Diss. ETH No. 28412

ADVANCING COST EFFICIENCY AND ROBUSTNESS OF MACHINE LEARNING THROUGH THE LENS OF DATA

A thesis submitted to attain the degree of

DOCTOR OF SCIENCES of ETH ZURICH

(Dr. sc. ETH Zurich)

presented by

NEZİHE MERVE GÜREL

MSc in Computer and Communication Sciences, EPFL

born on 09.05.1990

citizen of Turkey

Accepted on the recommendation of

Prof. Dr. Ce Zhang (ETH Zurich), examiner
Prof. Dr. Andreas Krause (ETH Zurich), co-examiner
Prof. Dr. Markus Püschel (ETH Zurich), co-examiner
Prof. Dr. Umut Şimşekli (INRIA), co-examiner

2022

ADVANCING COST EFFICIENCY AND ROBUSTNESS OF MACHINE LEARNING THROUGH THE LENS OF DATA

ABSTRACT

Machine Learning (ML) systems contend with an ever-growing processing load of physical world data. These systems are required to deliver high-quality learning and decision-making *often constrained by limited resources*. This need has led to a proliferation of optimization techniques at model and implementation levels over the past decades. The model and implementation-focused nature of these techniques, however, challenges their generalizability across different application domains and different stages of the ML pipeline where the problem may be as acute. This dissertation identifies several open problems to which current cost-optimization strategies do not directly apply or are ineffective, and offers theoretically sound and repeatable strategies that maintain practical performance without any discernible loss in quality. These strategies adopt a data-focused view to reduce dependency on the learner, and enhance the cost-effectiveness of ML pipelines by reducing the amount of data to process and their robustness through supplying domain knowledge in replacement of robust training data. The contributions of this dissertation are threefold:

Hardware-Efficiency via Data Quantization

First, we focus on *hardware efficiency* and investigate training with *low precision data representation* to accelerate the processing of compute-intensive workloads on hardware. Inspired by the number of application domains associated with it, we focus on sparse signal reconstruction problems where compressive sensing can be employed. By lowering the data precision and co-designing the reconstruction algorithm, we show that compressive sensing can be significantly accelerated on hardware such as Field Programmable Gate Arrays (FPGA) and Central Processing Units (CPU) with negligible loss of reconstruction quality. We develop theory which analyzes the scaling of recovery error with respect to bit precision, and empirically demonstrate the benefit of low precision compressive sensing in the context of real-world applications.

Label-Efficiency via Data Sampling

Next, we move our attention to labor-intensive workloads across the ML pipeline. We specifically focus on the post-training stages — which often encounter a mismatch between the distributions of production and training data, and requires curation for it. To account for that in a labor-efficient manner, we introduce an active model selection strategy for pretrained models where the best pretrained model for the downstream task can be found by labeling only a small portion of freshly collected production data. We show that such a specialized *data sampling* strategy can significantly improve *label efficiency* at the later stages of the ML pipeline by accounting for the production data shift. Closely related to the contribution of model selection, we also study the oversmoothing in graph neural networks and rigorously identify the role of architectural model differences in terms of graph decomposition.

Robustness via Knowledge Integration

The final contribution of this thesis is on the ML robustness front, where we improve *adversarial robustness* by using *domain knowledge*. In particular, we develop a knowledge enhanced ML pipeline, the first framework that integrates domain knowledge to enhance the adversarial *robustness* of ML classifiers against a diverse set of attacks throughout the pipeline. Our framework is generic, efficient, and can be applied at different stages of the ML pipeline. From the perspective of trustworthy ML, we show that domain knowledge, as a robust and tenable proxy of data, can mimic the robust features relating to the prediction variable and provide a defense whose robustness is agnostic to the type of adversary. Finally, we formulate a theoretical foundation to identify the regime of improvement in terms of quality of domain knowledge and demonstrate its practical performance against a diverse collection of attacks.

ZUSAMMENFASSUNG

Machine Learning (ML)-Systeme kämpfen mit einer ständig wachsenden Verarbeitungslast von Daten der physischen Welt. Diese Systeme sind erforderlich, um qualitativ hochwertiges Lernen und Entscheidungsfindung zu liefern *oft eingeschränkt durch begrenzte Ressourcen*. Dieser Bedarf hat in den letzten Jahrzehnten zu einer Verbreitung von Optimierungstechniken auf Modell- und Implementierungsebene geführt. Die modell- und implementierungsorientierte Natur dieser Techniken stellt jedoch ihre Verallgemeinerbarkeit über verschiedene Anwendungsdomänen und verschiedene Stufen der ML-Pipeline in Frage, wo das Problem ebenso akut sein kann. Diese Dissertation identifiziert mehrere offene Probleme, auf die aktuelle Kostenoptimierungsstrategien nicht direkt anwendbar oder unwirksam sind, und bietet theoretisch fundierte und wiederholbare Strategien, die praktische Leistung ohne erkennbaren Qualitätsverlust aufrechterhalten. Diese Strategien verfolgen eine datenorientierte Sichtweise, um die Abhängigkeit vom Lernenden zu verringern und die Kosteneffizienz von ML-Pipelines zu verbessern, indem sie die zu verarbeitende Datenmenge und ihre Robustheit reduzieren, indem Domänenwissen anstelle robuster Trainingsdaten bereitgestellt wird. Die Beiträge dieser Dissertation sind dreifach:

Hardware-Effizienz durch Datenquantisierung

Zuerst konzentrieren wir uns auf *Hardware-Effizienz* und untersuchen das Training mit *Datendarstellung mit niedriger Genauigkeit*, um die Verarbeitung rechenintensiver Arbeitslasten auf Hardware zu beschleunigen. Inspiriert von der Anzahl der damit verbundenen Anwendungsdomänen konzentrieren wir uns auf Probleme der Signalrekonstruktion mit geringer Dichte, bei denen Compression Sensing eingesetzt werden kann. Indem wir die Datenpräzision verringern und den Rekonstruktionsalgorithmus mitgestalten, zeigen wir, dass Compressive Sensing auf Hardware wie Field Programmable Gate Arrays (FPGA) und Central Processing Units (CPU) mit vernachlässigbarem Verlust an Rekonstruktionsqualität erheblich beschleunigt werden kann. Wir entwickeln eine Theorie, die die Skalierung des Wiederherstellungsfehlers

in Bezug auf die Bitpräzision analysiert, und demonstrieren empirisch den Nutzen der Kompressionsmessung mit niedriger Genauigkeit im Kontext realer Anwendungen.

Label-Effizienz durch Daten-Sampling

Als nächstes lenken wir unsere Aufmerksamkeit auf arbeitsintensive Workloads in der ML-Pipeline. Wir konzentrieren uns speziell auf die Post-Training-Phasen – die oft auf eine Diskrepanz zwischen den Verteilungen von Produktions und Trainingsdaten stossen und dafür eine Kuration erfordern. Um dies auf arbeitseffiziente Weise zu berücksichtigen, führen wir eine aktive Modellauswahlstrategie für vortrainierte Modelle ein, bei der das beste vortrainierte Modell für die nachgelagerte Aufgabe gefunden werden kann, indem nur ein kleiner Teil der frisch gesammelten Produktionsdaten gekennzeichnet wird. Wir zeigen, dass eine solche spezialisierte *Daten-Sampling*-Strategie die *Label-Effizienz* in den späteren Phasen der ML-Pipeline erheblich verbessern kann, indem sie die Verschiebung der Produktionsdaten berücksichtigt. Eng verbunden mit dem Beitrag der Modellauswahl untersuchen wir auch die Überglättung in neuronalen Netzwerken von Graphen und identifizieren rigoros die Rolle von Architekturmodellunterschieden in Bezug auf die Graphenzerlegung.

Robustheit durch Wissensintegration

Der letzte Beitrag dieser Doktorarbeit betrifft die ML-Robustheitsfront, wo wir *adversarial robustness* durch Verwendung von *Domänenwissen* verbessern. Insbesondere entwickeln wir eine wissenserweiterte ML-Pipeline, das erste Framework, das Domänenwissen integriert, um die gegnerische *Robustheit* von ML-Klassifikatoren gegenüber einer Reihe von diversen Angriffen zu verbessern Pipeline. Unser Framework ist generisch, effizient und kann in verschiedenen Phasen der ML-Pipeline angewendet werden. Aus der Perspektive eines vertrauenswürdigen ML zeigen wir, dass Domänenwissen als robuster und haltbarer Proxy von Daten die robusten Merkmale in Bezug auf die Vorhersagevariable nachahmen und eine Verteidigung bieten kann, deren Robustheit für den Typ von agnostisch ist Gegner. Schliesslich formulieren wir eine theoretische Grundlage, um das Regime der Verbesserung in Bezug auf die Qualität des Domänenwissens zu identifizieren und seine praktische Leistung gegen eine Vielzahl von Angriffen zu demonstrieren.

ACKNOWLEDGEMENTS

I thank my advisor, Prof. Ce Zhang. Ce, your training significantly enriched my scientific perspective and equipped me with countless new skills. Thank you for giving us your students scientific freedom and motivation to pursue our goals. Thank you for your endless support and never caring that I read so slow, and regardless constantly injecting motivation and confidence. Thank you for everything you did that cannot fit into a single paragraph.

I thank my co-advisor, Prof. Andreas Krause, and committee member Prof. Markus Püschel for the very enjoyable collaborations and technical guidance and support they have provided. I feel nothing less than very grateful to have worked with you throughout this journey. I also thank Prof. Umut Simsekli for being a part of my committee. Thank you, Umut, for your valuable comments and recommending several future directions for this thesis. My very warm thanks go to Prof. Bo Li as well. Thank you, Bo, for your guidance and support in this journey. It has been simply amazing to have worked with you and to have listened to your perspectives on robust learning.

The PhD is a long journey. There are likely to be multiple motivational up and downs. During all those times, I received great support from Prof. Timothy Roscoe. Thank you, Mothy. I also thank Prof. Onur Mutlu for listening to me and sharing his perspectives during those times. Finally, I thank the WiML board of directors for welcoming me into the community and encouraging me to be a part of the change with their great trust.

I am very lucky to be a part of DS3Lab and Systems Group. I thank all of my colleagues and friends within the group. In particular, thank you, Nora, Cédric, Luka, Bojan, Thomas, Fabio and Vojislav for your friendship and for making the PhD life very enjoyable. I also thank our system admins, Jena, Simonetta, and Nadia, who help us every step of the way with the administrative load.

A big thank you to my beloved friends, Irem, Kaan, Alp, and Beliz. Thank you for all the therapeutic gatherings and your friendship throughout this journey as well as in life.

Last but certainly not least, I would like to share my warm thanks with my parents, Muazez and Hilmi, and my brother Kutan. They taught me that the biggest privilege an individual can have is a loving and supportive family. I also thank Regie and Patrick for always welcoming me to the Netherlands with wide-open arms. Bellatrix, my little cat, thank you for always reminding me of the beauty of this world. Finally, I thank my partner and best friend, Simon. I could not have been more happy to have someone like you by my side.

Nezihe Merve Gürel
May 2022

The results enclosed in this dissertation are from previously published work, which are products of joint work with Alen Stojanov, Andreas Krause, Bo Li, Bojan Karlas, Ce Zhang, Cédric Renggli, Dan Alistarh, Johannes Rausch, Kaan Kara, Luka Rimanic, Markus Püschel, Mohammad Reza Karimi, Tyler Smith, and Xiangyu Qi. Although this dissertation carries my name, this collection of research would not have been possible without their contributions.

CONTENTS

III ROBUSTNESS VIA KNOWLEDGE INTEGRATION

LIST OF FIGURES

LIST OF TABLES

ACRONYMS

FoV	Field of View
ML	Machine Learning
GD	Gradient Descent
MI	Mutual Information
GNN	Graph Neural Networks
DNN	Deep Neural Networks
SNR	Signal-to-Noise-Ratio

LOFAR	LOw Frequency ARray
FPGA	Field Programmable Gate Arrays
SGD	Stochastic Gradient Descent
IHT	Iterative Hard Thresholding
GPU	Graphics Processing Units
CPU	Central Processing Units
RIP	Restricted Isometry Property
MRI	Magnetic Resonance Imaging
CNN	Convolutional Neural Networks
GCN	Graph Convolutional Networks
GraphCNN	Graph Convolutional Neural Networks
RICs	Restricted Isometric Constants
QIHT	Quantized Iterative Hard Thresholding
CoSaMP	Compressive Sampling Matching Pursuit
KEMLP	Knowledge Enhanced Machine Learning Pipeline

1 INTRODUCTION

1.1 BACKGROUND AND MOTIVATION

The last decade has witnessed the rapid development of Machine Learning (ML) techniques that are shown to be effective when large amounts of data and compute resources are available. When it comes to the adoption of these techniques into data and compute-intensive real-world applications, however, resources are limited, and learning and decision making are constrained to happen *with limited resources*. In order to tackle this problem, a host of efforts has been brought forward by the ML and system communities under the umbrella of *cost-effective* ML (also known as *resource-efficient* or *budgeted* ML). In broad terms, cost-effective ML is the practice of optimizing the quality of learning and decision making while keeping the cost of utilized resources under a budget. This inescapable interplay between cost and quality constantly prompts ML practitioners into answering: *Does this optimization strategy yield the desired outcome in quality given my budget? Which strategy yields the best outcome given my budget? How optimal is this strategy?* To date, answering these questions remains to be a challenge the ML community faces on a daily basis.

In this section, as preamble to presenting the research questions addressed in this thesis, we review current trends on cost-effectiveness of ML systems. Broadly speaking, the term "cost" may refer to various expenses, including but not limited to data collection, manual labeling, computing power, memory footprint, network-throughput, and power consumption. Considering the model and data to be the core foundations of the ML systems, we specifically focus on the aspects that are often deemed as particularly costly, namely *training on hardware* and *manual labeling of data*. Going beyond the measure of *accuracy*, we also consider *adversarial robustness*, an important attribute of trustworthiness for mission-critical ML tasks, to be the third aspect. After this general overview of current trends and challenges that motivate this thesis, in the subsequent sections, we pose the research questions addressed within its scope, and present its contributions and structure.

1.1.1 *Hardware Efficiency*

Hardware efficiency for ML workloads heavily revolves around designing efficient and scalable techniques that ease the load of training and inference. The rising complexity of ML models, such as deep neural networks, further require ML systems to substantially optimize for the energy efficient use of hardware in light of the large amount of data movement and heavy computation. Recently, several research communities have been working on strategies to achieve this on different optimization levels. The system community has focused on implementation level techniques to improve energy efficiency and computation speed, including data reuse from local memory (Chen, Emer, and Sze, 2016; Sze et al., 2017a; Vanhoucke, Senior, and Mao, 2011); in the context of deep learning, exploiting sparsity to skip unnecessary computational operations (Sze et al., 2017b); reducing number of arithmetic operations (Dubout and Fleuret, 2012; Mathieu, Henaff, and LeCun, 2014) or interventions on computation that results in it (Cong and Xiao, 2014; Lavin and Gray, 2016); mixed signal circuit design (LiKamWa et al., 2016; Murmann et al., 2015; Wang, Schapire, and Verma, 2014; Zhang, Wang, and Verma, 2015, 2016), and recent technologies such as neuromorphic computing and in-memory processing (Izhikevich, 2004). The ML community, on the other hand, develops model level optimization techniques (Lee et al., 2021; Menghani, 2021; Pernkopf et al., 2018) such as model quantization (Alistarh et al., 2017; Courbariaux, Bengio, and David, 2015; De Sa et al., 2015; Gupta et al., 2013; Seide et al., 2014); pruning (Han et al., 2015; LeCun, Denker, and Solla, 1989); compact convolution (Sandler et al., 2018) and knowledge distillation (Hinton, Vinyals, Dean, et al., 2015), to name a few.

The shared objective of both communities has recently resulted in a growing interest in *co-designing* optimization techniques spanning different levels. There are several notable efforts in this direction, including distributed learning (see Liu and Zhang, 2020 and references therein for a comprehensive overview) and co-designs of hardware and model that exploit sparsity as well as compression (Albericio et al., 2016; Chen, Emer, and Sze, 2016; Han et al., 2016; Han, Mao, and Dally, 2015; Yang, Chen, and Sze, 2017). Another undoubtedly prominent outcome of such co-designs is low precision training on hardware such as CPU (De Sa et al., 2015; Noel and Osindero, 2014; Stojanov et al., 2018), Graphics Processing Units (GPU) (Alistarh et al., 2017; Hubara et al., 2017; Noel and Osindero, 2014; Seide et al., 2014) and FPGA (Gupta et al., 2013; Kara et al., 2017; Zhang et al., 2017). In many of these works, low precision training that employs Gradient Descent (GD)-based methods is shown to retain quality of predictions even when both data and model are significantly quantized to as low as 8-bit fixed precision. With little to no loss of performance, compression of bit-widths reduces not only the data movement but also computation cycles and memory requirements during training. Spurred on by its already demonstrated effectiveness in many application domains, we believe that tailoring these co-designs of low precision training and hardware for a more extensive set of inference problems and training strategies can have a profound impact. One such domain which holds great promise is that of modern scientific instruments that employ sensing devices, for instance in the fields of interferometry, medical imaging and remote sensing. These instruments acquire vast amount of high dimensional signals on a daily basis. These factors make such sensing based instruments an opportune candidate for low precision training.

1.1.2 *Label Efficiency*

Despite recent advances on hardware-efficient and scalable training of supervised learning methods mentioned earlier, one challenge still lingers: they are notoriously *data-hungry*. Typically, unlabeled data is abundant and can be easily and inexpensively collected. For many real-world applications, however, labels are slow and expensive to acquire as they require manual annotation, creating a need for ML models to attain good predictive performance in a sample-efficient manner.

The need for sample-efficient learning methods has thus naturally led to rapid growth of strategies over several decades. Semi-supervised learning, for instance, tackles it by pseudo-labeling of unlabeled instances with models trained on the limited labeled data. Similarly, programmatic weak supervision (Ratner et al., 2017) performs noisy labeling of instances via labeling functions introduced by domain experts. Techniques as bootstrapping and data augmentation (Wong et al., 2016), for instance, re-use the existing labelled set to create a more diverse and rich dataset. Active learning, on the other hand, covers the strategies where the learner interactively queries the label of an instance that it finds informative for the learning task at hand — up to a user-defined labeling budget.

One of the limitations of these approaches is that label efficiency is considered merely for improving the *training* of ML classifiers. Yet, there are several post-training data-hungry challenges that ML practitioners and engineers face throughout the iterations of the ML pipeline. In particular, when the trained model enters into the operational phase, data distribution may shift and thereby performance degradation may occur. In order to mitigate the effect of this shift, transfer learning and domain adaptation techniques learn the translation of the trained model on the new adaptation domain. In the case of concept drift, additional queries may be requested by the labeling oracle for retraining the model with the drifted data distribution. The domain robustness of a deployed method, therefore, is often targeted by going back to the modeling block of the ML pipeline. The question remains open as how to avoid repeatedly firing up the expensive ML retraining stage or domain adaptation process when the distribution shift occurs, and instead perform an automatic and repeatable selection of the most suitable model for the drifted production data directly at the deployment level, moreover in label-efficient manner.

1.1.3 *Adversarial Robustness*

ML models are vulnerable to different types of adversarial examples, which are adversarially manipulated inputs aiming to mislead ML models to make arbitrarily incorrect predictions (Bhattad et al., 2020; Eykholt et al., 2018; Goodfellow, Shlens, and Szegedy, 2015; Szegedy et al., 2013). Such attacks can potentially compromise the reliability of an ML system and are particularly threatening when deployed in safety-critical applications such as self-driving cars, medical diagnosis and face recognition in cyber-physical systems.

In response to these threats, recent years have witnessed a rapid growth of empirical defense techniques. A powerful strategy is to train Deep Neural Networks (DNN) over a mixture of clean and adversarial examples, the so-called *adversarial training* (Madry et al., 2017). Adversarial training has shown to be effective, also together with feature quantization (Xu, Evans, and Qi, 2017) and reconstruction approaches (Samangouei, Kabkab, and Chellappa, 2018). There exists also other defenses such as input processing (Ross and Doshi-Velez, 2018) and approaches with certified robustness against perturbation bounded attacks (distinctly from empirical defenses). Notable works include (Cohen, Rosenfeld, and Kolter, 2019; Gehr et al., 2018; Mirman, Gehr, and Vechev, 2018; Yang et al., 2020) and (Balunovic and Vechev, 2020; Mirman et al., 2021; Singh et al., 2018a,b). Despite these advancements, empirical defense techniques still get caught in the nets of trade-off between practical performance and computational efficiency. This problem is particularly pressing when constructing adversarial examples from the clean training data at each iteration of adversarial training with projected gradient descent, which is a separate optimization problem itself. This creates a necessity to access not only the training data and its statistics which are not always available due to data privacy and proprietary rights (Nayak, Rawal, and Chakraborty, 2022), but also compute resources for adversarial training, as it is computationally much less feasible compared to vanilla training of DNN (Dolatabadi, Erfani, and Leckie, 2021; Sriramanan, Addepalli, Baburaj, et al., 2021). To cope with these challenges, on the robust training front, the fast gradient sign method or the use of regularizers are shown to be efficient alternatives to adversarial training (Chang, He, and Li, 2018; Sriramanan, Addepalli, Baburaj, et al., 2021; Wong, Rice, and Kolter, 2020). On the robust data front, several other methods are proposed including robust training with only a subset of training data (Dolatabadi, Erfani, and Leckie, 2021); regular training with a robustness curation on important image pixels (Zhu, Wei, and Zhu, 2021); detecting and correcting adversaries at test-time (Nayak, Rawal, and Chakraborty, 2022) and noisy data augmentation (Liu et al., 2022). However, from the perspective of trustworthiness, these adversarial defenses can still be adaptively attacked again (Athalye, Carlini, and Wagner, 2018; Carlini and Wagner, 2017a) or their robustness is not preserved against other attacks (Kang et al., 2019; Schott et al., 2018). Thus, despite the rapid recent progress on robust learning, it is still challenging to provide general adversarial defenses that are simple yet effective against a diverse set of attacks.

1.2 RESEARCH SCOPE

The efficient use of resources heavily hinges on the data processing workload, and therefore cost-optimization and general robustness are eventually bound to the data to be processed across the ML pipeline. Therefore, we believe that curation of cost inefficiency and costly trustworthy attributes such as robustness through the lens of data holds great potential. Within the scope of this thesis, we adopt such a data-focused perspective and addresses several open challenges towards cost-effective and robust ML from this perspective. Namely, we study different application domains and stages of the ML pipeline where the cost of operations is acute. In particular, we seek to provide answers to the following questions:

Question 1: (Hardware Efficient Compressive Sensing via Quantized Data) Can training compressive sensing-based applications with low precision data representation enable accelerated signal recovery on hardware with recovery guarantees and good practical performance?

Compressive sensing (Candes, Romberg, and Tao, 2006a,b; Donoho, 2006) is a powerful mathematical framework behind many sensing-based scientific instruments. Compressive sensing solvers can learn the sparse representation of analog signals from only a few samples, enabling the efficient collection, processing, and storage of very large amounts of data. In this thesis, we explore the extension of low precision training to compressive sensing with the hardware that accommodates it. As an inspiring fact, data quantization for compressive sensing problems is shown to exhibit a great empirical success as many compressive sensing solvers are tolerant to noise introduced by quantization. Several previous studies have taken advantage of this, decreasing the precision of data representation to as low as a single bit (Ai et al., 2014; Boufounos and Baraniuk, 2008; Jacques et al., 2013; Laska et al., 2011; Plan and Vershynin, 2013a,b) and (Gopi et al., 2013; Gupta et al., 2013). We expand this direction further and investigate the design of a compressive sensing solver which quantizes *all* data *(both measurement matrix and observation vector)*, while imposing a more general set of assumptions on the model than that of most existing work. This *quantize-all* strategy enables us to fully unleash the potential of arithmetic operations for accelerated computation on hardware, as demonstrated by (Kara et al., 2017; Zhang et al., 2017) on FPGA and (Stojanov et al., 2018) on CPU in the context of GD and SGD training. We further investigate the effectiveness of low precision compressive sensing in the context of two real-world data-intensive applications: radio interferometers and magnetic resonance imaging.

Question 2.1: (Label Efficient and Robust Model Selection via Selective Sampling) How can we select the best trained model for a freshly arriving production data stream — in a label-efficient manner?

Real-world data distribution shifts in an uncontrollable way. This problem is especially acute when models are at the production phase and can no longer retain high quality predictions due to the production data shifting away from the original training data. This thus results in a necessity to mitigate the effect of distribution shift into the model *in action*. This scenario is also no foreigner to industrial companies — they often train distinct models on different sliding windows of data and automatically adapt each of these models for new production stream. Consequently, they accumulate a pool of candidate models that are ready to be deployed, and hope to select the best one to make predictions on the new production stream in a *cost-efficient* manner.

Given a manual labeling budget, an effortless solution would be randomly labeling the freshly arriving production data, but it is often scarce when deployed for model selection and may result in an unfair evaluation of classifiers. Application of existing active learning strategies to this scenario, on the other hand, is non-trivial and depart from traditional active learning setting. In this thesis, we address this problem and study label-efficient model selection. In particular, we ask: *Given k pretrained classifiers and a stream of unlabeled data examples, how can we actively decide when to query a label so that we can distinguish the best model from the rest under a limited labeling budget?* To answer this question, we first visit the existing active learning strategies and adapt them for model selection. We then propose a selective sampling strategy that actively selects informative examples to label and upon exceeding a labeling budget, outputs the best model with high probability. We introduce a novel evaluation framework for stream-based setting and illustrate the effectiveness of our proposal.

Related to problem of model selection and motivated by the wide spectrum of applications that uses graph-structured data, we also investigate the model selection problem for graph neural networks. In particular, we attempt to understand why they suffer from performance degradation when the network goes deeper, the so-called *oversmoothing* problem, and how graph decomposition as an architectural (model) change helps prevent it although being exposed to the same training data. This thus leads us to the following research question:

Question 2.2: (Reliable Model Selection for Label Efficiency) How does decomposition help with oversmoothing in Graph Neural Networks?

Extending Convolutional Neural Networks (CNN)s over images to graphs has attracted recent interest, with an early attempt called Graph Convolutional Networks (GCN) model proposed by (Kipf and Welling, 2016a). When applying GCNs to many practical applications, one discrepancy lingers — although a traditional CNN usually achieves higher accuracy when it goes deeper, GCNs, as a natural extension of CNNs, does not seem to benefit much from going

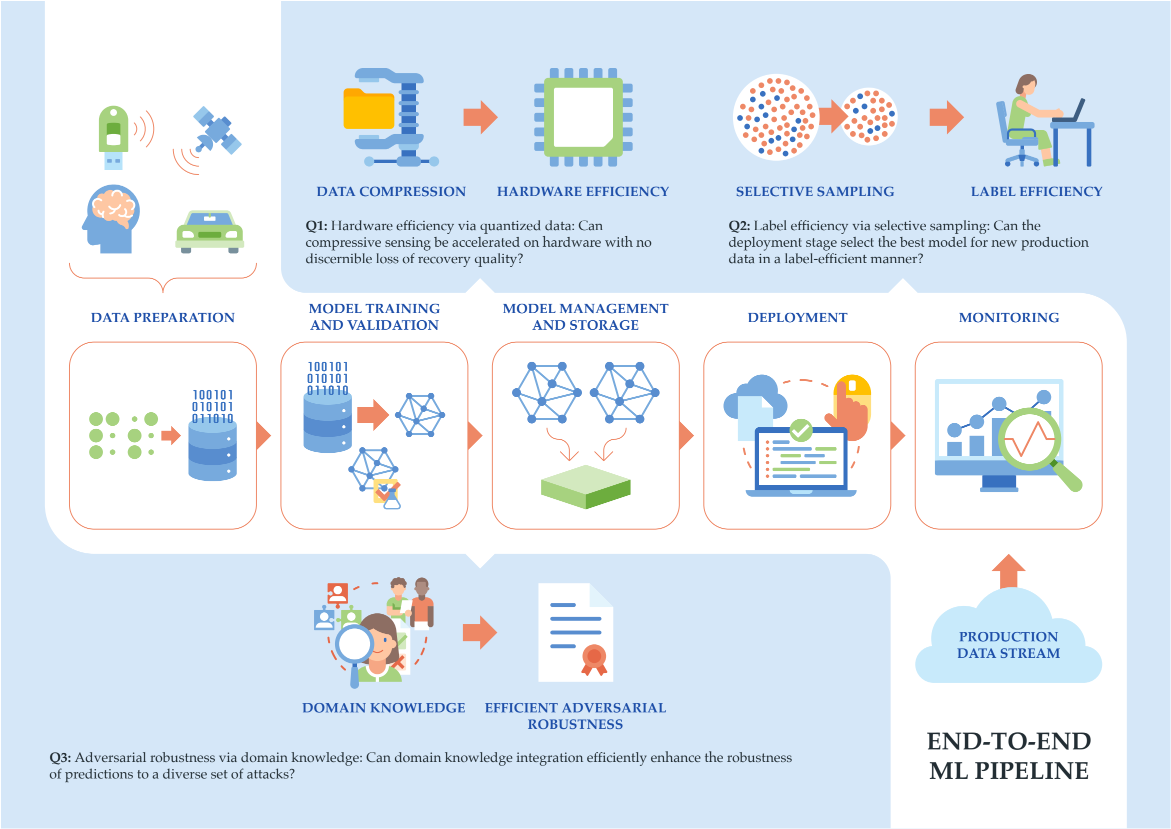

DATA COMPRESSION
HARDWARE EFFICIENCY
SELECTIVE SAMPLING
LABEL EFFICIENCY
Q1: Hardware efficiency via quantized data: Can compressive sensing be accelerated on hardware with no discernible loss of recovery quality?
Q2: Label efficiency via selective sampling: Can the deployment stage select the best model for new production data in a label-efficient manner?
DATA PREPARATION
MODEL TRAINING AND VALIDATION
MODEL MANAGEMENT AND STORAGE
DEPLOYMENT
MONITORING
100101 010101 011010
DOMAIN KNOWLEDGE
EFFICIENT ADVERSARIAL ROBUSTNESS
PRODUCTION DATA STREAM
Q3: Adversarial robustness via domain knowledge: Can domain knowledge integration efficiently enhance the robustness of predictions to a diverse set of attacks?
END-TO-END ML PIPELINE

Figure 1.1: Research scope overview of this thesis. We achieve three different advancements in cost efficiency and robustness: 1: *Hardware efficiency with low precision compressive sensing*, 2: *Label efficient model selection with selective sampling*, and 3: *Adversarial robustness with domain knowledge integration*.

deeper by stacking multiple layers together although being exposed to the same training data.

This phenomenon has been the focus of multiple prior work (Li et al., 2019b; Li, Han, and Wu, 2018; Oono and Suzuki, 2019). On the theoretical side, (Li, Han, and Wu, 2018) and (Oono and Suzuki, 2019) identified the problem as *oversmoothing* — under certain conditions, when multiple GCN layers are stacked together, the output will converge to a region that is independent of weights and inputs. On the empirical side, (Li et al., 2019b) showed that many techniques that were designed to train a deep CNN, for instance, the skip connections in ResNet (Kaiming et al., 2016), can also make it easier for GCN to go deeper. Integrating techniques such as residual connections (ResGCN) and dense connections (DenseGCN) can help accommodate this problem to a certain extend; however, this limitation remains (Kipf and Welling, 2017).

Partitioning the graph with a hand-picked structure can also potentially help a range of tasks. For example, thinking of an image as a graph, if we decompose it into multiple subgraphs, it is possible to design a GCN-variant to implement a standard CNN-like model. Graph Convolutional Neural Networks (GraphCNN) (Such et al., 2017) (distinct from GCN) is one such example of taking advantage of graph decomposition whose performance benefits from going deeper. The benefit of graph decomposition further opens up research questions such as *how should I decompose my graph and set my GCN model?* We take a first step towards answering this question and conduct a theoretical analysis to understand the impact of graph decomposition on the performance of GNN via the lens of information theory. We explore the regimes where oversmoothing occurs in GCN and GraphCNN, and explain how graph decomposition helps with oversmoothing.

Question 3: (Efficient and Reliable Adversarial Robustness) Can we enhance adversarial robustness of ML systems via domain knowledge integration against diverse attacks, without requiring adversarial training or other specialized defenses?

The anatomy of adversarial examples has recently spawned an interest in the ML security community to understand how imperceptibly small perturbations can easily fool the state-of-the-art ML algorithms. The seminal work of Ilyas et al., 2019 investigates this and introduces a conceptual model of adversarial examples, where the features in a dataset are categorized based on their robustness to perturbations and how it translates to the prediction variable. One key observation is that adversarial examples can easily be constructed by crafting certain features without affecting human recognition, and hence the adversarial vulnerability of ML systems can be linked to the presence of such *non-robust* and *human-imperceptible* features. Such a perspective posits adversarial examples as *"human-centric phenomenon"* (Ilyas et al., 2019). This observation has been enabling more advanced adversarial defense strategies, where the concept of robust features is taken into consideration (Zhu, Wei, and Zhu, 2021). Providing a generic and comprehensive treatment to this problem is, however, far from trivial. In this thesis, we take a different perspective towards training robust ML models against adversarial attacks. Given the observation that human with knowledge is quite resilient against these attacks, we integrate domain knowledge during prediction as a proxy of robust and human-perceptible attributes. We further investigate how and when knowledge helps with robustness and attempt to understand how robustness improvement affects the clean accuracy.

1.3 TECHNICAL CONTRIBUTIONS AND STRUCTURE

In Part i of this thesis, we focus on efficient training of compressive sensing solvers on hardware via lowering the data precision. We show that sparsity-constrained minimization methods can enable significant computational speed-up with low precision data, while still maintaining the theoretical guarantees and practical performance under mild constraints:

- We conduct a theoretical analysis of a sparsity-constrained minimization method, the normalized Iterative Hard Thresholding (IHT) algorithm, when all input data, meaning both the measurement matrix and the observation vector, are quantized aggressively. We present a variant of low precision normalized IHT that, under mild conditions, can still provide recovery guarantees.

- We conduct several numerical studies to understand the effect of data quantization on the recovery error under various noise levels and different structures of measurement matrix.

- To illustrate the benefit for sparse signal recovery problems, we apply our quantization framework to radio astronomy and magnetic resonance imaging. Towards that, we model the interferometric radio imaging as a compressive sensing problem. For both applications, we show that lowering the precision of the data can significantly accelerate image recovery.

- We implement our approach on both Central Processing Units (CPU) and Field Programmable Gate Arrays (FPGA) platforms, demonstrating speed-ups of up to 7× and 9× for full recovery, respectively, on instances with a quantized dense model matrix.

In Part ii, we switch our focus to label efficiency for model selection. We show that active model selection can tackle with the distribution drift within the production data in a label-efficient manner:

- We introduce active model selection of pretrained classifiers, the first framework that aims to perform efficient data-labeling to rank pretrained classifiers.
- We adapt existing active learning strategies for model selection problem.
- We develop a novel, principled and efficient active model selection approach Model Picker for a setting where the production data examples arrive in a stream.
- We introduce a fair evaluation framework and compare Model Picker to the adapted active learning strategies.
- We furthermore conduct extensive experiments on well-studied ML benchmarks. To reach the same accuracy, competing methods can often require up to 2.5× more labels. Apart from the relative performance, on the ImageNet dataset, Model Picker requires a mere 13% labeled instances to select the best among 102 pre-trained models with 90% confidence, while having up to 1.3× lower regret. These results establish Model Picker as the state-of-the-art for this problem.

here.

Inspired by the idea of label-efficient model selection, in Part ii, we also identify the role of graph decomposition in leading GNN to maintain a better predictive performance although being exposed to the same training data:

- We characterize GCN from the information-theoretic perspective and show that under certain conditions, the mutual information between the output after l layers and the input of GCN converges to 0 exponentially with respect to l. On the other hand, we show that graph decomposition can potentially weaken the condition of such convergence rate, alleviating the information loss when GCN becomes deeper.

- We demonstrate that our theoretical analysis can enable further understanding on the goodness of graph decomposition and facilitates novel and effective graph decomposition techniques.

In Part iii of this thesis, we take a *first* step towards integrating *domain knowledge* to ML systems to improve its robustness against *diverse* attacks. We make contributions on both theoretical and empirical fronts:

- We propose the Knowledge Enhanced Machine Learning Pipeline (KEMLP), which integrates a main task ML model with a set of weak auxiliary task models, together with different knowledge rules connecting them.
- Theoretically, we provide the robustness guarantees for KEMLP and prove that under mild conditions, the prediction of KEMLP is more robust than that of a single main task model.
- Empirically, we develop KEMLP based on different main task models and evaluate them against a diverse set of attacks, including physical attacks, $\mathcal{L}_p$ bounded attacks, unforeseen attacks, and common corruptions. We show that the robustness of KEMLP outperforms all baselines by a wide margin, with comparable and often higher clean accuracy.

Part I

HARDWARE EFFICIENCY VIA DATA QUANTIZATION

2

LOW PRECISION COMPRESSIVE SENSING

It can scarcely be denied that the supreme goal of all theory is to make the irreducible basic elements as simple and as few as possible without having to surrender the adequate representation of a single datum of experience.

