$-robust at $\tilde{z}$; $\hat{h}$ correctly classifies $\tilde{z}$.
(b) h is not $\epsilon/3$-robust at $\tilde{z}$; $\hat{h}$ rejects $\tilde{z}$.

fier without rejection, h, to the selective classifier (see Equation 3.1) $\hat{h} = F_{\mathcal{U}^{1/3}}(h)$:

$$\hat{h}(x) = \begin{cases} h(x) & \text{if } \forall x' \in \mathcal{U}^{-1/3}(x), h(x') = h(x) \\ \perp & \text{otherwise} \end{cases}. \tag{3.2}$$

That is, $\hat{h}$ rejects x if it is within $\epsilon/3$ from h's decision boundary, otherwise accepts and predicts $h(x)$.

Now, consider a clean test sample $(\tilde{x}, \tilde{y})$ and $\tilde{x}$'s adversarial perturbation $\tilde{z}$. Define an intermediate perturbation $\tilde{z}' = \tilde{x} + (\tilde{z} - \tilde{x})/3$. We will show that if h is correct at $\tilde{z}'$, then $\hat{h}$ makes no error at $\tilde{z}$.

If $\tilde{z} = \tilde{x}$, then $\tilde{z}' = \tilde{x} = \tilde{z}$. Since h is $\epsilon/3$-robust at $\tilde{z}'$, $h(\tilde{z}) = h(\tilde{z}') = \tilde{y}$ and so $\hat{h}(\tilde{z}) = \tilde{y}$ which is correct. Otherwise, we need to consider two cases: **(a)** h is $\epsilon/3$-robust at $\tilde{z}$; **(b)** h is not. See visualization in Figure 3.1. In both cases, the $\epsilon/3$-balls about $\tilde{z}$ and $\tilde{z}'$ intersect. Let $\tilde{z}''$ be some point in the intersection. Since h is $\epsilon/3$-robust at $\tilde{z}'$, $h(\tilde{z}'') = h(\tilde{z}') = \tilde{y}$. Now, in case (a) where h is $\epsilon/3$-robust at

$\tilde{z}$, $h(\tilde{z}) = h(\tilde{z}'') = \tilde{y}$, which is correct. In case (b) where h is not $\epsilon/3$-robust at $\tilde{z}$, $\hat{h}$ rejects $\tilde{z}$ and makes no error.

Hence if h is correct at $\tilde{z}'$, then $\hat{h}$ makes no error at $\tilde{z}$. So the error bound for h implies the desired error bound for any $\hat{h}$ in the set

$$\Delta' = \left\{ \hat{h} = F_{\mathcal{U}^{1/3}}(h) : h \in \Delta_{\mathcal{H}}^{\mathcal{U}^{1/3}}(\boldsymbol{x}, \boldsymbol{y}, \tilde{\boldsymbol{z}}') \right\}. \tag{3.3}$$

As we have access only to the adversarial test data $\tilde{\boldsymbol{z}}$, to ensure $\epsilon/3$-robustness at the unknown $\tilde{\boldsymbol{z}}'$, we need to ensure ϵ-robustness at $\tilde{\boldsymbol{z}}$. Let

$$\Delta'' := \cup \left\{ \hat{h} = F_{\mathcal{U}^{1/3}}(h) : h \in \bigcap_{\tilde{\boldsymbol{z}}' \in \mathcal{U}^{-2/3}(\tilde{\boldsymbol{z}})} \Delta_{\mathcal{H}}^{\mathcal{U}^{1/3}}(\boldsymbol{x}, \boldsymbol{y}, \tilde{\boldsymbol{z}}') \right\} \tag{3.4}$$

and let $\hat{\Delta} = \bigcup_{\tilde{\boldsymbol{z}}' \in \mathcal{U}^{-2/3}(\tilde{\boldsymbol{z}})} \Delta_{\mathcal{H}}^{\mathcal{U}^{1/3}}(\boldsymbol{x}, \boldsymbol{y}, \tilde{\boldsymbol{z}}')$. By the above, as $\Delta'' \subseteq \Delta'$, any $\hat{h}$ in Δ'' achieves the desired bound. If $|\hat{\Delta}| = 1$, then $|\Delta'| = 1$ and as $\Delta' \subseteq \hat{\Delta}$, $\hat{\Delta} = \Delta'$ and so any $\hat{h}$ in $\Delta'' \cup \hat{\Delta}$ likewise achieves the bound.

Hence, if we let

$$\Delta = \begin{cases} \Delta'' \cup \hat{\Delta} & |\hat{\Delta}| = 1, \\ \Delta'' & \text{otherwise} \end{cases}, \tag{3.5}$$

we obtain the theorem statement.

3.5 Defense by Transduction and Rejection

The analysis of Theorem 3.1 suggests the following defense algorithm: (1) first obtain a classifier h that is robust and correct on the training data and also robust on the test inputs, (2) then transform h to a selective classifier $\hat{h}$ by rejecting inputs too close to the decision boundary of h. We describe the resulting defense below:

Step (1) To get h, we perform adversarial training on both the training set and the test set, using a robust cross-entropy objective. As in TADV (Chen et al., 2022) we train with private randomness. Specifically, we train a model with softmax output as the class prediction probabilities h^s and the class prediction is $h(x) =$

$\arg\max_{y \in \mathcal{Y}} h^s_y(x)$. Given the labeled training data $(\mathbf{x}, \mathbf{y})$ and the test inputs $\tilde{\mathbf{z}}$, we optimize the following objective:

$$\min_h \left(\begin{array}{l} \frac{1}{n} \sum_{(x,y)\in(\mathbf{x},\mathbf{y})} \left[\begin{array}{l} \mathcal{L}_{CE}(h^s(x), y) \\ + \max_{x' \in \mathcal{U}(x)} \mathcal{L}_{CE}\left(h^s(x'), y\right) \end{array} \right] \\ + \frac{\lambda}{m} \sum_{\tilde{z}\in\tilde{\mathbf{z}}} \left[\max_{\tilde{z}' \in \mathcal{U}(\tilde{z})} \mathcal{L}_{CE}\left(h^s(\tilde{z}'), h(\tilde{z})\right) \right] \end{array} \right) \tag{3.6}$$

where $\mathcal{L}_{CE}$ is the cross-entropy loss and $\lambda > 0$ is a hyper-parameter.

Step (2) Having learned h, we now turn h into a selective classifier $\hat{h}$. To do this, we need to compute the transformation $F_{\mathcal{U}^{1/3}}$, however, the construction following (Tramèr, 2022) is computationally inefficient and as such is not practical for an empirical defense. Hence, we present a simple but effective approach which performs similarly in practice, as we show in Section 3.6.4.

Empirical Classifier to Selective Classifier Transformation: Recall that $\hat{h}$ rejects the input x if there exists $x' \in \mathcal{U}^{1/3}(x)$ with $h(x) \neq h(x')$; otherwise accepts and predicts the label $h(x)$. So we only need to determine the existence of $x' \in \mathcal{U}^{1/3}(x)$ with $h(x) \neq h(x')$.

We use a standard inductive attack, PGD, for this by solving:

$$\arg\max_{x' \in \mathcal{U}^{1/3}(x)} \quad \mathcal{L}_{CE}(h^s(x'), h(x)). \tag{3.7}$$

When $\mathcal{U}$ is ℓ_p norm ball of radius ϵ, the constraint is then $\|x' - x\| \leqslant \epsilon/3$. In practice, we can generalize this to $\|x' - x\| \leqslant \epsilon_{\text{defense}}$ where $\epsilon_{\text{defense}}$ is a hyper-parameter we call the *rejection radius*.

Taken together, we obtain a strong defense with transduction and rejection which significantly outputerforming existing baselines (see Section 3.6.3), which we refer to as **TLDR** (Transductive Learning Defense with Rejection).

Discussion on Computational Cost. The computational cost of training with TLDR is higher than that of standard adversarial training, in particular, by a factor of at most two; the cost of the transformation $F_{\mathcal{U}^{1/3}}$ is the same as that of PGD. As with general transductive defenses, the training process must be repeated for each new batch of samples; hence TLDR is suited to applications with minimal latency

requirements, which may amortize the cost of training over a large batch of test samples, on the order of the full training set.

Discussion on Evaluation. Adversarial evaluations of novel defenses are well known to be challenging (Chen et al., 2022; Zimmermann et al., 2022); hence, we construct an adaptive attack targeting our defense in Section 3.5.1 and thorougly evaluate it in Section 3.6. As we incorporate GMSA (Chen et al., 2022) in our attack, we must perform multiple iterations of training in evaluation, each of which is computationally costly. Hence, attacking and evaluating TLDR is extremely computationally expensive.

3.5.1 Adaptive Attacks

Since no strong adaptive attacks exist for the new transduction+rejection setting to our knowledge, we design one here. Our attack is based on GMSA in (Chen et al., 2022), which has been shown to be a strong attack for transductive defense (without rejection).

The goal of the attack is to find perturbations $\tilde{\mathbf{z}}$ of the clean test inputs $\tilde{\mathbf{x}}$ such that the transductive learner has a large error when given $(\mathbf{x}, \mathbf{y}, \tilde{\mathbf{z}})$. GMSA runs in stages; in each stage t, it simulates the transductive learner on the current data set $(\mathbf{x}, \mathbf{y}, \tilde{\mathbf{z}}_t)$ to get a classifier h_t, and then maximizes the minimum or average loss of $\{h_i\}_{i=1}^t$ to get the updated perturbations of the test inputs $\tilde{\mathbf{z}}_{t+1}$ (called GMSA_{MIN} and GMSA_{AVG}, respectively). See (Chen et al., 2022) for the details.

GMSA does not directly apply to our setting since we have selective classifiers $\hat{h}$ with a rejection option which is not considered in GMSA. Our contribution is to design a method to get the updated perturbations $\tilde{\mathbf{z}}$ of the test inputs in each stage such that the selective classifier incurs a large error. Recall that $\hat{h}$ constructed from h incurs error in two cases: (1) it accepts $\tilde{z}$ and misclassifies with $h(\tilde{z}) \neq y$; (2) $\tilde{z} = \tilde{x}$ and it rejects $\tilde{z}$. We consider the two cases below.

Case (1) We will propose a novel loss measuring the loss of a selective classifier $\hat{h}$ on a perturbation $(\tilde{z}, y)$ from a clean test point $(\tilde{x}, y)$ for such kind of error; maximizing this loss gives the desired $\tilde{z}$. Recall that we need $\tilde{z}$ to be accepted and

also the prediction $h(\tilde{z}) \neq y$. For the latter, we can maximize $\mathcal{L}_{CE}(h^s(\tilde{x}), y)$ where h^s is the class probabilities of h (i.e., its softmax output). The former is equivalent to $\min_{h(\tilde{z}') \neq h(\tilde{z})} \|\tilde{z} - \tilde{z}'\| \geqslant \epsilon_{\text{defense}}$.

Now, suppose $\mathcal{L}_{DB,h}(\tilde{z}')$ is a *surrogate loss* function on the closeness to the decision boundary; it increases when $\tilde{z}'$ gets closer to the decision boundary of h. Then the condition is equivalent to $\|\tilde{z} - p(\tilde{z})\| = \epsilon_{\text{defense}}$ where $p(\tilde{z}) = \arg\max_{\|\tilde{z}' - \tilde{z}\| \leqslant \epsilon_{\text{defense}}} \mathcal{L}_{DB,h}(\tilde{z}')$. Now, as the maximum value of $\|\tilde{z} - p(\tilde{z})\|$ is exactly $\epsilon_{\text{defense}}$, we would like to maximize $\|\tilde{z} - p(\tilde{z})\|$ to satisfy the condition.

Summing up, for this case, we would like to maximize:

$$\begin{aligned} \mathcal{L}_{REJ}(\tilde{z}, y) &:= \mathcal{L}_{CE}(h^s(\tilde{z}), y) + \lambda' \|\tilde{z} - p(\tilde{z})\|, \\ \text{where } p(\tilde{z}) &= \underset{\|\tilde{z}' - \tilde{z}\| \leqslant \epsilon_{\text{defense}}}{\arg\max} \mathcal{L}_{DB,h}(\tilde{z}') \end{aligned} \tag{3.8}$$

and $\lambda' > 0$ is a hyper-parameter. Finally, for $\mathcal{L}_{DB,h}$, the following definition works well in our experiments: $\mathcal{L}_{DB,h}(\tilde{z}') := \text{rank}_2\, h^s(\tilde{z}') - \max h^s(\tilde{z}')$, which is maximized at the decision boundary as the top-two class probabilities are equal.

Case (2) A critical step in an effective application of $\mathcal{L}_{REJ}$ to a transductive attack is the selection of which points to perturb. To do this, we apply a post-processing step after finding $\tilde{z}$ by maximizing (equation 3.8). We must predict whether $\hat{h}$ is more likely to incur error on $\tilde{z}$ or on the clean input $\tilde{x}$ (i.e., $\hat{h}(\tilde{x}) \neq y$). If we expect that the clean point is likely to be incorrectly classified or rejected, then we update $\tilde{z}$ to $\tilde{x}$. In GMSA, we have access to a series of models trained on previous attack iterations; we estimate the likelihood of success at $\tilde{z}$ and $\tilde{x}$ by the fraction of previous models which fail at each point.

Summing up the two cases and combining with GMSA gives our final attack (details in Algorithm 3 in Appendix B.2.5).

3.6 Experiments

This section performs experiments to evaluate the proposed method TLDR and compare it with baseline methods (e.g., those using only rejection or transduc-

tion). Our main findings are: **1**) TLDR outperforms the baselines significantly in robustness, confirming the advantage of combining transduction and rejection. **2**) Our adaptive attack is significantly stronger than existing attacks which were not designed for the new setting, providing a strong evaluation. **3**) Rejection rates rise steadily with the rejection radius, but few clean samples are rejected and the robust accuracy remains stable.

3.6.1 Datasets and Defense/Attack Setup

We evaluate on MNIST (LeCun, 1998) and CIFAR-10 (Krizhevsky et al., 2009). We consider an adversarial budget of $\epsilon = 0.3$ in l_∞ on MNIST and $\epsilon = 8/255$ in l_∞ on CIFAR-10. For defense, on MNIST, we use a LeNet architecture; on CIFAR-10 we use a ResNet-20 architecture. In both cases, we train for 40 epochs with a learning rate of 0.001 using ADAM for optimization. On MNIST, we use 40 iterations of PGD during training with a step size of 0.01. On CIFAR-10, we use 10 iterations of PGD in training with a step size of 2/255. In training TLDR, we set $\lambda = 0.176$ after a warm start period in which $\lambda = 0$. We use a rejection radius of $\epsilon/4$ for selective classifiers. For attack, we use 10 iterations of GMSA on both datasets. On MNIST, we use 200 steps of PGD with a stepsize of 0.01 while generating adversarial examples. On CIFAR-10, the PGD attacks use 100 steps with a stepsize of $1/255$. Defense settings used while training models in GMSA (including internal PGD settings) are the standard defense settings. Internal optimizations in the calculation of $\mathcal{L}_{\text{REJ}}$ use 10 steps of PGD with a stepsize of 15% of the rejection radius. We use $\lambda' = 1$ in $\mathcal{L}_{\text{REJ}}$; we observe little sensitivity to the parameter.

3.6.2 Attack Evaluation

Table 3.3 shows the results of different attack methods on TLDR. Previous work (Chen et al., 2022) shows that transduction-aware attacks are necessary against transductive defenses; we observe that attacks (PGD on $\mathcal{L}_{\text{CE}}$ or $\mathcal{L}_{\text{REJ}}$ and AutoAttack) from the traditional setting perform poorly against our defense. We can also see that GMSA significantly outperforms even a rejection-aware transfer attack (referred to

Table 3.3: Robust accuracy by different attacks on TLDR. The strongest attack is **boldfaced**.

Attack	MNIST	CIFAR-10
PGD ($\mathcal{L}_{\text{CE}}$)	0.991	0.794
PGD ($\mathcal{L}_{\text{REJ}}$)	0.988	0.781
AutoAttack	0.989	0.756
GMSA ($\mathcal{L}_{\text{CE}}$)	0.988	0.853
GMSA ($\mathcal{L}_{\text{REJ}}$)	**0.972**	**0.739**

Table 3.4: Robust accuracy under different attack losses on a fixed adversarially trained model with rejection, AutoAttack for comparison. The strongest attack is **boldfaced**.

Loss	MNIST	CIFAR-10
AutoAttack (Croce and Hein, 2020c)	0.980	0.592
$\mathcal{L}_{\text{CE}}$	0.977	0.524
$\mathcal{L}_{\text{REJ}}(\mathcal{L}_{\text{CE}})$	0.974	0.470
$\mathcal{L}_{\text{REJ}}$	**0.973**	**0.458**

as PGD targeting $\mathcal{L}_{\text{REJ}}$; note that PGD and AutoAttack do *not* target the final model in this case, given the transductive setting, but instead target a proxy trained by the adversary); see Algorithm 4 in Appendix B.2.5 for the full details.

This shows that GMSA is critical for attacking a transductive defender; while PGD and AutoAttack are strong against an inductive model, they performs poorly facing transduction. Finally, we observe that GMSA with $\mathcal{L}_{\text{CE}}$ is much weaker than GMSA with $\mathcal{L}_{\text{REJ}}$. This shows another key component in our adaptive attack, the loss $\mathcal{L}_{\text{REJ}}$, is also critical to get a strong attack against our defense.

To further investigate the importance of $\mathcal{L}_{\text{REJ}}$, we attack an adversarially trained model with rejection with PGD on different losses: $\mathcal{L}_{\text{REJ}}$, cross-entropy $\mathcal{L}_{\text{CE}}$, and $\mathcal{L}_{\text{REJ}}$ with $\mathcal{L}_{\text{DB},h}$ replaced by $\mathcal{L}_{\text{CE}}$, with AutoAttack given for comparison. Table 3.4

Table 3.5: Results on MNIST and CIFAR-10. Robust accuracy is 1 - robust error; see Section 3.3. p_{REJ} is the percentage of inputs rejected. The baseline results are from (Chen et al., 2022). The strongest attack against each defense is shown. The best result is **boldfaced**.

Setting	Defense	Attacker	MNIST		CIFAR-10	
			p_{REJ}	Robust accuracy	p_{REJ}	Robust accuracy
Induction	AT (Madry et al., 2018a)	AutoAttack	–	0.897	–	0.448
Rejection only	AT (with rejection)	PGD ($\mathcal{L}_{REJ}$)	0.852	0.968	0.384	0.634
Transduction only	RMC (Wu et al., 2020c)	GMSA ($\mathcal{L}_{CE}$)	–	0.588	–	0.396
	DANN (Ganin et al., 2016a)	GMSA ($\mathcal{L}_{CE}$)	–	0.062	–	0.055
	TADV (Chen et al., 2022)	GMSA ($\mathcal{L}_{CE}$)	–	0.943	–	0.541
Transduction+Rejection	URejectron (Goldwasser et al., 2020a)	GMSA ($\mathcal{L}_{DISC}$)	0.274	0.721	**0.000**	0.145
Transduction+Rejection	**TLDR (ours)**	GMSA ($\mathcal{L}_{REJ}$)	**0.126**	**0.972**	0.208	**0.739**

Table 3.6: Comparison with state-of-the-art (Peng et al., 2023; Wang et al., 2023b; Croce et al., 2020) on CIFAR-10 and CIFAR-100 under l_∞ perturbations with budget 8/255. The best result is **boldfaced**.

Setting	Defense	Architecture	Attacker	CIFAR-10		CIFAR-100	
				p_{REJ}	Robust accuracy	p_{REJ}	Robust accuracy
Induction	(Peng et al., 2023)	Ra WideResNet-28-10	AutoAttack	–	0.651	–	0.372
Induction	(Peng et al., 2023)	Ra WideResNet-70-16	AutoAttack	–	0.711	–	0.388
Induction	(Wang et al., 2023b)	WideResNet-28-10	AutoAttack	–	0.673	–	0.388
Induction	(Wang et al., 2023b)	WideResNet-70-16	AutoAttack	–	0.707	–	0.427
Transduction+Rejection	**TLDR (ours)**	ResNet-20	GMSA ($\mathcal{L}_{REJ}$)	0.208	0.739	–	–
Transduction+Rejection	**TLDR (ours)**	WideResNet-28-10	GMSA ($\mathcal{L}_{REJ}$)	0.111	**0.816**	0.171	**0.579**

shows that $\mathcal{L}_{REJ}$ significantly outperforms both PGD targeting alternative losses and AutoAttack. See Appendix B.3 for an evaluation of the effectiveness with which $\mathcal{L}_{REJ}$ targets rejection using the binarization test (Zimmermann et al., 2022).

3.6.3 Robustness of TLDR

Baselines. (1) AT: adversarial training (Madry et al., 2018a); (2) AT (with rejection): adversarial training (AT) with rejection; (3) RMC (Wu et al., 2020c); (4) DANN (Ganin et al., 2016a); (5) TADV (Chen et al., 2022); (6) Rejectron (Goldwasser et al., 2020a). Among them, (1) is in the traditional induction setting, (2) is rejection only, (3)(4)(5) are transduction only, and (6) incorporates both transduction and rejection.

Evaluation. We attack the defenses and report the robust accuracy (1 - the robust error defined in Section 3.3). To attack inductive classifiers, we use AutoAttack (Croce and Hein, 2020c). For inductive selective classifiers, we use PGD on the rejection-aware loss $\mathcal{L}_{\text{REJ}}$ from Eqn (3.8). For transductive classifiers, we use GMSA which has been shown to be a strong adaptive attack on transduction (Chen et al., 2022). Finally, for our transductive selective classifiers, we use our adaptive attack in Section 3.5.1 (roughly GMSA with $\mathcal{L}_{\text{REJ}}$). For Rejectron (Goldwasser et al., 2020a) we use GMSA with a loss function $\mathcal{L}_{\text{DISC}}$ targeting their defense; see Appendix B.2.6 for the details. We include an ablation of the two core components of TLDR (the transductive loss term and the transformation into a selective classifier) in Appendix B.3.

For transductive models, we report the stronger of GMSA_{MIN} and GMSA_{AVG}. Inductive models are trained with standard adversarial training (Goodfellow et al., 2015b), and transductive models with the TLDR loss in Eqn (3.6). As Rejectron depends heavily on a key hyperparameter determining confidence needed to reject, we report the results for the parameter value strongest against our attack. The best-performing value on CIFAR-10 effectively eliminated the possibility of rejection (hence the rejection rate of 0); other choices resulted in near-0 robust accuracy.

Comparison of Defenses. Table 3.5 shows the robust accuracy and rejection rate of different methods. We observe that either transduction or rejection can improve the performance, while combining both techniques leads to the best results. In particular, our defense outperforms existing transductive defenses such as RMC and DANN. Results for l_2 perturbations are given for l_2 in Appendix B.3. See Table 3.6 for a comparison to the state-of-the-art. With a much smaller ResNet-20 architecture, TLDR outperforms the strongest existing baseline on CIFAR-10, and, with a WideResNet-28-10 architecture, we obtain an in robust accuracy; on CIFAR-100 of over 10%, we obtain an improvement in robust accuracy of over 15%.

Discussion on Evaluation. As our key focus is on demonstrating the potential advantages of one setting (transduction+rejection) over others, comparisons between settings are necessary. In each setting, robust accuracy represents the same

Table 3.7: Comparison of our rejection-only defense with budget ϵ to induction-only defenses with budget $\epsilon/2$.

Dataset	Model	Defense	Attacker	ϵ (training)	ϵ (attack)	R
CIFAR-10	Resnet-20	AT (Madry et al., 2018a)	PGD ($\mathcal{L}_{\text{CE}}$)	8/255	8/255	
CIFAR-10	Resnet-20	AT (Madry et al., 2018a)	PGD ($\mathcal{L}_{\text{CE}}$)	4/255	4/255	
CIFAR-10	Resnet-20	**AT (with rejection) [ours]**	PGD ($\mathcal{L}_{\text{REJ}}$)	4/255	8/255	
CIFAR-10	WideResnet-28-10	AT (Madry et al., 2018a)	PGD ($\mathcal{L}_{\text{CE}}$)	8/255	8/255	
CIFAR-10	WideResnet-28-10	AT (Madry et al., 2018a)	PGD ($\mathcal{L}_{\text{CE}}$)	4/255	4/255	
CIFAR-10	WideResNet-28-10	**AT (with rejection) [ours]**	PGD ($\mathcal{L}_{\text{REJ}}$)	4/255	8/255	
CIFAR-100	WideResNet-28-10	AT (Madry et al., 2018a)	PGD ($\mathcal{L}_{\text{CE}}$)	8/255	8/255	
CIFAR-100	WideResnet-28-10	AT (Madry et al., 2018a)	PGD ($\mathcal{L}_{\text{CE}}$)	4/255	4/255	
CIFAR-100	WideResNet-28-10	**AT (with rejection) [ours]**	PGD ($\mathcal{L}_{\text{REJ}}$)	4/255	8/255	

concept, the fraction of samples on which we are correct. The difference between settings lies in their different notions of "correctness"; each concept of correctness incorporates both the potential advantages and the disadvantages of each setting, e.g. in the rejection case, a new type of error is possible: rejecting a clean sample. Hence, we compare the fraction of samples on which we can be correct between settings (and between defenses in the same setting).

3.6.4 Rejection-Only Defense

(Tramèr, 2022) shows that the existence of a classifier with x robust accuracy with adversarial budget ϵ implies the existence of a selective classifier with x robust accuracy with adversarial budget 2ϵ; however, as the construction used is computationally inefficient, this has not yet been realized in practice. Table 3.7 evaluates our rejection-only defense by comparing its results on the full adversarial budget (8/255) to the theoretical bound obtained by a classifier with the half-budget of 4/255. In each case, our empirical transformation results in a robust accuracy is very close to the results obtained by Tramèr's idealized computationally inefficient approach. In this way, our approach enables practical realization of Tramèr's upper

bound on gains from rejection in the inductive case.

3.7 Conclusion

Existing works on leveraging transduction and rejection gave mixed results on their benefits for adversarial robustness. In this work we take a step in realizing their promise in practical deep learning settings. Theoretically, we show that a novel application of Tramèr's results give improved sample complexity for robust learning in the bounded perturbations setting. Guided by our theory, we identified a practical robust learning algorithm leveraging both transduction and rejection. Systematic experiments confirm the benefits of our constructions. There are many future avenues to explore, such as improving the theoretical bounds, and improving the efficiency of our algorithms.

Part II

Model Adaptation in Large Language Models

4 HUMOR IN AI: MASSIVE SCALE CROWD-SOURCED PREFERENCES AND BENCHMARKS FOR CARTOON CAPTIONING

Building on the earlier exploration of model adaptation for robustness, we now shift our focus to examining adaptation from an application-driven perspective, particularly within Large Language Models (LLMs). Adaptation in LLMs occurs prominently at two levels: *across-context adaptation* (fine-tuning) and *within-context adaptation* (in-context learning). The pretraining-finetuning paradigm enables efficient adaptation to customized datasets without incurring substantial pretraining costs, while in-context learning has emerged as a fundamental capability behind LLMs' empirical successes (Brown et al., 2020; Lu et al., 2023). In the next two chapters, we will explore two concrete applications highlighting each of these adaptation methods.

In this chapter, we explore a concrete application of across-context adaptation through finetuning LLMs for creative generation tasks. Creative generation presents unique challenges for language models, particularly in fostering diversity, originality, and humor. To address these, we introduce a large-scale multimodal preference dataset derived from The New Yorker's cartoon caption contests, comprising over 250 million human ratings. This rich dataset not only enables comprehensive evaluation of current preference-based fine-tuning methods but also exposes key limitations in established techniques such as RLHF and Direct Preference Optimization (DPO). Our findings highlight gaps between human and model capabilities, underscoring important avenues for future research in enhancing AI-driven creative expression.

4.1 Introduction

This paper introduces a dataset and benchmark designed to investigate alignment in Large Language Models (LLMs). The dataset comprises a collection of over a quarter of a billion human ratings, curated from the New Yorker's funny cartoon

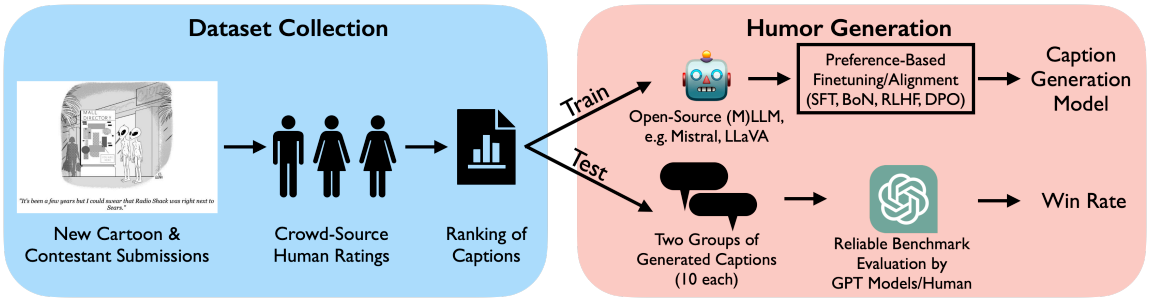


Figure 4.1: Overview of our workflow. During data collection, a new cartoon is released each week and thousands of captions are submitted. We then collect caption ratings through a crowd-sourcing procedure driven by a bandit algorithm. Our dataset is a collection of 365 contests, over 2.2M captions and over 250M human ratings. This dataset is utilized for our Humor generation task and benchmark. We experiment with finetuned open-source models and close-sourced API calls (both LLMs and MLLMs). Our novel and low-cost evaluator provides better reliability in evaluating captions.

caption contest. The task of writing funny captions comes with significant challenges due to subjectivity and variability in human judgments. This benchmark represents a distinctive challenge for AI alignment, mirroring the complexities found in other tasks where expert humans consistently outshine state-of-the-art (SOTA) AI systems. Our exploration of alignment in LLMs through this dataset sheds light on fundamental questions regarding evaluation and alignment with human values and preferences.

Specifically, the paper studies the multifaceted realm of humor expression in LLMs, probing whether these models can effectively recognize humor and generate genuinely amusing captions resonating with human audiences. While LLMs may not have been explicitly tailored for humor, their exposure to diverse comedic content during training suggests an inherent capacity for humor recognition and expression within their computational repertoire. We propose a robust and scalable benchmark for evaluating a model's humor capabilities that leverages advanced systems like GPT4o to potentially exceed human performance. This research aims to foster the development of more reliable, capable, and aligned AI systems, ultimately advancing the field of AI alignment and its practical applications.

Our main contributions and findings are summarized as follows.

Dataset: The paper presents a pioneering dataset, consisting of massively crowd-sourced ratings of human-generated cartoon captions from The New Yorker's famous weekly cartoon caption contest. This dataset provides a unique opportunity to delve into the intricacies of humor generation in LLMs and marks the first large-scale dataset with human judgments for evaluating creative tasks. With over a quarter of a billion human ratings, this dataset offers a rich and diverse collection of examples, enabling researchers to explore the nuances of humor expression and perception in AI systems like never before. The dataset has already catalyzed groundbreaking research; a portion of it was shared with other researchers who utilized it in a recent award-winning paper at ACL 2023 Hessel et al. (2022). By curating such a vast dataset, the paper not only facilitates deeper insights into the capabilities of LLMs but also lays the groundwork for future studies in this emerging field.

Benchmark: In addition to the dataset, the paper introduces novel metrics specifically tailored for evaluating the quality of humor generated by LLMs, leveraging cutting-edge techniques such as GPT. These metrics provide researchers with a standardized framework for assessing humor in AI-generated content, enabling precise and objective evaluations compared to traditional subjective methods. By establishing this benchmark, the paper not only enhances our understanding of humor in AI systems but also paves the way for more rigorous and systematic evaluations in the future.

Generative Capabilities of SOTA Models: Using the newly developed benchmark, the paper evaluates the performance of SOTA models such as GPT4o and Claude in generating humorous content relative to human-generated examples. By comparing the outputs of these models against human-generated humor, the paper offers valuable insights into the current capabilities and limitations of LLMs in the domain of humor generation. This analysis not only highlights areas where AI systems excel but also identifies areas for improvement, guiding future research efforts in enhancing the humor generation capabilities of LLMs.

Alignment Strategy Evaluation: Furthermore, the paper utilizes the benchmark to assess the effectiveness of various alignment strategies, including Reinforcement

Learning from Human Feedback (RLHF), Direct Preference Optimization (DPO), and Best-of-N sampling (BoN), in enhancing humor generation in LLMs. By systematically comparing the performance of these alignment strategies against baseline models, the paper sheds light on their relative strengths and weaknesses, informing the development of more effective approaches for aligning AI systems with human preferences and values. This analysis not only advances our understanding of alignment techniques but also provides practical insights for improving the humor generation capabilities of LLMs in real-world applications.

In summary, the paper's findings and contributions significantly advance LLM capabilities in humor evaluation and generation by providing a comprehensive dataset, establishing a benchmark for evaluation, and offering insights into the performance of SOTA models and alignment strategies. By addressing these key aspects, the paper not only enhances our understanding of humor in AI systems but also lays the foundation for future research and development in this rapidly evolving field. We open-source our dataset and code as linked in Appendix C.1.

4.2 Related Work

New Yorker Caption Contest. Since its original conception as part of the NEXT crowdsourcing system Jamieson et al. (2015); Sievert et al. (2017), the New Yorker Caption Contest Dataset has been updated on a weekly basis for the last several years. During this time, the dataset has been primarily used for the evaluation of online algorithms and, similar to this work, to study the nature of humor. Works in the former camp include Mason et al. (2020); Tanczos et al. (2017); Yang et al. (2017). Perhaps the most relevant work to ours is Hessel et al. (2022). They formulated three tasks, matching, quality ranking and explanation generation for studying whether current AI systems *understand* humor. Additional prior work includes Shahaf et al. (2015); Radev et al. (2015); King et al. (2013), which utilize judgements made by the editors of the New Yorker directly to analyze a smaller number of contests (< 50) and attempt to identify features that correlate with caption performance such as length, perplexity, readability and sentiment.

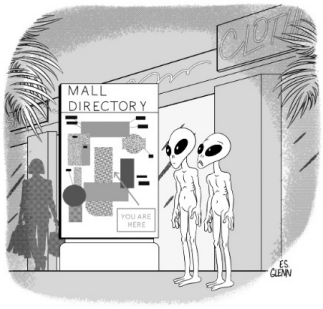
Alignment of LLMs. Finetuning of LLMs has proved a critical step in aligning the behavior of pretrained models to downstream tasks. A standard pipeline is to first finetune the pretrained model via *supervised fine-tuning* (SFT)—to imitate expert demonstrations—followed by *reinforcement learning from human feedback* (RLHF) (Christiano et al., 2017)—where a reward model is trained on human preferences, and then the SFT model is trained to maximize this reward via PPO (Schulman et al., 2017). This pipeline has been successfully applied for finetuning frontier models (Ziegler et al., 2019; Bai et al., 2022a; Ouyang et al., 2022; Touvron et al., 2023), and has inspired a vast amount of follow-up work refining and extending the SFT (Yuan et al., 2023; Zhao et al., 2023; Gulcehre et al., 2023; Mukobi et al., 2023; Ethayarajh et al., 2024) and RLHF (Bakker et al., 2022; Dumoulin et al., 2023; Liu et al., 2023; Siththaranjan et al., 2023; Munos et al., 2023; Swamy et al., 2024; Chakraborty et al., 2024; Chang et al., 2024) methodologies. *Direct preference optimization* (DPO) methods (Rafailov et al., 2024) have recently emerged as a simpler yet still effective replacement to the RLHF paradigm. Instead of training a reward model and then optimizing this reward, DPO combines these steps by directly optimizing the SFT model on offline human preference data, and has inspired a variety of extensions (Hejna et al., 2023; Azar et al., 2024; Song et al., 2024; Rosset et al., 2024; Tang et al., 2024; Yin et al., 2024). While the aforementioned works focus on finetuning on human feedback, a related line of works has sought to finetune on AI-generated feedback (Yang et al., 2023; Xu et al., 2023; Lee et al., 2023; Burns et al., 2023; Chen et al., 2024b; Yuan et al., 2024; Bai et al., 2022b). Despite extensive research into various fine-tuning methodologies, understanding their effectiveness remains nascent. While several studies have explored when and why different methods are most effective (Gao et al., 2023; Kirk et al., 2023; Wu et al., 2024; Chan et al., 2024; Zhou et al., 2024; Sharma et al., 2024), they primarily address standard tasks like reducing harmfulness and increasing helpfulness, and do not assess fine-tuning methods for tasks requiring creativity, the focus of this work.

RLHF Datasets. Existing Reinforcement Learning with Human Feedback (RLHF) datasets, consisting of various responses to a prompt along with a preference ordering of those responses, have been critical for aligning existing AI systems to

human preferences. We briefly review some of the most popular ones. Anthropic's HH-RLHF dataset (Bai et al., 2022a) consists of chosen and rejected texts focusing on helpfulness and harmlessness. Stanford's SHP Dataset (Ethayarajh et al., 2022) and Stack Exchange preferences dataset Askell et al. (2021) have aggregated questions and answers along with their ratings from various online platforms. OpenAI's summarization dataset (Hu et al., 2022) includes rankings of paired answers derived from human evaluations of text summaries. The data comes from a variety of sources, such as news articles and scientific papers, where human annotators compare the quality, coherence, and relevance of two different AI-generated summaries for the same text. The WebGPT comparisons (Nakano et al., 2021) offer a dataset of human comparisons of AI-generated web search results, emphasizing the importance of high-quality, relevant information retrieval. Finally, we mention the Nectar dataset (Zhu et al., 2023), which consists of a large series of prompts along with a list of five answers generated by various LLM's along with a ranking of these prompts by GPT-4.

4.3 New Yorker Caption Contest

Every week The New Yorker publishes an uncaptioned cartoon and solicits humorous captions from its readers through their website. The cartoon editors then review this list of captions and choose the top three funniest ones according to their judgement. The contest began in 2005, and at the time this work was written, there have been roughly 900 contests. For the last eight years, starting with contest 530, the New Yorker has utilized an online crowdsourced rating system (see Figure 4.2) where users are presented with captions and can rate whether the caption is funny (a reward of 3), somewhat funny (a reward of 2), or unfunny (a reward of 1). Each week a large number of captions are submitted (on average more than 6,000). These captions are first filtered by the New Yorker's editorial staff to remove captions that are not humorous or include personal information and/or offensive content, and then are sent to the crowdsourcing platform for large-scale rating. Finally, the New Yorker editors make their final decisions based on the crowdsourced ratings.



"It's been a few years but I could swear that Radio Shack was right next to Sears."

UNFUNNY | SOMEWHAT FUNNY | FUNNY

Figure 4.2: Example voting page for contest 895

Table 4.1: Dataset statistics

Number of contest	365
Average #captions/contest	6044
STD #captions/contest	1794
Total number of ratings	284,183,913
Average #ratings/contest	778,586
STD #ratings/contest	325,156
Max #ratings/contest	2,249,813
Min #ratings/contest	31,173
Average rating	1.214($\pm$0.12)
Top 10 average rating	1.824 ($\pm$0.15)

The rating process utilizes a multi-armed bandit-based algorithm, namely a UCB-variant (see Jamieson et al. (2015); Tanczos et al. (2017) and Appendix C.4 for details), to present users with higher-performing captions more frequently in order to efficiently identify the best caption. Additionally, since many of the captions are unfunny, this keeps the rating engaging by presenting users interesting captions to rate compared to random sampling. On average the contest receives close to 780,000 ratings per week. The top 5% of captions receive an average of 821 ratings, and the bottom 50% of captions receive around 85 ratings.

The crowdsourced voting system for the New Yorker Caption Contest (NYCC) has resulted in an extensive dataset on human preferences and is a key contribution of this work. The dataset can be accessed at `https://huggingface.co/datasets/yguooo/newyorker_caption_ranking`. It consists of the cartoons, captions, and ratings for each one of 365 contests from contests 530 to 895. It provides an extensive labeled dataset on humor for researchers across multiple domains to study. In the related works, we describe some other works that have utilized this dataset. See Table 4.1 for more dataset statistics.

4.4 HumorousAI Benchmark: Funny Cartoon Caption Generation

In this section, we establish a benchmark method for evaluating the ability of large language models to generate funny captions. We start by describing the tasks in Section 4.4.1 followed by our proposed evaluation methods described in Section 4.4.2. Lastly, in Section 4.4.3, we give a brief overview of the various finetuning methods we explore in this paper.

4.4.1 Task

We focus on the cartoon captioning task in this paper, where a model is given the information about the cartoon and is asked to generate funny captions about it. Specifically, we evaluate both multimodel large language models (MLLMs) and language-only models (LLMs). For MLLMs, we provide the raw cartoon images. For language-only models, we instead provide the descriptions and object entities of the cartoons. The text format of these descriptions is either written by human (Hessel et al., 2022) or generated by MLLMs by given the images (see Appendix C.2.1 for details). See Table C.1 for the example descriptions.

We hold out a set of 91 out of the 358 contests for evaluation by an evaluator (see Section 4.4.2). For each contest and its corresponding cartoon, we ask the language model to generate ten captions. This group of ten captions is then compared against four groups of past human submissions by the evaluator. For each contest, the four groups are captions ranked #1-10, #200-209, #1000-1009 and the ten captions that received median ranking. The evaluations are conducted along three dimensions:

1. **Overall comparison**: In this setting, the evaluator compares the overall funniness of the group of model-generated captions against each group of contestant-submitted captions. Win rates of the model-generated captions will be reported in Section 4.5 and Table 4.3.

2. **Best pick comparison**: We ask the evaluator to first pick the funniest caption

Table 4.2: **Evaluation reliability measure**: Ranking accuracy of captions ranked #1-10 vs captions ranked #1000-1009 averaged over 200 pairs. See Appendix C.2.1 for details on how the cartoon descriptions are generated.

Comparison Method	Evaluator	Description/Image	Ranking Accuracy(%)
Pairwise	Human (worker)	GPT4o-vision	61.67±3.45
	Human (worker)	Cartoon Image	60.79±3.46
	GPT4-Turbo-vision	Cartoon Image	61±3.46
	GPT4o-vision	Cartoon Image	60.5±3.47
	GPT4o	GPT4o-vision	65±3.38
	GPT4-Turbo	**GPT4o-vision**	**67±3.33**
	GPT4-Turbo	GPT4-vision	66±3.36
	GPT4-Turbo	Hessel et al. (2022)	66.5±3.35
Group (Overall)	Human (worker)	GPT4o-vision	59.23±1.45
	Human (worker)	Cartoon Image	57.5.42±1.37
	Human (expert)	Cartoon Image	94.28±2.79
	GPT4-Turbo-vision	Cartoon Image	63±3.42
	GPT4o-vision	Cartoon Image	74±3.11
	GPT4-Turbo	GPT4o-vision	73±3.15
	GPT4-Turbo	GPT4-vision	74±3.11
	GPT4-Turbo	**Hessel et al. (2022)**	**77.5±2.96**
Group (Best Pick)	Human (worker)	GPT4o-vision	56 ± 2.22
	Human (worker)	Cartoon Image	63.66±1.96
	GPT4o-vision	**Cartoon Image**	**70.5±3.23**
	GPT4-Turbo	GPT4o-vision	61.5±3.45
	GPT4-Turbo	Hessel et al. (2022)	60±3.47

from each of the two groups and then choose the funnier caption accordingly. Win rates are reported similarly to above.

3. **Caption diversity**: We measure the diversity of captions within each group of captions either generated by language models or submitted by human contestants in the past. Similarly to the study (Kirk et al., 2023) on measuring the output diversity for non-creative tasks (summarization and instruction following), we use the expectation-adjusted distinct N-grams (denoted as **Average EAD**) (Li et al., 2015) and the Sentence-BERT embedding cosine similarity (denoted as **SBERT**) (Reimers and Gurevych, 2019) to measure

the per-contest diversity. **Average EAD** measures the token-level similarity of the generated captions, while **SBERT** measures the semantic-level similarity. We do not use the NLI diversity from (Stasaski and Hearst, 2022) as it is conversation-specific.

Our evaluation primarily focuses on comparing groups of captions since evaluation reliability can be significantly improved as we now discuss below.

4.4.2 Evaluation Method

> *Humor is notoriously subjective. Humans cannot infallibly predict what other humans will find funny. If they could, no joke would ever fall flat. We just do the best we can, always hoping we can do better. Likewise for these models.*
> –Bob Mankoff, former cartoon editor of The New Yorker

In this section, we aim to find a comparably reliable evaluation method for judging model-generated captions against human submissions. We experimented with various versions of GPT-4 and also human evaluations from Prolific (Palan and Schitter, 2018). This task has been studied widely before within the context of humor (Shahaf et al., 2015; Radev et al., 2015; King et al., 2013; Hessel et al., 2022). However, unlike these previous studies that only evaluate two candidate captions at a time (denoted by **Pairwise**), we introduce the novel group comparison techniques for evaluation (denoted by **Group Overall** and **Group Best Pick**). As described in Section 4.4.1, we compare groups of ten captions from different sources, such as human submissions from different ranking levels, or captions generated by different language models. To measure the reliability of different evaluators, as reported in Table 4.2, we compare their accuracy in judging human-submitted captions from top #10 versus #1000-1009 across 200 different contests. For the **Pairwise** comparisons, we uniformly at random choose one caption from each of the two groups, which exactly corresponds to the *ranking* task proposed by Hessel et al. (2022). For group comparisons, we provide all ten captions from each group to a single query to an LLM/human rater. The detailed prompts can be found in Appendix C.2 for various

language models. All of the prompts for evaluation utilize the 5-shot in-context prompting technique, which provides five caption comparison examples from other contests before asking the model to rank the pair/groups of captions for the given cartoon.

As shown in Table 4.2, language models are generally more accurate in detecting the higher-ranked favorable captions in a group comparison paradigm compared to the pairwise paradigm. These models also outperform average humans (crowd workers) in judging the funniness across all three comparison settings. Notably, in the overall group comparisons we also included evaluations from a human expert (the former cartoon editor for The New Yorker). The expert significantly outperforms all other evaluators (AI and human), exposing a significant gap between human experts and SOTA AI systems in this domain. Also, the group comparisons are somewhat more challenging for crowd workers than pairwise comparisons, but group comparisons make the language model evaluations much more reliable and accurate. Further details about the evaluations can be found in Appendix C.3.

In conclusion, we establish two benchmark evaluation methods for the rest of this paper: **Group Comparison (Overall)** using GPT4-Turbo as evaluator with descriptions from Hessel et al. (2022) and **Group Comparison (Best Pick)** using GPT4o-vision as evaluator with raw cartoon images.

4.4.3 Alignment Finetuning Methods

In our study, we compare the performance of a 0-shot model (with standard and Best-of-N sampling) to that of an SFT finetuned model, an RLHF finetuned model, and a DPO finetuned model. We briefly outline these methods here, and refer the reader to (Christiano et al., 2017; Bai et al., 2022a; Ouyang et al., 2022; Rafailov et al., 2024) for further details. In all cases, we adopt the implementation from the TRL package (von Werra et al.).

Supervised Finetuning (SFT): SFT assumes access to a dataset $\mathcal{D}_{\text{sft}} = \{(x^{(i)}, y^{(i)})\}_{i=1}^{N}$ of prompt-completion pairs, where $y^{(i)}$ is assumed to be an "expert" completion for prompt $x^{(i)}$. SFT then tunes the weight of the base model to maximize the

likelihood of completions $y^{(i)}$ given prompt $x^{(i)}$.

Reinforcement Learning from Human Feedback (RLHF): RLHF assumes access to a preference dataset $\mathcal{D}_{\text{pref}} = \{(x^{(i)}, y_w^{(i)}, y_l^{(i)})\}_{i=1}^M$, where $x^{(i)}$ is a prompt, and $y_w^{(i)}, y_l^{(i)}$ two possible completions to $x^{(i)}$, where $y_w^{(i)}$ is preferred over $y_l^{(i)}$. RLHF assumes these preferences are consistent with an (unknown) reward function $r^\star$, typically assumed to follow the Bradley-Terry model (Bradley and Terry, 1952). It first trains a reward model $\widehat{r}$ on $\mathcal{D}_{\text{pref}}$, and then finetunes the base language model to maximize $\widehat{r}$, typically running PPO (Schulman et al., 2017) and regularizing the training to ensure it does not deviate significantly from the SFT model.

Direct Preference Optimization (DPO): DPO operates under the same assumptions as RLHF, but skips the reward modeling step entirely, and instead finetunes the base language model on $\mathcal{D}_{\text{pref}}$ directly, tuning it to produce next-token likelihoods with orderings consistent with $\mathcal{D}_{\text{pref}}$.

Best-of-N Sampling (BoN): Best-of-N sampling does not modify the weights of the base model. Instead, it samples N completions from the base model for any prompt x, and chooses the completion with the highest reward, as quantified by the reward $\widehat{r}$ obtained from the RLHF reward-learning step.

Preference Dataset Construction: In our setting, we take $\mathcal{D}_{\text{sft}}$ to be a dataset of cartoon-caption pairs, where the captions $y^{(i)}$ are drawn at random from the entire training set of captions for cartoon $x^{(i)}$. $\mathcal{D}_{\text{pref}}$ is constructed by taking a cartoon $x^{(i)}$ and then two captions $y_w^{(i)}$ and $y_l^{(i)}$, where $y_w^{(i)}$ is set to a caption with a higher human rating then $y_l^{(i)}$. Specifically, we sample the pair to be at least 3 standard deviation apart from each other, i.e.

$$\text{Rating}(y_w^{(i)}) - \text{Rating}(y_l^{(i)}) \geqslant 3 \cdot \sqrt{\text{STD}(y_w^{(i)})^2 + \text{STD}(y_l^{(i)})^2}, \tag{4.1}$$

where $\text{Rating}(y)$ is the average score of caption y from human raters according to rewards defined in Section 4.3 (note this is different from the rewards from the reward model of RLHF). $\text{STD}(y)$ is the corresponding standard deviation of scores from human raters.

Table 4.3: Evaluation of captions generated by various language models. We utilize group comparison strategies mentioned in Section 4.4.2. The generated captions are compared against four groups of human contestant entries at different ranking levels. Win rates are based on 91 held-out cartoons.

Generated Caption Model	Overall Win Rate (%) ↑				Best Pick Win Rate (%) ↑			
	Top 10	#200-#209	#1000-#1009	Contestant Median	Top 10	#200-#209	#1000-#1009	Contestant Median
LLaVA	3.85	2.20	4.40	13.19	2.75	6.59	4.95	12.64
LLaVA SFT	2.75	3.30	7.14	17.03	2.20	4.95	6.59	10.99
Mistral-7B 0-Shot	4.95	8.79	11.54	25.82	1.65	1.65	3.85	12.64
Mistral-7B BoN	6.59	**16.48**	**21.43**	**35.71**	1.65	2.20	3.30	10.44
Mistral-7B SFT	3.85	4.40	7.14	14.29	0.55	2.20	1.65	8.24
Mistral-7B RLHF	8.79	9.34	11.54	24.73	2.20	3.30	8.24	13.19
Mistral-7B DPO	**9.34**	13.74	17.58	31.32	**10.44**	**15.93**	**14.29**	**30.22**
GPT-3.5 Turbo	33.52	52.75	62.09	76.92	23.63	46.7	48.35	70.88
GPT-4o	44.51	69.23	79.12	86.81	42.86	59.89	73.63	79.67
GPT-4o Vision	42.31	63.74	76.92	85.16	**47.80**	**65.93**	**79.67**	**85.71**
Claude-3-Opus	**54.40**	**70.88**	**81.87**	**88.46**	40.11	59.89	63.74	79.67

4.5 Experiments

In this study, we evaluate the performance of caption generation. We experiment with two open-source large language models, Mistral 7b Instuct (mistralai/Mistral-7B-Instruct-v0.1)(Jiang et al., 2023) and the multimodal model LLaVa 7b (llava-hf/llava-v1.6-mistral-7b-hf)(Liu et al., 2024) finetuned with methods in Section 4.4.3. We also evaluate state-of-the-art close-sourced models including GPT4o and Claude 3 Opus. See Appendix C.3.3 for more details. Our code is available at `https://github.com/yguooo/cartoon-caption-generation`.

4.5.1 Experimental Results

In Table 4.3, we report the result for pretrained and finetuned model generations evaluated by GPT models. In Table 4.4, we ask human workers and expert to evaluate the captions generated by SOTA models. Below, we document some of

our findings and research questions they inspire.

MLLMs vs LLMs. Surprisingly, language-only models such as the pretrained Mistral model outperform the multimodal LLaVa model that has access to the entire cartoon images. Similarly, for overall group comparison, GPT-4o is also preferred over GPT-4o with vision. One possible reason is that the training corpus for multimodal LLMs can be much less diverse than the training corpus for the LLM. For example, LLaVa is only trained on a small multi-modal instruction following dataset (~80K unique images) (Liu et al., 2024), whereas generic LLMs like Mistral or Llama are trained on much larger dataset.

Proposed Research Question #1: The multimodal large language models still underperform their language-only counterparts in caption generation. Can the vision-language integration in MLLMs be further improved to close this gap?

Finetuning Open Source Models. We observe that supervised fine-tuning hurts the model performance in the humor generation task in general. We believe this is primarily because we are aligning to captions in the top 1000, most of which are not particularly funny. However, we note this is an important step before RLHF and DPO training, as it trains the models to generate captions in the correct format. We also find that BoN sampling is able to substantially increase the Overall Win Rate metric, but falls short on the Best Pick Win Rate, which suggests the reward model is favoring a small set of good captions, but none of which generates particularly outstanding captions. We also observe in the next section that BoN indeed results in a less diverse group of generations.

As compared to BoN, running RLHF on the same reward model is unable to achieve as high a level of performance. As we show in Section C.5, running PPO does indeed yield generations with higher reward score as given by the reward model, and as our BoN results indicate, filtering captions based on their reward does give better performance. This suggests that, while our reward model is able to effectively filter generations, tuning a model to maximize it does not necessarily lead to improved performance. We hypothesize that this is due to the complex nature of humor and the potential for out-of-distribution generations when running RLHF.

While our reward model may effectively rank captions within a set of reasonable and in-distribution captions (for example those generated by the 0-shot model), small deviations from the training distribution could lead to an erroneous reward signal. Furthermore, for tasks such as humor generation very subtle changes (for example, minor changes in word choice) can drastically change how humorous a caption is—the distribution of humorous captions is extremely sensitive. Together, we believe these phenomenon make it challenging for PPO to effectively finetune the weights to obtain significantly more humorous generations.

Proposed Research Question #2: Can we train a reward model able to better capture humor? Can RLHF still be effectively applied to settings where the distribution of correct responses is highly sensitive?

In contrast to RLHF, DPO does yield a significant increase over the 0-shot model for the Best Pick Win Rate metric. Note that DPO only optimizes the model on offline preference data and, as such, does not require an evaluation of any out-of-distribution samples. We hypothesize that, in settings such as humor generation where the desired distribution is extremely sensitive, this could lead to better performance, as it avoids the aforementioned issue where RLHF may quickly drift to producing out-of-distribution samples, for which the reward signal is erroneous.

Proposed Research Question #3: Does DPO lead to better in-distribution generation, and produce a model more effectively able to match the distribution of the finetuning data?

Human Evaluation. We also ran a human evaluation using six workers from Prolific (Palan and Schitter, 2018) along with a humor expert (a former New Yorker editor) to understand how often people preferred caption generations from Claude vs top 10 ranked captions generated by humans. We find

Table 4.4: Rate of Claude-3-Opus generated captions preferred over Human Top 10.

Evaluator	Preference Rate
Human (expert)	**1.6%**
Human (worker)	**34%**

that people only prefer Claude's generations 34% of the time. Our expert preferred Claude's generation only 1.6% of the time. He said, *"I think I preferred human captions because from my "expert" vantage point they were better phrased and more concise even independent from being funny. At this point AI tends to be too verbose in almost any task but, for me that is a liability when it comes to creating a good caption."*

This suggests that, though SOTA LLMs can generate a diverse set of funny captions, there remains a significant gap in their humor and creativity when judged from the perspective of human experts.

Example Generation. As shown in Table 4.5, we provide some generation samples. Indeed, we see that LLMs generally produce longer and more verbose captions than top human ones.

4.5.2 Diversity Evaluation

We evaluated the token-level and semantic-level diversity of the generation with results given in Table 4.6. We found the Average EAD and SBERT share the same trend when the base model is the same. Within the human generated caption group, we noticed that their diversity scores are very similar under both metrics. And the human generated texts regardless of their funniness are much more diverse than any model-generated captions.

For pretrained models, the commercial models like GPT, Claude-3 generally outperform the open-source model, like Mistral or LLaVa, in terms of diversity. Introducing the SFT and PPO procedure can moderately improve the diversity metrics for the Mistral model. This is in contrast to the findings of (Kirk et al., 2023), which observed the opposite effect, that RLHF reduced diversity in regular text generation tasks. We also found that running DPO can yield a significant increase in the diversity of the model generations as compared to any other method. We hypothesize that this may be due to our finetuning dataset: for each cartoon, we run DPO with a variety of human-generated captions and it therefore learns not to prefer a single caption or type of caption, but a diversity of captions.

Table 4.5: Example caption generations for contest #895 (cartoon in Figure 4.2)

Mistral-7B 0-shot	When your GPS leads you to the wrong galaxy.
Mistral-7B BoN	What do you call it when aliens invade your favorite mall? A takeover by outer space retailers!
Mistral-7B DPO	I just assumed we were the only ones who knew how to pronounce ‘‘H&M’’.
GPT4-o-vision	We travel light-years, and we still need directions!
Claude-3-Opus	Let’s hit the food court first. I’m craving some Jupiter fries.
Human Winner	Do you think death rays would be considered electronics or sporting goods?

Table 4.6: Diversity evaluation on the generated captions. We use the expectation-adjusted distinct N-grams (Average EAD) (Li et al., 2015) and the Sentence-BERT embedding cosine similarity (SBERT) (Reimers and Gurevych, 2019) to measure the per-contest diversity on the token level and semantic level.

Caption Source	Average EAD ↑	SBERT ↑
Human (Top 10)	0.9456	0.7452
Human (#200-#209)	0.9564	0.7496
Human (#1000-#1009)	0.9608	0.7522
Human (Median)	0.9597	0.7489
LLaVA	0.8986	0.5220
LLaVA SFT	0.9002	0.5173
Mistral-7B Instruct 0-Shot	0.9037	0.5349
Mistral-7B Instruct BoN	0.8663	0.4868
Mistral-7B Instruct SFT	0.9043	0.5806
Mistral-7B Instruct RLHF	0.9006	0.5994
Mistral-7B Instruct DPO	0.9206	0.7075
GPT4-o	0.9602	0.5789
Claude-3-Opus	0.9533	0.6813

Proposed Research Question #4: DPO exhibits surprisingly good diversity metrics as compared to PPO and SFT. Does the data diversity used for finetuning explain this, or are other mechanisms at play?

4.6 Future Work and Societal Impact

This paper opens a suite of research problems and challenges going forward and we are excited to continue working on multiple directions of future work.

Improving creativity in LLM generation. While LLMs are largely applauded for their creativity today, our experiments reveal there is still a significant gap between top human-generated content and SOTA LLMs and MLLMs, especially when judged by an expert. We believe addressing the proposed research questions can not only improve funny caption generation, but also improve existing models on the creative generation tasks in general.

Gamified evaluation of AI generated captions by a crowd. As the nature of the funny cartoon captioning task is an engaging game by nature, we plan on building an AI versus Human battle ground rating game. Our envisioned game will allow users to submit their own captions. During rating, participants are presented with two sets of captions from different sources (human vs human, human vs AI and AI vs AI). This also provides us a more reliable system for evaluating new captions on new cartoons. At the same time, researchers are encouraged to submit AI model entries to test out their latest model/alignment methods.

Humor vs offensiveness tradeoff. Optimizing for humor abilities may result in increasing offensiveness and toxicity of model-generated content. We believe an important next step is to study the challenge of balancing humor with potential offensiveness. As the boundary between humorous and offensive is often blurred, the subjective nature of humor and cultural sensitivities needs to be further studied to ensure AI models align with human values.

5 RETRIEVAL-AUGMENTED GENERATION AS NOISY IN-CONTEXT LEARNING: A UNIFIED THEORY AND RISK BOUNDS

In the previous chapter, we investigated adaptation via fine-tuning for creative generation tasks. Now, we turn our attention to another powerful adaptation technique: in-context learning (within-context adaptation), specifically retrieval-augmented generation (RAG). RAG has emerged as an effective method to enhance the capabilities of large language models by retrieving and integrating external knowledge dynamically during inference. Despite its empirical successes, the theoretical foundations of RAG remain relatively unexplored. In this chapter, we address this gap by proposing a finite-sample generalization bound for RAG under an in-context linear regression setting, characterizing precisely the bias-variance trade-off. Our analysis unifies retrieval as a form of query-dependent noisy in-context learning, bridging classical in-context learning and standard RAG as special cases. We further demonstrate these theoretical insights empirically through experiments on established question-answering benchmarks.

5.1 Introduction

Retrieval-Augmented Generation (RAG) enhances language models by appending retrieved texts to the input, enabling access to information beyond pretraining. It is widely used in open-domain QA, fact-checking, and knowledge-intensive tasks (Huang et al., 2023; Lewis et al., 2020a; Ramos et al., 2022; Sarto et al., 2022; Zhao et al., 2024a). Retrieval sources typically fall into two categories: (1) *labeled dataset*, such as training dataset itself (Liu et al., 2021; Izacard et al., 2022; Huang et al., 2024), and (2) *generic corpora without labels*, such as Wikipedia (Chen et al., 2017). Despite its promise, empirical studies show that increasing the number of retrieved passages can degrade performance, especially when irrelevant or redundant texts are included (Levy et al., 2025, 2024). However, the theoretical aspects for understanding of how retrieval affects generalization remain underexplored.

To study its behavior, we frame RAG as noisy in-context learning (ICL). ICL refers to the ability of language models to adapt given the contextual information without updating model weights (Dong et al., 2024). Under this view, retrieved RAG examples can act as noisy context and its quality depends on the retrieval. This view has motivated the development of many work in in-context retrieval (Luo et al., 2024; Shi et al., 2022), where the goal is to retrieve high-quality demonstrate pairs, which reduces the noise of the retrieval.

From a theoretical standpoint, RAG becomes tractable when framed as structured in-context learning, where the context consists of fixed format demonstration pairs. Prior ICL work has analyzed this regime under clean, i.i.d. examples (Ahn et al., 2023; Zhang et al., 2024). These assumptions do not hold in RAG, where retrieved examples are noisy, and their noise level tends to be inversely correlated to their relevance. Currently, no theoretical framework has been developed to study RAG under this structured ICL formulation. Although retrieved examples close to the query should, in principle, improve the predictive accuracy, their quantitative contribution remains unknown because RAG introduces these examples only at the test time (absent during pretraining), thus imposing a distribution shift. In this work, we bridge this gap by modeling RAG as noisy ICL, where retrieved examples follow a structured but perturbed distribution. In particular, we model the retrieval noise both under the uniform (same across examples) and non-uniform (inversely correlated with the retrieval relevance). This view allows us to quantify the impact of retrieval noise and derive generalization bounds that depend on the number of in-context and RAG examples, and the retrieval distance from queries.

Our contributions are summarized as follows:

- We propose a theoretical framework for analyzing RAG and provide the first finite sample bounds for in-context linear regression with RAG. Our bounds show that the improvement from RAG shrinks as you add more retrieved examples, and can even flip to hurt performance, giving concrete guidance on when to stop.

- Our framework includes ICL and standard RAG as limit cases, and also mod-

els retrieved data under different noise regimes, uniform and non-uniform retrieval noise.

- We develop new tools for analyzing the query-dependent RAG data, e.g. a derivation of the expectation for 6th-order Gaussian monomial (Theorem D.3), which can be useful for future research on RAG.
- We conduct experiments for representative models on common QA datasets and demonstrate that early RAG retrieves lie in the uniform noise regime, while later ones shift to non-uniform noise regime, aligning with our theory.

5.2 Related Work

Retrieval Augmented Generation Retrieval-augmented generation (RAG) has emerged as a widely adopted paradigm for enriching LLMs with external knowledge by prepending retrieved passages to the input context (Lewis et al., 2020a; Izacard and Grave, 2020; Borgeaud et al., 2021). From a functional perspective, RAG transforms the model's input distribution by conditioning generation on retrieved textual evidence, often drawn from large-scale corpora via learned or heuristic retrieval mechanisms (Li et al., 2023; Meng et al., 2024; Chen et al., 2024a). While much of the literature focuses on improving retrieval quality, system performance (Asai et al., 2023; Li et al., 2024; Xu et al., 2024), and answer reliability (Xiang et al., 2024; Xu et al., 2024), the theoretical foundations of RAG remain underexplored.

In-context Learning (ICL) ICL obtains its popularity from the original GPT-3 paper (Brown et al., 2020), and becomes widely used in LLM applications (Dong et al., 2024; Min et al., 2021). The recent advance in ICL theory (Ahn et al., 2023; Zhang et al., 2024; Xie et al., 2021) provides a rigorous and versatile framework to study transformers and LLMs. People have used this ICL framework to study novel settings, like out-of-distributions tasks (Wang et al., 2024b) and test-time training (Gozeten et al., 2025). People also have also studied the noisy in-context

learning from robustness (Cheng et al., 2025) and calibration perspectives (Zhao et al., 2024b), which are different from our setup.

In-context Retrieval In-context retrieval (Luo et al., 2024) refers to retrieving a set of query-dependent demonstrations than using fixed set of demonstrations. The label of the demonstration pairs can come from various sources, such as in-domain training set (Izacard et al., 2022; Huang et al., 2024; Ye et al., 2023), cross-domain data (Cheng et al., 2023; Shi et al., 2022), automatic LLM generation (Zhang et al., 2022; Li and Qiu, 2023), pseudo-labels from unstructured data (Lyu et al., 2022; Li et al., 2022). In our theoretical analysis and experiments, we focus on the simplest in-context retrieval, in-domain retrieval from the training set, as in (Izacard et al., 2022; Huang et al., 2024). Note that in-context retrieval is a term developed later and some earlier papers discuss ICL with retrieval as retrieving relevant documents without labels (Ram et al., 2023).

5.3 Problem Setup

Our problem setup is similar to (Zhang et al., 2024; Garg et al., 2022), with RAG examples to form the additional in-context examples. It is worth noting that many works focus on ICL at test (inference) time, specifically without parameter updates (Dong et al., 2022). Our work adopts the framework of *ICL with warmup*, also known as, *supervised in-context training*. Specifically, we assume that the pretraining data is also formed by in-context examples. Then, during the test time, we formed prompts with in-context examples with additional RAG examples.

Notations We denote $[n] = \{1, \ldots, n\}$ for an integer $n \geqslant 1$. We denote the trace product of two matrices $A, B \in \mathbb{R}^{m \times n}$ as $\mathrm{tr}(AB^\top)$.

Pretraining Data We consider learning over linear regression data. The training data is a set of prompts. Each prompt is of size m: $(x_1, y_1, \ldots, x_m, y_m, x_q) \in \mathbb{R}^{d(m+1)+m}$ where $(x_1, y_1), \ldots, (x_m, y_m)$ form the m demonstration pairs. The goal

is to predict $\hat{y}_q$ for the query example x_q to match the true label y_q. The prompt is embedded in the following form:

$$P_m^{pt} := \begin{pmatrix} x_1 & x_2 & \dots & x_m & x_q \\ y_1 & y_2 & \dots & y_m & 0 \end{pmatrix} \in \mathbb{R}^{(d+1)\times(m+1)} \tag{5.1}$$

where $(x_1, y_1), \dots, (x_m, y_m), (x_q, y_q)$ *i.i.d.* $\mathcal{D}_{pt}$ (pt denoting Pretraining). The output follows the linear model:

$$y_i = x_i^\top \beta_{pt} + \epsilon_i, \quad \epsilon_i \textit{ i.i.d. } \mathcal{N}(0, \sigma^2) \quad \text{under} \quad \mathcal{D}_{pt} \tag{5.2}$$

where $i \in [m] \cup \{q\}$, β_{tt} is the weight vector in pretraining, and ϵ_i is the noise for example i.

Inference Data (with RAG) During inference/test time, the test prompt $P_{m,n}^{tt+rag}$ (tt denoting test-time) is formed by m in-context pairs $(x_1, y_1), \dots, (x_m, y_m)$, n retrieval-augmented pairs $(x_1^{rag}, y_1^{rag}), \dots, (x_n^{rag}, y^{rag})$, and the query pair x_q, y_q. The test prompt is embedded in the following form:

$$P_{m,n}^{tt+rag} := \begin{pmatrix} x_1 & \dots & x_m & x_1^{rag} & \dots & x_n^{rag} & x_q \\ y_1 & \dots & y_m & y_1^{rag} & \dots & y_n^{rag} & 0 \end{pmatrix} \in \mathbb{R}^{(d+1)\times(m+n+1)} \tag{5.3}$$

The input x in each in-context or query pair follows the test-time distribution $\mathcal{D}_{tt}$, and the label is:

$$y_i = x_i^\top \beta_{tt} + \epsilon_i, \quad \epsilon_i \textit{ i.i.d. } \mathcal{N}(0, \sigma^2) \quad \text{under} \quad \mathcal{D}_{tt} \tag{5.4}$$

where $i \in [m] \cup \{q\}$, ϵ_i is the noise of example i, and β_{tt} is the weight vector during test time. The input x in each RAG pair follows the corresponding RAG distribution $\mathcal{D}_{rag}(x_q)$: assume the RAG query $x_i^{rag} = x_q + r_i$ is generated around the query example x_q, where r_i is the offset. The label in the RAG example is given by:

$$y_i^{rag} = (x_i^{rag})^\top \beta_{tt} + \epsilon_i^{rag}, \quad \epsilon_i^{rag} \textit{ i.i.d. } \mathcal{N}(0, \sigma_{rag,i}^2) \quad \text{under} \quad \mathcal{D}_{rag}(x_q) \tag{5.5}$$

where $i \in [n]$, ϵ_i^{rag} is the noise of the i-th RAG example.

For the compactness of writing, we define the following matrices and vectors:

$$\boldsymbol{X}_{\text{icl}} := [\boldsymbol{x}_1^\top; \ldots; \boldsymbol{x}_m^\top],\ \boldsymbol{X}_{\text{rag}} := [(\boldsymbol{x}_1^{\text{rag}})^\top; \ldots; (\boldsymbol{x}_n^{\text{rag}})^\top],\ \boldsymbol{y}_{\text{icl}} := [y_1; \ldots; y_m],\ \boldsymbol{y}_{\text{rag}} := [y_1^{\text{rag}}; \ldots; y_n^{\text{rag}}],$$

$$\boldsymbol{\epsilon}_{\text{icl}} := [\epsilon_1; \ldots; \epsilon_m],\ \boldsymbol{\epsilon}_{\text{rag}} := [\epsilon_1^{\text{rag}}; \ldots; \epsilon_n^{\text{rag}}],\ \boldsymbol{r} = [\boldsymbol{r}_i^\top; \ldots; \boldsymbol{r}_n^\top]$$

$$\boldsymbol{X} = \begin{bmatrix} \boldsymbol{X}_{\text{icl}} \\ \boldsymbol{X}_{\text{rag}} \end{bmatrix} \in \mathbb{R}^{(m+n)\times d},\ \boldsymbol{X}_{\text{rag}} = \begin{bmatrix} \boldsymbol{x}_q + \boldsymbol{r}_1 \\ \vdots \\ \boldsymbol{x}_q + \boldsymbol{r}_n \end{bmatrix} \in \mathbb{R}^{n\times d},\ \boldsymbol{y} = \begin{bmatrix} \boldsymbol{y}_{\text{icl}} \\ \boldsymbol{y}_{\text{rag}} \end{bmatrix} \in \mathbb{R}^{m+n},\ \boldsymbol{\epsilon} = \begin{bmatrix} \boldsymbol{\epsilon}_{\text{icl}} \\ \boldsymbol{\epsilon}_{\text{rag}} \end{bmatrix} \in \mathbb{R}^{m+n}$$

Training and Testing We let $\boldsymbol{W}$ be the model parameters, and F be the model. Given an input prompt $\boldsymbol{P}_m^{\text{pt}}$ with demonstration pairs, the model predicts $\hat{y}_q := F(\boldsymbol{P}_m^{\text{pt}}; \boldsymbol{W})$. As a common practice in theoretical studies of LLM for feasible analysis, we use the MSE loss as the evaluation metrics (Zhang et al., 2024; Ahn et al., 2023; Xie et al., 2021). Then, the population loss on the pretraining data is:

$$\mathcal{L}_{\text{pt}}(\boldsymbol{W}) := \mathop{\mathbb{E}}_{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_m, y_m)(\boldsymbol{x}_q, y_q) \sim \mathcal{D}_{\text{pt}}} \left[\left(y_q - F\left(\boldsymbol{P}_m^{\text{pt}}; \boldsymbol{W} \right) \right)^2 \right] \tag{5.6}$$

Its minimizer is denoted as:

$$\bar{\boldsymbol{W}}^* := \min_{\boldsymbol{W}} \mathcal{L}_{\text{pt}}(\boldsymbol{W}). \tag{5.7}$$

To apply the pretrained $\boldsymbol{W}^*$ from the pretraining context size of m to the test-time context size of $m + n$, we will need to scale it properly (see Theorem D.1) and use

$$\boldsymbol{W}^* = \frac{m}{m+n} \bar{\boldsymbol{W}}^*. \tag{5.8}$$

During the test time we evaluate the population loss over the test prompt with RAG examples $\boldsymbol{P}_{m,n}^{\text{tt+rag}}$:

$$\mathcal{L}_{\text{tt+rag}}(\boldsymbol{W}) := \mathop{\mathbb{E}}_{\substack{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_m, y_m)(\boldsymbol{x}_q, y_q) \sim \mathcal{D}_{\text{tt}} \\ (\boldsymbol{x}_1^{\text{rag}}, y_1^{\text{rag}}), \ldots, (\boldsymbol{x}_n^{\text{rag}}, y_n^{\text{rag}}) \sim \mathcal{D}_{\text{rag}}(\boldsymbol{x}_q)}} \left[\left(y_q - F\left(\boldsymbol{P}_{m,n}^{\text{tt+rag}}; \boldsymbol{W} \right) \right)^2 \right] \tag{5.9}$$

Model Architecture We study the single-layer linear self-attention model (LSA) as the framework for theoretical analysis, similar to many existing studies (e.g., (Ahn et al., 2023; Zhang et al., 2024)). The prediction of the model F on a prompt $\mathbf{P}$ with query x_q is:

$$\hat{y}_q := F(\mathbf{P}) = [\mathbf{P}W_Q W_K^\top \mathbf{P}^\top \mathbf{P}W_V]_{m+n+1,d+1} = x_q^\top W X^\top y \tag{5.10}$$

where the query, key, and value matrices $W_Q, W, W_V \in \mathbb{R}^{(d+1)\times(d+1)}$ are parameterized by W in the follow way:

$$W_Q W_K^\top = \begin{bmatrix} W & \mathbf{0}_{d\times 1} \\ \mathbf{0}_{1\times d} & 0 \end{bmatrix}, \quad W_V = \begin{bmatrix} \mathbf{0}_{d\times d} & \mathbf{0}_{d\times 1} \\ \mathbf{0}_{1\times d} & 1 \end{bmatrix}$$

We note that this parameterization is commonly used in the previous works (Ahn et al., 2023; Zhang et al., 2024), and is shown to capture the key properties of in-context learning. Furthermore, (Ahn et al., 2023) shows that the formulation is the optimum converged from pretraining on Gaussian data.

5.4 Theoretical Analysis: Generalization Bound for RAG

To study test-time error and sample complexity in in-context linear regression with RAG examples, we consider two noise regimes: **uniform retrieval noise** and **non-uniform retrieval noise**. Uniform retrieval noise assumes the RAG noise ϵ_i^{rag} for each example i is i.i.d. Since its variance is distance-agnostic, it can model a scenario of retrieval where the noise is prevailing across data points. Non-uniform retrieval noise assumes either the variance or the label-corruption probability grows with the variance of retrieval vector — e.g. $\sigma^2_{rag,i}$ increases with δ_i^2 or probability of making mistakes increases with δ_i^2. This captures retrieval from datasets where near neighbors often supply the right signal while far ones are potentially noisy or misleading. Because the noise spectrum is now heavy-tailed, adding more RAG

examples past a threshold could yield diminishing benefits for RAG examples and even become counter-productive. Framing RAG through these two lenses allows precise clarification about when extra retrieved examples will pay off, and when they will hit the intrinsic ceiling and more retrieved examples don't help anymore. These are well corroborated by our experimental results on real data (see Section 5.5).

First, we introduce the key data assumptions.

Assumption 1 (Gaussian Retrieval Offset)**.** *We assume the retrieval offset* $\mathbf{r}_i$, $\forall i \in [n]$ *to follow a Gaussian distribution:* $\mathbf{r}_i \overset{i.i.d.}{\sim} \mathcal{N}\left(0, \delta_i^2 I_d\right)$.

The key property that we want to control for RAG examples is its distance from the query points $\mathbf{x}_q$. However, modeling the queried example directly through the retrieval distance leads to complicated theoretical analysis. Here, we note that the retrieval distance $\|\mathbf{r}_i\|_2$ converges to a distribution concentrated in an $\mathcal{O}(\delta_i\sqrt{d})$ ball around the query with respect to d (Cover and Hart, 1967). Thus, controlling the variance of the retrieval offset can alternatively control the retrieval distance. And we make the following additional data assumptions.

Assumption 2 (Data Assumption)**.** *We assume the data follows the following:*

1. *Pretraining Examples* ($\mathcal{D}_{pt}$). *For a pretraining prompt of length* $m+1$ *and for all* $i \in [m] \cup \{q\}$, *we assume* $\mathbf{x}_i \text{i.i.d.} \mathcal{N}(0, I)$, $\epsilon_i \text{i.i.d.} \mathcal{N}(0, \sigma^2)$, $\beta_{pt} \sim \mathcal{N}(0, I)$.
2. *Test Time Examples* ($\mathcal{D}_{tt}$). *For a test-time prompt of length* $m+n+1$ *and for all* $i \in [m] \cup \{q\}$, *we assume* $\mathbf{x}_i \text{i.i.d.} \mathcal{N}(0, I)$, $\epsilon_i \text{i.i.d.} \mathcal{N}(0, \sigma^2)$, $\beta_{tt} \sim \mathcal{N}(0, I)$.
3. *Test-Time RAG Examples* ($\mathcal{D}_{rag}(\mathbf{x}_q)$). *For a test-time prompt of length* $m+n+1$ *and for all* $i \in [m+1, \ldots, m+n]$, *we assume* $\mathbf{x}_i^{rag} \text{i.i.d.} \mathcal{N}(0, I)$, $\epsilon_i^{rag} \sim \mathcal{N}(0, \sigma_{rag,i}^2)$, *and the same* β_{tt} *as* (2).

Here, we assume the isotropic Gaussian property for the input, noise and the weight vector, a common assumption made in ICL theory (Ahn et al., 2023; Gozeten et al., 2025) for simple yet meaningful analysis.

Overview of the Key Results

tmargin=0em,noitemsep (*Uniform Noise*) RAG examples are as effective as ICL examples in reducing the variance-induced err but ineffective at reducing the bias-induced err, causing a loss plateau for $n \to \infty$,

tmargin=0em,noitemsep (*Non-Uniform Noise*) RAG could improve the variance-induced error up to a finite n at a cost of increasing bias-induced error.

Roadmap Under these assumptions and uniform retrieval noise, we will first derive the population loss of RAG, $\mathcal{L}_{\text{tt+rag}}(\boldsymbol{W})$, for general $\boldsymbol{W}$ as in Theorem 5.1, analyze its finite sample complexity under the optimal pretrained weight $\boldsymbol{W}^*$ as in Proposition 1 and derive an optimal number of RAG examples of n^* for a given number of ICL examples m as in Proposition 2. These discussions leads to our first key result. Then, under the non-uniform retrieval noise, we will prove the sample complexity under the distance-proportional noise (Theorem 5.2) and distance-weighted mixture noise (Theorem 5.3), and obtain our second key results above.

5.4.1 Uniform Retrieval Noise

Assumption 3 (Uniform Retrieval Noise)**.** *The RAG noise* $\boldsymbol{\epsilon}_{\text{rag}}$ *shares the same Gaussian distribution with variance* σ^2_{rag}*, i.e.* $\forall i \in [m+1, \ldots, m+n]$, $\sigma^2_{\text{rag},i} = \sigma^2_{\text{rag}}$.

First, we present the assumption for uniform retrieval noise. In other words, all RAG examples are as helpful, and its improvement on the actual prediction is determined by the retrieval distance.

Theorem 5.1 (Population Loss for ICL with RAG Examples)**.** *Under Assumption 1, 2, 3, the population loss of the linear self-attention predictor* $\hat{y}_q = \boldsymbol{x}_q^\top \boldsymbol{W} \boldsymbol{X}^\top \boldsymbol{y}$ *satisfies*

$$\mathcal{L}_{\text{tt+rag}}(\boldsymbol{W}) = \underbrace{\mathbb{E}\left(\mathbb{E}\left(\hat{y}_q\right) - \hat{y}_q\right)^2}_{:=\text{err}_{\text{variance}}(\boldsymbol{W})} + \underbrace{\mathbb{E}\left(\mathbb{E}\left(\hat{y}_q\right) - \mathbb{E}\left(y_q\right)\right)^2}_{:=\text{err}_{\text{bias}}(\boldsymbol{W})} + \underbrace{\sigma^2}_{\textit{irreducible noise}} \quad \textit{, and specifically,} \tag{5.11}$$

$$
\begin{aligned}
\underset{\text{variance}}{\text{err}}(\boldsymbol{W}) &= \left[m\sigma^2 + \left(1+\delta^2\right) n\sigma_{\text{rag}}^2\right] \operatorname{tr}(\boldsymbol{W}^\top \boldsymbol{W}) + n\sigma_{\text{rag}}^2 \operatorname{tr}(\boldsymbol{W}^2) + n\sigma_{\text{rag}}^2 \operatorname{tr}(\boldsymbol{W})^2 \\
\underset{\text{bias}}{\text{err}}(\boldsymbol{W}) &= \beta_{\text{tt}}^\top \left[I - (n\delta^2 + 2n + m)(\boldsymbol{W} + \boldsymbol{W}^\top) - 2n \operatorname{tr}(\boldsymbol{W}) I + M_4\right] \beta_{\text{tt}} \\
&= \beta_{\text{tt}}^\top \left[I - (n\delta^2 + 2n + m)(\boldsymbol{W} + \boldsymbol{W}^\top) - 2n \operatorname{tr}(\boldsymbol{W}) I \right. \\
&\quad + \left[n^2\left(2+\delta^2\right) + n\left(m+\delta^2\right)\right]\left(\boldsymbol{W}^2 + \left(\boldsymbol{W}^2\right)^\top\right) + 2n(n+\delta^2)\boldsymbol{W}\boldsymbol{W}^\top \\
&\quad + \left[m^2 + m + mn\left(2+2\delta^2\right) + n^2\left(2+2\delta^2+\delta^4\right) + n\left(2\delta^2+\delta^4\right)\right]\boldsymbol{W}^\top \boldsymbol{W} \\
&\quad + \left[n^2\left(2+\delta^2\right) + n\left(m+\delta^2\right)\right]\left(\operatorname{tr}(\boldsymbol{W})\left(\boldsymbol{W} + \boldsymbol{W}^\top\right)\right) \\
&\quad \left. + \left[n^2 + n\delta^2\right]\left(\operatorname{tr}(\boldsymbol{W})^2 + \operatorname{tr}\left(\boldsymbol{W}^2\right)\right) I + \left[m + n^2 + n\left(2\delta^2+\delta^4\right)\right] \operatorname{tr}\left(\boldsymbol{W}^\top \boldsymbol{W}\right) I\right] \beta_{\text{tt}}
\end{aligned}
$$

Here, we derive the exact bias-variance decomposition for ICL with RAG. The first line is the variance-induced error formed by a weighted sum of noise from ICL examples and RAG examples. Because of the implicit scaling of $\boldsymbol{W}$ as discussed in Theorem D.1, the second order term in $\boldsymbol{W}$ will introduce an additional weight scaling of $\frac{m^2}{(m+n)^2}$ when adapting from the weight learned on m size context to $m+n$ size context. Thus, larger n will let $\text{err}_{\text{variance}}(\boldsymbol{W}) \to 0$, and the convergence rate is affected by δ^2. Larger retrieval distance leads to a slower convergence. The bias-induced error is composed of all possible monomials of $\boldsymbol{W}$ up to the 2nd-order with tr operation. The complex dependency on m, n, δ^2, d requires additional assumptions on $\boldsymbol{W}$ to further interpret. As a sanity check, when $n = 0$ (ICL-only), this decomposition can exactly recover loss as in Lemma B.2 in (Gozeten et al., 2025).

As a proof sketch, we first compute $\text{err}_{\text{variance}}(\boldsymbol{W}) = \mathbb{E}(x_q^\top \boldsymbol{W} \boldsymbol{X}^\top \boldsymbol{\epsilon})^2$ by splitting the calculation for ICL and RAG examples based on $\boldsymbol{X}$. Then, we compute $\text{err}_{\text{bias}}(\boldsymbol{W}) = \mathbb{E}[(x_q^\top (I - \boldsymbol{W}\boldsymbol{X}^\top \boldsymbol{X})\beta_{\text{tt}})^2]$. The main technical challenge lies in the dependency of $\boldsymbol{X}_{\text{rag}}$ on x_q, and err_{bias} has a 6th-order dependency on x_q (2 from x_q and 4 from $\boldsymbol{X}$). As shown in Theorem D.3, $\mathbb{E}\left[x_q x_q^\top A x_q x_q^\top B x_q x_q^\top\right]$ gives 15 new terms that include all the second order monomials of $\boldsymbol{W}$ with tr. The calculation requires multiple careful applications of Isserlis' theorem (Isserlis, 1918), and the full proof can be seen in Section D.2. It is possible to prove this theorem for a design matrix with non-isotropic covariance, but computing the expectation of the

6th-order Gaussian monomial is more complicated.

Here, we present the finite sample bound for pretrained W^* for better interpretation.

Proposition 1 (Finite Sample Generalization Bound)**.** *Under Assumption 1, 2, 3, if* $\delta^2 \ll 1$,

$$\mathcal{L}_{\text{tt+rag}}(W^*) = \mathcal{O}\left(\sigma^2 + \underbrace{\frac{dm}{(m+n)^2}\sigma^2 + \frac{d^2 n}{(m+n)^2}\sigma_{\text{rag}}^2}_{\text{err}_{variance}(W^*)} + \underbrace{\|\beta_{\text{tt}}\|_2^2\left[\frac{d}{m} + d^2\left(\frac{n}{m+n}\right)^2\right]}_{\text{err}_{bias}(W^*)}\right)$$

$$\underset{\text{variance}}{\text{err}}(W^*) = \begin{cases} \mathcal{O}(\frac{d}{m}\sigma^2 + \frac{d^2}{m^2}\sigma_{\text{rag}}^2) = \mathcal{O}\left(\frac{1}{m}\right) & m \to \infty, n \textit{ fixed.} \\ \mathcal{O}(\frac{d}{n^2}\sigma^2 + \frac{d^2}{n}\sigma_{\text{rag}}^2) = \mathcal{O}\left(\frac{1}{n}\right) & n \to \infty, m \textit{ fixed} \\ \mathcal{O}(\frac{d}{m}\sigma^2 + \frac{d^2}{m}\sigma_{\text{rag}}^2) = \mathcal{O}\left(\frac{1}{m}\right) & m, n \to \infty, n = \Theta(m) \end{cases} \tag{5.12}$$

$$\underset{\text{bias}}{\text{err}}(W^*) = \begin{cases} \mathcal{O}\left(\|\beta_{\text{tt}}\|_2^2 \frac{d}{m}\right) & \textit{if } m \to \infty, n \textit{ is fixed} \\ \mathcal{O}\left(\|\beta_{\text{tt}}\|_2^2 d^2\right) = C_1 & \textit{if } n \to \infty, m \textit{ is fixed} \\ \mathcal{O}\left(\|\beta_{\text{tt}}\|_2^2\left(\frac{d}{m} + d^2\right)\right) = C_2 + \mathcal{O}(\|\beta_{\text{tt}}\|_2^2 \frac{d}{m}) & \textit{if } m \to \infty, n = \Theta(m) \end{cases} \tag{5.13}$$

Here, we assume $\delta^2 \ll 1$ as the test time example x_i has only a variance of I, and it is unrealistic to assume a higher retrieval variance than the input variance. On the limit case where $m \to \infty$ and n are fixed, we observe that both variance-induced and bias-induced error decay at a rate of $\mathcal{O}(1/m)$, matching the results from the existing paper (Ahn et al., 2023; Zhang et al., 2024). When $n \to \infty$, the variance-induced error decays as $\mathcal{O}(1/n)$ matching the $\mathcal{O}(1/m)$ rate. However, introducing the RAG is ineffective at reducing the bias-induced error. Even when $m \to \infty$, increasing n will cause a loss plateau.

This effect can be explained by the underlying adaptive ability of transformers. In an online learning setup, we could always use the mean of the queried data as

the prediction. However, in the LSA setup, the pretrained W^* serves as a proxy for $\mathbb{E}^{-1}(X^\top X)$. In order to retain the adaptivity to the entire distribution of β_{tt}, we cannot use the optimal linear classifier $(X^\top X)^{-1}X^\top y$ or use the mean of the retrieved examples ad hoc. At the test stage, X_{rag} only appears in $X^\top y$ and not in W^*. The difference between $\mathcal{D}_{rag}(x_q)$ and $\mathcal{D}_{tt}$ directly leads to the increase of variance worsened by the increase of n. See full proof in Section D.2. Now, a natural question is whether we can find a balance of variance and bias and obtain an optimal RAG example size n^*.

Proposition 2. *Under Assumption 1,2,3, $\delta^2 \ll 1$, and reasonable choice of σ^2, σ^2_{rag} ($\sigma^2, \sigma^2_{rag} \ll \|\beta_{tt}\|_2^2$), the optimal n^* that minimizes the RAG loss follows:*

$$n^* = \mathcal{O}\left(\frac{m\left(d^2\|\beta_{tt}\|_2^2 + d\sigma^2 - d^2\sigma^2_{rag}\right)}{md^2\|\beta_{tt}\|_2^2 - d^2\sigma^2_{rag}}\right) = \mathcal{O}\left(\frac{d\|\beta_{tt}\|_2^2 + \sigma^2 - d\sigma^2_{rag}}{d\|\beta_{tt}\|_2^2}\right) \tag{5.14}$$

and the improvement on loss from picking the optimal n^* over $n = 0$ is given as:

$$\mathcal{L}_{tt+rag}(W^*)|_{n=0} - \mathcal{L}_{tt+rag}(W^*)|_{n=n^*} = \mathcal{O}\left(\frac{1}{m^2}\right) \tag{5.15}$$

In fact, the optimal n^* does not scale with m omitting the lower-order terms. Note that for $\|\beta_{tt}\|_2^2 = \mathcal{O}(1)$, $\|\beta_{tt}\|_2^2$ will dominate the numerator for reasonable choices of σ^2 and σ^2_{rag}. A larger ICL noise σ^2 leads to a larger n^*, i.e. requiring more RAG examples to compensate for the loss. A larger RAG noise σ^2_{rag} leads to a smaller n^*, i.e. less efficiency on RAG examples. And the improvement converges at $\mathcal{O}(\frac{1}{m^2})$, diminishing for large m. See the full proof in Section D.2. Several empirical works also observe a performance drop when increasing the number of retrieved examples (Wang et al., 2024a; Levy et al., 2025).

5.4.2 Non-Uniform Retrieval Noise

The uniform-noise setup in Section 5.4.1 relies on a clean retrieval pool, so we could keep the variance σ^2_{rag} fixed. In open-domain retrieval, this assumption could

collapse: many retrieved examples could contain no answer or even a wrong answer. Empirically, people have observed that passages that are closer to the query vector x_q are more likely (Yang and Seo, 2020; Yoran et al., 2023; Lewis et al., 2020b) to contain the correct label. We want to theoretically investigate if the following hypothesis still holds:

Closer to query $x_q \implies$ *more likely* to contain *correct* answer.

Distance-Proportional Noise (DPN)

We first investigate the scenario where the retrieval noise is proportional to the retrieval distance. Since the ICL analysis only applies to the mean-squared error loss, we study the effect of RAG under DPN on the correctness of the predictions.

Assumption 4 (Distance-Proportional Noise)**.** *There exists a constant $\gamma_1 > 0$ such that, for every retrieved sample i, $\sigma^2_{\text{rag},i} = \gamma_1 \sigma^2 \delta_i^2$, i.e. the RAG noise variance grows linearly with the variance δ_i^2 that governs the retrieval distance.*

Under the new data assumption, we denote the corresponding RAG loss, bias-induced error, and variance-induced error for W to be $\hat{\mathcal{L}}_{\text{tt+rag}}(W)$, $\hat{\text{err}}_{\text{bias}}(W)$, and $\hat{\text{err}}_{\text{variance}}(W)$.

Theorem 5.2 (Finite Sample RAG Generalization Bound under DPN)**.** *Under Assumption 1, 2, 4, the population loss is given as:*

$$\hat{\text{err}}_{\text{variance}}(W) = m\sigma^2 \operatorname{tr}(W^\top W) + \sum_{i=1}^{n} \gamma_1 \delta_i^2 [(1+\delta_i^2)\operatorname{tr}(W^\top W) + \operatorname{tr}(W^2) + \operatorname{tr}(W)^2]$$

If the variance of the retrieval distance follows power law, i.e. $\exists \gamma_2 > 0, q \geqslant 0$ s.t. $\delta_i^2 = \gamma_2 i^q$, then

$$\hat{\text{err}}_{\text{bias}}(W^*) = \mathcal{O}\left(\underset{\textit{bias}}{\text{err}}(W^*) + \|\beta_{\text{tt}}\|_2^2 \left[\frac{d n^{2q+1} + n^{2q+2}}{(m+n)^2}\right]\right) \tag{5.16}$$

and

$$\hat{\text{err}}_{\text{variance}}(\boldsymbol{W}^*) = \mathcal{O}\left(\frac{dm\sigma^2 + d(n^{2q+1})\sigma^2}{(m+n)^2}\right) = \begin{cases} \mathcal{O}\left(dn^{2q-1}\sigma^2\right) & \text{if } n \to \infty,\ q \leqslant 1/2 \\ \text{diverges} & \text{if } n \to \infty,\ q > 1/2 \end{cases} \tag{5.17}$$

Here, we derive the sample complexity under DPN. A second order dependency on δ_i^2 shows up in both the variance-induced and bias-induced error (exact form seen in Section D.2). Thus, the δ_i^2-involved constant will dominate the other constants. Specifically, it even leads to divergence for $q > 1/2$ for the variance-induced error and $q > 0$ for the bias-induced error.

Distance-Weighted Mixture Noise

In this section, we discuss the scenario where further RAG examples are less likely to contain the correct answers. We use a pair of large and small noises to model the correct/incorrect examples.

Assumption 5 (Distance-Weighted Mixture Noise)**.** *We assume that the RAG noise is formed by a mixture of small and large noise:*

$$y(\boldsymbol{x}_{\text{rag}}) = \begin{cases} f(\boldsymbol{x}_{\text{rag},i}) + \epsilon_s & \textit{w.p. } p_i \\ f(\boldsymbol{x}_{\text{rag},i}) + \epsilon_l & \textit{w.p. } 1 - p_i \end{cases}$$

where $\epsilon_s \sim \mathcal{N}(0, c_s\sigma^2)$ *corresponds to the small noise and* $\epsilon_l \sim \mathcal{N}(0, c_l\sigma^2)$ *corresponds to the large noise, with* $c_l \geqslant c_s \geqslant 0$*. The probability of sampling small noise* p_i *follows an inverse power law of the variance of the retrieval distance, i.e.* $p_i = (1 + \delta_i^2)^{-\tilde{q}}$*,* $\tilde{q} \geqslant 0$*.*

Here, we choose the sampling probability (of small noise) p_i to follow a polynomial decay and the constant 1 here is to ensure $p_i = 0$ when $\delta_i^2 = 0$. Under the new data assumption, we denote the corresponding RAG loss, bias-induced error, and variance-induced error for $\boldsymbol{W}$ to be $\tilde{\mathcal{L}}_{\text{tt+rag}}(\boldsymbol{W})$, $\tilde{\text{err}}_{\text{bias}}(\boldsymbol{W})$, and $\tilde{\text{err}}_{\text{variance}}(\boldsymbol{W})$.

Theorem 5.3 (Finite Sample RAG Bound under Distance-Weighted Mixture Noise)**.** *Under Assumption 1, 2, 5, then* $\tilde{\text{err}}_{bias}(\boldsymbol{W}) = \hat{\text{err}}_{bias}(\boldsymbol{W})$*, and*

$$\hat{\text{err}}_{\text{variance}}(\boldsymbol{W}) = m\sigma^2 \operatorname{tr}(\boldsymbol{W}^\top \boldsymbol{W}) + \sum_{i=1}^{n} \left(p_i \sigma_s^2 + (1-p_i)\sigma_l^2\right) \left[(1+\delta_i^2) \operatorname{tr}(\boldsymbol{W}^\top \boldsymbol{W}) + \operatorname{tr}(\boldsymbol{W}^2) + \operatorname{tr}(\boldsymbol{W})^2\right]$$

If the variance of the retrieval distance follows power law, i.e. $\exists \gamma_2 > 0, q \geqslant 0$ *s.t.* $\delta_i^2 = \gamma_2 i^q$*, then:*

$$\tilde{\text{err}}_{\text{variance}}(\boldsymbol{W}^*) = \begin{cases} \mathcal{O}\left(c_l d n^{q-1} \sigma^2 - (c_l - c_s)\sigma^2 d n^{q-1-q\tilde{q}}\right) & \text{if } n \to \infty,\ q \leqslant 1 \\ \text{diverges} & \text{if } n \to \infty,\ q > 1 \end{cases} \tag{5.18}$$

The bias-induced error here is the same as in DPN, since we assume a polynomial dependency for δ_i^2 on i in both setting and the bias-induced error is independent of the variance of noise. Even though the variance of small/large noise are bounded, the dependency on the retrieval distance leads to the divergence at large q ($q > 1$). The large prediction noise will dominate the variance-induced error, but a larger gap between large and small noise ($c_l - c_s$) can mitigate the error by a ratio of $\mathcal{O}(n^{-q\tilde{q}})$. That is, the smaller q and $\tilde{q}$ are, the lower the error.

We note that the uniform noise scenario can also admit the mixture noise model by taking a constant p_i, $\forall i$, resulting in a form similar to the standard uniform retrieval noise in Proposition 1.

5.5 Experiments

We investigate the effect of RAG focusing on the following questions: (**Q1**) Whether RAG data outperform randomly sampled in-context examples? (**Q2**) What are the impacts of the RAG examples from training data and RAG passages from external corpora? (**Q3**) With a fixed budget of example numbers, what is the effect of varying the ratio between the two types of RAG data? Our experiments provide

the following findings: (**A1**) RAG data lead to better performance than in-context ones under different data budgets. (**A2**) Interestingly, the first few RAG training examples significantly improve performance, but later ones are harmful, because the first few are highly relevant but later ones are noise rather than signal. In contrast, RAG passages from external corpora can slowly but monotonically improve the performance, because external corpora are large enough to provide noisy but still relevant data. These are captured by different noise models in our theory. **(A3)** The performance is not monotonic with the ratio, and the sweet spot depends on the data/model.

Setup For Natural Questions (NQ), the retrieval index is constructed from the December 2018 Wikipedia dump. For TriviaQA, we use the December 2021 version. To accommodate hardware limitations, we randomly subsample 10% of the full index for both datasets. This reduces retrieval cost and memory usage, allowing all experiments to be conducted on a single NVIDIA A100 or L40 GPU.

We use representative models **ATLAS** Izacard et al. (2022) and **RAVEN** Huang et al. (2024) on two standard open-domain question answering benchmarks **Natural Questions (NQ)** Kwiatkowski et al. (2019) and **TriviaQA** Joshi et al. (2017). For evaluation, the context consists of m in-context examples, and n RAG data points (including n_1 RAG examples from the training data and n_2 RAG passages from external corpora like Wikipedia, so $n = n_1 + n_2$). We choose different m, n_1, n_2's for our study purpose and report the standard exact match (EM) accuracy on 1000 random samples from the test set.

RAG v.s. In-Context For a budget c, we compare using RAG only ($m = 0, n_1 = n_2 = c/2$) and in-context examples only ($m = c, n = 0$). The results in Figure 5.1 show that RAG consistently outperforms in-context examples, as RAG provides query-relevant data with more signals to address the query, consistent with our analysis.

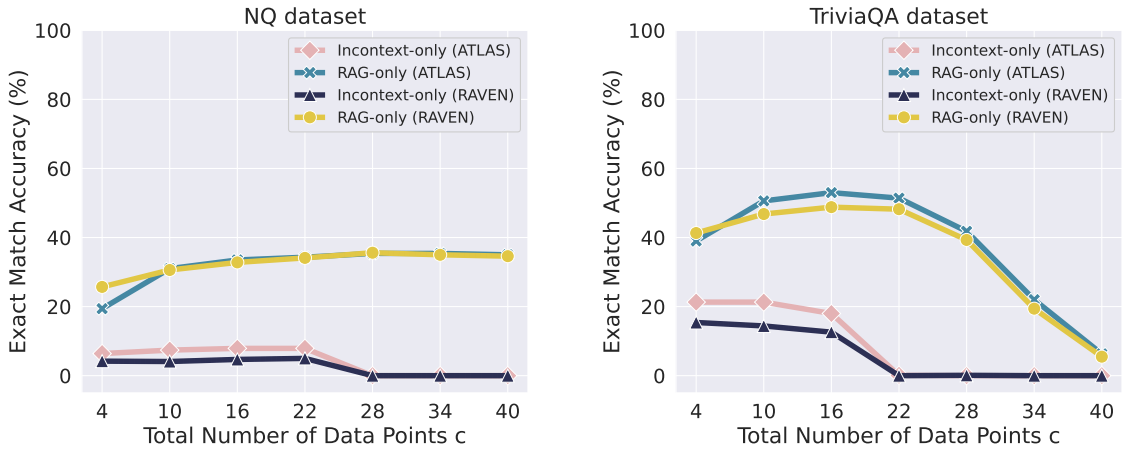


Figure 5.1: We compare performance between the RAG-only ($c = m$) versus in-context-only methods ($c = n_1 + n_2, n_1 = n_2$), where c is the total number of data, n_1 refers to retrieved examples and n_2 to passages.

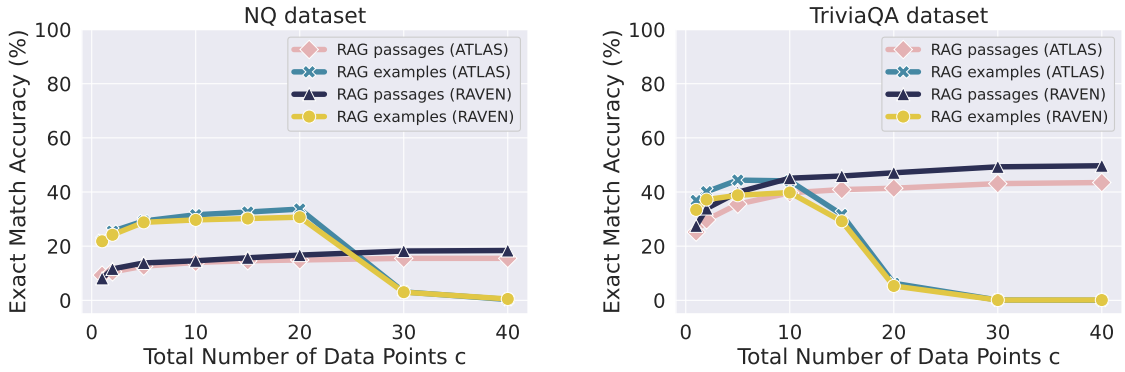
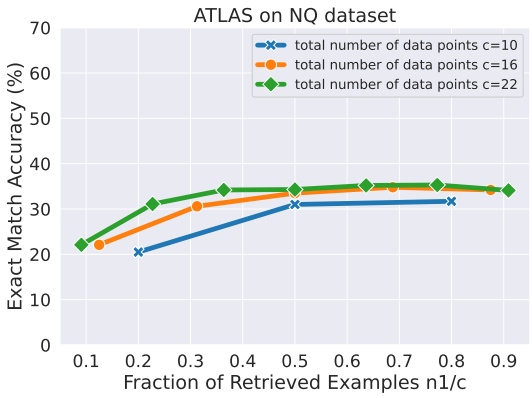
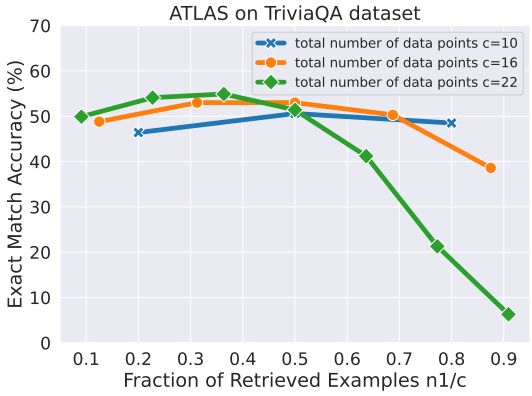


Figure 5.2: We compare the performance of RAG using examples ($c = n_1$) versus passages ($c = n_2$).