— Albert Einstein (1933)

2.1 OVERVIEW

Hardware efficiency via low precision training is an emerging research area with applications in ML. Several previous studies illustrated its benefit in accelerating computation and improving on the memory usage due to compression of bit-widths of the data and the model such as (Alistarh et al., 2017; De Sa et al., 2015; Zhang et al., 2017; Zhao et al., 2021) and the line of work on partial or end-to-end low-precision training of deep networks (Gupta et al., 2013; Seide et al., 2014), to name a few. In these works, data and model quantization is often studied in the context of Stochastic Gradient Descent (SGD) training. Our goal in this part is to extend the usability of low precision training to sparse signal recovery methods, to which the existing results on SGD do not directly apply. An interesting property of the compressive sensing problem and of many compressive sensing solvers is their tolerance to noise introduced by data quantization. As mentioned earlier, this has been utilized by many earlier frameworks (Ai et al., 2014; Boufounos and Baraniuk, 2008; Gopi et al., 2013; Gupta et al., 2013; Jacques et al., 2013; Laska et al., 2011; Plan and Vershynin, 2013a,b). Most of these previous work focused on the case where quantization is carried out only on the observation vector Table 2.1. In only one single previous study, both the observation vector and the measurement matrix were quantized by imposing additional assumptions on the measurement matrix (sub-Gaussian or binary) (Gopi et al., 2013). We take this direction further and investigate the design of a compressive sensing solver which quantizes both the measurement matrix *and* the observation vector, while imposing a more general set of assumptions on the measurement matrix than that of most existing work. These references also present a comprehensive analysis of the provable performance guarantees for such sparsity-constrained minimization methods, in terms of convergence to fixed point of ℓ_0-regularized cost functions and the optimality of such approximations. However, when applied to real-life problems, prior work faces additional challenges. For provable guarantees, it is often required that (a) the measurement matrix $\mathbf{\Phi}$ satisfies the Restricted Isometry Property (RIP) (Candes, 2008; Chartrand and Staneva, 2008), and that (b) the sparsity level is chosen appropriately. Motivated by these, in this thesis, we focus on normalized Iterative Hard Thresholding (IHT) (Blumensath and Davies, 2010), a popular iterative thresholding algorithm for compressive sensing. The normalized IHT relaxes the RIP condition by introducing a step size parameter and enables rigorous guarantees for a broader class of practical problems. We show that the normalized IHT converges with guarantees on the recovery quality even when both the measurement matrix and the observation vector are stored in lower precision — provided that the measurement matrix satisfies non-symmetric RIP (Blumensath and Davies, 2010; Eldar and Kutyniok, 2012). We moreover apply our proposal in the context of two real-world applications radio astronomy and Magnetic Resonance Imaging (MRI), and demonstrate the benefit of low precision training in accelerated sparse signal recovery. Finally, we implement our approach on both Central Processing Units (CPU) and Field Programmable Gate Arrays (FPGA) platforms, and achieve significant speed-ups in sparse signal recovery.[1]

NOTATION In the rest of this thesis, scalars will be written in italics, vectors in bold lower-case and matrices in bold upper-case letters. Particularly for

[1] 1053-587X ©2020 IEEE. Personal use the material in this chapter is permitted. Permission from IEEE must be obtained for all other uses of the material in this chapter, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works. Digital Object Identifier: 10.1109/TSP.2020.3010355

this part, on the other hand, we define $\boldsymbol{x}$ as an N-dimensional real or complex sparse vector and y as an M-dimensional real or complex observation vector. For an $M \times N$ real or complex measurement matrix $\boldsymbol{\Phi}$, the matrix element in the mth row and nth column is denoted as $\boldsymbol{\Phi}_{m,n}$ and its Hermitian transpose as Φ^T. Also, $\boldsymbol{\phi}_n$ is the nth column of $\boldsymbol{\Phi}$ such that $\boldsymbol{\Phi} = [\boldsymbol{\phi}_n]_{n=\{1,2,\ldots,N\}}$. The submatrix of Φ obtained by selecting the columns with indices in Γ is written as $\Phi_\Gamma = [\boldsymbol{\phi}_n]_{n\in\Gamma}$, and the p-norm by $\|\cdot\|_p$. For the sake of simplicity, we drop p whenever $p = 2$. Finally, a 32-bit representation is used for the full precision scheme and $\mathrm{b}_\Phi/\mathrm{b}_y$ denotes the number of bits used to represent the elements of the measurement matrix $\boldsymbol{\Phi}$ and the observation vector $\boldsymbol{y}$, respectively.

2.2 RELATED WORK

Several studies have applied quantization in compressive sensing problems (Table 2.1). They explore binary measurements for sparse signal recovery under different assumptions on the measurement matrix. Sparse signal recovery with a scale factor when measurements preserve only sign information was demonstrated in (Boufounos and Baraniuk, 2008). Further, approximately sparse signals can be robustly recovered from single-bit measurements when sampled with a sub-Gaussian distribution (Ai et al., 2014; Davenport et al., 2012). A similar setting is studied in (Jacques et al., 2013; Laska et al., 2011) with a Gaussian measurement matrix (Binary IHT) (Plan and Vershynin, 2013a,b), which proposes a computationally tractable and optimal recovery of a 1-bit compressive sensing problem. The theoretical guarantees to recover the *support* of high-dimensional sparse signals from 1-bit measurements are provided by (Gopi et al., 2013; Gupta, Nowak, and Recht, 2010).

Our work differs from prior work in two main ways. First, our assumption that the measurement matrix is non-symmetric RIP is critical in real-life applications, and none of the assumptions made in prior work would fit this use case. Second, to the best of our knowledge, we are the only work besides (Gopi et al., 2013) that quantizes *both* the measurement matrix $\boldsymbol{\Phi}$ and the observation vector $\mathbf{y}$. The problem of building a binary measurement matrix that can provide good recovery guarantees is considered in (Gopi

Table 2.1: Comparison of low precision iterative hard thresholding with previous work. $Q(\mathbf{\Phi})$ and $Q(\mathbf{y})$ indicate whether quantization of the measurement matrix $\mathbf{\Phi}$ or quantization of the observation vector $\mathbf{y}$ are considered (✓: yes, × : no).

	Assumption on $\mathbf{\Phi}$	$Q(\mathbf{\Phi})$	$Q(\mathbf{y})$
Boufounos and Baraniuk, 2008	Gaussian	×	✓
Ai et al., 2014	unit variance	×	✓
Jacques et al., 2013	RIP	×	✓
Laska et al. (Laska et al., 2011)	Gaussian & RIP	×	✓
Plan and Vershynin, 2013a	Gaussian & RIP	×	✓
Plan and Vershynin, 2013b	Gaussian	×	✓
Gupta et al., 2013	Gaussian	×	✓
Gopi et al., 2013	sub-Gaussian/binary & RIP	✓	✓
This work	non-symmetric RIP	✓	✓

et al., 2013) given only one-bit measurements. By contrast, we consider a practical setting where we must quantize a given full-precision measurement matrix as well as possible, and thus can trade off higher precision for better recovery guarantees.

There has been significant research on designing efficient algorithms for sparse recovery (Blanchard, Tanner, and Wei, 2013; Blumensath, 2012; Cevher, 2011; Liu et al., 2017; Wei, 2015). We focus here on normalized IHT, and extension to other work can be a promising future direction. We further note the work on recovery using sparse binary matrices (see (Berinde and Indyk, 2010) for a survey). These matrix constructions could be applied in our scenario in some cases, as they are pre-quantized with similar guarantees. However, in certain applications such as the radio astronomy and magnetic resonance imaging considered here, the measurement matrix is fixed and highly dense.

2.3 PROBLEM DEFINITION

Compressive sensing (Candes, Romberg, and Tao, 2006a,b; Donoho, 2006) is a technique in sparse signal reconstruction that offers a range of efficient algorithms acquiring high dimensional signals from inaccurate and incomplete samples with an underlying sparse structure. Many real-world applications including medical imaging, interferometry, and genomic data analysis benefit from these techniques.

In mathematical terms, compressive sensing is formulated as follows: Let a sparse or approximately sparse signal $\mathbf{x}$ be sampled via a linear sampling operator $\mathbf{\Phi}$. This means that the observation vector $\mathbf{y}$ is

$$\mathbf{y} = \mathbf{\Phi}\mathbf{x} + \mathbf{e} \tag{2.1}$$

where $\mathbf{e}$ is M-dimensional observation noise. We illustrate this model in Figure 2.1.

Compressive sensing recovery algorithms iteratively compute a sparse estimate $\tilde{\mathbf{x}}$ with $N \gg M$ such that $\mathbf{\Phi}\tilde{\mathbf{x}}$ approximates $\mathbf{y}$ well, that is, $\|\mathbf{y} - \mathbf{\Phi}\tilde{\mathbf{x}}\|$ is small. This problem is NP-hard due to its combinatorial nature. Thus, most compressive sensing algorithms resort to a convex relaxation of the underlying sparse optimization problem. A collection of thresholding and greedy methods solving this problem have been proposed including IHT (Blumensath and Davies, 2008, 2009), Compressive Sampling Matching Pursuit (CoSaMP) (Needell and Tropp, 2008), as well as others (Blumensath, 2013; Liu et al., 2017; Yuan, Li, and Zhang, 2014, 2016)

These references also present a comprehensive analysis of the provable performance guarantees for such sparsity-constrained minimization methods, in terms of convergence to fixed point of ℓ_0-regularized cost functions and the optimality of such approximations. However, when applied to real-life problems this prior work faces additional challenges. For provable guarantees, it is often required that (a) the measurement matrix $\mathbf{\Phi}$ satisfies the Restricted Isometry Property (RIP) (Candes, 2008; Chartrand and Staneva, 2008), and that (b) the sparsity level is chosen appropriately. The Normalized IHT method (Blumensath and Davies, 2010), relaxes the RIP condition by

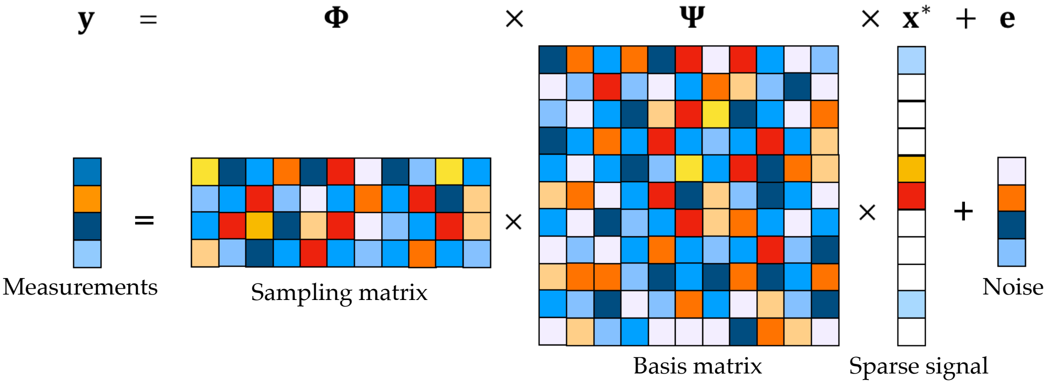

$\mathbf{y}$ = $\boldsymbol{\Phi}$ × $\boldsymbol{\Psi}$ × $\mathbf{x}^*$ + $\mathbf{e}$
=
×
×
+
Measurements
Sampling matrix
Basis matrix
Sparse signal
Noise

Figure 2.1: Compressive sensing model as a sparse expansion of measurements such that $\mathbf{y} = \boldsymbol{\Phi}\boldsymbol{\Psi}\mathbf{x}^* + \mathbf{e}$. The measurement matrix is given by the product of $\boldsymbol{\Phi}$ and $\boldsymbol{\Psi}$. In our analysis, we consider $\boldsymbol{\Psi}$ to be embedded in $\boldsymbol{\Phi}$ and denote the measurement matrix by $\boldsymbol{\Phi}$.

introducing a step size parameter, which enables rigorous guarantees for a broader class of practical problems. Our efforts in this part build upon this line of work.

We consider the sparse signal recovery problem in Equation 2.1 described as: given $\mathbf{y}$ and $\boldsymbol{\Phi}$, find $\mathbf{x}$ minimizing the cost function

$$\|\mathbf{y} - \boldsymbol{\Phi}\mathbf{x}\|^2 \text{ subject to } \|\mathbf{x}\|_0 \leq s, \tag{2.2}$$

where $\|\mathbf{x}\|_0 = |\text{supp}(\mathbf{x})| = |\{i : x_i \neq 0\}|$ and s is number of sparse coefficients we want to recover.

NORMALIZED IHT Normalized IHT (Blumensath and Davies, 2010) is an iterative solver of the optimization problem in Equation 2.2 that is shown to outperform other methods such as traditional IHT and CoSaMP when the non-symmetric RIP condition holds. It uses the following update rule:

$$\mathbf{x}^{[n+1]} = H_s(\mathbf{x}^{[n]} + \mu^{[n]}\boldsymbol{\Phi}^T(\mathbf{y} - \boldsymbol{\Phi}x^{[n]})), \tag{2.3}$$

where $\mathbf{x}^{[0]} = 0$ and $\mu^{[n]} > 0$ is the adaptive step size parameter, $H_s(\mathbf{x})$ is a nonlinear operator preserving only the largest s entries of $\mathbf{x}$ in magnitude, setting the other entries to zero.

If $\mathbf{x}$ has no more than s nonzero elements, the proposed update rule converges to a local minimum of the cost function $\|\mathbf{y} - \mathbf{\Phi}\mathbf{x}\|^2$. Furthermore, if the measurement matrix $\mathbf{\Phi}$ satisfies the non-symmetric RIP condition, normalized IHT is guarantees stability and performance,i.e., the result is near-optimal. The properties of normalized IHT are discussed in greater detail in 2.4.1.

OUR SETTING In this work, we consider the properties of the normalized IHT algorithm in a lossy compression setting, where both the data $\mathbf{y}$ and $\mathbf{\Phi}$ consist of floating-point values and undergo a stochastic quantization process to a small set of discrete levels, using a transformation operator. We denote the transformation operator by $Q(\cdot, b)$ where b is the bit precision used by the representation. The goal of applying $Q(\cdot, b)$ is to reduce the high cost of data transmission between the sensor or storage and the computational device (CPU, GPU and FPGA). We thus want to recover $\mathbf{x}$ using the modified normalized IHT update rule

$$\mathbf{x}^{[n+1]} = H_s(\mathbf{x}^{[n]} + \mu^{[n]} Q(\mathbf{\Phi}, b_\Phi)^T (Q(\mathbf{y}, b_y) - Q(\mathbf{\Phi}, b_\Phi)\mathbf{x}^{[n]})).$$

2.4 NORMALIZED ITERATIVE THRESHOLDING

In this section, we review existing results on the normalized IHT algorithm (Blumensath and Davies, 2010; Blumensath et al., 2012). These can be generalized to the traditional IHT if the measurement matrix satisfies $\|\mathbf{\Phi}\| < 1$ (Blumensath and Davies, 2008, 2009). The reader who is familiar with normalized IHT can skip this section.

2.4.1 *The Algorithm*

Let $\mathbf{x}^{[0]} = 0$. As introduced in Equation 2.3, normalized IHT has the following update rule:

$$\mathbf{x}^{[n+1]} = H_s(\mathbf{x}^{[n]} + \mu^{[n]} \mathbf{\Phi}^T (\mathbf{y} - \mathbf{\Phi} x^{[n]})),$$

where $H_s(\mathbf{x})$ is the thresholding operator that preserves the largest s entries (in magnitude), and $\mu^{[n]} > 0$ is an adaptive step size parameter. The recovery performance of normalized IHT depends conditionally on the step size parameter $\mu^{[n]}$, unlike the traditional IHT approach in which $\mu^{[n]} = 1$. While the traditional approach requires a rescaling of the measurement matrix such that $\|\mathbf{\Phi}\| < 1$ to ensure convergence, introducing a step size parameter that enables the arbitrary scaling of $\mathbf{\Phi}$, and hence relaxes the bounds on its norm. Specifically, the role of $\mu^{[n]}$ is to compensate for this rescaling by avoiding the undesirable amplification of noise, i.e., by keeping the ratio $\|\mathbf{\Phi x}\|/\|\mathbf{e}\|$ unchanged.

STEP SIZE DETERMINATION Normalized IHT adaptively sets the step size as follows: if the support of $\mathbf{x}^{[n]}$ is preserved between iterations, one can set the step size adaptively to

$$\mu^{[n]} = \frac{\left(\mathbf{g}^{[n]}_{\Gamma^{[n]}}\right)^T \mathbf{g}^{[n]}_{\Gamma^{[n]}}}{\left(\mathbf{g}^{[n]}_{\Gamma^{[n]}}\right)^T \mathbf{\Phi}^T_{\Gamma^{[n]}} \mathbf{\Phi}_{\Gamma^{[n]}} g^{[n]}_{\Gamma^{[n]}}}, \tag{2.4}$$

where $\mathbf{g}^{[n]} = \mathbf{\Phi}^T(y - \mathbf{\Phi x}^{[n]})$ and $\Gamma^{[n]} = \text{supp}(\mathbf{x}^{[n]})$. This is shown to result in the maximal reduction of the cost function. However, if the support of $\mathbf{x}^{[n+1]}$ differs from that of $\mathbf{x}^{[n]}$, a sufficient convergence condition is shown to be

$$\mu^{[n]} \leq (1-c)\frac{\|\mathbf{x}^{[n+1]} - \mathbf{x}^{[n]}\|^2}{\|\mathbf{\Phi}(\mathbf{x}^{[n+1]} - \mathbf{x}^{[n]})\|^2}$$

for any small constant c. If the above condition is not met, a new proposal for $\mathbf{x}^{[n+1]}$ can be calculated by setting $\mu^{[n]} \leftarrow \mu^{[n]}/(k(1-c))$, where k is a shrinkage parameter satisfying $k > 1/(1-c)$.

2.4.2 *Recovery Guarantees*

The analysis of hard thresholding algorithms relies on the scaling properties of $\mathbf{\Phi}$. Concretely, one often considers the non-symmetric RIP condition: a matrix $\mathbf{\Phi}$ satisfies the non-symmetric RIP if there are $0 < \alpha_s, \beta_s \in \mathbb{R}$ and $\alpha_s \leq \beta_s$ such that

$$\alpha_s \leq \frac{\|\mathbf{\Phi x}\|}{\|\mathbf{x}\|} \leq \beta_s \text{ for all } \mathbf{x} \text{ with } \|\mathbf{x}\|_0 \leq s. \tag{2.5}$$

α_s and β_s are the so-called Restricted Isometric Constants (RICs). Note that for any support set Γ such that $|\Gamma| \leq s$, α_s and β_s are lower and upper bounded by the smallest and largest singular values of $\mathbf{\Phi}_{|\Gamma|}$, respectively.

The main convergence result of normalized IHT can be stated as follows.

Theorem 1 (Blumensath et al., 2012)**.** *Let $\mathbf{\Phi}$ be full rank and $s \leq m$. If $\beta_{2s} \leq \mu^{-1}$, then normalized IHT converges to a local minimum of Equation 2.2.*

When setting the step size parameter, the condition $\beta_{2s} \leq \mu^{-1}$, which ensures convergence, poses a challenge. To date, there is no efficient strategy to determine the exact values of the RICs β_s and α_s for an arbitrary measurement matrix in a computationally efficient manner. However, these constants can be bounded efficiently, and it can be shown that randomly constructed measurement matrices can satisfy the RIP with high probability (Candes, 2008; Chartrand and Staneva, 2008).

The adaptive setting of the step size parameter is further shown to provide a non-symmetric RIP variant recovery result as follows.

Theorem 2 (Blumensath and Davies, 2010)**.** *Consider a noisy observation $\mathbf{y} = \mathbf{\Phi x} + \mathbf{e}$ with an arbitrary vector $\mathbf{x}$, and let $\mathbf{x}^s$ be the best s-term approximation of $\mathbf{x}$. If $\operatorname{rank}(\mathbf{\Phi}) = M$ and $\operatorname{rank}(\mathbf{\Phi}_\Gamma) = s$ for all Γ with $|\Gamma| = s$, then the normalized IHT algorithm converges to a local minimum of the cost function in Equation 2.2. Also, assume $\mathbf{\Phi}$ has the non-symmetric RIP when projected onto $2s$-sparse vectors, with RICs α_{2s} and β_{2s}.*

We further define $\gamma_{2s} = \beta_{2s}/\alpha_{2s} - 1$ if the normalized IHT algorithm uses the step size defined in Equation 2.4 at each iteration, and $\gamma_{2s} = \max(1 - \alpha_{2s}/k\beta_{2s}, \beta_{2s}/\alpha_{2s} - 1)$ otherwise, where $k > 1$ is a shrinkage parameter introduced earlier. If $\gamma_{2s} \leq 1/8$, then the recovery error after n iterations is bounded as

$$\|\mathbf{x} - \mathbf{x}^{[n]}\| \leq 2^{-n}\|\mathbf{x}^s\| + 8\varepsilon_s \tag{2.6}$$

where

$$\varepsilon_s = \|\mathbf{x} - \mathbf{x}^s\| + \frac{\|\mathbf{x} - \mathbf{x}^s\|_1}{\sqrt{s}} + \frac{1}{\beta_{2s}}\|\mathbf{e}\|. \tag{2.7}$$

Corollary 1. *After at most $n^* = \log_2(\|x^s\|/\varepsilon_s)$ iterations, the recovery error bound in (2.6) can be further simplified to*

$$\|\mathbf{x} - \mathbf{x}^{[n]}\| \leq 9\varepsilon_s.$$

The above result suggests that, after a sufficiently large number of iterations, the reconstruction error is induced only by the noise $\mathbf{e}$ and that $\mathbf{x}$ is not exactly s-sparse.

2.5 LOW PRECISION ITERATIVE THRESHOLDING

We now introduce the quantized version of normalized IHT, called Quantized Iterative Hard Thresholding (QIHT), and analyze it in terms of signal recovery performance. The key idea here is that, by reducing the bit widths of the data points in a structured manner, we can upper bound the recovery error and fine tune the bit precision to still guarantee provable recovery performance. In Section 2.6, we will show that the recovery error bound reflects the true scaling of parameters in the regime where the non-symmetric RIP holds, and that for specific applications, in particular radio astronomy and magnetic resonance imaging, we expect the recovery error to be small, thanks to the structure of the measurement matrix.

2.5.1 *The Algorithm*

The QIHT algorithm follows the modified update rule of, assuming $\mathbf{x}^{[0]} = 0$:

$$\mathbf{x}^{[n+1]} = H_s\big(\mathbf{x}^{[n]} + \hat{\mu}^{[n]} Q(\mathbf{\Phi}^T, b_\Phi)(Q(\mathbf{y}, b_y) - Q(\mathbf{\Phi}, b_\Phi)\mathbf{x}^{[n]})\big) \tag{2.8}$$

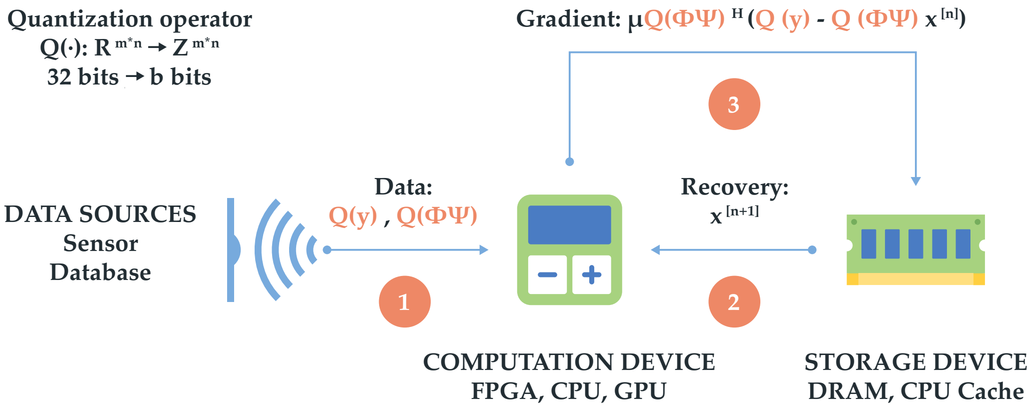


Figure 2.2: Low Precision IHT (QIHT) data flow. (1) Quantized data, namely the observation vector and measurement matrix is sent to the computation device, (2) which takes the previous update from the storage and (3) calculates the gradient and send it to the storage device, where $\mathbf{x}^{[n]}$ will be updated to $\mathbf{x}^{[n+1]}$. This process continues until the convergence.

where the step size $\hat{\mu}^{[n]}$ is determined based on Equation 2.4, and $Q(\cdot, b)$ is an element-wise quantization operator that maps single-precision floating-point values to b-bit precision.

In the following, we will use the stochastic quantization operator $Q(v, b)$, which quantizes v to b-bit precision as follows. Let $\ell = 2^b$ and $q_1, \dots, q_\ell$ denote ℓ equally spaced points in $[-1, 1]$ such that $q_1 = -1 \leq q_2 \leq \cdots \leq q_\ell = 1$. Assume that $v \in [q_i, q_{i+1}]$ for some i. Stochastic quantization maps v to one of the two nearest points as follows:

$$Q(v, b) = \begin{cases} q_i, & \text{with probability } \frac{q_{i+1}-v}{q_{i+1}-q_i}, \\ q_{i+1}, & \text{otherwise.} \end{cases}$$

Note that the quantization $Q(\cdot, b)$ is unbiased, i.e., $\mathbb{E}[Q(v, b)] = v$, and matrices and vectors are quantized element-wise.

Note that in Equation 2.8, we use two independent stochastic quantizations for $\mathbf{\Phi}^T$ and $\mathbf{\Phi}$, the so-called double sampling (Zhang et al., 2017). This leads to an unbiased gradient estimator, that is,

$$\mathbb{E}\left[Q(\mathbf{\Phi}^T, b_{\mathbf{\Phi}})(Q(\mathbf{y}, b_y) - Q(\mathbf{\Phi}, b_\Phi)\mathbf{x}^{[n]})\right] = \mathbf{\Phi}^T(y - \mathbf{\Phi}\mathbf{x}^{[n]}),$$

Algorithm 1 QIHT: Low Precision IHT

```
Input: number of iterations $n^*$, $2n^*$ realizations of the low precision measurement matrix $Q(\boldsymbol{\Phi})$: $\hat{\Phi}_1, \hat{\Phi}_2, \dots, \hat{\Phi}_{2n^*}$, $n^*$ realizations of the low precision observation vector $Q(\mathbf{y})$: $\hat{y}_1, \hat{y}_2, \dots, \hat{y}_{n^*}$, sparsity parameter $s$, step size tuning parameters $k$, $c$
Output: The recovery vector $\mathbf{x}^{[n^*]}$
Initialize $\mathbf{x}^{[0]} = 0$, $\Gamma^{[0]} = \text{supp}\left(H_s(\hat{\boldsymbol{\Phi}}_1^T \hat{\mathbf{y}})\right)$.
for $n = 1$ to $n^*$ do
    $\mathbf{g}^{[n-1]} = \hat{\boldsymbol{\Phi}}_{2n-1}^T (\hat{\mathbf{y}} - \hat{\boldsymbol{\Phi}}_{2n} \mathbf{x}^{[n]})$
    $\hat{\mu}^{[n-1]} = \left(\mathbf{g}^{[n-1]}_{\Gamma^{[n-1]}}\right)^T \mathbf{g}^{[n-1]}_{\Gamma^{[n-1]}} / \left((\boldsymbol{\Phi}_{2n-1})_{\Gamma^{[n-1]}} \mathbf{g}^{[n-1]}_{\Gamma^{[n-1]}}\right)^T (\Phi_{2n})_{\Gamma^{[n-1]}} \mathbf{g}^{[n-1]}_{\Gamma^{[n-1]}}$
    $\mathbf{x}^{[n]} = H_s(\mathbf{x}^{[n-1]} + \hat{\mu}^{[n-1]} \mathbf{g}^{[n-1]})$
    $\Gamma^{[n]} = \text{supp}(\mathbf{x}^{[n]})$
    if $\Gamma^{[n]} = \Gamma^{[n-1]}$ then
        $\mathbf{x}^{[n]} = \mathbf{x}^{[n-1]}$
    else
        $b^{[n]} = (\|\mathbf{x}^{[n]} - \mathbf{x}^{[n-1]}\|_2^2) / (\|\hat{\Phi}_{2n-1}(\mathbf{x}^{[n]} - \mathbf{x}^{[n-1]})\|_2^2)$
        if $\hat{\mu}^{[n]} \leq (1-c) b^{[n]}$ then
            $\mathbf{x}^{[n]} = \mathbf{x}^{[n-1]}$
        else
            while $\hat{\mu}^{[n]} > (1-c) b^{[n]}$ do
                $\hat{\mu}^{[n]} = \hat{\mu}^{[n]} / (k(1-c))$
                $\mathbf{x}^{[n]} = H_s(\mathbf{x}^{[n-1]} + \hat{\mu}^{[n-1]} \mathbf{g}^{[n-1]})$
            end
        end
    end
    $\Gamma^{[n]} = \text{supp}(\mathbf{x}^{[n]})$
end
```

which provides better convergence results.

A detailed description of QIHT is given in Algorithm 1.

2.5.2 *Performance Guarantees*

We study the theoretical guarantees of QIHT by analyzing two properties, namely convergence and recovery error.

2.5.2.1 *Convergence*

We start with stating the convergence result. In the following, we set $\hat{\mathbf{\Phi}} = Q(\mathbf{\Phi}, b_{\Phi})$ and $\hat{\mathbf{y}} = Q(\mathbf{y}, b_y)$. We also use $\hat{\mathbf{\Phi}}_j$ to denote the j^{th} quantization $\hat{\mathbf{\Phi}}$.

Theorem 3 (Convergence guarantee of QIHT)**.** *The QIHT algorithm attains a local minimum of the cost function $\mathbb{E}[\|\hat{\mathbf{y}} - \hat{\mathbf{\Phi}}\mathbf{x}\|^2]$ such that $\|\mathbf{x}\|_0 \leq s$.*

Proof of Theorem 3. $\mathbb{E}[\|\hat{\mathbf{y}} - \hat{\mathbf{\Phi}}\mathbf{x}\|^2]$ can be majorized by the following surrogate objective function

$$\mathbb{E}[\|\mu^{0.5}\hat{\mathbf{y}} - \hat{\mathbf{\Phi}}\mathbf{x}\|^2 + \|\mathbf{x} - \mathbf{x}^{[n]}\|^2 - \|\mu^{0.5}\hat{\mathbf{\Phi}}(\mathbf{x} - \mathbf{x}^{[n]})\|^2],$$

whenever $\|\mu^{0.5}\hat{\mathbf{\Phi}}\|^2 < 1$. This condition is met due to the step size determination introduced in Equation 2.4. The minimizer of the above surrogate objective $\mathbf{x}^{[n+1]}$, therefore, ensures that $\mathbb{E}[\|\hat{\mathbf{y}} - \hat{\mathbf{\Phi}}\mathbf{x}^{[n+1]}\|^2] \leq \mathbb{E}[\|\hat{\mathbf{y}} - \hat{\mathbf{\Phi}}\mathbf{x}^{[n]}\|^2]$. Using the arguments of (Blumensath and Davies, 2008), Equation 2.8 can be shown to minimize the expected cost $\mathbb{E}[\|\hat{\mathbf{y}} - \hat{\mathbf{\Phi}}\mathbf{x}\|^2]$. □

2.5.2.2 *Recovery Error*

The following theorem states our main analytic result, which characterizes the recovery error of QIHT, specifically focusing on the additional error introduced by the quantization procedure.

Theorem 4 (Recovery Error of QIHT)**.** *Consider an M-dimensional noisy observation vector $\mathbf{y} = \mathbf{\Phi}\mathbf{x} + \mathbf{e}$ where $\mathbf{\Phi}$ is an $M \times N$-dimensional real or complex matrix, and $\mathbf{x}$ is an N-dimensional arbitrary vector. Let $H_s(\mathbf{x}) = \mathbf{x}^s$ with $s \leq M$ and assume the full precision measurement matrix $\mathbf{\Phi}$ and the quantized measurement matrix $\hat{\mathbf{\Phi}}$ satisfy the non-symmetric RIP in Equation 2.5 and Equation 2.11, with RICs α_s, β_s and $\hat{\alpha}_s, \hat{\beta}_s$, respectively. We also define $\gamma_{2s} = \beta_{2s}/\alpha_{2s} - 1$ if the normalized IHT algorithm uses the step size defined in Equation 2.4 at each* *iteration and* $\gamma_{2s} = \max(1 - \alpha_{2s}/k\beta_{2s},\ \beta_{2s}/\alpha_{2s} - 1)$ *otherwise. Similarly, let* $\hat{\gamma}_{2s} = \hat{\beta}_{2s}/\hat{\alpha}_{2s} - 1$ *if the QIHT algorithm uses the step size defined in Equation 2.4 and* $\hat{\gamma}_{2s} = \max(1 - \hat{\alpha}_{2s}/k\hat{\beta}_{2s},\ \hat{\beta}_{2s}/\hat{\alpha}_{2s} - 1)$ *otherwise. If* γ_{2s} *and* $\hat{\gamma}_{2s}$ *satisfy* $\gamma_{2s} \leq 1/24$ *and* $\hat{\gamma}_{2s} \leq 1/24$*, then at each iteration n, the QIHT algorithm outputs an approximation of* $\mathbf{x}^s$, $\mathbf{x}^{[n]}$ *such that*

$$\mathbb{E}[\|\hat{\mathbf{x}}^{[n]} - \mathbf{x}^s\|] \leq 2^{-n}\|\mathbf{x}^s\| + 9\varepsilon_s + 4.5\varepsilon_q, \tag{2.9}$$

where ε_s *is given by*

$$\varepsilon_s = \|\mathbf{x} - \mathbf{x}^s\| + \frac{\|\mathbf{x} - \mathbf{x}^s\|_1}{\sqrt{s}} + \frac{1}{\min(\beta_{2s}, \hat{\beta}_{2s})}\|\mathbf{e}\|$$

and

$$\varepsilon_q = \frac{\sqrt{M}}{\hat{\beta}_{2s}}\Big(\frac{\|\mathbf{x}^s\|}{2^{b_\Phi - 1}} + \frac{1}{2^{b_y - 1}}\Big).$$

Here, b_Φ *and* b_y *are the number of bits used to represent* $\mathbf{\Phi}$ *and* $\mathbf{y}$*, respectively.*

Corollary 2. *A natural stopping criterion is* $n^* = \lceil \log_2(\|\mathbf{x}^s\|/\varepsilon_s) \rceil$*, which means the algorithm computes successive approximations of* $\mathbf{x}^s$ *with accuracy* $\mathbb{E}[\|\mathbf{x}^{[n^*]} - \mathbf{x}^s\|] \leq 10\varepsilon_s + 4.5\varepsilon_q$.

DETERMINING THE BIT PRECISION b One constraint in the above theorem is that both $\hat{\mathbf{\Phi}}$ and $\mathbf{\Phi}$ satisfy the non-symmetric RIP with $\gamma_{2s}, \hat{\gamma}_{2s} \leq 1/24$. The following lemma describes the relationship between the non-symmetric RIP properties of $\mathbf{\Phi}$ and $\hat{\mathbf{\Phi}}$. The result suggests that one can ensure that the non-symmetric RIP holds for $\hat{\mathbf{\Phi}}$ using sufficient bit precision.

Lemma 1. *Let* $\varepsilon > 0$ *and let* $\mathbf{\Phi}_\Gamma$ *satisfy the non-symmetric RIP with* $\gamma_{|\Gamma|} \leq 1/24 - \varepsilon$ *for any support set* Γ*. If* $b_\Phi \geq \log\left(\frac{2\sqrt{|\Gamma|}}{\varepsilon\alpha_{|\Gamma|}}\right)$*, then* $\hat{\mathbf{\Phi}}_\Gamma$ *is guaranteed to satisfy* $\hat{\gamma}_{|\Gamma|} \leq 1/24$.

2.5.3 *Proofs*

In this section, we will present the proofs for Theorem 4, Corollary 2 and Lemma 1. Before presenting them, we introduce our notation and state the auxiliary results that will be used throughout their proof.

PRELIMINARIES We begin by introducing our notation.

$\mathbf{x}^s$	such that $\mathbf{y} = \mathbf{\Phi x} + \mathbf{e} = \mathbf{\Phi x}^s + \mathbf{\Phi}(\mathbf{x} - \mathbf{x}^s) + \mathbf{e}$
ε	$\mathbf{\Phi}(\mathbf{x} - \mathbf{x}^s) + \mathbf{e}$, hence $\mathbf{y} = \mathbf{\Phi x}^s + \varepsilon$
$\varepsilon_{\mathbf{y}}$	$Q(\mathbf{y}, b) - \mathbf{y}$
$\Gamma^{[n]}$	$\text{supp}\{\mathbf{x}^{[n]}\}$
$\hat{\Gamma}^{[n]}$	$\text{supp}\{\hat{\mathbf{x}}^{[n]}\}$
Γ^s	$\text{supp}\{x^s\}$
$\hat{B}^{[n]}$	$\hat{\Gamma}^{[n]} \cup \Gamma^s$
$\mathbf{a}^{[n+1]}$	$\hat{\mathbf{x}}^{[n]} + \mu^{[n]}\Phi^{\dagger}(\mathbf{y} - \Phi\hat{\mathbf{x}}^{[n]})$
$\hat{\mathbf{a}}^{[n+1]}$	$\hat{\mathbf{x}}^{[n]} + \hat{\mu}^{[n]}Q_1(\mathbf{\Phi})^{\dagger}(\mathbf{y} - Q_2(\mathbf{\Phi})\hat{\mathbf{x}}^{[n]})$
$\mathbf{x}^{[n+1]}$	$H_s(\mathbf{a}^{[n+1]})$
$\hat{\mathbf{x}}^{[n+1]}$	$H_s(\hat{\mathbf{a}}^{[n+1]})$
$\mathbf{r}^{[n]}$	$\hat{\mathbf{x}}^{[n]} - \mathbf{x}^s$

Next, assume $\mathbf{\Phi}$ satisfies the non-symmetric RIP

$$\alpha_s \leq \frac{\|\mathbf{\Phi x}\|_2}{\|\mathbf{x}\|_2} \leq \beta_s \tag{2.10}$$

for all $\mathbf{x} : \|\mathbf{x}\|_0 \leq s$, where $\alpha_s \in \mathbb{R}$ and $\beta_s \in \mathbb{R}$ are the lowest and largest singular value of $\mathbf{\Phi}$ such that $0 < \alpha_s \leq \beta_s$, the RICs. Inherent from its definition, the RIP for the quantized measurement matrix denoted by $Q(\mathbf{\Phi}, b_m)$ refers to that $\forall Q(\mathbf{\Phi}, b_m)$, we have

$$\hat{\alpha}_s \leq \frac{\|Q(\mathbf{\Phi}, b_m)\mathbf{x}\|_2}{\|\mathbf{x}\|_2} \leq \hat{\beta}_s \tag{2.11}$$

where $\hat{\alpha}_s$ and $\hat{\beta}_s$ are the associated RICs. For simplicity, we drop b_m, and use $Q(\mathbf{\Phi})$ instead.

The adaptive setting of step size parameter $\mu^{[n]}$ in normalized IHT is shown to satisfy $1/\beta_{2s}^2 \leq \mu^{[n]} \leq 1/\alpha_{2s}^2$ if $\mu^{[n]}$ is set to $\mathbf{g}_{\Gamma^{[n]}}^{\dagger}\mathbf{g}_{\Gamma^{[n]}}/\mathbf{g}_{\Gamma^{[n]}}^{\dagger}\Phi_{\Gamma^{[n]}}^{\dagger}\Phi_{\Gamma^{[n]}}\mathbf{g}_{\Gamma^{[n]}}$ at each iteration and $1/k\beta_{2s}^2 \leq \mu^{[n]} \leq 1/\alpha_{2s}^2$ otherwise (Blumensath and Davies, 2010). Similar inequality also holds in the quantized setting such that $1/\hat{\beta}_{2s}^2 \leq \hat{\mu}^{[n]} \leq 1/\hat{\alpha}_{2s}^2$ if $\hat{\mu}^{[n]}$ is set to $\hat{\mathbf{g}}_{\hat{\Gamma}^{[n]}}^{\dagger}\hat{\mathbf{g}}_{\hat{\Gamma}^{[n]}}/\hat{\mathbf{g}}_{\hat{\Gamma}^{[n]}}^{\dagger}\hat{\Phi}_{\hat{\Gamma}^{[n]}}^{\dagger}\hat{\mathbf{\Phi}}_{\hat{\Gamma}^{[n]}}\hat{\mathbf{g}}_{\hat{\Gamma}^{[n]}}$ at each iteration, and $1/k\hat{\beta}_{2s}^2 \leq \hat{\mu}^{[n]} \leq 1/\hat{\alpha}_{2s}^2$ otherwise.