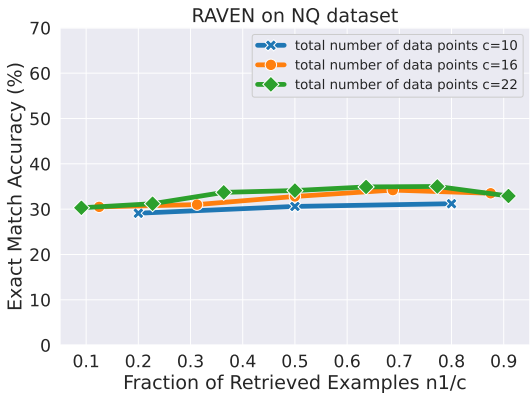
RAG Examples v.s. RAG External Passages Next, we compare using RAG examples from training data only ($m = 0, n_1 = c, n_2 = 0$) and RAG passages from external corpora only ($m = 0, n_1 = 0, n_2 = c$). The results in Figure 5.2 show interesting patterns. For RAG examples only, with more examples, the performance first significantly improves but later drops. This suggests that the first few examples



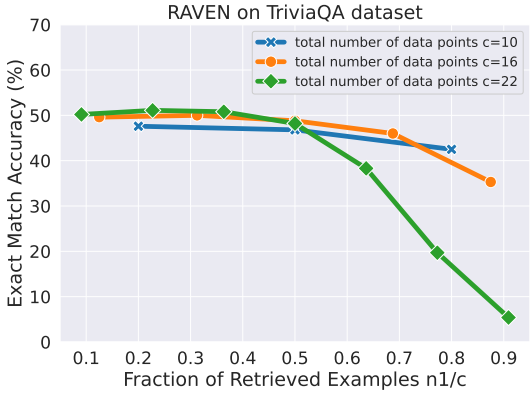
(a) ATLAS Performance as a function of n_1/c under different data points c on NQ.



(b) ATLAS Performance as a function of n_1/c under different data points c on TriviaQA.



(c) RAVEN Performance as a function of n_1/c under different data points c on NQ.



(d) RAVEN Performance as a function of n_1/c under different data points c on TriviaQA.

Figure 5.3: Performance sensitivity to the ratio n_1/n under different data points c, where n_1 refers to retrieved examples and n_2 to passages.

are highly relevant but later ones contain more noise than signal. In contrast, for RAG passages only, the performance increases more slowly but steadily for larger budgets. This suggests the passages retrieved are noisy but still have relevant signals. This aligns with our noise modeling. When n_1 is small ($\leqslant 20$ for NQ and $\leqslant 10$ for TriviaQA), RAG examples resemble *uniform noise* due to the relevance of

retrieved examples. As n_1 increases, n_1 introduces more irrelevant or conflicting examples (i.e., *non-uniform noise*). On the other hand, n_2 resembles a *uniform noise* regime as the retrieval pool is broad with relevant data but also noisy.

When the retrieval budget is small, retrieval from training examples yield higher accuracy than from passages, even though both operate in the uniform-noise regime. This discrepancy follows from the mixture-noise effects: a passage judged relevant may still lack any answer-bearing text, raising its effective noise level relative to examples. Furthermore, the significant drop for the retrieval from examples as opposed to retrieval from passages can be explained by the size difference for the training data and passages pool (i.e. Wikipedia). Since the passages provide a denser coverage of the semantic space, more passages will remain relevant as opposed to examples. In all, our theory covers both practical data types and matches the empirical results.

Ratio between RAG Examples and Passages The different noise properties of the two kinds of RAG data imply that we should find a proper ratio between them when the total budget c is fixed. Figure 5.3 in the appendix shows that as the ratio n_1/c increases, the performance initially improves—benefiting from signal information—but eventually declines as low-quality examples dominate the context. This again supports our theoretical view of signal versus noise in the retrieved data. The results demonstrate that performance initially improves as more signal (examples) is added, but eventually declines due to increasing noise from irrelevant or low-quality examples. This supports the theoretical perspective of balancing signal and noise in retrieval-augmented inputs.

5.6 Conclusion and Limitations

We model RAG as query-dependent noisy in-context learning, derive the first finite-sample error bounds for linear regression that isolate the contributions of retrieval signal and noise, and extend those bounds to different noise regimes and test-time

training. Experiments on Natural Questions, TriviaQA with RAVEN, and ATLAS corroborated our theoretical analysis.

Regarding limitations, our bounds focus on the linear setting, opening avenues for future studies on nonlinear methods like kernels and neural networks. While our framework accounts for common RAG noise models, new models may be needed for other types of RAG data. A further direction is to combine RAG with test-time training, studying how on-the-fly adaptation affects both theoretical guarantees and empirical performance. Our experiments feature representative models and datasets, but future research can explore newer retrievers, LLMs like Qwen 3 and Llama 4, and more advanced RAG applications.

A APPENDIX FOR CHAPTER 2

Supplementary Material

Towards Evaluating the Robustness of Neural Networks Learned by Transduction

A.1 Experimental Details

A.1.1 General Setup

Computing Infrastructure

We run all experiments with PyTorch and NVIDIA GeForce RTX 2080Ti GPUs.

Dataset

We use three datasets MNIST, CIFAR-10 and GTSRB in our experiments. The details about these datasets are described below.

MNIST. The MNIST (LeCun, 1998) is a large dataset of handwritten digits. Each digit has 5,500 training images and 1,000 test images. Each image is a 28×28 grayscale. We normalize the range of pixel values to $[0, 1]$.

CIFAR-10. The CIFAR-10 (Krizhevsky et al., 2009) is a dataset of 32x32 color images with ten classes, each consisting of 5,000 training images and 1,000 test images. The classes correspond to dogs, frogs, ships, trucks, etc. We normalize the range of pixel values to $[0, 1]$.

GTSRB. The German Traffic Sign Recognition Benchmark (GTSRB) (Stallkamp et al., 2012) is a dataset of color images depicting 43 different traffic signs. The images are not of a fixed dimensions and have rich background and varying light conditions as would be expected of photographed images of traffic signs. There are about 34,799 training images, 4,410 validation images and 12,630 test images. We resize each image to 32×32. The dataset has a large imbalance in the number

of sample occurrences across classes. We use data augmentation techniques to enlarge the training data and make the number of samples in each class balanced. We construct a class preserving data augmentation pipeline consisting of rotation, translation, and projection transforms and apply this pipeline to images in the training set until each class contained 10,000 examples. We also preprocess images via image brightness normalization and normalize the range of pixel values to $[0, 1]$.

Implementation Details of the Attacks

We use Projected Gradient Descent (PGD) (Madry et al., 2018b) to solve the attack objectives of PGD attack, FPA, GMSA-AVG and GMSA-MIN. For GMSA-AVG, at the i-th iteration, when applying PGD on the data point $\mathbf{x}$ to generate the perturbation δ, we need to do one backpropagation operation for each model in $\{F^{(j)}\}_{j=0}^{i}$ per PGD step. We do the backpropagation for each model sequentially and then accumulate the gradients to update the perturbation δ since we might not have enough memory to store all the models and compute the gradients at once, especially when i is large. For GMSA-MIN, we find that it requires more PGD steps to solve the attack objective at the i-th iteration where we need to attack $i+1$ models simultaneously. Thus, we scale the number of PGD steps at the i-th iteration by a factor of $i+1$ for GMSA-MIN.

A.1.2 Setup for RMC Experiments

We follow the original settings in Wu et al. (2020b) to perform experiments on MNIST and CIFAR-10 datasets to evaluate the adversarial robustness of RMC. We consider two kinds of base models for RMC: one is the model trained via standard supervised training; the other is the model trained using the adversarial training (Madry et al., 2018b). We describe the settings for each dataset below.

MNIST

Model architecture and training configuration. We use a neural network with two convolutional layers, two full connected layers and batch normalization layers.

For both standard training and adversarial training, we train the model for 100 epochs using the Adam optimizer with a batch size of 128 and a learning rate of 10^{-3}. We use the L_∞ norm PGD attack as the adversary for adversarial training with a perturbation budget ϵ of 0.3, a step size of 0.01, and number of steps of 40.

RMC configuration. We set $K = 1024$. Suppose the clean training set is $\mathbb{D}$. Let $\mathbb{D}'$ contain $|\mathbb{D}|$ clean inputs and $|\mathbb{D}|$ adversarial examples. So $N' = 2|\mathbb{D}|$. We generate the adversarial examples using the L_∞ norm PGD attack with a perturbation budget ϵ of 0.3, a step size of 0.01, and number of steps of 100. We extract the features from the penultimate layer of the model and use the Euclidean distance in the feature space of the model to find the top-K nearest neighbors of the inputs. When adapting the model, we use Adam as the optimizer and set the learning rate to be 2×10^{-4}. We train the model until the early-stop condition holds. That is the training epoch reaches 100 or the validation loss doesn't decrease for 5 epochs.

Attack configuration. We use the same threat model for all attacks: L_∞ norm perturbation with a perturbation budget ϵ of 0.3. Cross entropy loss is used as the loss function for PGD attack, FPA, GMSA-AVG and GMSA-MIN: $L_a(F, V) = \frac{1}{|V|} \sum_{(\mathbf{x},y)\in V} -\log f(\mathbf{x})_y$, where $f(\mathbf{x})$ is the softmax output of the model F. We use PGD with a step size of 0.01, the number of steps of 100, random start and no restarts. We set $T = 9$ for FPA, GMSA-AVG and GMSA-MIN.

CIFAR-10

Model architecture and training configuration. We use the ResNet-32 network (He et al., 2016). For both standard training and adversarial training, we train the model for 100 epochs using Stochastic Gradient Decent (SGD) optimizer with Nesterov momentum and learning rate schedule. We set momentum 0.9 and ℓ_2 weight decay with a coefficient of 10^{-4}. The initial learning rate is 0.1 and it decreases by 0.1 at 50, 75 and 90 epoch respectively. The batch size is 128. We augment the training images using random crop and random horizontal flip. We use the L_∞ norm PGD attack as the adversary for adversarial training with a perturbation budget ϵ of 8/255, a step size of 2/255, and number of steps of 10.

RMC configuration. We set $K = 1024$. Suppose the clean training set is $\mathbb{D}$. Let $\mathbb{D}'$ contain $|\mathbb{D}|$ clean inputs and $4|\mathbb{D}|$ adversarial examples. So $N' = 5|\mathbb{D}|$. We generate the adversarial examples using the L_∞ norm PGD attack with a perturbation budget ϵ of $8/255$, a step size of $1/255$, and number of steps of 40. We extract the features from the penultimate layer of the model and use the Euclidean distance in the feature space of the model to find the top-K nearest neighbors of the inputs. We use Adam as the optimizer and set the learning rate to be 2.5×10^{-5}.

Attack configuration. We use the same threat model for all attacks: L_∞ norm perturbation with a perturbation budget ϵ of $8/255$. Cross entropy loss is used as the loss function for PGD attack, FPA, GMSA-AVG and GMSA-MIN: $L_a(F, V) = \frac{1}{|V|} \sum_{(\mathbf{x},y)\in V} -\log f(\mathbf{x})_y$, where $f(\mathbf{x})$ is the softmax output of the model F. We use PGD with a step size of $1/255$, the number of steps of 40, random start and no restarts. We set $T = 9$ for FPA, GMSA-AVG and GMSA-MIN.

A.1.3 Setup for DENT Experiments

DENT configuration. We perform experiments to evaluate the best version of DENT (DENT+ in Wang et al. (2021)) on CIFAR-10 following the experimental settings in Wang et al. (2021). We use the pre-trained robust models on CIFAR-10 under the L_∞ norm perturbation threat model from RobustBench Model Zoo[1] as the static base models for DENT, including models with model ID Wu2020Adversarial_extra (Wu et al., 2020a), Carmon2019Unlabeled (Carmon et al., 2019), Sehwag2020Hydra (Sehwag et al., 2020), Wang2020Improving (Wang et al., 2020), Hendrycks2019Using (Hendrycks et al., 2019), Wong2020Fast (Wong et al., 2020), and Ding2020MMA (Ding et al., 2020). For the test-time adaptation, only the affine scale γ and shift β parameters in the batch normalization layers of the base model are updated. DENT updates sample-wise with different affine parameters (γ_i, β_i) for each input $\mathbf{x}_i$. The input adaptation of Σ is not used as suggested in Wang et al. (2021). The model is adapted for six steps by AdaMod (Ding et al., 2019) with learning rate of 0.006, batch size of 128 and no weight decay. The adaptation objective is entropy minimization with

[1] https://github.com/RobustBench/robustbench

the information maximization regularization:

$$\min_{\theta_i} \sum_{i=1}^{b} -\sum_{c=1}^{C} f(\mathbf{x}_i;\theta_i)_c \cdot \log f(\mathbf{x}_i;\theta_i)_c + \sum_{c=1}^{C}\sum_{i=1}^{b} f(\mathbf{x}_i;\theta_i)_c \cdot \log \sum_{i=1}^{b} f(\mathbf{x}_i;\theta_i)_c \quad (A.1)$$

where b is the batch size, C is the number of classes and $f(\mathbf{x}_i;\theta_i)$ is the softmax output of the model f with the affine parameters $\theta_i = (\gamma_i, \beta_i)$ for the input $\mathbf{x}_i$.

Attack configuration. We use the same threat model for all attacks: L_∞ norm perturbation with a perturbation budget ϵ of 8/255. For PGD attack, FPA, GMSA-AVG and GMSA-MIN, we use the following loss function to find adversarial examples with high confidence: $L_a(F,V) = \frac{1}{|V|}\sum_{(\mathbf{x},y)\in V} \max_{k\neq y} f(\mathbf{x})_k$, where $f(\mathbf{x})$ is the softmax output of the model F. However, it is hard to optimize this loss function. Thus, we use two alternative loss functions to find adversarial examples. One is the untargeted CW loss (Carlini and Wagner, 2017b): $L_a^1(F,V) = \frac{1}{|V|}\sum_{(\mathbf{x},y)\in V} -Z(\mathbf{x})_y + \max_{k\neq y} Z(\mathbf{x})_k$, where $Z(\mathbf{x})$ is the logits of the model F (the output of the layer before the softmax layer). The other is the targeted CW loss: $L_a^2(F,V) = \frac{1}{|V|}\sum_{(\mathbf{x},y)\in V} -Z(\mathbf{x})_y + Z(\mathbf{x})_t$, where t is the targeted label and $t \neq y$. For each attack, we use 14 PGD subroutines to solve its attack objective, including 5 PGD subroutines using the untargeted CW loss L_a^1 with different random restarts and 9 PGD subroutines using the targeted CW loss L_a^2 with different targeted labels. So for each clean test input $\mathbf{x}$, these PGD subroutines will return 14 adversarial examples $\mathbf{x}'_1, \ldots, \mathbf{x}'_{14}$. Among these adversarial examples, we select the one that maximizes the attack loss with the loss function $L_a(F,V)$ as the final adversarial example $\mathbf{x}'$ for $\mathbf{x}$. We use the same hyper-parameters for all PGD subroutines: the step size is 1/255, the number of steps is 100, and the random start is used. We set $T = 2$ for FPA, GMSA-AVG and GMSA-MIN.

A.1.4 Setup for DANN Experiments

We perform experiments on MNIST and CIFAR-10 datasets. We describe the settings for each dataset below.

MNIST

Model architecture. We use the same model architecture as the one used in Chuang et al. (2020), which is shown below.

Encoder
nn.Conv2d(3, 64, kernel_size=5)
nn.BatchNorm2d
nn.MaxPool2d(2)
nn.ReLU
nn.Conv2d(64, 128, kernel_size=5)
nn.BatchNorm2d
nn.Dropout2d
nn.MaxPool2d(2)
nn.ReLU
nn.Conv2d(128, 128, kernel_size=3, padding=1)
nn.BatchNorm2d
nn.ReLU
$\times 2$

Predictor
nn.Conv2d(128, 128, kernel_size=3, padding=1)
nn.BatchNorm2d
nn.ReLU
$\times 3$
flatten
nn.Linear(2048, 256)
nn.BatchNorm1d
nn.ReLU
nn.Linear(256, 10)
nn.Softmax
Discriminator
nn.Conv2d(128, 128, kernel_size=3, padding=1)
nn.ReLU
$\times 5$
Flatten
nn.Linear(2048, 256)
nn.ReLU
nn.Linear(256, 2)
nn.Softmax

Training configuration. We train the models for 100 epochs using the Adam optimizer with a batch size of 128 and a learning rate of 10^{-3}. For the representation matching in DANN, we adopt the original progressive training strategy for the discriminator (Ganin et al., 2016b) where the weight α for the domain-invariant loss is initiated at 0 and is gradually changed to 0.1 using the schedule $\alpha = (\frac{2}{1+\exp(-10 \cdot p)} - 1) \cdot 0.1$, where p is the training progress linearly changing from 0 to 1.

Attack configuration. We use the same threat model for all attacks: L_∞ norm perturbation with a perturbation budget ϵ of 0.3. Cross entropy loss is used as the loss function for PGD attack, FPA, GMSA-AVG and GMSA-MIN: $L_a(F, V) = \frac{1}{|V|} \sum_{(\mathbf{x},y)\in V} -\log f(\mathbf{x})_y$, where $f(\mathbf{x})$ is the softmax output of the model F. We use

PGD with a step size of 0.01, the number of steps of 200, random start and no restarts. We set $T = 9$ for FPA, GMSA-AVG and GMSA-MIN.

CIFAR-10

Model architecture. We use the ResNet-18 network (He et al., 2016) and extract the features from the third basic block for representation matching. The detailed model architecture is shown below.

Encoder
nn.Conv2d(3, 64, kernel_size=3)
nn.BatchNorm2d
nn.ReLU
BasicBlock(in_planes=64, planes=2, stride=1)
BasicBlock(in_planes=128, planes=2, stride=2)
BasicBlock(in_planes=256, planes=2, stride=2)
Predictor
BasicBlock(in_planes=512, planes=2, stride=2)
avg_pool2d
flatten
nn.Linear(512, 10)
nn.Softmax
Discriminator
BasicBlock(in_planes=512, planes=2, stride=2)
avg_pool2d
flatten
nn.Linear(512, 2)
nn.Softmax

Training configuration. We train the models for 100 epochs using stochastic gradient decent (SGD) optimizer with Nesterov momentum and learning rate schedule. We set momentum 0.9 and ℓ_2 weight decay with a coefficient of 10^{-4}. The

initial learning rate is 0.1 and it decreases by 0.1 at 50, 75 and 90 epoch respectively. The batch size is 64. We augment the training images using random crop and random horizontal flip. For the representation matching in DANN, we adopt the original progressive training strategy for the discriminator (Ganin et al., 2016b) where the weight α for the domain-invariant loss is initiated at 0 and is gradually changed to 1 using the schedule $\alpha = \frac{2}{1+\exp(-10 \cdot p)} - 1$, where p is the training progress linearly changing from 0 to 1.

Attack configuration. We use the same threat model for all attacks: L_∞ norm perturbation with a perturbation budget ϵ of 8/255. Cross entropy loss is used as the loss function for PGD attack, FPA, GMSA-AVG and GMSA-MIN: $L_a(F, V) = \frac{1}{|V|} \sum_{(\mathbf{x},y)\in V} -\log f(\mathbf{x})_y$, where $f(\mathbf{x})$ is the softmax output of the model F. We use PGD with a step size of 1/255, the number of steps of 100, random start and no restarts. We set $T = 9$ for FPA, GMSA-AVG and GMSA-MIN.

A.1.5 Setup for TADV Experiments

We perform experiments on MNIST and CIFAR-10 datasets. We describe the settings for each dataset below.

MNIST

Model architecture and Training configuration. We use the LeNet network architecture. We train the models for 100 epochs using the Adam optimizer with a batch size of 128 and a learning rate of 10^{-3}. We use the L_∞ norm PGD attack as the adversary to generate adversarial training examples with a perturbation budget ϵ of 0.3, a step size of 0.01, and number of steps of 40. We train on 50% clean and 50% adversarial examples per batch.

Attack configuration. We use the same threat model for all attacks: L_∞ norm perturbation with a perturbation budget ϵ of 0.3. Cross entropy loss is used as the loss function for PGD attack, FPA, GMSA-AVG and GMSA-MIN: $L_a(F, V) = \frac{1}{|V|} \sum_{(\mathbf{x},y)\in V} -\log f(\mathbf{x})_y$, where $f(\mathbf{x})$ is the softmax output of the model F. We use

PGD with a step size of 0.01, the number of steps of 200, random start and no restarts. We set $T = 9$ for FPA, GMSA-AVG and GMSA-MIN.

CIFAR-10

Model architecture and Training configuration. We use the ResNet-20 network architecture (He et al., 2016). We train the models for 110 epochs using stochastic gradient decent (SGD) optimizer with Nesterov momentum and learning rate schedule. We set momentum 0.9 and ℓ_2 weight decay with a coefficient of 5×10^{-4}. The initial learning rate is 0.1 and it decreases by 0.1 at 100 and 105 epoch respectively. The batch size is 128. We augment the training images using random crop and random horizontal flip. We use the L_∞ norm PGD attack as the adversary to generate adversarial training examples with a perturbation budget ϵ of 8/255, a step size of 2/255, and number of steps of 10. We train on 50% clean and 50% adversarial examples per batch.

Attack configuration. We use the same threat model for all attacks: L_∞ norm perturbation with a perturbation budget ϵ of 8/255. Cross entropy loss is used as the loss function for PGD attack, FPA, GMSA-AVG and GMSA-MIN: $L_a(F, V) = \frac{1}{|V|} \sum_{(\mathbf{x},y)\in V} -\log f(\mathbf{x})_y$, where $f(\mathbf{x})$ is the softmax output of the model F. We use PGD with a step size of 1/255, the number of steps of 100, random start and no restarts. We set $T = 9$ for FPA, GMSA-AVG and GMSA-MIN.

A.1.6 Setup for URejectron Experiments

We use a subset of the GTSRB augmented training data for our experiments, which has 10 classes and contains 10,000 images for each class. We implement URejectron (Goldwasser et al., 2020b) on this dataset using the ResNet18 network (He et al., 2016) in the transductive setting. Following Goldwasser et al. (2020b), we implement the basic form of the URejectron algorithm, with $T = 1$ iteration. That is we train a discriminator h to distinguish between examples from P and Q, and train a classifier F on P. Specifically, we randomly split the data into a training set D_{train} containing 63,000 images, a validation set D_{val} containing 7,000 images and a

test set D_{test} containing 30,000 images. We then use the training set D_{train} to train a classifier f using the ResNet18 network. We train the classifier f for 10 epochs using Adam optimizer with a batch size of 128 and a learning rate of 10^{-3}. The accuracy of the classifier on the training set D_{train} is 99.90% and its accuracy on the validation set D_{val} is 99.63%. We construct a set $\tilde{x}$ consisting of 50% normal examples and 50% adversarial examples. The normal examples in the set $\tilde{x}$ form a set z. We train the discriminator h on the set D_{train} (with label 0) and the set $\tilde{x}$ (with label 1). We then evaluate URejectron's performance on $\tilde{x}$: under a certain threshold used by the discriminator h, we measure the fraction of normal examples in z that are rejected by the discriminator h and the error rate of the classifier f on the examples in the set $\tilde{x}$ that are accepted by the discriminator h. The set z can be D_{test} or a set of corrupted images generated on D_{test}. We use the method proposed in Hendrycks and Dietterich (2019) to generate corrupted images with the corruption type of brightness and the severity level of 1. The accuracy of the classifier on the corrupted images is 98.90%. The adversarial examples in $\tilde{x}$ are generated by the PGD attack (Madry et al., 2018b) or the CW attack (Carlini and Wagner, 2017b). For PGD attack, we use L_∞ norm with perturbation budget $\epsilon = 8/255$ and random initialization. The number of iterations is 40 and the step size is 1/255. The robustness of the classifier under the PGD attack is 3.66%. For CW attack, we use L_2 norm as distance measure and set $c = 1$ and $\kappa = 0$. The learning rate is 0.01 and the number of steps is 100. The robustness of the classifier under the CW attack is 0.00%.

A.1.7 Evaluate DENT under the adversarially-ordered game

We evaluate the robustness of DENT under the adversarially-ordered game where the adversary can choose a "worst-case" order of perturbed data points after receiving a large amount of test data and then sends them in batches one at a time to the defender. Specifically, each time the attacker will generate adversarial examples on up to 256 data points, and then sort the adversarial examples by their labels from lowest to highest, and finally send the sorted adversarial examples in

Base Model	Robustness					
	Static	DENT				
	AA	AA	PGD	FPA	GMSA-AVG	GMSA-MIN
Wu et al. (2020a)	58.00	58.00	50.40	**50.30**	50.40	50.40
Carmon et al. (2019)	57.30	58.00	**51.80**	**51.80**	**51.80**	**51.80**
Sehwag et al. (2020)	54.90	55.90	50.10	**49.80**	50.00	50.00
Wang et al. (2020)	53.60	55.90	49.00	49.10	48.90	**48.70**
Hendrycks et al. (2019)	51.80	53.10	**48.10**	48.20	48.30	48.30
Wong et al. (2020)	42.40	44.60	40.10	**39.90**	40.00	40.00
Ding et al. (2020)	39.70	42.10	36.20	35.70	35.30	**35.00**

Table A.1: Results of evaluating DENT on CIFAR-10 under the adversarially-ordered game. **Bold** numbers are worst results.

batches one at a time to the defender. Other experimental settings are the same as those described in Appendix A.1.3. The results in Table A.1 show that under the adversarially-ordered game, we can reduce the robustness of DENT to be lower than that of static base models.

A.1.8 Multiple Random Runs of the RMC Experiment

In Section 2.6, we describe the experimental setup for evaluating RMC. The performance of RMC is evaluated on a sequence of test points $\mathbf{x}^{(1)}, \cdots, \mathbf{x}^{(n)}$ randomly sampled from the test dataset. We repeat this experiment five times with different random seeds and report the mean and standard deviation of the results over the multiple random runs of the experiment. When evaluating the robustness of RMC, we only use the GMSA-AVG attack and the GMSA-MIN attack since they are the strongest attacks. From Table A.2, we can see that the results don't vary much across different random runs and the conclusion that the proposed GMSA attacks can break RMC still holds.

Dataset	Base Model	Accuracy		Robustness		
		Static	RMC	Static	RMC	
				AA	GMSA-AVG	GMSA-MIN
MNIST	Standard	99.40±0.15	98.62±0.23	0.00±0.00	0.54±0.05	0.80±0.19
	Madry et al.	99.24±0.22	96.50±0.91	87.86±0.71	57.14±5.83	59.48±4.52
CIFAR-10	Standard	93.96±0.38	93.56±0.34	0.00±0.00	8.50±1.29	8.92±0.93
	Madry et al.	83.70±1.11	91.46±0.71	43.58±1.30	38.50±2.07	39.00±1.58

Table A.2: Results of evaluating RMC. We also evaluate the static base model for comparison. We report the mean and standard deviation of the accuracy or robustness (mean±std) over the five random runs of the experiment.

B APPENDIX FOR CHAPTER 3

B.1 Proof Details

Before introducing the proof for the generalization results, we first need to make some additional definitions. We define the *empirical robust risk* as

$$\hat{R}_{\mathcal{U}}(h;S) = \sum_{(x,y)\in S}\left[\sup_{z\in\mathcal{U}(x)} \mathbb{1}\{h(z)\neq y\}\right]$$

And we can define the *empirical robust risk under rejection* accordingly:

$$\hat{R}^{rej}_{\mathcal{U}}(h;S) = \sum_{(x,y)\in S}\left[\sup_{z\in\mathcal{U}(x)} \mathbb{1}\{h(x)\neq y \vee h(z)\notin\{y,\perp\}\}\right]$$

And we can define the corresponding robust empirical risk minimization procedure (under rejection) as follows:

$$\text{RERM}_{\mathcal{H}}(S) := \underset{h\in\mathcal{H}}{\text{argmin}}\ \hat{R}_{\mathcal{U}}(h;S)$$

$$\text{RERM}^{rej}_{\mathcal{H}}(S) := \underset{h\in\mathcal{H}}{\text{argmin}}\ \hat{R}^{rej}_{\mathcal{U}}(h;S)$$

B.1.1 Rejection Only: Realizable Case

Definition B.1 (Realizable Robust PAC Learnability under Rejection)**.** *For $\mathcal{Y}=\{0,1\}$, $\forall\epsilon,\delta\in(0,1), \mathcal{H}=\mathcal{H}_c\times\mathcal{H}_r$, the sample complexity of realizable robust (ϵ,δ) - PAC learning of $\mathcal{H}$ with respect adversary $\mathcal{U}$ under rejection, denoted as $\mathcal{M}_{RE}(\epsilon,\delta;\mathcal{H},\mathcal{U})$, is defined as the smallest $m\in\mathbb{N}\cup\{0\}$ for which there exists a learning rule $\mathcal{A}:(\mathcal{X}\times\mathcal{Y})^m\longmapsto(\mathcal{Y}\cup\{\perp\})^{\mathcal{X}}$ s.t. for every data distribution $\mathcal{D}$ over $(\mathcal{X}\times\mathcal{Y})^m$ where there exists a predictor with rejection option $h^*\in\mathcal{H}$ with 0 risk, $R_{\mathcal{U},rej}(h^*;\mathcal{D})=0$ with probability at least $1-\delta$*

over $S \sim \mathcal{D}^m$,

$$R^{\text{rej}}_{\mathcal{U}}(\mathcal{A}(S); \mathcal{D}) \leqslant \epsilon$$

If no such m *exists,* $\mathcal{M}_{RE}(\epsilon, \delta; \mathcal{H}, \mathcal{U}) = \infty$. *We say that* $\mathcal{H}$ *is robustly PAC learnable under rejection in the realizable setting with respect to adversary* $\mathcal{U}$ *if* $\forall \epsilon, \delta \in (0,1)$, $\mathcal{M}_{RE}(\epsilon, \delta; \mathcal{H}, \mathcal{U})$ *is finite.*

Theorem B.2 (Sample Complexity for Realizable Robust PAC Learning under Rejection)**.** *In the realizable setting, for any* $\mathcal{H} = \mathcal{H}_c \times \mathcal{H}_r$ *and* $\mathcal{U}$, *and any* $\epsilon, \delta \in (0, 1/2)$,

$$\mathcal{M}_{RE}(\epsilon, \delta; \mathcal{H}, \mathcal{U}) = 2^{O((d_r + d_c)\log(d_r + d_c))} \frac{1}{\varepsilon} \log\left(\frac{1}{\varepsilon}\right) + O\left(\frac{1}{\varepsilon} \log\left(\frac{1}{\delta}\right)\right) \tag{B.1}$$

where $d_r = \text{VC}(\mathcal{H}_r)$, $d_c = \text{VC}(\mathcal{H}_c)$.

The idea of the proof is to adapt the classical sample compression argument (Littlestone and Warmuth, 1986) with improvements based on (Montasser et al., 2019; Hanneke et al., 2019; Moran and Yehudayoff, 2016). The generalization result in the inductive case directly comes from Equation (B.26).

Proof. First, we define the concept of *sample compression scheme* and *sample compression algorithm*.

Definition B.3 (Sample Compression Scheme)**.** *Given* $\forall m \in \mathbb{N}$ *samples,* $S \sim \mathcal{D}^m$, *a sample compression scheme of size* k *is defined by the following pair of functions:*

1. *Compression function* $\kappa : (\mathcal{X} \times \mathcal{Y})^m \mapsto (\mathcal{X} \times \mathcal{Y})^{\leqslant k}$.
2. *Reconstruction function:* $\rho : (\mathcal{X} \times \mathcal{Y})^{\leqslant k} \mapsto \mathcal{H}$.

An algorithm $\mathcal{A}$ *is a sample compression algorithm if* $\exists \kappa, \rho$ *s.t.* $\mathcal{A}(S) = (\kappa \circ \rho)(S)$.

Fix $\epsilon, \delta \in (0,1)$, $m > 2(d_r + d_c)\log(d_r + d_c)$. Let the compression parameter, $n = \mathcal{O}((d_r + d_c)\log(d_r + d_c))$. Let $\mathcal{D}$ be any distribution, then by realizability of the learner, $\inf_{h \in \mathcal{H}} R^{\text{rej}}_{\mathcal{U}}(h; \mathcal{D}) = 0$. Thus, $\forall S$ sampled from $\mathcal{D}$, we have $\hat{R}^{\text{rej}}_{\mathcal{U}}(\text{RERM}^{\text{rej}}_{\mathcal{H}}(S); S) = 0$.

Compression First, we define a compression function κ as through the following inflation and discretization procedure. Given the training data $S := \{(x_i, y_i)\}_{i\in[m]}$, we define the following index mapping:

$$I(x) = \min\{i \in [m] : x \in \mathcal{U}(x_i)\}, \quad \forall x \in \bigcup_{i\in[m]} \mathcal{U}(x_i). \tag{B.2}$$

In another word, this index function outputs the first indexed training sample to include x in its neighborhood.

Then, we consider the set of RERM mapping learned by a size n subset of the training data:

$$\hat{\mathcal{H}} = \{\text{RERM}^{\text{rej}}_{\mathcal{H}}(L) : L \subseteq S, |L| = n\}. \tag{B.3}$$

Note that

$$|\hat{\mathcal{H}}| \leqslant |\{L : L \subseteq S, |L| = n\}| = \binom{m}{n} \leqslant \left(\frac{em}{n}\right)^n. \tag{B.4}$$

Then, we inflate the data in the following way:

$$S_{\mathcal{U}} = \bigcup_{i\in[m]} \left\{\left(x_{I(x)}, x, y_{I(x)}\right) : x \in \mathcal{U}\left(x_i\right)\right\}. \tag{B.5}$$

Note that $x_{I(x)}$ can be different from x_i.

Let's define the following transformation T:

$$T(h)(x, x', y) := \mathbb{1}\{h(x) \neq y \vee h(x') \notin \{y, \perp\}\},\ h \in \mathcal{H}. \tag{B.6}$$

And we can obtain the transformed hypothesis class $T(\mathcal{H}) := \{T(h)|h \in \mathcal{H}\}$.

Now, we proceed to define the *dual space* $\mathcal{G}$ of $T(\mathcal{H})$ as the following set of functions.

$$\mathcal{G} := \{g_{(x,x',y)}|g_{(x,x',y)}(t) = t(x, x', y),\ t \in T(\mathcal{H})\}. \tag{B.7}$$

We denote the VC dimension of the dual space as $VC^*(T(\mathcal{H})) := VC(\mathcal{G})$.

By Lemma B.1.1,

$$VC(T(\mathcal{H})) = \mathcal{O}\left((d_r + d_c)\log(d_r + d_c)\right). \tag{B.8}$$

By the classic result in (Assouad, 1983), the VC dimension of the dual space satisfies the following inequality:

$$VC^*(T(\mathcal{H})) < 2^{VC(T(\mathcal{H}))+1}. \tag{B.9}$$

Now, we can construct the compressed dataset $\hat{S}_{\mathcal{U}}$ as the following. For each $(x, x', y) \in S_{\mathcal{U}}$, $\{g_{(x,x',y)}(t)\}_{t\in T(\hat{\mathcal{H}})}$ gives a labeling. When ranging over $(x, x', y) \in S_{\mathcal{U}}$, the labeling may not be unique. So for each unique labeling, we choose a representative $(x, x', y) \in S_{\mathcal{U}}$, and let $\hat{S}_{\mathcal{U}}$ be the set of the representatives. That is:

$$\hat{S}_{\mathcal{U}} = \left\{ (x, x', y) \in S_{\mathcal{U}} \;\middle|\; \{g_{(x,x',y)}(t)\}_{t\in T(\hat{\mathcal{H}})} \text{ provides a unique labeling} \right\}. \tag{B.10}$$

Intuitively, $\hat{S}_{\mathcal{U}}$ split the infinite size dataset $S_{\mathcal{U}}$ into finite size according to the labeling of $T(\hat{\mathcal{U}})$ on the dual space. Thus, $\hat{S}_{\mathcal{U}}$ is not necessarily unique but always exists. And $|\hat{S}_{\mathcal{U}}|$ equals the number of possible labeling for $T(\hat{\mathcal{H}})$.

Let $d_* := VC(\mathcal{G}) = VC^*(T(\mathcal{H}))$ denote the VC-dimension of $\mathcal{G}$, the dual hypothesis class of $T(\hat{\mathcal{H}})$ (Assouad, 1983). By applying Sauer's Lemma, we obtain that for $|T(\hat{\mathcal{H}})| > d_*$,

$$|\hat{S}_{\mathcal{U}}| \leqslant \left(\frac{e|T(\hat{\mathcal{H}})|}{d_*}\right)^{d_*}. \tag{B.11}$$

Let $n = \Theta\left(\mathrm{VC}\left(\mathsf{T}\left(\mathcal{H}\right)\right)\right)$. For $m \geqslant n$, we have

$$|\hat{S}_{\mathcal{U}}| \leqslant \left(e|\mathsf{T}(\hat{\mathcal{H}})|\right)^{d_*} \tag{B.12}$$

$$\leqslant \left(e|\hat{\mathcal{H}}|\right)^{d_*} \tag{B.13}$$

$$\leqslant \left(e\left(\frac{em}{n}\right)^{n}\right)^{d_*} \tag{B.14}$$

$$\leqslant \left(\frac{e^2 m}{n}\right)^{n d_*} \tag{B.15}$$

$$= \left(\frac{e^2 m}{\mathrm{VC}(\mathsf{T}(\mathcal{H}))}\right)^{\Theta(\mathrm{VC}(\mathsf{T}(\mathcal{H}))\cdot \mathrm{VC}(\mathsf{T}(\mathcal{H}^*)))}. \tag{B.16}$$

Now we have obtain the compression map: $\kappa(S) = \hat{S}_{\mathcal{U}}$.

Reconstruction Now, we want to reconstruct a hypothesis from $\hat{S}_{\mathcal{U}}$. First, suppose we have a data distribution over $\hat{S}_{\mathcal{U}}$, denoted as $\mathcal{P}$. This distribution $\mathcal{P}$ over samples will be later used in the α—boosting procedure.

Then, we sample the set of n i.i.d. samples from $\mathcal{P}$ and obtain $S' \in \hat{S}_{\mathcal{U}}$. By classic PAC learning guarantee (Blumer et al., 1989), for $n = \Theta(\mathrm{VC}(\mathsf{T}(\mathcal{H}))) = \Theta(d_r + d_c)\log(d_r + d_c)$, we have with non-zero probability $\forall t \in \mathsf{T}(\mathcal{H})$ with $\sum_{(x,x',y)\in S'} t(x,x',y) = 0$ implies $\mathbb{E}_{(x,x',y)\sim\mathcal{P}} t(x,x',y) < 1/9$. Let $L = \{(x,y) : (x,x',y) \in S'\} \subseteq S$, and $t_{\mathcal{P}} = \mathsf{T}(\mathrm{RERM}^{\mathrm{rej}}_{\mathcal{H}}(L))$. Since $\hat{R}^{\mathrm{rej}}_{\mathcal{U}}(\mathrm{RERM}^{\mathrm{rej}}_{\mathcal{H}}(L); L) = 0$, $\forall (x,x',y) \in S', t_{\mathcal{P}}(x,x',y) = 0$. Thus, $\forall\, \mathcal{P}$ over $\hat{S}_{\mathcal{U}}$, there exists a weak learner $t_{\mathcal{P}} \in \mathsf{T}(\hat{\mathcal{H}})$, s.t. $\mathbb{E}_{(x,x',y)\sim\mathcal{P}}\, t_{\mathcal{P}}(x,x',y) < 1/9$.

Now, we use $t_{\mathcal{P}}$ as a *weak hypothesis* in a boosting algorithm, specifically α—boost algorithm from (Schapire and Freund, 2012) with $\hat{S}_{\mathcal{U}}$ as the dataset and $\mathcal{P}_k$ generated at each round of the algorithm. Then with appropriate choice of α, running α—boosting for $K = \mathcal{O}(\log(|\hat{S}_{\mathcal{U}}|))$ rounds gives a sequence of hypothesis

$h_1, \ldots, h_K \in \hat{\mathcal{H}}$ and the corresponding $t_i = T(h_i)$ such that $\forall (x, x', y) \in \hat{S}_{\mathcal{U}}$,

$$\frac{1}{K} \sum_{k=1}^{K} \mathbb{1}\{h_k(x) \neq y \vee h_k(x') \notin \{y, \perp\}\} \tag{B.17}$$

$$= \frac{1}{K} \sum_{k=1}^{K} t_k(x, x', y) \tag{B.18}$$

$$< \frac{2}{9} < \frac{1}{3}. \tag{B.19}$$

Since $\hat{S}_{\mathcal{U}}$ includes all the unique labellings, $\frac{1}{K} \sum_{k=1}^{K} t_k(x, x', y) < \frac{1}{3}$, $\forall (x, x', y) \in \hat{S}_{\mathcal{U}}$ implies

$$\frac{1}{K} \sum_{k=1}^{K} t_k(x, x', y) < \frac{1}{3}, \ \forall (x, x', y) \in S_{\mathcal{U}}. \tag{B.20}$$

Let $\bar{h} := \text{Majority}(h_1, \ldots, h_K)$, i.e., $\bar{h}$ outputs the prediction in $\mathcal{Y} \cup \{\perp\}$ that receives the most votes from $\{h_1, \ldots, h_K\}$. Then $\forall (x, x', y) \in \hat{S}_{\mathcal{U}}$,

$$\mathbb{1}\{\bar{h}(x) \neq y \vee \bar{h}(x') \notin \{y, \perp\}\} = 0. \tag{B.21}$$

This is because: (1) on x, less than $1/3$ of h_i's do not output y, so $\bar{h}(x) = y$; (2) on x', less than $1/3$ of h_i's do not output y or $\perp$, so the majority vote must be in y or $\perp$, i.e., $\bar{h}(x) \in \{y, \perp\}$.

In summary, given the same m training samples, we can simply find a $\bar{h}$ with 0 robust error on S:

$$\hat{R}_{\mathcal{U}}^{\text{rej}}(\bar{h}; \mathcal{D}) = \sum_{i=1}^{m} \left[\sup_{z \in \mathcal{U}(x)} \mathbb{1}\{\bar{h}(x) \neq y \vee \bar{h}(z) \notin \{y, \perp\}\} \right] = 0. \tag{B.22}$$

Now we have the compression set with size:

$$nK = \mathcal{O}(\text{VC}(T(\mathcal{H})) \log(|\hat{S}_{\mathcal{U}}|)) = \mathcal{O}(\text{VC}(T(\mathcal{H}))^2 \, \text{VC}^*(T(\mathcal{H})) \log(m / \text{VC}(T(\mathcal{H}))))$$

Then, we apply Lemma 11 of (Montasser et al., 2019) (Replacing $R_{\mathcal{U}}$ with $R^{\text{rej}}_{\mathcal{U}}$ still holds), we obtain for sufficiently large m, with probability at least $1-\delta$,

$$R^{\text{rej}}_{\mathcal{U}}(\bar{h};\mathcal{D}) \leqslant \mathcal{O}\left(\text{VC}(T(\mathcal{H}))^2\,\text{VC}^*(T(\mathcal{H}))\frac{1}{m}\log(m/\text{VC}(T(\mathcal{H})))\log(m)+\frac{1}{m}\log(1/\delta)\right). \tag{B.23}$$

We then can extend the sparsification procedure from (Moran and Yehudayoff, 2016; Montasser et al., 2019) to the rejection scenario. Since $t_1,\ldots,t_K \in T(\hat{\mathcal{H}})$, the classic uniform convergence results (Shalev-Shwartz and Ben-David, 2014) implies that we can sample $N = \mathcal{O}(\text{VC}^*(T(\mathcal{H})))$ i.i.d. indices $i_1,\ldots,i_N \sim \text{Uniform}([K])$ and obtain:

$$\sup_{(x,x',y)\in S_{\mathcal{U}}}\left|\frac{1}{N}\sum_{j=1}^{N} t_{i_j}(x,x',y)-\frac{1}{K}\sum_{i=1}^{T} t_i(x,x',y)\right| < \frac{1}{18} \tag{B.24}$$

And thus, we can combine Equation (B.17) with Equation (B.24) and obtain:

$$\forall(x,x',y)\in S_{\mathcal{U}}, \frac{1}{N}\sum_{j=1}^{N} t_{i_j}(x,x',y) \leqslant -\frac{1}{18}+\frac{1}{K}\sum_{i=1}^{K} t_k(x,x',y) < -\frac{1}{18}+\frac{4}{9}=\frac{1}{2}$$

we can further obtain an improved hypothesis $\bar{t}' := \text{Majority}(t_{i_1},\ldots t_{i_N})$ with

$$\bar{t}'(x,x',y)=0, \forall(x,x',y)\in S_{\mathcal{U}}$$

Thus, the compression set has a reduced size:

$$nN = \mathcal{O}(\text{VC}(T(\mathcal{H}))\cdot\text{VC}^*(T(\mathcal{H})))$$

Now, we apply Lemma 11 of (Montasser et al., 2019) and can obtain the following improved bound. Applying similar strategy from Equation (B.21), we can obtain

$$\bar{h}' := \text{Majority}(h_{i_1},\ldots h_{i_N}) = \rho(\hat{S}_{\mathcal{U}}) = \mathcal{A}(S) \tag{B.25}$$

which is our full reconstruction map.

Then, for large sample size $m \geqslant c\ \mathrm{VC}(T(\mathcal{H}))\,\mathrm{VC}^*(T(\mathcal{H}))$ (c is a sufficiently large constant), with probability at least $1-\delta$,

$$R_{\mathcal{U},\mathrm{rej}}(\bar{h}';\mathcal{D}) \leqslant \mathcal{O}\left(\mathrm{VC}(T(\mathcal{H}))\,\mathrm{VC}^*(\mathcal{H})\frac{1}{m}\log(m)+\frac{1}{m}\log(1/\delta)\right) \tag{B.26}$$

Plugging in Lemma Section B.1.1 and solving for m gives

$$\mathcal{M}_{\mathrm{RE}}(\epsilon,\delta;\mathcal{H},\mathcal{U}) = 2^{\mathcal{O}(\mathrm{VC}(T(\mathcal{H})))}\frac{1}{\varepsilon}\log\left(\frac{1}{\varepsilon}\right)+O\left(\frac{1}{\varepsilon}\log\left(\frac{1}{\delta}\right)\right) \tag{B.27}$$

$$= 2^{\mathcal{O}((d_r+d_c)\log(d_r+d_c))}\frac{1}{\varepsilon}\log\left(\frac{1}{\varepsilon}\right)+O\left(\frac{1}{\varepsilon}\log\left(\frac{1}{\delta}\right)\right) \tag{B.28}$$

□

Lemma [VC dimension of robust loss with rejection] Let $\mathrm{VC}(\mathcal{H}_c) = d_c$, and $\mathrm{VC}(\mathcal{H}_r) = d_r$. Then, $\mathrm{VC}(T(\mathcal{H})) = \mathcal{O}\left((d_r+d_c)\log(d_r+d_c)\right)$.

Proof. Suppose $d > d_r + d_c$.

By definition of VC dimension, the max number of labeling of d points is 2^d on $h \in T(\mathcal{H})$. And since the label of h is a deterministic function of h_c and h_r, by Sauer's Lemma, the number of labeling of h is at most $O(d^{d_r}) \times O(d^{d_c}) = O(d^{d_r+d_c})$.

Thus, $2^d = \mathcal{O}(d^{d_r+d_c})$. And $d = O((d_r+d_c)\log(d_r+d_c))$.

If $d < d_r + d_c$, $d = \mathcal{O}(d_r+d_c)\log(d_r+d_c)$ by definition.

□

B.1.2 Rejection Only: Agnostic Case

Now, we define notion of PAC learnability in the agnostic case under rejection setting as the follows:

Definition B.4 (Robust PAC Learnability under Rejection)**.** *For $\mathcal{Y} = \{0,1\}$, $\forall \epsilon, \delta \in (0,1)$, $\mathcal{H} = \mathcal{H}_c \times \mathcal{H}_r$, the sample complexity of robust (ϵ,δ) - PAC learning of $\mathcal{H}$ with*

respect to perturbation $\mathcal{U}$ under rejection, denoted as $\mathcal{M}_{AG}(\epsilon, \delta; \mathcal{H}, \mathcal{U})$, is defined as the smallest $m \in \mathbb{N} \cup \{0\}$ for which there exists a learning rule $\mathcal{A} : (\mathcal{X} \times \mathcal{Y})^m \longmapsto (\mathcal{Y} \cup \{\perp\})^{\mathcal{X}}$ s.t. for every data distribution $\mathcal{D}$ over $(\mathcal{X} \times \mathcal{Y})^m$,

$$R^{rej}_{\mathcal{U}}(\mathcal{A}(S); \mathcal{D}) \leqslant OPT^{rej}_{\mathcal{U}} + \epsilon$$

with probability at least $1 - \delta$ over $S \sim \mathcal{D}^m$. If no such m exists, $\mathcal{M}_{AG}(\epsilon, \delta; \mathcal{H}, \mathcal{U}) = \infty$. We say that $\mathcal{H}$ is robustly PAC learnable under rejection if $\mathcal{M}_{AG}(\epsilon, \delta; \mathcal{H}, \mathcal{U})$ is finite for all $\epsilon, \delta \in (0, 1)$.

Lemma B.5. *Let $\mathcal{M}_{RE} = \mathcal{M}_{RE}(1/3, 1/3; \mathcal{H}, \mathcal{U})$. Then,*

$$\mathcal{M}_{AG}(\epsilon, \delta; \mathcal{H}, \mathcal{U}) = \mathcal{O}\left(\frac{\mathcal{M}_{RE}}{\epsilon^2} \log^2\left(\frac{\mathcal{M}_{RE}}{\epsilon}\right) + \frac{1}{\epsilon^2} \log\left(\frac{1}{\delta}\right)\right) \tag{B.29}$$

Proof. The proof detail follows exactly the same from the Proof of Theorem 8 from (Montasser et al., 2019) with the loss replaced. □

Theorem B.6 (Sample Complexity for Agnostic Robust PAC Learning under Rejection)**.** *In the agnostic setting, for any $\mathcal{H} = \mathcal{H}_c \times \mathcal{H}_r$ and $\mathcal{U}$, and any $\epsilon, \delta \in (0, 1/2)$,*

$$\mathcal{M}_{AG}(\epsilon, \delta; \mathcal{H}, \mathcal{U}) = \mathcal{O}\Bigg(VC(T(\mathcal{H}))\, VC^*(T(\mathcal{H})) \log\left(VC(T(\mathcal{H}))\, VC^*(T(\mathcal{H}))\right) \tag{B.30}$$

$$\frac{1}{\varepsilon^2} \log^2\left(\frac{VC(T(\mathcal{H}))\, VC^*(T(\mathcal{H}))}{\varepsilon}\right) + \frac{1}{\varepsilon^2} \log\left(\frac{1}{\delta}\right)\Bigg) \tag{B.31}$$

$$= 2^{O(VC(\mathcal{H}))} \frac{1}{\varepsilon^2} \log^2\left(\frac{1}{\varepsilon}\right) + O\left(\frac{1}{\varepsilon^2} \log\left(\frac{1}{\delta}\right)\right) \tag{B.32}$$

$$= 2^{O((d_r + d_c) \log(d_r + d_c))} \frac{1}{\varepsilon^2} \log^2\left(\frac{1}{\varepsilon}\right) + O\left(\frac{1}{\varepsilon^2} \log\left(\frac{1}{\delta}\right)\right) \tag{B.33}$$

where $d_r = VC(\mathcal{H}_r), d_c = VC(\mathcal{H}_c)$.

Proof. Combining results from Lemma Theorem B.5 and Theorem B.2 gives the complexity result.

Solving Equation (B.32) gives the following generalization result given in Table 3.1

$$\Pr_{(\mathbf{x},\mathbf{y})\sim\mathcal{D}^n}\left[\mathrm{R}^{\mathrm{rej}}_{\mathcal{U}}(\mathcal{A}(\mathbf{x},\mathbf{y});\mathcal{D}) \leqslant \epsilon\right] \geqslant 1-\delta$$

where $\epsilon = \mathcal{O}\left(\sqrt{\frac{2^{\mathrm{VC}(\mathrm{T}(\mathcal{H}))}+\log(1/\delta)}{n}}\right)$. □

B.1.3 Transduction+Rejection: Realizable Case

We will prove a more general result which then implies Theorem 3.1. First, the training data can also be perturbed, i.e., the adversary perturbs $z \in \mathcal{U}(\mathbf{x})$ and $\tilde{z} \in \mathcal{U}(\tilde{\mathbf{x}})$, and the learner $\mathbb{A}$ are given $(z, \mathbf{y}, \tilde{z})$ instead of $(\mathbf{x}, \mathbf{y}, \tilde{z})$. The criterion in the transductive rejection error (see Table 3.2) is then the worst case over both $z \in \mathcal{U}(\mathbf{x})$ and $\tilde{z} \in \mathcal{U}(\tilde{\mathbf{x}})$. Second, we will consider $\mathrm{OPT}_{\mathcal{U}^3} = 0$ and prove the guarantee tolerating $\mathcal{U}^2$. This then implies the guarantee tolerating $\mathcal{U}$ when $\mathrm{OPT}_{\mathcal{U}^{3/2}} = 0$.

In general the set of optimally learned classifiers Δ is defined as follows Montasser et al. (2021):

$$\Delta^{\mathcal{U}}_{\mathcal{H}}(z,\mathbf{y},\tilde{z}) = \begin{cases} \{h \in \mathcal{H} : \mathrm{R}_{\mathcal{U}^{-1}}(h;z,\mathbf{y}) = 0 \wedge \mathrm{R}_{\mathcal{U}^{-1}}(h;\tilde{z}) = 0\} & \text{(Realizable Case)} \\ \arg\min_{h\in\mathcal{H}} \max\{\mathrm{R}_{\mathcal{U}^{-1}}(h;z,\mathbf{y}), \mathrm{R}_{\mathcal{U}^{-1}}(h;\tilde{z})\} & \text{(Agnostic Case)} \end{cases}$$

where

$$\mathrm{R}_{\mathcal{U}}(h;z,\mathbf{y}) = \sup_{\tilde{x}\in\mathcal{U}(z)} \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}\{h(\tilde{x}_i) \neq y_i\}$$

and

$$\mathrm{R}_{\mathcal{U}}(h;z) = \mathrm{R}_{\mathcal{U}}(h;z,h(z)).$$

Recall the transformation F which we define following Tramèr Tramèr (2022) in Section 3.4.

Then, we define the *relaxed robust shattering dimension* following Montasser et al. (2021):

Definition B.7 (Relaxed Robust Shattering Dimension)**.** *A sequence $z_1, \ldots, z_k \in \mathcal{X}$ is*

relaxed $\mathcal{U}$-robustly shattered by $\mathcal{H}$, if $\forall y_1, \ldots, y_k \in \{\pm 1\}$: $\exists x_1^{y_1}, \ldots, x_k^{y_k} \in \mathcal{X}$ *and* $\exists h \in \mathcal{H}$ *such that* $z_i \in \mathcal{U}(x_i^{y_i})$ *and* $h(\mathcal{U}(x_i^{y_i})) = y_i$, $\forall 1 \leqslant i \leqslant k$. *The relaxed $\mathcal{U}$-robust shattering dimension* $\mathrm{rdim}_{\mathcal{U}}(\mathcal{H})$ *is defined as the largest* k *for which there exist* k *points that are relaxed $\mathcal{U}$-robustly shattered by $\mathcal{H}$.*

Define the set of *intermediate perturbations* as follows:

Definition B.8 (Intermediate Perturbations)**.** *Given* x *and* z *and perturbations* $\mathcal{U}_1$ *and* $\mathcal{U}_2$*, the set of possible intermediate perturbations between* x *and* z *is*

$$\mathrm{ip}_{\mathcal{U}_1,\mathcal{U}_2}(x, z) = \begin{cases} \{x\} & \text{if } x = z \\ \mathcal{U}_1(x) \cap \mathcal{U}_2^{-1}(z) & \text{otherwise} \end{cases}$$

Theorem B.9. *For any* $n \in \mathbb{N}$, $\delta > 0$, *class* $\mathcal{H}$, *perturbation set* $\mathcal{U}$, *and distribution* $\mathcal{D}$ *over* $\mathcal{X} \times \mathcal{Y}$ *satisfying* $\mathrm{OPT}_{\mathcal{U}^{-1}\mathcal{U}} = 0$*:*

$$\Pr_{\substack{(\mathbf{x},\mathbf{y})\sim\mathcal{D}^n \\ (\tilde{\mathbf{x}},\tilde{\mathbf{y}})\sim\mathcal{D}^n}} \left[\begin{array}{l} \forall \mathbf{z} \in \mathcal{U}^3(\mathbf{x}), \forall \mathbf{z}_0 \in \mathrm{ip}_{\mathcal{U},\mathcal{U}^2}(\mathbf{x}, \mathbf{z}), \forall \tilde{\mathbf{z}} \in \mathcal{U}^3(\tilde{\mathbf{x}}), \forall \tilde{\mathbf{z}}_0 \in \mathrm{ip}_{\mathcal{U},\mathcal{U}^2}(\tilde{\mathbf{x}}, \tilde{\mathbf{z}}), \\ \forall \hat{h} \in \mathrm{F}_{\mathcal{U}} \left(\Delta^{\mathcal{U}}_{\mathcal{H}}(\mathbf{z}_0, \mathbf{y}, \tilde{\mathbf{z}}_0) \right) : \mathrm{err}^{\mathrm{rej}}(\hat{h}; \mathbf{x}, \mathbf{y}, \tilde{\mathbf{x}}, \tilde{\mathbf{z}}, \tilde{\mathbf{y}}) \leqslant \epsilon \end{array} \right] \geqslant 1 - \delta$$

where $\epsilon = \frac{\mathrm{rdim}_{\mathcal{U}^{-1}}(\mathcal{H}) \log(2n) + \log(1/\delta)}{n} \leqslant \frac{\mathrm{VC}(\mathcal{H}) \log(2n) + \log(1/\delta)}{n}$.

Proof. We adapt the strategy of Theorem 5 of Tramèr (2022) for the rejection scenario.

By setting $\mathbf{z} = \mathbf{z}_0, \tilde{\mathbf{z}} = \tilde{\mathbf{z}}_0$ and applying Theorem 1 of Montasser et al. (2021), we obtain the following

$$\Pr_{\substack{(\mathbf{x},\mathbf{y})\sim\mathcal{D}^n \\ (\tilde{\mathbf{x}},\tilde{\mathbf{y}})\sim\mathcal{D}^n}} \left[\forall \mathbf{z}_0 \in \mathcal{U}(\mathbf{x}), \forall \tilde{\mathbf{z}}_0 \in \mathcal{U}(\tilde{\mathbf{x}}), \forall h \in \Delta^{\mathcal{U}}_{\mathcal{H}}(\mathbf{z}_0, \mathbf{y}, \tilde{\mathbf{z}}_0) : \mathrm{err}_{\tilde{\mathbf{z}}_0,\tilde{\mathbf{y}}}(h) \leqslant \epsilon \right] \geqslant 1 - \delta \tag{B.34}$$

as $\mathrm{OPT}_{\mathcal{U}^{-1}(\mathcal{U})} = 0$.

Suppose $(\mathbf{x}, \mathbf{y}), (\tilde{\mathbf{x}}, \tilde{\mathbf{y}}) \sim \mathcal{D}^n$. Now, let $\mathbf{z} \in \mathcal{U}^3(\mathbf{x}), \tilde{\mathbf{z}} \in \mathcal{U}^3(\tilde{\mathbf{x}})$ and take some $\mathbf{z}_0 \in \mathrm{ip}_{\mathcal{U},\mathcal{U}^2}(\mathbf{x}, \mathbf{z}), \tilde{\mathbf{z}}_0 \in \mathrm{ip}_{\mathcal{U},\mathcal{U}^2}(\tilde{\mathbf{x}}, \tilde{\mathbf{z}})$, both of which are necessarily nonempty as $\mathcal{U}^3 = \mathcal{U}^2\mathcal{U}$, and $\hat{h} \in \mathrm{F}_{\mathcal{U}} \left(\Delta^{\mathcal{U}}_{\mathcal{H}}(\mathbf{z}_0, \mathbf{y}, \tilde{\mathbf{z}}_0) \right)$.

Write $\hat{h} = F_{\mathcal{U}}(h)$ for some $h \in \Delta^{\mathcal{U}}_{\mathcal{H}}(z_0, \mathbf{y}, \tilde{\mathbf{z}}_0)$.

From Equation (B.34) (replacing z with z_0 and $\tilde{z}$ with $\tilde{z}_0$), it is enough to show that

$$\mathrm{err}^{\mathrm{rej}}(\hat{h}; \mathbf{x}, \mathbf{y}, \tilde{\mathbf{x}}, \tilde{\mathbf{z}}, \tilde{\mathbf{y}}) \leqslant \mathrm{err}_{\tilde{z}_0, \tilde{\mathbf{y}}}(h).$$

Suppose that $\hat{h}$ incurs an error under rejection at point $\tilde{z}_i$; it is enough to show that h incurs an error at $\tilde{z}_{0_i}$. Furthermore, note that because $h \in \Delta^{\mathcal{U}}_{\mathcal{H}}(z_0, \mathbf{y}, \tilde{\mathbf{z}}_0)$, we have that $h(\mathcal{U}^{-1}(\tilde{z}_{0_i})) = \{h(\tilde{z}_{0_i})\}$ as $\tilde{z}_{0_i} \in \mathcal{U}^{-1}(\tilde{z}_{0_i})$. Write $h(\tilde{z}_{0_i}) = \hat{y}_i$.

We have one of the following:

1. $\hat{h}(\tilde{z}_i) \neq \tilde{y}_i$ and $\tilde{z}_i = \tilde{x}_i$
2. $\hat{h}(\tilde{z}_i) \notin \{\tilde{y}_i, \perp\}$ and $\tilde{z}_i \neq \tilde{x}_i$

In the first case, we must have $\tilde{z}_{0_i} = \tilde{x}_i$ as well as $\tilde{z}_{0_i}$ is an intermediate perturbation between $\tilde{x}_i$ and $\tilde{z}_i$, so, as $h(\mathcal{U}^{-1}(\tilde{z}_i)) = h(\mathcal{U}^{-1}(\tilde{z}_{0_i})) = \hat{y}_i$, $\hat{h}$ does not reject $\tilde{z}_{0_i}$ and $\hat{h}(\tilde{z}_{0_i}) = \hat{y}_i$. Hence, $h(\tilde{z}_{0_i}) = \hat{y}_i$ as well so, as $\hat{h}$ makes an error at $\tilde{z}_i$, $\hat{y}_i \neq y$ and so h makes an error at $\tilde{z}_{0_i}$.

In the second case, if $h(\mathcal{U}^{-1}(\tilde{z}_i)) \neq \{h(\tilde{z}_i)\}$, then $\hat{h}$ would reject $\tilde{z}_i$ and hence not incur an error. So $h(\mathcal{U}^{-1}(\tilde{z}_i)) = \{h(\tilde{z}_i)\}$ and so $\hat{h}(\tilde{z}_i) = h(\tilde{z}_i)$. Since $\tilde{z}_{0_i} \in \mathcal{U}(\tilde{x}_i) \cap \mathcal{U}^{-2}(\tilde{z}_i)$, there exists some $\tilde{z}'_{0_i} \in \mathcal{U}(\tilde{z}_{0_i}) \cap \mathcal{U}^{-1}(\tilde{z}_i)$ and so, $h(\tilde{z}_{0_i}) = h(\tilde{z}'_{0_i}) = h(\tilde{z}_i) = \hat{h}(\tilde{z}_i) = \hat{y}_i$, so h incurs an error at $\tilde{z}_{0_i}$.

In either case, we have that h makes an error at $\tilde{z}_{0_i}$, showing the result. □

Remark: More direct approaches may seem possible, but have surprising pitfalls. At first glance, this approach may seem less natural than simply applying the analysis of (Montasser et al., 2021) to a potential $\tilde{z}' \in \mathcal{U}^{1/2}(\tilde{x})$ with the condition of $\mathrm{OPT}_{\mathcal{U}}$, obtaining a $\mathcal{U}^{1/2}$-robust classifer h', and deriving an ϵ-robust selective classifier by the transformation $F_{\mathcal{U}^{1/2}}$. While this seems possible at first, as (Tramèr, 2022) shows that applying this transformation results in doubled robustness, this isn't possible in this situation, as h' is only guaranteed to be $\mathcal{U}^{1/2}$-robust at $\tilde{z}'$, not at *every* $\epsilon/2$ perturbation of $\tilde{x}$ as needed by the analysis. Similarly, it might seem possible to obtain an $\epsilon/2$-robust classifier at $\tilde{z}$ using (Montasser et al., 2021), and

derive the desired ϵ-robust classifier from $F_{\mathcal{U}^{1/2}}$; this, however, requires the condition $\mathrm{OPT}_{\mathcal{U}^2}$, as the analysis of (Montasser et al., 2021) only applies on perturbations up to half the margin; hence, this approach gains no advantage from rejection.

Sample Complexity Given ϵ and δ, we need

$$\frac{\mathrm{rdim}_{\mathcal{U}^{-1}}(\mathcal{H})\log(2n)+\log(1/\delta)}{n} \leqslant \epsilon$$

for the result to hold.

Now, noting that $\log(2n) = 1+\log n \leqslant 1+\sqrt{n}$ for $n \geqslant 16$; hence we need to solve for the n such that

$$\frac{\mathrm{rdim}_{\mathcal{U}^{-1}}(\mathcal{H})(1+\sqrt{n})+\log(1/\delta)}{n} = \epsilon$$

or, equivalently

$$\frac{\mathrm{rdim}_{\mathcal{U}^{-1}}(\mathcal{H})+\log(\frac{1}{\delta})+\sqrt{n}}{n} = \epsilon$$

or

$$\sqrt{n} = n\epsilon - \mathrm{rdim}_{\mathcal{U}^{-1}}(\mathcal{H}) - \log(\frac{1}{\delta})$$

or

$$n = n^2\epsilon^2 - 2\epsilon\left(\mathrm{rdim}_{\mathcal{U}^{-1}}(\mathcal{H})+\log(\frac{1}{\delta})\right)n + \left(\mathrm{rdim}_{\mathcal{U}^{-1}}(\mathcal{H})+\log(\frac{1}{\delta})\right)^2$$

or

$$n^2\epsilon^2 - \left(2\epsilon\left(\mathrm{rdim}_{\mathcal{U}^{-1}}(\mathcal{H})+\log(\frac{1}{\delta})\right)+1\right)n + \left(\mathrm{rdim}_{\mathcal{U}^{-1}}(\mathcal{H})+\log(\frac{1}{\delta})\right)^2 = 0.$$

Solving, the result holds if

$$n \geqslant \frac{2\epsilon\left(\mathrm{rdim}_{\mathcal{U}^{-1}}(\mathcal{H}) + \log(\frac{1}{\delta})\right) + 1 + \sqrt{(2\epsilon\left(\mathrm{rdim}_{\mathcal{U}^{-1}}(\mathcal{H}) + \log(\frac{1}{\delta})\right) + 1)^2 - 4\left(\mathrm{rdim}_{\mathcal{U}^{-1}}(\mathcal{H}) + \log(\frac{1}{\delta})\right)}}{2\epsilon^2}$$

$$= O\left(\frac{\mathrm{rdim}_{\mathcal{U}^{-1}}(\mathcal{H}) + \log(\frac{1}{\delta})}{\epsilon} + \frac{\sqrt{\mathrm{rdim}_{\mathcal{U}^{-1}}(\mathcal{H}) + \log(\frac{1}{\delta})}}{\epsilon^{\frac{3}{2}}}\right)$$

and, similarly, using

$$\frac{\mathrm{rdim}_{\mathcal{U}^{-1}}(\mathcal{H})\log(2n) + \log(1/\delta)}{n} \leqslant \frac{\mathrm{VC}(\mathcal{H})\log(2n) + \log(1/\delta)}{n}$$

we have the result if

$$n = O\left(\frac{\mathrm{VC}(\mathcal{H}) + \log(\frac{1}{\delta})}{\epsilon} + \frac{\sqrt{\mathrm{VC}(\mathcal{H}) + \log(\frac{1}{\delta})}}{\epsilon^{\frac{3}{2}}}\right)$$

Remark: If $\mathrm{OPT}_{\mathcal{U}^{-1}\mathcal{U}} = 0$, we can guarantee the existence of an $\hat{h}$ which satisfies our conditions, but we can't guarantee that we will find it, as we cannot find $\Delta^{\mathcal{U}}_{\mathcal{H}}(z_0, y, \tilde{z}_0)$ without z_0 and $\tilde{z}_0$. We can, however, construct that an algorithm which, if it returns a model, always returns on which meets the conditions.

Simplified Result To obtain a bound which does not involve an intermediate perturbation step, we may let

$$\Delta^{\mathcal{U}}_{\mathrm{rej},\mathcal{H}}(z, y, \tilde{z}) := \begin{cases} \hat{\Delta} \cup \Delta^{\mathcal{U}\prime}_{\mathrm{rej},\mathcal{H}}(z, y, \tilde{z}) & \hat{\Delta}^{\mathcal{U}}_{\mathcal{H}}(z, y, \tilde{z})(\tilde{z})| = 1, \text{ and} \\ l\Delta^{\mathcal{U}\prime}_{\mathrm{rej},\mathcal{H}}(z, y, \tilde{z}) & \text{otherwise} \end{cases}$$

where

$$\Delta^{\mathcal{U}\prime}_{\mathrm{rej},\mathcal{H}}(z, y, \tilde{z}) = \bigcap_{\tilde{z}' \in \mathcal{U}^{-2}(\tilde{z})} \Delta^{\mathcal{U}}_{\mathcal{H}}(z, y, \tilde{z}')$$

where

$$\hat{\Delta}^{\mathcal{U}}_{\mathcal{H}}(z, y, \tilde{z}) = \bigcup_{\tilde{z}' \in \mathcal{U}^{-2}(\tilde{z})} \Delta^{\mathcal{U}}_{\mathcal{H}}(z, y, \tilde{z}').$$

If $|\hat{\Delta}^{\mathcal{U}}_{\mathcal{H}}(z, y, \tilde{z})(\tilde{z})| = 1$, then as $\Delta^{\mathcal{U}}_{\mathcal{H}}(z_0, y, \tilde{z}_0)(\tilde{z}) \subseteq \hat{\Delta}^{\mathcal{U}}_{\mathcal{H}}(z, y, \tilde{z})(\tilde{z})$, $\hat{\Delta}^{\mathcal{U}}_{\mathcal{H}}(z, y, \tilde{z})(\tilde{z}) = \Delta^{\mathcal{U}}_{\mathcal{H}}(z_0, y, \tilde{z}_0)(\tilde{z})$ since $\Delta^{\mathcal{U}}_{\mathcal{H}}(z_0, y, \tilde{z}_0)$ is nonempty as $\mathrm{OPT}_{\mathcal{U}^{-1}(\mathcal{U})} = 0$.

Note that for common classes of perturbations, we can simplify the Δ'_{rej}. Note that the conditions of the theorem hold for perturbations defined via ϵ-balls in a metric.

Let

$$\Delta^{\mathcal{U},\mathcal{U}'}_{\mathcal{H}}(z, y, \tilde{z}) = \{h \in \mathcal{H} : \mathrm{R}_{\mathcal{U}^{-1}}(h; z, y) = 0 \wedge \mathrm{R}_{\mathcal{U}'^{-1}}(h; \tilde{z}) = 0\}.$$

Lemma B.10. *In the realizable case, if* $\mathcal{U} = \mathcal{U}^{-1}$,

$$\Delta^{\mathcal{U}\prime}_{\mathrm{rej},\mathcal{H}}(z, y, \tilde{z}) = \Delta^{\mathcal{U},\mathcal{U}^3}_{\mathcal{H}}(z, y, \tilde{z})$$

Proof. Suppose $h \in \Delta^{\mathcal{U}\prime}_{\mathrm{rej},\mathcal{H}}(z, y, \tilde{z})$. Then by the definitions of Δ_{rej} and Δ, $\mathrm{R}_{\mathcal{U}^{-1}}(h; z, y) = 0$ and for any $z' \in \mathcal{U}^{-2}(z)$, $\tilde{z}' \in \mathcal{U}^{-2}(\tilde{z})$, we have that, for any $x \in \mathcal{U}^{-1}(z')$ and $\tilde{x} \in \mathcal{U}^{-1}(\tilde{z}')$, $h(x_i) = h(z'_i)$ and $h(\tilde{x}_i) = h(\tilde{z}'_i)$. Now, as there exists some $z'' \in \mathcal{U}(z') \cap \mathcal{U}^{-1}(bz)$ and $h(x) = h(z') = h(z'') = h(z)$ by an argument similar to that in Theorem B.9 and similarly for $\tilde{x}$ and $\tilde{z}$, we have that for any $x \in \mathcal{U}^{-3}(z)$ and $\tilde{x} \in \mathcal{U}^{-3}(\tilde{z})$, $h(x_i) = h(z_i)$ and $h(\tilde{x}_i) = h(\tilde{z}_i)$, and so

$$\Delta^{\mathcal{U}\prime}_{\mathrm{rej},\mathcal{H}}(z, y, \tilde{z}) \subseteq \Delta^{\mathcal{U},\mathcal{U}^3}_{\mathcal{H}}(z, y, \tilde{z})$$

Now, if $h \in \Delta^{\mathcal{U},\mathcal{U}^3}_{\mathcal{H}}(z, y, \tilde{z})$, we have that, $\mathrm{R}_{\mathcal{U}^{-1}}(h; z, y) = 0$ and for any $x \in \mathcal{U}^{-3}(z)$ and $\tilde{x} \in \mathcal{U}^{-3}(\tilde{z})$, $h(x_i) = h(z_i)$ and $h(\tilde{x}_i) = h(\tilde{z}_i)$. Now, suppose $z' \in \mathcal{U}^{-2}(z)$, $\tilde{z}' \in \mathcal{U}^{-2}(\tilde{z})$. Since $x \in \mathcal{U}(x)$ for all x, $z' \in \mathcal{U}^{-3}(z)$, $\tilde{z}' \in \mathcal{U}^{-3}(\tilde{z})$ as well. Hence, $h(z'_i) = h(z_i)$ and $h(\tilde{z}'_i) = h(\tilde{z}_i)$. Now, if $x \in \mathcal{U}^{-1}(z')$ and $\tilde{x} \in \mathcal{U}^{-1}(\tilde{z}')$, we have $x \in \mathcal{U}^{-3}(z)$ and $\tilde{x} \in \mathcal{U}^{-3}(\tilde{z})$ and so $h(x_i) = h(z_i)$ and $h(\tilde{x}_i) = h(\tilde{z}_i)$. But then $h(x_i) = h(z'_i)$ and

$h(\tilde{x}_i) = h(\tilde{z}'_i)$. Hence, we have that

$$\Delta_{\mathcal{H}}^{\mathcal{U},\mathcal{U}^3}(\boldsymbol{z},\boldsymbol{y},\tilde{\boldsymbol{z}}) \subseteq \Delta_{\text{rej},\mathcal{H}}^{\mathcal{U}\prime}(\boldsymbol{z},\boldsymbol{y},\tilde{\boldsymbol{z}})$$

and the result follows. □

From this, we immediately derive the corollary

$$\Delta_{\text{rej},\mathcal{H}}^{\mathcal{U}}(\boldsymbol{z},\boldsymbol{y},\tilde{\boldsymbol{z}}) \supseteq \Delta_{\mathcal{H}}^{\mathcal{U}^3}(\boldsymbol{z},\boldsymbol{y},\tilde{\boldsymbol{z}}).$$

Remark: Note that this means that $\Delta_{\text{rej},\mathcal{H}}^{\mathcal{U}\prime}(\boldsymbol{z},\boldsymbol{y},\tilde{\boldsymbol{z}})$ is nonempty if $\text{OPT}_{\mathcal{U}^6} = 0$, and, by the definition of Δ, Δ is also nonempty if $|\hat{\Delta}_{\mathcal{H}}^{\mathcal{U}}(\boldsymbol{z},\boldsymbol{y},\tilde{\boldsymbol{z}})(\tilde{\boldsymbol{z}})| = 1$, i.e. if there exists only one possible labeling of the $\tilde{\boldsymbol{z}}$ which is robust at some possible intermediate perturbation.

Now, by the above and from Theorem B.9 we may immediately derive Theorem 3.1 by noting that if $\mathcal{U} = \mathcal{U}^{-1}$, $\mathcal{U}^{-1}\mathcal{U} = \mathcal{U}^2$, and if $\hat{h} \in F_{\mathcal{U}}(\Delta_{\mathcal{H}}^{\mathcal{U}}(\boldsymbol{z},\boldsymbol{y},\tilde{\boldsymbol{z}})) = F_{\mathcal{U}^{1/3}}(\Delta_{\text{rej},\mathcal{H}}^{\mathcal{U}^{1/3}}(\boldsymbol{z},\boldsymbol{y},\tilde{\boldsymbol{z}}))$ then we have $\hat{h} \in F_{\mathcal{U}^{1/3}}\left(\Delta_{\mathcal{H}}^{\mathcal{U}^{1/3}}(\boldsymbol{z}_0,\boldsymbol{y},\tilde{\boldsymbol{z}}_0)\right)$ for some $\boldsymbol{z}_0 \in \text{ip}_{\mathcal{U}^{1/3},\mathcal{U}^{2/3}}(\boldsymbol{x},\boldsymbol{z})$ and $\tilde{\boldsymbol{z}}_0 \in \text{ip}_{\mathcal{U}^{1/3},\mathcal{U}^{2/3}}(\tilde{\boldsymbol{x}},\tilde{\boldsymbol{z}})$.

Furthermore, following from Theorem B.10, $\Delta_{\text{rej},\mathcal{H}}^{\mathcal{U}^{1/3}}(\boldsymbol{z},\boldsymbol{y},\tilde{\boldsymbol{z}})$ is nonempty is $\text{OPT}_{\mathcal{U}^2} = 0$, showing completeness that the Δ of Theorem 3.1 is nonempty under that condition, as well as, as noted above, under the condition that there exists only one possible labeling consistent on a potential intermediate perturbation.

Now, we demonstrate that there exists a data distribution for which the transductive learner implied by Δ finds a solution for which the bound applies, but where no transductive learner has zero asymptotic robust error

Theorem B.11. *There exists a distribution $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$, a hypothesis class $\mathcal{H}$, and perturbation set $\mathcal{U}$ for which, with probability $\geqslant 1 - 2^{1-n}$, for any $(\boldsymbol{x},\boldsymbol{y}),(\tilde{\boldsymbol{x}},\tilde{\boldsymbol{y}}) \sim \mathcal{D}^n$ and any $\tilde{\boldsymbol{z}} \in \mathcal{U}^3(\tilde{\boldsymbol{z}})$, $\Delta_{\text{rej},\mathcal{H}}^{\mathcal{U}}(\boldsymbol{x},\boldsymbol{y},\tilde{\boldsymbol{z}})$ is nonempty and for all $h \in \Delta_{\text{rej},\mathcal{H}}^{\mathcal{U}}(\boldsymbol{z},\boldsymbol{y},\tilde{\boldsymbol{z}})$, $\text{err}_{\mathcal{U}}^{\text{rej}}(h;\boldsymbol{x},\boldsymbol{y},\tilde{\boldsymbol{x}},\tilde{\boldsymbol{z}},\tilde{\boldsymbol{y}}) = 0$ but, there exists no transductive learner (without rejection) $\mathbb{A}$ for which $\lim_{n\to\infty} \mathbb{E}\left[\sup_{\tilde{\boldsymbol{z}}\in\mathcal{U}(\tilde{\boldsymbol{x}})} \text{err}_{\mathcal{U}}(\mathbb{A}(\boldsymbol{x},\boldsymbol{y},\tilde{\boldsymbol{z}});\boldsymbol{x},\boldsymbol{y},\tilde{\boldsymbol{z}},\tilde{\boldsymbol{y}})\right] < 1/2$.*

Proof. Consider the simple discrete distribution $\mathcal{D}$ with $(x, y) \sim \mathcal{D}$ is $(1, 1)$ with probability $1/2$ and $(-1, 0)$ with probability $1/2$. Now, let $\mathcal{U}(x) = \{y \mid |y - x| < 1.5\}$ and let $\mathcal{H}$ be the class of

Now, let $\mathcal{H}$ be the class of threshold functions $h_w(x) = \mathbb{1}_{x \geqslant w}$ and $h_w^-(x) = \mathbb{1}_{x < w}$ for integer w.

First, note that with probability $1 - 2^{1-n}$ both $(-1, 0)$ and $(1, 1)$ appear in $\mathbf{x}$. In that case, any $h \in \Delta_{\mathcal{H}}^{\mathcal{U}}(\mathbf{x}, \mathbf{y}, \tilde{\mathbf{z}}')$ must be robust at -1 and 1 up to a radius of $1/2$; and hence h must be h_w for some $w \in [-1/2, 1/2]$ (and hence, $w = 0$). Hence, $|\hat{\Delta}| \leqslant 1$; note that for any possible perturbation of -1 or 1 is within $\mathcal{U}^2$ (i.e. within 1 unit of) either -1 or 1; hence, there always exists some $\tilde{\mathbf{z}}'$ where $\Delta_{\mathcal{H}}^{\mathcal{U}}(\mathbf{x}, \mathbf{y}, \tilde{\mathbf{z}}')$ is nonempty.

But then, there must exist exactly one element in $\hat{\Delta}$, and so Δ is nonempty. Consider $\tilde{\mathbf{z}}_i$. We have two cases:

If $\tilde{\mathbf{z}}_i \in [-1, -1/2] \cup [1/2, 1]$, then, as h is robustly correct with radius $1/2$ about 1 and -1, then $\tilde{\mathbf{x}}_i = \text{sign}(\tilde{\mathbf{z}}_i)$ and hence $h(\tilde{\mathbf{x}}_i) = \text{sign}(\tilde{\mathbf{z}}_i)$. If $\tilde{\mathbf{x}}_i = \tilde{\mathbf{z}}_i$ we do not reject as h is robust with radius $1/2$ about -1 and 1. Thus, we do not incur an error at $\tilde{\mathbf{z}}_i$.

If $\tilde{\mathbf{z}}_i \in (-1/2, 1/2)$, then $\tilde{\mathbf{z}}_i$ must be perturbed. But we have both positive and negative values within $1/2$ of $\tilde{\mathbf{z}}_i$, and so $F_{\mathcal{U}}(\tilde{\mathbf{z}}_i) = \perp$. Hence, we do not occur an error at $\tilde{\mathbf{z}}_i$.

In all cases, we do not incur an error if both $x = -1$ and $x = 1$ appear in the training data, and so $\text{err}_{\mathcal{U}}^{\text{rej}}(h; \mathbf{x}, \mathbf{y}, \tilde{\mathbf{x}}, \tilde{\mathbf{z}}, \tilde{\mathbf{y}})$ is 0 with probability $\geqslant 1 - 2^{1-n}$.

To see that there exists no transductive algorithm (without rejection) that can have asympotic error below $1/2$, note that any $\tilde{\mathbf{x}}$ can be perturbed to $\tilde{\mathbf{z}}$ where all $\tilde{\mathbf{z}}$ are 0; hence, samples from class 0 and class 1 are indistinguishable and the minimum error on $\tilde{\mathbf{z}}$ achievable by h is the minimum of the fraction of the $\tilde{\mathbf{x}}$ which are -1 and the fraction which are 1. As $n \to \infty$, these both tend to $1/2$ and the result follows. □

B.1.4 Transduction+Rejection: Agnostic Case

Note that, if $\mathcal{U}$ can be decomposed into a form $\mathcal{U} = (\mathcal{U}^{1/3})^3$ where $\mathcal{U}^{1/3} = \mathcal{U}^{-1/3}$ (as with standard perturbations in l_p), we obtain a bound which depends on

$\mathrm{OPT}_{\mathcal{U}^{2/3}}$ rather than $\mathrm{OPT}_{\mathcal{U}^2}$, enabling, for $\hat{h}$ satisfying the conditions, much stronger guarantees if $\mathrm{OPT}_{\mathcal{U}^{2/3}} << \mathrm{OPT}_{\mathcal{U}^2}$. Note that as $\forall x\, x \in \mathcal{U}(x)$, $\forall x\, \mathcal{U}^{2/3}(x) \subseteq \mathcal{U}^2(x)$, and so $\mathrm{OPT}_{\mathcal{U}^{2/3}} \leqslant \mathrm{OPT}_{\mathcal{U}^2}$.

Theorem B.12. *For any $n \in \mathbb{N}$, $\delta > 0$, class $\mathcal{H}$, perturbation set $\mathcal{U}$, and distribution $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$:*

$$\Pr_{\substack{(\mathbf{x},\mathbf{y})\sim\mathcal{D}^n \\ (\tilde{\mathbf{x}},\tilde{\mathbf{y}})\sim\mathcal{D}^n}} \left[\begin{array}{l} \forall \mathbf{z} \in \mathcal{U}^3(\mathbf{x}), \forall \mathbf{z}_0 \in \mathrm{ip}_{\mathcal{U},\mathcal{U}^2}(\mathbf{x},\mathbf{z}), \forall \tilde{\mathbf{z}} \in \mathcal{U}^3(\tilde{\mathbf{x}}), \forall \tilde{\mathbf{z}}_0 \in \mathrm{ip}_{\mathcal{U},\mathcal{U}^2}(\tilde{\mathbf{x}},\tilde{\mathbf{z}}), \\ \forall \hat{h} \in \mathrm{F}_{\mathcal{U}}\left(\Delta^{\mathcal{U}}_{\mathcal{H}}(\mathbf{z}_0,\mathbf{y},\tilde{\mathbf{z}}_0)\right) : \mathrm{err}^{\mathrm{rej}}(\hat{h};\mathbf{x},\mathbf{y},\tilde{\mathbf{x}},\tilde{\mathbf{z}},\tilde{\mathbf{y}}) \leqslant \epsilon \end{array} \right] \geqslant 1-\delta$$

where

$$\epsilon = \min\left\{ 2\,\mathrm{OPT}_{\mathcal{U}^{-1}\mathcal{U}} + \mathrm{O}\left(\sqrt{\frac{\mathrm{VC}(\mathcal{H}) + \log(1/\delta)}{n}}\right), 3\mathrm{OPT}_{\mathcal{U}^{-1}\mathcal{U}} + \mathrm{O}\left(\sqrt{\frac{\mathrm{rdim}\,\mathcal{U}(\mathcal{H})\ln(2n) + \ln(1/\delta)}{n}}\right)\right.$$

Proof. Suppose $(\mathbf{x},\mathbf{y}), (\tilde{\mathbf{x}},\tilde{\mathbf{y}}) \sim \mathcal{D}^n$. Now, let $\mathbf{z} \in \mathcal{U}^3(\mathbf{x}), \tilde{\mathbf{z}} \in \mathcal{U}^3(\tilde{\mathbf{x}})$ and take some $\mathbf{z}_0 \in \mathrm{ip}_{\mathcal{U},\mathcal{U}^2}(\mathbf{x},\mathbf{z}), \tilde{\mathbf{z}}_0 \in \mathrm{ip}_{\mathcal{U},\mathcal{U}^2}(\tilde{\mathbf{x}},\tilde{\mathbf{z}})$, both of which are necessarily nonempty, and $\hat{h} \in \mathrm{F}_{\mathcal{U}}\left(\Delta^{\mathcal{U}}_{\mathcal{H}}(\mathbf{z}_0,\mathbf{y},\tilde{\mathbf{z}}_0)\right)$.

Write $\hat{h} = \mathrm{F}_{\mathcal{U}}(h)$ for some $h \in \Delta^{\mathcal{U}}_{\mathcal{H}}(\mathbf{z}_0,\mathbf{y},\tilde{\mathbf{z}}_0)$.

We will begin as in Theorem B.9. As before, there are two cases in which $\hat{h}$ can incur an error at $\tilde{z}_i$:

1. $\hat{h}(\tilde{z}_i) \neq \tilde{y}_i$ and $\tilde{z}_i = \tilde{x}_i$
2. $\hat{h}(\tilde{z}_i) \notin \{\tilde{y}_i, \perp\}$ and $\tilde{z}_i \neq \tilde{x}_i$

Now, if $\tilde{z}_i = \tilde{x}_i$, an error occurs if $\hat{h}$ rejects $\tilde{z}_i$ or if h robustly predicts some $\hat{y}_i \neq \tilde{y}_i$; hence an error occurs if h is not $\mathcal{U}^{-1}$-robust at $\tilde{z}_{0_i}$ or if $h(\tilde{z}_{0_i}) \neq \tilde{y}_i$.

Otherwise, h must be $\mathcal{U}^{-1}$-robust at $\tilde{z}_i$, as, otherwise, $\hat{h}$ would reject $\tilde{z}_i$. Hence, as there exists some $\tilde{z}'_{0_i} \in \mathcal{U}(\tilde{z}_{0_i}) \cap \mathcal{U}^{-1}(\tilde{z}_i)$, if h is $\mathcal{U}$-robust at $\tilde{z}_{0_i}$, we must have $h(\tilde{z}_i) = h(\tilde{z}_{0_i})$, and so, if $\hat{h}$ makes an error, h is not $\mathcal{U}^{-1}$-robust at $\tilde{z}_{0_i}$ or $h(\tilde{z}_{0_i}) \neq \tilde{y}_i$.

Now, in both cases, errors only occur if h is not $\mathcal{U}^{-1}$-robust at $\tilde{z}_{0_i}$ or $h(\tilde{z}_{0_i}) \neq \tilde{y}_i$. As $\tilde{x}_i \in \mathcal{U}^{-1}(\tilde{z}_{0_i})$, we have, equivalently, that an error occurs if h is not $\mathcal{U}^{-1}$-robust at $\tilde{z}_{0_i}$ or $h(\tilde{x}_i) \neq \tilde{y}_i$.

Hence,

$$\operatorname{err}^{\text{rej}}(\hat{h}; \mathbf{x}, \mathbf{y}, \tilde{\mathbf{x}}, \tilde{\mathbf{z}}, \tilde{\mathbf{y}}) \leqslant \operatorname{err}^{\text{rej}}(h; \tilde{x}, \tilde{y}) + R_{\mathcal{U}^{-1}}(h; \tilde{\mathbf{z}}_0)$$

Now, the right hand is exactly what is bounded in Theorem 2 of Montasser et al. (2021); as we have $h \in \Delta^{\mathcal{U}}_{\mathcal{H}}(\mathbf{z}_0, \mathbf{y}, \tilde{\mathbf{z}}_0)$, we have

$$\operatorname{err}^{\text{rej}}(\hat{h}; \mathbf{x}, \mathbf{y}, \tilde{\mathbf{x}}, \tilde{\mathbf{z}}, \tilde{\mathbf{y}}) \leqslant \operatorname{err}^{\text{rej}}(h; \tilde{x}, \tilde{y}) + R_{\mathcal{U}^{-1}}(h; \tilde{\mathbf{z}}_0) \leqslant \epsilon$$

where

$$\epsilon = \min \left\{ 2\operatorname{OPT}_{\mathcal{U}^{-1}\mathcal{U}} + O\left(\sqrt{\frac{\operatorname{VC}(\mathcal{H}) + \log(1/\delta)}{n}}\right), 3\operatorname{OPT}_{\mathcal{U}^{-1}\mathcal{U}} + O\left(\sqrt{\frac{\operatorname{rdim} \mathcal{U}(\mathcal{H}) \ln(2n) + \ln(1/\delta)}{n}}\right)\right.$$

with probability $\geqslant 1 - \delta$ by its proof. □

As in the realizable case, we can immediately derive the following corollary. However, we cannot simplify the definition of Δ_{rej} as before; see Lemma B.14.

Corollary B.13. *For any $n \in \mathbb{N}$, $\delta > 0$, class $\mathcal{H}$, perturbation set $\mathcal{U}$ where $\mathcal{U} = \mathcal{U}^{-1}$, and distribution $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$:*

$$\Pr_{\substack{(\mathbf{x},\mathbf{y})\sim\mathcal{D}^n \\ (\tilde{\mathbf{x}},\tilde{\mathbf{y}})\sim\mathcal{D}^n}} \left[\begin{array}{l} \forall \mathbf{z} \in \mathcal{U}^3(\mathbf{x}), \forall \tilde{\mathbf{z}} \in \mathcal{U}^3(\tilde{\mathbf{x}}), \forall \hat{h} \in F_{\mathcal{U}}\left(\Delta^{\mathcal{U}}_{\text{rej},\mathcal{H}}(\mathbf{z}, \mathbf{y}, \tilde{\mathbf{z}})\right) : \\ \operatorname{err}^{\text{rej}}(\hat{h}; \mathbf{x}, \mathbf{y}, \tilde{\mathbf{x}}, \tilde{\mathbf{z}}, \tilde{\mathbf{y}}) \leqslant \epsilon \end{array} \right] \geqslant 1 - \delta$$

where

$$\epsilon = \min \left\{ 2\operatorname{OPT}_{\mathcal{U}^{-1}\mathcal{U}} + O\left(\sqrt{\frac{\operatorname{VC}(\mathcal{H}) + \log(1/\delta)}{n}}\right), 3\operatorname{OPT}_{\mathcal{U}^{-1}\mathcal{U}} + O\left(\sqrt{\frac{\operatorname{rdim} \mathcal{U}(\mathcal{H}) \ln(2n) + \ln(1/\delta)}{n}}\right)\right.$$

Lemma B.14. *In the agnostic case, we have that if* $\mathcal{U} = \mathcal{U}^{-1}$,

$$\Delta^{\mathcal{U}}_{\text{rej},\mathcal{H}}(\mathbf{z}, \mathbf{y}, \tilde{\mathbf{z}}) \subseteq \Delta^{\mathcal{U}^3}_{\mathcal{H}}(\mathbf{z}, \mathbf{y}, \tilde{\mathbf{z}})$$

Proof. By the definition of R, we have

$$\begin{aligned}
R_{\mathcal{U}^{-3}}(h; \tilde{\mathbf{z}}) &= \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}\left\{\exists \tilde{x}_i \in \mathcal{U}^{-3}(\tilde{z}_i) : h(\tilde{x}_i) \neq h(\tilde{z}_i)\right\} \\
&= \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}\left\{ \exists \tilde{z}'_i \in \mathcal{U}^{-2}(\tilde{z}_i)\, \exists \tilde{x}_i \in \mathcal{U}^{-1}(\tilde{z}'_i) : h(\tilde{x}_i) \neq h(\tilde{z}_i) \right\} \\
&= \max_{\tilde{z}'_i \in \mathcal{U}^{-2}(\tilde{z}_i)} \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}\left\{\exists \tilde{x}_i \in \mathcal{U}^{-1}(\tilde{z}'_i) : h(\tilde{x}_i) \neq h(\tilde{z}_i)\right\} \\
&= \max_{\tilde{z}'_i \in \mathcal{U}^{-2}(\tilde{z}_i)} R_{\mathcal{U}^{-1}}(h; \tilde{\mathbf{z}}')
\end{aligned}$$

where the last equality holds as $x \in \mathcal{U}(x)$ for all x and as $\mathcal{U} = \mathcal{U}^{-1}$, which together show that if for some $\tilde{z}_i$ and $\tilde{z}'_i \in \mathcal{U}^{-2}(\tilde{z}_i)$ we have that $h(\tilde{z}'_i) \neq h(\tilde{z}_i)$, that either there exists some $\tilde{z}''_i \in \mathcal{U} = \mathcal{U}^{-1}(\tilde{z}'_i)$ such that $h(\tilde{z}''_i) \neq h(\tilde{z}'_i)$ or there exists some $\tilde{z}''_i \in \mathcal{U} = \mathcal{U}^{-1}(\tilde{z}_i)$ such that $h(\tilde{z}''_i) \neq h(\tilde{z}_i)$ (as before, note that $\tilde{z}_i = \mathcal{U}(\tilde{z}''_i)$ for some $\tilde{z}''_i \in \mathcal{U}(\tilde{z}'_i)$ by the definition of $\mathcal{U}^3$); the reverse is similar.

We can derive a result for $R_{\mathcal{U}^{-3}}(h; \mathbf{z}, \mathbf{y})$ similarly.

Suppose $h \in \Delta^{\mathcal{U}}_{\text{rej},\mathcal{H}}(\mathbf{z}, \mathbf{y}, \tilde{\mathbf{z}})$. Then, h minimizes $\max\{R_{\mathcal{U}^{-1}}(h; \mathbf{z}', \mathbf{y}), R_{\mathcal{U}^{-1}}(h; \tilde{\mathbf{z}}')\}$ for all $\mathbf{z}' \in \mathcal{U}^{-2}(\mathbf{z})$, $\tilde{\mathbf{z}}' \in \mathcal{U}^{-2}(\tilde{\mathbf{z}})$, so by the above, h must also minimize

$$\begin{aligned}
&\max_{\mathbf{z}' \in \mathcal{U}^{-2}(\mathbf{z}), \tilde{\mathbf{z}}' \in \mathcal{U}^{-2}(\tilde{\mathbf{z}})} \max\{R_{\mathcal{U}^{-1}}(h; \mathbf{z}', \mathbf{y}), R_{\mathcal{U}^{-1}}(h; \tilde{\mathbf{z}}')\} \\
&= \max\left\{ \max_{\mathbf{z}' \in \mathcal{U}^{-2}(\mathbf{z})} R_{\mathcal{U}^{-1}}(h; \mathbf{z}', \mathbf{y}), \max_{\tilde{\mathbf{z}}' \in \mathcal{U}^{-2}(\tilde{\mathbf{z}})} R_{\mathcal{U}^{-1}}(h; \tilde{\mathbf{z}}') \right\} \\
&= \max\{R_{\mathcal{U}^{-3}}(h; \tilde{\mathbf{z}}), R_{\mathcal{U}^{-3}}(h; \mathbf{z}, \mathbf{y})\}
\end{aligned}$$

and so $h \in \Delta^{\mathcal{U}^3}_{\mathcal{H}}(\mathbf{z}, \mathbf{y}, \tilde{\mathbf{z}})$.

However, minimizing

$$\max_{z' \in \mathcal{U}^{-2}(z), \tilde{z}' \in \mathcal{U}^{-2}(\tilde{z})} \max\{R_{\mathcal{U}^{-1}}(h; z', y), R_{\mathcal{U}^{-1}}(h; \tilde{z}')\}$$

does not necessarily imply that h minimizes $\max\{R_{\mathcal{U}^{-1}}(h; z', y), R_{\mathcal{U}^{-1}}(h; \tilde{z}')\}$ for all $z' \in \mathcal{U}^{-2}(z), \tilde{z}' \in \mathcal{U}^{-2}(\tilde{z})$, so the reverse may not hold. □

B.1.5 Extension to Unbalanced Training and Test Data

We provide a sketch of a proof that allows extending Theorem 1 of (Montasser et al., 2021) to unbalanced training and test sets; however, for simplicity, we will work with the original form. The assumptions are the same, except that we have n training points and m test points.

The proof is exactly as before up to the "Finite robust labelings" portion (which points are and are not labelled don't matter up to then and the symmetry arguments still apply). The basic idea of determining the probability of zero loss on the training and test sets and error $> \epsilon$ on the test examples with permutation still applies. Let $E_{\sigma,x}$ be the event that there exists a labelling $\hat{h}(x_{\sigma(1:n+m)})$ in the allowable set where this occurs.

We have

$$\Pr_{\sigma}[E_{\sigma,x}] \leqslant \Pr_{\sigma}\left[\exists \hat{h} \in \Pi^{\mathcal{U}}_{\mathcal{H}}(x_1, \ldots, x_{n+m}) : \operatorname*{err}_{x_{\sigma(1:n)}, y_{\sigma(1:n)}}(\hat{h}) = 0 \wedge \operatorname*{err}_{x_{\sigma(n:n+m)}, y_{\sigma(n:n+m)}}(\hat{h}) > \epsilon\right]$$

and, as in (Montasser et al., 2021), note the probability of choosing such a perturbation σ for a fixed $\hat{h}$ is at most

$$\left(\frac{m}{n+m}\right)^{s} \leqslant \left(\frac{m}{n+m}\right)^{\lceil \epsilon m \rceil} = \left(\frac{n+m}{m}\right)^{-\lceil \epsilon m \rceil} \leqslant \left(\frac{n+m}{m}\right)^{\lceil -\epsilon m \rceil}$$

if we assume the number of total errors $s \geqslant \lceil \epsilon m \rceil$ without loss of generality (otherwise, err $> \epsilon$ would be impossible).

Hence, by a union bound,

$$\Pr_{\sigma}[E_{\sigma,x}] \leqslant \left|\Pi^{\mathcal{U}}_{\mathcal{H}}(x_1,\ldots,x_{n+m})\right| \left(\frac{n+m}{m}\right)^{\lceil -\epsilon m\rceil}$$

and so

$$\Pr_{\sigma}[E_{\sigma,x}] \leqslant (n+m)^{\mathrm{rdim}_{\mathcal{U}^{-1}}(\mathcal{H})} \left(\frac{n+m}{m}\right)^{\lceil -\epsilon m\rceil}$$

by Sauer's Lemma (in the form of Lemma 3 of (Montasser et al., 2021)).

Now, we bound the probability by δ, we need

$$(n+m)^{\mathrm{rdim}_{\mathcal{U}^{-1}}(\mathcal{H})} \left(\frac{n+m}{m}\right)^{\lceil -\epsilon m\rceil} \leqslant \delta$$

which, solving, gives us

$$\epsilon \geqslant \frac{\mathrm{rdim}_{\mathcal{U}^{-1}}(\mathcal{H}) \log_{\frac{n+m}{m}}(n+m) + \log_{\frac{n+m}{m}} \frac{1}{\delta}}{m} = \frac{\mathrm{rdim}_{\mathcal{U}^{-1}}(\mathcal{H}) \log(n+m) + \log \frac{1}{\delta}}{m \log\left(1+\frac{m}{n}\right)}$$

Which reduces to the original result if $n = m$ (note that the logarithms are base-2).

Corollary If we fix $n+m$, $\mathcal{H}$, and δ, the guarantee is strongest (i.e. we minimize ϵ) when $n = m$. To see this, consider the denominator. Write $\alpha = \frac{m}{n}$. Then, we wish to maximize $n\alpha\log(1+\alpha)$ (or equivalently $f(\alpha) = \alpha\log(1+\alpha)$ subject to $\alpha \geqslant 0$. Now, note that $f'(\alpha) = \log(1+\alpha) - 1 = 0$ when $\alpha = 1$, i.e. when $m = n$.

Also, we can see from the result above, that if we fix m and δ, then the minimum value of ϵ tends towards ∞ as $n \to \infty$, so there does not necessarily exist a labelled training set sampled from $\mathcal{D}$ which provides a guarantee with high probability of arbitrarily low error on a fixed test set.

B.2 Experimental Details

B.2.1 Computing Infrastructure

We used a SLURM cluster with A100 GPUs to run our experiments.

B.2.2 Baseline Details

The baselines are trained with standard adversarial training (Goodfellow et al., 2015b) (Madry et al., 2018a). Attacks against AT without rejection use standard PGD with a cross-entropy objective, while attacks against AT with rejection use PGD targeting $\mathcal{L}_{\text{REJ}}$ as described in algorithm 5. In all cases, the parameters for PGD in training are the same as those used in TLDR's training process for the same dataset.

B.2.3 Defense

In our implementation, we begin to incorporate the transductive term in our objective (see Equation (3.6)) after initially training the model with the inductive loss term only; this allows learning a better baseline before we begin to enforce robustness about the test points. In our experiments, we use the transductive loss in the final half of the training epochs.

B.2.4 Adaptive Attack

Solving for the perturbation $\tilde{x}$ by iteratively optimizing $\mathcal{L}_{\text{REJ}}$ poses several difficulties.

First, the rejection-avoidance term $\left\|\tilde{x} - \arg\max_{||x'-\tilde{x}||\leqslant\epsilon} \mathcal{L}_{\text{DB},h}(x')\right\|$ is not differentiable with respect to $\tilde{x}$. While it is possible to approximate the derivative with the derivative of a proxy (e.g. differentiating though some fixed number of PGD steps, necessitating second-order optimization), this is extremely expensive and does not improve results in our experiments (see below).

Intuitively, we might see that this would be the case: if the decision boundary is smooth, we might expect the maximizers in $\mathcal{U}(x + \Delta)$ and $\mathcal{U}(x)$ to be the same for small Δ unless x' is near the border of $\mathcal{U}(x)$ given that $\mathcal{U}(x + \Delta) \approx \mathcal{U}(x)$. In this case, approximating x' as constant with respect to x is reasonable.

In addition, note that if $h(x) = y$, the adversary must find a $\tilde{x}$ where $h(\tilde{x}) \neq y$ which is not rejected: if maximizing $\mathcal{L}_{\text{REJ}}$ with PGD, the rejection-avoidance term penalizes moving $\tilde{x}$ towards the decision boundary. As this is necessary to find a valid attack (when $h(\tilde{x}) = y$ at initialization), we adjust λ adaptively during optimization by setting it to zero when $h(\tilde{x}) = y$.

B.2.5 Transductive Attack Details

We present two rejection-aware transductive attacks: a stronger but more computationally intensive rejection-aware GMSA (Algorithm 3) and a weaker but faster rejection-aware transfer attack which takes the transductive robust rejection risk into account (Algorithm 4).

Finally, note the attack with $\mathcal{L}_{\text{REJ}}$, without GMSA, is effective against selective classifiers based on the transformation F (and via Tramèr's equivalency, selective classifiers in general). So we summarize this attack on a fixed model in Algorithm 5.

B.2.6 Rejectron Experiments

Goldwasser et al.'s implementation of Rejectron (Goldwasser et al., 2020a) trains a classifier (call it h_c) on the training set and a discriminator (h_d) to distinguish between the (clean) training and (potentially-perturbed) test data. Samples are rejected if the discriminator classifies them as test data; otherwise, the classifier's prediction is returned. Our adaptive attack is then very simple: we follow the approach of Algorithm 3 but with a loss function $\mathcal{L}_{\text{DISC}}$ which targets the defense.