Recall from Theorem 4 that depending on the step size, γ_s is defined as either $\beta_s/\alpha_s - 1$ or $\max\{1 - \alpha_s/k\beta_s,\ \beta_s/\alpha_s - 1\}$. That also holds for $\hat{\gamma}_s$ by replacing the RICs with that of quantized measurement matrix $Q(\mathbf{\Phi})$; $\hat{\alpha}_s$ and $\hat{\beta}_s$.

Remark 1. *Using the definitions of γ_s and $\hat{\gamma}_s$ as well as bounds on $\mu^{[n]}$ and $\hat{\mu}^{[n]}$, we further have*

$$(1 - \gamma_{2s})/\alpha_{2s}^2 \leq \mu^{[n]} \leq (1 + \gamma_{2s})/\beta_{2s}^2,$$

$$(1 - \hat{\gamma}_{2s})/\hat{\alpha}_{2s}^2 \leq \hat{\mu}^{[n]} \leq (1 + \hat{\gamma}_{2s})/\hat{\beta}_{2s}^2.$$

Based on the properties above, the RIP and the adaptive step size, which we will require repeatedly throughout the proof of Theorem 4, has several other consequences, summarized as follows.

Lemma 2. *Suppose $\mathbf{\Phi}$ and $Q(\mathbf{\Phi})$ satisfy RIP in Equation 2.5 and Equation 2.11, respectively. Let moreover Γ, Υ and Λ has cardinality at most* $\min\big(\operatorname{rank}(\mathbf{\Phi}), \operatorname{rank}(Q(\mathbf{\Phi}))\big)$ *and Υ and Γ are disjoint, $\Upsilon \cap \Lambda = \emptyset$. Then*

$$\|\big(\mu^{[n]}\Phi_{\Gamma}^T - \hat{\mu}^{[n]}Q(\Phi)_{\Gamma}^T\big)\mathbf{x}_{\Gamma}\|_2 \overset{(1)}{\leq} \max\big((1 + \gamma_{|\Gamma|})/\beta_{|\Gamma|}, 1 + \hat{\gamma}_{|\Gamma|})/\hat{\beta}_{|\Gamma|}\big)\|\mathbf{x}_{\Gamma}\|_2,$$

$$\|\big(\mu^{[n]}\Phi_{\Gamma}^T\mathbf{\Phi}_{\Gamma} - \mu^{[n]}Q_1(\mathbf{\Phi})_{\Gamma}^T Q_2(\mathbf{\Phi})_{\Gamma}\big)\mathbf{x}_{\Gamma}\|_2 \overset{(2)}{\leq} (\gamma_{|\Gamma|} + \hat{\gamma}_{|\Gamma|})\|\mathbf{x}_{\Gamma}\|_2,,$$

$$\|\big(\mu^{[n]}\mathbf{\Phi}_{\Upsilon}^T\mathbf{\Phi}_{\Lambda} - \mu^{[n]}Q_1(\mathbf{\Phi})_{\Upsilon}^T Q_2(\mathbf{\Phi})_{\Lambda}\big)\mathbf{x}_{\Lambda}\|_2 \overset{(3)}{\leq} (\gamma_{|\Upsilon\cup\Lambda|} + \hat{\gamma}_{|\Upsilon\cup\Lambda|})\|\mathbf{x}_{\Lambda}\|_2.$$

Proof of Lemma 2. As a simple consequence of RIP, the singular values of $\mathbf{\Phi}_\Gamma$ lie between $\alpha_{|\Gamma|}$ and $\beta_{|\Gamma|}$. Remark 1 further implies that the singular values of $\mu^{[n]}\Phi_\Gamma$ are in $[(1-\gamma_{|\Gamma|})/\alpha_{|\Gamma|}, (1+\gamma_{|\Gamma|})/\beta_{|\Gamma|}]$. Using the similar bound for $Q(\Phi)_\Gamma$, maximum singular value of $(\mu^{[n]}\Phi_\Gamma^T - \hat{\mu}^{[n]}Q(\Phi)_\Gamma^T)$, i.e., its operator norm, is given by $(1+\gamma_{|\Gamma|})/\beta_{|\Gamma|} - (1-\hat{\gamma}_{|\Gamma|})/\alpha_{|\Gamma|}$. In the first inequality of Lemma 2, we use a looser bound $(1+\gamma_{|\Gamma|})/\beta_{|\Gamma|}$ for simplicity.

Similar argument holds for the second inequality, that is, the singular values of $\mu^{[n]}\Phi_\Gamma^T\mathbf{\Phi}_\Gamma$ and $\mu^{[n]}Q_1(\mathbf{\Phi})_\Gamma^T Q_2(\mathbf{\Phi})_\Gamma$ fall into $[1-\gamma_{|\Gamma|}, 1+\gamma_{|\Gamma|}]$ and $[1-\hat{\gamma}_{|\Gamma|}, 1+\hat{\gamma}_{|\Gamma|}]$, respectively. Then $\|\mu^{[n]}\mathbf{\Phi}_\Gamma^T\mathbf{\Phi}_\Gamma - \mu^{[n]}Q_1(\mathbf{\Phi})_\Gamma^T Q_2(\mathbf{\Phi})_\Gamma\|_2$ is upper bounded by $\gamma_{|\Gamma|} + \hat{\gamma}_{|\Gamma|}$, which proves the second inequality.

The third inequality is a consequence of the fact that $-\mu^{[n]}\mathbf{\Phi}_\Upsilon^T\mathbf{\Phi}_\Lambda$ is a submatrix of $I - \mu^{[n]}\mathbf{\Phi}_{\Upsilon\cup\Lambda}^T\mathbf{\Phi}_{\Upsilon\cup\Lambda}$ As previously shown, eigenvalues of $\mu^{[n]}\mathbf{\Phi}_{\Upsilon\cup\Lambda}^T\mathbf{\Phi}_{\Upsilon\cup\Lambda}$ lie in $[1-\gamma_{|\Upsilon\cup\Lambda|},\ 1+\gamma_{|\Upsilon\cup\Lambda|}]$. Hence, eigenvalues of $\mu^{[n]}\mathbf{\Phi}_\Upsilon^T\mathbf{\Phi}_\Lambda$ are in $[-\gamma_{|\Upsilon\cup\Lambda|}, \gamma_{|\Upsilon\cup\Lambda|}]$. The maximum eigenvalue of $(\mu^{[n]}\mathbf{\Phi}_\Upsilon^T\mathbf{\Phi}_\Lambda - \mu^{[n]}Q_1(\mathbf{\Phi})_\Upsilon^T Q_2(\mathbf{\Phi})_\Lambda)$, hence its operator norm, can then be upper bounded by $\gamma_{|\Upsilon\cup\Lambda|} + \ \hat{\gamma}_{|\Upsilon\cup\Lambda|}$. □

Lemma 3. *(Blumensath and Davies, 2010) For any* $\mathbf{x}$, *let* $\mathbf{x}^s$ *be the best s-term approximation to x and* Υ *be a set with at most s elements. Then*

$$\|\mu^{[n]}\mathbf{\Phi}_\Upsilon^T\mathbf{\Phi}(\mathbf{x}-\mathbf{x}^s)\|_2 \le (1+\gamma_{2s})\left[\|x-x^s\|_2] + \frac{\|\mathbf{x}-\mathbf{x}^s\|_1}{\sqrt{s}}\right]. \tag{2.12}$$

Lemma 4. *Let* $Q(\cdot,b) : \mathbb{R}^d \times \mathbb{Z}^+ \to \mathbb{R}^d$ *denote quantization operator. For any* $\mathbf{v} \in \mathbb{R}^d$, *the norm of quantization error can be bounded by*

$$\mathbb{E}[\|Q(\mathbf{v},b) - \mathbf{v}\|_2] \le \frac{c_v\sqrt{M}}{2^{b-1}} \tag{2.13}$$

where c_v *is the maximum value of the components of v in magnitude.*

Remark 2. *For efficient fixed-point computation on FPGA, we need an odd number of quantization levels, and therefore total number of levels for b bit quantization is* $2^{b-1}+1$. *That is, the interval between two consecutive levels is* $1/2^{b-2}$ *provided the values are confined in the interval* $[-1,1]$ *a priori.*

Proof of Lemma 4. Let $\tilde{v} = v/c_v$. Using Jensen's inequality we can easily show that

$$\begin{aligned}\mathbb{E}[\|Q(\tilde{\mathbf{v}},b)-\tilde{\mathbf{v}}\|_2] &\le \sqrt{\mathbb{E}[\|Q(\tilde{\mathbf{v}},b)-\tilde{\mathbf{v}}\|_2^2]} = \sqrt{\sum_{i=1}^{M}\mathbb{E}[\big(Q(\tilde{\mathbf{v}},b)_i-\tilde{v}_i\big)^2]} \\ &\le \sqrt{\sum_{i=1}^{M}\mathbb{P}(Q(\tilde{\mathbf{v}},b)_i=\ell_j)(\tilde{\mathbf{v}}_i-\ell_j)^2+\mathbb{P}(Q(\tilde{\mathbf{v}},b)_i=\ell_{j+1})(\ell_{j+1}-\tilde{v}_i)^2}.\end{aligned}$$

Our quantization scheme uses a stochastic approach such that $\mathbb{P}(Q(\hat{\mathbf{v}},b)_i = \ell_j) = \frac{\ell_{j+1}-\tilde{v}_i}{\ell_{j+1}-\ell_j}$, and hence $\mathbb{P}(Q(\tilde{\mathbf{v}},b)_i = \ell_{j+1}) = 1 - \frac{\ell_{j+1}-\tilde{v}_i}{\ell_{j+1}-\ell_j}$. Substituting these into the above inequality we have

$$\mathbb{E}[\|Q(\tilde{\mathbf{v}},b)-\tilde{\mathbf{v}}\|_2] \le \sqrt{\sum_{i=1}^{n}(l_{j+1}-Q(\tilde{\mathbf{v}},b)_i)(Q(\tilde{\mathbf{v}},b)_i-\ell_j)}. \tag{2.14}$$

It can easily be seen that $(l_{j+1} - Q(\hat{\mathbf{v}},b)_i)(Q(\hat{v},b)_i - \ell_j)$ is maximized when $Q(\hat{\mathbf{v}},b)_i) = \frac{\ell_{j+1}-\ell_j}{2}$, moreover the quantization function implies that $\ell_{j+1} - \ell_j = \frac{1-(-1)}{l} = \frac{1}{2^{b-2}}$

$$\mathbb{E}[\|Q(\tilde{\mathbf{v}},b)-\tilde{\mathbf{v}}\|_2] \le \sqrt{\sum_{i=1}^{M}\frac{(\ell_{j+1}-\ell_j)^2}{4}} \le \frac{\sqrt{M}(\ell_{j+1}-\ell_j)}{2} \le \frac{\sqrt{M}}{2^{b-1}}. \tag{2.15}$$

□

We move to the proof of Theorem 4.

Proof of Theorem 4. The recovery error can be split into two parts by using triangle inequality

$$\begin{aligned}&\mathbb{E}[\|\hat{\mathbf{x}}^{[n+1]}-\mathbf{x}^s\|_2|\hat{\mathbf{x}}^{[n]}] = \mathbb{E}[\|\hat{\mathbf{x}}^{[n+1]}_{\hat{B}^{[n+1]}}-\mathbf{x}^s_{\hat{B}^{[n+1]}}\|_2|\hat{\mathbf{x}}^{[n]}] \\ &\le \mathbb{E}[\|\hat{\mathbf{x}}^{[n+1]}_{\hat{B}^{[n+1]}}-\hat{\mathbf{a}}^{[n+1]}_{\hat{B}^{[n+1]}}\|_2|\hat{\mathbf{x}}^{[n]}] + \mathbb{E}[\|\hat{\mathbf{a}}^{[n+1]}_{\hat{B}^{[n+1]}}-\mathbf{x}^s_{\hat{B}^{[n+1]}}\|_2|\hat{\mathbf{x}}^{[n]}].\end{aligned} \tag{2.16}$$

where the equality follows from that $\hat{\mathbf{x}}^{[n+1]} - \mathbf{x}^s$ is supported over the set $\hat{B}^{[n+1]} = \hat{\Gamma}^{[n+1]} \cup \Gamma^s$.

Recall that $\hat{\mathbf{x}}^{[n+1]}_{\hat{B}^{[n+1]}}$ is a better s-term approximation to $\hat{\mathbf{a}}^{[n+1]}_{\hat{B}^{[n+1]}}$ than $\mathbf{x}^{s}_{\hat{B}^{[n+1]}}$ $\big($namely, $\|\hat{\mathbf{x}}^{[n+1]} - \hat{\mathbf{a}}^{[n+1]}_{\hat{B}^{[n+1]}}\|_2 \leq \|\hat{\mathbf{a}}^{[n+1]}_{\hat{B}^{[n+1]}} - \mathbf{x}^{s}\|_2\big)$. Then

$$\mathbb{E}[\|\hat{\mathbf{x}}^{[n+1]} - \mathbf{x}^{s}\|_2|\hat{\mathbf{x}}^{[n]}] \leq 2\mathbb{E}[\|\hat{\mathbf{a}}^{[n+1]}_{\hat{B}^{[n+1]}} - x^{s}_{\hat{B}^{[n+1]}}\|_2|\hat{\mathbf{x}}^{[n]}] \tag{2.17}$$

Using triangle inequality, we further have

$$\mathbb{E}[\|\hat{\mathbf{x}}^{[n+1]} - \mathbf{x}^{s}\|_2|\hat{\mathbf{x}}^{[n]}] \leq 2\big[\mathbb{E}[\|\hat{\mathbf{a}}^{[n+1]}_{\hat{B}^{[n+1]}} - \mathbf{a}^{[n+1]}_{\hat{B}^{[n+1]}}\|_2 + \|\mathbf{a}^{[n+1]}_{\hat{B}^{[n+1]}} - \mathbf{x}^{s}_{\hat{B}^{[n+1]}}\|_2|\hat{\mathbf{x}}^{[n]}]\big] \tag{2.18}$$

We now continue with the analysis referring to two terms on the right hand side of Equation 2.18 separately.
(a) Expanding $\hat{\mathbf{a}}^{[n+1]}_{\hat{B}^{[n+1]}}$ and $\mathbf{a}^{[n+1]}_{\hat{B}^{[n+1]}}$ we have

$$\begin{aligned}
&\mathbb{E}[\|\hat{\mathbf{a}}^{[n+1]}_{\hat{B}^{[n+1]}} - \mathbf{a}^{[n+1]}_{\hat{B}^{[n+1]}}\|_2|\hat{\mathbf{x}}^{[n]}] \\
&\quad = \mathbb{E}[\|\hat{\mu}^{[n]}Q_1(\boldsymbol{\Phi})^{T}_{\hat{B}^{[n+1]}}\big(Q_y(\mathbf{y}) - Q_2(\boldsymbol{\Phi})\hat{\mathbf{x}}^{[n]}\big) - \mu^{[n]}\boldsymbol{\Phi}^{T}_{\hat{B}^{[n+1]}}(\mathbf{y} - \boldsymbol{\Phi}\hat{\mathbf{x}}^{[n]})\|_2|\hat{\mathbf{x}}^{[n]}] \\
&\quad = \mathbb{E}[\|\hat{\mu}^{[n]}Q_1(\boldsymbol{\Phi})^{T}_{\hat{B}^{[n+1]}}\big(\boldsymbol{\Phi}x^{s} + \varepsilon + \varepsilon_y - Q_2(\boldsymbol{\Phi})\hat{x}^{[n]}\big) - \mu^{[n]}\boldsymbol{\Phi}^{T}_{\hat{B}^{[n+1]}}(\Phi\mathbf{x}^{s} + \varepsilon - \boldsymbol{\Phi}\hat{\mathbf{x}}^{[n]})\|_2|\hat{\mathbf{x}}^{[n]}] \\
&\quad = \mathbb{E}[\|\hat{\mu}^{[n]}Q_1(\boldsymbol{\Phi})^{T}_{\hat{B}^{[n+1]}}\big(- Q_2(\boldsymbol{\Phi})\mathbf{r}^{[n]} + \varepsilon + \varepsilon_y \\
&\qquad + (\boldsymbol{\Phi} - Q_2(\boldsymbol{\Phi}))\mathbf{x}^{s}\big) + \mu^{[n]}\boldsymbol{\Phi}^{T}_{\hat{B}^{[n+1]}}(\boldsymbol{\Phi}\mathbf{r}^{[n]} - \varepsilon)\|_2|\hat{\mathbf{x}}^{[n]}] \\
&\quad \leq \|\big(\mu^{[n]}\boldsymbol{\Phi}^{T}_{\hat{B}^{[n+1]}}\boldsymbol{\Phi} - \hat{\mu}^{[n]}Q_1(\boldsymbol{\Phi})^{T}_{\hat{B}^{[n+1]}}Q_2(\boldsymbol{\Phi})\big)\mathbf{r}^{[n]}\|_2 + \|\big(\mu^{[n]}\boldsymbol{\Phi}^{T}_{\hat{B}^{[n+1]}} - \hat{\mu}^{[n]}Q_1(\boldsymbol{\Phi})^{T}_{\hat{B}^{[n+1]}}\big)\varepsilon\|_2 \\
&\qquad + \mathbb{E}[\|\hat{\mu}^{[n]}Q_1(\boldsymbol{\Phi})^{T}_{\hat{B}^{[n+1]}}\varepsilon_y\|_2] + \mathbb{E}[\|\hat{\mu}^{[n]}Q_1(\boldsymbol{\Phi})_{\hat{B}^{[n+1]}}{}^{T}\big(\boldsymbol{\Phi} - Q_2(\boldsymbol{\Phi})\big)\mathbf{x}^{s}\|_2].
\end{aligned} \tag{2.19}$$

where we used the expansion $\mathbf{r}^{[n]} = \hat{\mathbf{x}}^{[n]} - \mathbf{x}^{s}$. We further derive the terms governing the above expression in four steps below.
(a.1) Since $\mathbf{r}^{[n]}$ is supported over $\hat{B}^{[n]}$, we clearly have

$$\begin{aligned}
&\|\big(\mu^{[n]}\boldsymbol{\Phi}^{T}_{\hat{B}^{[n+1]}}\boldsymbol{\Phi} - \hat{\mu}^{[n]}Q_1(\boldsymbol{\Phi})^{T}_{\hat{B}^{[n+1]}}Q_2(\boldsymbol{\Phi})\big)\mathbf{r}^{[n]}\|_2 \\
&\leq \|\big(\mu^{[n]}\boldsymbol{\Phi}^{T}_{\hat{B}^{[n+1]}}\boldsymbol{\Phi}_{\hat{B}^{[n+1]}} - \hat{\mu}^{[n]}Q_1(\boldsymbol{\Phi})^{T}_{\hat{B}^{[n+1]}}Q_2(\boldsymbol{\Phi})_{\hat{B}^{[n+1]}}\big)\mathbf{r}^{[n]}_{\hat{B}^{[n+1]}}\|_2 \\
&\quad + \|\big(\mu^{[n]}\boldsymbol{\Phi}^{T}_{\hat{B}^{[n+1]}}\boldsymbol{\Phi}_{\hat{B}^{[n]}\setminus\hat{B}^{[n+1]}} \\
&\quad - \hat{\mu}^{[n]}Q_1(\boldsymbol{\Phi})^{T}_{\hat{B}^{[n+1]}}Q_2(\boldsymbol{\Phi})_{\hat{B}^{[n]}\setminus\hat{B}^{[n+1]}}\big)\mathbf{r}^{[n]}_{\hat{B}^{[n]}\setminus\hat{B}^{[n+1]}}\|_2.
\end{aligned}$$

Using the second inequality in Lemma 2 we have

$$\|(\mu^{[n]}\boldsymbol{\Phi}^T_{\hat{B}^{[n+1]}}\boldsymbol{\Phi}_{\hat{B}^{[n+1]}} - \hat{\mu}^{[n]}Q_1(\boldsymbol{\Phi})^T_{\hat{B}^{[n+1]}}Q_2(\boldsymbol{\Phi})_{\hat{B}^{[n+1]}})\mathbf{r}^{[n]}_{\hat{B}^{[n+1]}}\|_2 \leq (\gamma_{2s}+\hat{\gamma}_{2s})\|\mathbf{r}^{[n]}_{\hat{B}^{[n+1]}}\|_2. \tag{2.20}$$

Let now $\hat{B}^{[n+1]}$ be split into two disjoint sets Γ_1 and Γ_2, where $\Gamma_1 \cap \Gamma_2 = \emptyset$ and $|\Gamma_1|, |\Gamma_2| \leq s$. By the third inequality in Lemma 2, we have

$$\begin{aligned}
&\|(\mu^{[n]}\boldsymbol{\Phi}_{\hat{B}^{[n+1]}}{}^T\boldsymbol{\Phi}_{\hat{B}^{[n]}\setminus\hat{B}^{[n+1]}} - \hat{\mu}^{[n]}Q_1(\boldsymbol{\Phi})_{\hat{B}^{[n+1]}}{}^TQ_2(\boldsymbol{\Phi})_{\hat{B}^{[n]}\setminus\hat{B}^{[n+1]}})\mathbf{r}^{[n]}_{\hat{B}^{[n]}\setminus\hat{B}^{[n+1]}}\|_2 \\
&\leq \Big(\|(\mu^{[n]}\boldsymbol{\Phi}_{\Gamma_1}{}^T\boldsymbol{\Phi}_{\hat{B}^{[n]}\setminus\hat{B}^{[n+1]}} - \hat{\mu}^{[n]}Q_1(\boldsymbol{\Phi})_{\Gamma_1}{}^TQ_2(\boldsymbol{\Phi})_{\hat{B}^{[n]}\setminus\hat{B}^{[n+1]}})\mathbf{r}^{[n]}_{\hat{B}^{[n]}\setminus\hat{B}^{[n+1]}}\|_2^2 \\
&\quad + \|(\mu^{[n]}\boldsymbol{\Phi}_{\Gamma_2}{}^T\boldsymbol{\Phi}_{\hat{B}^{[n]}\setminus\hat{B}^{[n+1]}} - \hat{\mu}^{[n]}Q_1(\boldsymbol{\Phi})_{\Gamma_2}{}^TQ_2(\boldsymbol{\Phi})_{\hat{B}^{[n]}\setminus\hat{B}^{[n+1]}})\mathbf{r}^{[n]}_{\hat{B}^{[n]}\setminus\hat{B}^{[n+1]}}\|_2^2\Big)^{\frac{1}{2}} \\
&\leq \sqrt{2}(\gamma_{2s}+\hat{\gamma}_{2s})\|\mathbf{r}^{[n]}_{\hat{B}^{[n]}\setminus\hat{B}^{[n+1]}}\|_2.
\end{aligned} \tag{2.21}$$

Combining Equation 2.20 and Equation 2.21,

$$\begin{aligned}
&\|(\mu^{[n]}\boldsymbol{\Phi}^T_{\hat{B}^{[n+1]}}\boldsymbol{\Phi} - \hat{\mu}^{[n]}Q_1(\boldsymbol{\Phi})^T_{\hat{B}^{[n+1]}}Q_2(\boldsymbol{\Phi}))\mathbf{r}^{[n]}\|_2 \\
&= (\gamma_{2s}+\hat{\gamma}_{2s})\|\mathbf{r}^{[n]}_{\hat{B}^{[n+1]}}\|_2 + \sqrt{2}(\gamma_{2s}+\hat{\gamma}_{2s})\|\mathbf{r}^{[n]}_{\hat{B}^{[n]}\setminus\hat{B}^{[n+1]}}\|_2 \leq 2(\gamma_{2s}+\hat{\gamma}_{2s})\|\mathbf{r}^{[n]}\|_2
\end{aligned} \tag{2.22}$$

where the last inequality follows from the fact that $\mathbf{r}^{[n]}_{\hat{B}^{[n+1]}}$ and $\mathbf{r}_{\hat{B}^{[n]}\setminus\hat{B}^{[n+1]}}$ are orthogonal.

(a.2) Expanding the second term in Equation 2.19

$$\begin{aligned}
&\|(\mu^{[n]}\boldsymbol{\Phi}_{\hat{B}^{[n+1]}} - \hat{\mu}^{[n]}Q_1(\boldsymbol{\Phi})^T_{\hat{B}^{[n+1]}})\boldsymbol{\varepsilon}\|_2 \\
&\leq \|(\mu^{[n]}\boldsymbol{\Phi}_{\hat{B}^{[n+1]}} - \hat{\mu}^{[n]}Q_1(\boldsymbol{\Phi})^T_{\hat{B}^{[n+1]}})\mathbf{e}\|_2 + \|(\mu^{[n]}\boldsymbol{\Phi}_{\hat{B}^{[n+1]}} - \hat{\mu}^{[n]}Q_1(\boldsymbol{\Phi})^T_{\hat{B}^{[n+1]}})\boldsymbol{\Phi}(\mathbf{x}-\mathbf{x}^s)\|_2.
\end{aligned} \tag{2.23}$$

Using Equation 2.12 and Lemma 2 we have

$$\begin{aligned}
&\|(\mu^{[n]}\boldsymbol{\Phi}_{\hat{B}^{[n+1]}} - \hat{\mu}^{[n]}Q_1(\boldsymbol{\Phi})^T_{\hat{B}^{[n+1]}})e\|_2 \\
&\le \max\left((1+\gamma_{2s})/\beta_{2s}, (1+\hat{\gamma}_{2s})/\hat{\beta}_{2s}\right)\|\mathbf{e}\|_2\|(\mu^{[n]}\boldsymbol{\Phi}_{\hat{B}^{[n+1]}} - \hat{\mu}^{[n]}Q_1(\boldsymbol{\Phi})^T_{\hat{B}^{[n+1]}})\boldsymbol{\Phi}(\mathbf{x}-\mathbf{x}^s)\|_2 \\
&\le \left(\|(\mu^{[n]}\boldsymbol{\Phi}_{\Gamma_1} - \hat{\mu}^{[n]}Q_1(\boldsymbol{\Phi})^T_{\Gamma_1})\Phi(\mathbf{x}-\mathbf{x}^s)\|_2^2 + \|(\mu^{[n]}\boldsymbol{\Phi}_{\Gamma_2} - \hat{\mu}^{[n]}Q_1(\boldsymbol{\Phi})^T_{\Gamma_2})\boldsymbol{\Phi}(\mathbf{x}-\mathbf{x}^s)\|_2^2\right)^{1/2} \\
&\le \sqrt{2}(\hat{\gamma}_{2s}+\hat{\gamma}_{2s})\left[\|\mathbf{x}-\mathbf{x}^s\|_2 + \frac{\|\mathbf{x}-\mathbf{x}^s\|_1}{\sqrt{s}}\right].
\end{aligned} \tag{2.24}$$

Combining results obtained in Equation 2.24

$$\begin{aligned}
&\|(\mu^{[n]}\boldsymbol{\Phi}_{\hat{B}^{[n+1]}} - \hat{\mu}^{[n]}Q_1(\boldsymbol{\Phi})^T_{\hat{B}^{[n+1]}})\boldsymbol{\varepsilon}\|_2 \\
&\le\ \max\left((1+\gamma_{2s})/\beta_{2s}, (1+\hat{\gamma}_{2s})/\hat{\beta}_{2s}\right)\|\mathbf{e}\|_2 + \sqrt{2}(\gamma_{2s}+\hat{\gamma}_{2s})\left[\|\mathbf{x}-\mathbf{x}^s\|_2 + \frac{\|\mathbf{x}-\mathbf{x}^s\|_1}{\sqrt{s}}\right].
\end{aligned} \tag{2.25}$$

(a.3) The third term of Equation 2.19

$$\mathbb{E}[\|\hat{\mu}^{[n+1]}Q_1(\boldsymbol{\Phi})^T_{\hat{B}^{[n+1]}}\boldsymbol{\varepsilon}_y\|_2] \overset{(1)}{\le} \frac{(1+\hat{\gamma}_{2s})}{\hat{\beta}_{2s}}\mathbb{E}[\|\boldsymbol{\varepsilon}_y\|_2] \overset{(2)}{\le} \frac{(1+\hat{\gamma}_{2s})c_y\sqrt{M}}{\hat{\beta}_{2s}2^{b_y-1}} \tag{2.26}$$

where the inequalities follows from (1) Equation 2.11 together with Remark 1, and (2) Lemma 4.

(a.4) Combining with Equation 2.11, Remark 1, Cauchy-Bunyakovsky-Schwarz, Jensen inequalities and the similar discussion above

$$\begin{aligned}
\mathbb{E}[\|\hat{\mu}^{[n]}Q_1(\Phi)^T_{\hat{B}^{[n+1]}}(\boldsymbol{\Phi}-Q_2(\boldsymbol{\Phi}))\mathbf{x}^s\|_2] &\le \frac{(1+\hat{\gamma}_{2s})}{\hat{\beta}_{2s}}\mathbb{E}[\|(\boldsymbol{\Phi}-Q_2(\boldsymbol{\Phi}))\mathbf{x}^s\|_2] \\
\le \frac{(1+\hat{\gamma}_{2s})}{\hat{\beta}_{2s}}\sqrt{\sum_i^M\sum_j^N \mathbb{E}[(\boldsymbol{\Phi}_{i,j}-Q_2(\boldsymbol{\Phi}_{i,j})\mathbf{x}_j^s)^2]} &= \frac{(1+\hat{\gamma}_{2s})c_{\boldsymbol{\Phi}}\sqrt{M}}{\hat{\beta}_{2s}2^{b_{\Phi}-1}}\|\mathbf{x}^s\|_2.
\end{aligned} \tag{2.27}$$

(b) Finally, we bound the second term on the right hand side of Equation 2.18 as follows.

$$\begin{aligned}
&\|\mathbf{a}_{\hat{B}[n+1]}^{[n+1]} - \mathbf{x}_{\hat{B}[n+1]}^{s}\|_2 = \|\hat{\mathbf{x}}_{\hat{B}[n+1]}^{[n]} + \mu^{[n]}\mathbf{\Phi}_{\hat{B}[n+1]}^{T}(\mathbf{y} - \mathbf{\Phi}\hat{\mathbf{x}}^{[n]}) - \mathbf{x}_{\hat{B}[n+1]}^{s}\|_2 \\
&= \|\hat{\mathbf{x}}_{\hat{B}[n+1]}^{[n]} + \mu^{[n]}\mathbf{\Phi}_{\hat{B}[n+1]}^{T}(\mathbf{\Phi}\mathbf{x}^{s} + \boldsymbol{\varepsilon} - \mathbf{\Phi}\hat{\mathbf{x}}^{[n]}) - \mathbf{x}_{\hat{B}[n+1]}^{s}\|_2 \\
&= \|\mathbf{r}_{\hat{B}[n+1]}^{[n]} - \mu^{[n]}\mathbf{\Phi}_{\hat{B}[n+1]}^{T}(\mathbf{\Phi}\mathbf{r}^{[n]} - \boldsymbol{\varepsilon})\|_2 \\
&= \|\mathbf{r}_{\hat{B}[n+1]}^{[n]} - \mu^{[n]}\mathbf{\Phi}_{\hat{B}[n+1]}^{T}(\mathbf{\Phi}_{\hat{B}[n+1]}\mathbf{r}_{\hat{B}[n+1]}^{[n]} + \mathbf{\Phi}_{\hat{B}[n]\setminus\hat{B}[n+1]}\mathbf{r}_{\hat{B}[n]\setminus\hat{B}[n+1]}^{[n]} - \boldsymbol{\varepsilon})\|_2 \\
&\leq \|(\boldsymbol{I} - \mu^{[n]}\mathbf{\Phi}_{\hat{B}[n+1]}^{T}\mathbf{\Phi}_{\hat{B}[n+1]})\mathbf{r}_{\hat{B}[n+1]}^{[n]}\|_2 + \|\mu^{[n]}\mathbf{\Phi}_{\hat{B}[n+1]}^{T}\mathbf{\Phi}_{\hat{B}[n]\setminus\hat{B}[n+1]}\mathbf{r}_{\hat{B}[n]\setminus\hat{B}[n+1]}^{[n]}\|_2 \\
&\qquad + \|\mu^{[n]}\mathbf{\Phi}_{\hat{B}[n+1]}^{T}\boldsymbol{\varepsilon}\|_2.
\end{aligned} \tag{2.28}$$

It can be verified by using Equation 2.5, Remark 1 and Equation 2.12 that

$$\begin{aligned}
&\|(\boldsymbol{I} - \mu^{[n]}\mathbf{\Phi}_{\hat{B}[n+1]}{}^{T}\mathbf{\Phi}_{\hat{B}[n+1]})\mathbf{r}_{\hat{B}[n+1]}^{[n]}\|_2 \\
&\overset{(1)}{\leq} \gamma_{2s}\|\mathbf{r}_{\hat{B}[n+1]}^{[n]}\|_2\|\mu^{[n]}\mathbf{\Phi}_{\hat{B}[n+1]}^{T}\mathbf{\Phi}_{\hat{B}[n]\setminus\hat{B}[n+1]}\mathbf{r}_{\hat{B}[n]\setminus\hat{B}[n+1]}^{[n]}\|_2 \\
&\leq \left(\|\mu^{[n]}\mathbf{\Phi}_{\Gamma_1}^{T}\mathbf{\Phi}_{\hat{B}[n]\setminus\hat{B}[n+1]}\mathbf{r}_{\hat{B}[n]\setminus\hat{B}[n+1]}^{[n]}\|_2^2 + \|\mu^{[n]}\mathbf{\Phi}_{\Gamma_2}^{T}\mathbf{\Phi}_{\hat{B}[n]\setminus\hat{B}[n+1]}\mathbf{r}_{\hat{B}[n]\setminus\hat{B}[n+1]}^{[n]}\|_2^2\right)^{1/2} \\
&\overset{(2)}{\leq} \sqrt{2}\gamma_{2s}\|\mathbf{r}_{\hat{B}[n]\setminus\hat{B}[n+1]}^{[n]}\|_2\|\mu^{[n]}\mathbf{\Phi}_{\hat{B}[n+1]}^{T}\varepsilon\|_2 \\
&\overset{(3)}{\leq} \frac{1+\gamma_{2s}}{\beta_{2s}}\|\mathbf{e}\|_2 + \sqrt{2}(1+\gamma_{2s})\left[\|\mathbf{x} - \mathbf{x}^{s}\|_2 - \frac{\|\mathbf{x} - \mathbf{x}^{s}\|_1}{\sqrt{s}}\right].
\end{aligned} \tag{2.29}$$

By the orthogonality between $\mathbf{r}_{\hat{B}[n+1]}^{[n]}$ and $\mathbf{r}_{\hat{B}[n]\setminus\hat{B}[n+1]}^{[n]}$, (2.28) can further be simplified to

$$\begin{aligned}
&\|\mathbf{a}_{\hat{B}[n+1]}^{[n+1]} - \mathbf{x}_{\hat{B}[n+1]}^{s}\|_2|\mathbf{x}^{[n]} \\
&\qquad \leq 2\gamma_{2s}\|\mathbf{r}^{[n]}\|_2 + \frac{1+\gamma_{2s}}{\beta_{2s}}\|\mathbf{e}\|_2 + \sqrt{2}(1+\gamma_{2s})\left[\|x - \mathbf{x}^{s}\|_2 - \frac{\|\mathbf{x} - \mathbf{x}^{s}\|_1}{\sqrt{s}}\right].
\end{aligned} \tag{2.30}$$

Substituting Equation 2.22, Equation 2.25, Equation 2.26, Equation 2.27 and Equation 2.30 into Equation 2.17, the norm of recovery error is given by

$$\begin{aligned}\mathbb{E}[\|\mathbf{r}^{[n+1]}\|_2|\mathbf{r}^{[n]}] &\leq 12\max(\gamma_{2s},\hat{\gamma}_{2s})\|\mathbf{r}^{[n]}\|_2 + 4\max\left(\frac{1+\gamma_{2s}}{\beta_{2s}},\frac{1+\hat{\gamma}_{2s}}{\hat{\beta}_{2s}}\right)\|\mathbf{e}\|_2\\ &+ 2\sqrt{2}(3\max(\gamma_{2s},\hat{\gamma}_{2s})+1)\left[\|\mathbf{x}-\mathbf{x}^s\|_2 + \frac{\|\mathbf{x}-\mathbf{x}^s\|_1}{\sqrt{s}}\right]\\ &+ 2\frac{(1+\hat{\gamma}_{2s})\sqrt{M}}{\hat{\beta}_{2s}}\left(\frac{c_\Phi\|x^s\|_2}{2^{b_\Phi-1}}+\frac{c_y}{2^{b_y-1}}\right)\end{aligned} \tag{2.31}$$

Let $\gamma_{2s}, \hat{\gamma}_{2s} \leq t$. For $t \leq 1/24$, we have

$$\begin{aligned}\mathbb{E}[\|\hat{\mathbf{x}}^{[n+1]} - \mathbf{x}^s\|_2|\hat{\mathbf{x}}^{[0]} = 0] &\leq 2^{-n}\|\mathbf{x}^s\|_2 + \frac{8.4}{\min(\beta_{2s},\hat{\beta}_{2s})}\|\mathbf{e}\|_2\\ &+ 6.4\left[\|\mathbf{x}-\mathbf{x}^s\|_2 + \frac{\|\mathbf{x}-\mathbf{x}^s\|_1}{\sqrt{s}}\right] + \frac{4.2\sqrt{M}}{\hat{\beta}_{2s}}\left(\frac{c_\Phi\|x^s\|_2}{2^{b_\Phi-1}}+\frac{c_y}{2^{b_y-1}}\right)\end{aligned}$$

and using the following notation:

$$\varepsilon_s := \|\mathbf{x}-\mathbf{x}^s\|_2 + \frac{\|\mathbf{x}-\mathbf{x}^s\|_1}{\sqrt{s}} + \frac{1}{\min(\beta_{2s},\hat{\beta}_{2s})}||\mathbf{e}||_2$$

$$\varepsilon_q := \frac{\sqrt{M}}{\hat{\beta}_{2s}}\left(\frac{\|c_\Phi\mathbf{x}^s\|_2}{2^{b_\Phi-1}}+\frac{c_y}{2^{b_y-1}}\right)$$

we finally have

$$\mathbb{E}[\|\hat{\mathbf{x}}^{[n+1]} - x^s\|_2|\hat{\mathbf{x}}^{[0]} = 0] \leq 2^{-n}\|\mathbf{x}^s\|_2 + 9\varepsilon_s + 4.5\varepsilon_q.$$

□

Proof of Corollary 2. Inserting $n^* = \lceil\log_2(\|\mathbf{x}^s\|/\varepsilon_s)\rceil$ into the $2^{-n}\|\mathbf{x}^s\|$ term in Equation 2.9 yields the result. □

Proof of Lemma 1. Assume that $\mathbf{\Phi}_\Gamma$ has the singular values confined in $[\alpha_{|\Gamma|}, \beta_{|\Gamma|}]$. Through the perturbation of singular values of a matrix upon corruption of entries with noise, it is shown that Bernoulli noise, corrupting the entries of the matrix independently, lifts up the singular values of the matrix, and at most by $\sigma_{\max}\sqrt{|\Gamma|}$ where $\sigma_{\max}$ is the maximum of the noise standard deviations (Stewart, 1990; Stewart, 2006; Vaccaro and Kot, 1987). Therefore, singular values of $\hat{\mathbf{\Phi}}_\Gamma$ is in $[\alpha_{|\Gamma|}, \beta_{|\Gamma|} + \sigma_{\max}\sqrt{|\Gamma|}]$. Moreover, we previously showed that the variance of the quantization noise is at most $1/2^{b-1}$, hence we have $\sigma_{\max} = 1/2^{b-1}$. Thus, $\hat{\gamma}_{|\Gamma|}$ satisfies

$$\hat{\gamma}_{|\Gamma|} \leq \gamma_{|\Gamma|} + \frac{\sqrt{|\Gamma|}}{2^{b-1}\alpha_{|\Gamma|}}$$

The above equation guarantees that whenever $\gamma_{|\Gamma|} + \varepsilon \leq 1/24$, for some $\varepsilon \geq \frac{\sqrt{|\Gamma|}}{2^{b-1}\alpha_{|\Gamma|}}$, $\hat{\gamma}_{|\Gamma|}$ is guaranteed to be lower than $1/24$. □

2.5.4 *Discussion and Limitations*

We now examine the error bound provided by Theorem 4. We note that this bound is slightly simplified, that is, our proof above is tighter. From there, we conclude that the RIP condition is scaled by a small factor which lies in the interval $(2, 3)$ when the measurement matrix is quantized.