Given a sample (x, y), the attacker's goal is to flip the label, and, simultaneously, to avoid rejection; hence, we maximize the following loss:

$$\mathcal{L}_{\text{DISC}}(x, y) = \mathcal{L}_{\text{CE}}(h_c^s(x), y) + \lambda \mathcal{L}_{\text{CE}}(h_d^s(x), 1)$$

where class 1 for h_d corresponds to test data, signalling rejection, and where h^s returns the softmax activations of h. Maximizing $\mathcal{L}_{\text{DISC}}$ then minimizes the confidence in the true label and the probability of rejection.

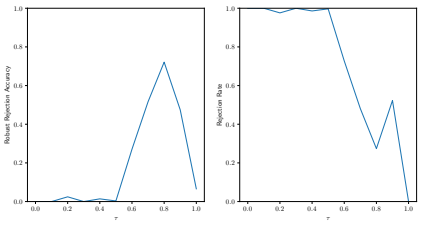


Figure B.1: Effects of τ on performance of Rejectron on MNIST with attacker GMSA ($\mathcal{L}_{\text{DISC}}$).

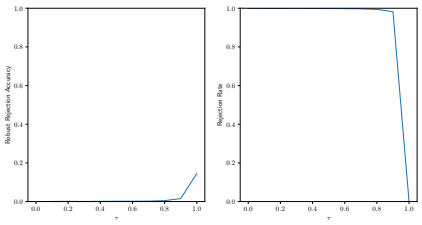


Figure B.2: Effects of τ on performance of Rejectron on CIFAR-10 with attacker GMSA ($\mathcal{L}_{\text{DISC}}$).

Figures B.1 and B.2 show our adaptive attack's performance on MNIST and CIFAR-10. τ is a key hyperparameter of Rejectron, which determines the confidence needed by h_d to reject a sample; to evaluate Rejectron fairly, we report the results on best-performing value of τ, based on (transductive) robust rejection accuracy; see Table 3.5. On CIFAR-10, performance is near-zero and rejection rate is near 100% for small values of τ. The best-performing value of τ is 1 (effectively eliminating the possibility of rejection), leading to a rejection rate of 0; this behavior on CIFAR-10 illustrates the algorithm's struggles with the practical high-complexity deep learning setting.

B.3 Additional Experiments

B.3.1 Ablation Study of TLDR

Compared to traditional defenses, TLDR has two novel components: using the given test inputs in training the classifier (the second term in Equation (3.6), referred to as L_{test}), and transforming the trained classifier into one with rejection. Table B.1 shows

TLDR Components		Attacker	MNIST		CIFAR-10	
Rejection	L_{test}		p_{REJ}	Robust accuracy	p_{REJ}	Robust accuracy
✓	✓	GMSA ($\mathcal{L}_{REJ}$)	**0.588**	0.967	0.208	**0.739**
✓	×	GMSA ($\mathcal{L}_{REJ}$)	0.646	**0.975**	**0.179**	0.725
×	✓	GMSA ($\mathcal{L}_{CE}$)	–	0.900	–	0.516
×	×	GMSA ($\mathcal{L}_{CE}$)	–	0.935	–	0.516

Table B.1: Ablation study of TLDR. The best result is **boldfaced**.

Table B.2: Effects of warm start period on TLDR.

Warm start (epochs)	Rejection Rate	Robust Rejection Accuracy
0	0.813	0.153
500	**0.531**	**0.177**
1000	0.830	0.171

the results of the ablation study on these two components. In all cases, rejection significantly improves results. The use of transduction is helpful on CIFAR-10, but reduces performance on MNIST, potentially due to the lower difficulty of deriving robust predictions on MNIST; hence, the knowledge of the specific test inputs is less useful.

B.3.2 Warm Start in TLDR

Here we perform experiments showing that in training TLDR, it is best to first trains a baseline model without transductive regularization L_{test} in the early stage (warm start) and then add transductive regularization for later training.

We generate the data with 100 Gaussians (one per class) equally spaced in l_∞ with a separation of 3 units between means. The adversarial budget is 2 units, and we ensure that the data is sparse by generating 10 samples per class. The models are 10 layer feedforward networks with skip connections.

The synthetic models are trained for 1000 epochs total; we see the best performance when the model has transductive regularization but is allowed to learn an initial baseline model before transductive regularization is used in training. Doing so reduces the risk of the regularization term harming performance.

B.3.3 GMSA Method

Table B.3: Full ablation results of TLDR.

TLDR Components		Attacker	MNIST		CIFAR-10	
Rejection	Transductive Regularization		p_{REJ}	Robust accuracy	p_{REJ}	Robust accuracy
✓	✓	$GMSA_{AVG}$ ($\mathcal{L}_{REJ}$)	0.796	0.968	0.195	0.744
✓	✓	$GMSA_{MIN}$ ($\mathcal{L}_{REJ}$)	0.588	0.967	0.208	0.739
✓	×	$GMSA_{AVG}$ ($\mathcal{L}_{REJ}$)	0.646	0.975	0.179	0.725
✓	×	$GMSA_{MIN}$ ($\mathcal{L}_{REJ}$)	0.202	0.980	0.182	0.733
×	✓	$GMSA_{AVG}$ ($\mathcal{L}_{CE}$)	–	0.900	–	0.516
×	✓	$GMSA_{MIN}$ ($\mathcal{L}_{CE}$)	–	0.914	–	0.601
×	×	$GMSA_{AVG}$ ($\mathcal{L}_{CE}$)	–	0.935	–	0.516
×	×	$GMSA_{MIN}$ ($\mathcal{L}_{CE}$)	–	0.942	–	0.556

We present extended results of our defense ablation and compare the results of $GMSA_{AVG}$, which optimizes the average loss of past iterations, and $GMSA_{MIN}$, which optimizes the worst-case loss. See (Chen et al., 2022). We can see that while the two perform about the same on the full TLDR defense ($GMSA_{MIN}$ performs slightly better), $GMSA_{AVG}$ is much stronger for models not incorporating both components.

B.3.4 Rejection Radius

The rejection radius $\epsilon_{defense}$ is an important hyper-parameter for TLDR; however, the model's performance is not very sensitive to it. Figure B.3 shows the trend of robust accuracy, the rejection rate on adversarial test data, and the rejection rate on clean test data, for the inductive classifier on MNIST; Figure B.4 shows those for TLDR. The robust accuracy remains stable. The theoretical analysis suggests setting the radius to $\epsilon/3$ where ϵ is the adversarial budget. Given TLDR's low sensitivity to the parameter, we use $\epsilon/4$ for consistency as the inductive case performs best

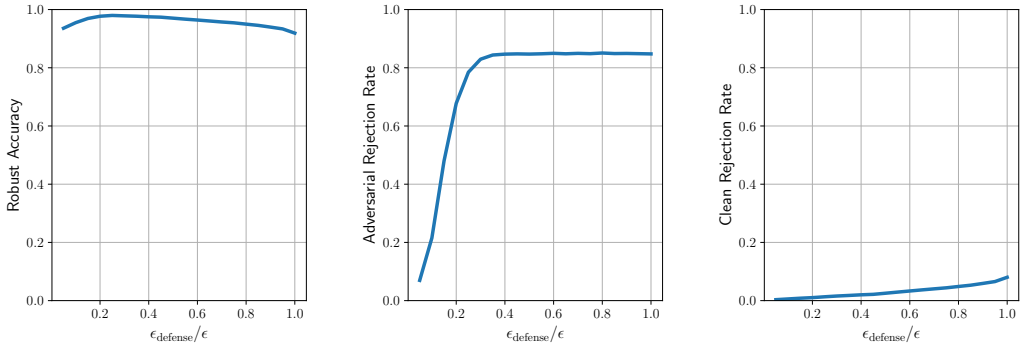


Figure B.3: Effects of rejection radius $\epsilon_{\text{defense}}$ on MNIST (inductive) with attacker PGD ($\mathcal{L}_{\text{REJ}}$).

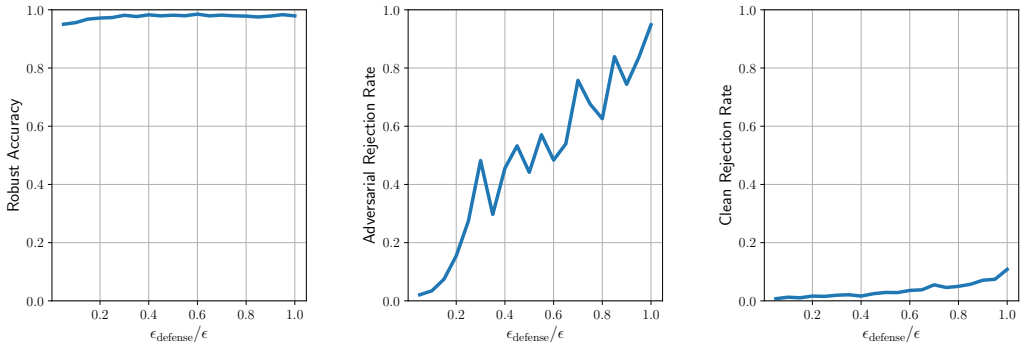


Figure B.4: Effects of rejection radius $\epsilon_{\text{defense}}$ on MNIST (TLDR) with attacker GMSA ($\mathcal{L}_{\text{REJ}}$).

with that setting. The rejection rate on the adversarial test data rises rapidly with the rejection radius (reaching 0.949 for TLDR for $\epsilon_{\text{defense}} = \epsilon$), but the rejection rate on clean data increases much more slowly (0.108 when $\epsilon_{\text{defense}} = \epsilon$). So among all rejected inputs only a few are clean inputs, leading to low errors as desired.

The rejection rate on clean inputs is presented for the transductive case in order to illustrate the difference in effects on clean and perturbed data, but, as the adversary may select to perturb, some clean points were not in the training set, and, hence, the clean rejection rates should not be considered reliable. The rejection rates rise with the rejection radius: adversarial rejection rates increase rapidly as the rejection radius increases, while clean rejection rates increase only slowly. In all cases, far

more perturbed samples are rejected than clean samples.

B.3.5 Binarization test on PGD ($\mathcal{L}_{\text{REJ}}$)

Table B.4: Results of the binarization test applied to PGD ($\mathcal{L}_{\text{REJ}}$).

	MNIST				CIFAR-10			
Decision Boundary Closeness	ASR	RASR	Inverted ASR	Inverted RASR	ASR	RASR	Inverted ASR	Inverted RASR
0.9	0.935	0.451	1.0	0.375	0.973	0.824	0.971	0.781
0.999	0.945	0.394	1.0	0.447	0.976	0.813	0.964	0.790
0.99999	0.953	0.414	0.981	0.434	0.974	0.819	0.938	0.813

Finally, to evaluate the effectiveness with which $\mathcal{L}_{\text{REJ}}$ targets rejection, we apply the binarization test (Zimmermann et al., 2022). As the binarization test is designed for inductive defenses we evaluate on PGD ($\mathcal{L}_{\text{REJ}}$), and as the binarization test assumes that rejection does not depend on the generated dataset or the modified model, we modified $\mathcal{L}_{\text{REJ}}$ to target the original model in the calcuation of $\mathcal{L}_{\text{DB},h}$ (e.g. we wish to avoid rejection as if the model was unchanged).

For the inverted case, we modified λ', setting it to -10 (we are seeking rejection, not avoiding it). As noted in Appendix B.3 , we drop the rejection-avoidance term when $h(\tilde{x}) = y$; hence, the negated second term poses issues for maximization in PGD (e.g. PGD would preferentially select perturbations which do not succeed). To avoid this issue, we have added an additional success indicator to our attack objective, which we use to ensure that PGD selects the loss-maximizing successful perturbation. Without these modifications, we observed low attack success rates in the inverted test; however, the results with these simple changes do indicate that our attack does take the rejection component of the defense into account, the key purpose of the inverted test.

The attack settings for the regular test are unchanged from those used for evaluation. For the test settings, we chose values as close as possible to those used in (Zimmermann et al., 2022), with a single boundary sample, with 200 samples sampled from each of the surfaces and corners of the l_∞ ball, with 512 trials per

experiment. We used 81 inner samples for MNIST, and 253 inner samples for CIFAR-10, selected to maximize subject to the requirement that the total sample count is below the dimensionality of the features. In both cases, the base model is a standard adversarially trained model trained on that dataset, transformed into a selective classifier with the transformation F.

We ran the test for a range of values for the decision boundary closeness, a hyperparameter determining the test hardness. ASR is the rate at which the attack successfuly found a perturbation which both flips the label and evades detection; RASR is the maximum of the success rates on surfaces and corners. While the ASR values in some experiments are slightly below the cutoff of 0.95 and are technically failures, they do indicate that the attack is successfully targeting the defense. While a slightly stronger attack may exist, these results do not indicate significant unreliability in our evaluation of the robustness of TLDR.

B.3.6 Ablation on Attacks: Attack Radius

The theory suggests that incorporating rejection can allow a transductive learner to tolerate perturbations twice as large; we investigate how transduction and rejection affects the robustness as ϵ grows (models are adversarially trained with the corresponding ϵ and the selective classifiers use a rejection radius of $\epsilon/2$). The results are shown for the natural choice of adversary, as in the experiment section (e.g. GMSA with $\mathcal{L}_{\text{REJ}}$ for the transduction+rejection). For selective classifiers, the rejection rate scaling is shown.

We see that the combination of rejection and transduction does indeed maintain high accuracy for larger ϵ; at $\epsilon = 0.6$, it has 96.2% of the robust accuracy that transduction alone had for $\epsilon = 0.3$. This aligns with the theory, given the increased constant factors of $\text{OPT}_{\mathcal{U}^2}$ in Corollary B.13 compared to the results for classifiers in (Montasser et al., 2021).

Note also the behavior of the inductive classifier: accuracy improves past $\epsilon = 0.6$. To see why, note that a model adversarially trained for $\epsilon \geqslant 1$ will return near-uniform predictions for all classes (resulting in a robust accuracy of approximately

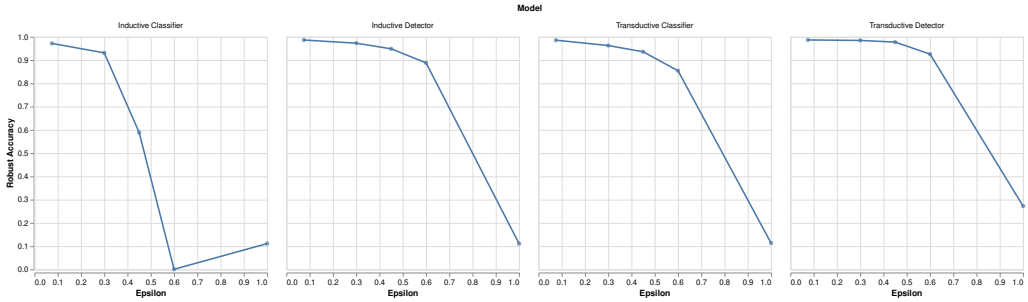


Figure B.5: Robustness scaling with adversarial budget ϵ on MNIST

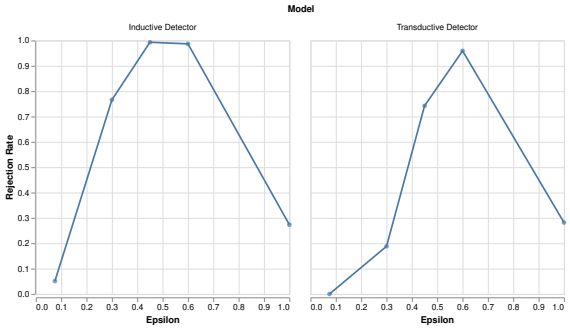


Figure B.6: Rejection rate scaling with adversarial budget ϵ on MNIST.

10%, as seen), making finding adversarial examples slightly more difficult than for smaller ϵ where this does not occur. The decline in rejection rate for very large ϵ is a similar phenomenon.

B.3.7 Weighting of $\mathcal{L}_{\text{REJ}}$

We examine the effect of the hyperparameter λ' between the cross-entropy and rejection-avoidance terms in $\mathcal{L}_{\text{REJ}}$ on MNIST; see Equation 3.8. In the inductive case, as shown in Figure B.7, there is little sensitivity to λ' in either attack success rate or rejection rate. When targeting TLDR, there is little sensitivity in terms of attack success rate as seen in Figure B.8; rejection rate is highest for intermediate values of λ' but, as expected, rejection rate declines with λ' beyond that.

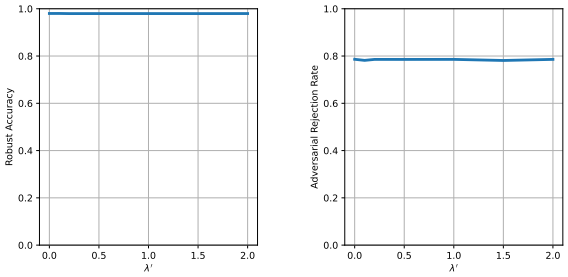


Figure B.7: Effects of λ' on results of PGD optimizing $\mathcal{L}_{\text{REJ}}$ targeting adversarial training with rejection on MNIST.

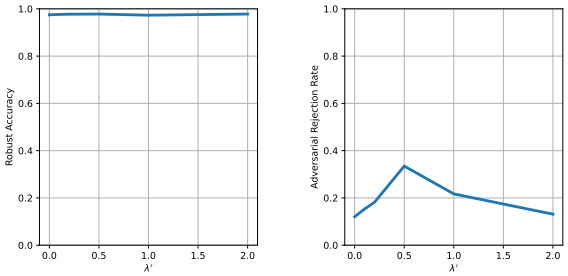


Figure B.8: Effects of λ' on results of GMSA optimizing $\mathcal{L}_{\text{REJ}}$ targeting TLDR on MNIST.

B.3.8 Robustness to l_2

Table B.5: Results on MNIST and CIFAR-10 up to l_2 budget. The strongest attack against each defense is shown. The best result is **boldfaced**.

Setting	Defense	Attacker	MNIST		CIFAR-10	
			p_{REJ}	Robust accuracy	p_{REJ}	Robust accuracy
Induction	AT (Madry et al., 2018a)	AutoAttack	–	0	–	0.445
Rejection only	AT (with rejection)	PGD ($\mathcal{L}_{\text{REJ}}$)	0.112	0.921	0.130	0.754
Transduction only	TADV (Chen et al., 2022)	GMSA ($\mathcal{L}_{\text{CE}}$)	–	0.913	–	0.813
Transduction+Rejection	**TLDR (ours)**	GMSA ($\mathcal{L}_{\text{REJ}}$)	**0.078**	**0.933**	**0.007**	**0.845**

To evaluate our defense's generality, we consider robustness to l_2 as well and compare to the strongest defenses from each setting in Table B.5; on MNIST we use $\epsilon = 5$ and on CIFAR-10 we use $\epsilon = 128/255$. We observe strong performance from TLDR, outperforming defenses with transduction or rejection alone.

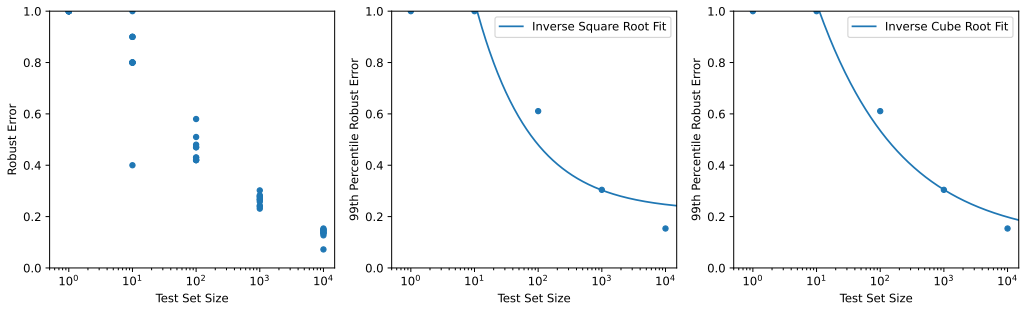


Figure B.9: Generalization of TLDR with equal train and test size on MNIST.

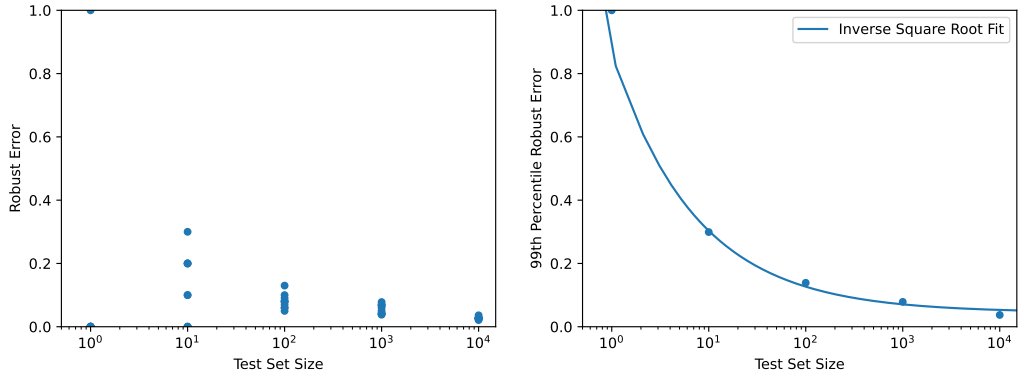


Figure B.10: Generalization of TLDR with full training set on MNIST.

B.3.9 Generalization of TLDR

To evaluate how closely TLDR's generalization follows the our provided bounds in Theorems B.9 and B.12, we apply TLDR to randomly-sampled subsets of the MNIST training and test sets. In each case, we run ten trials and present the robust error (1 - robust accuracy) with attacker GMSA ($\mathcal{L}_{\text{REJ}}$). Given the large VC dimension of the model considered (LeNet) (Bartlett et al., 2017), the results shown are consistent with Theorem B.12; we wish to determine whether the actual errors observed follow the inverse-square relationship of the theorem.

In Figure B.9, the size of the training set is set equal to the size of the test set (the standard assumption for our results); in Figure B.10, the full training set is used and only the test set size is changed. See Appendix B.1.5 for a discussion of

generalization bounds for train and test sets of differing sizes.

As the bounds are in PAC form, we use an estimate of the 99th percentile of error in order to evaluate the generalization of TLDR; these are calculated with a best-fit beta distribution of the results on each instance size.

We then consider the inverse-square-root fit of these 99th percentile error estimates; as the gurarantee takes the form of an upper bound, and error is upper bounded by 1, we exclude any error values equal to 1 (corresponding to instances where all trials had a robust accuracy of 0). We find that in the case where train size is fixed, the 99th percentile errors closely follow the inverse-square-root trend in the test set size m; while the results for equal train and test set sizes more closely follow an inverse-cube-root relationship in m.

B.4 Limitations

While our framework is theoretical-sound with lower sampled complexity than the rejection-only case and with more relaxed optimality condition than the transductive-only case, our sample complexity proof under the transductive rejection case requires the non-emptiness of Δ in Theorem 3.1. While weaker conditions don't guarantee that we find a model satisfying the conditions, the result demonstrate that empirical defense incorporating both transduction and rejection have the potential to outperform others. Our proposed defense algorithm TLDR, though effective at improving the robust accuracy under rejection, incurs a high computational cost relative to standard adversarial training due to the joint training with the unlabeled data. If it is possible to delay evaluation until a sufficiently large batch of samples arrives, the cost can be made insignificant via amortization. The need to perform a full training process prior to evaluation means, however, that the defense is not suitable for latency-sensitive applications. Our adaptive attack is even more costly, as effectively attacking this defense using GMSA requires multiple iterations of the full transductive training process; hence, adversaries attacking TLDR require substantial resources.

Algorithm 3 REJECTION-AWARE GMSA

Require: A clean training set T, a clean test set E, a transductive learning algorithm for classifiers $\mathbb{A}$, an adversarial budget of ϵ, $mode$ either MIN or AVG, a radius used for rejection $\epsilon_{\text{defense}}$, and a maximum number of iterations $N \geqslant 1$. $E|_X$ refers to the projection on the feature space for E.

Search for a perturbation of the test set which fools the model space induced by $(T, \mathcal{U}(E|_X))$.
$E' = E$
$\hat{E} = E$
$\text{err}_{\max} = -\inf$
for i=0,...,N-1 **do**
  Train a transductive model on the perturbed data.
  $h^{(i)} = \mathbb{A}(T, E'|_X)$

$$\text{err} = \frac{1}{|E'|} \sum_{i=1}^{|E'|} \mathbb{1} \left\{ \left(F(h^{(i)})(\tilde{x}_i) \notin \{\tilde{y}_i\} \wedge \tilde{x}_i = x_i \right) \vee \left(F(h^{(i)})(\tilde{x}_i) \notin \{\tilde{y}_i, \perp\} \wedge \tilde{x}_i \neq x_i \right) \right\}$$

{The $\tilde{x}_i$ and the x_i are the i^{th} datapoints of E' and E, repectively; y_i is the true label.}
  if $\text{err}_{\max} < \text{err}$ **then**
    $\hat{E} = E'$
  end if
  for $j = 1, \ldots, |E|$ **do**
    if $mode = \text{MIN}$ **then**

$$\tilde{x}_j = \arg \max_{\|\tilde{x} - x_j\| \leqslant \epsilon} \min_{1 \leqslant k \leqslant i} \mathcal{L}_{\text{REJ}_{h^{(k)}}}(\tilde{x}, y_j)$$

    else

$$\tilde{x}_j = \arg \max_{\|\tilde{x} - x_j\| \leqslant \epsilon} \frac{1}{i} \sum_{k=1}^{i} \mathcal{L}_{\text{REJ}_{h^{(k)}}}(\tilde{x}, y_j)$$

    end if
    {Select whether to perturb by comparing success rates against past models for the clean and perturbed samples.}

$$\underset{\text{clean}}{\text{err}} = \frac{1}{i} \sum_{0 \leqslant k \leqslant i} \mathbb{1} \left[F\left(h^{(k)}\right)(x_j) \neq y_j \right]$$

$$\underset{\text{perturbed}}{\text{err}} = \frac{1}{i} \sum_{0 \leqslant k \leqslant i} \mathbb{1} \left[F\left(h^{(k)}\right)(\tilde{x}_j) \notin \{y_j, \perp\} \right]$$

    {Do not perturb if the perturbation reduces robust rejection accuracy less on average than leaving the points unchanged.}
    if $\text{err}_{\text{perturbed}} < \text{err}_{\text{clean}}$ **then**
      $\tilde{x}_j = x_j$
    end if
    $E'_j = \tilde{x}_j, y_i$
  end for
end for
Return: $\hat{E}$

Algorithm 4 Transductive Rejection-Aware Transfer

Require: A model h, a clean labelled test point (x, y), an adversarial budget of ϵ, and a radius used for rejection $\epsilon_{\text{defense}}$.

{Search for a perturbation $\tilde{x}$ of x for which h predicts $\hat{y} \neq y$ robustly.}

1:
$$\tilde{x} = \arg \max_{\|\tilde{x}-x\| \leqslant \epsilon} \left[\mathcal{L}_{\text{CE}}(h^s(\tilde{x}), y) + \lambda \left\| \tilde{x} - \arg \max_{\|x'-\tilde{x}\| \leqslant \epsilon_{\text{defense}}} \mathcal{L}_{\text{DB},h}(x') \right\|, \right]$$

where $\mathcal{L}_{\text{CE}}$ is the cross-entropy loss, h^s returns the softmax activations of h and where $\mathcal{L}_{\text{DB},h}(x) = \text{rank}_2 h^s(x) - \max h^s(x)$.

{If the attack did not succeed against h (in other words, if h does not robustly predict $\hat{y} \neq y$), check whether to leave x unperturbed.}

2:
$$x' = \arg \max_{\|x'-\tilde{x}\| \leqslant \epsilon_{\text{defense}}} \mathcal{L}_{\text{CE}}(h^s(x'), h(\tilde{x}))$$

3: **if** $h(x') \neq h(\tilde{x}) \vee h(\tilde{x}) = y$ **then**
4: Leave x unperturbed if $F(h)$ rejects it, or if $h(x) \neq y$.
5:
$$x'' = \arg \max_{\|x''-x\| \leqslant \epsilon_{\text{defense}}} \mathcal{L}_{\text{CE}}(h^s(x''), h(x))$$

6: **if** $h(x) \neq y \vee h(x'') \neq h(x)$ **then**
7: $\tilde{x} = x$
8: **end if**
9: **end if**
10: **Return:** $\tilde{x}$

Algorithm 5 Inductive Rejection-Aware Attack

Require: A model h, and a clean labelled test point (x, y), an adversarial budget of ϵ, and a radius used for rejection $\epsilon_{\text{defense}}$.

1: Search for a perturbation $\tilde{x}$ of x for which h predicts $\hat{y} \neq y$ robustly.
$$\tilde{x} = \arg \max_{\|\tilde{x}-x\| \leqslant \epsilon} \left[\mathcal{L}_{\text{CE}}(h^s(\tilde{x}), y) + \lambda \left\| \tilde{x} - \arg \max_{\|x'-\tilde{x}\| \leqslant \epsilon_{\text{defense}}} \mathcal{L}_{\text{DB},h}(x') \right\| \right]$$

where $\mathcal{L}_{\text{CE}}$ is the cross-entropy loss, h^s returns the softmax activations of h and where $\mathcal{L}_{\text{DB},h}(x') = \text{rank}_2 h^s(x') - \max h^s(x')$

2: **Return:** $\tilde{x}$

C APPENDIX FOR CHAPTER 4

C.1 Links to Resources

Our dataset is available at https://huggingface.co/datasets/yguooo/newyorker_caption_ranking under Creative Commons Attribution Non Commercial 4.0. Our codebase is available at https://github.com/yguooo/cartoon-caption-generation under Apache 2.0.

C.2 Language Model Prompts

C.2.1 Description Generation

We use GPT-4o to generate descriptions for each cartoon. In the dataset from Hessel et al. (2022) each cartoon has a canny description, an uncanny description, a location, and a list of entity. Entity are words that is related to the cartoon. We used the five shot method to generate a set of descriptions. The five examples are randomly selected from the testing set, and we use the these same five example for every cartoon descriptions generation. An example of our prompt is shown below.

User: In this task, you will see a cartoon, then write two descriptions about the cartoon, one uncanny description and one canny description, then write the cartoon's location, and the entities of the cartoon. I am going to give you five examples first and you write the last sets of description.
User: <Insert Cartoon Image>
Assistant: The canny description is <insert canny description> and the uncanny description is <insert uncanny description>, and the cartoon's location is <insert location>, and the entities of the cartoon are <insert entities>
.....*Repeat user/assistant for four more examples*.....
User: <Insert Cartoon Image>. The set of description is

Table C.1: Examples of Generated Cartoon Descriptions

Type of descriptions	GPT-4o	Human Written (Hessel et al., 2022)
Canny description	`A knight in armor is riding a horse, holding a lance with a traffic light on top. A line of businessmen in suits follows behind him.`	`There are two men on a horse. They are wearing soldier outfits. Businessmen follow behind them.`
Uncanny Description	`It's unusual to see a medieval knight leading modern businessmen as if going into battle.`	`There are businessmen following a two guys on horses who are soldiers.`
Location	`an open field`	`a hilly path`
Entities	`Knight, Horse, Businessmen, Traffic light`	`Warrior, Horses in warfare, Businessperson`

C.2.2 Caption Evaluation

We evaluate various models that generate captions by comparing the generated captions against four groups of human contestant entries at different ranking levels, which include top10, #200-#209, #1000-#1009, and contestant median. As concluded based on Table 4.2, we use GPT4-Turbo as evaluator with descriptions from Hessel et al. (2022) in Overall Comparison and GPT4o-vision as evaluator with raw cartoon images in Best Pick Comparison. For both group comparison methods, we utilize the 5-shot in-context prompting technique, as mentioned in Section 4.4.2.

An example of Overall Comparison is shown below.

System: You are a judge for the new yorker cartoon caption contest.
User: In this task, you will see two description for a cartoon. Then, you will see two captions that were written about the cartoon. Then you will choose which captions is funnier. I am going to give you five examples first and you answer the last example with either A or B.
User: For example, the descriptions for the images are <Insert Canny Description> and <Insert Uncanny Description>. The two captions are A: <Insert CaptionA> B: <Insert CaptionB>
Assistant: The caption that is funnier is <Insert Answer>
.....*Repeat user/assistant for four more examples*.....
User: The descriptions for the images are <Insert Canny Description> and <Insert Uncanny Description>. The two groups of captions are group A: <Insert Caption Group A> group B: <Insert Caption Group B>
User: Choose the group of captions that is funnier. Answer with only one letter A or B, and nothing else.

An example of Best Pick Comparison is shown below.

System: You are a judge for the new yorker cartoon caption contest. Your job is to find the funniest caption.
User: In this task, you will see a cartoon first and two captions that were written about it then. The task is to choose which caption is funnier. I am going to show you five cartoons, corresponding captions and their answers first. In the end, for the last cartoon, answer with only one letter A or B, and nothing else.
User: <Insert Cartoon Image>
User: For this example, the two captions are A: <Insert CaptionA> B: <Insert CaptionB>. The answer is
Assistant: <Insert Answer>
.....*Repeat user/assistant for four more examples*.....
User: <Insert Cartoon Image>
User: Find the funniest caption for each group. Then choose the funnier group based on these funniest captions. Think step by step but finish the last line of your answer with only one letter A or B, and nothing else. A: <Insert Caption Group A> or B: <Insert Caption Group B>

C.2.3 Caption Generation

We used GPT-3.5-turbo, Claude-3-opus, and GPT-4-o to generate captions for each cartoons. We first use the system role to prompt it to generate 10 captions. Then we provide the image descriptions and then the image itself. For GPT-3.5-turbo, we simply only provided the image descriptions. For GPT-4-o, we have two versions where in one we provide the image itself, and the other we only provided the image descriptions. For Claude, we always provide both image description and image itself.

System: I want you to act as a sophisticated reader of The New Yorker Magazine. You are competing in The New Yorker Cartoon Caption Contest. Your task is to generate funny captions for a cartoon. Here are some ideas for developing funny captions. First think about characteristics associated with the objects and people featured in the cartoon. Then consider what are the unusual or absurd elements in the cartoon. It might help to imagine conversations between the characters. Then think about funny and non-obvious connections that can be made between the objects and characters. Try to come up with funny captions that fit the cartoon, but are not too direct. It may be funnier if the person reading the caption has to think a little bit to get the joke. Next, I will describe a cartoon image and then you should generate 10 funny captions for the cartoon along with an explanation for each.
User: <Insert Cartoon Image>
User: The cartoon's description is: <insert canny description>.The uncanny description is: <insert uncanny description>. The location of the cartoon is:<insert location>. The entities of the cartoon are: <insert image entities>

C.3 Additional Experiment Setups

C.3.1 Human Experiment Details

Each participant provided informed consent in compliance with our Institutional IRB and was compensated for their time. We paid participants $12 an hour and spent about $600 on data collection. The following instructions were used for the human experiments.

Human Pairwise with description generated by GPT4o-vision

In each trial of this task, you will see a description of a cartoon and two captions: the cartoon description is on the top, and the two caption choices are beneath the cartoon description. For each trial, please select the caption that is the funniest for the cartoon.

Human Pairwise with Cartoon Image

In each trial of this task, you will see one cartoon and two captions: the cartoon is on top, and the two caption choices are beneath the cartoon. For each trial, please select the caption that is the funniest for the cartoon.

Human Group (Overall) with description generated by GPT4o-vision

In each trial of this task, you will see a description of a cartoon and two groups of captions: the cartoon description is on the top, and the two grouped caption choices are beneath the cartoon description. For each trial, please select the group of captions that is the funniest for the cartoon.

Human Group (Overall) with Cartoon Image

In each trial of this task, you will see a cartoon and two groups of captions: the cartoon is on the top, and the two grouped caption choices are beneath the cartoon. For each trial, please select the group of captions that is the funniest for the cartoon.

Human Group (Best Pick) with description generated by GPT4o-vision

In each trial of this task, you will see a description of a cartoon and two groups of captions: the cartoon description is on the top, and the two grouped caption choices are beneath the cartoon description. For each trial, please select the group of captions that contains the funniest caption for the cartoon. First, pick the funniest caption in each group, and then compare between the two captions to pick the funniest group.

Human Group (Best Pick) with Cartoon Image

In each trial of this task, you will see a cartoon and two groups of captions: the cartoon is on the top, and the two grouped caption choices are beneath the cartoon. For each trial, please select the group of captions that contains the funniest caption

for the cartoon. First, pick the funniest caption in each group, and then compare between the two captions to pick the funniest group.

Human top 10 vs Claude generated captions

In each trial of this task, you will see a cartoon and two groups of captions: the cartoon is on the top, and the two grouped caption choices are beneath the cartoon. For each trial, please select the group of captions that is the funniest for the cartoon.

C.3.2 Recalibration of GPT Models for Ranking

For group comparisons without chain of thought, we observe a strong bias of GPT4 models choosing A over B. In other words, for some examples, the model always chooses option A even after we flip the two groups. Therefore, this suggests we need to calibrate the model predictions. We adopt a simple approach by readjusting the decision threshold. Let s_i^A, s_i^B denote the log probabilities of choosing A and B by the GPT4 model for two groups of human submitted captions x_i^A and x_i^B respectively. We use a small validation set of m examples $\{x_i^A, x_i^B\}_{i=1}^m$ with sigmoid scores $\{s_i^A, s_i^B\}_{i=1}^m$ and ground truth preference by the crowd denoted as $\{y_i \in \{A, B\}\}_{i=1}^m$. The current decision rule takes the form of $\widehat{f}(x_i^A, x_i^B) = \begin{cases} A & \text{if } s_i^A - s_i^B > 0 \\ B & \text{otherwise} \end{cases}$.

We simply set a different threshold τ, which induces $\widehat{f}_\tau(x_i^A, x_i^B) = \begin{cases} A & \text{if } s_i^A - s_i^B > \tau \\ B & \text{otherwise} \end{cases}$.

The threshold τ^* is chosen so that the accuracy over the validation set is maximized:

$$\tau^* = \arg\max_\tau \sum_{i=1}^m \mathbf{1}\{y_i = \widehat{f}_\tau(x_i^A, x_i^B)\}.$$

Ties are broken arbitrarily above. We then use the recalibrated decision rule with τ^* for all of our evaluations.

C.3.3 Finetuning Experiment Details

Our training and test split for finetuning range from contest 530 to 890. In particular, our dataset includes all the data of (Hessel et al., 2022) with ranking information within this range. ((Hessel et al., 2022) only contains contests up to #763.) Thus, we choose our test split to be the combination of testing (47 contests) and validation split (44 contests) of (Hessel et al., 2022) within the 530-890 range. The rest available contests form our training split.

Our finetuning methods are trained from Mistral 7B Instruct v0.1 and LLaVa v1.6 Mistral (multimodal case) via LoRA updates (Hu et al., 2021). We use a variant of Mistral 7b model as our initial reward model to finetune from [1]. The choice of reward is based on our benchmarking results of top reward models on our caption generation dataset (Table C.2). For SFT methods, we train on 1000 pairs of captions from each contest, with the preferred caption from the top 1000 captions and the alternative randomly sampled from the rest. For reward modeling, DPO and RLHF, we train on 1000 pairs of captions with three standard deviations apart according to Equation (4.1) per contest. Additionally, we train our model using the default choice of optimizer from TRL up to 1 epoch. Then, we search for the best hyper-parameter over the neighborhood of default parameters and pick the best performing model under our GPT-based group comparison metrics. For our reward model, we pick the best model based on the reward evaluation on the holdout set. For both pretrained and finetuned models, we use the same generation configuration file with temperature 0.7, top-p sampling probability 0.95, repetition penalty 1.15. When evaluating using the Best-of-N (BoN) method, we pick the top 10 captions based on the trained reward model, out of 50 generated candidates from caption generation models. Our choice of batch is 64 for SFT and reward model, and 128 for all other settings.

During the training process of DPO, PPO, SFT, we create a separate padding tokens and resize the token embedding of the pretrained model so that the text generation can terminate properly. Furthermore, in the loss design of SFT case,

[1] We use the pretrained reward model from https://huggingface.co/weqweasdas/RM-Mistral-7B

we only evaluate the next-token prediction loss on the caption segment, as all the training texts contain similar prompts. Since we only reported the iteration with the best results, early stopping occurs before a single epoch for the choice of best iterations.

We also noted that PPO performs the best when starting from the pretrained Mistral Instruct 7B model, whereas DPO performs the best from a sft checkpoint of Mistral. This SFT checkpoint needs to be tuned on simple prompts and does not render a better performance than the sft tuned on the best prompt (with all those descriptions).

Choice of Prompts In Table C.4, we document the best prompt we found for each training algorithm. Generally speaking, the zero-shot, SFT, preference learning algorithm each require simpler prompts than the one preceding them.

Computation Cost Finetuning a SFT, DPO, PPO model usually takes 2-4 days to train till convergence on a A100 machine. Evaluating a single number of each scenario cost roughly $5 on the openai platform.

C.4 Crowdsourced Caption Contest Ratings

As described in the text, we used a UCB Auer (2002) variant to encourage high-performing captions to recieve the votes. We experimented with standard UCB (see Algorithm 6) and KL-UCB specifically optimized for discrete rewards (Tanczos et al., 2017). The data repository labels datasets according to which algorithm was employed for each contest. In practice, using UCB in high-traffic asynchronous environments faces specific challenges. For example, we wanted to ensure that voters could only vote on one caption at a time, that the model sent batches of captions to users to reduce round trips to the server, and that the underlying model was able to update as frequently as possible. For more details on overcoming such challenges, see (Jamieson et al., 2015).

Algorithm 6 Upper Confidence Bound (UCB) Algorithm

1: **Initialization:** For each caption x, initialize $N_x(0) = 0$ and $\hat{\mu}_x = 0$.
2: **for** $t = 1$ to T **do**
3: Select caption $x_t = \arg\max_x \left(\hat{\mu}_x + \sqrt{\frac{2\ln(4N_x(t)^2)}{N_x(t)}} \right)$.
4: Observe the reward $r_t \in \{1, 2, 3\}$ for caption x_t.
5: Update the number of times action x_t has been selected: $N_{x_t}(t) = N_{x_t}(t-1) + 1$.
6: Update the empirical mean reward of action x_t:

$$\hat{\mu}_{x_t} = \frac{N_{x_t}(t-1) \cdot \hat{\mu}_{x_t} + r_t}{N_{x_t}(t)}$$

7: **end for**

C.5 Additional Results

We benchmark the performance of different reward model as in Table C.2. weqweasdas/RM-Mistral-7B and Eurus-RM-7B Instruct are the top two models with the highest reward ranking accuracy. We choose to use weqweasdas/RM-Mistral-7B because it generally achieves better ranking accuracy for various data settings that we experimented on.

In our experiment, we noticed that PPO algorithm requires a much more aggressive early stopping scheme than DPO and SFT. Thus, we further look at the training dynamics of the PPO algorithm in Table C.3. Here, the batch size is 128. It is worth noting that the result at iteration 0 has an lower overall win rate than the zero shot result in Table 4.3. The reason is that our PPO and DPO algorithms need to use a simpler prompt as in Table C.4 to generate meaningful texts. From Table C.3, we verified the steady increase of the mean reward and decrease of the training loss. However, the improvement on these metrics does not corresponds to an improvement of the overall humorous generation. We hypothesize that this is due to the complex nature of humor and the potential for out-of-distribution generations when running RLHF.

Table C.2: Reward model benchmark

	Reward Ranking Acc (%)
Mistral-7B Instruct	73.17
Llama-3-8B Instruct	74.01
Llama-2-7B Chat	72.63
weqweasdas/RM-Mistral-7B	**74.05**
Eurus-RM-7B	**74.18**
FsfairX-LLaMA3-RM-v0.1	73.72
Qwen1.5-7B-Chat	72.26

Table C.3: Training Dynamics of PPO

Iteration	0	10	20	30	40	50
Contestant Median (Overall Win Rate (%)) ↑	17.03	**24.73**	16.48	9.89	6.04	4.95
Mean Reward ↑	0.0057	0.0260	0.0186	0.1309	0.1356	**0.2587**
Loss ↓	0.3592	0.2001	0.1773	0.1709	0.0848	**0.0584**

Table C.4: Best choice of prompts for each training algorthm

	Best Choice of Prompt
Zero-Shot	[INST] <> I want you to act as a sophisticated reader of The New Yorker Magazine. You are competing in The New Yorker Cartoon Caption Contest. Your task is to generate funny captions for a cartoon. Here are some ideas for developing funny captions. First think about characteristics associated with the objects and people featured in the cartoon. Then consider what are the unusual or absurd elements in the cartoon. It might help to imagine conversations between the characters. Then think about funny and non-obvious connections that can be made between the objects and characters. Try to come up with funny captions that fit the cartoon, but are not too direct. It may be funnier if the person reading the caption has to think a little bit to get the joke. Next, I will describe a cartoon image and then you should generate 1 funny caption for the cartoon along with an explanation for each. scene: <*scene*> description: <*description*> uncanny description: <*uncanny description*> entities: <*entities*> <> funny caption: [/INST] <*sample caption*>
SFT	[INST]I want you to act as a sophisticated reader of The New Yorker Magazine. You are competing in The New Yorker Cartoon Caption Contest. Your task is to generate funny captions for a cartoon. Here are some ideas for developing funny captions. First think about characteristics associated with the objects and people featured in the cartoon. Then consider what are the unusual or absurd elements in the cartoon. It might help to imagine conversations between the characters. Then think about funny and non-obvious connections that can be made between the objects and characters. Try to come up with funny captions that fit the cartoon, but are not too direct. It may be funnier if the person reading the caption has to think a little bit to get the joke. Next, I will describe a cartoon image and then you should generate 1 funny caption for the cartoon[/INST] scene: <*scene*> description: <*description*> uncanny description: <*uncanny description*> entities: <*entities*>

	Best Choice of Prompt
LLaVA	[INST] I want you to act as a sophisticated reader of The New Yorker Magazine. You are competing in The New Yorker Cartoon Caption Contest. Your task is to generate funny captions for a cartoon. Here are some ideas for developing funny captions. First think about characteristics associated with the objects and people featured in the cartoon. Then consider what are the unusual or absurd elements in the cartoon. It might help to imagine conversations between the characters. Then think about funny and non-obvious connections that can be made between the objects and characters. Try to come up with funny captions that fit the cartoon, but are not too direct. It may be funnier if the person reading the caption has to think a little bit to get the joke. Next, I will provide a cartoon image with descriptions and then you should generate 1 funny caption for the cartoon along with an explanation for each. image: *<image>* scene: *<scene>* description: *<description>* uncanny description: *<uncanny description>* entities: *<entities>* Generate a funny caption for the image: [/INST] *<sample caption>*
DPO/PPO/Reward Model	scene: *<scene>* description: *<description>* uncanny description: *<uncanny description>* entities: *<entities>* funny caption: *<sample caption>*

D APPENDIX FOR CHAPTER 5

D.1 Technical Preliminaries

Additional Notations For two integer indices i and j, we denote $\delta_{ij} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases}$ as Kronecker delta.

Lemma D.1 (Adapt W to Different Context Size)**.** *Suppose $\bar{W}$ is the weight with context length m, then the induced W when evaluating on context of length m' is:*

$$W = \frac{m}{m'}\bar{W}$$

Proof. We note that $\bar{W}$ is the un-normalized weight, i.e. scaling with the inverse context size $1/m$. Only the normalized weight is preserved when applying to a sentence with a different context length.

Then, the prediction is given as:

$$\begin{aligned} \hat{y} &:= x_q^\top W X^\top y \\ &= x_q^\top \frac{1}{m'} W_{\text{Normalized}} X^\top y \\ &= x_q^\top \underbrace{\frac{1}{m'} m \bar{W}}_{=W} X^\top y \end{aligned}$$

Thus,

$$W = \frac{m}{m'}\bar{W}$$

□

Lemma D.2 (Mixed 4th-Order Moment of Gaussian)**.** *Suppose $x \sim \mathcal{N}(0, I), r \sim \mathcal{N}(0, \delta^2 I)$, then*

1.

$$\mathbb{E}[rr^\top W^\top x x^\top W r r^\top] = 2\delta^4 W^\top W + \delta^4 \operatorname{tr}(W^\top W) I \tag{D.1}$$

2.
$$\mathbb{E}[\mathbf{r}\mathbf{x}^\top \boldsymbol{W}^\top \mathbf{x}\mathbf{x}^\top \boldsymbol{W}\mathbf{x}\mathbf{r}^\top] = \left(\operatorname{tr}\left(\boldsymbol{W}^2\right) + \operatorname{tr}\left(\boldsymbol{W}^\top \boldsymbol{W}\right) + \operatorname{tr}\left(\boldsymbol{W}\right)^2\right)\delta^2 \mathrm{I} \tag{D.2}$$

3.
$$\mathbb{E}[\mathbf{x}\mathbf{r}^\top \boldsymbol{W}^\top \mathbf{x}\mathbf{x}^\top \boldsymbol{W}\mathbf{r}\mathbf{x}^\top] = 2\delta^2 \boldsymbol{W}\boldsymbol{W}^\top + \delta^2 \operatorname{tr}(\boldsymbol{W}^\top \boldsymbol{W})\mathrm{I} \tag{D.3}$$

4.
$$\mathbb{E}[\mathbf{r}\mathbf{x}^\top \boldsymbol{W}^\top \mathbf{x}\mathbf{x}^\top \boldsymbol{W}\mathbf{r}\mathbf{x}^\top] = \delta^2\left(\boldsymbol{W}^\top \boldsymbol{W} + \boldsymbol{W}^\top \boldsymbol{W}^\top + \boldsymbol{W}^\top \operatorname{tr}(\boldsymbol{W})\right) \tag{D.4}$$

5.
$$\mathbb{E}[\mathbf{r}\mathbf{r}^\top \boldsymbol{W}^\top \mathbf{x}\mathbf{x}^\top \boldsymbol{W}\mathbf{x}\mathbf{x}^\top] = \delta^2\left(\boldsymbol{W}^\top \boldsymbol{W} + \boldsymbol{W}^\top \boldsymbol{W}^\top + \boldsymbol{W}^\top \operatorname{tr}(\boldsymbol{W})\right) \tag{D.5}$$

Proof. 1. We have

$$\begin{aligned}
\mathop{\mathbb{E}}_{\mathbf{x},\mathbf{r}}[\mathbf{r}\mathbf{r}^\top \boldsymbol{W}^\top \mathbf{x}\mathbf{x}^\top \boldsymbol{W}\mathbf{r}\mathbf{r}^\top] &= \mathop{\mathbb{E}}_{\mathbf{r}}[\mathbf{r}\mathbf{r}^\top \boldsymbol{W}^\top \boldsymbol{W}\mathbf{r}\mathbf{r}^\top] \\
&= 2\delta^2 \mathrm{I}\boldsymbol{W}^\top \boldsymbol{W}\delta^2 \mathrm{I} + \operatorname{tr}(\boldsymbol{W}^\top \boldsymbol{W}\delta^2 \mathrm{I})\delta^2 \mathrm{I} \\
&= 2\delta^4 \boldsymbol{W}^\top \boldsymbol{W} + \delta^4 \operatorname{tr}(\boldsymbol{W}^\top \boldsymbol{W})\mathrm{I} \\
&= 2\delta^4 \boldsymbol{W}^\top \boldsymbol{W} + \delta^4 \operatorname{tr}(\boldsymbol{W}^\top \boldsymbol{W})\mathrm{I}
\end{aligned} \tag{D.6}$$

where the first step follows from Equation (D.14).

2.
$$\begin{aligned}
\mathop{\mathbb{E}}_{\mathbf{x},\mathbf{r}}[\mathbf{r}\mathbf{x}^\top \boldsymbol{W}^\top \mathbf{x}\mathbf{x}^\top \boldsymbol{W}\mathbf{x}\mathbf{r}^\top] &= \mathop{\mathbb{E}}_{\mathbf{r}}\left[\mathbf{r}\mathop{\mathbb{E}}_{\mathbf{x}}\left[\mathbf{x}^\top \boldsymbol{W}^\top \mathbf{x}\mathbf{x}^\top \boldsymbol{W}\mathbf{x}\right]\mathbf{r}^\top\right] \\
&= \left(\operatorname{tr}\left(\boldsymbol{W}^2\right) + \operatorname{tr}\left(\boldsymbol{W}^\top \boldsymbol{W}\right) + \operatorname{tr}\left(\boldsymbol{W}\right)^2\right)\delta^2 \mathrm{I}
\end{aligned} \tag{D.7}$$

where the first step follows from Equation (D.16).

3.
$$\begin{aligned}
\mathbb{E}[\mathbf{x}\mathbf{r}^\top \boldsymbol{W}^\top \mathbf{x}\mathbf{x}^\top \boldsymbol{W}\mathbf{r}\mathbf{x}^\top] &= \mathbb{E}\left[\mathbf{x}\,\mathbb{E}\left[\operatorname{tr}\left(\mathbf{r}^\top \boldsymbol{W}^\top \mathbf{x}\mathbf{x}^\top \boldsymbol{W}\mathbf{r}\right)\right]\mathbf{x}^\top\right] \\
&= \mathbb{E}\left[\mathbf{x}\,\mathbb{E}\left[\operatorname{tr}\left(\mathbf{r}\mathbf{r}^\top \boldsymbol{W}^\top \mathbf{x}\mathbf{x}^\top \boldsymbol{W}\right)\right]\mathbf{x}^\top\right] \\
&= \delta^2\,\mathbb{E}\,\mathbf{x}\operatorname{tr}(\boldsymbol{W}^\top \mathbf{x}\mathbf{x}^\top \boldsymbol{W})\mathbf{x}^\top \\
&= \delta^2\,\mathbb{E}\,\mathbf{x}\operatorname{tr}(\mathbf{x}^\top \boldsymbol{W}\boldsymbol{W}^\top \mathbf{x})\mathbf{x}^\top \\
&= \delta^2\,\mathbb{E}\,\mathbf{x}\mathbf{x}^\top \boldsymbol{W}\boldsymbol{W}^\top \mathbf{x}\mathbf{x}^\top \\
&= 2\delta^2 \boldsymbol{W}\boldsymbol{W}^\top + \delta^2 \operatorname{tr}(\boldsymbol{W}\boldsymbol{W}^\top)\mathrm{I} \\
&= 2\delta^2 \boldsymbol{W}\boldsymbol{W}^\top + \delta^2 \operatorname{tr}(\boldsymbol{W}^\top \boldsymbol{W})\mathrm{I}
\end{aligned} \tag{D.8}$$

where the first three steps follow from the cyclic property of trace and the last step follows from Equation (D.14).

4.
$$\begin{aligned}
\mathbb{E}[\mathbf{r}\mathbf{x}^\top \boldsymbol{W}\mathbf{x}\mathbf{x}^\top \boldsymbol{W}\mathbf{r}\mathbf{x}^\top] &= \mathbb{E}\left[\mathbf{r}(\mathbf{x}^\top \boldsymbol{W}\mathbf{r})^\top \mathbf{x}^\top \boldsymbol{W}\mathbf{x}\mathbf{x}^\top\right] \\
&= \mathbb{E}\left[\mathbf{r}\mathbf{r}^\top \boldsymbol{W}^\top \mathbf{x}\mathbf{x}^\top \boldsymbol{W}\mathbf{x}\mathbf{x}^\top\right] \\
&= \delta^2\,\mathbb{E}\left[\boldsymbol{W}^\top \mathbf{x}\mathbf{x}^\top \boldsymbol{W}\mathbf{x}\mathbf{x}^\top\right] \\
&= \delta^2 \boldsymbol{W}^\top\left(\boldsymbol{W} + \boldsymbol{W}^\top + \operatorname{tr}(\boldsymbol{W})\mathrm{I}\right) \\
&= \delta^2\left(\boldsymbol{W}^\top \boldsymbol{W} + \boldsymbol{W}^\top \boldsymbol{W}^\top + \boldsymbol{W}^\top \operatorname{tr}(\boldsymbol{W})\right)
\end{aligned}$$

where the first step follows from $\mathbf{x}^\top \boldsymbol{W}\mathbf{r}$ being scalar, and the third step follows from Equation (D.14).

5.
$$\begin{aligned}
\mathbb{E}[\mathbf{r}\mathbf{r}^\top \boldsymbol{W}^\top \mathbf{x}\mathbf{x}^\top \boldsymbol{W}\mathbf{x}\mathbf{x}^\top] &= \delta^2 \boldsymbol{W}^\top\left(\boldsymbol{W} + \boldsymbol{W}^\top + \operatorname{tr}\left(\boldsymbol{W}\right)\right) \\
&= \delta^2\left(\boldsymbol{W}^\top \boldsymbol{W} + \boldsymbol{W}^\top \boldsymbol{W}^\top + \boldsymbol{W}^\top \operatorname{tr}(\boldsymbol{W})\right)
\end{aligned} \tag{D.9}$$

It follows from the application of Equation (D.14).

□

Lemma D.3 (Expectation of 6th-Order Gaussian Monomial)**.** *If* $\boldsymbol{x} \sim \mathcal{N}(0, \mathrm{I})$*, then*

$$\begin{aligned}\mathbb{E}[\boldsymbol{x}\boldsymbol{x}^\top A\boldsymbol{x}\boldsymbol{x}^\top B\boldsymbol{x}\boldsymbol{x}^\top] = {} & AB + AB^\top + A^\top B + A^\top B^\top + B^\top A + B^\top A^\top + BA + BA^\top \\ & + \operatorname{tr}(B)A + \operatorname{tr}(B)A^\top + \operatorname{tr}(A)B + \operatorname{tr}(A)B^\top \\ & + \operatorname{tr}(A)\operatorname{tr}(B)\mathrm{I} + \operatorname{tr}\left(AB^\top\right)\mathrm{I} + \operatorname{tr}(AB)\mathrm{I}\end{aligned} \tag{D.10}$$

$$\begin{aligned}\mathbb{E}[\boldsymbol{x}\boldsymbol{x}^\top \boldsymbol{W}^\top \boldsymbol{x}\boldsymbol{x}^\top \boldsymbol{W}\boldsymbol{x}\boldsymbol{x}^\top] &= \mathbb{E}[\boldsymbol{x}\boldsymbol{x}^\top \boldsymbol{W}\boldsymbol{x}\boldsymbol{x}^\top \boldsymbol{W}\boldsymbol{x}\boldsymbol{x}^\top] \\ &= 2\left(\boldsymbol{W}^2 + \boldsymbol{W}^\top\boldsymbol{W}^\top + \boldsymbol{W}^\top\boldsymbol{W} + \boldsymbol{W}\boldsymbol{W}^\top + \operatorname{tr}(\boldsymbol{W})\boldsymbol{W} + \operatorname{tr}(\boldsymbol{W})\boldsymbol{W}^\top\right) \\ &\quad + \operatorname{tr}(\boldsymbol{W})^2\,\mathrm{I} + \operatorname{tr}\left(\boldsymbol{W}^2\right)\mathrm{I} + \operatorname{tr}\left(\boldsymbol{W}^\top\boldsymbol{W}\right)\mathrm{I}\end{aligned} \tag{D.11}$$

Proof. Let $T := \mathbb{E}[\boldsymbol{x}\boldsymbol{x}^\top A\boldsymbol{x}\boldsymbol{x}^\top B\boldsymbol{x}\boldsymbol{x}^\top]$. Then, let's consider one scalar entry:

$$T_{ij} = \mathbb{E}\left[\sum_{k,\ell,m,n} x_i x_k A_{k\ell} x_\ell x_m B_{mn} x_n x_j\right] = \sum_{k,\ell,m,n} A_{k\ell}B_{mn} \cdot \mathbb{E}\left[x_i x_k x_\ell x_m x_n x_j\right] \tag{D.12}$$

We now need to compute the 6th-order central moment of standard normal variables. This can be computed using the Isserlis' Theorem (Isserlis, 1918):

$$\mathbb{E}[x_1 \cdots x_s] = \sum_{p \in P_s^2} \prod_{(i,j)\in p} \mathbb{E}[x_i x_j] \tag{D.13}$$

where P_s^2 stands for all distinct ways of partitioning $\{1, \ldots, s\}$ into pairs i, j (perfect matching), and the product is over the pairs contained in p.

We note that the number of perfect matching for s examples is given as:

$$\#\text{perfect matching} = \frac{s!}{2^{s/2}(s/2)!}$$

where $2^{s/2}$ is for ignoring the ordering inside pairs and $(s/2)!$ is for ignoring the ordering between pairs.

We note that there are $\frac{6!}{2^3 \cdot 3!} = 15$ distinct partitions for the 6-th order product of Gaussian random variable. Suppose $(x_a, x_b), (x_c, x_d), (x_e, x_f)$ is a valid pairing, then:

$$\mathbb{E}[x_a x_b]\,\mathbb{E}[x_c x_d]\,\mathbb{E}[x_e x_f] = \begin{cases} 1 & \text{if } a = b, c = d, e = f \\ 0 & \text{else} \end{cases} = \delta_{ab} \cdot \delta_{cd} \cdot \delta_{ef}$$

where $\delta_{ij} := \mathbb{1}[i = j]$ stands for the Kronecker delta.

Here, we will discuss the result for all 15 distinct pairings:

1. $(i,k)(l,m)(n,j)$

$$\sum_{k,l,m,n} A_{kl} B_{mn} = \sum_{m} A_{im} B_{mj} = A_{i\cdot} B_{\cdot j} = (AB)_{ij}$$

2. $(i,k)(l,n)(m,j)$

$$\sum_{k,l,m,n} A_{kl} B_{mn} = \sum_{m} A_{im} B_{jm} = A_{i\cdot} B_{j\cdot} = (AB^\top)_{ij}$$

3. $(i,k)(l,j)(m,n)$

$$\sum_{k,l,m,n} A_{kl} B_{mn} = \sum_{m} A_{ij} B_{mm} = \mathrm{tr}(B) A_{ij}$$

4. $(i,l)(k,m)(n,j)$

$$\sum_{k,l,m,n} A_{kl} B_{mn} = \sum_{m} A_{mi} B_{mj} = A_{\cdot i} B_{\cdot j} = (A^\top B)_{ij}$$

5. $(i,l)(k,n)(m,j)$

$$\sum_{k,l,m,n} A_{kl} B_{mn} = \sum_{k} A_{ki} B_{jk} = A_{\cdot i} B_{j\cdot} = (A^\top B^\top)_{i,j}$$

6. $(i,l)(k,j)(m,n)$

$$\sum_{k,l,m,n} A_{kl}B_{mn} = \sum_{m} A_{ji}B_{mm} = (A^\top)_{ij}\,\mathrm{tr}(B)$$

7. $(i,m)(k,l)(n,j)$

$$\sum_{k,l,m,n} A_{kl}B_{mn} = \sum_{k} A_{kk}B_{ij} = \mathrm{tr}(A)B_{ij}$$

8. $(i,m)(k,n)(l,j)$

$$\sum_{k,l,m,n} A_{kl}B_{mn} = \sum_{k} A_{kj}B_{ik} = A_{\cdot j}B_{i\cdot} = (BA)_{ij}$$

9. $(i,m)(k,j)(l,n)$

$$\sum_{k,l,m,n} A_{kl}B_{mn} = \sum_{l} A_{jl}B_{il} = A_{j\cdot}B_{i\cdot} = (BA^\top)_{ij}$$

10. $(i,n)(k,l)(m,j)$

$$\sum_{k,l,m,n} A_{kl}B_{mn} = \sum_{k} A_{kk}B_{ji} = \mathrm{tr}(A)(B^\top)_{ij}$$

11. $(i,n)(k,m)(l,j)$

$$\sum_{k,l,m,n} A_{kl}B_{mn} = \sum_{m} A_{mj}B_{mi} = A_{\cdot j}B_{\cdot i} = (B^\top A)_{ij}$$

12. $(i,n)(k,j)(l,m)$

$$\sum_{k,l,m,n} A_{kl}B_{mn} = \sum_{m} A_{jm}B_{mi} = A_{j\cdot}B_{\cdot i} = (B^\top A^\top)_{ij}$$

13. $(i,j)(k,l)(m,n)$

$$\sum_{k,l,m,n} A_{kl}B_{mn} = \sum_{k,m} A_{kk}B_{mm} = \mathrm{tr}(A)\,\mathrm{tr}(B)\delta_{ij}$$

14. $(i,j)(k,m)(l,n)$

$$\sum_{k,l,m,n} A_{kl}B_{mn} = \sum_{k,l} A_{kl}B_{kl} = \mathrm{tr}(AB^\top)\delta_{ij}$$

15. $(i,j)(k,n)(l,m)$

$$\sum_{k,l,m,n} A_{kl}B_{mn} = \sum_{m,k} A_{km}B_{mk} = \mathrm{tr}(AB)\delta_{ij}$$

Summing up all of these 15 terms together, we obtain Eq. equation D.10. Then, we plug in $A = \boldsymbol{W}, B = \boldsymbol{W}^\top$, we obtain Eq. equation D.11. □

Lemma D.4 (4th-Order Gaussian Monomial)**.** *Let* $\boldsymbol{x}, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_m \sim \mathcal{N}(0, I)$ *and* $\boldsymbol{X} = [\boldsymbol{x}_1^\top; \ldots; \boldsymbol{x}_m^\top]$*. Then, we have*

$$\begin{aligned} \mathbb{E}\,\boldsymbol{x}\boldsymbol{x}^\top \boldsymbol{W} \boldsymbol{x}\boldsymbol{x}^\top &= \boldsymbol{W} + \boldsymbol{W}^\top + \mathrm{tr}(\boldsymbol{W})I \\ &= 2\boldsymbol{W} + \mathrm{tr}(\boldsymbol{W})I \quad \textit{if } \boldsymbol{W} \textit{ is symmetric} \end{aligned} \tag{D.14}$$

and

$$\begin{aligned} \mathbb{E}\,\boldsymbol{X}^\top \boldsymbol{X} \boldsymbol{W} \boldsymbol{X}^\top \boldsymbol{X} &= m^2\boldsymbol{W} + m\boldsymbol{W}^\top + m\,\mathrm{tr}(\boldsymbol{W})I \\ &= m(m+1)\boldsymbol{W} + m\,\mathrm{tr}(\boldsymbol{W})I \quad \textit{if } \boldsymbol{W} \textit{ is symmetric} \end{aligned} \tag{D.15}$$

$$\mathbb{E}\,\boldsymbol{x}^\top A \boldsymbol{x}\boldsymbol{x}^\top B \boldsymbol{x} = \mathrm{tr}\left(A\left(B + B^\top\right)\right) + \mathrm{tr}(A)\,\mathrm{tr}(B) \tag{D.16}$$

If $A = \boldsymbol{W}^\top, B = \boldsymbol{W}$*, then*

$$\mathbb{E}\,\boldsymbol{x}^\top \boldsymbol{W}^\top \boldsymbol{x}\boldsymbol{x}^\top \boldsymbol{W}\boldsymbol{x} = \mathbb{E}\,\boldsymbol{x}^\top \boldsymbol{W}\boldsymbol{x}\boldsymbol{x}^\top \boldsymbol{W}\boldsymbol{x} = \mathrm{tr}(\boldsymbol{W}^\top\boldsymbol{W}) + \mathrm{tr}(\boldsymbol{W}^2) + \mathrm{tr}(\boldsymbol{W})^2 \tag{D.17}$$

Proof. Equation (D.14) follows from section 8.2.4 of (Petersen et al., 2008) by plugging in mean 0 and variance I.

$$\begin{aligned}\mathbb{E}\,\mathbf{X}^\top \mathbf{X}\boldsymbol{W}\mathbf{X}^\top \mathbf{X} &= \sum_i \mathbf{x}_i \mathbf{x}_i^\top \boldsymbol{W} \mathbf{x}_i \mathbf{x}_i^\top + \sum_{i \neq j} \mathbf{x}_i \mathbf{x}_i^\top \boldsymbol{W} \mathbf{x}_j \mathbf{x}_j^\top \\ &= m\left(\boldsymbol{W} + \boldsymbol{W}^\top + \operatorname{tr}(\boldsymbol{W})\mathrm{I}\right) + m(m-1)\boldsymbol{W} \\ &= m^2 \boldsymbol{W} + m\boldsymbol{W}^\top + m \operatorname{tr}(\boldsymbol{W})\mathrm{I} \\ &= m(m+1)\boldsymbol{W} + m \operatorname{tr}(\boldsymbol{W})\mathrm{I}\end{aligned} \tag{D.18}$$

where the second step follows from plugging in Equation (D.14).

Equation (D.16) follows from section 8.2.4 of (Petersen et al., 2008) by plugging in mean 0 and variance I.

□

D.2 Additional Proof for RAG

Here, we provide an overview of the organization of the proof. First, we consider the uniform retrieval noise scenario, and compute the population loss for generic $\boldsymbol{W}$ in Theorem 5.1. Then, we plug in the special case $\boldsymbol{W}^*$ (isotropic pretrained weight), and provide a closed-form loss in Proposition 3. Then, we analyze its finite sample complexity in Proposition 1 and the optimal RAG examples in relation to ICL examples in Proposition 2.

Later on, we provide an finite sample complexity analysis for non-uniform retrieval noise, Theorem 5.2 for Distance Proportional Noise, and Theorem 5.3 for Distance-Weighted Mixture Noise.

D.2.1 Uniform Retrieval Noise

Theorem (Restatement of Theorem 5.1)**.** *Under Assumption 1, 2, 3, the population loss of the linear self-attention predictor* $\hat{y}_q = x_q^\top W X^\top y$ *satisfies*

$$\mathcal{L}_{tt+rag}(W) = \underbrace{\mathbb{E}\left(\mathbb{E}\left(\hat{y}_q\right) - \hat{y}_q\right)^2}_{:=\mathrm{err}_{variance}(W)} + \underbrace{\mathbb{E}\left(\mathbb{E}\left(\hat{y}_q\right) - \mathbb{E}\left(y_q\right)\right)^2}_{:=\mathrm{err}_{bias}(W)} + \underbrace{\sigma^2}_{irreducible\ noise} \tag{D.19}$$

Specifically,

$$\begin{aligned}
\underset{variance}{\mathrm{err}}(W) &= \left[n\sigma^2 + \left(1+\delta^2\right) n\sigma_{rag}^2\right] \mathrm{tr}(W^\top W) + n\sigma_{rag}^2 \,\mathrm{tr}(W^2) + n\sigma_{rag}^2 \,\mathrm{tr}(W)^2 \\
\underset{bias}{\mathrm{err}}(W) &= \beta_{tt}^\top \left[I - (n\delta^2 + 2n + m)(W + W^\top) - 2n\,\mathrm{tr}(W) I + M_4\right] \beta_{tt} \\
&= \beta_{tt}^\top \left[I - (n\delta^2 + 2n + m)(W + W^\top) - 2n\,\mathrm{tr}(W) I \right. \\
&\quad + \underbrace{\left[n^2\left(2+\delta^2\right) + n\left(m+\delta^2\right)\right]}_{:=c_1} \left(W^2 + \left(W^2\right)^\top\right) + \underbrace{2n(n+\delta^2)}_{:=c_2} WW^\top \\
&\quad + \underbrace{\left[m^2 + m + mn\left(2+2\delta^2\right) + n^2\left(2+2\delta^2+\delta^4\right) + n\left(2\delta^2+\delta^4\right)\right]}_{:=c_3} W^\top W \\
&\quad + \underbrace{\left[n^2\left(2+\delta^2\right) + n\left(m+\delta^2\right)\right]}_{:=c_4,\ c_4=c_1} \left(\mathrm{tr}(W)\left(W + W^\top\right)\right) \\
&\quad \left. + \underbrace{\left[n^2 + n\delta^2\right]}_{:=c_5} \left(\mathrm{tr}(W)^2 + \mathrm{tr}\left(W^2\right)\right) I + \underbrace{\left[m + n^2 + n\left(2\delta^2+\delta^4\right)\right]}_{:=c_6} \mathrm{tr}\left(W^\top W\right) I \right] \beta_{tt}
\end{aligned} \tag{D.20}$$

Proof. For computational convenience, I will define the following quantities for Gram matrix: $G_0 = X_{icl}^\top X_{icl}$, $G_i := (x_q + r_i)(x_q + r_i)^\top$, and $G := G_0 + \sum_{i\in[n]} G_i$.

We write down the error explicitly:

$$\begin{aligned}
y_q - x_q^\top W X^\top y &= x_q^\top \beta_{tt} + \epsilon_q - x_q^\top W X^\top X \beta_{tt} - x_q^\top W X^\top \epsilon \\
&= x_q^\top \left(\beta_{tt} - WG\beta_{tt}\right) - x_q^\top W X^\top \epsilon + \epsilon_q \\
&= x_q^\top \left(I - WG\right) \beta_{tt} - x_q^\top W X^\top \epsilon + \epsilon_q
\end{aligned} \tag{D.21}$$

Therefore, the population loss is equal to:

$$\mathcal{L}_{\text{tt+rag}}(\boldsymbol{W}) = \mathop{\mathbb{E}}_{(\mathbf{x}_q,y_q),(\mathbf{X},\mathbf{y}),\boldsymbol{\epsilon}} \left[\left(\mathbf{x}_q^\top \left(\mathrm{I} - \boldsymbol{W}\mathbf{G} \right) \beta_{\text{tt}} - \mathbf{x}_q^\top \boldsymbol{W}\mathbf{X}^\top \boldsymbol{\epsilon} \right)^2 \right] + \sigma^2$$

We note that both $\boldsymbol{\epsilon}_{\text{icl}}$ and $\boldsymbol{\epsilon}_{\text{rag}}$ are independent of $\mathbf{x}_q, \mathbf{X}$ (including $\mathbf{r}$), and $\mathbb{E}[\boldsymbol{\epsilon}] = 0$.

$$\mathop{\mathbb{E}}_{\boldsymbol{\epsilon}} \left[-2 \left(\mathbf{x}_q^\top \left(\mathrm{I} - \boldsymbol{W}\mathbf{G} \right) \beta_{\text{tt}} \right) \left(\mathbf{x}_q^\top \boldsymbol{W}\mathbf{X}^\top \boldsymbol{\epsilon} \right) \right] = 0$$

And therefore, we have the following loss decomposition:

$$\mathcal{L}_{\text{tt+rag}}(\boldsymbol{W}) = \mathop{\mathbb{E}}_{\mathbf{x}_q,\mathbf{X},\boldsymbol{\epsilon}} \left[\left(\mathbf{x}_q^\top \boldsymbol{W}\mathbf{X}^\top \boldsymbol{\epsilon} \right)^2 \right] + \mathop{\mathbb{E}}_{\mathbf{x}_q,\mathbf{X}} \left[\left(\mathbf{x}_q^\top \left(\mathrm{I} - \boldsymbol{W}\mathbf{G} \right) \beta_{\text{tt}} \right)^2 \right] + \sigma^2 \tag{D.22}$$

Then, we compute the mean of the prediction and the label:

$$\mathop{\mathbb{E}}_{\epsilon_q} y_q = \mathop{\mathbb{E}}_{\epsilon_q} \left(\mathbf{x}_q^\top \beta_{\text{tt}} + \epsilon_q \right) = \mathbf{x}_q^\top \beta_{\text{tt}}$$

$$\begin{aligned} \mathop{\mathbb{E}}_{\boldsymbol{\epsilon}} \hat{y}_q &= \mathbb{E}\, \mathbf{x}_q^\top \boldsymbol{W}\mathbf{X}^\top \mathbf{y} \\ &= \mathbb{E}\, \mathbf{x}_q^\top \boldsymbol{W}\mathbf{X}^\top (\mathbf{X}\beta_{\text{tt}} + \boldsymbol{\epsilon}) \\ &= \mathbb{E}\, \mathbf{x}_q^\top \boldsymbol{W}\mathbf{G}\beta_{\text{tt}} \end{aligned}$$

And further, we have

$$\begin{aligned} \mathop{\mathbb{E}}_{\epsilon_q} \left(y_q - \mathop{\mathbb{E}}_{\epsilon_q} y_q \right)^2 &= \mathop{\mathbb{E}}_{\epsilon_q} \left(\mathbf{x}_q^\top \beta_{\text{tt}} + \epsilon_q - \mathbf{x}_q^\top \beta_{\text{tt}} \right)^2 = \mathop{\mathbb{E}}_{\epsilon_q} \epsilon_q^2 = \sigma^2 \\ \left(\hat{y}_q - \mathop{\mathbb{E}}_{\boldsymbol{\epsilon}} \hat{y}_q \right)^2 &= \left(\mathbf{x}_q^\top \boldsymbol{W}\mathbf{X}^\top (\mathbf{X}\beta_{\text{tt}} + \boldsymbol{\epsilon}) - \mathbf{x}_q^\top \boldsymbol{W}\mathbf{X}^\top \mathbf{X}\beta_{\text{tt}} \right)^2 = \left(\mathbf{x}_q^\top \boldsymbol{W}\mathbf{X}^\top \boldsymbol{\epsilon} \right)^2 \\ \left(\mathop{\mathbb{E}}_{\epsilon_q} (y_q) - \mathop{\mathbb{E}}_{\boldsymbol{\epsilon}} \hat{y}_q \right)^2 &= \left(\mathbf{x}_q^\top \beta_{\text{tt}} - \mathbf{x}_q^\top \boldsymbol{W}\mathbf{G}\beta_{\text{tt}} \right)^2 = \left(\mathbf{x}_q^\top (\mathrm{I} - \boldsymbol{W}\mathbf{G}) \beta_{\text{tt}} \right)^2 \end{aligned} \tag{D.23}$$

If we plug Equation (D.23) into the loss decomposition Equation (D.22), we have

$$\begin{aligned}\mathcal{L}_{\text{tt+rag}}(\boldsymbol{W}) &= \mathop{\mathbb{E}}_{\boldsymbol{x}_q,\boldsymbol{X},\boldsymbol{\epsilon}}\left[\left(\boldsymbol{x}_q^\top \boldsymbol{W}\boldsymbol{X}^\top\boldsymbol{\epsilon}\right)^2\right] + \mathop{\mathbb{E}}_{\boldsymbol{x}_q,\boldsymbol{X}}\left[\left(\boldsymbol{x}_q^\top\left(\mathrm{I}-\boldsymbol{W}\mathbf{G}\right)\beta_{\text{tt}}\right)^2\right] + \sigma^2 \\ &= \underbrace{\mathbb{E}\left(\mathop{\mathbb{E}}_{\boldsymbol{\epsilon}}(\hat{y}_q) - \hat{y}_q\right)^2}_{:=\text{err}_{\text{variance}}(\boldsymbol{W})} + \underbrace{\mathbb{E}\left(\mathop{\mathbb{E}}_{\epsilon_q}(\hat{y}_q) - \mathop{\mathbb{E}}_{\boldsymbol{\epsilon}}(y_q)\right)^2}_{:=\text{err}_{\text{bias}}(\boldsymbol{W})} + \underbrace{\mathbb{E}\left(y_q - \mathop{\mathbb{E}}_{\epsilon_q} y_q\right)^2}_{\substack{=\sigma^2 \\ \text{(irreducible noise)}}} \end{aligned} \tag{D.24}$$

and we can obtain the bias-variance tradeoff as given in Equation (D.19).

Compute $\mathbb{E}_{\boldsymbol{x}_q,\boldsymbol{X},\boldsymbol{r},\boldsymbol{\epsilon}}\left[\left(\boldsymbol{x}_q^\top \boldsymbol{W}\boldsymbol{X}^\top\boldsymbol{\epsilon}\right)^2\right]$ First, we let

$$z := \boldsymbol{x}_q^\top \boldsymbol{W}\boldsymbol{X}^\top\boldsymbol{\epsilon} = \sum_{i=1}^{m+n} \boldsymbol{x}_q^\top \boldsymbol{W}\boldsymbol{x}_i \cdot \epsilon_i$$

Then,

$$z^2 = \sum_{i,j=1}^{m+n} (\boldsymbol{x}_q^\top \boldsymbol{W}\boldsymbol{x}_i)(\boldsymbol{x}_q^\top \boldsymbol{W}\boldsymbol{x}_j)\epsilon_i\epsilon_j = \sum_{i,j=1}^{m+n} (\boldsymbol{x}_i^\top \boldsymbol{W}^\top\boldsymbol{x}_q)(\boldsymbol{x}_q^\top \boldsymbol{W}\boldsymbol{x}_j)\epsilon_i\epsilon_j$$

Taking expectation:

$$\mathop{\mathbb{E}}_{\boldsymbol{\epsilon}}[z^2] = \sum_{i,j=1}^{m+n} (\boldsymbol{x}_i^\top \boldsymbol{W}^\top\boldsymbol{x}_q)(\boldsymbol{x}_q^\top \boldsymbol{W}\boldsymbol{x}_j) \cdot \mathbb{E}[\epsilon_i\epsilon_j]$$

Because the noise terms are independent and zero-mean, we have:

$$\mathbb{E}[\epsilon_i\epsilon_j] = \begin{cases} \sigma^2, & i = j \leqslant m \\ \sigma_{\text{rag}}^2, & i = j > m \\ 0, & i \neq j \end{cases}$$

So only the diagonal terms survive:

$$\mathbb{E}[z^2] = \sum_{i=1}^{m} \sigma^2 \cdot \mathbb{E}\left[(x_q^\top W x_i)^2\right] + \sum_{i=m+1}^{m+n} \sigma_{\text{rag}}^2 \cdot \mathbb{E}\left[(x_q^\top W (x_q + r_{i-m}))^2\right]$$

- **ICL Term:** Since $x_q, x_i \sim \mathcal{N}(0, I)$ and are independent,

$$\begin{aligned} \mathbb{E}[(x_q^\top W x_i)^2] &= \mathbb{E}\left[x_i^\top W^\top x_q x_q^\top W x_i\right] \\ &= \mathbb{E}\left[\text{tr}\left(W^\top x_q x_q^\top W x_i x_i^\top\right)\right] \\ &= \text{tr}(W^\top W) \end{aligned} \tag{D.25}$$

  where the first step follows from the cyclic property of trace, the last step follows from the symmetry of W.

$$\Rightarrow \quad \text{ICL contribution} = m \cdot \sigma^2 \cdot \text{tr}(W^\top W) \tag{D.26}$$

- **RAG Term:**

  Each row in RAG has the form $x_q + r_i$, so:

$$x_q^\top W (x_q + r_i) = x_q^\top W x_q + x_q^\top W r_i$$

  Then, we plug in Equation (D.16) into the RAG term:

$$\begin{aligned} \mathbb{E}\left[\left(x_q^\top W (x_q + r_i)\right)^2\right] &= \mathbb{E}[(x_q^\top W x_q)^2] + \mathbb{E}[(x_q^\top W r_i)^2] + 2\,\mathbb{E}[x_q^\top W x_q \cdot x_q^\top W r_i] \\ &= \mathbb{E}[(x_q^\top W x_q)^2] + \mathbb{E}[(x_q^\top W r_i)^2] \\ &= \mathbb{E}[(x_q^\top W x_q)^2] + \delta^2 \cdot \text{tr}(W^\top W) \\ &= \left[\text{tr}(W^\top W) + \text{tr}(W^2) + \text{tr}(W)^2\right] + \delta^2 \cdot \text{tr}(W^\top W) \\ &= \text{tr}(W^\top W) + \text{tr}(W^2) + \delta^2\,\text{tr}(W^\top W) + \text{tr}(W)^2 \end{aligned} \tag{D.27}$$

  where the second step follows from $\mathbb{E}[r_i] = 0$, the third step follows from the cyclic property of trace.

$$\Rightarrow \quad \text{RAG contribution} = n \cdot \sigma_{\text{rag}}^2 \cdot \left[(1+\delta^2)\,\text{tr}(\boldsymbol{W}^\top \boldsymbol{W}) + \text{tr}(\boldsymbol{W}^2) + \text{tr}(\boldsymbol{W})^2\right]$$

Thus, we can combine the two terms above and obtain the following:

$$\mathbb{E}\left[\left(\boldsymbol{x}_q^\top \boldsymbol{W}\boldsymbol{X}^\top \boldsymbol{\epsilon}\right)^2\right] = \left[m\sigma^2 + \left(1+\delta^2\right) n\sigma_{\text{rag}}^2\right] \text{tr}(\boldsymbol{W}^\top \boldsymbol{W}) + n\sigma_{\text{rag}}^2\,\text{tr}(\boldsymbol{W}^2) + n\sigma_{\text{rag}}^2\,\text{tr}(\boldsymbol{W})^2 \tag{D.28}$$

Compute $\mathbb{E}_{x_q,X}\left[(\boldsymbol{x}_q^\top (I - \boldsymbol{W}\boldsymbol{G})\,\beta_{tt})^2\right]$ First, we can expand the expectation and decompose the inner terms into 4 terms:

$$\begin{aligned}\mathop{\mathbb{E}}_{x_q,X}\left[(I-\boldsymbol{W}\boldsymbol{G})^\top \boldsymbol{x}_q \boldsymbol{x}_q^\top (I-\boldsymbol{W}\boldsymbol{G})\right] &= \mathop{\mathbb{E}}_{x_q,X}\left(I - \boldsymbol{G}\boldsymbol{W}^\top\right)\boldsymbol{x}_q\boldsymbol{x}_q^\top (I-\boldsymbol{W}\boldsymbol{G}) \\ &= \underbrace{\mathbb{E}\,\boldsymbol{x}_q\boldsymbol{x}_q^\top}_{:=M_1} - \underbrace{\mathbb{E}\,\boldsymbol{x}_q\boldsymbol{x}_q^\top \boldsymbol{W}\boldsymbol{G}}_{:=M_2} - \underbrace{\mathbb{E}\,\boldsymbol{G}\boldsymbol{W}^\top \boldsymbol{x}_q\boldsymbol{x}_q^\top}_{:=M_3} + \underbrace{\mathbb{E}\,\boldsymbol{G}\boldsymbol{W}^\top \boldsymbol{x}_q\boldsymbol{x}_q^\top \boldsymbol{W}\boldsymbol{G}}_{:=M_4}\end{aligned} \tag{D.29}$$

We denote the four pieces M_1, M_2, M_3, M_4 in order. First, we note that:

$$M_1 = \mathbb{E}\left[\boldsymbol{x}_q\boldsymbol{x}_q^\top\right] = I$$

Then, we expand out the terms in M_2:

$$
\begin{aligned}
\mathop{\mathbb{E}}_{\mathbf{x}_q,\mathbf{r}} \mathbf{x}_q\mathbf{x}_q^\top \boldsymbol{W}\mathbf{G} &= \left(\mathop{\mathbb{E}}_{\mathbf{x}_q,\mathbf{r}} \mathbf{x}_q\mathbf{x}_q^\top\right)\boldsymbol{W}\mathbf{G}_0 + \mathop{\mathbb{E}}_{\mathbf{x}_q,\mathbf{r}} \mathbf{x}_q\mathbf{x}_q^\top \boldsymbol{W}\sum_{i=1}^{n}(\mathbf{x}_q+\mathbf{r}_i)(\mathbf{x}_q+\mathbf{r}_i)^\top \\
&= \boldsymbol{W}\mathbf{G}_0 + \mathop{\mathbb{E}}_{\mathbf{x}_q,\mathbf{r}} \mathbf{x}_q\mathbf{x}_q^\top \boldsymbol{W}\sum_{i=1}^{n}(\mathbf{x}_q+\mathbf{r}_i)(\mathbf{x}_q+\mathbf{r}_i)^\top \\
&= \boldsymbol{W}\mathbf{G}_0 + \mathop{\mathbb{E}}_{\mathbf{x}_q,\mathbf{r}} \mathbf{x}_q\mathbf{x}_q^\top \boldsymbol{W}\sum_{i=1}^{n}(\mathbf{x}_q\mathbf{x}_q^\top+\mathbf{r}_i\mathbf{r}_i^\top) \\
&= \boldsymbol{W}\mathbf{G}_0 + \mathop{\mathbb{E}}_{\mathbf{x}_q,\mathbf{r}} \mathbf{x}_q\mathbf{x}_q^\top \boldsymbol{W}\sum_{i=1}^{n}(\mathbf{x}_q\mathbf{x}_q^\top+\delta^2\mathrm{I}) \\
&= \boldsymbol{W}\mathbf{G}_0 + n(\boldsymbol{W}+\boldsymbol{W}^\top+\operatorname{tr}(\boldsymbol{W})\mathrm{I}) + n\delta^2\boldsymbol{W} \\
&= \boldsymbol{W}\mathbf{G}_0 + n(1+\delta^2)\boldsymbol{W} + n\boldsymbol{W}^\top + n\operatorname{tr}(\boldsymbol{W})\mathrm{I}
\end{aligned}
\tag{D.30}
$$

where the first step follows from the independence between $\mathbf{X}$ and $\mathbf{x}_q$, the second step follows from $\mathbb{E}\,\mathbf{r}_i = 0,\ \forall i \in [n]$, the third step follows from the expectation of $\mathbf{r}_i\mathbf{r}_i^\top = \delta^2\mathrm{I}$, and the last step follows from Equation (D.14). Then,

$$
\begin{aligned}
M_2 &= \boldsymbol{W}\mathbf{G}_0 + n(1+\delta^2)\boldsymbol{W} + n\boldsymbol{W}^\top + n\operatorname{tr}(\boldsymbol{W})\mathrm{I} \\
&= (n\delta^2 + n + m)\boldsymbol{W} + n\boldsymbol{W}^\top + n\operatorname{tr}(\boldsymbol{W})\mathrm{I}
\end{aligned}
\tag{D.31}
$$

Similarly, $M_3 = M_2^\top = (n\delta^2 + n + m)\boldsymbol{W}^\top + n\boldsymbol{W} + n\operatorname{tr}(\boldsymbol{W})\mathrm{I}$. Now, we perform similar expansion for M_4:

$$
\begin{aligned}
M_4 &= \mathop{\mathbb{E}}_{x_q, X}[G W^\top x_q x_q^\top W G] \\
&= \mathop{\mathbb{E}}_{x_q, X}\left[\left(G_0 + \sum_{i\in[n]} G_i\right) W^\top x_q x_q^\top W \left(G_0 + \sum_{i\in[n]} G_i\right)\right] \\
&= \mathop{\mathbb{E}}_{x_q, X}\Bigg[G_0 W^\top x_q x_q^\top W G_0 + G_0 W^\top x_q x_q^\top W \sum_{i\in[n]} G_i + \sum_{i\in[n]} G_i W^\top x_q x_q^\top W G_0 \\
&\quad + \sum_{i\in n} G_i W^\top x_q x_q^\top W G_i + \sum_{i,j\in n, i\neq j} G_i W^\top x_q x_q^\top W G_j\Bigg] \\
&= \mathop{\mathbb{E}}_{x_q, X}\Bigg[G_0 W^\top x_q x_q^\top W G_0 + n \underbrace{G_0 W^\top x_q x_q^\top W G_i}_{i\in[n]} + n \underbrace{G_i W^\top x_q x_q^\top W G_0}_{i\in[n]} \\
&\quad + n \underbrace{G_i W^\top x_q x_q^\top W G_i}_{i\in[n]} + n(n-1) \underbrace{G_i W^\top x_q x_q^\top W G_j}_{i,j\in[n], i\neq j}\Bigg]
\end{aligned}
\tag{D.32}
$$

First, we can compute that:

$$
\begin{aligned}
M_{41} := \mathop{\mathbb{E}}_{x_q, X}[G_0 W^\top x_q x_q^\top W G_0] &= \mathop{\mathbb{E}}_{X}[G_0 W^\top W G_0] \\
&= m(m+1) W^\top W + m \operatorname{tr}(W^\top W) I
\end{aligned}
\tag{D.33}
$$

where the last line follows from Equation (D.15) and symmetry of $W^\top W$. Then, $\forall i \in [n]$, we have:

$$
\begin{aligned}
M_{42} := \mathop{\mathbb{E}}_{x_q, X} \mathbf{G}_0 \boldsymbol{W}^\top \boldsymbol{x}_q \boldsymbol{x}_q^\top \boldsymbol{W} \mathbf{G}_i &= \mathop{\mathbb{E}}_{x_q, X} \mathbf{G}_0 \boldsymbol{W}^\top \boldsymbol{x}_q \boldsymbol{x}_q^\top \boldsymbol{W} \left(\boldsymbol{x}_q + \mathbf{r}_i\right)\left(\boldsymbol{x}_q + \mathbf{r}_i\right)^\top \\
&= \mathop{\mathbb{E}}_{x_q, X} \mathbf{G}_0 \boldsymbol{W}^\top \boldsymbol{x}_q \boldsymbol{x}_q^\top \boldsymbol{W} \left(\boldsymbol{x}_q \boldsymbol{x}_q^\top + \mathbf{r}_i \mathbf{r}_i^\top\right) \\
&= \mathop{\mathbb{E}}_{x_q, X} \mathbf{G}_0 \boldsymbol{W}^\top \left(\boldsymbol{W} + \boldsymbol{W}^\top + \operatorname{tr}(\boldsymbol{W}) + \boldsymbol{W}\delta^2\right) \\
&= m \left(\boldsymbol{W}^\top \boldsymbol{W} + \boldsymbol{W}^\top \boldsymbol{W}^\top + \operatorname{tr}(\boldsymbol{W}) \boldsymbol{W}^\top + \delta^2 \boldsymbol{W}^\top \boldsymbol{W}\right) \\
&= m \left((1+\delta^2) \boldsymbol{W}^\top \boldsymbol{W} + \boldsymbol{W}^\top \boldsymbol{W}^\top + \operatorname{tr}(\boldsymbol{W}) \boldsymbol{W}^\top\right)
\end{aligned} \tag{D.34}
$$

where the first steps follows from $\mathbb{E}[\mathbf{r}_i] = 0$, the second step follows from Equation (D.14).

Moreover, we note that $\forall i \in [n]$:

$$
\begin{aligned}
M_{43} := \mathop{\mathbb{E}}_{x_q, X, \mathbf{r}_i} \mathbf{G}_i \boldsymbol{W}^\top \boldsymbol{x}_q \boldsymbol{x}_q^\top \boldsymbol{W} \mathbf{G}_i &= (\boldsymbol{x}_q + \mathbf{r}_i)(\boldsymbol{x}_q + \mathbf{r}_i)^\top \boldsymbol{W}^\top \boldsymbol{x}_q \boldsymbol{x}_q^\top \boldsymbol{W} (\boldsymbol{x}_q + \mathbf{r}_i)(\boldsymbol{x}_q + \mathbf{r}_i)^\top \\
&= (\boldsymbol{x}_q \boldsymbol{x}_q^\top + \mathbf{r}_i \boldsymbol{x}_q^\top + \boldsymbol{x}_q \mathbf{r}_i^\top + \mathbf{r}_i \mathbf{r}_i^\top) \boldsymbol{W}^\top \boldsymbol{x}_q \boldsymbol{x}_q^\top \boldsymbol{W} (\boldsymbol{x}_q \boldsymbol{x}_q^\top + \mathbf{r}_i \boldsymbol{x}_q^\top + \boldsymbol{x}_q \mathbf{r}_i^\top + \\
&= \underbrace{\boldsymbol{x}_q \boldsymbol{x}_q^\top \boldsymbol{W}^\top \boldsymbol{x}_q \boldsymbol{x}_q^\top \boldsymbol{W} \boldsymbol{x}_q \boldsymbol{x}_q^\top}_{\text{0 order in } \mathbf{r}_i} + \underbrace{\mathbf{r}_i \mathbf{r}_i^\top \boldsymbol{W}^\top \boldsymbol{x}_q \boldsymbol{x}_q^\top \boldsymbol{W} \mathbf{r}_i \mathbf{r}_i^\top}_{\text{4th-order in } \mathbf{r}_i} \\
&+ \underbrace{(\mathbf{r}_i \boldsymbol{x}_q^\top + \boldsymbol{x}_q \mathbf{r}_i^\top) \boldsymbol{W}^\top \boldsymbol{x}_q \boldsymbol{x}_q^\top \boldsymbol{W} (\mathbf{r}_i \boldsymbol{x}_q^\top + \boldsymbol{x}_q \mathbf{r}_i^\top)}_{\text{2nd-order in } \mathbf{r}_i} \\
&+ \underbrace{\mathbf{r}_i \mathbf{r}_i^\top \boldsymbol{W}^\top \boldsymbol{x}_q \boldsymbol{x}_q^\top \boldsymbol{W} \boldsymbol{x}_q \boldsymbol{x}_q^\top + \boldsymbol{x}_q \boldsymbol{x}_q^\top \boldsymbol{W}^\top \boldsymbol{x}_q \boldsymbol{x}_q^\top \boldsymbol{W} \mathbf{r}_i \mathbf{r}_i^\top}_{\text{2nd-order in } \mathbf{r}_i}
\end{aligned} \tag{D.35}
$$

It worth noting that given Gaussian vector $\mathbf{r}_i$, then its monomial of odd order has 0 expectation according to Isserlis' Theorem (Isserlis, 1918). And we can thus obtain the third line by keeping only the even order monomials of $\mathbf{r}_i$.

By adding up Theorem D.3 and all the terms above, we obtain that:

$$
\begin{aligned}
&\mathop{\mathbb{E}}_{\mathbf{x}_q,\mathbf{X},\mathbf{r}_i} \mathbf{G}_i \boldsymbol{W}^\top \mathbf{x}_q \mathbf{x}_q^\top \boldsymbol{W} \mathbf{G}_i \\
&= \left.\begin{array}{l} 2\left(\boldsymbol{W}^2 + (\boldsymbol{W}^2)^\top + \boldsymbol{W}^\top\boldsymbol{W} + \boldsymbol{W}\boldsymbol{W}^\top + \operatorname{tr}(\boldsymbol{W})(\boldsymbol{W}+\boldsymbol{W}^\top)\right) \\ + \operatorname{tr}(\boldsymbol{W})^2 \mathrm{I} + \operatorname{tr}\left(\boldsymbol{W}^2\right)\mathrm{I} + \operatorname{tr}\left(\boldsymbol{W}^\top\boldsymbol{W}\right)\mathrm{I} \end{array}\right\} \text{0th-order in } \mathbf{r}_i\text{, Theorem D.3} \\
&\quad + \underbrace{2\delta^4 \boldsymbol{W}^\top\boldsymbol{W} + \delta^4 \operatorname{tr}(\boldsymbol{W}^\top\boldsymbol{W})\mathrm{I}}_{\text{4th-order in } \mathbf{r}_i\text{, Equation (D.1)}} \\
&\quad + \underbrace{\delta^2\left[\operatorname{tr}(\boldsymbol{W})\left(\boldsymbol{W}^\top + \boldsymbol{W}\right) + \boldsymbol{W}^2 + (\boldsymbol{W}^2)^\top + 2\boldsymbol{W}^\top\boldsymbol{W}\right]}_{\text{Equation (D.5) and its transpose}} \\
&\quad + \underbrace{\left(\operatorname{tr}\left(\boldsymbol{W}^2\right) + \operatorname{tr}\left(\boldsymbol{W}^\top\boldsymbol{W}\right) + \operatorname{tr}(\boldsymbol{W})^2\right)\delta^2\mathrm{I}}_{\text{Equation (D.2)}} \\
&\quad + \underbrace{2\delta^2\boldsymbol{W}\boldsymbol{W}^\top + \delta^2\operatorname{tr}(\boldsymbol{W}^\top\boldsymbol{W})\mathrm{I}}_{\text{Equation (D.3)}} \\
&\quad + \underbrace{\delta^2\left[\operatorname{tr}(\boldsymbol{W})\left(\boldsymbol{W}^\top + \boldsymbol{W}\right) + \boldsymbol{W}^2 + (\boldsymbol{W}^2)^\top + 2\boldsymbol{W}^\top\boldsymbol{W}\right]}_{\text{Equation (D.4) and its transpose}} \\
&= (2+2\delta^2)\left[\operatorname{tr}(\boldsymbol{W})\left(\boldsymbol{W}^\top + \boldsymbol{W}\right) + \boldsymbol{W}^2 + (\boldsymbol{W}^2)^\top\right] \\
&\quad + (2+4\delta^2)\boldsymbol{W}^\top\boldsymbol{W} + 2\boldsymbol{W}\boldsymbol{W}^\top \\
&\quad + (1+\delta^2)\left[\operatorname{tr}(\boldsymbol{W})^2\mathrm{I} + \operatorname{tr}\left(\boldsymbol{W}^2\right)\mathrm{I} + \operatorname{tr}\left(\boldsymbol{W}^\top\boldsymbol{W}\right)\mathrm{I}\right] \\
&\quad + 2\delta^4\boldsymbol{W}^\top\boldsymbol{W} + \delta^4\operatorname{tr}(\boldsymbol{W}^\top\boldsymbol{W})\mathrm{I} + 2\delta^2\boldsymbol{W}\boldsymbol{W}^\top + \delta^2\operatorname{tr}(\boldsymbol{W}^\top\boldsymbol{W})\mathrm{I} \\
&= (2+2\delta^2)\left[\operatorname{tr}(\boldsymbol{W})\left(\boldsymbol{W}^\top + \boldsymbol{W}\right) + \boldsymbol{W}^2 + (\boldsymbol{W}^2)^\top\right] \\
&\quad + (2+4\delta^2+2\delta^4)\boldsymbol{W}^\top\boldsymbol{W} + (2+2\delta^2)\boldsymbol{W}\boldsymbol{W}^\top \\
&\quad + (1+\delta^2)\left(\operatorname{tr}(\boldsymbol{W})^2 + \operatorname{tr}\left(\boldsymbol{W}^2\right)\right)\mathrm{I} + (1+2\delta^2+\delta^4)\operatorname{tr}(\boldsymbol{W}^\top\boldsymbol{W})\mathrm{I}
\end{aligned}
\tag{D.36}
$$

Also, we expand the cross-term out for $\forall i, j \in [n], i \neq j$:

$$
\begin{aligned}
M_{44} := \mathbb{E}\, \mathbf{G}_i \boldsymbol{W}^\top \mathbf{x}_q \mathbf{x}_q^\top \boldsymbol{W} \mathbf{G}_j &= \mathbb{E}(\mathbf{x}_q + \mathbf{r}_i)(\mathbf{x}_q + \mathbf{r}_i)^\top \boldsymbol{W}^\top \mathbf{x}_q \mathbf{x}_q^\top \boldsymbol{W} (\mathbf{x}_q + \mathbf{r}_j)(\mathbf{x}_q + \mathbf{r}_j)^\top \\
&= \left(\mathbf{x}_q \mathbf{x}_q^\top + \mathbf{r}_i \mathbf{r}_i^\top\right) \boldsymbol{W}^\top \mathbf{x}_q \mathbf{x}_q^\top \boldsymbol{W} \left(\mathbf{x}_q \mathbf{x}_q^\top + \mathbf{r}_j \mathbf{r}_j^\top\right) \\
&= \mathbf{x}_q \mathbf{x}_q^\top \boldsymbol{W}^\top \mathbf{x}_q \mathbf{x}_q^\top \boldsymbol{W} \mathbf{x}_q \mathbf{x}_q^\top + \mathbf{r}_i \mathbf{r}_i^\top \boldsymbol{W}^\top \mathbf{x}_q \mathbf{x}_q^\top \boldsymbol{W} \mathbf{x}_q \mathbf{x}_q^\top \\
&\quad + \mathbf{x}_q \mathbf{x}_q^\top \boldsymbol{W}^\top \mathbf{x}_q \mathbf{x}_q^\top \boldsymbol{W} \mathbf{r}_j \mathbf{r}_j^\top + \mathbf{r}_i \mathbf{r}_i^\top \boldsymbol{W}^\top \mathbf{x}_q \mathbf{x}_q^\top \boldsymbol{W} \mathbf{r}_j \mathbf{r}_j^\top \\
&= 2\left(\boldsymbol{W}^2 + \left(\boldsymbol{W}^2\right)^\top + \boldsymbol{W}^\top \boldsymbol{W} + \boldsymbol{W}\boldsymbol{W}^\top + \operatorname{tr}(\boldsymbol{W})\boldsymbol{W} + \operatorname{tr}(\boldsymbol{W})\boldsymbol{W}^\top\right) \\
&\quad + \operatorname{tr}(\boldsymbol{W})^2 \mathrm{I} + \operatorname{tr}\left(\boldsymbol{W}^2\right)\mathrm{I} + \operatorname{tr}\left(\boldsymbol{W}^\top \boldsymbol{W}\right)\mathrm{I} \\
&\quad + \delta^2\left(\boldsymbol{W}^2 + (\boldsymbol{W}^2)^\top + 2\boldsymbol{W}^\top \boldsymbol{W}\right) + \operatorname{tr}(\boldsymbol{W})(\boldsymbol{W} + \boldsymbol{W}^\top) + \delta^4 \boldsymbol{W}^\top \boldsymbol{W} \\
&= (2 + \delta^2)\left(\boldsymbol{W}^2 + \left(\boldsymbol{W}^2\right)^\top + \operatorname{tr}(\boldsymbol{W})\boldsymbol{W} + \operatorname{tr}(\boldsymbol{W})\boldsymbol{W}^\top\right) \\
&\quad + (2 + 2\delta^2)\boldsymbol{W}^\top \boldsymbol{W} + 2\boldsymbol{W}\boldsymbol{W}^\top \\
&\quad + \operatorname{tr}(\boldsymbol{W})^2 \mathrm{I} + \operatorname{tr}\left(\boldsymbol{W}^2\right)\mathrm{I} + \operatorname{tr}\left(\boldsymbol{W}^\top \boldsymbol{W}\right)\mathrm{I} + \delta^4 \boldsymbol{W}^\top \boldsymbol{W} \\
&= (2 + \delta^2)\left(\boldsymbol{W}^2 + \left(\boldsymbol{W}^2\right)^\top + \operatorname{tr}(\boldsymbol{W})\boldsymbol{W} + \operatorname{tr}(\boldsymbol{W})\boldsymbol{W}^\top\right) \\
&\quad + (2 + 2\delta^2 + \delta^4)\boldsymbol{W}^\top \boldsymbol{W} + 2\boldsymbol{W}\boldsymbol{W}^\top \\
&\quad + \operatorname{tr}(\boldsymbol{W})^2 \mathrm{I} + \operatorname{tr}\left(\boldsymbol{W}^2\right)\mathrm{I} + \operatorname{tr}\left(\boldsymbol{W}^\top \boldsymbol{W}\right)\mathrm{I}
\end{aligned}
\tag{D.37}
$$

where the first step follows from the independence of $\mathbf{x}_q, \mathbf{r}_i, \mathbf{r}_j$, and the second step follows from applying Theorem D.3 and Equation (D.14).