The QIHT algorithm is guaranteed to asymptotically provide a sparse approximation of $\mathbf{x}$ up to multiples of ε_s and ε_q in the noise term $\mathbf{e}$ when $\gamma_{2s}, \hat{\gamma}_{2s} \leq 1/12$, and with rate 2^{-n} when $\gamma_{2s}, \hat{\gamma}_{2s} \leq 1/24$. We refer to Equation 2.31 for the details of the former. ε_s is the approximation error when $\mathbf{x}$ is represented by a sparse vector $\mathbf{x}^s$, and ε_q is the noise introduced by the quantization operator.

CONDITION ON $\hat{\gamma}_{2s}$ Compared to Normalized IHT, the condition under which the performance guarantee holds is stricter in our approach, i.e., $\gamma_{2s}, \hat{\gamma}_{2s} \leq 1/24$, whereas the standard analysis requires $\gamma_{2s} \leq 1/8$ Theorem 2 for the same rate of convergence. Although it is hard to meet this constraint in practice, the small scaling factor between the convergence rates suggests that we can still expect good practical performance in the low precision setting, similarly to high precision Normalized IHT.

LIMITATIONS ON β_{2s} AND $\hat{\beta}_{2s}$ In QIHT, the measurement matrix $\mathbf{\Phi}$ is scale-invariant, and rigorous theoretical guarantees are achievable provided its scaling onto sparse vectors is confined in certain intervals, i.e., the RIP condition.

The recovery error bound satisfying Theorem 4 depends on the error terms in Equation 2.8, ε_s and ε_q, which are inversely proportional to β_{2s} and $\hat{\beta}_{2s}$, respectively. For sufficiently large values, which would compensate for $\|\mathbf{e}\|$ and $\sqrt{M}\|\mathbf{x}^s\|$, the low precision approach appears competitive with the unmodified algorithm where the recovery error is bounded by $9\varepsilon_s$ in Equation 2.6. Furthermore, the scale-invariance property of the measurement matrix Φ permits us to scale up β_{2s}, and hence $\hat{\beta}_{2s}$, retaining a strong recovery guarantee, similar to that of the full precision algorithm. Scaling Φ has no effect on the RIP condition.

ON THE QUANTIZATION ERROR ε_q From the definition of ε_q we infer that the quantization errors introduced by the low precision measurement matrix and the measurements individually differ by a scale factor of $\|\mathbf{x}_s\|$ for the same bit widths. We argue that the approximation error caused by quantizing the measurement matrix would get smaller as s gets smaller. Moreover, the scale invariant property of the measurement matrix can enable $\|\mathbf{x}_s\| < 1$ to hold, yet can potentially strengthen the effect of noise.

COMPARISON TO OTHER STATE-OF-THE-ART The compressive sensing literature covers a range of algorithms including ℓ_1-minimization and greedy- and thresholding-based methods, each with its own trade-offs. CoSaMP, normalized IHT and ℓ_1-minimization exhibit similar empirical performance in (Blumensath and Davies, 2010), when applied to the problems with dense Gaussian matrices. Moreover, after tuning of the step size parameter, normalized IHT is competitive to these powerful methods with similar provable guarantees (Blumensath et al., 2012). Considering that the performance of normalized IHT compared to other state-of-the-art methods is already well-studied in the literature and is superior in most cases, we focus only on comparing QIHT to the normalized IHT in this work.

2.6 EXPERIMENTAL VALIDATION

The goal of this section is to examine the practical performance of our method. To provide more intuition, we first run synthetic experiments to quantify the performance gap between QIHT and normalized IHT on a toy example: artificially generated data where the data points are drawn from i.i.d. Gaussian distributions. It is shown, for instance in (Baraniuk et al., 2008; Xu, Wang, and Shim, 2014), that the Gaussian matrices satisfy the RIP with high probability. The choice of such experimental data therefore helps us to better understand how the performance gap scales with the reduced number of precision levels, in a regime where the theoretical conditions do hold.

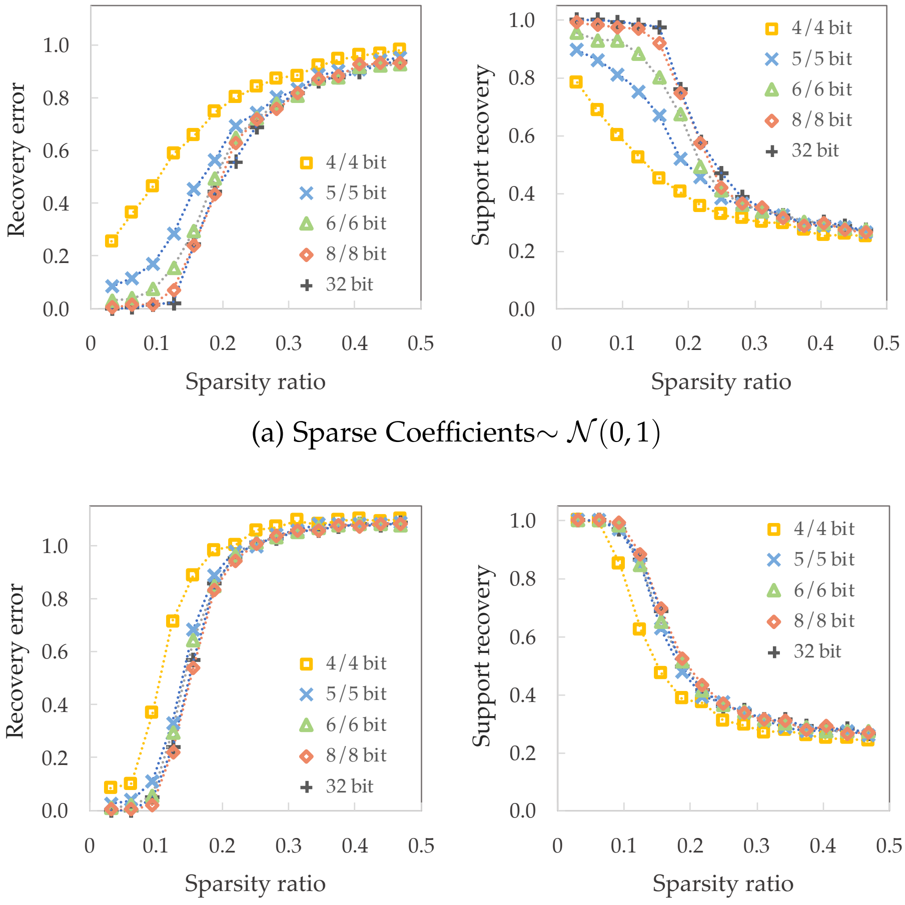
We then extend our focus to real-world larger-scale problems from radio astronomy and magnetic resonance imaging. We apply QIHT to (a) the radio interferometer measurements recorded by a real telescope: the LOw Frequency ARray (LOFAR)[2], and (b) k-space subsamples recorded from the two-dimensional Fourier domain of a representative brain image. For both applications, we model the imaging problem in the iterative hard thresholding framework and demonstrate that the accuracy achieved in the low precision setting is comparable with the one obtained by high precision solvers. Finally, we examine the speedups obtained by FPGA and CPU implementations.

2.6.1 *Experiments on Synthetic Data*

DATA We draw the entries of $\mathbf{\Phi} \in \mathbb{R}^{128\times1024}$, $\mathbf{x}^s \in \mathbb{R}^{1024}$ and $\mathbf{e} \in \mathbb{R}^{128}$ from an i.i.d. Gaussian distribution with zero mean and unit variance, $\mathcal{N}(0,1)$, such that the sparsity of $\mathbf{x}$, i.e., $s = |\operatorname{supp}(\mathbf{x})|$, varies from 4 to 128 in steps of 4. Similar to (Blumensath and Davies, 2010) we compute the recovery error using different error measures and by averaging over 1,000 realizations of data.

ACCURACY We first compare the recovery performance of QIHT to IHT in the absence of noise, i.e., $\mathbf{y} = \mathbf{\Phi x}$. To quantify the performance gap, we

[2] https://www.astron.nl/telescopes/lofar/



(a) Sparse Coefficients$\sim \mathcal{N}(0, 1)$

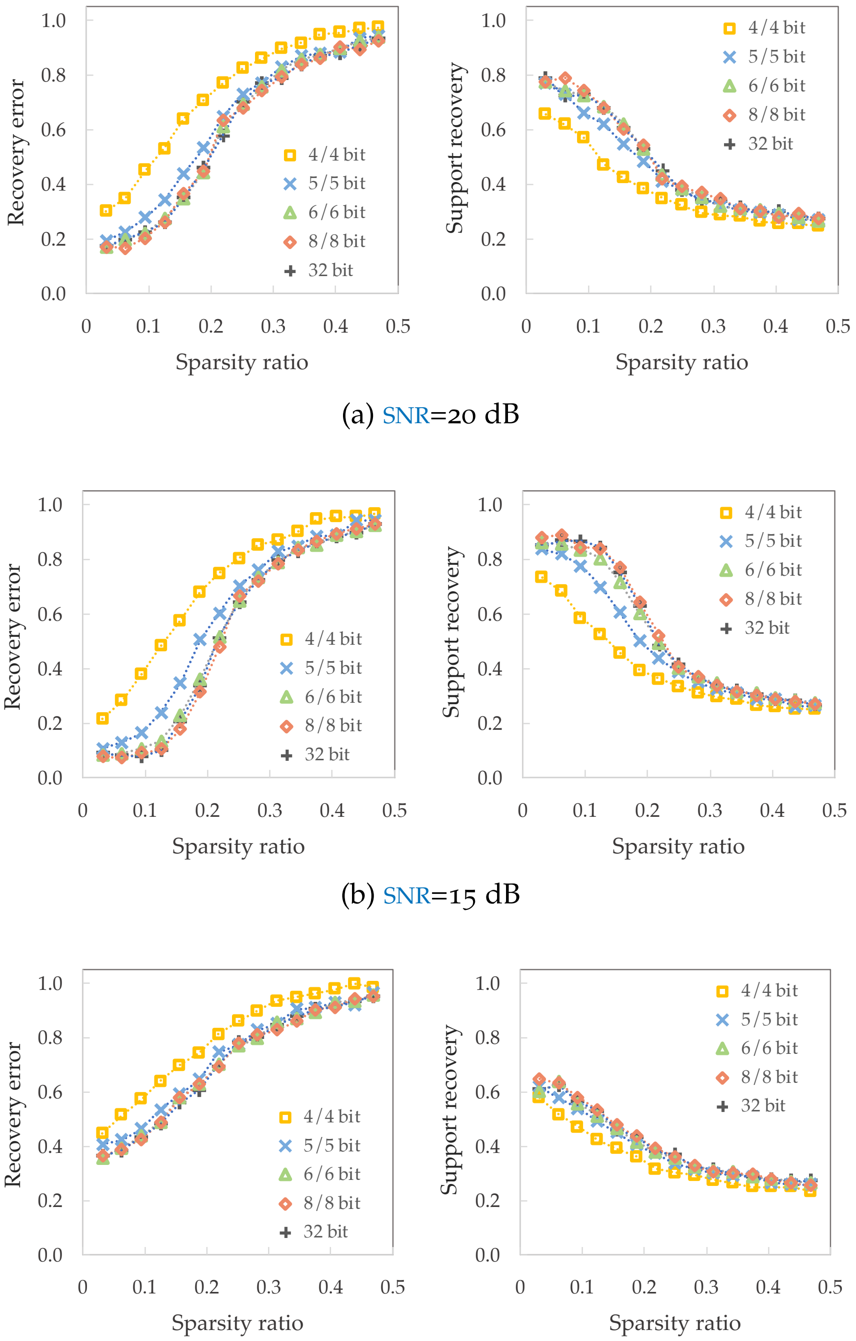
(b) Sparse Coefficients=1

Figure 2.3: Recovery error and support recovery of QIHT at different bit precision levels for (a) Gaussian and (b) unity sparse coefficients.

estimate (a) the recovery error: $\|\mathbf{x}^n - \mathbf{x}\| / \|\mathbf{x}\|$, and (b) the support recovery, i.e., the normalized support of $\mathbf{x}$ that is successfully recovered. We estimate the above measures by averaging over 100 realizations of data.

The results shown in Figure 2.3(a) indicate that QIHT can achieve a recovery performance that is close to the normalized IHT even when as few as 5 bits are used. As expected, QIHT performs slightly worse when the precision is too aggresively lowered, for instance, down to $b_\Phi = 4$ and $b_y = 4$ bits. Yet the precision levels that preserve the quality of the results still can provide a significant speed-up in computation time for recovery.

We now consider a case, which is often considered challenging for sparse recovery algorithms: when the entries of $\mathbf{x}$ are of equal magnitude. We repeated the above experiment by setting the nonzero entries of $\mathbf{x}$ to 1 and demonstrate



(a) SNR=20 dB

(b) SNR=15 dB

(c) SNR=10 dB

Figure 2.4: Recovery error and the support recovery of QIHT at different bit precision levels and for different amounts of noise corruptions. SNR level is given by $10 \log_{10} \|\mathbf{x}^*\|^2 / \|\mathbf{e}\|^2$.

the performance of QIHT at several precision levels in Figure 2.3(b). While normalized IHT in general seems markedly worse at this setting, QIHT yields a performance as good as its high precision variant and the recovery gap between both methods becomes negligible.

This can be justified as follows: When the data is at low precision, the sparse coefficients are recovered by repeatedly using a linear transformation that has only a few number of precision levels, enforcing the sparse coefficients that are close to each other to be recovered as the similar magnitude. Hence, when the coefficients are of the same value, the precision of the recovered values is not important, leading eventually to less recovery error.

ROBUSTNESS TO NOISE In real-life applications, measurements are usually corrupted by noise. The theoretical bounds of the low precision variant of the Normalized IHT on recovery error, as given in Equation 2.6, Equation 2.7 and Equation 2.9, suggest that lowering the precision of the input data only slightly increases the noise sensitivity.

We therefore investigate the influence of lowering the precision on the recovery performance by corrupting the observations with different levels of noise. Figure 2.4 demonstrates the performance of QIHT for various levels of noise corruption, validating our theoretical observations that quantization does not amplify the effect of noise corruption on the sparse recovery.

Comparison of the normalized IHT to other state-of-the-art methods such as CoSaMP, ℓ_1-minimization for similarly generated artificial data is performed, for example, in (Blumensath and Davies, 2010; Blumensath et al., 2012). We defer to these references for further comparison.

2.6.2 *Real-World Applications*

Motivated by the success of QIHT on artificially generated data, we apply our framework to two larger scale real-world settings in radio astronomy and magnetic resonance imaging. The measurement matrix used in the compressive sensing formulation for these applications contains spatial information on a two-dimensional Fourier space, i.e., relative distances between entries are induced by the respective sensor locations. Therefore, useful information in the linear transformation matrix is preserved even when the precision is lowered, which results in little loss of visual information for the underlying image we aim to recover. In the following experiments, we show that this intuition is indeed correct and confirmed by the good performance of QIHT.

2.6.2.1 *Radio Astronomy*

We consider a radio astronomy application, in which radio interferometers at various locations on the ground record radio waves emitted by the celestial sources over a certain time interval, and then store and process these signals to deduce a sky image (Högbom, 1974). Interferometers first estimate the cross-correlation between the time series measurements, called visibilities. The visibilities correspond to subsamples of a sky map in the Fourier domain where the sample point is a function of the antenna locations (van Cittert-Zernike theorem (Taylor and Carilli, 1999)). For the point source recovery problem, radio interferometry imaging inherently can be formulated as a sparse signal recovery problem.

The usual strategy to date is to deconvolve the inverse Fourier transform of visibilities to form a sky map by iteratively removing a fraction of the highest peak, convolved with an instrument-based point spread function (Högbom, 1974). Moreover, recently, the radio astronomy community has started to formalize the radio interferometer problems also as compressive sensing (Li, Cornwell, and Hoog, 2011; Wenger et al., 2010; Wiaux et al., 2009).

The following is a standard formulation of the problem. Assume the sky is observed by employing L antennas over a stationary time interval where the earth's rotation is negligible. Denote the vectorized sky image by $\mathbf{x} \in \mathbb{R}^N$ with $N = r^2$ where r is the resolution of images, i.e., height and width of the image in pixels.

We formulate the interferometer pipeline as a compressive sensing problem such that $\mathbf{y} = \mathbf{\Phi x} + \mathbf{e}$ where $\mathbf{\Phi} \in \mathbb{C}^{M \times N}$ is the measurement matrix with complex entries as a function of the inner product between antenna and pixel locations, $\mathbf{y} \in \mathbb{C}^M$ contains the visibilities where $M = L^2$, and $\mathbf{e} \in \mathbb{C}^M$ is the noise vector. In what follows, we derive the measurement matrix $\mathbf{\Phi}$ from the baseband representation of interferometric radio signals. We note that the baseband representation has been previously derived in this fashion by (Ocal et al., 2015; Simeoni, 2015).

FORMATION OF $\mathbf{\Phi}$ We begin with the following modeling assumptions (Taylor, Carilli, and Perley, 1999).

(a) Celestial sources are in far field, the series emanating from sources and captured by the antennas are thus parallel,

(b) Series emitted by celestial sources are narrow band zero mean circularly-symmetric complex Gaussian processes,

(c) Series originating from different directions in the sky are uncorrelated.

(a) follows from the assumption sources lie on a hypothetical sphere, the so-called celestial sphere. This implies that we can not measure how far the sources are.

Let $\hat{s}(t, \mathbf{r})$ denote the series emitted by the source coming from the direction $\mathbf{r} \in \mathbb{S}^2$. (b), therefore, implies that $\hat{s}(t, \mathbf{r}) \sim \mathbb{C}\mathcal{N}(0, I(\mathbf{r}))$ where $I(\mathbf{r})$ is the intensity of the source series emanating from direction $\mathbf{r}$. Given the center frequency $f_0 \in \mathbb{R}$, (b) yields the following baseband representation of $\hat{s}(t, \mathbf{r})$

$$s(t, \mathbf{r}) = \hat{s}(t, \mathbf{r})e^{j2\pi f_0 t}. \tag{2.32}$$

Note also that $\mathbb{E}[s(t, \mathbf{r}_1)s^*(t, \mathbf{r}_2)] = 0$, for all $\mathbf{r}_1, \mathbf{r}_2 \in \mathbb{S}^2$ follows from (c), where $*$ denotes the conjugate operator. The series coming from direction $\mathbf{r}$ and recorded by antenna i is thus given by

$$x_i(t, \mathbf{r}) = s(t - \tau_i(\mathbf{r}), \mathbf{r}). \tag{2.33}$$

where $\tau_i(\mathbf{r})$ denotes the time delay for the series reaching from the source to reach at the antenna i. Combining Equation 2.32 and Equation 2.33 we have

$$x_i(t, \mathbf{r}) = \hat{s}(t - \tau_i(\mathbf{r}), \mathbf{r})e^{j2\pi f_0 (t - \tau_i(\mathbf{r}))}. \tag{2.34}$$

Above derivations concern merely a specific source. We now focus on the series measured by the antenna i coming from multiple sources (from all directions), whose formal expression is given by

$$x_i(t) = \oiint\limits_{\mathbb{S}^2} \hat{s}(t - \tau_i(\mathbf{r}), \mathbf{r}) e^{j2\pi f_0 (t - \tau_i(\mathbf{r}))} d\mathbf{r}. \tag{2.35}$$

As correlations between the antenna time-series are of special interest of imaging, we will have a closer look at the algebraic equations regarding the correlations. Recall that the source series coming from different directions in space are uncorrelated. Using this, we have the following closed-form expression:

$$\begin{aligned} &\mathbb{E}[x_i(t)x_k(t)^*] \\ &= \oiint\limits_{\mathbb{S}^2} \mathbb{E}[\hat{s}(t - \tau_i(\mathbf{r}), \mathbf{r})\hat{s}^*(t - \tau_k(\mathbf{r}), \mathbf{r})] e^{-j2\pi f_0 (\tau_i(\mathbf{r}) - \tau_k(\mathbf{r}))} d\mathbf{r} \end{aligned} \tag{2.36}$$

where i and k denote different antennas.

By the assumption of (b), the series $\hat{s}(t, \mathbf{r})$ remain constant over the time shift we further have

$$\mathbb{E}[x_i(t)x_k(t)^*] = \oiint\limits_{\mathbb{S}^2} I(\mathbf{r}) e^{-j2\pi f_0 (\tau_i(\mathbf{r}) - \tau_k(\mathbf{r}))} d\mathbf{r} \tag{2.37}$$

where $I(\mathbf{r})$ denotes the variance of the series emitted from direction $\mathbf{r}$ referred to as sky image. We further have

$$\tau_i(\mathbf{r}) = \frac{1}{c} \langle \mathbf{r}_{\text{norm}}, \mathbf{p}_i \rangle \tag{2.38}$$

where $\mathbf{r}_{\text{norm}} = \frac{\mathbf{r}}{\|\mathbf{r}\|_2}$ and $\mathbf{p}_i$ denotes the position of antenna i.

Above relation together with Equation 2.37 gives us that

$$\mathbb{E}[x_i(t)x_k(t)^*] = \oiint\limits_{\mathbb{S}^2} I(\mathbf{r}) e^{-j2\pi \frac{f_0}{c} \langle \mathbf{r}_{\text{norm}},\, \mathbf{p}_i - \mathbf{p}_k \rangle} d\mathbf{r}. \tag{2.39}$$

Note that $\frac{f_0}{c} = \frac{1}{\lambda_0}$ where λ_0 is the wavelength of the observation, Equation 2.39 can be further simplified to

$$\mathbb{E}[x_i(t)x_k(t)^*] = \oiint_{\mathbb{S}^2} I(\mathbf{r})e^{-j2\pi\langle \mathbf{r}_{\text{norm}}, \frac{\mathbf{p}_i - \mathbf{p}_k}{\lambda_0}\rangle} d\mathbf{r} \tag{2.40}$$

so-called *measurement equation*.

Consider now a specific region of interest centered around $\mathbf{r}_0$ and basis vectors $\hat{\mathbf{e}}_1$, $\hat{\mathbf{e}}_2$ and $\hat{\mathbf{e}}_3$ where $\hat{\mathbf{e}}_1$ points in the direction of rotation of the earth, $\hat{\mathbf{e}}_3$ denotes the direction of $\mathbf{r}_0$ and $\hat{\mathbf{e}}_2$ is perpendicular to $\hat{\mathbf{e}}_1$ and $\hat{\mathbf{e}}_3$. $\mathbf{r}$ can be therefore approximated to $r \approx l\hat{\mathbf{e}}_1 + m\hat{\mathbf{e}}_2 + 1\hat{\mathbf{e}}_3$[3]. Similarly, $\frac{\mathbf{p}_i - \mathbf{p}_k}{\lambda_0}$ can also be expressed in terms of the basis vectors, that is, $\frac{\mathbf{p}_i - \mathbf{p}_k}{\lambda_0} = u_{i,k}\hat{\mathbf{e}}_1 + v_{i,k}\hat{\mathbf{e}}_2 + w_{i,k}\hat{\mathbf{e}}_3$. Substituting these into Equation 2.40, we get

$$\begin{aligned}&\mathbb{E}[x_i(t)x_k^*(t)] \\ &= \iint_{K\subset\mathbb{R}^2} I(l,m)\frac{\exp\left(-j2\pi(w_{i,k}(\sqrt{1-l^2-m^2}-1) + u_{i,k}l + v_{i,k}m)\right)}{\sqrt{1-l^2-m^2}}\, dl\, dm\end{aligned} \tag{2.41}$$

where K is the compact support of $I \in \mathbb{R}^2$. The above equation is called *tangent plane measurement equation*.

As we indicated earlier, FoV is small pointing $\mathbf{r}_0$ leading to small values of l and m. Hence $\frac{\exp(-j2\pi w_{i,k})}{\sqrt{1-l^2-m^2}-1}$ and $\sqrt{1-l^2-m^2}$ terms can be assumed to be constant. Above representation can therefore be further simplified to

$$\mathbb{E}[x_i(t)x_k^*(t)] = \iint_{K\subset\mathbb{R}^2} I(l,m)\exp\big(-j2\pi(w_{i,k} + u_{i,k}l + v_{i,k}m)\big) dldm. \tag{2.42}$$

Definition 1 (Taylor and Carilli, 1999)**.** The *visibility* function is defined by removing the constant phase $\exp(-j2\pi w_{i,k})$ as follows:

$$V(u,v) \overset{\text{def}}{=} \iint_{K\subset\mathbb{R}^2} I(l,m)\exp\left(-j2\pi(ul + vm)\right)\, dldm. \tag{2.43}$$

[3] More clearly; $(\mathbf{r} - \mathbf{r}_0) + \mathbf{r}_0 \approx l\hat{\mathbf{e}}_1 + m\hat{\mathbf{e}}_2 + 1\hat{\mathbf{e}}_3$.

Therefore, each sample of this function is given by $V(u_{i,k}, v_{i,k}) = \mathbb{E}[x_i(t)x_k^*(t)]$ the so-called *visibility* measurement.

Remark that *visibility* equation is an integration of the product of sky intensity and a complex exponential over the unit sphere, which can be represented as a two dimensional Fourier transform. Therefore, visibility function is equivalent to Fourier transform of the image I(l, m). This relation is known as van Cittert-Zernike theorem (Taylor, Carilli, and Perley, 1999). Consequently, the samples of $V(u, v)$ are sufficient information for sky image recovery where each baseline gives an approximate sample from Fourier transform of the sky image. This can be mathematically stated as follows.

$$V_{i,k} \sim V(u_{i,k}, v_{i,k}) = \iint_{K \subset \mathbb{R}^2} I(l, m) \exp\left(-j2\pi(u_{i,k}l + v_{i,k}m)\right) dldm. \tag{2.44}$$

Given the visibility equation, we then form a grid of sky map by $\mathbf{I}_{l,m} \in \mathbb{R}^{r\times r}$ for $l, m \in \{1, 2, ..., r\}$, where r is the resolution of the map. Let $\mathbf{p}_{i,k} \in \mathbb{R}^2$ denote the two-dimensional distance between i'th and k'th antenna, and $\mathbf{r}_{l,m} \in \mathbb{R}^2$ stand for two-dimensional position of pixel in l'th row and m'th column of $\mathbf{I}$, respectively.

Using Equation 2.44, we can approximate the noisy visibilities by

$$\mathbf{V}_{i,k} = \sum_{l,m} \mathbf{I}_{l,m} \exp\left(-j2\pi f_0 \langle \mathbf{p}_{i,k}, \mathbf{r}_{l,m}\rangle\right) + \delta_{i,k} e_i. \tag{2.45}$$

where e_i denotes the noise in antenna i.

Definition 2. Let $\mathbf{A} \in \mathbb{K}^{M\times N}$ be a matrix, with field $\mathbb{K}$. The $vec(\cdot)$ operator is defined

$$vec(\cdot) : \mathbb{K}^{M\times N} \to \mathbb{K}^{MN}.$$

Using above definition we can reformulate Equation 2.45 as follows.

$$\mathbf{y} = \mathbf{\Phi}\mathbf{x} + \mathbf{e} \tag{2.46}$$

with $\mathbf{y}, \mathbf{e} \in \mathbb{C}^M$, $\mathbf{\Phi} \in \mathbb{C}^{M\times N}$ and $\mathbf{x} \in \mathbb{R}^N$ such that $\mathbf{y} = vec(\mathbf{V})$ and $\mathbf{x} = vec(\mathbf{I})$. Finally

$$\mathbf{\Phi}_{z,w} = e^{-j2\pi f_0 \langle \mathbf{p}_{i,k}, \mathbf{r}_{l,m} \rangle} \tag{2.47}$$

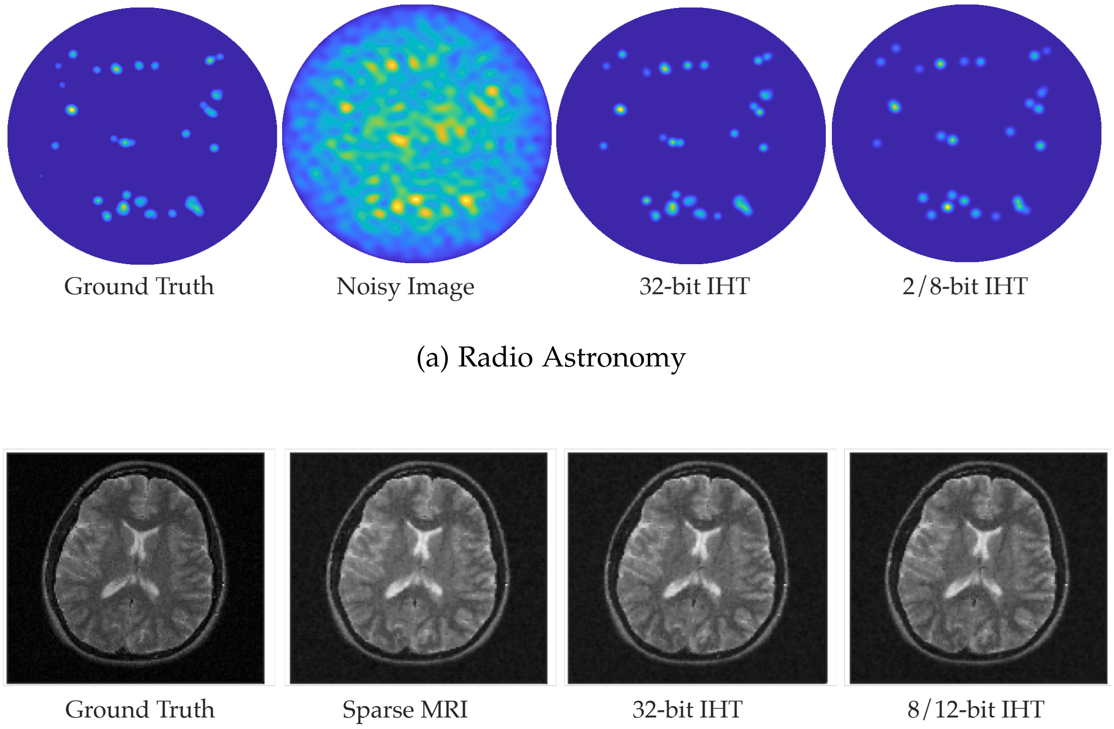
where $z = i + L(k-1), \quad i,k = \{1,2,...,L\}$ and $w = l + r(m-1), \quad l,m = \{1,2,...,r\}$.

EXPERIMENTS We recover a sky image with a resolution of 256×256 pixels ($\mathbf{x} \in \mathbb{R}^{65{,}536}$ in vectorized form) by employing 30 low-band antennas of the LOFAR CS302 station that operate in the 15–80 MHz frequency band and in a Field of View (FoV) of 2 degrees where the sky is populated with 30 strong sources, that is, $\mathbf{y} \in \mathbb{C}^{900}$, $\hat{\mathbf{\Phi}} \in \mathbb{C}^{900\times 65{,}536}$. We note here that 30 antennas lead to a visibility matrix of size 30×30, i.e., the measurement vector is of size 900. The Signal-to-Noise-Ratio (SNR) is assumed to be 5 dB at the antenna level, i.e., $10\log_{10}(\|\mathbf{\Phi x}\|^2/\|\mathbf{e}\|^2) = 5$ dB.

Figure 2.5(a) provides an example of sky recoveries: (a) ground truth estimated over 12 hours of observation, (b) a least square estimate of underlying sky (or dirty image in the nomenclature of radio astronomy), (c) 32 bit and (d) 2/8 bit QIHT which uses 2 bit for the measurement matrix and 8 bit for the observation. This experiment indicates that QIHT captures the sky sources successfully even when only 2 bits are used to compress $\mathbf{\Phi}$. Thus, we can drastically reduce the data precision without significantly degrading the sky image quality.

This strong empirical performance is not completely surprising. Mathematically, the measurement matrix we formed here reflects the phase relations induced by the antenna locations. That is, each time $r_{m,n}$ or $c_{m,n}$ flips its sign where $\hat{\mathbf{\Phi}}_{m,n} = r_{m,n} + jc_{m,n}$, $m = 1,2,...,M$ and $n = 1,2,...,N$, the change in horizontal and vertical directions on the ground enables preserving the phase information required for interferometric imaging even at very low precision.

We evaluate QIHT through (1) the recovery error, and (2) the support recovery. In radio astronomy, it is customary to use a number of true celestial sources resolved in the recovered image as a performance metric, i.e., true-positive findings. That is, the performance of the algorithms is no longer



(a) Radio Astronomy

(b) Magnetic Resonance Imaging

Figure 2.5: Illustration of the main results for the radio astronomy and the magnetic resonance imaging applications. When representing all input data with low precision, IHT achieves a negligible loss of recovery quality on the data recorded by (a) LOFAR station CS302 with 2 bit measurement matrix and 8-bit observation, (b) subsampling k-space measurements (the 2D Fourier transform) of Magnetic Resonance Imaging (MRI) images with 8 bit measurement matrix and 12 bit observation.

described by its ability to recover support entirely but the sky objects, which possess higher error tolerance.

Next, in order to bridge the gap between theory and practice, we revisit the theoretical guarantees and nonsymmetric RIP condition. By definition, the entries of $\mathbf{\Phi}$ have unity magnitude. Moreover, when $\mathbf{\Phi}$ is formed by using closely located antennas, the phase difference between the entries is even smaller, which suggests that, compressive sensing problem in radio inteferometry has the least desirable measurement matrix structure regarding the performance of ℓ_0-minimization methods on such matrices. To understand if the nonsymmetric RIP, hence the theoretical guarantees, holds for such

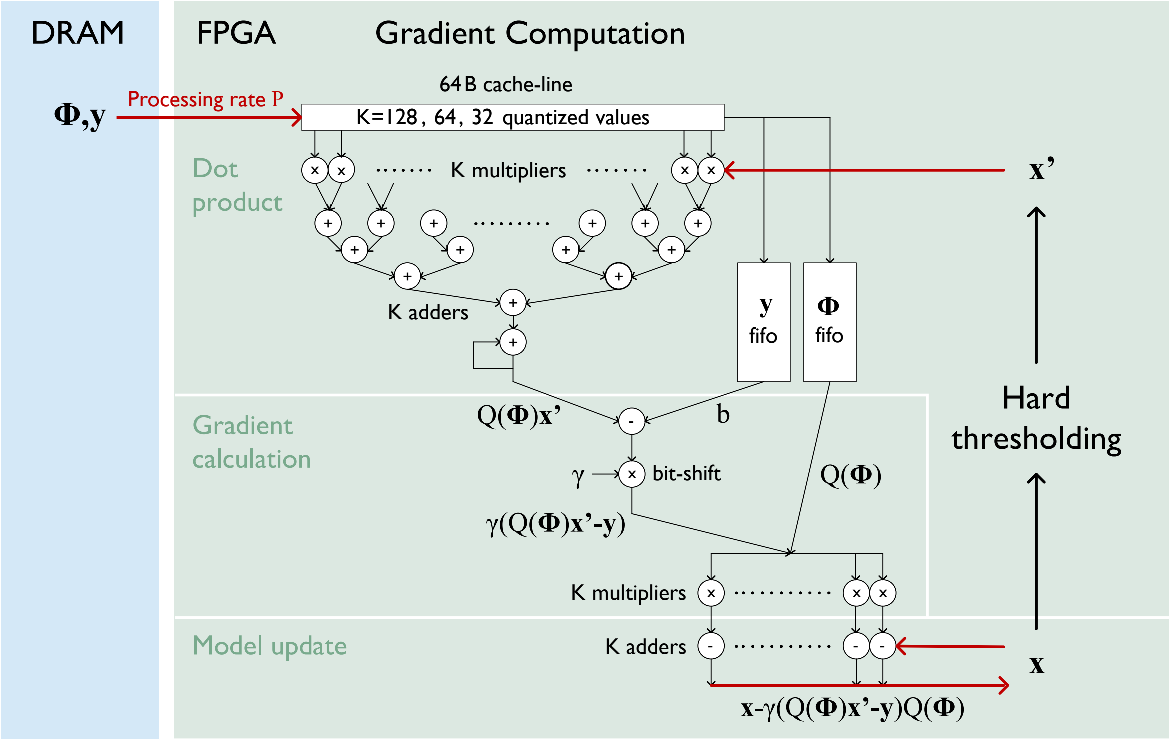


Figure 2.6: Implementation of QIHT on an FPGA-based system (Image credit: Kaan Kara).

a measurement matrix in practice, we present a related note in the next paragraph.

A NOTE ON THE NONSYMMETRIC RIP Convergence is straightforward in low precision setting provided that $\hat{\mu}^{[n]}$ is chosen adaptively as indicated by the update rules of low precision IHT and the normalized IHT (Blumensath and Davies, 2010). Regarding theoretical performance guarantees, however, the RIP condition is to be satisfied. Real-life problems usually do not satisfy the traditional RIP condition $\|\boldsymbol{\Phi}\| < 1$. The scale-invariant feature of the measurement matrix used in normalized IHT however alleviates the RIP issue and imposes a fairly mild constraint, that is, the non-symmetric RIP. In a series of papers (Blumensath and Davies, 2010; Blumensath et al., 2012), CoSaMP is shown to perform markedly worse when the RIP condition fails. The normalized IHT, however, still preserves its near-optimal recoveries far beyond the region of RIP. Motivated by this, we apply normalized IHT and QIHT to real radio telescope data.

Recall from Theorem 4 the two conditions ensuring performance guarantees: (a) $\hat{\beta}_{2s}$ must be large to minimize the quantization error ε_q, and (b) $\gamma_{2s}, \hat{\gamma}_{2s} \leq 1/24$. The former, (a), can be achieved via rescaling of $\mathbf{\Phi}$, whereas for (b), this approach helps not as α_{2s} and $\hat{\alpha}_{2s}$ will be scaled accordingly by the same weight. Therefore we need more sophisticated strategies to achieve this. Fortunately, in the radio astronomy application, we hold a control over γ_{2s} and $\hat{\gamma}_{2s}$ via a set of preprocessing applied on $\mathbf{\Phi}$, such as changing the FoV and the set of antennas that are used. We start by forming a grid on $[l, m], \quad l, m \in [-d, d]$ where the sky map is displayed. Given the antenna locations and d, we can exactly compute $\mathbf{\Phi}$. By changing d, γ_{2s} and $\hat{\gamma}_{2s}$ can be tuned such that nonsymmetric RIP bounds hold. Yet, the cost of this operation lies in the meaning of d, that is, d limits the FoV the antennas observe. For instance, if we drastically increase it to extend FoV without changing the resolution which makes the RIP condition more likely to hold at the same time, we may no longer observe a sky source in the outer field. Hence, we must tune d with caution with an objective to set it according to the quality of image. To overcome the limitation on this, we can benefit from the flexibility we have on the number and location of antennas, and form $\mathbf{\Phi}$ in a way that the nonsymmetric RIP condition holds. That is, the sparsity ratio s/M decreases with number of antennas. Without loss of generality, a smaller ratio indicates that the RIP condition is more likely hold. This can be exploited to satisfy the RIP condition upon setting d. We remark that d In our experiments, we use 30 low-band antennas and set $d = 1$ which falls into the practical range we observed. For $d = 1$, we note an upper bound on γ_{2s} to be 0.0396 that is $< 1/24$. Moreover, using Lemma 4, we find that b can be as low as 2 bits by still satisfying the nonsymmetric RIP property. In an ablation study where we set $d \in [0.2, 10]$, we observe that an upper bound γ_{2s} is in $[0.027, 0.091]$, which indicates that the regime of γ_{2s} are small in quantity and hence the RIP condition is likely to be satisfied.

2.6.2.2 *Magnetic Resonance Imaging*

Compressive sensing enables faster MRI by acquiring less data through undersampling in the measurement space, hence accelerating the scan time.

While *Nyquist* criteria are violated due to the undersampling, the image is still reconstructed with little or no perceptible loss of visual information, established by a substantial body of work, for example (Lustig, Donoho, and Pauly, 2007, 2008). The key ingredient behind this success is that magnetic resonance images exhibit a sparse representation in a known and fixed mathematical transform domain, i.e., the wavelet transform domain. A standard strategy is, therefore, to decode the sparse coefficients based on the undersampled measurements and store them for later encoding and reconstruction of the image.

In MRI, the measurements are two-dimensional Fourier coefficients of the image, the so-called k-space samples. Inverse Fourier reconstruction of the image from the undersampled k-space data, however, is known to produce aliasing artifacts. In order to mitigate undersampling artifacts, the compressive sensing algorithm iteratively finds an estimate of sparse coefficients. In our notation, $\mathbf{\Phi}$ is formed by Fourier and inverse wavelet transforms and sampling operator, $\mathbf{x}$ has one-dimensional sparse coefficients, and finally, $\mathbf{y}$ is a vector of undersampled k-space data.

The performance of normalized IHT on the *Shepp-Logan phantom* was previously studied in (Blumensath and Davies, 2010). Instead, we tested QIHT on a representative brain image[1] of size 512×512 in pixels and compare our results to the reconstructed image through ℓ_1-minimization using the SparseMRI software[4]. We subsample k-space data by a factor of 3 using a radial sampling mask.

The brain image reconstructed by various algorithms depicted in Figure 2.5(b) reveals that QIHT still yields a similarly good performance as the normalized IHT and ℓ_1-minimization when the bit-widths of the k-space data and the transformation matrices are lowered down to 8 and 12 bit, respectively.

While offering accelerated image recovery for MRI, low precision data representation can potentially reduce the storage required to keep patients raw data as discussed in (Langer, 2011; Poldrack, Mumford, and Nichols, 2011).

[4] Available on http://people.eecs.berkeley.edu/~mlustig/Software

2.6.3 *Implementation and Performance*

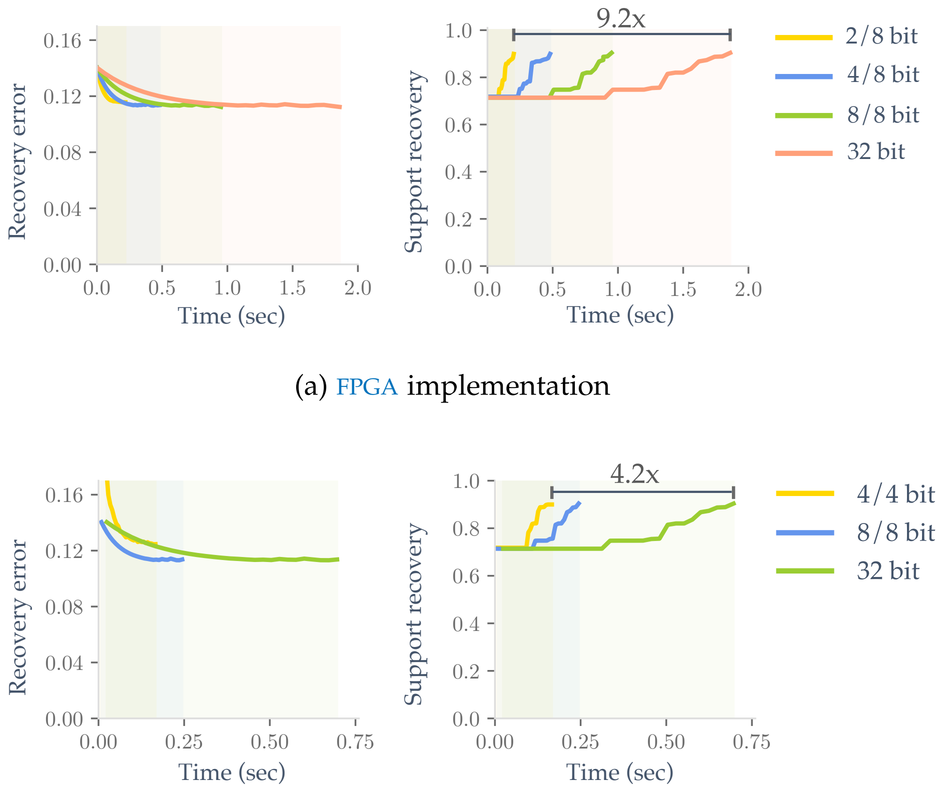
We demonstrate the speed-up obtained by performing QIHT in the previous two applications on both FPGA and CPU when reducing the number of bits used for the data representation. The tailoring of hardware to accommodate the QIHT algorithm is not the contribution of the thesis author, but achievements of the co-authors of (Nezihe Merve Gürel et al., 2020), namely, Kaan Kara (on FPGA), Alen Stojanov and Tyler Smith (on CPU). We present the necessary details taken directly from (Nezihe Merve Gürel et al., 2020) below for the sake of completeness for the reader.

2.6.3.1 *FPGA implementation*

FPGA are an alternative to commonly used GPU for accelerated processing of compute-intensive signal processing workloads. The reconfigurable logic fabric of an FPGA enables the design of custom compute units, that can be advantageous when working on low-precision and uncommon numeric formats, such as 2 bit numbers. Thanks to this microarchitectural flexibility, it is possible to achieve near linear speed-up when lowering the precision of data that is read from memory. This has been shown recently for Stochastic Gradient Descent (SGD) when training linear models (Kara et al., 2017; Zhang et al., 2017). In this work, we use the open-source FPGA implementation[5] from the above mentioned works and modify it to perform QIHT.

In terms of the computation, we modify two parts of the design to convert it from performing SGD to IHT. First, instead of updating the model after a mini-batch count is reached, we update it after all samples are processed and the true gradient is available. Second, after each epoch, we perform a binary search on the updated model to find the threshold value satisfying that only top s values are larger than the threshold. The rest of the design stays the same, including the fixed-point computation, utilized to minimize the usage of available FPGA resources.

[5] https://github.com/fpgasystems/ZipML-PYNQ

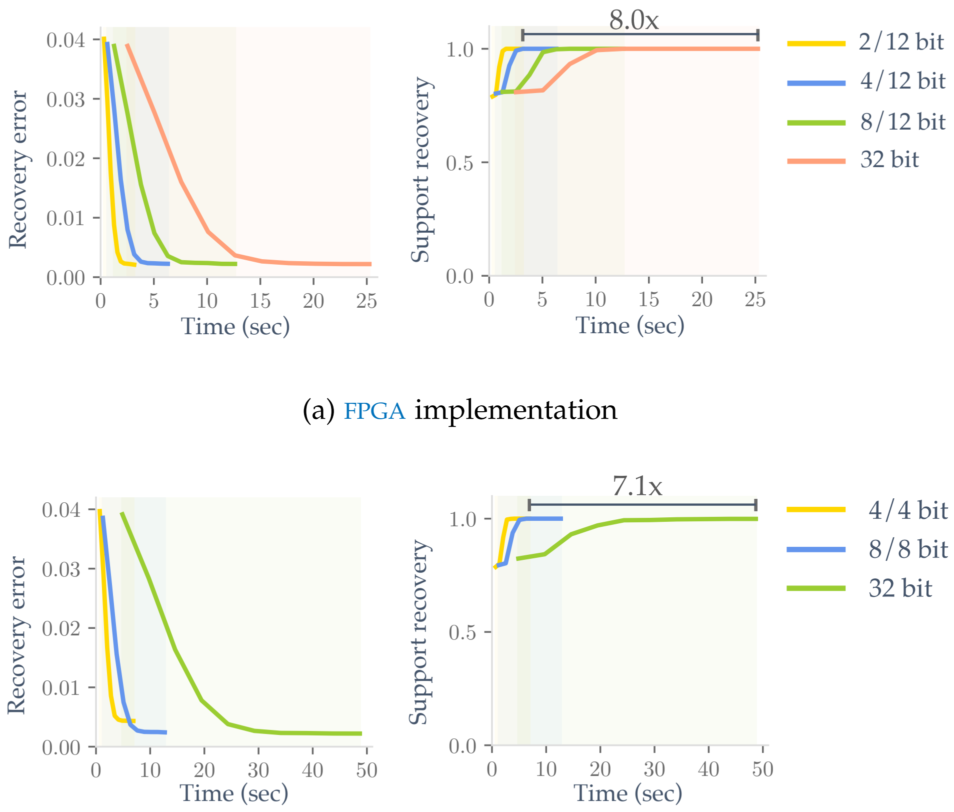


(a) FPGA implementation

(b) CPU implementation

Figure 2.7: Speed-up on image recovery enabled by QIHT on (a) FPGA and (b) CPU on real radio telescope data.

FPGA PERFORMANCE ANALYSIS The gradient computation unit in Figure 2.6 reads the measurement matrix $\mathbf{\Phi}$ and the measurements $\mathbf{y}$ from the main memory and keeps $\mathbf{x}$ in on-chip memory. We note that transferring $\mathbf{\Phi}$ from main memory will be necessary in most practical settings, where the matrix $\mathbf{\Phi}$ is too large to fit onto the FPGA. The FPGA is able to consume and process the data from the memory at a rate of $P = 12.8$ GB/s. Thus, the performance is bounded by P for processing $\mathbf{\Phi}$ and $\mathbf{y}$. The time for each iteration is approximately $T = \text{size}(\Phi)/P$, since $size(y) \ll size(\Phi)$. Theoretically, we can achieve a significant speed-up by using a quantized $\mathbf{\Phi}$, simply because we reduce the amount of data to be consumed by the FPGA: more entries arrive with each transfer from the main memory. The essential idea behind achieving linear speed-up is lowering the precision of $\mathbf{\Phi}$ while keeping P constant. This is possible, because we can adapt the gradient computation



(a) FPGA implementation

(b) CPU implementation

Figure 2.8: Speed-up on image recovery enabled by QIHT on (a) FPGA and (b) CPU on Magnetic Resonance Imaging (MRI) data.

unit's microarchitecture and increase its internal parallelism to handle more values per incoming line, thanks to the FPGA's architectural flexibility.

COMPUTING **Φ** ON THE FLY The above analysis focuses on the case where **Φ** is stored in main memory, in which case quantization helps to reduce the amount of data transferred between the main memory and FPGA. In some applications, **Φ** can be calculated on the fly, inside the FPGA. Also in this case, quantization can help in achieving better performance. The reason is that quantizing Φ also saves other crucial resources (e.g., multipliers) that are limited on an FPGA. These resource savings, in turn, enable higher internal parallelism, for instance, to speed up the computation of Φx. For example, it has been shown that to increase the dot-product parallelism from 64 to 128 while maintaining the rate of operations per cycle, it is necessary to lower the precision of one side of the dot product to 2-bits; otherwise, the resource consumption is too high to fit the design to one FPGA (Kara et al., 2017).

The performance of the FPGA-based implementation is presented in Figure 2.7(a) and Figure 2.7(a). For the time spent per iteration, we see that quantization, and the resulting compression of the measurement matrix $\mathbf{\Phi}$ leads to a near linear speed-up for recovering the support vector. All variants (full precision to lowest precision) of the normalized IHT on FPGA can consume $\mathbf{\Phi}$ at the same rate, and therefore the runtime of QIHT depends linearly on the size of $\mathbf{\Phi}$, yielding the linear speed-ups that we observe in the experiments. In terms of end-to-end performance, we measure the time needed for each precision level to reach support recovery ratio 90% and calculate the speed-up. The 2/8 bit QIHT reaches the same support recovery ratio 9.2× faster.

2.6.3.2 *CPU implementation*

On a CPU, it is possible to achieve near-linear speedup when reducing the size of the data representation despite lacking the necessary instructions to compute with 4, 8, or 16-bit integer operands. This has been previously demonstrated for both Gradient Descent (GD) and IHT (Stojanov et al., 2018).

In order to perform the low-precision computations on a CPU without instructions supporting low-precision arithmetic, low-precision data is first converted to 32-bit floating point. Thus the instructions used to for low-precision arithmetic are actually less efficient than using single-precision 32-bit arithmetic. The advantage on a CPU is that the low-precision representation results in less data movement.

To perform CPU experiments, we build on the implementation Clover from (Stojanov et al., 2018). The main extension needed was support for complex arithmetic. The bulk of the computation for both GD and IHT is dominated by two matrix-vector multiplication operations. The first is a dense matrix times a sparse vector, and the second a dense matrix times a dense vector. The former is implemented as a loop around a dot-product operation for 4 and 8bit and uses the BLAS *gemv* routine for 32 bit. The latter is implemented as a loop around a dense scale and add operation for all three supported datatypes.

Our CPU implementation uses handwritten code in AVX2 intrinsics and supports 4-bit, 8-bit, and 32-bit precisions. We use OpenMP to parallelize our implementation, XORShift to generate random numbers for stochastic rounding, and the Intel math kernel library (MKL) for the 32 bit matrix-vector multiplication. We used two different systems for our experiments. The radio astronomy experiment was run on an Intel Xeon CPU E3-1285L v3 3.10GHz, with 32GB of RAM and 25.6 GB/s bandwidth to main memory, running Debian GNU/Linux 8 (jessie), kernel 3.16.43-2+deb8u3. The MRI experiment was run on an Intel Xeon E5-2690 v4 CPU with 512 GB of RAM, 153.6 GB/s bandwidth to main memory, running Ubuntu 16.04.6 LTS with kernel version 4.4.0-148-generic. We use the Intel icc compiler 17.0.0, Intel IPP 2017.0.0 (r52494), and Intel MKL 2017.0.0 (Build 20160801). The RDTSC instruction is used to measure the cycle count for each iteration, and we report the median. Turbo Boost and Hyper-threading were disabled to avoid the effects of frequency scaling and resource sharing on the measurements.

We show performance plots for CPU speed-up in Figure 2.7(b) and Figure 2.7(b). On both data sets, we obtain up to a 2.84× speed-up for the the 8-bit implementation, and 7.1× for the 4 bit implementation, with similar recovery properties as for FPGA.

2.7 SUMMARY

In this part, we focused on the hardware efficiency at the training stage. By reducing *precision* of data, we showed that training of compute-intensive compressive sensing applications can be significantly accelerated on hardware with no visible loss of image quality. We investigated low precision training for sparse signal recovery problems with particular focus on the case in which both the observation vector and the measurement matrix are quantized. As the main contribution, we introduced a low-precision normalized IHT variant for stochastically quantized data, QIHT. We derived theoretical guarantees and demonstrated its practical performance both in terms of accuracy and recovery time in two application areas, radio astronomy and MRI, on both CPU and FPGA.

In what follows in Part ii, we transition our focus to the post-training stage, and study manual labor demanding task of model selection, where we will perform selective sampling to reduce the *amount* of data to label.

Part II

LABEL EFFICIENCY VIA DATA SAMPLING

3 LABEL-EFFICIENT MODEL SELECTION

All models are wrong, but some are useful.

— George E. P. Box (1976)

3.1 OVERVIEW

Curation of cost-inefficiency through directly reducing the amount of data to be processed, as natural as it sounds, is highly challenging to automate in dynamic and uncertain physical world environments. In Part i, we studied this problem for hardware efficiency by reducing the data size via arithmetic-level manipulations. In this part, we focus on an another cost demanding operation across the ML pipeline: manual labeling of data for supervised learning problems. In particular, we address this for model selection of pretrained models at the deployment phase, at which the selected model will be used to make predictions on freshly arriving production data whose distribution has potentialy shifted away from that of training data.

Depending on the data availability, one can consider two settings: (i) the *pool-based* setting assumes that the learner has access to a pool of unlabeled data, and she can select informative data samples from the pool to achieve her task, and (ii) the *stream-based* setting assumes the data is arriving one example at a time (that is, in a stream), and the learner randomly decides to query the label of the sample on the go or to just throw it away. While offering fewer options on which data to label, this setting alleviates the scalability challenge of storing and processing a large pool of examples in the pool-based setting. Motivated by this, we focus on a setting where a stream of unlabelled data examples arrive *sequentially* from a data source, and query decisions are made per each example.

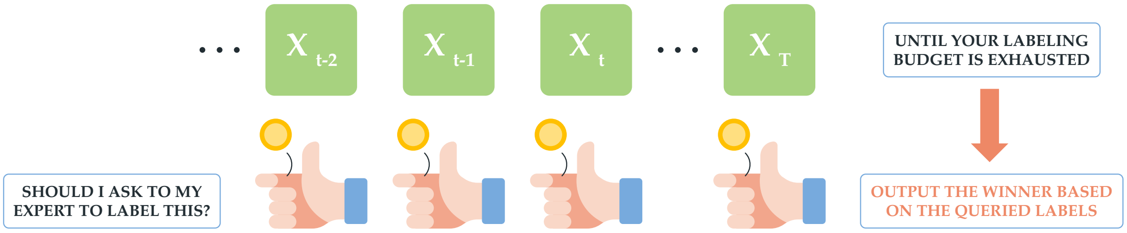

X_{t-2}
X_{t-1}
X_t
X_T
UNTIL YOUR LABELING BUDGET IS EXHAUSTED
SHOULD I ASK TO MY EXPERT TO LABEL THIS?
OUTPUT THE WINNER BASED ON THE QUERIED LABELS

Figure 3.1: Overview of the active model selection process in the stream-based setting. For each incoming instance, a coin is tossed to decide whether to query the label of that instance or not. Upon exhausting the labeling budget, the winner model is returned based on previously queried labels (evidence).

We impose *no statistical assumptions* on the stream — known as the *adversarial* setting, where an adversary chooses the order of the data points. This enables us to account for a large coverage of scenarios where the distribution of production data shifts during the model selection process, in which i.i.d. assumption is violated.

In the first section of this part, Section 3.2, we focus on active model selection of pretrained models. We first review and adapt existing active learning strategies for model selection, and then develop a novel, principled and efficient model selection approach for the stream setting: Model Picker. Model Picker sequentially receives unlabelled instances and returns the best model at any time by requesting only a small portion of labels, therefore reducing the labelling cost. Our query strategy is randomized and leverages hypothetical query answers to decide which data examples are likely to be informative for identifying the best model with no regret for adversarial streams. To illustrate its benefit, we conduct extensive experiments on a host of pretrained model collections. Our experimental study establishes Model Picker as the state-of-the-art for this problem.

In the second section of this part, Section 3.3, we transition our focus from the selection of pretrained models to building a theoretical understanding on GCN models with existence of graph decomposition. In particular, we take the first step towards the theoretical analysis on the impact of graph decomposition in learning with graphs. We take an information theoretical view and analyze the infinite-sample behaviour of Shannon's *mutual information* between the input and output layers. We show that, under certain conditions (on the singular values of the graph), mutual information between the input and output layers of GCN converges to 0 exponentially fast with respect to the depth, corresponding to the oversmoothing problem of GCN that is encountered in practice. On the other hand, for GraphCNN (Such et al., 2017), a variant of GCN with graph decomposition, the oversmoothing occurs in a much smaller regime of parameters. In the respective section, we will present our theoretical analysis and illustrate that such a study that focuses on understanding the role of decomposition can further enable novel graph decomposition algorithms.

3.2 ACTIVE MODEL SELECTION

3.2.1 *Related Work*

Our approach relates to several bodies of literature. For each related area, we reference similar works that match the objective of our work.

ACTIVE MODEL SELECTION (Madani, Lizotte, and Greiner, 2012) develop their method for the online setting. They seek to identify the best model via probing models, one at a time, with i.i.d. samples, while having a fixed budget for the number of probes. In contrast, our approach applies even to adversarial streams and allows one to make predictions online, while minimizing the number of queries made. Most of other previous works (Ali, Caruana, and Kapoor, 2014; Gardner et al., 2015; Sawade, Landwehr, and Scheffer, 2012) and (Katariya, Iyer, and Sarawagi, 2012; Sawade et al., 2010) and (Kumar and Raj, 2018; Leite and Brazdil, 2010) focus on pool-based sampling of informative instances, where the learner ranks the entire pool of unlabeled data and greedily selects the most informative examples. This setting substantially differs from the streaming setting, and we focus on the latter for reasons of scalability and applicability to many real-world situations.

ACTIVE LEARNING Active learning aims to query the label of those instances that help improving the *training* of classifiers, rather than selecting among pre-trained models. Here we review those methods that can potentially be adapted for model selection. The celebrated query-by-committee (QBC) paradigm (Seung, Opper, and Sompolinsky, 1992) forms a committee of classifiers to vote on the labeling of incoming examples. The query decision is made based on the degree of disagreement among the committee members. The general strategy is to query those instances that help the learner prune the committee and only keep those classifiers with higher accuracies. There are other QBC approaches in active learning, such as (Abe and Mamitsuka, 1998; Cohn, Atlas, and Ladner, 1994; McCallum and Nigam, 1998; Melville and Mooney, 2004a; Settles and Craven, 2008a; Zhu et al., 2007). One limitation of these algorithms is that they often focus on pool-based sampling, which limits their scalability. Several other approaches consider active learning in the streaming setting. The seminal works of (Dasgupta, Hsu, and Monteleoni, 2008) and (Balcan, Beygelzimer, and Langford, 2009), followed by (Beygelzimer et al., 2010; Zhang and Chaudhuri, 2014), use disagreement-based strategies. The idea of using importance weights in active learning is studied by a series of works including (Alina, Sanjoy, and Langford, 2008; Sugiyama, 2006) and (Alina et al., 2011; Bach, 2007), where importance weights are introduced to correct sampling bias and provide statistically consistent convergence to the optimal classifier in the PAC learning setting. All of the above approaches on stream-based active learning focus on i.i.d. streams and try to improve the supervised training of classifiers, whereas our approach applies to the more general adversarial streams and performs no training.

ONLINE LEARNING AND BANDITS In the context of online learning, the closest strategy to ours is label-efficient prediction (Cesa-Bianchi, Lugosi, and Stoltz, 2005), where they query the label with a *fixed* probability at each round, which is simply the passive learning scenario. However, we use the disagreement among the models predictions to *adapt the probability* of querying to the instance at hand. Another line of study that is similar to ours is consistent online learning (Altschuler and Talwar, 2018; Karimi et al., 2019), where the learner observes the loss at every round and hopes to optimize the number of action switches. In a setting like ours, where the goal is to minimize the number of queries, we do not request labels in most rounds; hence cannot compute loss at every round. Finally, in the context of multi-armed bandit, our framework is related to the EXP3 algorithm (Auer, Cesa-Bianchi, and Fischer, 2002) for adversarial bandits. While the EXP3 algorithm use the probability of selecting an arm for making unbiased estimators, we use the probability of observing the *whole* loss.

3.2.2 *Problem Definition*

We consider a setting where we have access to

1. k pretrained models that are ready to be deployed,
2. a set of unlabeled instances that are being freshly collected in a streaming fashion,
3. a labeling budget b.

Our goal is to develop an active learner that identifies the pretrained model with the highest accuracy[1] on this freshly collected unlabeled data by querying no more than b labels. In a way, we hope to find the pretrained model that generalizes best on the target distribution at hand by querying the least number of labels possible.

We define the classification problem on an input space $\mathcal{X}$ and output space $\mathcal{Y} \sim \{1, 2, \ldots, C\}$. We refer to x_i as *instances* and y_i as *true labels*. As the instances are collected in a streaming fashion, they arrive one at a time, and we call each time stamp an instance arrive as *round* denoted by t. At each round t, the learner makes a query decision based on how informative it finds x_t to be. This can be simply considered as an adaptive coin flipping strategy. That is, at every round, the bias is computed in an adaptive fashion taking the previously queried labels and the disagreement x_t creates among the pretrained models into account towards striking a balance of exploitation and exploration. Upon querying a total of b labels, the active learner returns the pretrained model it believes to be the best. An overview of the process is shown in Figure 3.1.

[1] The pretrained model with highest accuracy on the entire stream if all labels were known.

We moreover list the notation for this section as follows.

k	number of pretrained models
C	number of classes
b	labeling budget
t	stream index
$x_t \in \mathcal{X}$	streaming instance at round t
$y_t \in \mathcal{Y}$	true label of x_t
z_t	probability of querying the label of x_t
$Z_t \in \{0,1\}$	the query decision at round t
$p_{t,i}$	predicted label of x_t by model $i \in [k]$ and at round t
$\mathbf{p}_t$	$\mathbf{p}_t = p_{t,i=[k]}$: predicted label of x_t by all models at round t
$\ell_{t,i}$	$\ell_{t,i} = \mathbb{1}\{p_{t,i} \neq y_t\}$: Loss of model i at round t
$\ell_{t,i}^{(c)}$	$\ell_{t,i}^{(c)} = \mathbb{1}\{p_{t,i} \neq c\}$: Loss of model i at round t if $y_t = c$ for $c \in \mathcal{Y}$
$\hat{\ell}_{t,i}$	$\hat{\ell}_{t,i} = \mathbb{1}\{p_{t,i} \neq y_t\} Z_t / z_t$: Loss estimate of model i at round t
$L_{t,i}$	$L_{t,i} = \sum_{s \in [t]} \ell_{s,i}$: Accumulated loss of model i at round t
$\hat{L}_{t,i}$	$\hat{L}_{t,i} = \sum_{s \in [t]} \hat{\ell}_{s,i} Z_s / z_s$: Accumulated loss of model i at round t
$\pi_{t,i}$	posterior probability of model i at round t
$\boldsymbol{\pi}_t$	$\boldsymbol{\pi}_t = \pi_{t,i=[k]}$: posterior probability at round t

3.2.3 *Adapting Existing Strategies for Model Selection*

We adapt existing selective sampling strategies such as query-by-committee (Ido and Sean, 1995; Tosh and Dasgupta, 2018), random sampling, and importance weighted active learning (Alina et al., 2011; Alina, Sanjoy, and Langford, 2008) for model selection. To select informative instances in a streaming setting, these methods follow a coin flipping strategy similar to online learning: upon seeing an instance x_t, a coin is flipped with a bias z_t, and the label y_t is requested if and only if the coin comes up heads ($Z_t = 1$).

- *Passive Learning (Label Efficient Prediction/Random Sampling)*: We begin with adapting the passive learning baseline (label-efficient prediction or random sampling) where the sampling decision is made randomly with a fixed sampling probability $z_t = \varepsilon$. We restrict our interest merely to the instances in which at least two models disagree, as otherwise instances are non-informative in the ranking of models. Given a labeling budget b that is less than and equal to $\sum_{t\in[T]} \mathbb{1}\{\exists i, j \in [k] \text{ s.t. } p_{t,i} \neq p_{t,j}\}$, one can set the query probability to

$$z_t = b / \sum_{t\in[T]} \mathbb{1}\{\exists i, j \in [k] \text{ s.t. } p_{t,i} \neq p_{t,j}\} \tag{3.1}$$

where T is the size of streaming instances, and $z_t = 0$ for the rounds that all models agree. This query probability ensures that $z - t \leq 1$ and the practitioner queries no more than b instances in expectation. Although some reader might think this approach is trivial, it has shown to be very powerful in the existence of noise (Settles and Craven, 2008b). We will illustrate the effectiveness of this generic baseline in Section 3.2.5.

- *Query by Committee*: Next, we adapt the query-by-committee paradigm proposed in (Ido and Sean, 1995) for model selection in the streaming setting. Generally speaking, this strategy aims to measure the information of instances by forming a committee of classifiers and identifying which instances creates the largest disagreement among the committee members. It consists of two sub-strategies ensemble learning and determining a maximal disagreement measure. The ensemble learning indicates how the committee is formed from the candidate classifiers. This step is crucial to make the disagreement measure more reliable, hence the candidate classifiers must have high accuracy. In literature, there exist many ensemble learning methods including (Abe and Mamitsuka, 1998; Breiman, 1996; Freund and Schapire, 1995; Melville and Mooney, 2004b). Most, if not all, of these methods are either designed for pool-based sampling or for cases where observed data is stored. Bagging predictors (Breiman, 1996) proposes to improve performance of a single predictor by forming a committee from multiple versions of it, where the versions are trained on the bootstrap replicates of training data. This is followed by (Abe and Mamitsuka, 1998) where diverse ensembles are generated using bagging and boosting techniques (Freund and Schapire, 1995). These strategies focus on a setting where the observed data is stored as opposed to our setting. Another popular ensemble learning algorithm, active-decorate relies on the existence of artificial training data to form a diverse set of examples. In our setting, however, we assume neither storing of previously seen data nor availability of artificial data. In the online setting, luckily, one could benefit from the strategy introduced in (Freund and Schapire, 1995). Upon seeing the label y_t, the authors propose to update the belief on the models using the observed loss and a hyperparameter β such that $\boldsymbol{\pi}_t \propto \boldsymbol{\pi}_{t-1}\beta^{\ell_t}$. We note that, this update rule very closely resembles that of the structural query by committee, which we will review next. In fact, it is identical when both of β are tuned to query budget b amount of label in average over many realizations.

As for the disagreement measure, popular choices include vote margin, vote entropy and Kullback-Leibler divergence between the label distributions of each committee member and the consensus in (Settles and Craven, 2008b). We note that the latter two are equivalent for 0-1 loss functions ℓ. The former, vote margin, on the other hand, is measured by the difference between the votes of most voted and second most voted label. We omit this methods in our adaptation based on the preliminary observation on the success of entropy over the vote margin.

For the model selection problem where there are pretrained models and their predictions on the unlabeled instances, one consider every pretrained model as a committee member, or simply adapt an ensemble learning method to shrink it over the streaming rounds t. However, for an arbitrary set of pretrained models with no additional information than their predictions, there is no principle on how to adaptively update information on the models ranking.

We consider the query-by-committee paradigm proposed in (Ido and Sean, 1995) and adapt it to the streaming setting as a disagreement-based selective sampling baseline. We take all pretrained models as committee of models.

Upon seeing each instance x_t, we measure the disagreement between the model predictions $p_{t,i}, i \in [k]$ to compute the sampling probability. In our adaptation, we use *vote entropy* as the disagreement measure. Formally, at each new instance t, we set the sampling probability z_t to the normalized entropy of the committee votes on the labels, that is,

$$z_t = \frac{1}{\max(\log C, \log k)} \sum_{c \in [C]} v_{t,c} \log \frac{1}{v_{t,c}}, \quad \text{where } v_{t,c} := \frac{1}{C} \sum_{i \in [k]} \mathbb{1}\{p_{t,i} = c\}. \tag{3.2}$$

In order to limit the number of the queried instances to at most b, we introduce a hyperparameter β to scale sampling probability uniformly over the entire stream such that $z_t \leftarrow z_t \beta$ where the value of β depends on the labeling budget.

- *Structural Query by Committee*: The (interactive) structural query by committee algorithm (Tosh and Dasgupta, 2018), aims to minimize the interaction with the oracle while learning. The algorithm is built upon the query-by-committee principle, and its sampling probability is specified via the disagreement between competing structures that are drawn from a posterior distribution π_t. After each new query, the posterior is updated as $\pi_t \propto \pi_{t-1} \exp(-\beta \ell_t)$, where β is a fixed constant.

  In our adaptation, at each time instance t, we draw two models i and j from π_t and set the query probability to be the fraction of disagreement between i and j up to round t, that is $z_t = \frac{1}{t} \sum_{s \le t} \mathbb{1}\{p_{s,i} \neq p_{s,j}\}$. In order to meet the constraint on the labeling budget, we consider the exponent β as hyperparameter that can be tuned accordingly. We note that the strategy introduced in (Freund and Schapire, 1995) very closely resembles the update rule of the structural query by committee. In fact, it is identical when both of β are tuned to query budget b amount of label in average over many realizations.

- *Importance Weighted Active Learning*: Importance weighted approaches including (Alina, Sanjoy, and Langford, 2008; Beygelzimer et al., 2010) and (Alina et al., 2011) have been shown to achieve substantial improvement in label complexity when applied to supervised learning problems for binary classification.

Formally, given an incoming unlabeled instance x_t, these approaches compute a query probability z_t based on the maximal disagreement x_t creates among the models in hypothesis class at time t: $\mathcal{H}_t$, and the query probability is given by $z_t = \max_{i,j \in \mathcal{H}_t, c \in [C]} \ell_{t,i}^{(c)} - \ell_{t,j}^{(c)}$. In our setting, we assume access only to hard predictions, hence use 0-1 loss for ℓ. Our query probability indicates that the label of x_t will be requested if it is in the region of disagreement of surviving hypothesis at time t.

The update of the hypothesis class at every round is a key step in making the maximal disagreement measure reliable. To realize this, the algorithm computes a rejection threshold θ_t using sample complexity bounds with $\theta_t = \sqrt{(8/t)\ln(2t(t+1)|\mathcal{H}_t|^2/\delta)}$ for some δ, and update the hypothesis space $\mathcal{H}_t$ to contain only the models whose weighted error is θ_t greater than weighted error of the current best model at time t. The weight per loss of each instance is simply set to the reciprocal of the respective query probability such that $\hat{\ell}_t = \frac{Z_t}{z_t} \mathbb{1}\{p_{t,i} \neq y_t\}$ in order to correct for the sampling bias. In our adaptation, we consider the confidence parameter δ and a constant scaling of z_t for all t as hyperparameters, which can be tuned in order to keep the number of queries at the end of stream below the labeling budget.

Among other adaptations of importance sampling variants such as (Alina et al., 2011), we only focus on the superior one in our empirical evaluation here and leave the details of other one to the Appendix A.1.2.

Note on the adaptation: In the context of training, the hypothesis space is being shrunk to contain classifiers with high accuracy despite being trained with limited data. This method in particular relies on this purpose using sample complexity bounds and focusing on identifying the decision boundary. Hence, it is natural for all of the methods listed above to lack translation to the model selection task.

3.2.4 *Model Picker*

We now introduce a stream-based active learning algorithm, Model Picker, whose sole purpose is to query labels to select the best pretrained model among many others. Expectedly, Model Picker follows an adaptive strategy by sequentially querying the labels of those instances it finds informative. The Model Picker framework is a joint work with Mohammad Reza Karimi, Andreas Krause and Ce Zhang. From the research question, problem scoping to the algorithmic details, this framework would not have been possible without any of these contributors.

3.2.4.1 *The Algorithm*

In a nutshell, at each round t, Model Picker computes a query probability z_t based on the model predictions $\boldsymbol{p}_t$ as well as the posterior belief on the models $\boldsymbol{\pi}_t$. Upon computing z_t, it makes a random decision via $Z_t \sim Ber(z_t)$. The label y_t is requested if and only if $Z_t = 1$. The Model Picker algorithm then computes a loss estimate of round t such that $\hat{\ell}_t = \mathbb{1}\{p_{t,i} \neq y_t\} \cdot Z_t / z_t$, and updates its posterior belief $\boldsymbol{\pi}_t$ similarly as in the Exponential Weights algorithm (Littlestone and Warmuth, 1994) using the accumulated loss by each model up to round t: $\hat{L}_t = \sum_{s \leq t} \hat{\ell}_s$ and a decaying learning rate η_t. Further algorithmic details and remarks are listed below:

- For a careful optimization of exploitation-exploration trade-off, the formation of query probability z_t is required to strike a balance between the disagreement x_t creates among the models (exploration) and the evidence collected up to round t (exploitation). As the true label of y_t is unknown before the query decision is made, the active learner has to rely on the hypothetical losses together with the posterior belief. Model Picker therefore, achieves this by computing the maximum possible variance of the hypothetical loss over the posterior distribution: $v(\boldsymbol{p}_t, \boldsymbol{\pi}_t) = \max_{c \in \mathcal{Y}} \mathrm{Var}_{i \sim \pi_t}(\ell^c_{t,i})$. That is,

$$z_t = \begin{cases} \max\{v(\boldsymbol{p}_t, \boldsymbol{\pi}_t), \eta_t\} & \text{if } \; v(\boldsymbol{p}_t, \boldsymbol{\pi}_t) \neq 0 \\ 0 & \text{otherwise.} \end{cases} \tag{3.3}$$

Algorithm 2 Model Picker

```
Set $\hat{L}_{0,i} = 0$ for all $i \in [k]$
for $t = 1, 2, \ldots$ do
    $\eta_t := \sqrt{(\log k)/(2t)}$
    Compute the posterior belief $\mathbf{w}_t$ over models, with $\mathbf{w}_t \propto \exp\{-\eta_t \hat{L}_{t-1,i}\}$
    Collect predictions $\mathbf{p}_t$ of models for the incoming instance $x_t$
    Compute $z_t$ as in Equation 3.3 and sample $Z_t \sim \mathrm{Ber}(z_t)$
    if $Z_t = 1$ then
        Query the label $y_t$
        $\hat{L}_{t,i} = \hat{L}_{t-1,i} + \frac{1}{z_t}\mathbb{1}\{p_{t,i} \neq y_t\}, \forall i \in [k]$
    else
        $\hat{L}_{t,i} = \hat{L}_{t-1,i}, \quad \forall i \in [k]$
    end
end
```

- When there is a disagreement among the models such that $v(\boldsymbol{p}_t, \boldsymbol{\pi}_t) \neq 0$, the query probability z_t is lower bounded so as to prevent unboundedness. Also note that η_t decreases over t.