Combining the above terms together, we have

$$
\begin{aligned}
M_4 &= M_{41} + n(M_{42} + M_{42}^\top) + nM_{43} + n(n-1)M_{44} \\
&= m(m+1)\boldsymbol{W}^\top\boldsymbol{W} + m\,\mathrm{tr}(\boldsymbol{W}^\top\boldsymbol{W})\mathrm{I} + mn\left(\left(2+2\delta^2\right)\boldsymbol{W}^\top\boldsymbol{W} + \boldsymbol{W}^2 + (\boldsymbol{W}^2)^\top + \mathrm{tr}\left(\boldsymbol{W}\right)\left(\boldsymbol{W}+\boldsymbol{W}^\top\right)\right. \\
&\quad + nM_{43} + n(n-1)M_{44} \\
&= 2n\left(2n+\delta^2\right)\boldsymbol{W}^2 + 2n\left(n+\delta^2\right)\boldsymbol{W}\boldsymbol{W}^\top \\
&\quad + \left[m^2 + m + (4+2\delta^2)mn + n^2\left(2+4\delta^2+\delta^4\right) + n\left(2\delta^2+\delta^4\right)\right]\boldsymbol{W}^\top\boldsymbol{W} \\
&\quad + \left[n^2\left(2+\delta^2\right) + n\left(m+\delta^2\right)\right]\mathrm{tr}(\boldsymbol{W})\left(\boldsymbol{W}+\boldsymbol{W}^\top\right) \\
&\quad + \left(n^2+n\delta^2\right)\left(\mathrm{tr}(\boldsymbol{W})^2 + \mathrm{tr}(\boldsymbol{W}^2)\right)\mathrm{I} + \left[m+n^2+n\left(2\delta^2+\delta^4\right)\right]\mathrm{tr}\left(\boldsymbol{W}^\top\boldsymbol{W}\right)\mathrm{I} \\
&= \left[n^2\left(2+\delta^2\right) + n\left(m+\delta^2\right)\right]\left(\boldsymbol{W}^2 + \left(\boldsymbol{W}^2\right)^\top + \mathrm{tr}(\boldsymbol{W})\left(\boldsymbol{W}+\boldsymbol{W}^\top\right)\right) \\
&\quad + \left[2n^2 + 2n\delta^2\right]\boldsymbol{W}\boldsymbol{W}^\top \\
&\quad + \left[m^2 + m + mn\left(2+2\delta^2\right) + n\left(2\delta^2+\delta^4\right) + n^2\left(2+2\delta^2+\delta^4\right)\right]\boldsymbol{W}^\top\boldsymbol{W} \\
&\quad + \left[n^2+n\delta^2\right]\left(\mathrm{tr}(\boldsymbol{W})^2 + \mathrm{tr}\left(\boldsymbol{W}^2\right)\right)\mathrm{I} \\
&\quad + \left[m+n^2+n\left(2\delta^2+\delta^4\right)\right]\mathrm{tr}\left(\boldsymbol{W}^\top\boldsymbol{W}\right)\mathrm{I}
\end{aligned}
\tag{D.38}
$$

In summary, combining all terms together, we have:

$$\mathcal{L}(\boldsymbol{W}) := \underset{\text{variance}}{\mathrm{err}} + \underset{\text{bias}}{\mathrm{err}} + \sigma^2$$

where the *irreducible variance* is σ^2, and the *reducible variance* (variance of ICL + RAG) is

$$\text{Variance of ICL+Variance of RAG} = \left[m\sigma^2 + \left(1+\delta^2\right)n\sigma_{\text{rag}}^2\right]\mathrm{tr}(\boldsymbol{W}^\top\boldsymbol{W}) + n\sigma_{\text{rag}}^2\,\mathrm{tr}(\boldsymbol{W}^2) + n\sigma_{\text{rag}}^2\,\mathrm{tr}(\boldsymbol{W})^2$$

And the err from the bias term is given as:

$$
\begin{aligned}
\underset{\text{bias}}{\text{err}} &= \beta_{tt}^{\top}[M_1 - M_2 - M_3 + M_4]\beta_{tt} \\
&= \beta_{tt}^{\top}\left[I - (n\delta^2 + 2n + m)(\boldsymbol{W} + \boldsymbol{W}^{\top}) - 2n\,\mathrm{tr}(\boldsymbol{W})I + M_4\right]\beta_{tt} \\
&= \beta_{tt}^{\top}\Big[I - (n\delta^2 + 2n + m)(\boldsymbol{W} + \boldsymbol{W}^{\top}) - 2n\,\mathrm{tr}(\boldsymbol{W})I \\
&\quad + \left[n^2\left(2+\delta^2\right) + n\left(m+\delta^2\right)\right]\left(\boldsymbol{W}^2 + \left(\boldsymbol{W}^2\right)^{\top}\right) + 2n(n+\delta^2)\boldsymbol{W}\boldsymbol{W}^{\top} \\
&\quad + \left[m^2 + m + mn\left(2+2\delta^2\right) + n^2\left(2+2\delta^2+\delta^4\right) + n\left(2\delta^2+\delta^4\right)\right]\boldsymbol{W}^{\top}\boldsymbol{W} \\
&\quad + \left[n^2\left(2+\delta^2\right) + n\left(m+\delta^2\right)\right]\left(\mathrm{tr}(\boldsymbol{W})\left(\boldsymbol{W} + \boldsymbol{W}^{\top}\right)\right) \\
&\quad + \left[n^2 + n\delta^2\right]\left(\mathrm{tr}(\boldsymbol{W})^2 + \mathrm{tr}\left(\boldsymbol{W}^2\right)\right)I + \left[m + n^2 + n\left(2\delta^2+\delta^4\right)\right]\mathrm{tr}\left(\boldsymbol{W}^{\top}\boldsymbol{W}\right)I\Big]\beta_{tt}
\end{aligned}
$$

□

The previous theorem gives the exact form the RAG population with general $\boldsymbol{W}$. In the following proposition, we will compute the population under special $\boldsymbol{W}$ in order to obtain a more fine-grained complexity analysis.

Proposition 3 (RAG Population loss under isotropic setting)**.** *Assuming* $\boldsymbol{W}^* = \frac{m}{(m+d+1)(m+n)}I$. *Then, the population loss are given as:*

$$
\begin{aligned}
\mathcal{L}_{tt+rag}(\boldsymbol{W}^*) &= \underset{variance}{\text{err}}\,(\boldsymbol{W}^*) + \underset{bias}{\text{err}}(\boldsymbol{W}^*) + \sigma^2 \\
\underset{variance}{\text{err}}\,(\boldsymbol{W}^*) &= \frac{m^3 d}{[(m+d+1)(m+n)]^2}\sigma^2 + \frac{dm^2n(2+\delta^2+d)}{[(m+d+1)(m+n)]^2}\sigma_{\text{rag}}^2 \\
\underset{bias}{\text{err}}(\boldsymbol{W}^*) &= \|\beta_{TT}\|_2^2\left[1 - \frac{2m}{(m+d+1)(m+n)}\left(n\delta^2 + 2n + m + nd\right) + \frac{P(m,n,d,\delta)m^2}{(m+d+1)^2(m+n)^2}\right]
\end{aligned}
$$

where

$$
\begin{aligned}
P(m,n,d,\delta) =& 6n^2 + 4n\delta^2 + m^2 + m + \left(4+2\delta^2\right)mn \\
&+ n^2\left(2+4\delta^2+\delta^4\right) + n\left(2\delta^2+\delta^4\right) + 2dn^2\left(2+\delta^2\right) + 2dn\left(m+\delta^2\right) \\
&+ d(d+1)\left(n^2+n\delta^2\right) + dm + dn^2 + dn\left(2\delta^2+\delta^4\right)
\end{aligned}
$$

Proof. Plugging in the value of $\boldsymbol{W}^*$, we first compute the error from input variance.

$$\begin{aligned}\operatorname{tr}((\boldsymbol{W}^*)^2) &= \frac{dm^2}{(m+d+1)^2(m+n)^2}\\ \operatorname{tr}(\boldsymbol{W}^*) &= \frac{dm}{(m+d+1)(m+n)}\end{aligned}$$

$$\begin{aligned}\underset{\text{variance}}{\operatorname{err}}(\boldsymbol{W}^*) &= \left[m\sigma^2 + \left(1+\delta^2\right)n\sigma_{rag}^2\right]\operatorname{tr}\left(\boldsymbol{W}^\top\boldsymbol{W}\right) + n\sigma_{rag}^2\operatorname{tr}\left(\boldsymbol{W}^2\right) + n\sigma_{rag}^2\operatorname{tr}(\boldsymbol{W})^2\\ &= \frac{dm^2[m\sigma^2 + (1+\delta^2)n\sigma_{rag}^2]}{(m+d+1)^2(m+n)^2} + n\sigma_{rag}^2\frac{dm^2}{(m+d+1)^2(m+n)^2} + n\sigma_{rag}^2\frac{d^2m^2}{(m+d+1)^2(m+n)^2}\\ &= \frac{m^3d}{[(m+d+1)(m+n)]^2}\sigma^2 + \frac{dm^2n(2+\delta^2+d)}{[(m+d+1)(m+n)]^2}\sigma_{rag}^2\end{aligned}$$

Then, we proceed to plug in the value and compute the error from the estimation bias.

$$\begin{aligned}\underset{\text{bias}}{\operatorname{err}}(\boldsymbol{W}^*) &= \|\beta_{tt}\|_2^2\left[1 - \frac{2m(n\delta^2+2n+m)}{(m+n)(m+d+1)} - \frac{2ndm}{(m+n)(m+d+1)} + \frac{m^2}{(m+d+1)^2(m+n)^2}\underbrace{(\dots)}_{P(m,n,d,\delta}\right.\\ &= \|\beta_{TT}\|_2^2\left[1 - \frac{2m}{(m+d+1)(m+n)}\left(n\delta^2+2n+m+nd\right) + \frac{P(m,n,d,\delta)m^2}{(m+d+1)^2(m+n)^2}\right]\end{aligned}$$

where

$$\begin{aligned}P(m,n,d,\delta) &= (2c_1+c_2+c_3) + 2dc_4 + (d^2+d)c_5 + dc_6\\ &= 2\left(n^2\left(2+\delta^2\right) + n\left(m+\delta^2\right)\right) + 2n(n+\delta^2)\\ &\quad + \left[m^2 + m + mn\left(2+2\delta^2\right) + n^2\left(2+2\delta^2+\delta^4\right) + n\left(2\delta^2+\delta^4\right)\right]\\ &\quad + 2d[n^2(2+\delta^2) + n(m+\delta^2)] + (d^2+d)(n^2+n\delta^2) + d[m+n^2+n(2\delta^2+\delta^4)]\\ &= 6n^2 + 4n\delta^2 + m^2 + m + \left(4+2\delta^2\right)mn\\ &\quad + n^2\left(2+4\delta^2+\delta^4\right) + n\left(2\delta^2+\delta^4\right) + 2dn^2\left(2+\delta^2\right) + 2dn\left(m+\delta^2\right)\\ &\quad + d(d+1)\left(n^2+n\delta^2\right) + dm + dn^2 + dn\left(2\delta^2+\delta^4\right)\end{aligned}$$

□

Finite Sample Complexity of RAG

Proposition (Restatement of Proposition 1)**.** *Under Assumption 1, 2, 3, if* $\delta^2 \ll 1$,

$$\mathcal{L}_{tt+rag}(\boldsymbol{W}^*) = \mathcal{O}\left(\sigma^2 + \underbrace{\frac{dm}{(m+n)^2}\sigma^2 + \frac{d^2 n}{(m+n)^2}\sigma_{rag}^2}_{\mathrm{err}_{variance}(\boldsymbol{W}^*)} + \underbrace{\|\beta_{tt}\|_2^2\left[\frac{d}{m} + d^2\left(\frac{n}{m+n}\right)^2\right]}_{\mathrm{err}_{bias}(\boldsymbol{W}^*)}\right)$$

$$\underset{variance}{\mathrm{err}}(\boldsymbol{W}^*) = \begin{cases} \mathcal{O}(\frac{d}{m}\sigma^2 + \frac{d^2}{m^2}\sigma_{rag}^2) = \mathcal{O}\left(\frac{1}{m}\right) & m \to \infty, n \textit{ fixed.} \\ \mathcal{O}(\frac{d}{n^2}\sigma^2 + \frac{d^2}{n}\sigma_{rag}^2) = \mathcal{O}\left(\frac{1}{n}\right) & n \to \infty, m \textit{ fixed} \\ \mathcal{O}(\frac{d}{m}\sigma^2 + \frac{d^2}{m}\sigma_{rag}^2) = \mathcal{O}\left(\frac{1}{m}\right) & m, n \to \infty, n = \Theta(m) \end{cases} \tag{D.39}$$

$$\underset{bias}{\mathrm{err}}(\boldsymbol{W}^*) = \begin{cases} \mathcal{O}\left(\|\beta_{tt}\|_2^2 \frac{d}{m}\right) & \textit{if } m \to \infty, n \textit{ is fixed} \\ \mathcal{O}\left(\|\beta_{tt}\|_2^2 d^2\right) = C_1 & \textit{if } n \to \infty, m \textit{ is fixed} \\ \mathcal{O}\left(\|\beta_{tt}\|_2^2\left(\frac{d}{m} + d^2\right)\right) = C_2 + \mathcal{O}(\|\beta_{tt}\|_2^2 \frac{d}{m}) & \textit{if } m \to \infty, n = \Theta(m) \end{cases} \tag{D.40}$$

Proof. We will bound the variance-induced error and the bias-induced error separately.

Variance-Induced Error First, we try to bound $\text{err}_{\text{variance}}(\boldsymbol{W}^*)$:

$$\begin{aligned}
\underset{\text{variance}}{\text{err}}(\boldsymbol{W}^*) &= \frac{dm^3}{(m+d+1)^2(m+n)^2}\sigma^2 + \frac{dm^2n(2+\delta^2+d)}{(m+d+1)^2(m+n)^2}\sigma_{\text{rag}}^2 \\
&\leqslant \frac{dm^3}{m^2(m+n)^2}\sigma^2 + \frac{dm^2n(d+\delta^2+2)}{m^2(m+n)^2}\sigma_{\text{rag}}^2 \\
&= \frac{dm}{(m+n)^2}\sigma^2 + \frac{d(2+\delta^2+d)n}{(m+n)^2}\sigma_{\text{rag}}^2 \\
&= \mathcal{O}\left(\frac{dm}{(m+n)^2}\sigma^2 + \frac{d^2n}{(m+n)^2}\sigma_{\text{rag}}^2\right) \\
&= \begin{cases} \mathcal{O}(\frac{d}{m}\sigma^2 + \frac{d^2}{m^2}\sigma_{\text{rag}}^2) = \mathcal{O}\left(\frac{1}{m}\right) & m \to \infty, n \text{ fixed.} \\ \mathcal{O}(\frac{d}{n^2}\sigma^2 + \frac{d^2}{n}\sigma_{\text{rag}}^2) = \mathcal{O}\left(\frac{1}{n}\right) & n \to \infty, m \text{ fixed} \\ \mathcal{O}(\frac{d}{m}\sigma^2 + \frac{d^2}{m}\sigma_{\text{rag}}^2) = \mathcal{O}\left(\frac{1}{m}\right) & m, n \to \infty, n = \Theta(m) \end{cases}
\end{aligned} \tag{D.41}$$

where the second line follows from $(m + d + 1) \geqslant m$ and the fourth line follows from the fact that δ^2 is small relative to m, n, d.

Bias-Induced Error We will expand out the term

$$\underset{\text{bias}}{\text{err}}(\boldsymbol{W}^*) = \|\beta_{\text{tt}}\|_2^2 \frac{Q(m, n; d, \delta^2)}{(m+d+1)^2(m+n)^2} \tag{D.42}$$

where

$$\begin{aligned}
Q(m,n;d,\delta^2) &:= (m+n)^2(m+d+1)^2 - 2m(m+n)(m+d+1)(n\delta^2+2n+m+nd) + m^2P(m,n,\\
&= (d+1)m^3 + \underbrace{(d^2+2d\delta^2+4d+\delta^4+2\delta^2+5)}_{:=\kappa_{22}} m^2n^2\\
&\quad + \underbrace{(d^2\delta^2-2d^2+d\delta^4+3d\delta^2-4d+\delta^4+4\delta^2-2)}_{:=\kappa_{21}} m^2n\\
&\quad \underbrace{-(2d^2+2d\delta^2+4d+2\delta^2+2)}_{:=\kappa_{12}} mn^2 + (d^2+2d+1)(m+n)^2\\
&= (d+1)m^3 + \kappa_{22}m^2n^2 + |\kappa_{21}|m^2n + \text{lower-order terms}\\
&\leqslant (d+1)m^3 + \kappa_{22}m^2n^2 + |\kappa_{21}|m^2n + (d+1)^2(m+n)^2
\end{aligned} \tag{D.43}$$

where the last line follows from $\kappa_{12} < 0$.

Note that we assume $\delta^2 \ll 1$. Now, we can bound each of the term in Q divided individually:

- Cubic term:
$$\frac{(d+1)m^3}{m^2(m+n)^2} = \frac{d+1}{m}\left(\frac{m}{m+n}\right)^2 \leqslant \frac{d+1}{m} \tag{D.44}$$

- Skew-cubic term:
$$\frac{|\kappa_{21}|\,m^2n}{m^2(m+n)^2} = |\kappa_{21}|\,\frac{n}{(m+n)^2} \leqslant |\kappa_{21}|\,\frac{n}{(m+n)^2} \tag{D.45}$$

- Quartic term:
$$\frac{\kappa_{22}m^2n^2}{m^2(m+n)^2} = \kappa_{22}\left(\frac{n}{m+n}\right)^2 \tag{D.46}$$

- last term:
$$(d+1)^2(m+n)^2\frac{1}{m^2(m+n)^2} = \frac{d^2}{m^2}$$

Combining Equation (D.43), Equation (D.44), Equation (D.45), Equation (D.46),

we can obtain that

$$\begin{aligned}\underset{\text{bias}}{\text{err}}(\boldsymbol{W}^*) &= \mathcal{O}\left(\|\beta_{\text{tt}}\|_2^2\left[\frac{dm}{(m+n)^2}+d^2\frac{n^2}{(m+n)^2}+\frac{d^2}{m^2}\right]\right)\\ &= \begin{cases}\mathcal{O}\left(\|\beta_{\text{tt}}\|_2^2\frac{d}{m}\right) & \text{if } m\to\infty, n \text{ is fixed}\\ \mathcal{O}\left(\|\beta_{\text{tt}}\|_2^2 d^2\right)=C_1 & \text{if } n\to\infty, m \text{ is fixed}\\ \mathcal{O}\left(\|\beta_{\text{tt}}\|_2^2\left(\frac{d}{m}+d^2\right)\right)=C_2+\mathcal{O}(\|\beta_{\text{tt}}\|_2^2\frac{d}{m}) & \text{if } m\to\infty, n=\Theta(m)\end{cases}\end{aligned} \tag{D.47}$$

where the third step follows from plugging in the highest order monomial of d from κ_{21}, κ_{22}.

□

Optimality of Number of RAG Examples

Proposition (Restatement of Proposition 2)**.** *Under Under Assumption 1,2,3, $\delta^2 \ll 1$, and reasonable choice of σ^2, σ^2_{rag} ($\sigma^2, \sigma^2_{rag} \ll \|\beta_{\text{tt}}\|_2^2$), the optimal n^* that minimizes the RAG loss follows:*

$$n^* = \mathcal{O}\left(\frac{m\left(d^2\|\beta_{\text{tt}}\|_2^2+d\sigma^2-d^2\sigma^2_{rag}\right)}{md^2\|\beta_{\text{tt}}\|_2^2-d^2\sigma^2_{rag}}\right)=\mathcal{O}\left(\frac{d\|\beta_{\text{tt}}\|_2^2+\sigma^2-d\sigma^2_{\text{rag}}}{d\|\beta_{\text{tt}}\|_2^2}\right) \tag{D.48}$$

and the improvement on loss from picking the optimal n^* over $n=0$ is given as:

$$\mathcal{L}_{tt+rag}(\boldsymbol{W}^*)|_{n=0}-\mathcal{L}_{tt+rag}(\boldsymbol{W}^*)|_{n=n^*}=\mathcal{O}\left(\frac{1}{m^2}\right) \tag{D.49}$$

Proof. First, we define several constants that can lead to a cleaner calculation. Let $\omega_1 := d$, $\omega_2 := d^2$. Then,

$$\begin{aligned}\underset{\text{variance}}{\text{err}}(\boldsymbol{W}^*) &= \frac{dm^3}{(m+d+1)^2(m+n)^2}\sigma^2+\frac{dm^2n(2+\delta^2+d)}{(m+d+1)^2(m+n)^2}\sigma^2_{\text{rag}}\\ &\approx \frac{m^3}{(m+d+1)^2(m+n)^2}\omega_1\sigma^2+\frac{m^2n}{(m+d+1)^2(m+n)^2}\omega_2\sigma^2_{\text{rag}}\end{aligned}$$

where the last line follows from $\delta^2 \ll 1$. Let $Q(m, n, d, \delta^2) := \frac{\mathrm{err}_{\mathrm{bias}}(\boldsymbol{W}^*)(m+d+1)^2(m+n)^2}{\|\beta_{tt}\|_2^2}$ as in Equation (D.43). Then,

$$\begin{aligned}
Q(m, n; d, \delta^2) &= (m+n)^2(m+d+1)^2 - 2m(m+n)(m+d+1)(n\delta^2 + 2n + m + nd) + m^2 P(m, n, \\
&= (d+1)m^3 + (d^2 + 2d\delta^2 + 4d + \delta^4 + 2\delta^2 + 5)m^2 n^2 \\
&\quad + (d^2\delta^2 - 2d^2 + d\delta^4 + 3d\delta^2 - 4d + \delta^4 + 4\delta^2 - 2)m^2 n \\
&\quad - \left(2d^2 + 2d\delta^2 + 4d + 2\delta^2 + 2\right) mn^2 + (d^2 + 2d + 1)(m^2 + n^2) \\
&\approx \underbrace{d}_{:=\tau_{30}} m^3 + \underbrace{d^2}_{\tau_{22}} m^2 n^2 \underbrace{-2d^2}_{:=\tau_{21}} m^2 n \underbrace{-2d^2}_{:=\tau_{12}} mn^2 + \underbrace{d^2}_{:=\tau_2}(m^2 + n^2) \\
&= \tau_{30} m^3 + \tau_{22} m^2 n^2 + \tau_{21} m^2 n + \tau_{12} mn^2 + \tau_2 (m^2 + n^2)
\end{aligned} \tag{D.50}$$

Now, we want to find the optimal n^* w.r.t. $\mathcal{L}_{tt+rag}$. That is, we want to minimize

$$\left[m^3\omega_1\sigma^2 + m^2 n\omega_2\sigma_{rag}^2 + \|\beta_{tt}\|_2^2 \left(\tau_{30}m^3 + \tau_{22}m^2n^2 + \tau_{21}m^2 n + \tau_{12}mn^2 + \tau_2\left(m^2+n^2\right)\right)\right] \frac{1}{(m+n)^2(m} \tag{D.51}$$

where all τ, ω are positive except that τ_{12} is negative. First, we take out the terms that does not depend on n, and we equivalently minimize

$$L(n) := \left[m^3\omega_1\sigma^2 + m^2 n\omega_2\sigma_{rag}^2 + \|\beta_{tt}\|_2^2 \left(\tau_{30}m^3 + \tau_{22}m^2n^2 + \tau_{21}m^2 n + \tau_{12}mn^2 + \tau_2\left(m^2+n^2\right)\right)\right] \frac{}{(m-}$$

Let

$$\begin{aligned}
A &= m^3\omega_1\sigma^2 + \|\beta_{tt}\|^2\tau_{30}m^3 + \|\beta_{tt}\|^2\tau_2 m^2, \\
B &= m^2\left(\omega_2\sigma_{rag}^2 + \|\beta_{tt}\|^2\tau_{21}\right), \\
C &= \|\beta_{tt}\|^2\left(\tau_{22}m^2 + \tau_{12}m + \tau_2\right).
\end{aligned} \tag{D.52}$$

Then,

$$L(n) = (A + Bn + Cn^2)/(m+n)^2$$

Then, by the rule for derivative of quotient,

$$\begin{aligned}
\frac{\partial\left(\mathcal{L}_{\text{tt+rag}}(\boldsymbol{W}^*)\right)}{\partial n} &= \frac{(B+2Cn)(m+n)^2 - 2(m+n)\left(A+Bn+Cn^2\right)}{(m+n)^4} \\
&= \frac{(B+2Cn)(m+n) - 2\left(A+Bn+Cn^2\right)}{(m+n)^3} \\
&= \frac{Bm+Bn+2Cmn+2Cn^2-2A-2Bn-2Cn^2}{(m+n)^3} \\
&= \frac{Bm-Bn+2Cmn-2A}{(m+n)^3}
\end{aligned}$$

Set the derivative to be zero, we have

$$Bm - Bn + 2Cmn - 2A = 0$$

and

$$
\begin{aligned}
n^\star &= \frac{Bm - 2A}{B - 2Cm} \\
&= \frac{m(m^2\left(\omega_2\sigma_{rag}^2 + \|\beta_{tt}\|^2\tau_{21}\right)) - 2(m^3\omega_1\sigma^2 + \|\beta_{tt}\|^2\tau_{30}m^3 + \|\beta_{tt}\|^2\tau_2 m^2)}{(m^2\left(\omega_2\sigma_{rag}^2 + \|\beta_{tt}\|^2\tau_{21}\right)) - 2m(\|\beta_{tt}\|^2(\tau_{22}m^2 + \tau_{12}m + \tau_2))} \\
&= \frac{m\left(2\|\beta_{tt}\|_2^2 dm + 2\|\beta_{tt}\|_2^2 d + 2\|\beta_{tt}\|_2^2 m - dm\sigma_{rag}^2 + 2m\sigma^2\right)}{d\left(2\|\beta_{tt}\|_2^2 m^2 - 2\|\beta_{tt}\|_2^2 m + 2\|\beta_{tt}\|_2^2 - m\sigma_{rag}^2\right)} \\
&\leqslant \frac{md\left(2\|\beta_{tt}\|_2^2 dm - dm\sigma_{rag}^2 + 2m\sigma^2\right)}{d^2\left(2\|\beta_{tt}\|_2^2 m^2 - 2\|\beta_{tt}\|_2^2 m + 2\|\beta_{tt}\|_2^2 - m\sigma_{rag}^2\right)} \\
&= \mathcal{O}\left(\frac{md\left(2\|\beta_{tt}\|_2^2 dm - dm\sigma_{rag}^2 + 2m\sigma^2\right)}{d^2\left(2\|\beta_{tt}\|_2^2 m^2 - m\sigma_{rag}^2\right)}\right) \\
&= \mathcal{O}\left(\frac{m\left(d^2\|\beta_{tt}\|_2^2 + d\sigma^2 - d^2\sigma_{rag}^2\right)}{md^2\|\beta_{tt}\|_2^2 - d^2\sigma_{rag}^2}\right) \\
&= \mathcal{O}\left(\frac{d\|\beta_{tt}\|_2^2 + \sigma^2 - d\sigma_{rag}^2}{d\|\beta_{tt}\|_2^2}\right)
\end{aligned}
$$

where the third step follows from upper bounding the numerator, and the fourth step follows from lower bounding the denominator.

n^* **as Global Minimizer** Now, we will show that the stationary point is the global minimizer. The second order derivative is give as:

$$
\frac{\partial\left(\mathcal{L}_{tt+rag}(\boldsymbol{W}^*)\right)}{\partial n} = \frac{2\left(Cm^2 - 2Cmn - 2Bm + Bn + 3A\right)}{(m+n)^4} \tag{D.53}
$$

Plug in $Bm - Bn^* + 2Cmn^* - 2A = 0$, we have

$$\frac{\partial \left(\mathcal{L}_{\text{tt+rag}}(\boldsymbol{W}^*)\right)}{\partial n}|_{n=n^*} = \frac{2\left(Cm^2 - A\right)}{(m-n)(m+n)^3} \geqslant 0 \tag{D.54}$$

Since $n^* = \mathcal{O}(1)$, we have $m > n^*$ for large m. Also, we have $Cm^2 > A$ for large m, thus we have $\frac{\partial\left(\mathcal{L}_{\text{tt+rag}}(\boldsymbol{W}^*)\right)}{\partial n}|_{n=n^*} \geqslant 0$, and n^* is the local minimum. Now, we check the first order derivative of $n \geqslant n^*$,

$$\begin{aligned} Bm - Bn + 2Cmn - 2A &= Bm - Bn + 2Cmn - 2A - (Bm - Bn^* + 2Cmn^* - 2A) \\ &= -B(n - n^*) + 2Cm(n - n^*) \geqslant 0 \end{aligned}$$

where it follows from $B \leqslant 0, C \geqslant 0$. Similarly, we can show that $Bm - Bn + 2Cmn - 2A \leqslant 0, \quad \forall n \leqslant n^*$. Thus, we have n^* to be the global minimum of the loss.

Improvement from n^* Here, we plug in $n = n^*$ and $n = 0$ into Equation (D.51). We have

$$\begin{aligned} \mathcal{L}_{\text{tt+rag}}|_{n=n^*}(\boldsymbol{W}^*) &= \frac{A + Bn^* + C(n^*)^2}{(m+n^*)^2(m+d+1)^2} \\ \mathcal{L}_{\text{tt+rag}}|_{n=0}(\boldsymbol{W}^*) &= \frac{A}{m^2(m+d+1)^2} \end{aligned} \tag{D.55}$$

Then, the improvement is give as

$$\begin{aligned} \mathcal{L}_{\text{tt+rag}}|_{n=0}(\boldsymbol{W}^*) - \mathcal{L}_{\text{tt+rag}}|_{n=n^*}(\boldsymbol{W}^*) &= \frac{A(m+n^*)^2 - m^2(A + Bn^* + C(n^*)^2)}{m^2(m+n^*)^2(m+d+1)^2} \\ &= \frac{(n^*)^2(2Cm - B)}{2m^2\,(m+n^*)\,(m+d+1)^2} \\ &= \mathcal{O}\left(\frac{Cm}{m^5}\right) \\ &= \mathcal{O}\left(\frac{m^2 d^2 \|\beta_{\text{tt}}\|_2^2 m}{m^5}\right) \\ &= \mathcal{O}\left(\frac{1}{m^2}\right) \end{aligned} \tag{D.56}$$

where the second step step follows from $Bm - Bn^* + 2Cmn^* - 2A = 0$ and the third step follows from $n^* = \mathcal{O}(1)$, and the four step follows from $B \leqslant 0$ and $|B| = \mathcal{O}(C)$. It finishes the proof. □

D.2.2 Non-Uniform Retrieval Noise

Now, we proceed to the proof for non-uniform retrieval noise.

Distance-Proportional Noise

Theorem (Restatement of Theorem 5.2)**.** *Under Assumption 1, 2, 4, the population loss is given as:*

$$\hat{\text{err}}_{\textit{variance}}(\boldsymbol{W}) = m\sigma^2 \operatorname{tr}(\boldsymbol{W}^\top \boldsymbol{W}) + \sum_{i=1}^{n} \gamma_1 \delta_i^2 [(1+\delta_i^2)\operatorname{tr}(\boldsymbol{W}^\top \boldsymbol{W}) + \operatorname{tr}(\boldsymbol{W}^2) + \operatorname{tr}(\boldsymbol{W})^2]$$

If the variance of the retrieval distance follows power law, i.e. $\exists \gamma_2 > 0, q \geqslant 0$ *s.t.* $\delta_i^2 = \gamma_2 i^q$*, then*

$$\hat{\text{err}}_{\textit{bias}}(\boldsymbol{W}^*) = \mathcal{O}\left(\underset{\textit{bias}}{\text{err}}(\boldsymbol{W}^*) + \|\beta_{tt}\|_2^2 \left[\frac{d n^{2q+1} + n^{2q+2}}{(m+n)^2}\right]\right) \tag{D.57}$$

and

$$\hat{\text{err}}_{\textit{variance}}(\boldsymbol{W}^*) = \mathcal{O}\left(\frac{dm\sigma^2 + d(n^{2q+1})\sigma^2}{(m+n)^2}\right) = \begin{cases} \mathcal{O}\left(d n^{2q-1}\sigma^2\right) & \textit{if } n \to \infty,\ q \leqslant 1/2 \\ \textit{diverges} & \textit{if } n \to \infty,\ q > 1/2 \end{cases} \tag{D.58}$$

Proof. We first write down the error explicitly similar to Equation (D.21).

$$y_q - x_q^\top \boldsymbol{W} \boldsymbol{X}^\top \boldsymbol{y} = x_q^\top (I - \boldsymbol{W}\boldsymbol{G})\beta_{tt} - x_q^\top \boldsymbol{W}\boldsymbol{X}^\top \boldsymbol{\epsilon} + \epsilon_q$$

And we can break down the population loss as

$$\hat{\mathcal{L}}_{\text{tt+rag}}(W) = \mathop{\mathbb{E}}_{(x_q, y_q), (X,y), \epsilon, r} \left(x_q^\top (I - WG)\, \beta_{\text{tt}}\right)^2 + \left(x_q^\top W X^\top \epsilon\right)^2 + \sigma^2 \tag{D.59}$$

Variance-Induced Error

$$\begin{aligned} \hat{\text{err}}_{\text{variance}}(W) &= \mathbb{E}(x_q^\top W X^\top \epsilon)^2 \\ &= \sum_{i,j=1}^{m+n} (x_i^\top W^\top x_q)(x_q^\top W x_j)\, \mathbb{E}(\epsilon_i \epsilon_j) \end{aligned} \tag{D.60}$$

Because the noise are independent and zero-mean, we have

$$\mathbb{E}[\epsilon_j \epsilon_j] = \begin{cases} \sigma^2, & i = j \leqslant m \\ \sigma^2_{\text{rag},i}, & i = j > m \\ 0, & i \neq j \end{cases}$$

Then,

$$\text{LHS} = \sum_{i=1}^{m} \sigma^2\, \mathbb{E}[(x_q^\top W x_i)^2] + \sum_{i=m+1}^{m+n} \sigma^2_{\text{rag},i-m} \cdot \mathbb{E}[x_q^\top W (x_q + r_{i-m})^2]$$

Thus, the ICL contribution remains the same as Theorem 5.1, i.e.

$$\sum_{i=1}^{m} \sigma^2\, \mathbb{E}[(x_q^\top W x_i)^2] = m\sigma^2 \operatorname{tr}(W^\top W)$$

To compute the RAG contribution, we evaluate the formula similar to Equation (D.27).

$$\begin{aligned} \mathbb{E}\left[\left(x_q^\top W (x_q + r_i)\right)^2\right] &= \mathbb{E}[(x_q^\top W x_q)^2] + \mathbb{E}[(x_q^\top W r_i)^2] + 2\,\mathbb{E}[x_q^\top W x_q \cdot x_q^\top W r_i] \\ &= \operatorname{tr}(W^\top W) + \operatorname{tr}(W^2) + \delta_i^2 \operatorname{tr}(W^\top W) + \operatorname{tr}(W)^2 \end{aligned} \tag{D.61}$$

And thus, the RAG error contribution is

$$\sum_{i=m+1}^{m+n} \sigma_{\text{rag},i-m}^2 \cdot \mathbb{E}[x_q^\top W(x_q + r_{i-m})^2] = \sum_{i=1}^{n} \sigma_{\text{rag},i}^2 [(1+\delta_i^2)\,\text{tr}(W^\top W) + \text{tr}(W^2) + \text{tr}(W)^2]$$

Plug in $\sigma_{\text{rag},i}^2 = \gamma_1 \delta_i^2$, and combining all terms together, we have

$$\hat{\text{err}}_{\text{variance}}(W) = m\sigma^2\,\text{tr}(W^\top W) + \sum_{i=1}^{n} \gamma_1 \delta_i^2 [(1+\delta_i^2)\,\text{tr}(W^\top W) + \text{tr}(W^2) + \text{tr}(W)^2]$$

Now, if we further assume $\delta_i^2 = \gamma_2 i^q$, and plug in the value of

$$\begin{aligned}
\hat{\text{err}}_{\text{variance}}(W^*) &= m\sigma^2\,\text{tr}\left((W^*)^\top W^*\right) + \sum_{i=1}^{n} \gamma_1\gamma_2 i^q [(1+\gamma_2 i^q)\,\text{tr}\left((W^*)^\top W^*\right) + \text{tr}\left((W^*)^2\right) + \text{tr}(W^*)^2] \\
&= \frac{m^2}{(m+d+1)^2(m+n)^2}\left[dm\sigma^2 + \gamma_1\gamma_2\left[(2d+d^2)\sum_{i=1}^{n} i^q\sigma^2 + d\gamma_2\sum_{i=1}^{n} i^{2q}\sigma^2\right]\right] \\
&= \frac{m^2}{(m+d+1)^2(m+n)^2}\left[dm\sigma^2 + \gamma_1\gamma_2\sigma^2\left[(2d+d^2)\,\mathcal{O}\left(\frac{n^{q+1}}{q+1} + \frac{n^q}{2}\right) + d\gamma_2\mathcal{O}\left(\frac{n^{2q+}\right.\right.\right. \\
&= \mathcal{O}\left(\frac{dm\sigma^2 + d(n^{2q+1})}{(m+n)^2}\right) \\
&= \begin{cases} \mathcal{O}\left(dn^{2q-1}\sigma^2\right) & \text{if } n \to \infty,\ q < 1/2 \\ \mathcal{O}\left(d\sigma^2\right) & \text{if } n \to \infty,\ q = 1/2 \\ \text{diverges} & \text{if } n \to \infty,\ q > 1/2 \end{cases}
\end{aligned}$$

where the second step follows from the Euler–Maclaurin expansion of the power sum.

Bias-Induced Error From Equation (D.38), we note that

$$\underset{\text{bias}}{\text{err}}(W) = \beta_{tt}^\top \left[M_1 - M_2 - M_3 + M_{41} + \sum_{i=1}^{n}(M_{42} + M_{42}^\top) + \sum_{i=1}^{n} M_{43} + \sum_{i\neq j, i,j\in[n]} M_{44}\right]\beta_{tt}$$

Specifically,

$$\begin{aligned}\mathop{\mathbb{E}}_{\mathbf{x}_q,\mathcal{X}}\left[(\mathrm{I}-\boldsymbol{W}\mathbf{G})^\top \mathbf{x}_q\mathbf{x}_q^\top(\mathrm{I}-\boldsymbol{W}\mathbf{G})\right] &= \mathop{\mathbb{E}}_{\mathbf{x}_q,\mathcal{X}}\left(\mathrm{I}-\mathbf{G}\boldsymbol{W}^\top\right)\mathbf{x}_q\mathbf{x}_q^\top(\mathrm{I}-\boldsymbol{W}\mathbf{G}) \\ &= \underbrace{\mathbb{E}\,\mathbf{x}_q\mathbf{x}_q^\top}_{:=M_1} - \underbrace{\mathbb{E}\,\mathbf{x}_q\mathbf{x}_q^\top\boldsymbol{W}\mathbf{G}}_{:=M_2} - \underbrace{\mathbb{E}\,\mathbf{G}\boldsymbol{W}^\top\mathbf{x}_q\mathbf{x}_q^\top}_{:=M_3} + \underbrace{\mathbb{E}\,\mathbf{G}\boldsymbol{W}^\top\mathbf{x}_q\mathbf{x}_q^\top\boldsymbol{W}\mathbf{G}}_{:=M_4}\end{aligned} \tag{D.62}$$

To avoid the repeated computation, we will highlight the calculation that involves δ_i, omit some calculation steps given in the standard case and discuss its bound after allowing for non-uniform offset. We will only compute δ_i^2-involving term and use … to denote the rest terms, since we assume $\delta^2 \ll 1$ in proving Theorem 5.1. The final bound will be given as

$$\hat{\mathrm{err}}_{\mathrm{bias}}(\boldsymbol{W}^*) = \underset{\mathrm{bias}}{\mathrm{err}}(\boldsymbol{W}^*) + \delta^2\text{-involved terms}$$

$M_1 = \mathbb{E}\left[\mathbf{x}_q\mathbf{x}_q^\top\right] = \mathrm{I}$ and remains the same. Let $s_\delta := \sum_i \delta_i^2$, $S_\delta := \sum_i(\delta_i^2)^2$. Then, we expand out the terms in M_2:

$$\begin{aligned}M_2 = \mathop{\mathbb{E}}_{\mathbf{x}_q,\mathbf{r}}\mathbf{x}_q\mathbf{x}_q^\top\boldsymbol{W}\mathbf{G} &= \left(\mathop{\mathbb{E}}_{\mathbf{x}_q,\mathbf{r}}\mathbf{x}_q\mathbf{x}_q^\top\right)\boldsymbol{W}\mathbf{G}_0 + \mathop{\mathbb{E}}_{\mathbf{x}_q,\mathbf{r}}\mathbf{x}_q\mathbf{x}_q^\top\boldsymbol{W}\sum_{i=1}^n(\mathbf{x}_q+\mathbf{r}_i)(\mathbf{x}_q+\mathbf{r}_i)^\top \\ &= \boldsymbol{W}\mathbf{G}_0 + \mathop{\mathbb{E}}_{\mathbf{x}_q,\mathbf{r}}\mathbf{x}_q\mathbf{x}_q^\top\boldsymbol{W}\sum_{i=1}^n(\mathbf{x}_q\mathbf{x}_q^\top+\mathbf{r}_i\mathbf{r}_i^\top) \\ &= \boldsymbol{W}\mathbf{G}_0 + \mathop{\mathbb{E}}_{\mathbf{x}_q,\mathbf{r}}\mathbf{x}_q\mathbf{x}_q^\top\boldsymbol{W}\sum_{i=1}^n(\mathbf{x}_q\mathbf{x}_q^\top+\delta_i^2\mathrm{I}) \\ &= \cdots + s_\delta\boldsymbol{W}\end{aligned} \tag{D.63}$$

Similarly, $M_3 = M_2^\top = \cdots + s_\delta\boldsymbol{W}^\top$. Now, we perform similar expansion for M_4.

First, we note that $M_{41} = \mathbb{E}_{\mathbf{x}_q,\mathcal{X}}[\mathbf{G}_0\boldsymbol{W}^\top\mathbf{x}_q\mathbf{x}_q^\top\boldsymbol{W}\mathbf{G}_0]$ is independent of δ_i^2.

$$
\begin{aligned}
\sum_{i\in[n]} M_{42} &:= \sum_{i\in[n]} \mathop{\mathbb{E}}_{x_q,X} \mathbf{G}_0 \boldsymbol{W}^\top \boldsymbol{x}_q \boldsymbol{x}_q^\top \boldsymbol{W} \mathbf{G}_i \\
&= \sum_{i\in[n]} \mathop{\mathbb{E}}_{x_q,X} \mathbf{G}_0 \boldsymbol{W}^\top \boldsymbol{x}_q \boldsymbol{x}_q^\top \boldsymbol{W} \left(\boldsymbol{x}_q + \mathbf{r}_i\right) \left(\boldsymbol{x}_q + \mathbf{r}_i\right)^\top \\
&= \sum_{i\in[n]} \mathop{\mathbb{E}}_{x_q,X} \mathbf{G}_0 \boldsymbol{W}^\top \boldsymbol{x}_q \boldsymbol{x}_q^\top \boldsymbol{W} \left(\boldsymbol{x}_q \boldsymbol{x}_q^\top + \mathbf{r}_i \mathbf{r}_i^\top\right) \\
&= \sum_{i\in[n]} \mathop{\mathbb{E}}_{x_q,X} \mathbf{G}_0 \boldsymbol{W}^\top \left(\boldsymbol{W} + \boldsymbol{W}^\top + \operatorname{tr}(\boldsymbol{W}) + \boldsymbol{W}\delta^2\right) \\
&= \sum_{i\in[n]} m \left(\boldsymbol{W}^\top \boldsymbol{W} + \boldsymbol{W}^\top \boldsymbol{W}^\top + \operatorname{tr}(\boldsymbol{W}) \boldsymbol{W}^\top + \delta^2 \boldsymbol{W}^\top \boldsymbol{W}\right) \\
&= \cdots + m s_\delta \boldsymbol{W}^\top \boldsymbol{W}
\end{aligned}
\tag{D.64}
$$

Following the derivation of the 6th-order and 4th-order moments as in Theorem D.3

and Theorem D.2, we have

$$
\begin{aligned}
\sum_{i\in[n]} M_{43} &:= \sum_{i\in[n]} \mathop{\mathbb{E}}_{\mathbf{x}_q,\mathbf{X},\mathbf{r}_i} \mathbf{G}_i W^\top \mathbf{x}_q \mathbf{x}_q^\top W \mathbf{G}_i \\
&= \left.\begin{aligned} &2\left(W^2 + (W^2)^\top + W^\top W + WW^\top + \operatorname{tr}(W)\,(W + W^\top)\right) \\ &+ \operatorname{tr}(W)^2\,\mathrm{I} + \operatorname{tr}\left(W^2\right)\mathrm{I} + \operatorname{tr}\left(W^\top W\right)\mathrm{I} \end{aligned}\right\} \text{0th-order in } \mathbf{r}_i\text{, Theorem D.3} \\
&\quad + \underbrace{2\delta_i^4 W^\top W + \delta_i^4 \operatorname{tr}(W^\top W)\mathrm{I}}_{\text{4th-order in } \mathbf{r}_i\text{, Equation (D.1)}} \\
&\quad + \underbrace{\delta_i^2\left[\operatorname{tr}(W)\left(W^\top + W\right) + W^2 + (W^2)^\top + 2W^\top W\right]}_{\text{Equation (D.5) and its transpose}} \\
&\quad + \underbrace{\left(\operatorname{tr}\left(W^2\right) + \operatorname{tr}\left(W^\top W\right) + \operatorname{tr}(W)^2\right)\delta_i^2\mathrm{I}}_{\text{Equation (D.2)}} \\
&\quad + \underbrace{2\delta_i^2 WW^\top + \delta_i^2 \operatorname{tr}(W^\top W)\mathrm{I}}_{\text{Equation (D.3)}} \\
&\quad + \underbrace{\delta_i^2\left[\operatorname{tr}(W)\left(W^\top + W\right) + W^2 + (W^2)^\top + 2W^\top W\right]}_{\text{Equation (D.4) and its transpose}} \\
&= \sum_{i\in[n]} (2 + 2\delta_i^2)\left[\operatorname{tr}(W)\left(W^\top + W\right) + W^2 + (W^2)^\top\right] \\
&\quad + \sum_{i\in[n]} (2 + 4\delta_i^2) W^\top W + \sum_{i\in[n]} 2WW^\top \\
&\quad + \sum_{i\in[n]} (1 + \delta_i^2)\left[\operatorname{tr}(W)^2\,\mathrm{I} + \operatorname{tr}\left(W^2\right)\mathrm{I} + \operatorname{tr}\left(W^\top W\right)\mathrm{I}\right] \\
&\quad + \sum_{i\in[n]} \left(2\delta_i^4 W^\top W + \delta_i^4 \operatorname{tr}(W^\top W)\mathrm{I} + 2\delta_i^2 WW^\top + \delta_i^2 \operatorname{tr}(W^\top W)\mathrm{I}\right) \\
&= \cdots + 2s_\delta\left[\operatorname{tr}(W)\left(W^\top + W\right) + W^2 + (W^2)^\top\right] \\
&\quad + (4s_\delta + 2S_\delta) W^\top W + 2s_\delta WW^\top \\
&\quad + s_\delta\left(\operatorname{tr}(W)^2 + \operatorname{tr}\left(W^2\right)\right)\mathrm{I} + (2s_\delta + S_\delta)\operatorname{tr}(W^\top W)\mathrm{I}
\end{aligned}
\tag{D.65}
$$

Also, we expand the cross-term out for $\forall i, j \in [n], i \neq j$:

$$\begin{aligned}
\sum_{i\neq j} M_{44} &:= \sum_{i\neq j} \mathbb{E}\, \mathbf{G}_i \boldsymbol{W}^\top \boldsymbol{x}_q \boldsymbol{x}_q^\top \boldsymbol{W} \mathbf{G}_j \\
&= \sum_{i\neq j} \left(\boldsymbol{x}_q \boldsymbol{x}_q^\top \boldsymbol{W}^\top \boldsymbol{x}_q \boldsymbol{x}_q^\top \boldsymbol{W} \boldsymbol{x}_q \boldsymbol{x}_q^\top + \mathbf{r}_i \mathbf{r}_i^\top \boldsymbol{W}^\top \boldsymbol{x}_q \boldsymbol{x}_q^\top \boldsymbol{W} \boldsymbol{x}_q \boldsymbol{x}_q^\top \right) \\
&\quad + \sum_{i\neq j} \left(\boldsymbol{x}_q \boldsymbol{x}_q^\top \boldsymbol{W}^\top \boldsymbol{x}_q \boldsymbol{x}_q^\top \boldsymbol{W} \mathbf{r}_j \mathbf{r}_j^\top + \mathbf{r}_i \mathbf{r}_i^\top \boldsymbol{W}^\top \boldsymbol{x}_q \boldsymbol{x}_q^\top \boldsymbol{W} \mathbf{r}_j \mathbf{r}_j^\top \right) \\
&= \cdots + \sum_{i\neq j} \delta_i^2 \left(\boldsymbol{W}^2 + \boldsymbol{W}^\top \boldsymbol{W} + \operatorname{tr}(\boldsymbol{W}) \boldsymbol{W} \right) \\
&\quad + \sum_{i\neq j} \delta_i^2 \left(\left(\boldsymbol{W}^2\right)^\top + \boldsymbol{W}^\top \boldsymbol{W} + \operatorname{tr}(\boldsymbol{W}) \boldsymbol{W}^\top \right) \\
&\quad + \sum_{i\neq j} \delta_i^2 \delta_j^2 \boldsymbol{W}^\top \boldsymbol{W}
\end{aligned} \tag{D.66}$$

In the non-uniform noise scenario, 4th-order term in δ_i will dominate the 2nd-order term in δ_i. Thus, we will plug $\delta_i^2 = \gamma_2 i^q, \boldsymbol{W}^* = \frac{m}{(m+d+1)(m+n)}$ into $\operatorname{err}_{\text{bias}}$:

$$\begin{aligned}
\underset{\text{bias}}{\operatorname{err}}(\boldsymbol{W}^*) &= \underset{\text{bias}}{\operatorname{err}}(\boldsymbol{W}^*) + \mathcal{O}\left(\beta_{tt}^\top \left[2 \sum_i^n (\delta_i^2)^2 (\boldsymbol{W}^*)^\top \boldsymbol{W}^* + \sum_{i\neq j, i\in[n], j\in[n]} \delta_i^2 \delta_j^2 (\boldsymbol{W}^*)^\top \boldsymbol{W}^* + \sum_i^n (\delta_i^2)^2 \operatorname{tr}((\boldsymbol{W}^* \right. \right. \\
&= \underset{\text{bias}}{\operatorname{err}}(\boldsymbol{W}^*) + \mathcal{O}\left(\beta_{tt}^\top \left[d n^{2q+1} (\boldsymbol{W}^*)^\top \boldsymbol{W}^* + n^{2q+2} (\boldsymbol{W}^*)^\top \boldsymbol{W}^* \right] \beta_{tt} \right) \\
&= \underset{\text{bias}}{\operatorname{err}}(\boldsymbol{W}^*) + \mathcal{O}\left(\beta_{tt}^\top \left[\frac{d n^{2q+1} + n^{2q+2}}{(m+n)^2} \right] \beta_{tt} \right)
\end{aligned}$$

It finishes the proof. □

Distance-Weighted Probabilistic Noise

Theorem (Restatement of Theorem 5.3)**.** *Under Assumption 1, 2, 5, then* $\tilde{\text{err}}_{bias}(\boldsymbol{W}) = \hat{\text{err}}_{bias}(\boldsymbol{W})$*, and*

$$\hat{\text{err}}_{variance}(\boldsymbol{W}) = m\sigma^2 \operatorname{tr}(\boldsymbol{W}^\top \boldsymbol{W}) + \sum_{i=1}^n \left(p_i \sigma_s^2 + (1-p_i)\sigma_l^2 \right) \left[(1+\delta_i^2) \operatorname{tr}(\boldsymbol{W}^\top \boldsymbol{W}) + \operatorname{tr}(\boldsymbol{W}^2) + \operatorname{tr}(\boldsymbol{W})^2 \right]$$

If the variance of the retrieval distance follows power law, i.e. $\exists \gamma_2 > 0, q \geqslant 0$ *s.t.* $\delta_i^2 = \gamma_2 i^q$, *then:*

$$\tilde{\text{err}}_{\textit{variance}}(\boldsymbol{W}^*) = \begin{cases} \mathcal{O}\left(c_l d n^{q-1} \sigma^2 - (c_l - c_s)\sigma^2 d n^{q-1-q\tilde{q}}\right) & \textit{if } n \to \infty,\ q \leqslant 1 \\ \textit{diverges} & \textit{if } n \to \infty,\ q > 1 \end{cases} \tag{D.67}$$

Proof. First, we note that $\tilde{\text{err}}_{\text{bias}}(\boldsymbol{W}) = \hat{\text{err}}_{\text{bias}}(\boldsymbol{W})$, since both are independent of σ_{rag}^2 and depend on the same set of $\forall i,\ \delta_i^2$.