- The principle behind the choice of z_t to be maximum possible variance is as follows. The amount of regret accumulating due to missing the update $\boldsymbol{\pi}_t$ to $\boldsymbol{\pi}_{t+1}$ (that is, $Z_t = 0$) is shown to be proportional to the variance of $\boldsymbol{\ell}_t^c$. Hence, the choice of z_t in Equation 3.3 can be seen as optimizing for the *worst-case* scenario at each round in the view of regret.

- Using the posterior belief, it is possible to perform label-efficient prediction such that $\hat{y}_t = p_{t,i_t^*}$ where $i_t^* := \arg\max_{i \in [k]} \pi_{t,i}$.

The Model Picker algorithm is depicted in Algorithm 2.

3.2.4.2 *Guarantees*

We briefly visit the theoretical guarantees. The following results hold both for adversarial and stochastic streams. The detailed results can be found in (Mohammad Reza Karimi, Nezihe Merve Gürel[†] et al., 2021). The theoretical guarantees of Model Picker is conducted by the co-first author of the respective paper: Mohammad Reza Karimi; not the contribution of the thesis author. We briefly review them here for completeness, and refer to the original paper for the proof and further details.

identification probability For both stream settings, the probability of misidentifying the *true* best model decreases nearly exponentially fast with $k\exp(-\lambda\mathcal{O}(\sqrt{T\log k}))$. This result is very promising for the goal of model selection and hints on the practical performance of Model Picker in selecting the best model from the model pool.

accuracy gap At times where Model Picker returns a model other than the best one, we are interested in the *quality* of the returned model. Motivated by this need, we introduce a new metric, accuracy gap, that is the expected gap between the accuracy of returned model and that of the best model. Towards establishing the theoretical guarantee on the accuracy gap, we introduce a term λ that indicates the hardness of an instance. λ is simply given by $\lambda = \min_{j\in[k]\{i^*\}} \delta_j^2/\theta_j$ where δ_j quantifies the excessive true loss of jth model over the best model, and θ_j is the probability that jth model and the best model disagrees on an instance. Similarly as in the case of identification probability, Model Picker guarantees to return a model whose accuracy gap, denoted by ε, satisfies $T \geq \widetilde{\mathcal{O}}(\log(1/\varepsilon)^2/\lambda^2)$.

regret Finally, we assess the label prediction performance of Model Picker via its regret. The regret of Model Picker is scaled by $\mathcal{O}(\sqrt{T\log k})$, which indicates that the prediction capability of Model Picker is very close to that of the *true* best model.

3.2.5 *Experimental Validation*

We conduct an extensive set of experiments to demonstrate the practical performance of Model Picker for online model selection and sequential label prediction. We first run experiments on common data sets where the instances come i.i.d. from a fixed data distribution. This setting allows us to empirically assess the performance in the stochastic setting. We then consider a more challenging scenario where examples come from a drifting data distribution, which we treat as an adversarial stream. For both sets of experiments, we examine the capability of MODEL PICKER and other adapted active learning baselines with the same labeling budget, upon having seen the entire stream of examples.

3.2.5.1 *Datasets and Model Collection*

We conduct our experiments using various models trained on common datasets such as the SemEval 2019 dataset (EMOCONTEXT) for emotion detection (*Semeval-EmoContext* 2019) and the long-term gas sensor drift dataset (DRIFT) from the UCI ML Repository (Vergara, 2012; Vergara et al., 2012) as well as on more complex datasets of natural images such as CIFAR-10 and IMAGENET. These datasets cover a wide range of scale: CIFAR-10, EMOCONTEXT and DRIFT are of smaller scale while IMAGENET is a large scale dataset. Each dataset consists of a large test set (which we later use to construct streams of examples) and (possibly multiple) training sets. For each dataset, we collect a collection of pretrained models by training various models on the training sets. For CIFAR-10, we trained 80 classifiers varying in model, architecture, and parameter settings available on Pytorch Hub[2]. The ensemble contains models having accuracies between 55-92% on a test set consisting of 10 000 CIFAR-10 images. The IMAGENET dataset poses a 1,000-class classification problem. We collected 102 image classifiers that are available on TensorFlow Hub[3]. The accuracy of these models is in the range 50-82%. For the test set, we use the whole official test set with 50,000 images. For the EMOCONTEXT dataset, we collected 8 pretrained models that are the development history of a participant in SemEval 2019. The accuracy of the models varies in 88-92% on a test set of size 5,509. Lastly, for the DRIFT dataset, we trained an SVM classifier on each of 9 batches of gas sensor data that were measured in different months. We use the last batch as a test set, which is of size 3,000. Due to the drift behaviour of sensor data among different time intervals, the accuracy of the models on the test set is relatively low, and lies in 25-60%.

[2] https://pytorch.org/hub/

[3] https://tfhub.dev/

Table 3.1: Characteristics of the datasets and model collections

Dataset	Number of Classes	Number of Instances	Number of Models	Accuracy of Models
CIFAR-10	10	10 000	80	55-92%
ImageNet	1 000	50 000	102	50-80%
Drift	6	3 000	9	26-65%
EmoContext	4	5 509	8	88-92%

3.2.5.2 *Baselines*

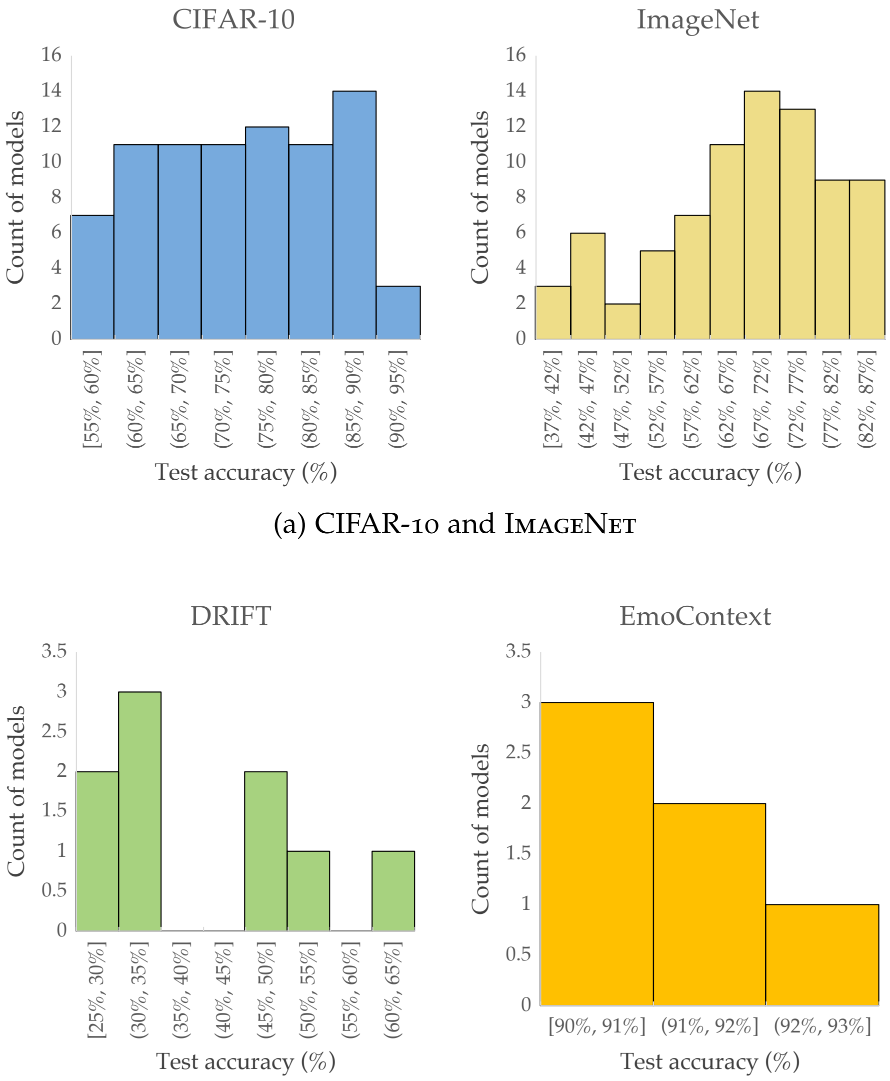
To compare with existing active learning strategies, we implement the strategies that we adapted in Section 3.2.3. These are variations of QBC, namely, vote entropy (Entropy) and structural QBC (S-QBC) as well as label efficient prediction (Efficient) and importance weighted active learning (Importance). It is crucial to note that none of the methods above are tailored for the task of *ranking pretrained models* and (except for (Cesa-Bianchi, Lugosi, and Stoltz, 2005)) for *sequential label prediction*.

3.2.5.3 *Experimental Setup*

EVALUATION PROTOCOL AND TUNING For a fair comparison, we focus on the following protocol. In order to mimic the streaming setting, we sequentially draw n i.i.d. instances uniformly at random from the entire pool of test instances, then input it into each algorithm as a stream, and call it a realization.

In each realization, the pretrained model with the highest accuracy on that stream (considering all labels) is denoted as the *true* best model of the realization. For each realization, at time t, we declare the winner of Model Picker as $i(t) = \arg\max \pi_{t,i}$ and of other methods as $i(t) = \arg\min_{i\in[k]} \sum_{s\leq t} Z_s$ $\mathbb{1}\{p_{s,i} \neq y_s\}$ where Z_s indicates if at time s is queried. Upon exhausting the stream ($t = n$), we evaluate the performance of each method based on the model that is output. We realize this process many times to have an estimate of the expected performance.

We index realizations by r and declare the model with the highest accuracy as the *true* winner of the stream, and denote it by i_r^*, that is, $i_r^* = \arg\max_{i\in[k]} \mathrm{acc}(i)$ where $\mathrm{acc}(i) = \frac{1}{n} \sum_{t\in[n]} \mathbb{1}\{p_{t,i} = y_t\}$.



(a) CIFAR-10 and ImageNet

(b) Drift and EmoContext

Figure 3.2: Histograms of model accuracies on each dataset.

For comparing the methods under the same budget constraint, we tune the (hyper-)parameter of each method to query the same number of instances, and compare their average performance under various labeling budgets. For Structural QBC, we treat β (in the posterior) as the hyperparameter. For QBC with vote entropy, importance weighted active learning and Model Picker, we *introduce* a hyperparameter β to scale the query probability according to the given labeling budget. Note that by default, Model Picker needs no hyperparameters, and we introduce β for the sole reason of fair comparison with other methods. We perform hyperparameter selection via a grid search. The hyperparameters used for each budget, together with a large range of hyperparameters and their respective budgets can be found in Appendix A.1.1.

PERFORMANCE METRICS. Recall that i_r^* denotes the true winner of the realization $r \in [R]$, and we define $i_r|b$ be the respective winner declared by a method for a labeling budget b. For any round t, we further denote the winner of realization r for budget b up to round t by $i_r(t)|b$.

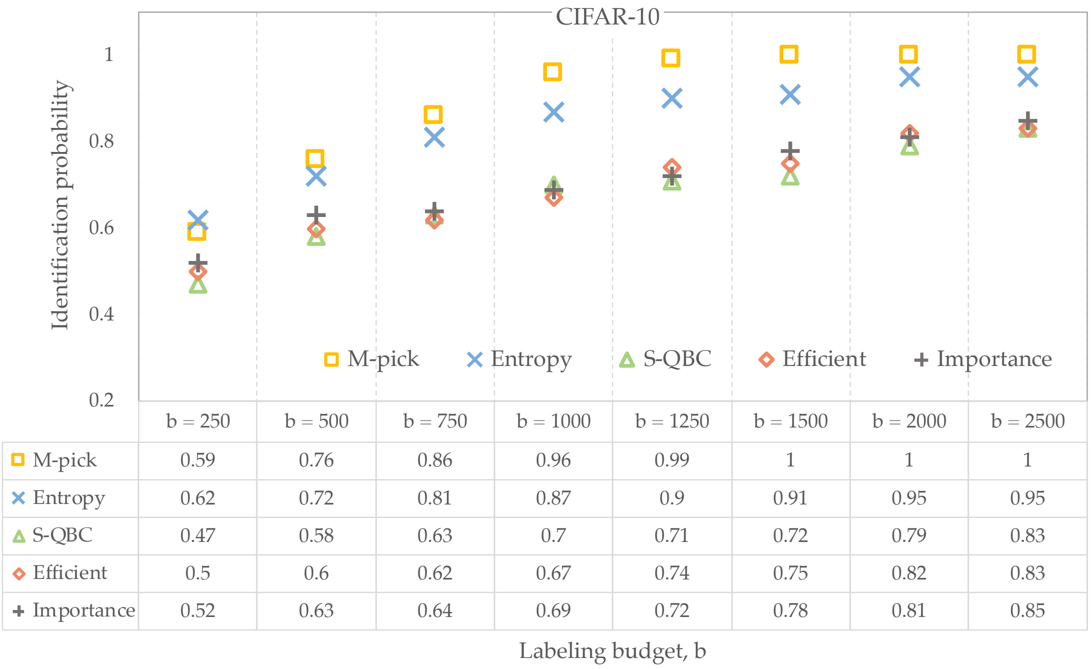
For a given labeling budget b, we consider the following key quantities as performance measures:

IDENTIFICATION PROBABILITY. The fraction of realizations that methods identify the true best model, computed as $\frac{1}{R}\sum_{r\in[R]} \mathbb{1}\{i_r = i_r^*|b\}$.

ACCURACY GAP. The accuracy gap between the returned model and the true best model, $\mathrm{acc}(i_r^*|b) - \mathrm{acc}(i_r|b)$. We report both the *average* accuracy gap and the *90th %-tile* accuracy gap over all R realizations.

REGRET. The expected regret of the algorithm considered as $\frac{1}{R}\sum_{r\in[R]}$ $\mathrm{regret}(i_r(t)|b)$ where $\mathrm{regret}(i_r(t)|b)$ is set to the difference between the accumulated loss of returned winner and true winner, up to round t. That is, $\mathrm{regret}(i_r(t)|b) = \sum_{s\leq t} \mathbb{1}\left\{p_{s,i_r(s)} \neq y_s|b\right\} - \mathbb{1}\{p_{s,i_r^*} \neq y_s\}$.

SCALING AND COMPUTATION COST. We conduct our experiments on different stream sizes. We choose sizes of 5 000, 10 000, 1 000 and 2 500 for CIFAR-10, IMAGENET, EMOCONTEXT and DRIFT test sets, respectively. We implement MODEL PICKER, along with all other baseline methods in Python. All the baseline methods combined, each realization takes between 1 second (for EMOCONTEXT) and 4 minutes (for IMAGENET) when executed on a single CPU core. MODEL PICKER alone takes between 75 miliseconds (for EMOCONTEXT) and 47 seconds (for IMAGENET). For all datasets we run 500 independent realizations for each budget constraint. To improve the overall runtime, we run the realizations in parallel over a cluster with 400 cores.



	b = 250	b = 500	b = 750	b = 1000	b = 1250	b = 1500	b = 2000	b = 2500
□ M-pick	0.59	0.76	0.86	0.96	0.99	1	1	1
× Entropy	0.62	0.72	0.81	0.87	0.9	0.91	0.95	0.95
△ S-QBC	0.47	0.58	0.63	0.7	0.71	0.72	0.79	0.83
◇ Efficient	0.5	0.6	0.62	0.67	0.74	0.75	0.82	0.83
+ Importance	0.52	0.63	0.64	0.69	0.72	0.78	0.81	0.85

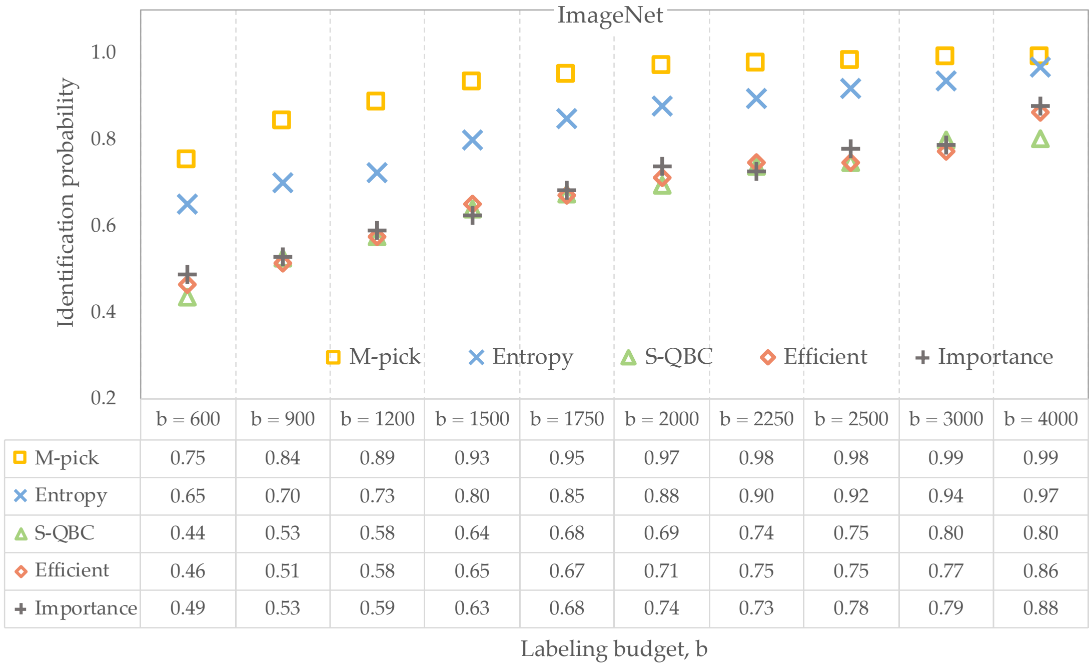
Figure 3.3: Comparison of identification probabilities for various labeling budgets on CIFAR-10

3.2.5.4 *Experimental Results*

We investigate the capability of Model Picker as well as other adapted baselines on model selection, using the true best model as the reference point. For each of our metrics introduced earlier, we observe the following:

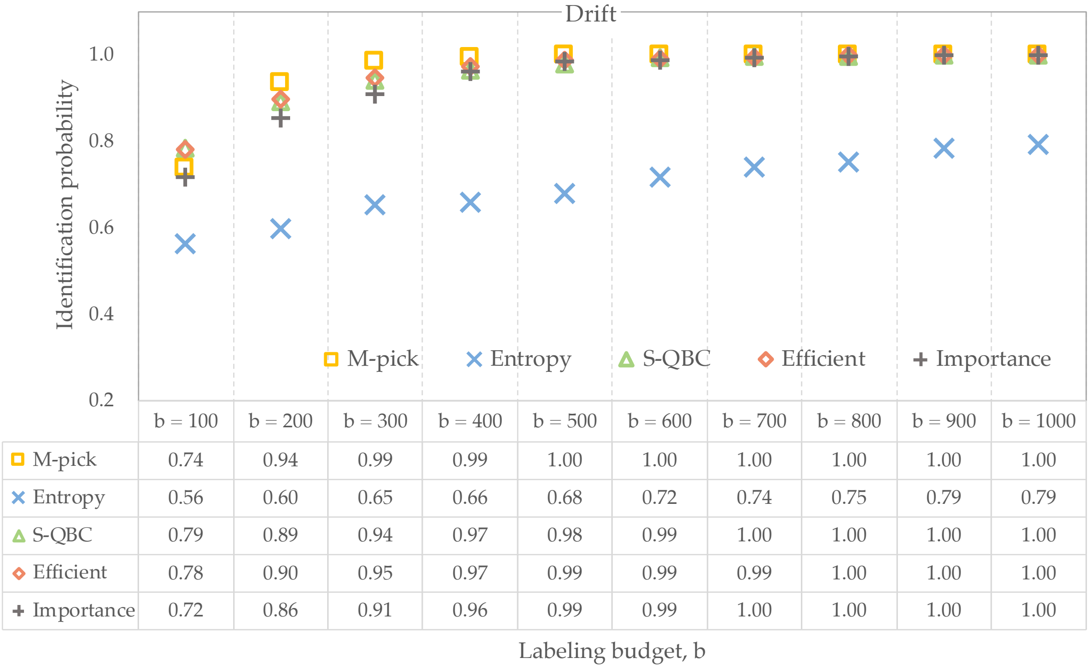
IDENTIFICATION PROBABILITY As illustrated in Figure 3.3 to Figure 3.6, Model Picker achieves significant improvements of up to 2.6× in labeling cost while returning the true best model and requesting far fewer labels than other adapted methods. For CIFAR-10, ImageNet, Drift and EmoContext datasets, Model Picker queries 2.5×, 2×, 1.5× and 1.18× fewer labels respectively than that of the best competing method (mainly Entropy) to reach confidence levels 96%, 97%, 94% and 84%, respectively. This shows that Model Picker is able to achieve the same identification power as the adapted baselines at a much lower labeling cost.

ACCURACY GAP Next, we consider the average accuracy gap over the realizations. Figure 3.7 to Figure 3.10 show that the accuracy gaps for Model



	b = 600	b = 900	b = 1200	b = 1500	b = 1750	b = 2000	b = 2250	b = 2500	b = 3000	b = 4000
□ M-pick	0.75	0.84	0.89	0.93	0.95	0.97	0.98	0.98	0.99	0.99
× Entropy	0.65	0.70	0.73	0.80	0.85	0.88	0.90	0.92	0.94	0.97
△ S-QBC	0.44	0.53	0.58	0.64	0.68	0.69	0.74	0.75	0.80	0.80
◇ Efficient	0.46	0.51	0.58	0.65	0.67	0.71	0.75	0.75	0.77	0.86
+ Importance	0.49	0.53	0.59	0.63	0.68	0.74	0.73	0.78	0.79	0.88

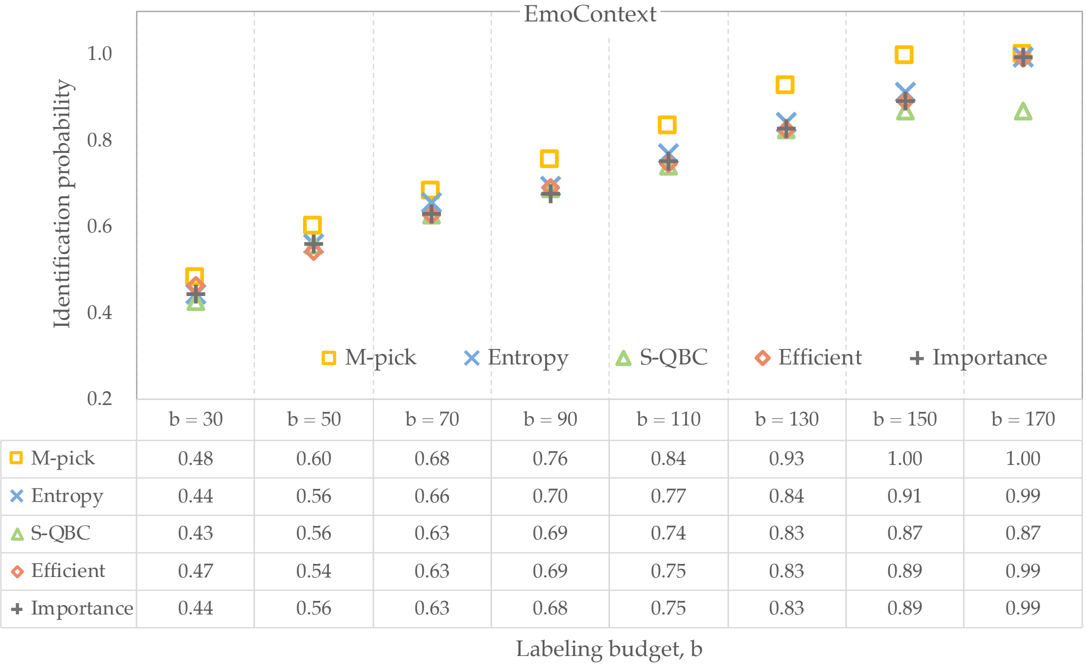
Figure 3.4: Comparison of identification probabilities for various labeling budgets on ImageNet



	b = 100	b = 200	b = 300	b = 400	b = 500	b = 600	b = 700	b = 800	b = 900	b = 1000
□ M-pick	0.74	0.94	0.99	0.99	1.00	1.00	1.00	1.00	1.00	1.00
× Entropy	0.56	0.60	0.65	0.66	0.68	0.72	0.74	0.75	0.79	0.79
△ S-QBC	0.79	0.89	0.94	0.97	0.98	0.99	1.00	1.00	1.00	1.00
◇ Efficient	0.78	0.90	0.95	0.97	0.99	0.99	0.99	1.00	1.00	1.00
+ Importance	0.72	0.86	0.91	0.96	0.99	0.99	1.00	1.00	1.00	1.00

Figure 3.5: Comparison of identification probabilities for various labeling budgets on Drift

Picker are much smaller than that of other adapted methods under the same

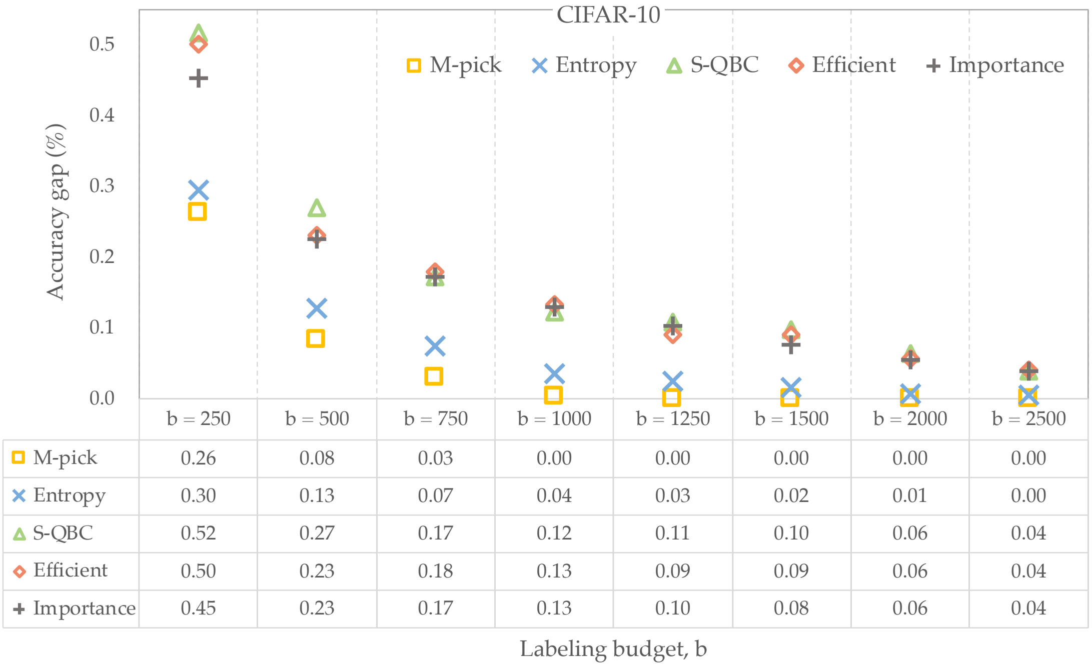


	b = 30	b = 50	b = 70	b = 90	b = 110	b = 130	b = 150	b = 170
□ M-pick	0.48	0.60	0.68	0.76	0.84	0.93	1.00	1.00
× Entropy	0.44	0.56	0.66	0.70	0.77	0.84	0.91	0.99
△ S-QBC	0.43	0.56	0.63	0.69	0.74	0.83	0.87	0.87
◇ Efficient	0.47	0.54	0.63	0.69	0.75	0.83	0.89	0.99
+ Importance	0.44	0.56	0.63	0.68	0.75	0.83	0.89	0.99

Figure 3.6: Comparison of identification probabilities for various labeling budgets on EmoContext

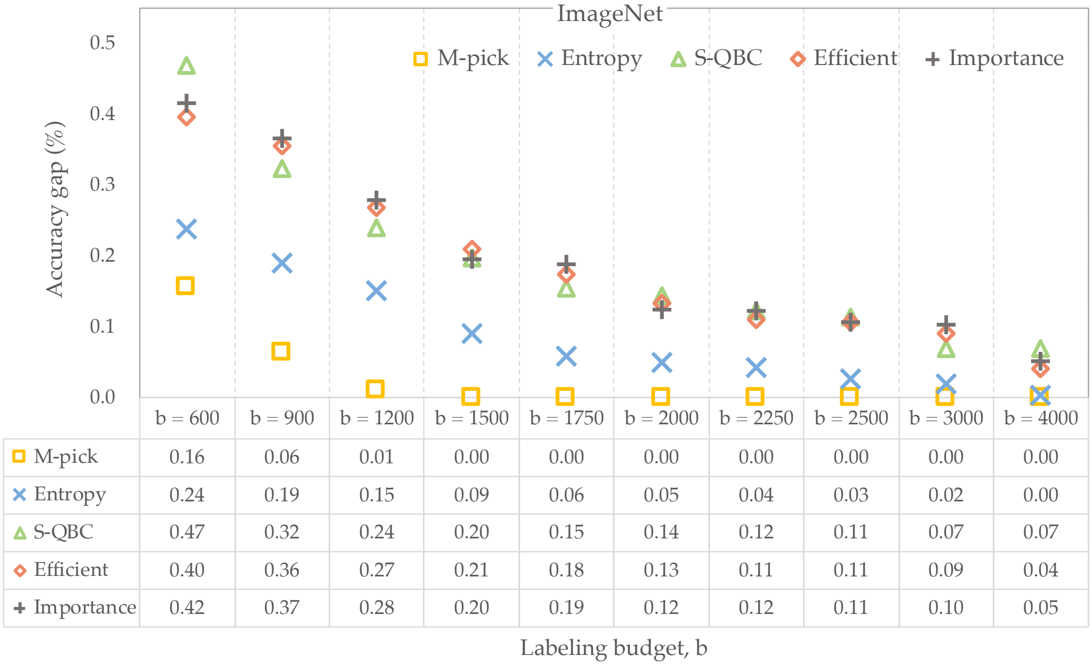
budget constraints. Quantitatively, in both CIFAR-10 and ImageNet datasets, Model Picker achieves the same expected accuracy gap as Entropy by querying nearly 2.5× less labels. For the Drift dataset, for instance, Model Picker returns a model that is within a 0.3%-neighborhood of the accuracy of best model after querying merely 12% of the entire stream of examples (when the budget is 300 for a stream of size 2 500). Note that active learning over drifting data distribution is a very challenging task, and Efficient (Label Efficient Prediction/Passive Learning) is considered the strongest baseline (Settles, 2009). Our experiments thus suggest that, even for small labeling budgets, Model Picker returns a model whose accuracy is close to that of the best model, if not the best model itself.

ON THE ROBUSTNESS OF MODEL PICKER Practitioners are often interested in the relative quality of the output model compared to the true best model in a single trial. We conduct further numerical analysis on the accuracy of the outputted models over a large number of realizations to investigate if Model Picker performs well *with high probability*. We compute the 90th per-



	b = 250	b = 500	b = 750	b = 1000	b = 1250	b = 1500	b = 2000	b = 2500
□ M-pick	0.26	0.08	0.03	0.00	0.00	0.00	0.00	0.00
× Entropy	0.30	0.13	0.07	0.04	0.03	0.02	0.01	0.00
△ S-QBC	0.52	0.27	0.17	0.12	0.11	0.10	0.06	0.04
◇ Efficient	0.50	0.23	0.18	0.13	0.09	0.09	0.06	0.04
+ Importance	0.45	0.23	0.17	0.13	0.10	0.08	0.06	0.04

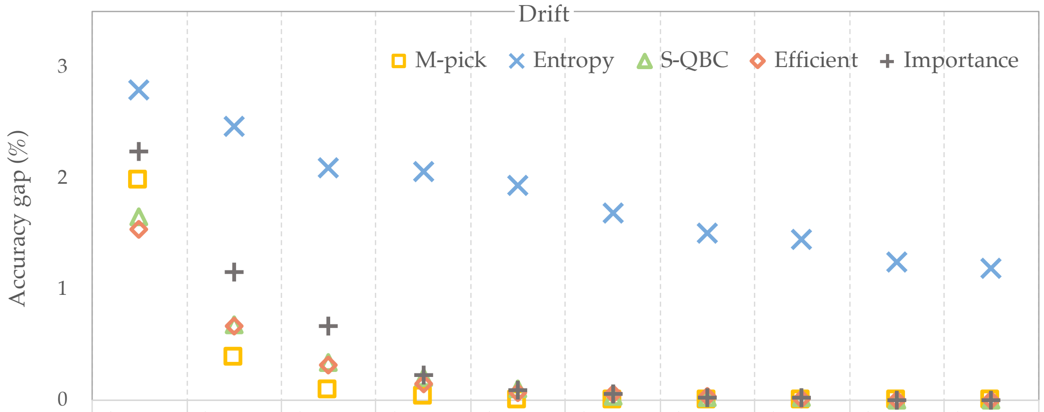
Figure 3.7: Comparison of accuracy gaps for various labeling budgets on CIFAR-10



	b = 600	b = 900	b = 1200	b = 1500	b = 1750	b = 2000	b = 2250	b = 2500	b = 3000	b = 4000
□ M-pick	0.16	0.06	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.00
× Entropy	0.24	0.19	0.15	0.09	0.06	0.05	0.04	0.03	0.02	0.00
△ S-QBC	0.47	0.32	0.24	0.20	0.15	0.14	0.12	0.11	0.07	0.07
◇ Efficient	0.40	0.36	0.27	0.21	0.18	0.13	0.11	0.11	0.09	0.04
+ Importance	0.42	0.37	0.28	0.20	0.19	0.12	0.12	0.11	0.10	0.05

Figure 3.8: Comparison of accuracy gaps for various labeling budgets on ImageNet

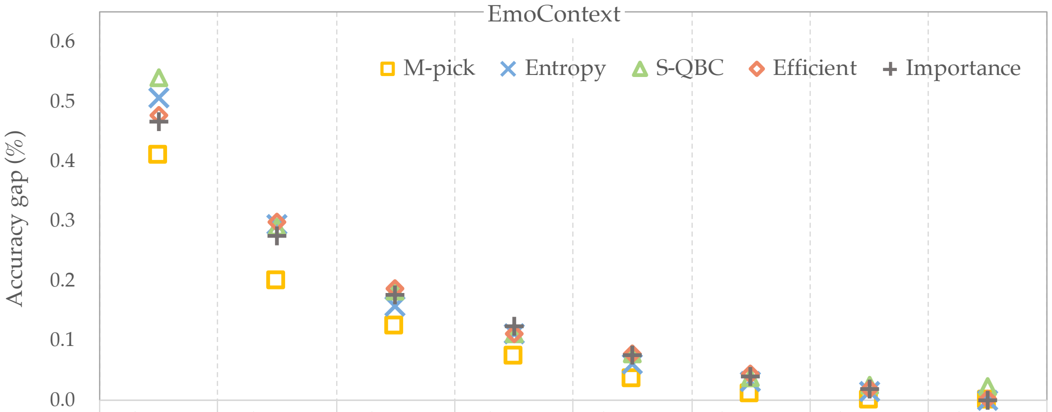
centile of accuracy gap as a proxy for the behaviour of the algorithms in the high probability regime (see Figure 3.11 to Figure 3.14). In the Drift dataset, for instance, Model Picker returns the true best model after querying merely



	b = 100	b = 200	b = 300	b = 400	b = 500	b = 600	b = 700	b = 800	b = 900	b = 1000
□ M-pick	1.98	0.38	0.09	0.03	0.00	0.00	0.00	0.00	0.00	0.00
× Entropy	2.80	2.47	2.09	2.06	1.94	1.69	1.50	1.45	1.25	1.19
△ S-QBC	1.65	0.68	0.34	0.20	0.10	0.04	0.02	0.02	0.00	0.00
◇ Efficient	1.54	0.67	0.32	0.15	0.07	0.05	0.03	0.01	0.00	0.00
+ Importance	2.24	1.15	0.66	0.23	0.09	0.06	0.03	0.02	0.01	0.00

Labeling budget, b

Figure 3.9: Comparison of accuracy gaps for various labeling budgets on Drift

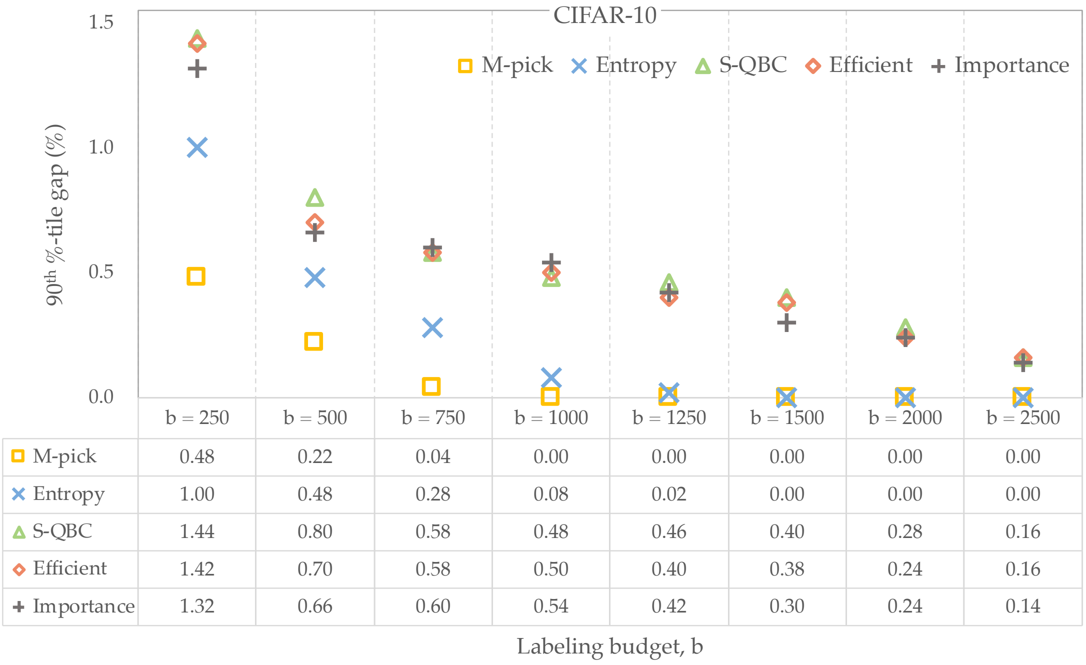


	b = 30	b = 50	b = 70	b = 90	b = 110	b = 130	b = 150	b = 170
□ M-pick	0.41	0.20	0.12	0.07	0.03	0.01	0.00	0.00
× Entropy	0.51	0.30	0.16	0.11	0.06	0.03	0.01	0.00
△ S-QBC	0.54	0.29	0.18	0.11	0.08	0.04	0.02	0.02
◇ Efficient	0.48	0.30	0.19	0.11	0.08	0.04	0.02	0.00
+ Importance	0.47	0.28	0.18	0.12	0.07	0.04	0.02	0.00

Labeling budget, b

Figure 3.10: Comparison of accuracy gaps for various labeling budgets on EmoContext

8% of the labels (when the budget is 200 with a stream size of 2 500). For the CIFAR-10 and ImageNet datasets, Model Picker returns the true best model

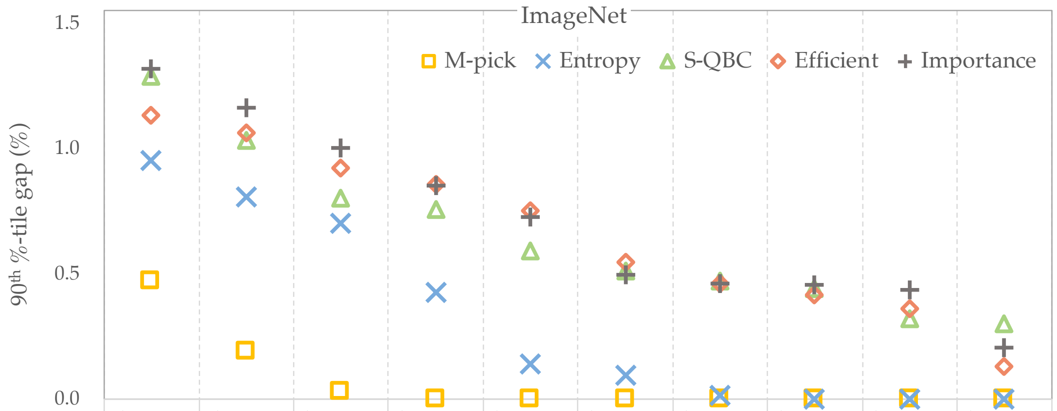


	b = 250	b = 500	b = 750	b = 1000	b = 1250	b = 1500	b = 2000	b = 2500
□ M-pick	0.48	0.22	0.04	0.00	0.00	0.00	0.00	0.00
× Entropy	1.00	0.48	0.28	0.08	0.02	0.00	0.00	0.00
△ S-QBC	1.44	0.80	0.58	0.48	0.46	0.40	0.28	0.16
◇ Efficient	1.42	0.70	0.58	0.50	0.40	0.38	0.24	0.16
+ Importance	1.32	0.66	0.60	0.54	0.42	0.30	0.24	0.14

Figure 3.11: Comparison of 90th %-tile gaps for various labeling budgets on CIFAR-10

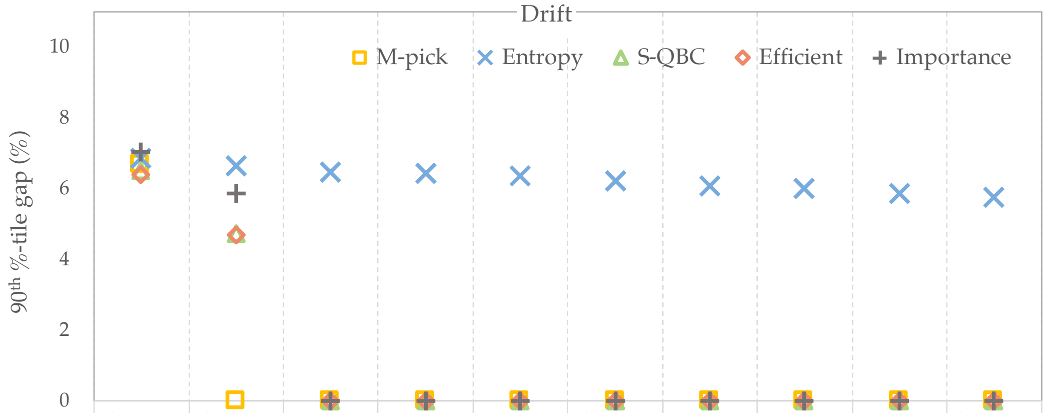
after querying respectively 20% and 15% of the entire stream of examples whereas the best competing method achieves this after querying 30% and 25% of the same stream of examples, respectively. These results demonstrate that Model Picker outputs nearly the best model if not the best.

REGRET We measure the regret across all rounds and for those budgets where Model Picker returns the best model with high confidence. Namely, we set the budget to 1 250, 1 200, 130 and 1 000 for the CIFAR-10, ImageNet, EmoContext and Drift datasets, respectively. The regret behaviour is shown in Figure 3.15. In all cases, the regret grows sub-linearly for all algorithms. The regret of our algorithm in all cases is smaller up to a factor of 1.3×, which shows that Model Picker can be used for sequential label prediction tasks as well as model selection.



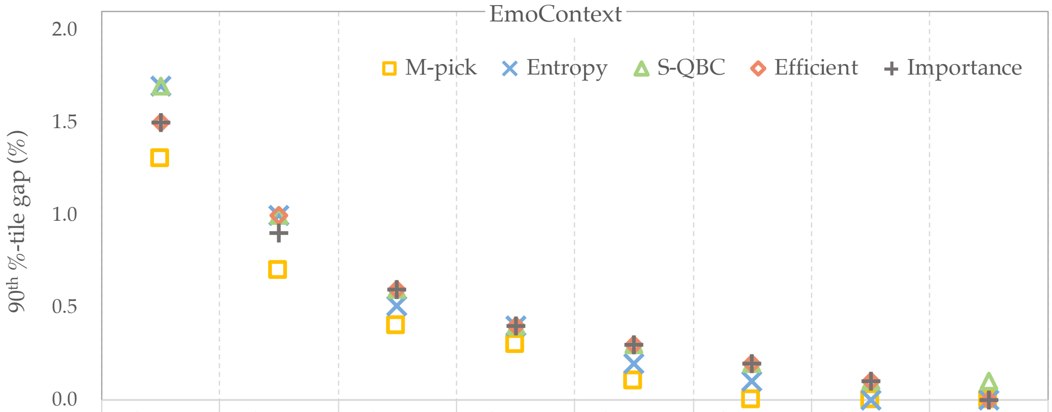
	b = 600	b = 900	b = 1200	b = 1500	b = 1750	b = 2000	b = 2250	b = 2500	b = 3000	b = 4000
□ M-pick	0.47	0.19	0.03	0.00	0.00	0.00	0.00	0.00	0.00	0.00
× Entropy	0.95	0.81	0.70	0.43	0.14	0.10	0.02	0.00	0.00	0.00
△ S-QBC	1.29	1.03	0.80	0.76	0.59	0.51	0.47	0.44	0.32	0.30
◇ Efficient	1.13	1.06	0.92	0.86	0.75	0.55	0.46	0.42	0.36	0.13
+ Importance	1.32	1.16	1.00	0.85	0.73	0.50	0.46	0.46	0.44	0.21

Figure 3.12: Comparison of 90th %-tile gaps for various labeling budgets on ImageNet



	b = 100	b = 200	b = 300	b = 400	b = 500	b = 600	b = 700	b = 800	b = 900	b = 1000
□ M-pick	6.68	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
× Entropy	6.88	6.64	6.48	6.44	6.36	6.24	6.08	6.00	5.88	5.76
△ S-QBC	6.52	4.73	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
◇ Efficient	6.40	4.68	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
+ Importance	7.04	5.88	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Figure 3.13: Comparison of 90th %-tile gaps for various labeling budgets on Drift



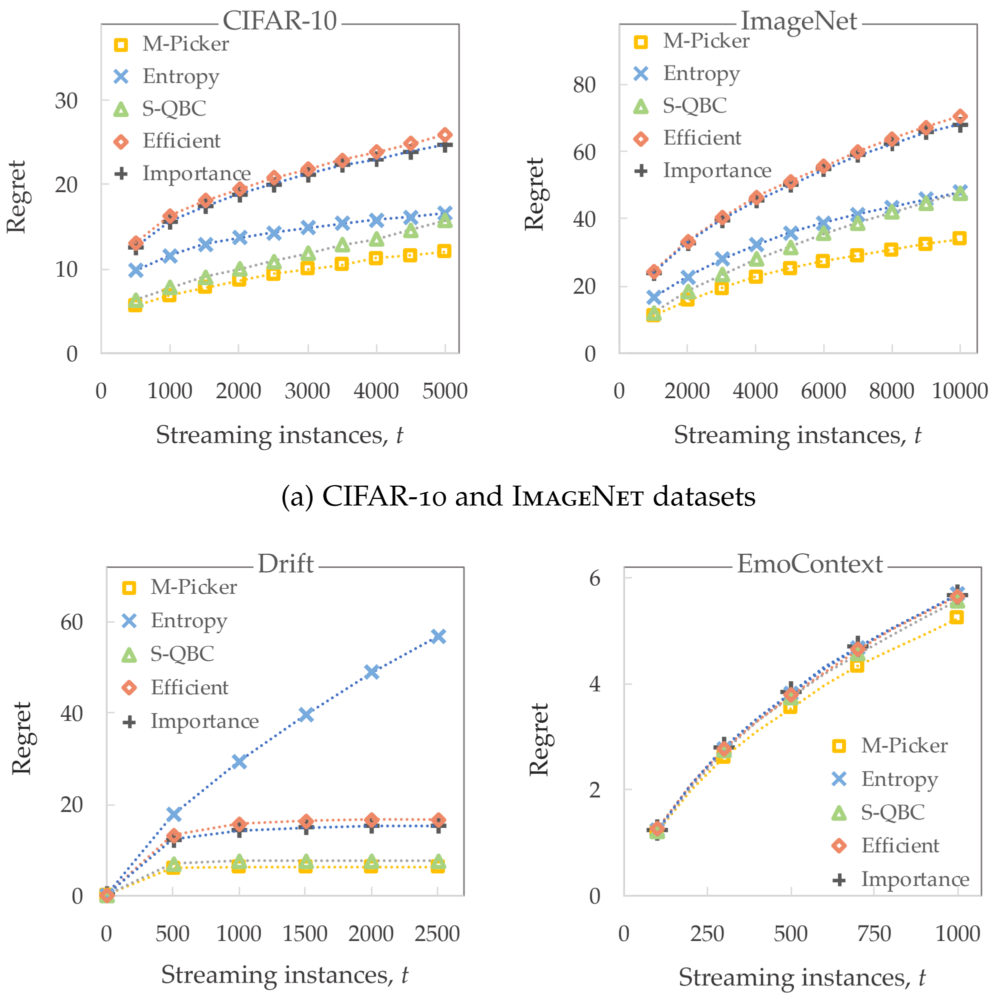
	b = 30	b = 50	b = 70	b = 90	b = 110	b = 130	b = 150	b = 170
□ M-pick	1.30	0.70	0.40	0.30	0.10	0.00	0.00	0.00
× Entropy	1.70	1.00	0.51	0.40	0.20	0.10	0.00	0.00
△ S-QBC	1.70	1.00	0.60	0.40	0.30	0.20	0.10	0.10
◇ Efficient	1.50	1.00	0.60	0.40	0.30	0.20	0.10	0.00
+ Importance	1.50	0.90	0.60	0.40	0.30	0.20	0.10	0.00

Figure 3.14: Comparison of 90th %-tile gaps for various labeling budgets on EmoContext

3.3 OVERSMOOTHING IN GRAPH NEURAL NETWORKS

3.3.1 *Related Work*

Applying deep neural networks to graphs has attracted intense interest in recent years. Motivated by the success of Convolutional Neural Networks (CNN) (Krizhevsky, Sutskever, and Hinton, 2012), Spectral CNN (Bruna et al., 2013) models the filters as learnable parameters based on the spectrum of the graph Laplacian. ChebNet (Defferrard, Bresson, and Vandergheynst, 2016) reduces computation complexity by approximating the filter with Chebyshev polynomials of the diagonal matrix of eigenvalues; Graph Convolutional Networks (GCN) (Kipf and Welling, 2016b) go further, introducing a first-order approximation of ChebNet and making several simplifications. GCN and its variants have been widely applied in various graph-related applications, including semantic relationship recognition (Xu et al., 2017), graph-to-sequence learning (Beck, Haffari, and Cohn, 2018), traffic forecasting (Li et al., 2017) and molecule classification (Such et al., 2017).



(a) CIFAR-10 and ImageNet datasets

(b) Drift and EmoContext datasets

Figure 3.15: Comparison of regret throughout the data stream (for a fixed budget) on different datasets

GCN and its variants have achieved promising results on various graph applications, while one limitation of GCN is that its performance would not improve with the increase of network depths. For instance, (Kipf and Welling, 2017) show that a two-layer GCN would achieve the best performance on a classic graph dataset while stacking more layers cannot help to improve the performance. Several studies have been conducted (Wu et al., 2019b; Zhou et al., 2018) trying to figure out the reasons behind the depth limitation and provide workarounds. (Wu et al., 2019a) hypothesizes that nonlinearity between GCN layers is not critical, which essentially implies that the deep GCN model lacks sufficient expressive ability since it is a linear model. DropEdge (Rong et al., 2019) aims to address the oversmoothing problem by randomly removing some edges from the graph. There is also a rising interest in deepening GCN by utilizing some techniques that are used to build deeper CNN architectures (e.g., ResGCN (Kipf and Welling, 2017), DenseGCN (Li et al., 2019b), JK-Net (Xu et al., 2018)). However, these lacks of evidence showing whether these techniques are helpful to improve the performance of general Graph Neural Networks (GNN).

To further understand this phenomenon in GCN, (Li, Han, and Wu, 2018) shows that GCN is a special form of Laplacian smoothing, and they prove that, under certain conditions, by repeatedly applying Laplacian smoothing many times, the features of vertices within each connected component of the graph will converge to the same value. Therefore, the oversmoothing property of GCN will make the features indistinguishable and thus hurt the classification accuracy. (Oono and Suzuki, 2019) conducts more engaged theoretical analysis. The goal of this work is to go beyond the analysis of oversmoothing, instead, we to analyze how graph decomposition can help and propose practical algorithms inspired by our analysis.

In addition, GMI (Zhen et al., 2020) proposes to maximize the correlation between input graphs and high-level hidden representations; and improves the performance on both transductive and inductive tasks. Compared with these work, we aim to develop the theoretic analysis to explain the information loss in GNN directly from the information theoretic perspective. Our work also builds on GraphCNN (Such et al., 2017), which consists of multiple adjacency matrices. As shown by (Such et al., 2017), this formulation is more expressive than CNN. In this work, we focus on providing a novel empirical study and theoretical analysis to understand the behavior of GCN and the power of graph decomposition, which in turn inspires a connectivity-aware graph decomposition method for general graph-structured data.

3.3.2 *Motivation and Problem Definition*

The success of state-of-the-art CNN go well beyond having multiple convolutional layers. Many optimization techniques such as stride, skip connection and pooling are proposed by the computer vision community over the past years to maximize the accuracy of the network. These techniques have also been shown to be beneficial for GCN, for instance, by (Li et al., 2019b). Although it is established by the prior work that a deep GraphCNN can match the accuracy of state-of-the-art CNN whereas a deep GCN cannot benefit from deep architectures, it yet remains to be an open question whether GCN can ever match the accuracy of GraphCNN when all such optimization techniques are used. As we aim to understand the role of decomposition in GraphCNN, in this section, we conduct an empirical study comparing GCN to GraphCNN and CNN when they are fully capacitated by such techniques. Our inherent goal is to understand if the graph decomposition employed by GraphCNN is a step whose impact cannot be offset or not. This will later motivate our theoretical analysis on the impact of decomposition.

We take the CIFAR-10 dataset and construct an equivalent graphical representation of the images. We treat each pixel as one node in the graph and the surrounding pixels in 9 directions (including itself) as neighboring nodes to mimic the behavior of a 3×3 convolution. The dataset consists of 60,000 images of 32×32 pixels with RGB channels: in the graphical representation, each image corresponds to a graph with 1,024 (32×32) vertices, each of which connects to the 8 neighbors plus a self-connection.

In a nutshell, we compare GCN to GraphCNN and CNN. The architecture of CNN (Krizhevsky, Sutskever, and Hinton, 2012) is stacked by 3×3 convolution layers. The input channel of the first layer is 3 (including RGB) and the output channel is set as 128. All the input and output channels of the succeeding convolution layers are 128. As for GCN (Kipf and Welling, 2016b), we treat all edges in the graph equally and leverage a similar network architecture as CNN. The only difference is that we replace each 3×3 convolution layer with a GCN layer. Finally, for GraphCNN (Such et al., 2017), we replace each convolution layer with a GraphCNN layer, which is decomposed as illustrated in Figure 3.16. Specifically, we decompose the adjancency matrix $\mathbf{A}$ into 9 submatrices $\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_9$. For two arbitrary pixels (i', j') and (m, n), we set the edges $e(p, q, m, n) = 1$ of each submatrix $\mathbf{A}_i$ when the following equation holds; otherwise the corresponding edges are set as zero in that matrix: (1) $p = q$ and $m = n$; (2) $p + 1 = q$ and $m = n$; (3) $p = q + 1$ and $m = n$; (4)

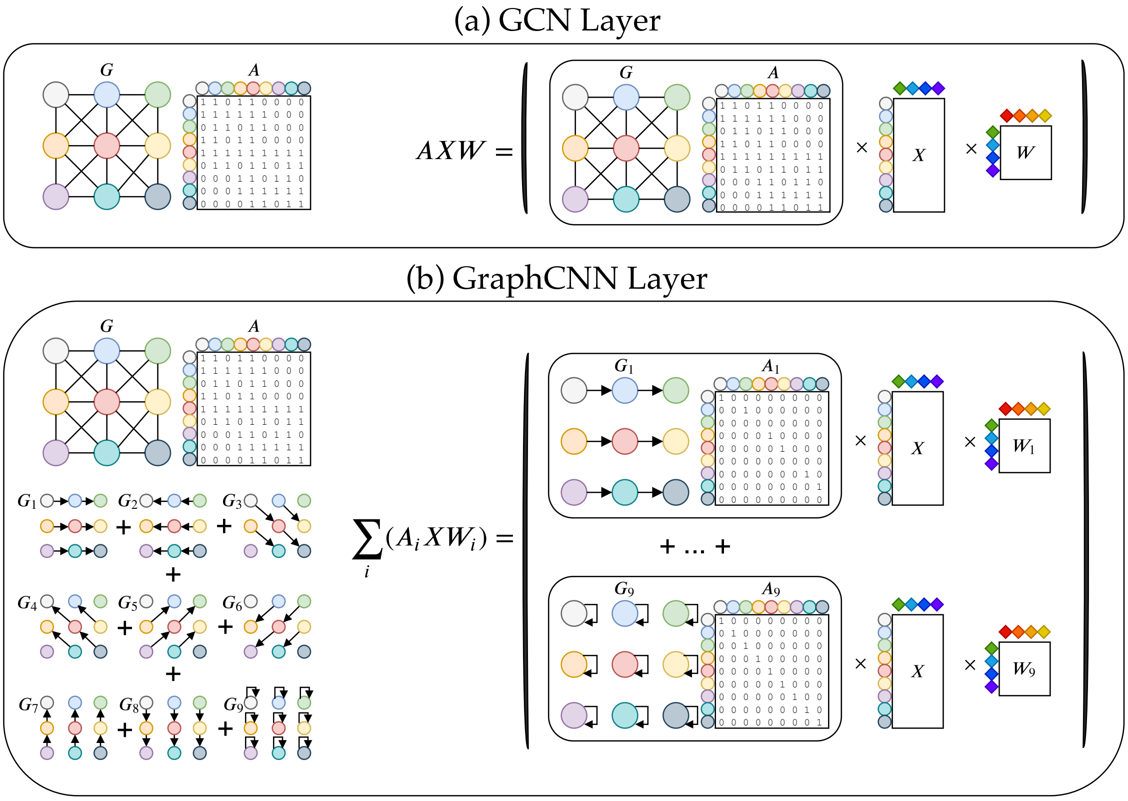
Models	GCN	GraphCNN	CNN
	(Train, Test Acc)	*(Train, Test Acc)*	*(Train, Test Acc)*
1 Layer	46.4%, 48.8%	69.4%, 66.1 %	66.7%, 68.3%
2 Layer	47.1%, 49.8%	81.1%, 80.6%	81.7%, 82.7%
5 Layer	56.9%, 57.0%	97.2%, 89,9%	93.7%, 86.6%
9 Layer	56.7%, 57.1%	99.0%, 90.3%	99.7%, 89.7%
13 Layer	56.8%, 56.9%	99.6%, 90.0%	99.04%, 87.8%
17 Layer	56.8%, 56.2%	99.1%, 88.6%	99.6%, 88.6%

Table 3.2: Comparison of GCN to GraphCNN and CNN at various depths (*original* setting)

$p = q$ and $m + 1 = n$; (5) $p = q$ and $m = n + 1$; (6) $p + 1 = q$ and $m + 1 = n$; (7) $p + 1 = q$ and $m = n + 1$; (8) $p = q + 1$ and $m + 1 = n$; (9) $p = q + 1$ and $m = n + 1$.

For each of these three architectures, we experiment with four different techniques. Namely, *original* setting where we apply convolution or graph convolution operations in each layer with $stride = 1$ and without skip connections; *stride* setting where the stride of each layer is aligned with ResNet-18; *stride+skip* setting where we add skip connections between the corresponding layers (that is, $1^{st} \rightarrow 3^{rd}$, $3^{rd} \rightarrow 5^{th}$, $5^{th} \rightarrow 7^{th}$, $7^{rd} \rightarrow 9^{th}$, $9^{rd} \rightarrow 11^{th}$, $11^{st} \rightarrow 13^{rd}$, $13^{rd} \rightarrow 15^{th}$, $15^{rd} \rightarrow 17^{th}$) following the standard architecture of ResNet-18 and *stride+skip+pooling* with average pooling on the top of stride and skip connections.

The performance of CNN, GCN and GraphCNN are listed for various settings and various depths in Table 3.2, Table 3.3, Table 3.4 and Table 3.5. Generally speaking, the accuracy of the different architectures improve with addition of different techniques. This can be observed by having a closer look at the accuracy over different settings and for the same architecture and depth thereof. Another observation we have is that the performance of GCN changes inconsistently with number of layers. For example, it degrades in the *stride* setting as shown in Table 3.3 whereas it has a slight improvement in other settings. Moreover, the performance of GCN seems to be far from that of GraphCNN and CNN for large number of layers despite the use of existing techniques. These results indicate that the decomposition, or lack thereof, plays a crucial role in deep learning with graphs.


(a) GCN Layer

(b) GraphCNN Layer

Figure 3.16: Illustration of one layer in GCN and one layer under one decomposition strategy in GraphCNN. $\mathbf{A}$ is the adjacency matrix, $\mathbf{X}$ is the input, and $\mathbf{W}$ ($\mathbf{W}_i$) are learnable weights. In GraphCNN, $\mathbf{A} = \sum_i \mathbf{A}_i$ and each entry of $\mathbf{A}_i$ is distinct from the respective entry of $\mathbf{A}_j$ for $i \neq j$. In our experiments and analysis, we follow the original normalized $\mathbf{A}$ in GCN (Kipf and Welling, 2017)

Models	GCN	GraphCNN	CNN
	(Train, Test Acc)	*(Train, Test Acc)*	*(Train, Test Acc)*
5 Layer	54.9%, 56.0%	97.3%, 89.0%	99.4%, 89.0%
9 Layer	63.8%, 60.2%	99.8%, 91.9%	99.6%, 91.4%
13 Layer	82.1%, 57.8%	99.9%, 92.9%	100%, 93.1%
17 Layer	88.9%, 53.1%	99.9%, 93.1%	100%, 93.1%

Table 3.3: Comparison of GCN to GraphCNN and CNN at various depths (*stride* setting)

So far, we observe that the depth of the network brings substantial improvement when the graph decomposition is employed by the network. One question still lingers as to *what is a good decomposition strategy?* To briefly investigate whether the decomposition strategy itself affect the performance of

Models	GCN	GraphCNN	CNN
	(Train, Test Acc)	*(Train, Test Acc)*	*(Train, Test Acc)*
5 Layer	59.1%, 58.8%	99.8%, 89.9%	99.5%, 88.7%
9 Layer	65.8%, 63.0%	100%, 93.2%	99.9%, 91.0%
13 Layer	73.5%, 64.4%	100%, 94.3%	100%, 93.0%
17 Layer	80.0%, 63.5%	100%, 94.5%	100%, 93.2%

Table 3.4: Comparison of GCN to GraphCNN and CNN at various depths (*stride+skip* setting)

Models	GCN	GraphCNN	CNN
	(Train, Test Acc)	*(Train, Test Acc)*	*(Train, Test Acc)*
5 Layer	64.7%, 63.7%	97.2%, 89.2%	99.5%, 88.7%
9 Layer	81.4%, 72.0%	99.8%, 92.1%	99.9%, 91.9%
13 Layer	90.5%, 74.7%	99.9%, 93.1%	100%, 92.9%
17 Layer	94.0%, 72.8%	99.9%, 93.2%	100%, 93.2%

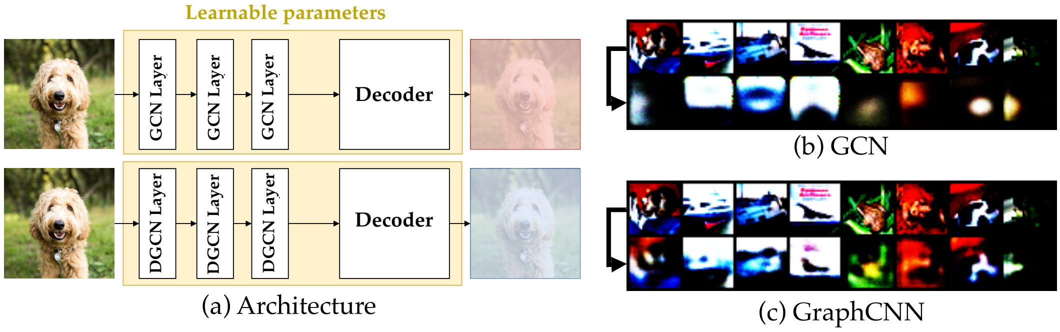
Table 3.5: Comparison of GCN to GraphCNN and CNN at various depths (*stride+skip+pooling* setting)

Setting	GCN	GraphCNN	GraphCNN (random #1, #2, #3)		
	(Train, Test Acc)	*(Train, Test Acc)*	*(Train, Test Acc)*		
original	56.8%, 56.2%	99.1%, 88.6%	69.3%, 67.7%	67.5%, 67.1%	68.3%, 68.0%
stride	88.9%, 53.1%	99.9%, 93.1%	96.1%, 74.9%	96.8%, 76.3%	97.2%, 75.0%
stride+skip	80.0%, 63.5%	100%, 94.5%	98.5%, 83.9%	99.0%, 84.8%	98.8%, 84.1%
stride+skip+pooling	94.0%, 72.8 %	99.9%, 93.2 %	97.1%, 83.6%	97.4%, 84.4%	96.9%, 83.5%

Table 3.6: Comparison of GraphCNNs with three different random decompositions using 17-layer architectures

GraphCNN or not, we evaluate GraphCNN under three random decomposition strategies and a GraphCNN decomposed by a human prior. The results are illustrated in Table 3.6. We observe that the test accuracy of decompositions varies significantly under the same conditions, indicating that the choice of decomposition strategy is decisive of the performance.

Although answering *"How does a different decomposition strategy impact the accuracy?"* is beyond the scope of our work presented here, we believe that understanding the role of decomposition itself is a first step taken at it. Then the question arises as to *How decomposition helps?* To answer this, we propose to monitor Mutual Information (MI) between the output after l GCN/GraphCNN layers and their inputs.



(a) Architecture

(b) GCN

(c) GraphCNN

Figure 3.17: (a) The neural network architecture that illustrates the Mutual Information (MI) decay after three GCN layers or three GraphCNN layers. Intuitively, the decoder estimates the MI in a similar way as MINE. (b/c) Reconstructions of test images from the output after 3 GCN/GraphCNN layers. The first row is the input images and the second row is the output images of the decoder.

As a preliminary, we perform a numerical study where we empirically measure MI between input and output layer of the network, and compare GCN and GraphCNN. We adapt an existing methodology (Belghazi et al., 2018) and use the architecture illustrated in Figure 3.17(a) as the *proxy* of the MI after l layers. Specifically, to measure the MI after l layers, we take the first l GCN/GraphCNN layers and add a fully connected layer that shrinks the hidden unit size. We then add a decoder that is a single fully connected layer that reconstructs the hidden unit size to the input. We measure the reconstruction error as modeled by l_1 loss (Janocha and Czarnecki, 2017). The idea is that, *if the network is expressive enough to preserve information after l layers, we should be able to train a decoder to recover the original input.*

Figure 3.17(b, c) illustrates the reconstruction results after $l = 3$ layers. The reconstruction error of GraphCNN (0.781) outperforms that of GCN (0.818) significantly. This empirical study is meant to show the over-smoothing phenomenon, which has been identified earlier (Li, Han, and Wu, 2018; Oono and Suzuki, 2019). In what follows, we will conduct a theoretical analysis to understand how MI scales with the number of layers l.

3.3.3 *Preliminaries*

GCN Let $G = (V, E)$ be an undirected graph with a vertex set $v_i \in V$ and set of edges $e_{i,j} \in E$. We refer to individual elements of v_i as nodes, and $\mathbf{x}_i \in \mathbb{R}^d$ associated with each v_i as features. We denote the node feature attributes by $\mathbf{X} \in \mathbb{R}^{n \times d}$ whose rows are given by $\mathbf{x}_i$. The adjacency matrix $\mathbf{A}$ (weighted or binary) is derived as an $n \times n$ matrix with $(\mathbf{A})_{i,j} = e_{i,j}$ if $e_{i,j} \in E$, and $(\mathbf{A})_{i,j} = 0$ elsewhere.

We define the following operator $f : \mathbb{R}^n \to \mathbb{R}^n$ that is composed of (1) a linear function parametrized by the adjacency matrix $\mathbf{A}$ and a weight matrix at layer $i + 1$ $\mathbf{W}^{(i+1)}$, and (2) an activation function as parametric ReLU such that $\sigma : x \to \max(x, ax)$ with $a \in (0, 1)$ that applies following the linear transformation of previous layer element-wise. Given the input matrix $\mathbf{X}$, let $\mathbf{Y}^{(0)} = \mathbf{X}$. Each layer of the network maps it to an output vector of the same shape:

$$\mathbf{Y}^{(i+1)} = f_{\mathbf{A},\mathbf{W}^{(i+1)}}(\mathbf{Y}^{(i)}) = \sigma(\mathbf{A}\mathbf{Y}^{(i)}\mathbf{W}^{(i+1)}). \tag{3.4}$$

GRAPHCNN In GraphCNN (Such et al., 2017), the adjacency matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$ is additively decomposed into K $n \times n$ matrices such that $\mathbf{A} = \sum_{k=1}^{K} \mathbf{A}_k$. The layer-wise propagation rule becomes:

$$\mathbf{Y}^{(i+1)} = g_{\mathbf{A}_k,\mathbf{W}_k^{(i+1)}}(\mathbf{X}) = \sigma\left(\sum_{k=1}^{K} \mathbf{A}_k\mathbf{X}\mathbf{W}_k^{(i+1)}\right). \tag{3.5}$$

In what follows, we denote the operator that vectorizes a matrix $\mathbf{A}$ by concatenating its columns by $vec(\mathbf{A})$. For matrices $\mathbf{A} \in \mathbb{R}^{m \times n}$ and $\mathbf{B} \in \mathbb{R}^{k \times l}$, we denote the Kronecker product of $\mathbf{A}$ and $\mathbf{B}$ by $\mathbf{A} \otimes \mathbf{B}$. Finally, we denote the jth largest singular value of a matrix $\mathbf{A}$ by $\lambda_j(\mathbf{A})$.

3.3.4 *An Anatomy of GNN*

The dramatic difference between GCN and GraphCNN can look quite counter-intuitive at the first glance. *Why can a simple decomposition of the adjacency matrix A have such a significant impact on both the accuracy and the preservation property of mutual information?* In this section, we provide a theoretical analysis of the mutual information between the l'th layer of either network and the input. Specifically, we identify the regimes where (1) the information after l GCN/GraphCNN layers with (parametric) ReLUs asymptotically converges to 0 exponentially fast, (2) the information after l GCN/GraphCNN layers with (parametric) ReLUs is perfectly preserved at the output. More importantly, compared with GCN, after l layers GraphCNN:

- requires a weaker condition for information to be perfectly preserved,
- requires a stronger condition for information to be fully lost.

Our theoretical analysis suggests that GraphCNN has a better data processing capability than that of GCN under the same characteristics of layer-wise weight matrices, justifying the observation that GraphCNN overcomes the *overcompression* introduced by GCN as we pile up more layers.

3.3.4.1 *Information Loss in GCN*

In this section, our goal is to investigate the regimes where GCN (1) does not benefit from going deeper, or (2) is guaranteed to preserve all information at its output. We aim to understand this by analyzing the behaviour of mutual information between input and output layer of the network at different depths.

First, we formulate the relationship between input and output layers incorporating the non-linear activation functions. In this work, we focus on the most popular choice, i.e., ReLU, and leave the study of other functions to future work.

As preparation to our analysis, we introduce further notations as follows. We denote the vectorized input $\mathbf{X}$ and lth layer output $\mathbf{Y}^{(l)}$ by $\mathbf{x}$ and $\mathbf{y}^{(l)}$, respectively. For some n-dimensional real random vectors $\mathbf{x}$ and $\mathbf{y}$ defined over finite alphabets $\mathcal{X}^n$ and Ω^n, we denote entropy of $\mathbf{x}$ by $\mathcal{H}(\mathbf{x})$ and mutual information between $\mathbf{x}$ and $\mathbf{y}$ by $\mathcal{I}(\mathbf{x};\mathbf{y})$. Moreover, information loss is defined by $\mathcal{L}(\mathbf{y}^{(l)}) = \mathcal{H}(\mathbf{x}|\mathbf{y}^{(l)})$, i.e., relative entropy of $\mathbf{x}$ with respect to $\mathbf{y}^{(l)}$.

First, the characteristics of the layer-wise propagation rule in Equation 3.4 lead us to the following result:

Lemma 5 (Linearization of GCN Layers via Kronecker Product)**.** *Let $\mathbf{P}^{(i+1)}$ be a diagonal matrix whose nonzero entries are in $\{a, 1\}$ with $a \in (0, 1)$ such that $(\mathbf{P}^{(i+1)})_{j,j} = 1$ if $\big((\mathbf{W}^{(i+1)} \otimes \mathbf{A})\mathbf{y}^{(i)}\big)_j \geq 0$, and $(\mathbf{P}^{(i+1)})_{j,j} = a$ elsewhere. $\mathbf{y}^{(l)}$ can then be written as*

$$\mathbf{y}^{(l)} = \mathbf{P}^{(l)}(\mathbf{W}^{(l)} \otimes \mathbf{A}) \cdots \mathbf{P}^{(2)}(\mathbf{W}^{(2)} \otimes \mathbf{A})\mathbf{P}^{(1)}(\mathbf{W}^{(1)} \otimes \mathbf{A})\mathbf{x}.$$

Following our earlier discussion, we will now state our first result which characterizes the regime in which the information propagated across the GCN layers exponentially decays to 0.

Theorem 5 (Information Loss in GCN)**.** *Let GCN follows the propagation rule introduced in Equation 3.4. Suppose $\sigma_{\mathbf{A}} = \max_j \lambda_j(\mathbf{A})$ and $\sigma_{\mathbf{W}} = \sup_{i \in \mathbb{N}^+} \max_j \lambda_j(\mathbf{W}^{(i)})$. If $\sigma_{\mathbf{A}}\sigma_{\mathbf{W}} < 1$, then $\mathcal{I}(\mathbf{x};\mathbf{y}^{(l)}) = \mathcal{O}\big((\sigma_{\mathbf{A}}\sigma_{\mathbf{W}})^l\big)$, and hence $\lim_{l\to\infty} \mathcal{I}(\mathbf{x};\mathbf{y}^{(l)}) = 0$*

This result indicates that under certain conditions the information after l GCN layers with (parametric) ReLUs asymptotically converges to 0 exponentially fast.[4] Interestingly, there are also regimes in which GCN will perfectly preserve the information, stated as follows:

Theorem 6 (Information Preservation in GCN)**.** *Following Theorem 5, let now $\gamma_{\mathbf{A}} = \min_j \lambda_j(\mathbf{A})$ and $\gamma_{\mathbf{W}} = \inf_{i \in \mathbb{N}^+} \min_j \lambda_j(\mathbf{W}^{(i)})$. If $a\gamma_{\mathbf{A}}\gamma_{\mathbf{W}} \geq 1$, then $\forall l \in \mathbb{N}^+$ $\mathcal{L}(\mathbf{y}^{(l)}) = 0$.*

EFFECT OF NORMALIZED LAPLACIAN: The results obtained above holds for any adjacency matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$. The unnormalized $\mathbf{A}$, however, comes with a major drawback as changing the scaling of feature vectors. To overcome this problem, $\mathbf{A}$ is often normalized such that its rows sum to one. We then adopt our results to GCN with normalized Laplacian whose largest singular value is one. We have the following result:

[4] Theorem 5 also holds for traditional ReLU with $f : x \to x^+ = \max(0, x)$.

Corollary 3 (Information Loss in GCN with Normalized Laplacian)**.** *Let* $\mathbf{D}$ *denote the degree matrix such that* $(\mathbf{D})_{j,j} = \sum_m (\mathbf{A})_{j,m}$*, and* $\mathbf{L}$ *be the associated normalized Laplacian* $\mathbf{L} = \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$*. Suppose GCN uses the following mapping* $\mathbf{Y}^{(i+1)} = \sigma(\mathbf{L}\mathbf{Y}^{(i)}\mathbf{W}^{(i)})$*. Let also* $\sigma_{\mathbf{W}} = \sup_i \max_j \lambda_j(\mathbf{W}^{(i+1)})$*. If* $\sigma_{\mathbf{W}} < 1$*, then* $\mathcal{I}(\mathbf{x};\mathbf{y}^{(l)}) = \mathcal{O}\big(\sigma_{\mathbf{W}}^l\big)$*, and hence* $\lim_{l\to\infty} \mathcal{I}(\mathbf{x};\mathbf{y}^{(l)}) = 0$*.*

3.3.4.2 *Information Loss in GraphCNN*

Motivated by the graph decomposition strategy adopted by several works including GraphCNN, in this section we aim to analyze the information loss after graph decomposition, and understand whether the information can be preserved by aggregating local sub-graphs. In particular, we take the GCN as as an example which sums the decomposed graphs together as the adjacency matrix to perform the analysis.

Similarly as in Lemma 5, $\mathbf{y}^{(l)}$ can be reduced to

$$\mathbf{y}^{(l)} = \mathbf{P}^{(l)} \sum_{k_l=1}^{K} (\mathbf{W}_{k_l}^{(l)} \otimes \mathbf{A}_{k_l}) \cdots (\mathbf{W}_{k_2}^{(2)} \otimes \mathbf{A}_{k_2})(\mathbf{W}_{k_1}^{(1)} \otimes \mathbf{A}_{k_1})\mathbf{x}$$

for a diagonal matrix $\mathbf{P}^{(i+1)}$ such that $(\mathbf{P}^{(i+1)})_{j,j} = 1$ if

$$\sum_{k_{i+1}=1}^{K} (\mathbf{W}_{k_{i+1}}^{(i+1)} \otimes \mathbf{A}_{k_{i+1}})\mathbf{y}^{(i)} \geq 0$$

and $(\mathbf{P}^{(i+1)})_{j,j} = a$ otherwise.