We write down error explicitly similar to Equation (D.21) and break down the population loss as:

$$\tilde{\mathcal{L}}_{\text{tt+rag}}(\boldsymbol{W}) = \mathop{\mathbb{E}}_{(\boldsymbol{x}_q, y_q),(\boldsymbol{X},\boldsymbol{y}),\boldsymbol{\epsilon},\boldsymbol{r}} \left(\boldsymbol{x}_q^\top (I - \boldsymbol{W}\boldsymbol{G})\, \beta_{\text{tt}}\right)^2 + \left(\boldsymbol{x}_q^\top \boldsymbol{W}\boldsymbol{X}^\top \boldsymbol{\epsilon}\right)^2 + \sigma^2 \tag{D.68}$$

We note that $\tilde{\text{err}}_{\text{bias}}(\boldsymbol{W}) = \text{err}_{\text{bias}}(\boldsymbol{W})$, since the error from bias does not depend on the sample complexity.

$$\begin{aligned} \tilde{\text{err}}_{\text{variance}}(\boldsymbol{W}) &= \mathbb{E}(\boldsymbol{x}_q^\top \boldsymbol{W}\boldsymbol{X}^\top \boldsymbol{\epsilon})^2 \\ &= \sum_{i,j=1}^{m+n} (\boldsymbol{x}_i^\top \boldsymbol{W}^\top \boldsymbol{x}_q)(\boldsymbol{x}_q^\top \boldsymbol{W}\boldsymbol{x}_j)\, \mathbb{E}(\epsilon_i \epsilon_j) \end{aligned} \tag{D.69}$$

Because the noise are independent and zero-mean, we have

$$\mathbb{E}[\epsilon_j \epsilon_j] = \begin{cases} \sigma^2, & i = j \leqslant m \\ \sigma_s^2, & i = j > m,\ \text{w.p. } p \\ \sigma_l^2, & i = j > m,\ \text{w.p. } 1 - p \\ 0, & i \neq j \end{cases}$$

Thus, the ICL contribution remains the same as Theorem 5.1, i.e.

$$\sum_{i=1}^{m} \sigma^2 \, \mathbb{E}[(\boldsymbol{x}_q^\top \boldsymbol{W} \boldsymbol{x}_i)^2] = m\sigma^2 \operatorname{tr}(\boldsymbol{W}^\top \boldsymbol{W})$$

To compute the RAG contribution, we evaluate the formula similar to Equation (D.27).

$$\begin{aligned} \mathbb{E}\left[\left(\boldsymbol{x}_q^\top \boldsymbol{W}(\boldsymbol{x}_q + \boldsymbol{r}_i)\right)^2\right] &= \mathbb{E}[(\boldsymbol{x}_q^\top \boldsymbol{W} \boldsymbol{x}_q)^2] + \mathbb{E}[(\boldsymbol{x}_q^\top \boldsymbol{W} \boldsymbol{r}_i)^2] + 2\,\mathbb{E}[\boldsymbol{x}_q^\top \boldsymbol{W} \boldsymbol{x}_q \cdot \boldsymbol{x}_q^\top \boldsymbol{W} \boldsymbol{r}_i] \\ &= \operatorname{tr}(\boldsymbol{W}^\top \boldsymbol{W}) + \operatorname{tr}(\boldsymbol{W}^2) + \delta_i^2 \operatorname{tr}(\boldsymbol{W}^\top \boldsymbol{W}) + \operatorname{tr}(\boldsymbol{W})^2 \end{aligned} \tag{D.70}$$

And thus, the RAG error contribution is

$$\sum_{i=1}^{n} \left(p_i \sigma_s^2 + (1 - p_i)\sigma_l^2\right) \left[(1 + \delta_i^2) \operatorname{tr}(\boldsymbol{W}^\top \boldsymbol{W}) + \operatorname{tr}(\boldsymbol{W}^2) + \operatorname{tr}(\boldsymbol{W})^2\right]$$

Plug in $\sigma_{\text{rag},i}^2 = \gamma_1 \delta_i^2$, and combining all terms together, we have

$$\hat{\text{err}}_{\text{variance}}(\boldsymbol{W}) = m\sigma^2 \operatorname{tr}(\boldsymbol{W}^\top \boldsymbol{W}) + \sum_{i=1}^{n} \left(p_i \sigma_s^2 + (1 - p_i)\sigma_l^2\right) \left[(1 + \delta_i^2) \operatorname{tr}(\boldsymbol{W}^\top \boldsymbol{W}) + \operatorname{tr}(\boldsymbol{W}^2) + \operatorname{tr}(\boldsymbol{W})^2\right]$$

Now we further assume $p_i = (1 + \delta_i^2)^{-\tilde{q}},\ \tilde{q} \geqslant 0$, and plug in the value of $\boldsymbol{W}^*$.

Let $B := \frac{m^2}{(m+d+1)^2(m+n)^2}$,

$$
\begin{aligned}
\tilde{\text{err}}_{\text{variance}}(\boldsymbol{W}^*) &= m\sigma^2 \operatorname{tr}(\boldsymbol{W}^\top \boldsymbol{W}) + \sum_{i=1}^{n} \left(p_i \sigma_s^2 + (1-p_i)\sigma_l^2\right) \left[(1+\delta_i^2)\operatorname{tr}(\boldsymbol{W}^\top \boldsymbol{W}) + \operatorname{tr}(\boldsymbol{W}^2) + \operatorname{tr}(\boldsymbol{W})^2\right] \\
&= B\left[dm\sigma^2 + \sum_{i=1}^{n} \left(c_l \sigma^2 - (1+\delta_i^2)^{-\tilde{q}}(c_l - c_s)\sigma^2\right)\left[(1+\delta_i^2)\cdot d + d + d^2\right]\right] \\
&\approx B\left[dm\sigma^2 + c_l\sigma^2 \sum_{i=1}^{n}\left(d\delta_i^2 + d^2\right) - (c_l - c_s)\sigma^2 \sum_{i=1}^{n}\left(d(1+\delta_i^2)^{1-\tilde{q}} + d^2(1+\delta_i^2)^{-\tilde{q}}\right)\right] \\
&\approx B\left[dm\sigma^2 + c_l\sigma^2 \sum_{i=1}^{n} d\delta_i^2 - (c_l - c_s)\sigma^2 \sum_{i=1}^{n} d(1+\delta_i^2)^{1-\tilde{q}}\right] \\
&\approx \begin{cases} B\left[dm\sigma^2 + c_l\sigma^2 d n^{q+1} - (c_l - c_s)\sigma^2 d\log(n)\right] & \text{if } \tilde{q} = 1 + 1/q \\ B\left[dm\sigma^2 + c_l\sigma^2 d n^{q+1} - (c_l - c_s)\sigma^2 d n^{1+q-q\tilde{q}}\right] & \text{else} \end{cases}
\end{aligned}
$$

where the second line follows from omitting the lower order term.

If $\tilde{q} = 1 + 1/q$, we note that the middle term will dominate the error. And combining all cases, we could obtain

$$
\tilde{\text{err}}_{\text{variance}}(\boldsymbol{W}^*) = \begin{cases} \mathcal{O}\left(c_l d n^{q-1}\sigma^2 - (c_l - c_s)\sigma^2 d n^{q-1-q\tilde{q}}\right) & \text{if } n \to \infty,\ q \leqslant 1 \\ \text{diverges} & \text{if } n \to \infty,\ q > 1 \\ \mathcal{O}\left(c_l d n^{q-1}\sigma^2 + (c_l - c_s) d^2 \frac{\log n}{n^2}\sigma^2\right) & \text{if } n \to \infty,\ \tilde{q} = 1 + 1/q \end{cases}
$$

□

2021. Online versus batch prediction. https://cloud.google.com/ai-platform/prediction/docs/online-vs-batch-prediction.

Achiam, Josh, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Ahn, Kwangjun, Xiang Cheng, Hadi Daneshmand, and Suvrit Sra. 2023. Transformers learn to implement preconditioned gradient descent for in-context learning. *Advances in Neural Information Processing Systems* 36:45614–45650.

Ajakan, Hana, Pascal Germain, Hugo Larochelle, François Laviolette, and Mario Marchand. 2014. Domain-adversarial neural networks. *stat* 1050:15.

Anthropic. 2023. Claude 3. https://www.anthropic.com/claude. Accessed: 2023-08-08.

Asai, Akari, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. 2023. Self-RAG: Learning to retrieve, generate, and critique through self-reflection. *arXiv preprint arXiv:2310.11511*.

Askell, Amanda, Yuntao Bai, Anna Chen, Dawn Drain, Deep Ganguli, Tom Henighan, Andy Jones, Nicholas Joseph, Ben Mann, Nova DasSarma, et al. 2021. A general language assistant as a laboratory for alignment. *arXiv preprint arXiv:2112.00861*.

Assouad, Patrick. 1983. Densité et dimension. In *Annales de l'institut fourier*, vol. 33, 233–282.

Athalye, Anish, Nicholas Carlini, and David A. Wagner. 2018. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *Proceedings of the 35th international conference on machine learning, ICML 2018,*

stockholmsmässan, stockholm, sweden, july 10-15, 2018, ed. Jennifer G. Dy and Andreas Krause, vol. 80 of *Proceedings of Machine Learning Research*, 274–283. PMLR.

Auer, Peter. 2002. Using confidence bounds for exploitation-exploration trade-offs. *Journal of Machine Learning Research* 3(Nov):397–422.

Azar, Mohammad Gheshlaghi, Zhaohan Daniel Guo, Bilal Piot, Remi Munos, Mark Rowland, Michal Valko, and Daniele Calandriello. 2024. A general theoretical paradigm to understand learning from human preferences. In *International conference on artificial intelligence and statistics*, 4447–4455. PMLR.

Baharlouei, Sina, Fatemeh Sheikholeslami, Meisam Razaviyayn, and Zico Kolter. 2022. Improving adversarial robustness via joint classification and multiple explicit detection classes. *arXiv preprint arXiv:2210.14410*.

Bai, Yuntao, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. 2022a. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*.

Bai, Yuntao, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. 2022b. Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073*.

Bakker, Michiel, Martin Chadwick, Hannah Sheahan, Michael Tessler, Lucy Campbell-Gillingham, Jan Balaguer, Nat McAleese, Amelia Glaese, John Aslanides, Matt Botvinick, et al. 2022. Fine-tuning language models to find agreement among humans with diverse preferences. *Advances in Neural Information Processing Systems* 35:38176–38189.

Bartlett, Peter L., Nick Harvey, Chris Liaw, and Abbas Mehrabian. 2017. Nearly-tight vc-dimension and pseudodimension bounds for piecewise linear neural networks. 1703.02930.

Ben-David, Shai, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. 2010. A theory of learning from different domains. *Machine learning* 79(1-2):151–175.

Blumer, Anselm, Andrzej Ehrenfeucht, David Haussler, and Manfred K Warmuth. 1989. Learnability and the vapnik-chervonenkis dimension. *Journal of the ACM (JACM)* 36(4):929–965.

Borgeaud, Sebastian, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George van den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, Diego de Las Casas, Aurelia Guy, Jacob Menick, Roman Ring, T. W. Hennigan, Saffron Huang, Lorenzo Maggiore, Chris Jones, Albin Cassirer, Andy Brock, Michela Paganini, Geoffrey Irving, Oriol Vinyals, Simon Osindero, Karen Simonyan, Jack W. Rae, Erich Elsen, and L. Sifre. 2021. Improving language models by retrieving from trillions of tokens. In *International conference on machine learning*.

Bradley, Ralph Allan, and Milton E Terry. 1952. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika* 39(3/4):324–345.

Brown, Tom, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems* 33:1877–1901.

Burns, Collin, Pavel Izmailov, Jan Hendrik Kirchner, Bowen Baker, Leo Gao, Leopold Aschenbrenner, Yining Chen, Adrien Ecoffet, Manas Joglekar, Jan Leike, et al. 2023. Weak-to-strong generalization: Eliciting strong capabilities with weak supervision. *arXiv preprint arXiv:2312.09390*.

Carlini, Nicholas, and David Wagner. 2017a. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, 39–57. Ieee.

Carlini, Nicholas, and David A. Wagner. 2017b. Towards evaluating the robustness of neural networks. In *2017 IEEE symposium on security and privacy, SP 2017, san jose, ca, usa, may 22-26, 2017*, 39–57. IEEE Computer Society.

Carmon, Yair, Aditi Raghunathan, Ludwig Schmidt, John C. Duchi, and Percy Liang. 2019. Unlabeled data improves adversarial robustness. In *Advances in neural information processing systems 32: Annual conference on neural information processing systems 2019, neurips 2019, 8-14 december 2019, vancouver, bc, canada*, ed. Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, 11190–11201.

Chakraborty, Souradip, Jiahao Qiu, Hui Yuan, Alec Koppel, Furong Huang, Dinesh Manocha, Amrit Singh Bedi, and Mengdi Wang. 2024. Maxmin-rlhf: Towards equitable alignment of large language models with diverse human preferences. *arXiv preprint arXiv:2402.08925*.

Chan, Alex J, Hao Sun, Samuel Holt, and Mihaela van der Schaar. 2024. Dense reward for free in reinforcement learning from human feedback. *arXiv preprint arXiv:2402.00782*.

Chang, Jonathan D, Wenhao Shan, Owen Oertell, Kianté Brantley, Dipendra Misra, Jason D Lee, and Wen Sun. 2024. Dataset reset policy optimization for rlhf. *arXiv preprint arXiv:2404.08495*.

Chen, Danqi, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading wikipedia to answer open-domain questions. *arXiv preprint arXiv:1704.00051*.

Chen, Jianlv, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. 2024a. Bge m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. In *Annual meeting of the association for computational linguistics*.

Chen, Jiefeng, Yang Guo, Xi Wu, Tianqi Li, Qicheng Lao, Yingyu Liang, and Somesh Jha. 2021a. Towards adversarial robustness via transductive learning. *arXiv preprint arXiv:2106.08387*.

Chen, Jiefeng, Jayaram Raghuram, Jihye Choi, Xi Wu, Yingyu Liang, and Somesh Jha. 2021b. Revisiting adversarial robustness of classifiers with a reject option. In *The aaai-22 workshop on adversarial machine learning and beyond*.

Chen, Jiefeng, Xi Wu, Yang Guo, Yingyu Liang, and Somesh Jha. 2022. Towards evaluating the robustness of neural networks learned by transduction. In *International conference on learning representations*.

Chen, Zixiang, Yihe Deng, Huizhuo Yuan, Kaixuan Ji, and Quanquan Gu. 2024b. Self-play fine-tuning converts weak language models to strong language models. *arXiv preprint arXiv:2401.01335*.

Cheng, Chen, Xinzhi Yu, Haodong Wen, Jingsong Sun, Guanzhang Yue, Yihao Zhang, and Zeming Wei. 2025. Exploring the robustness of in-context learning with noisy labels. In *Icassp 2025-2025 ieee international conference on acoustics, speech and signal processing (icassp)*, 1–5. IEEE.

Cheng, Daixuan, Shaohan Huang, Junyu Bi, Yuefeng Zhan, Jianfeng Liu, Yujing Wang, Hao Sun, Furu Wei, Denvy Deng, and Qi Zhang. 2023. Uprise: Universal prompt retrieval for improving zero-shot evaluation. *arXiv preprint arXiv:2303.08518*.

Christiano, Paul F, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. 2017. Deep reinforcement learning from human preferences. *Advances in neural information processing systems* 30.

Chuang, Ching-Yao, Antonio Torralba, and Stefanie Jegelka. 2020. Estimating generalization under distribution shifts via domain-invariant representations. In *Proceedings of the 37th international conference on machine learning, ICML 2020, 13-18 july 2020, virtual event*, vol. 119 of *Proceedings of Machine Learning Research*, 1984–1994. PMLR.

Cohen, Jeremy M., Elan Rosenfeld, and J. Zico Kolter. 2019. Certified adversarial robustness via randomized smoothing. In *Proceedings of the 36th international*

conference on machine learning, ICML 2019, 9-15 june 2019, long beach, california, USA, ed. Kamalika Chaudhuri and Ruslan Salakhutdinov, vol. 97 of *Proceedings of Machine Learning Research*, 1310–1320. PMLR.

Colson, Benoît, Patrice Marcotte, and Gilles Savard. 2007. An overview of bilevel optimization.

Cover, Thomas, and Peter Hart. 1967. Nearest neighbor pattern classification. *IEEE transactions on information theory* 13(1):21–27.

Croce, Francesco, Maksym Andriushchenko, Vikash Sehwag, Edoardo Debenedetti, Nicolas Flammarion, Mung Chiang, Prateek Mittal, and Matthias Hein. 2020. Robustbench: a standardized adversarial robustness benchmark. *arXiv preprint arXiv:2010.09670*.

Croce, Francesco, and Matthias Hein. 2020a. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *Icml*.

———. 2020b. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *Proceedings of the 37th international conference on machine learning, ICML 2020, 13-18 july 2020, virtual event*, vol. 119 of *Proceedings of Machine Learning Research*, 2206–2216. PMLR.

———. 2020c. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *International conference on machine learning*, 2206–2216. PMLR.

Deng, Jia, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *2009 ieee conference on computer vision and pattern recognition*, 248–255. Ieee.

Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Ding, Gavin Weiguang, Yash Sharma, Kry Yik Chau Lui, and Ruitong Huang. 2020. MMA training: Direct input space margin maximization through adversarial training. In *8th international conference on learning representations, ICLR 2020, addis ababa, ethiopia, april 26-30, 2020*. OpenReview.net.

Ding, Jianbang, Xuancheng Ren, Ruixuan Luo, and Xu Sun. 2019. An adaptive and momental bound method for stochastic learning. *CoRR* abs/1910.12249. 1910.12249.

Dong, Qingxiu, Lei Li, Damai Dai, Ce Zheng, Jingyuan Ma, Rui Li, Heming Xia, Jingjing Xu, Zhiyong Wu, Baobao Chang, Xu Sun, Lei Li, and Zhifang Sui. 2024. A survey on in-context learning. In *Proceedings of the 2024 conference on empirical methods in natural language processing*, 1107–1128. Miami, Florida, USA: Association for Computational Linguistics.

Dong, Qingxiu, Lei Li, Damai Dai, Ce Zheng, Zhiyong Wu, Baobao Chang, Xu Sun, Jingjing Xu, and Zhifang Sui. 2022. A survey for in-context learning. *arXiv preprint arXiv:2301.00234*.

Dumoulin, Vincent, Daniel D Johnson, Pablo Samuel Castro, Hugo Larochelle, and Yann Dauphin. 2023. A density estimation perspective on learning from pairwise human preferences. *arXiv preprint arXiv:2311.14115*.

Ethayarajh, Kawin, Yejin Choi, and Swabha Swayamdipta. 2022. Understanding dataset difficulty with $\mathcal{V}$-usable information. In *International conference on machine learning*, 5988–6008. PMLR.

Ethayarajh, Kawin, Winnie Xu, Niklas Muennighoff, Dan Jurafsky, and Douwe Kiela. 2024. Kto: Model alignment as prospect theoretic optimization. *arXiv preprint arXiv:2402.01306*.

Ganin, Yaroslav, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. 2016a. Domain-adversarial training of neural networks. *J. Mach. Learn. Res.* 17(1): 2096–2030.

Ganin, Yaroslav, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor S. Lempitsky. 2016b. Domain-adversarial training of neural networks. *J. Mach. Learn. Res.* 17:59:1–59:35.

Gao, Leo, John Schulman, and Jacob Hilton. 2023. Scaling laws for reward model overoptimization. In *International conference on machine learning*, 10835–10866. PMLR.

Garg, Shivam, Dimitris Tsipras, Percy S Liang, and Gregory Valiant. 2022. What can transformers learn in-context? a case study of simple function classes. *Advances in Neural Information Processing Systems* 35:30583–30598.

Goldwasser, Shafi, Adam Tauman Kalai, Yael Kalai, and Omar Montasser. 2020a. Beyond perturbations: Learning guarantees with arbitrary adversarial test examples. *Advances in Neural Information Processing Systems* 33:15859–15870.

Goldwasser, Shafi, Adam Tauman Kalai, Yael Tauman Kalai, and Omar Montasser. 2020b. Beyond perturbations: Learning guarantees with arbitrary adversarial test examples. *CoRR* abs/2007.05145. 2007.05145.

Goodfellow, Ian. 2019a. A research agenda: Dynamic models to defend against correlated attacks. *arXiv preprint arXiv:1903.06293*.

Goodfellow, Ian J. 2019b. A research agenda: Dynamic models to defend against correlated attacks. *CoRR* abs/1903.06293.

Goodfellow, Ian J, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.

Goodfellow, Ian J., Jonathon Shlens, and Christian Szegedy. 2015a. Explaining and harnessing adversarial examples. In *3rd international conference on learning representations, ICLR 2015, san diego, ca, usa, may 7-9, 2015, conference track proceedings*, ed. Yoshua Bengio and Yann LeCun.

———. 2015b. Explaining and harnessing adversarial examples. 1412.6572.

Gozeten, Halil Alperen, M Emrullah Ildiz, Xuechen Zhang, Mahdi Soltanolkotabi, Marco Mondelli, and Samet Oymak. 2025. Test-time training provably improves transformers as in-context learners. *arXiv preprint arXiv:2503.11842*.

Gulcehre, Caglar, Tom Le Paine, Srivatsan Srinivasan, Ksenia Konyushkova, Lotte Weerts, Abhishek Sharma, Aditya Siddhant, Alex Ahern, Miaosen Wang, Chenjie Gu, et al. 2023. Reinforced self-training (rest) for language modeling. *arXiv preprint arXiv:2308.08998*.

Hanneke, Steve, Aryeh Kontorovich, and Menachem Sadigurschi. 2019. Sample compression for real-valued learners. In *Algorithmic learning theory*, 466–488. PMLR.

He, Kaiming, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. 2017. Mask r-cnn. In *Proceedings of the ieee international conference on computer vision*, 2961–2969.

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *2016 IEEE conference on computer vision and pattern recognition, CVPR 2016, las vegas, nv, usa, june 27-30, 2016*, 770–778. IEEE Computer Society.

He, Zhiyuan, Yijun Yang, Pin-Yu Chen, Qiang Xu, and Tsung-Yi Ho. 2022. Be your own neighborhood: Detecting adversarial example by the neighborhood relations built on self-supervised learning. *arXiv preprint arXiv:2209.00005*.

Hejna, Joey, Rafael Rafailov, Harshit Sikchi, Chelsea Finn, Scott Niekum, W Bradley Knox, and Dorsa Sadigh. 2023. Contrastive prefence learning: Learning from human feedback without rl. *arXiv preprint arXiv:2310.13639*.

Hendel, Roee, Mor Geva, and Amir Globerson. 2023. In-context learning creates task vectors. *arXiv preprint arXiv:2310.15916*.

Hendrycks, Dan, and Thomas G. Dietterich. 2019. Benchmarking neural network robustness to common corruptions and perturbations. In *7th international con-*

ference on learning representations, ICLR 2019, new orleans, la, usa, may 6-9, 2019. OpenReview.net.

Hendrycks, Dan, and Kevin Gimpel. 2016. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *arXiv preprint arXiv:1610.02136*.

Hendrycks, Dan, Kimin Lee, and Mantas Mazeika. 2019. Using pre-training can improve model robustness and uncertainty. In *Proceedings of the 36th international conference on machine learning, ICML 2019, 9-15 june 2019, long beach, california, USA*, ed. Kamalika Chaudhuri and Ruslan Salakhutdinov, vol. 97 of *Proceedings of Machine Learning Research*, 2712–2721. PMLR.

Hessel, Jack, Ana Marasović, Jena D Hwang, Lillian Lee, Jeff Da, Rowan Zellers, Robert Mankoff, and Yejin Choi. 2022. Do androids laugh at electric sheep? humor" understanding" benchmarks from the new yorker caption contest. *arXiv preprint arXiv:2209.06293*.

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

Hu, Shu, Yiming Ying, Xin Wang, and Siwei Lyu. 2022. Sum of ranked range loss for supervised learning. *Journal of Machine Learning Research* 23(112):1–44.

Huang, Jie, Wei Ping, Peng Xu, Mohammad Shoeybi, Kevin Chen-Chuan Chang, and Bryan Catanzaro. 2024. Raven: In-context learning with retrieval-augmented encoder-decoder language models. 2308.07922.

Huang, Rongjie, Jia-Bin Huang, Dongchao Yang, Yi Ren, Luping Liu, Mingze Li, Zhenhui Ye, Jinglin Liu, Xiaoyue Yin, and Zhou Zhao. 2023. Make-an-audio: Text-to-audio generation with prompt-enhanced diffusion models. *ArXiv* abs/2301.12661.

Ilyas, Andrew, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. 2019. Adversarial examples are not bugs, they are features. *Advances in neural information processing systems* 32.

Isserlis, Leon. 1918. On a formula for the product-moment coefficient of any order of a normal frequency distribution in any number of variables. *Biometrika* 12(1/2): 134–139.

Izacard, Gautier, and Edouard Grave. 2020. Leveraging passage retrieval with generative models for open domain question answering. *ArXiv* abs/2007.01282.

Izacard, Gautier, Patrick Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. 2022. Atlas: Few-shot learning with retrieval augmented language models. 2208.03299.

Jacot, Arthur, Franck Gabriel, and Clement Hongler. 2018. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in neural information processing systems*, ed. S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, vol. 31. Curran Associates, Inc.

Jamieson, Kevin G, Lalit Jain, Chris Fernandez, Nicholas J Glattard, and Rob Nowak. 2015. Next: A system for real-world development, evaluation, and application of active learning. In *Advances in neural information processing systems*, 2656–2664.

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

Joachims, Thorsten, et al. 1999. Transductive inference for text classification using support vector machines. In *Icml*, vol. 99, 200–209.

Joshi, Mandar, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th annual meeting of the association for computational linguistics (volume 1: Long papers)*, ed. Regina Barzilay and Min-Yen Kan, 1601–1611. Vancouver, Canada: Association for Computational Linguistics.

Kato, Masahiro, Zhenghang Cui, and Yoshihiro Fukuhara. 2020. Atro: Adversarial training with a rejection option. *arXiv preprint arXiv:2010.12905*.

King, Ben, Rahul Jha, Dragomir Radev, and Robert Mankoff. 2013. Random walk factoid annotation for collective discourse. In *Proceedings of the 51st annual meeting of the association for computational linguistics (volume 2: Short papers)*, 249–254.

Kirk, Robert, Ishita Mediratta, Christoforos Nalmpantis, Jelena Luketina, Eric Hambro, Edward Grefenstette, and Roberta Raileanu. 2023. Understanding the effects of rlhf on llm generalisation and diversity. *arXiv preprint arXiv:2310.06452*.

Koh, Pang Wei, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Irena Gao, et al. 2021. Wilds: A benchmark of in-the-wild distribution shifts. In *International conference on machine learning*, 5637–5664. PMLR.

Kolter, Zico, and Aleksander Madry. 2018. Adversarial Robustness - Theory and Practice. https://adversarial-ml-tutorial.org/.

Krizhevsky, Alex, Geoffrey Hinton, et al. 2009. Learning multiple layers of features from tiny images.

Kurakin, Alexey, Ian J. Goodfellow, and Samy Bengio. 2017. Adversarial examples in the physical world. In *5th international conference on learning representations, ICLR 2017, toulon, france, april 24-26, 2017, workshop track proceedings*. OpenReview.net.

Kwiatkowski, Tom, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M.

Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics* 7:452–466.

Laidlaw, Cassidy, and Soheil Feizi. 2019. Playing it safe: Adversarial robustness with an abstain option. *arXiv preprint arXiv:1911.11253*.

LeCun, Yann. 1998. The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*.

Lee, Harrison, Samrat Phatale, Hassan Mansoor, Kellie Lu, Thomas Mesnard, Colton Bishop, Victor Carbune, and Abhinav Rastogi. 2023. Rlaif: Scaling reinforcement learning from human feedback with ai feedback. *arXiv preprint arXiv:2309.00267*.

Levy, Mosh, Alon Jacoby, and Yoav Goldberg. 2024. Same task, more tokens: the impact of input length on the reasoning performance of large language models. *arXiv preprint arXiv:2402.14848*.

Levy, Shahar, Nir Mazor, Lihi Shalmon, Michael Hassid, and Gabriel Stanovsky. 2025. More documents, same length: Isolating the challenge of multiple documents in rag. *arXiv preprint arXiv:2503.04388*.

Lewis, Patrick, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020a. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Proceedings of the 34th international conference on neural information processing systems*. NIPS '20, Red Hook, NY, USA: Curran Associates Inc.

Lewis, Patrick, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020b. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems* 33:9459–9474.

Li, Chaofan, Zheng Liu, Shitao Xiao, and Yingxia Shao. 2023. Making large language models a better foundation for dense retrieval. *ArXiv* abs/2312.15503.

Li, Jiwei, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2015. A diversity-promoting objective function for neural conversation models. *arXiv preprint arXiv:1510.03055*.

Li, Junlong, Jinyuan Wang, Zhuosheng Zhang, and Hai Zhao. 2022. Self-prompting large language models for zero-shot open-domain qa. *arXiv preprint arXiv:2212.08635*.

Li, Xiaonan, and Xipeng Qiu. 2023. Mot: Pre-thinking and recalling enable chatgpt to self-improve with memory-of-thoughts. *CoRR*.

Li, Xingxuan, Ruochen Zhao, Yew Ken Chia, Bosheng Ding, Shafiq Joty, Soujanya Poria, and Lidong Bing. 2024. Chain-of-knowledge: Grounding large language models via dynamic knowledge adapting over heterogeneous sources. In *The twelfth international conference on learning representations*.

Littlestone, Nick, and Manfred Warmuth. 1986. Relating data compression and learnability.

Liu, Haotian, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2024. Visual instruction tuning. *Advances in neural information processing systems* 36.

Liu, Jiachang, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. 2021. What makes good in-context examples for gpt-3? *arXiv preprint arXiv:2101.06804*.

Liu, Tianqi, Yao Zhao, Rishabh Joshi, Misha Khalman, Mohammad Saleh, Peter J Liu, and Jialu Liu. 2023. Statistical rejection sampling improves preference optimization. *arXiv preprint arXiv:2309.06657*.

Liu, Yanpei, Xinyun Chen, Chang Liu, and Dawn Song. 2016. Delving into transferable adversarial examples and black-box attacks. *CoRR* abs/1611.02770. 1611.02770.

Lorraine, Jonathan, and David Duvenaud. 2018. Stochastic hyperparameter optimization through hypernetworks. *CoRR* abs/1802.09419.

Lu, Sheng, Irina Bigoulaeva, Rachneet Sachdeva, Harish Tayyar Madabushi, and Iryna Gurevych. 2023. Are emergent abilities in large language models just in-context learning? *arXiv preprint arXiv:2309.01809*.

Luo, Man, Xin Xu, Yue Liu, Panupong Pasupat, and Mehran Kazemi. 2024. In-context learning with retrieved demonstrations for language models: A survey. *arXiv preprint arXiv:2401.11624*.

Lyu, Xinxi, Sewon Min, Iz Beltagy, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2022. Z-icl: Zero-shot in-context learning with pseudo-demonstrations. *arXiv preprint arXiv:2212.09865*.

Madry, Aleksander, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2018a. Towards deep learning models resistant to adversarial attacks. In *6th international conference on learning representations, conference track proceedings*. OpenReview.net.

———. 2018b. Towards deep learning models resistant to adversarial attacks. In *6th international conference on learning representations, ICLR 2018, vancouver, bc, canada, april 30 - may 3, 2018, conference track proceedings*. OpenReview.net.

Mann, Ben, N Ryder, M Subbiah, J Kaplan, P Dhariwal, A Neelakantan, P Shyam, G Sastry, A Askell, S Agarwal, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165* 1.

Mason, Blake, Lalit Jain, Ardhendu Tripathy, and Robert Nowak. 2020. Finding all ϵ-good arms in stochastic bandits. *Advances in Neural Information Processing Systems* 33:20707–20718.

Meng, Rui, Ye Liu, Shafiq Rayhan Joty, Caiming Xiong, Yingbo Zhou, and Semih Yavuz. 2024. Sfr-embedding-mistral:enhance text retrieval with transfer learning. Salesforce AI Research Blog.

Min, Sewon, Mike Lewis, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2021. Metaicl: Learning to learn in context. *arXiv preprint arXiv:2110.15943*.

Min, Sewon, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022. Rethinking the role of demonstrations: What makes in-context learning work? *arXiv preprint arXiv:2202.12837*.

Montasser, Omar, Steve Hanneke, and Nathan Srebro. 2019. Vc classes are adversarially robustly learnable, but only improperly. In *Conference on learning theory*, 2512–2530. PMLR.

———. 2021. Transductive robust learning guarantees. *arXiv preprint arXiv:2110.10602*.

Moosavi-Dezfooli, Seyed-Mohsen, Alhussein Fawzi, and Pascal Frossard. 2016. Deepfool: A simple and accurate method to fool deep neural networks. In *2016 IEEE conference on computer vision and pattern recognition, CVPR 2016, las vegas, nv, usa, june 27-30, 2016*, 2574–2582. IEEE Computer Society.

Moosavi-Dezfooli, Seyed-Mohsen, Alhussein Fawzi, and Pascal Frossard. 2016. Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of the ieee conference on computer vision and pattern recognition*, 2574–2582.

Moran, Shay, and Amir Yehudayoff. 2016. Sample compression schemes for vc classes. *Journal of the ACM (JACM)* 63(3):1–10.

Mukobi, Gabriel, Peter Chatain, Su Fong, Robert Windesheim, Gitta Kutyniok, Kush Bhatia, and Silas Alberti. 2023. Superhf: Supervised iterative learning from human feedback. *arXiv preprint arXiv:2310.16763*.

Munos, Rémi, Michal Valko, Daniele Calandriello, Mohammad Gheshlaghi Azar, Mark Rowland, Zhaohan Daniel Guo, Yunhao Tang, Matthieu Geist, Thomas Mesnard, Andrea Michi, et al. 2023. Nash learning from human feedback. *arXiv preprint arXiv:2312.00886*.

Nakano, Reiichiro, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, et al. 2021. Webgpt: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332*.

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

Palan, Stefan, and Christian Schitter. 2018. Prolific. ac—a subject pool for online experiments. *Journal of Behavioral and Experimental Finance* 17:22–27.

Pang, Tianyu, Huishuai Zhang, Di He, Yinpeng Dong, Hang Su, Wei Chen, Jun Zhu, and Tie-Yan Liu. 2022. Two coupled rejection metrics can tell adversarial examples apart. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, 15223–15233.

Peng, ShengYun, Weilin Xu, Cory Cornelius, Matthew Hull, Kevin Li, Rahul Duggal, Mansi Phute, Jason Martin, and Duen Horng Chau. 2023. Robust principles: Architectural design principles for adversarially robust cnns. 2308.16258.

Petersen, Kaare Brandt, Michael Syskind Pedersen, et al. 2008. The matrix cookbook. *Technical University of Denmark* 7(15):510.

Radev, Dragomir, Amanda Stent, Joel Tetreault, Aasish Pappu, Aikaterini Iliakopoulou, Agustin Chanfreau, Paloma de Juan, Jordi Vallmitjana, Alejandro Jaimes, Rahul Jha, et al. 2015. Humor in collective discourse: Unsupervised funniness detection in the new yorker cartoon caption contest. *arXiv preprint arXiv:1506.08126*.

Radford, Alec, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, 8748–8763. PMLR.

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

Ram, Ori, Yoav Levine, Itay Dalmedigos, Dor Muhlgay, Amnon Shashua, Kevin Leyton-Brown, and Yoav Shoham. 2023. In-context retrieval-augmented language models. *Transactions of the Association for Computational Linguistics* 11:1316–1331.

Ramos, Rita Parada, Bruno Martins, Desmond Elliott, and Yova Kementchedjhieva. 2022. Smallcap: Lightweight image captioning prompted with retrieval augmentation. *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* 2840–2849.

Redmon, Joseph, Santosh Divvala, Ross Girshick, and Ali Farhadi. 2016. You only look once: Unified, real-time object detection. In *Proceedings of the ieee conference on computer vision and pattern recognition*, 779–788.

Reimers, Nils, and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Rosset, Corby, Ching-An Cheng, Arindam Mitra, Michael Santacroce, Ahmed Awadallah, and Tengyang Xie. 2024. Direct nash optimization: Teaching language models to self-improve with general preferences. *arXiv preprint arXiv:2404.03715*.

Sarto, Sara, Marcella Cornia, Lorenzo Baraldi, and Rita Cucchiara. 2022. Retrieval-augmented transformer for image captioning. *Proceedings of the 19th International Conference on Content-based Multimedia Indexing*.

Schapire, Robert E, and Yoav Freund. 2012. Boosting. adaptive computation and machine learning. *MIT Press, Cambridge, MA* 1(1.2):9.

Schmidt, Ludwig, Shibani Santurkar, Dimitris Tsipras, Kunal Talwar, and Aleksander Madry. 2018. Adversarially robust generalization requires more data. In *Advances in neural information processing systems*, 5014–5026.

Schulman, John, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Sehwag, Vikash, Shiqi Wang, Prateek Mittal, and Suman Jana. 2020. HYDRA: pruning adversarially robust neural networks. In *Advances in neural information processing systems 33: Annual conference on neural information processing systems 2020, neurips 2020, december 6-12, 2020, virtual*, ed. Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin.

Shahaf, Dafna, Eric Horvitz, and Robert Mankoff. 2015. Inside jokes: Identifying humorous cartoon captions. In *Proceedings of the 21th acm sigkdd international conference on knowledge discovery and data mining*, 1065–1074.

Shalev-Shwartz, Shai, and Shai Ben-David. 2014. *Understanding machine learning: From theory to algorithms*. Cambridge university press.

Sharma, Archit, Sedrick Keh, Eric Mitchell, Chelsea Finn, Kushal Arora, and Thomas Kollar. 2024. A critical evaluation of ai feedback for aligning large language models. *arXiv preprint arXiv:2402.12366*.

Sheikholeslami, Fatemeh, Wan-Yi Lin, Jan Hendrik Metzen, Huan Zhang, and J Zico Kolter. 2022. Denoised smoothing with sample rejection for robustifying pretrained classifiers. In *Workshop on trustworthy and socially responsible machine learning, neurips 2022*.

Sheikholeslami, Fatemeh, Ali Lotfi, and J Zico Kolter. 2020. Provably robust classification of adversarial examples with detection. In *International conference on learning representations*.

Shi, Peng, Rui Zhang, He Bai, and Jimmy Lin. 2022. Xricl: Cross-lingual retrieval-augmented in-context learning for cross-lingual text-to-sql semantic parsing. *arXiv preprint arXiv:2210.13693*.

Sievert, Scott, Daniel Ross, Lalit Jain, Kevin Jamieson, Robert Nowak, and Robert Mankoff. 2017. Next: A system to easily connect crowdsourcing and adaptive data collection. In *Scipy*, 113–119.

Sinha, Aman, Hongseok Namkoong, and John C. Duchi. 2018. Certifying some distributional robustness with principled adversarial training. In *6th international conference on learning representations, ICLR 2018, vancouver, bc, canada, april 30 - may 3, 2018, conference track proceedings*. OpenReview.net.

Siththaranjan, Anand, Cassidy Laidlaw, and Dylan Hadfield-Menell. 2023. Distributional preference learning: Understanding and accounting for hidden context in rlhf. *arXiv preprint arXiv:2312.08358*.

Song, Feifan, Bowen Yu, Minghao Li, Haiyang Yu, Fei Huang, Yongbin Li, and Houfeng Wang. 2024. Preference ranking optimization for human alignment. In *Proceedings of the aaai conference on artificial intelligence*, vol. 38, 18990–18998.

Sotgiu, Angelo, Ambra Demontis, Marco Melis, Battista Biggio, Giorgio Fumera, Xiaoyi Feng, and Fabio Roli. 2020. Deep neural rejection against adversarial examples. *EURASIP Journal on Information Security* 2020(1):1–10.

Stallkamp, Johannes, Marc Schlipsing, Jan Salmen, and Christian Igel. 2012. Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition. *Neural Networks* 32:323–332.

Stasaski, Katherine, and Marti A Hearst. 2022. Semantic diversity in dialogue with natural language inference. *arXiv preprint arXiv:2205.01497*.

Stutz, David, Matthias Hein, and Bernt Schiele. 2020. Confidence-calibrated adversarial training: Generalizing to unseen attacks. In *International conference on machine learning*, 9155–9166. PMLR.

Sun, Yu, Xiaolong Wang, Liu Zhuang, John Miller, Moritz Hardt, and Alexei A. Efros. 2020. Test-time training with self-supervision for generalization under distribution shifts. In *Icml*.

Swamy, Gokul, Christoph Dann, Rahul Kidambi, Zhiwei Steven Wu, and Alekh Agarwal. 2024. A minimaximalist approach to reinforcement learning from human feedback. *arXiv preprint arXiv:2401.04056*.

Tanczos, Ervin, Robert Nowak, and Bob Mankoff. 2017. A kl-lucb bandit algorithm for large-scale crowdsourcing. In *Proceedings of the 31st international conference on neural information processing systems*, 5896–5905.

Tang, Yunhao, Zhaohan Daniel Guo, Zeyu Zheng, Daniele Calandriello, Rémi Munos, Mark Rowland, Pierre Harvey Richemond, Michal Valko, Bernardo Ávila Pires, and Bilal Piot. 2024. Generalized preference optimization: A unified approach to offline alignment. *arXiv preprint arXiv:2402.05749*.

Team, Gemini, Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. 2023. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*.

Touvron, Hugo, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Tramèr, Florian. 2022. Detecting adversarial examples is (nearly) as hard as classifying them. In *International conference on machine learning*, 21692–21702. PMLR.

Tramèr, Florian, Nicholas Carlini, Wieland Brendel, and Aleksander Madry. 2020. On adaptive attacks to adversarial example defenses. In *Advances in neural information processing systems 33: Annual conference on neural information processing systems 2020, neurips 2020, december 6-12, 2020, virtual*, ed. Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin.

Tramer, Florian, Nicholas Carlini, Wieland Brendel, and Aleksander Madry. 2020. On adaptive attacks to adversarial example defenses. *Advances in Neural Information Processing Systems* 33:1633–1645.

Tramèr, Florian, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. 2017. The space of transferable adversarial examples. *arXiv*.

Vapnik, Vladimir. 1998. *Statistical learning theory*. Wiley.

———. 2006. *Estimation of dependences based on empirical data*. Springer Science & Business Media.

Vapnik, Vladimir Naumovich, Vlamimir Vapnik, et al. 1998. *Statistical learning theory*. wiley New York.

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30.

Wang, Dequan, An Ju, Evan Shelhamer, David Wagner, and Trevor Darrell. 2021. Fighting gradients with gradients: Dynamic defenses against adversarial attacks. *arXiv preprint arXiv:2105.08714*.

Wang, Minzheng, Longze Chen, Cheng Fu, Shengyi Liao, Xinghua Zhang, Bingli Wu, Haiyang Yu, Nan Xu, Lei Zhang, Run Luo, et al. 2024a. Leave no document behind: Benchmarking long-context llms with extended multi-doc qa. *arXiv preprint arXiv:2406.17419*.

Wang, Qixun, Yifei Wang, Yisen Wang, and Xianghua Ying. 2024b. Can in-context learning really generalize to out-of-distribution tasks? *arXiv preprint arXiv:2410.09695*.

Wang, Yisen, Difan Zou, Jinfeng Yi, James Bailey, Xingjun Ma, and Quanquan Gu. 2020. Improving adversarial robustness requires revisiting misclassified examples. In *8th international conference on learning representations, ICLR 2020, addis ababa, ethiopia, april 26-30, 2020*. OpenReview.net.

Wang, Yufei, Wanjun Zhong, Liangyou Li, Fei Mi, Xingshan Zeng, Wenyong Huang, Lifeng Shang, Xin Jiang, and Qun Liu. 2023a. Aligning large language models with human: A survey. *arXiv preprint arXiv:2307.12966*.

Wang, Zekai, Tianyu Pang, Chao Du, Min Lin, Weiwei Liu, and Shuicheng Yan. 2023b. Better diffusion models further improve adversarial training. 2302.04638.

Wei, Jason, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2021. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*.

von Werra, Leandro, Younes Belkada, Lewis Tunstall, Edward Beeching, Tristan Thrush, and Nathan Lambert. TRL: Transformer Reinforcement Learning.

Wilson, Garrett, and Diane J Cook. 2020. A survey of unsupervised deep domain adaptation. *ACM Transactions on Intelligent Systems and Technology (TIST)* 11(5): 1–46.

Wong, Eric, Leslie Rice, and J. Zico Kolter. 2020. Fast is better than free: Revisiting adversarial training. In *8th international conference on learning representations, ICLR 2020, addis ababa, ethiopia, april 26-30, 2020*. OpenReview.net.

Wu, Dongxian, Shu-Tao Xia, and Yisen Wang. 2020a. Adversarial weight perturbation helps robust generalization. In *Advances in neural information processing systems 33: Annual conference on neural information processing systems 2020, neurips 2020, december 6-12, 2020, virtual*, ed. Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin.

Wu, Yi-Hsuan, Chia-Hung Yuan, and Shan-Hung Wu. 2020b. Adversarial robustness via runtime masking and cleansing. In *Proceedings of the 37th international conference on machine learning, ICML 2020, 13-18 july 2020, virtual event*, vol. 119 of *Proceedings of Machine Learning Research*, 10399–10409. PMLR.

Wu, Yi-Hsuan, Chia-Hung Yuan, and Shan-Hung Wu. 2020c. Adversarial robustness via runtime masking and cleansing. In *Proceedings of the 37th international conference on machine learning*, ed. Hal Daumé III and Aarti Singh, vol. 119 of *Proceedings of Machine Learning Research*, 10399–10409. PMLR.

Wu, Zeqiu, Yushi Hu, Weijia Shi, Nouha Dziri, Alane Suhr, Prithviraj Ammanabrolu, Noah A Smith, Mari Ostendorf, and Hannaneh Hajishirzi. 2024. Fine-grained human feedback gives better rewards for language model training. *Advances in Neural Information Processing Systems* 36.

Xiang, Chong, Tong Wu, Zexuan Zhong, David Wagner, Danqi Chen, and Prateek Mittal. 2024. Certifiably robust rag against retrieval corruption. *arXiv preprint arXiv:2405.15556*.

Xie, Sang Michael, Aditi Raghunathan, Percy Liang, and Tengyu Ma. 2021. An explanation of in-context learning as implicit bayesian inference. *arXiv preprint arXiv:2111.02080*.

Xu, Canwen, Corby Rosset, Luciano Del Corro, Shweti Mahajan, Julian McAuley, Jennifer Neville, Ahmed Hassan Awadallah, and Nikhil Rao. 2023. Contrastive post-training large language models on data curriculum. *arXiv preprint arXiv:2310.02263*.

Xu, Fangyuan, Weijia Shi, and Eunsol Choi. 2024. RECOMP: Improving retrieval-augmented LMs with context compression and selective augmentation. In *The twelfth international conference on learning representations*.

Yang, Fanny, Aaditya Ramdas, Kevin G Jamieson, and Martin J Wainwright. 2017. A framework for multi-a (rmed)/b (andit) testing with online fdr control. *Advances in Neural Information Processing Systems* 30.

Yang, Jingkang, Kaiyang Zhou, Yixuan Li, and Ziwei Liu. 2024. Generalized out-of-distribution detection: A survey. *International Journal of Computer Vision* 1–28.

Yang, Kevin, Dan Klein, Asli Celikyilmaz, Nanyun Peng, and Yuandong Tian. 2023. Rlcd: Reinforcement learning from contrast distillation for language model alignment. *arXiv preprint arXiv:2307.12950*.

Yang, Sohee, and Minjoon Seo. 2020. Is retriever merely an approximator of reader? *arXiv preprint arXiv:2010.10999*.

Ye, Jiacheng, Zhiyong Wu, Jiangtao Feng, Tao Yu, and Lingpeng Kong. 2023. Compositional exemplars for in-context learning. In *International conference on machine learning*, 39818–39833. PMLR.

Yin, Yueqin, Zhendong Wang, Yi Gu, Hai Huang, Weizhu Chen, and Mingyuan Zhou. 2024. Relative preference optimization: Enhancing llm alignment through contrasting responses across identical and diverse prompts. *arXiv preprint arXiv:2402.10958*.

Yoran, Ori, Tomer Wolfson, Ori Ram, and Jonathan Berant. 2023. Making retrieval-augmented language models robust to irrelevant context. *arXiv preprint arXiv:2310.01558*.

Yosinski, Jason, Jeff Clune, Yoshua Bengio, and Hod Lipson. 2014. How transferable are features in deep neural networks? *Advances in neural information processing systems* 27.

Yuan, Weizhe, Richard Yuanzhe Pang, Kyunghyun Cho, Sainbayar Sukhbaatar, Jing Xu, and Jason Weston. 2024. Self-rewarding language models. *arXiv preprint arXiv:2401.10020*.

Yuan, Zheng, Hongyi Yuan, Chuanqi Tan, Wei Wang, Songfang Huang, and Fei Huang. 2023. Rrhf: Rank responses to align language models with human feedback without tears. *arXiv preprint arXiv:2304.05302*.

Zhang, Hongyang, Yaodong Yu, Jiantao Jiao, Eric Xing, Laurent El Ghaoui, and Michael Jordan. 2019a. Theoretically principled trade-off between robustness and accuracy. In *International conference on machine learning*, 7472–7482. PMLR.

Zhang, Hongyang, Yaodong Yu, Jiantao Jiao, Eric P. Xing, Laurent El Ghaoui, and Michael I. Jordan. 2019b. Theoretically principled trade-off between robustness and accuracy. In *Proceedings of the 36th international conference on machine learning*,

ICML 2019, 9-15 june 2019, long beach, california, USA, ed. Kamalika Chaudhuri and Ruslan Salakhutdinov, vol. 97 of *Proceedings of Machine Learning Research*, 7472–7482. PMLR.

Zhang, Ruiqi, Spencer Frei, and Peter L Bartlett. 2024. Trained transformers learn linear models in-context. *Journal of Machine Learning Research* 25(49):1–55.

Zhang, Zhuosheng, Aston Zhang, Mu Li, and Alex Smola. 2022. Automatic chain of thought prompting in large language models. *arXiv preprint arXiv:2210.03493*.

Zhao, Penghao, Hailin Zhang, Qinhan Yu, Zhengren Wang, Yunteng Geng, Fangcheng Fu, Ling Yang, Wentao Zhang, Jie Jiang, and Bin Cui. 2024a. Retrieval-augmented generation for ai-generated content: A survey. *arXiv preprint arXiv:2402.19473*.

Zhao, Yao, Rishabh Joshi, Tianqi Liu, Misha Khalman, Mohammad Saleh, and Peter J Liu. 2023. Slic-hf: Sequence likelihood calibration with human feedback. *arXiv preprint arXiv:2305.10425*.

Zhao, Yufeng, Yoshihiro Sakai, and Naoya Inoue. 2024b. Noisyicl: A little noise in model parameters calibrates in-context learning. *arXiv preprint arXiv:2402.05515*.

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

Zhu, Banghua, Evan Frick, Tianhao Wu, Hanlin Zhu, and Jiantao Jiao. 2023. Starling-7b: Improving llm helpfulness & harmlessness with rlaif.

Zhuang, Fuzhen, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. 2020. A comprehensive survey on transfer learning. *Proceedings of the IEEE* 109(1):43–76.

Ziegler, Daniel M, Nisan Stiennon, Jeffrey Wu, Tom B Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. 2019. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*.

Zimmermann, Roland S., Wieland Brendel, Florian Tramer, and Nicholas Carlini. 2022. Increasing confidence in adversarial robustness evaluations. In *Advances in neural information processing systems*, ed. S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, vol. 35, 13174–13189. Curran Associates, Inc.