Following the same proof steps performed for GCN, we obtain the following result for GraphCNN:

Theorem 7 (Information Loss in GraphCNN)**.** *Let* $\sigma^{(i)}$ *denotes the maximum singular value of* $\mathbf{P}^{(i)} \sum_{k_i=1}^{K} (\mathbf{W}_{k_i}^{(i)} \otimes \mathbf{A}_{k_i})$ *such that* $\sigma^{(i)} = \max_j \lambda_j\big(\mathbf{P}^{(i)} \sum_{k_i} (\mathbf{W}_{k_i}^{(i)} \otimes \mathbf{A}_{k_i})\big)$*. If* $\sup_{i\in\mathbb{N}^+} \sigma^{(i)} < 1$*, then* $\mathcal{I}(\mathbf{x};\mathbf{y}^{(l)}) = \mathcal{O}\big((\sup_{i\in\mathbb{N}^+} \sigma^{(i)})^l\big)$*, and hence* $\lim_{l\to\infty} \mathcal{I}(\mathbf{x};\mathbf{y}^{(l)}) = 0$*.*

Theorem 7 describes the condition on the layer-wise weight matrices $\mathbf{W}_k$ where GraphCNN fails to capture the feature characteristics at its output in the asymptotic regime. We then state the second result for GraphCNN which ensures $\mathcal{L}(\mathbf{y}^{(l)}) = 0$ as follows.

Theorem 8 (Information Preservation in GraphCNN)**.** *Consider the propagation rule in Equation 3.5. Let $\gamma^{(i)}$ denotes the minimum singular value of $\mathbf{P}^{(i)} \sum_{k_i=1}^{K} (\mathbf{W}_{k_i}^{(i)}$ $\otimes \mathbf{A}_{k_i})$ such that $\gamma^{(i)} = \min_j \lambda_j \big(\mathbf{P}^{(i)} \sum_{k_i=1}^{K} (\mathbf{W}_{k_i}^{(i)} \otimes \mathbf{A}_{k_i})\big)$. If $\inf_i \gamma^{(i)} \geq 1$, then $\forall l \in \mathbb{N}^+$ we have $\mathcal{L}(\mathbf{y}^{(l)}) = 0$.*

In order to understand the role of decomposition in GraphCNN, we revisit the conditions on $\mathcal{I}(\mathbf{x}; \mathbf{y}^{(l)}) = 0$ and $\mathcal{H}(\mathbf{y}^{(l)}) = 0$ for a specific choice of decomposition, which will later be used to demonstrate the information processing capability of GraphCNN.

Corollary 4 (Information Loss in GraphCNN for Orthogonal Decomposition)**.** *Suppose the singular value decomposition of $\mathbf{A}$ is given by $\mathbf{A} = \mathbf{U}_{\mathbf{A}} \mathbf{S} \mathbf{V}_{\mathbf{A}}^T$, and each $\mathbf{A}_k$ is set to $\mathbf{A}_k = \mathbf{U}_{\mathbf{A}} \mathbf{S}_k \mathbf{V}_{\mathbf{A}}^T$ where $(\mathbf{S}_k)_{m,m} = \lambda_m(\mathbf{A})$ if $k = m$ and $(\mathbf{S}_k)_{m,m} =$ 0 elsewhere. We then have the following results: For $\sigma_{\mathbf{A}_k} = \lambda_k(\mathbf{A})$ and $\sigma_{\mathbf{W}_k} =$ $\sup_{i \in \mathbb{N}^+} \max_j \lambda_j (\mathbf{W}_k^{(i)})$, i.e., if $\sigma_{\mathbf{A}_k} \sigma_{\mathbf{W}_k} < 1$ $\underline{\forall k = \{1, 2, \ldots, n\}}$, then $\lim_{l \to \infty} \mathcal{I}(\mathbf{x}; \mathbf{y}^{(l)}) = 0$.*

Corollary 5 (Information Preservation in GraphCNN for Orthogonal Decomposition)**.** *Let $\gamma_{\mathbf{W}_k} = \inf_{i \in \mathbb{N}^+} \min_j \lambda_j (\mathbf{W}_k^{(i)})$. If $a \sigma_{\mathbf{A}_k} \gamma_{\mathbf{W}_k} \geq 1$, $\underline{\forall k \in \{1, 2, \ldots, n\}}$, then $\mathcal{L}(\mathbf{y}^{(l)}) = 0$ $\forall l \in \mathbb{N}^+$.*

While the universally optimal decomposition strategy is unknown and its existence is debatable, the choice of decomposition introduced above highlight the dramatic difference between the capabilities of GCN and GraphCNN.

Proof Sketch. Following Lemma 5, the next key step in proving above results is as follows.

Lemma 6. *Consider the singular value decomposition*

$$\mathbf{U} \mathbf{\Lambda} \mathbf{V}^T = \mathbf{P}^{(l)} (\mathbf{W}^{(l)} \otimes \mathbf{A}) \ldots \mathbf{P}^{(2)} (\mathbf{W}^{(2)} \otimes \mathbf{A}) \mathbf{P}^{(1)} (\mathbf{W}^{(1)} \otimes \mathbf{A})$$

$$\textit{such that} \quad (\mathbf{\Lambda})_{j,j} = \lambda_j (\mathbf{P}^{(l)} (\mathbf{W}^{(l)} \otimes \mathbf{A}) \ldots \mathbf{P}^{(2)} (\mathbf{W}^{(2)} \otimes \mathbf{A}) \mathbf{P}^{(1)} (\mathbf{W}^{(1)} \otimes \mathbf{A}))$$

and let $\tilde{\mathbf{x}} = \mathbf{V}^T\mathbf{x}$*. We have*

$$\mathcal{I}(\mathbf{x};\mathbf{y}^{(l)}) \overset{(1)}{=} \mathcal{I}(\tilde{\mathbf{x}};\Lambda\tilde{\mathbf{x}}) \overset{(2)}{\leq} \mathcal{H}(\tilde{\mathbf{x}}) \overset{(3)}{=} \mathcal{H}(\mathbf{x}) \tag{3.6}$$

where (1, 3) results from that $\mathbf{U}$ and $\mathbf{V}$ are invertible, and equality holds in (2) *iff* $\mathbf{\Lambda}$ is invertible, i.e., singular values of $\mathbf{P}^{(l)}(\mathbf{W}^{(l)} \otimes \mathbf{A})...\mathbf{P}^{(2)}(\mathbf{W}^{(2)} \otimes \mathbf{A})\mathbf{P}^{(1)}(\mathbf{W}^{(1)} \otimes \mathbf{A})$ are nonzero. □

Theorem 5, Theorem 6, Theorem 7 and Theorem 8 can easily be inferred from Lemma 6. That is, $\mathcal{I}(\mathbf{x};\mathbf{y}^{(l)}) = 0$ *iff* $\max_j(\mathbf{\Lambda}^l)_{j,j} = 0$ in the asymptotic regime. Similarly, *iff* $\min_j(\mathbf{\Lambda}^l)_{j,j} > 0$, $\mathcal{I}(\mathbf{x};\mathbf{y}^{(l)})$ is maximized and given by $\mathcal{H}(\mathbf{x})$, hence $\mathcal{L}(\mathbf{y}^{(l)}) = 0$.

Our results presented so far focus on covering the edge cases: $\mathcal{I}(\mathbf{x};\mathbf{y}^{(l)}) = 0$ or $\mathcal{L}(\mathbf{y}^{(l)}) = 0$. While our primary goal is to understand why GraphCNN has a better capability of going deep than that of GCN, we note several points about Lemma 6 in a viewpoint of entropy or uncertainty:

1. Rigorous theoretical guarantees quantifying the amount of information preserved across the network are not straightforward, and further require the knowledge on the statistical properties of node features. Despite its simplicity, Lemma 6 forms a direct link from the information processing capability of the network to the characteristics of the weights and entropy of the nodes, $\mathbf{x}_i$,

2. Whereas the compression and generalization capability of the network are closely related, we emphasize here that our analysis here is to understand why and when GraphCNN overcome the *overcompression* introduced by GCN. In future, we plan to investigate this via the information bottleneck principle,

3. In our formulation, we omit the effect of perturbation in the input nodes considering our discussion will remain valid under the same perturbation characteristics,

4. If all node features $\mathbf{x}_i$, for instance, have similar entropy, $\mathcal{I}(\mathbf{x};\mathbf{y}^{(l)})$ roughly linearly scales with the rank of $\mathbf{P}^{(l)}(\mathbf{W}^{(l)} \otimes \mathbf{A})...\mathbf{P}^{(2)}(\mathbf{W}^{(2)} \otimes \mathbf{A})\mathbf{P}^{(1)}(\mathbf{W}^{(1)} \otimes \mathbf{A})$,

5. Lifting up singular values of layer-wise weight matrices are beneficial for better data processing in a viewpoint of information theory. In the next section, we will demonstrate through edge cases how GraphCNN can overcome *overcompression* of GCN by achieving singular value lifting.

3.3.4.3 *Discussion: Impact of Decomposition*

Consider the setting where $\mathbf{A}$ is fixed and same for both GCN and GraphCNN. The discussions below will revolve around the regime of singular values of layer-wise weight matrices, $\mathbf{W}^{(i)}_{\text{GCN}}$ and $\mathbf{W}^{(i)}_{\text{GraphCNN}}$ where (1) $\mathcal{L}(\mathbf{y}^{(l)}) = 0$, and (2) $\mathcal{I}(\mathbf{x};\mathbf{y}^{(l)}) = 0$.

1. Recall from Theorem 6 and Corollary 5 that while GCN requires singular values of all weight matrices $\mathbf{W}^{(i)}_{\text{GCN}}$ to compensate for the minimum singular value of $\mathbf{A}$ such that $\min_j \lambda_j(\mathbf{W}^{(i)}_{\text{GCN}}) \geq \frac{1}{a \min_k \lambda_k(\mathbf{A})}$ to ensure $\mathcal{L}(\mathbf{y}^{(l)}) = 0$, GraphCNN relaxes this condition by introducing a milder constraint. That is, the singular values of its weight matrices $\mathbf{W}^{(i)}_{\text{k, GraphCNN}}$ need to compensate only for the singular value of their respective component $\mathbf{A}_k$, that is, $\min_j \lambda_j(\mathbf{W}^{(i)}_{\text{k, GraphCNN}}) \geq \frac{1}{a\lambda_k(\mathbf{A})}$ guarantees that $\mathcal{L}(\mathbf{y}^{(l})= 0$. In other words, singular values of weight matrices of GraphCNN are lower bounded by much smaller values than that of GCN such that information can be fully recovered at the output layer, hence $\mathcal{L}(\mathbf{y}^{(l)}) = 0$ results for GraphCNN in a much larger regime of weights.

2. The above discussion also applies to the regimes where $\mathcal{I}(\mathbf{x};\mathbf{y}^{(l)}) = 0$. From Theorem 5 and Corollary 4, we recall that the information contained in the output layer of GraphCNN exponentially decays to zero if $\forall k \in \{1, 2, \ldots, n\}$,

$$\max_j \lambda_j(\mathbf{W}^{(i)}_{\text{k, GraphCNN}}) < \frac{1}{\lambda_k(\mathbf{A})}$$

whereas this regime is much larger for GCN such that $\max_j \lambda_j(\mathbf{W}^{(i)}_{\text{GCN}}) < \frac{1}{\max_k \lambda_k(\mathbf{A})}$.

The decomposition makes deep GCN training easier by permitting a much larger regime of model weights where the information is still preserved. In other words, under the same weight characteristics (singular values of layer-wise weight matrices), the decomposed GCN will be able to preserve more information of the node features than the vanilla GCN when going deeper. So far, we theoretically justify the potential of graph decomposition in the infinite-sample regime. For the analysis in the finite-sample regime, one could possibly utilize the theory of information bottleneck (Saxe et al., 2019; Shamir, Sabato, and Tishby, 2010), we leave this as future work.

3.3.5 *Proofs*

NOTATION Hereafter, scalars will be written in italics, vectors in bold lower-case and matrices in bold upper-case letters. For an $m \times n$ real matrix $\mathbf{A}$, the matrix element in the ith row and jth column is denoted as $(\mathbf{A})_{ij}$, and ith entry of a vector $\mathbf{a} \in \mathbb{R}^m$ by $(\mathbf{a})_i$. Also, jth column of $\mathbf{A}$ is denoted by $(\mathbf{A})_j$, or $(\mathbf{A})_{[i=1,2,\ldots,m],j}$. Similarly, we denote ith row by $(\mathbf{A})_{i,[j=1,2,\ldots,n]}$. The inner product between two vectors $(\mathbf{A})_i$ and $(\mathbf{A})_{i'}$ is denoted by $\langle (\mathbf{A})_i, (\mathbf{A})_{i'} \rangle$.

We vectorize a matrix $\mathbf{A}$ by concatenating its columns such that

$$vec(\mathbf{A}) = \begin{bmatrix} (\mathbf{A})_1 \\ (\mathbf{A})_2 \\ \vdots \\ (\mathbf{A})_n \end{bmatrix}$$

and denote it by $vec(\mathbf{A})$. For matrices $\mathbf{A} \in \mathbb{R}^{m\times n}$ and $\mathbf{B} \in \mathbb{R}^{k\times l}$, we denote the kronecker product of $\mathbf{A}$ and $\mathbf{B}$ by $\mathbf{A} \otimes \mathbf{B}$ such that

$$\mathbf{A} \otimes \mathbf{B} = \begin{bmatrix} (\mathbf{A})_{11}\mathbf{B} & \ldots & (\mathbf{A})_{1n}\mathbf{B} \\ \vdots & \ddots & \vdots \\ (\mathbf{A})_{m1}\mathbf{B} & \ldots & (\mathbf{A})_{mn}\mathbf{B} \end{bmatrix}.$$

Note that $\mathbf{A} \otimes \mathbf{B}$ is of size $mk \times nl$.

We moreover denote the floor function and modulo operation by $\lfloor \cdot \rfloor$ and mod, respectively. Finally, we denote the jth largest singular value of a matrix $\mathbf{A}$ by $\lambda_j(\mathbf{A})$.

Next, we list some existing results which we require repeatedly throughout this section.

PRELIMINARIES.

1. Suppose $\mathbf{A} \in \mathbb{R}^{m\times n}$, $\mathbf{B} \in \mathbb{R}^{n\times k}$ and $\mathbf{C} \in \mathbb{R}^{k\times p}$. We have

$$vec(\mathbf{ABC}) = (\mathbf{C}^T \otimes \mathbf{A})vec(\mathbf{B}). \tag{3.7}$$

2. Let $\mathbf{A} \in \mathbb{R}^{m\times n}$, $\mathbf{B} \in \mathbb{R}^{n\times k}$ and $\mathbf{C} \in \mathbb{R}^{m'\times n'}$, $\mathbf{D} \in \mathbb{R}^{n'\times k'}$

$$(\mathbf{AB} \otimes \mathbf{CD}) = (\mathbf{A} \otimes \mathbf{C})(\mathbf{B} \otimes \mathbf{D}). \tag{3.8}$$

3. For $\mathbf{A} \in \mathbb{R}^{m\times m}$ and $\mathbf{B} \in \mathbb{R}^{n\times n}$, singular values of $\mathbf{A} \otimes \mathbf{B}$ is given by $\lambda_i(\mathbf{A})\lambda_j(\mathbf{B})$, $i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots, n$.

4. Let $\mathbf{x}$ and $\mathbf{y}$ be an n-dimensional random vector defined over finite alphabets $\mathcal{X}^n$ and Ω^n, respectively. We denote entropy of $\mathbf{x}$ by $\mathcal{H}(\mathbf{x})$ and mutual information between $\mathbf{x}$ and $\mathbf{y}$ by $\mathcal{I}(\mathbf{x}; \mathbf{y})$. We list the followings:

$$\begin{aligned} \mathcal{H}(f(\mathbf{x})) &\overset{(a)}{\leq} \mathcal{H}(\mathbf{x}) \\ \mathcal{I}(\mathbf{x}; f(\mathbf{y})) &\overset{(b)}{\leq} \mathcal{I}(\mathbf{x}; \mathbf{y}) \end{aligned} \tag{3.9}$$

such that $f : \mathbb{R} \to \mathbb{R}$ is some deterministic function, and equality holds for both inequalities *iff* f is bijective.

PROOFS. The proofs are listed below in order.

Proof of Lemma 5. Applying vectorization to the layer-wise propagation rule introduced in Equation 3.4, we have

$$\begin{aligned}
&\mathbf{y}^{(i+1)}{=}vec\big(\sigma(\mathbf{A}\mathbf{Y}^{(i)}\mathbf{W}^{(i+1)})\big)\\
&\mathbf{y}^{(i+1)} \overset{(a)}{=} \sigma\big(vec(\mathbf{A}\mathbf{Y}^{(i)}\mathbf{W}^{(i+1)})\big)\\
&\mathbf{y}^{(i+1)} \overset{(b)}{=} \sigma\big(((\mathbf{W}^{(i+1)})^T \otimes \mathbf{A})\mathbf{y}^{(i)}\big)\\
&\mathbf{y}^{(i+1)} \overset{(c)}{=} \mathbf{P}^{(i+1)}((\mathbf{W}^{(i+1)})^T \otimes \mathbf{A})\mathbf{y}^{(i)}
\end{aligned} \tag{3.10}$$

where (a) follows from the element-wise application of σ, (b) follows from Equation 3.7, and (c) results from introducing a diagonal matrix $\mathbf{P}^{(i+1)}$ with diagonal entries in $\{a, 1\}$ such that $(\mathbf{P}^{(i+1)})_{j,j} = 1$ if $\big((\mathbf{W}^{(i+1)} \otimes \mathbf{A})\mathbf{y}^{(i)}\big)_j \geq 0$, and $(\mathbf{P}^{(i+1)})_{j,j} = a$ elsewhere.

By a recursive application of Equation 3.10(c), we have

$$\mathbf{y}^{(l)} = \mathbf{P}^{(l)}(\mathbf{W}^{(l)} \otimes \mathbf{A}) \ldots \mathbf{P}^{(2)}(\mathbf{W}^{(2)} \otimes \mathbf{A})\mathbf{P}^{(1)}(\mathbf{W}^{(1)} \otimes \mathbf{A})\mathbf{x}.$$

□

We drop the transpose from $\mathbf{W}^{(i+1)}$ in order to avoid cumbersome notation. The singular values of $\mathbf{W}^{(i+1)}$ are our primary interest thereof our results still hold.

Proof of Lemma 6. Let $\mathbf{\Sigma}$ be a $n \times n$ matrix with singular value decomposition $\mathbf{\Sigma} = \mathbf{U}\mathbf{\Lambda}\mathbf{V}^T$. Inspired by the derivation for the capacity of deterministic channels introduced by (Telatar, 1999), we derive the following

$$\mathcal{I}(\mathbf{x};\mathbf{\Sigma}\mathbf{x}) = \mathcal{I}(\mathbf{x};\mathbf{U}\mathbf{\Lambda}\mathbf{V}^T\mathbf{x}) \overset{(a)}{=} \mathcal{I}(\mathbf{x};\mathbf{\Lambda}\mathbf{V}^T\mathbf{x})\mathcal{I}(\mathbf{x};\mathbf{\Sigma}\mathbf{x}) \overset{(b)}{=} \mathcal{I}(\mathbf{V}^T\mathbf{x};\mathbf{\Lambda}\mathbf{V}^T\mathbf{x}) \overset{(c)}{=} \mathcal{I}(\tilde{\mathbf{x}};\mathbf{\Lambda}\tilde{\mathbf{x}}). \tag{3.11}$$

(a) and (b) are a result of Equation 3.9(b) and that $\mathbf{U}$ and $\mathbf{V}$ are unitary hence invertible (bijective) transformations. (c) follows from the change of variables $\tilde{\mathbf{x}} = \mathbf{V}^T\mathbf{x}$.

Note that $\mathcal{I}(\tilde{\mathbf{x}}; \mathbf{\Lambda}\tilde{\mathbf{x}}) \leq \mathcal{H}(\mathbf{\Lambda}\tilde{\mathbf{x}})$. Using Equation 3.9a, we further have $\mathcal{H}(\mathbf{\Lambda}\tilde{\mathbf{y}}) \leq \mathcal{H}(\tilde{\mathbf{x}}) = \mathcal{H}(\mathbf{x})$ which completes the proof. □

We recall that we are interested in regimes where $\mathcal{I}(\mathbf{x}; \mathbf{y}^{(l)}) = 0$ and $\mathcal{L}(\mathbf{y}^{(l)}) = 0$. In Lemma 6, we show that $\mathcal{I}(\mathbf{x}; \mathbf{y}^{(l)}) = 0$ if $\max_j \lambda_j(\mathbf{P}^{(l)}(\mathbf{W}^{(l)} \otimes \mathbf{A}) \cdots \mathbf{P}^{(2)}(\mathbf{W}^{(2)} \otimes \mathbf{A})\mathbf{P}^{(1)}(\mathbf{W}^{(1)} \otimes \mathbf{A})) = 0$, and maximized (and given by $\mathcal{H}(\mathbf{x})$) when $\mathbf{P}^{(l)}(\mathbf{W}^{(l)} \otimes \mathbf{A}) \cdots \mathbf{P}^{(2)}(\mathbf{W}^{(2)} \otimes \mathbf{A})\mathbf{P}^{(1)}(\mathbf{W}^{(1)} \otimes \mathbf{A})$ is invertible. Therefore, maximum and minimum singular values of

$$\mathbf{P}^{(l)}(\mathbf{W}^{(l)} \otimes \mathbf{A}) \cdots \mathbf{P}^{(2)}(\mathbf{W}^{(2)} \otimes \mathbf{A})\mathbf{P}^{(1)}(\mathbf{W}^{(1)} \otimes \mathbf{A})$$

are of our interest.

Proof of Theorem 5. Let $\sigma_{\mathbf{A}} = \max_j \lambda_j(\mathbf{A})$ and $\sigma_{\mathbf{W}} = \sup_i \max_j \lambda_j(\mathbf{W}^{(i)})$. That is, given singular values of $\mathbf{P}^{(i)}$ is in $\{a, 1\}$, $\sup_i \max_j \lambda_j(\mathbf{P}^{(i)}(\mathbf{W}^{(i)} \otimes \mathbf{A})) = \sigma_{\mathbf{A}}\sigma_{\mathbf{W}}$. We, moreover, have

$$\max_j \lambda_j(\mathbf{P}^{(l)}(\mathbf{W}^{(l)} \otimes \mathbf{A}) \cdots \mathbf{P}^{(2)}(\mathbf{W}^{(2)} \otimes \mathbf{A})\mathbf{P}^{(1)}(\mathbf{W}^{(1)} \otimes \mathbf{A})) \leq (\sigma_{\mathbf{A}}\sigma_{\mathbf{W}})^l.$$

Therefore, if $\sigma_{\mathbf{A}}\sigma_{\mathbf{W}} < 1$, by Lemma 6 we have $\mathcal{I}(\mathbf{x}; \mathbf{y}^{(l)}) = \mathcal{O}((\sigma_{\mathbf{A}}\sigma_{\mathbf{W}})^l)$, and $\lim_{l\to\infty} \mathcal{I}(\mathbf{x}; \mathbf{y}^{(l)}) = 0$. □

Proof of Theorem 6. We now denote $\gamma_{\mathbf{A}} = \min_j \lambda_j(\mathbf{A})$ and $\gamma_{\mathbf{W}} = \inf_i \min_j \lambda_j(\mathbf{W}^{(i)})$. Hence $\inf_i \min_j \lambda_j(\mathbf{P}^{(i)}(\mathbf{W}^{(i)} \otimes \mathbf{A})) = a\gamma_{\mathbf{A}}\gamma_{\mathbf{W}}$. Moreover, $\min_j \lambda_j(\mathbf{P}^{(l)}(\mathbf{W}^{(l)} \otimes \mathbf{A}) \cdots \mathbf{P}^{(2)}(\mathbf{W}^{(2)} \otimes \mathbf{A})\mathbf{P}^{(1)}(\mathbf{W}^{(1)} \otimes \mathbf{A})) \geq (a\gamma_{\mathbf{A}}\gamma_{\mathbf{W}})^l$. If $a\gamma_{\mathbf{A}}\gamma_{\mathbf{W}} \geq 1$, $\min_j \lambda_j(\mathbf{P}_l(\mathbf{W}^{(l)} \otimes \mathbf{A}) \cdots \mathbf{P}_2(\mathbf{W}^{(2)} \otimes \mathbf{A})\mathbf{P}_1(\mathbf{W}^{(1)} \otimes \mathbf{A})) \geq 1 \; \forall l \in \mathbb{N}^+$, hence $\mathcal{I}(\mathbf{x}; \mathbf{y}^{(l)}) = \mathcal{H}(\mathbf{x})$ and $\mathcal{L}(\mathbf{y}^{(l)}) = 0$ results by Lemma 6. □

Proof of Corollary 3. Let $\mathbf{D}$ denote the degree matrix such that $(\mathbf{D})_{j,j} = \sum_m (\mathbf{A})_{j,m}$, and $\mathbf{L}$ be the associated normalized Laplacian $\mathbf{L} = \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$. Due to the property of normalized Laplacian such that $\max_j \lambda_j(\mathbf{L}) = 1$, we have $\sigma_{\mathbf{A}} = 1$. Inserting this into Theorem 5, the corollary results. □

Similarly as in Equation 3.10, $\mathbf{y}^{(i+1)}$ can be derived from Equation 3.5 as follows:

$$\begin{aligned}\mathbf{y}^{(i+1)} &= vec\big(\sigma(\sum_k \mathbf{A}_k \mathbf{Y}^{(i)} \mathbf{W}_k^{(i+1)})\big) \overset{(a)}{=} \sigma(\sum_k vec(\mathbf{A}_k \mathbf{Y}^{(i)} \mathbf{W}_k^{(i+1)})\big) \\ \mathbf{y}^{(i+1)} &\overset{(b)}{=} \sigma(\sum_k (\mathbf{W}_k^{(i+1)} \otimes \mathbf{A}_k)\mathbf{y}^{(i)}\sigma) \overset{(c)}{=} \mathbf{P}^{(i+1)} \sum_k (\mathbf{W}_k^{(i+1)} \otimes \mathbf{A}_k)\mathbf{y}^{(i)}\end{aligned} \tag{3.12}$$

where $\mathbf{P}^{(i+1)}$ is a diagonal matrix with diagonal entries in $\{a, 1\}$ with $a \in (0,1)$ such that $(\mathbf{P}^{(i)})_{j,j} = 1$ if $\big(\sum_k (\mathbf{W}_k^{(i+1)} \otimes \mathbf{A})\mathbf{y}^{(i)}\big)_j \geq 0$, and $(\mathbf{P}^{(i)})_{j,j} = a$ otherwise.

Therefore, $\mathbf{y}^{(l)}$ is given by

$$\mathbf{y}^{(l)} = \mathbf{P}^{(l)} \sum_{k_l} (\mathbf{W}_{k_l}^{(l)} \otimes \mathbf{A}_{k_l}) \cdots \mathbf{P}^{(2)} \sum_{k_2} (\mathbf{W}_{k_2}^{(2)} \otimes \mathbf{A}_{k_2}) \mathbf{P}^{(1)} \sum_{k_1} (\mathbf{W}_{k_1}^{(1)} \otimes \mathbf{A}_{k_1})\mathbf{x}.$$

Consider Equation 3.11 where $\mathbf{\Sigma}$ is replaced with $\mathbf{P}^{(l)} \sum_{k_l} (\mathbf{W}_{k_l}^{(l)} \otimes \mathbf{A}_{k_l}) \cdots \mathbf{P}^{(2)} \sum_{k_2} (\mathbf{W}_{k_2}^{(2)} \otimes \mathbf{A}_{k_2}) \mathbf{P}^{(1)} \sum_{k_1} (\mathbf{W}_{k_1}^{(1)} \otimes \mathbf{A}_{k_1})$. We deduce the followings:

Proof of Theorem 7. Suppose $\sigma^{(i)}$ denotes the largest singular value of $\mathbf{P}^{(i)} \sum_{k_i=1}^K (\mathbf{W}_{k_i}^{(i)} \otimes \mathbf{A}_{k_i})$ such that $\sigma^{(i)} = \max_j \lambda_j \big(\mathbf{P}^{(i)} \sum_{k_i} (\mathbf{W}_{k_i}^{(i)} \otimes \mathbf{A}_{k_i})\big)$. Following the same argument as in the proofs of Theorem 5 and Theorem 6, Lemma 6 implies that if $\sup_i \sigma^{(i)} < 1$, then $\mathcal{I}(\mathbf{x}; \mathbf{y}^{(l)}) = \mathcal{O}\big((\sup_i \sigma^{(i)})^l\big)$, and hence $\lim_{l\to\infty} \mathcal{I}(\mathbf{x}; \mathbf{y}^{(l)}) = 0$ results. □

Proof of Theorem 8. Let $\gamma^{(i)}$ denote the minimum singular value of $\mathbf{P}^{(i)} \sum_{k_i=1}^K (\mathbf{W}_{k_i}^{(i)} \otimes \mathbf{A}_{k_i})$ such that $\gamma^{(i)} = \min_j \lambda_j \big(\mathbf{P}^{(i)} \sum_{k_i=1}^K (\mathbf{W}_{k_i}^{(i)} \otimes \mathbf{A}_{k_i})\big)$. By Lemma 6, it immediately follows that if $\inf_i \sigma^{(i)} \geq 1$, then $\forall l \in \mathbb{N}^+$ we have $\mathcal{L}(\mathbf{y}^{(l)}) = 0$. □

Before we move on to the proofs of Corollary 4 and Corollary 5, we state the following lemma.

Lemma 7. *Let the singular value decomposition of $\mathbf{A} \in \mathbb{R}^{n\times n}$ is given by $\mathbf{A} = \mathbf{U}_\mathbf{A} \mathbf{S} \mathbf{V}_\mathbf{A}^T$ and we set each $\mathbf{A}_k$ to $\mathbf{A}_k = \mathbf{U}_\mathbf{A} \mathbf{S}_k \mathbf{V}_\mathbf{A}^T$ with $(\mathbf{S}_k)_{m,m} = \lambda_m(\mathbf{A})$ if $k = m$ and $(\mathbf{S}_k)_{m,m} = 0$ elsewhere. For such specific composition, we argue that singular values of $\sum_k \mathbf{W}_k \otimes \mathbf{A}_k$ for $\mathbf{W}_k \in \mathbb{R}^{d\times d}$ is given by $\lambda_k(\mathbf{A})\lambda_j(\mathbf{W}_k)$ for $k = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, d$.*

Proof of Lemma 7. Let the singular value decomposition of $\mathbf{W}_k$ be $\mathbf{W}_k = \mathbf{U}_{\mathbf{W}_k}\mathbf{S}_{\mathbf{W}_k}\mathbf{V}_{\mathbf{W}_k}^T$. By the property of kronecker product, we have

$$\sum_k \mathbf{W}_k \otimes \mathbf{A}_k = \sum_k (\mathbf{U}_{\mathbf{W}_k} \otimes \mathbf{U}_{\mathbf{A}})(\mathbf{S}_{\mathbf{W}_k} \otimes \mathbf{S}_k)(\mathbf{V}_{\mathbf{W}_k}^T \otimes \mathbf{V}_{\mathbf{A}}^T).$$

Next, we define a set of $nd \times nd$ mask matrices $\mathbf{M}_k$ such that $(\mathbf{M}_k)_{i,i'} = 1$ if $i = i'$ and i (hence i') is of the form $i = k + (j-1)n$ for $j = 1, 2, \ldots, d$, and $(\mathbf{M}_k)_{i,i'} = 0$ otherwise. Reminding that $(\mathbf{S}_k)_{m,m} = \lambda_m(\mathbf{A})$ if $k = m$ and $(\mathbf{S}_k)_{m,m} = 0$ elsewhere, above equation can be rewritten as

$$\sum_k \mathbf{W}_k \otimes \mathbf{A}_k = \sum_k (\mathbf{U}_{\mathbf{W}_k} \otimes \mathbf{U}_{\mathbf{A}})\mathbf{M}_k(\mathbf{S}_{\mathbf{W}_k} \otimes \mathbf{S}_k)\mathbf{M}_k(\mathbf{V}_{\mathbf{W}_k}^T \otimes \mathbf{V}_{\mathbf{A}}^T).$$

In other words, the mask matrix $\mathbf{M}_k$ applies on the columns (rows) of $\mathbf{U}_{\mathbf{W}_k} \otimes \mathbf{U}_{\mathbf{A}}$ $(\mathbf{V}_{\mathbf{W}_k}^T \otimes \mathbf{V}_{\mathbf{A}}^T)$ where the respective diagonal entries of $(\mathbf{S}_{\mathbf{W}_k} \otimes \mathbf{S}_k)$ are nonzero.

Next, we note that if $k = k'$, $\mathbf{M}_k\mathbf{M}_{k'} = \mathbf{M}_k$, and $\mathbf{M}_k$ and $\mathbf{M}_{k'}$ are orthogonal for $k \neq k'$. This leads us to

$$\begin{aligned}(\mathbf{U}_{\mathbf{W}_k} \otimes \mathbf{U}_{\mathbf{A}})\mathbf{M}_k(\mathbf{S}_{\mathbf{W}_k} \otimes \mathbf{S}_k)\mathbf{M}_k(\mathbf{V}_{\mathbf{W}_k}^T \otimes \mathbf{V}_{\mathbf{A}}^T)& \\ = \sum_{k'}(\mathbf{U}_{\mathbf{W}_{k'}} \otimes \mathbf{U}_{\mathbf{A}})\mathbf{M}_{k'}(\mathbf{S}_{\mathbf{W}_k} \otimes \mathbf{S}_k)\sum_{k''}(\mathbf{V}_{\mathbf{W}_{k''}}^T \otimes \mathbf{V}_{\mathbf{A}}^T)\mathbf{M}_{k''}.&\end{aligned}$$

By defining $\tilde{\mathbf{U}} = \sum_k (\mathbf{U}_{\mathbf{W}_k} \otimes \mathbf{U}_{\mathbf{A}})\mathbf{M}_k$ and $\tilde{\mathbf{V}} = \sum_k \mathbf{M}_k(\mathbf{V}_{\mathbf{W}_k}^T \otimes \mathbf{V}_{\mathbf{A}}^T)$ and using the above equation, we resume $\sum_k \mathbf{W}_k \otimes \mathbf{A}_k$ as

$$\sum_k \mathbf{W}_k \otimes \mathbf{A}_k = \tilde{\mathbf{U}} \sum_k (\mathbf{S}_{\mathbf{W}_k} \otimes \mathbf{S}_k)\tilde{\mathbf{V}}^T. \tag{3.13}$$

Next, we will show that $\tilde{\mathbf{U}}$ and $\tilde{\mathbf{V}}$ are unitary matrices through proving that $\tilde{\mathbf{U}}\tilde{\mathbf{U}}^T = \tilde{\mathbf{U}}^T\tilde{\mathbf{U}} = \mathbf{I}$ and $\tilde{\mathbf{V}}^T\tilde{\mathbf{V}} = \tilde{\mathbf{V}}\tilde{\mathbf{V}}^T = \mathbf{I}$. To avoid repeating the same procedure, we will only show it for $\tilde{\mathbf{U}}$, but the same result also holds for $\tilde{\mathbf{V}}$.

First, we show that $(\mathbf{A.1})$ $\tilde{\mathbf{U}}\tilde{\mathbf{U}}^T = \mathbf{I}$, and then $(\mathbf{A.2})$ $\tilde{\mathbf{U}}^T\tilde{\mathbf{U}} = \mathbf{I}$ to argue that $\tilde{\mathbf{U}}$ (and $\tilde{\mathbf{V}}$) is unitary.

($\mathbf{A}.\mathbf{1}$) We can simplify $\tilde{\mathbf{U}}\tilde{\mathbf{U}}^T$ as

$$\begin{aligned}
\tilde{\mathbf{U}}\tilde{\mathbf{U}}^T &= \sum_k \big((\mathbf{U}_{\mathbf{W}_k} \otimes \mathbf{U}_{\mathbf{A}})\mathbf{M}_k\big) \sum_{k'} \big((\mathbf{U}_{\mathbf{W}_{k'}} \otimes \mathbf{U}_{\mathbf{A}})\mathbf{M}_{k'}\big)^T \\
\tilde{\mathbf{U}}\tilde{\mathbf{U}}^T &= \sum_{k,k'} \big((\mathbf{U}_{\mathbf{W}_k} \otimes \mathbf{U}_{\mathbf{A}})\mathbf{M}_k\big)\big((\mathbf{U}_{\mathbf{W}_{k'}} \otimes \mathbf{U}_{\mathbf{A}})\mathbf{M}_{k'}\big)^T \\
\tilde{\mathbf{U}}\tilde{\mathbf{U}}^T &\overset{(a)}{=} \sum_k \big((\mathbf{U}_{\mathbf{W}_k} \otimes \mathbf{U}_{\mathbf{A}})\mathbf{M}_k\big)\big((\mathbf{U}_{\mathbf{W}_k} \otimes \mathbf{U}_{\mathbf{A}})\mathbf{M}_k\big)^T
\end{aligned} \tag{3.14}$$

where (a) follows from the orthogonality of $\mathbf{M}_k$ and $\mathbf{M}_{k'}$ for $k \neq k'$.

We will now take a closer look at $\sum_k \big((\mathbf{U}_{\mathbf{W}_k} \otimes \mathbf{U}_{\mathbf{A}})\mathbf{M}_k\big)\big((\mathbf{U}_{\mathbf{W}_k} \otimes \mathbf{U}_{\mathbf{A}})\mathbf{M}_k\big)^T$. The entries of summands, $\big((\mathbf{U}_{\mathbf{W}_k} \otimes \mathbf{U}_{\mathbf{A}})\mathbf{M}_k\big)\big((\mathbf{U}_{\mathbf{W}_k} \otimes \mathbf{U}_{\mathbf{A}})\mathbf{M}_k\big)^T$, are equivalent to inner product between the rows of $(\mathbf{U}_{\mathbf{W}_k} \otimes \mathbf{U}_{\mathbf{A}})\mathbf{M}_k$ for a fixed k. Recall that for a fixed k, the mask matrix satisfies $(\mathbf{M}_k)_{i,i} = 1$ if k is of the form $i = k + (j-1)n$ for $j = 1, 2, \cdots, d$, and $(\mathbf{M}_k)_{i,i} = 0$ elsewhere. We now define i_ω and i_α as indices such that $i_\omega = \lfloor i/n \rfloor + 1$ and $i_\alpha = \mod(i, \lfloor i/n \rfloor)$. Similarly, let $i'_\omega = \lfloor i'/n \rfloor + 1$ and $i'_\alpha = \mod(i', \lfloor i'/n \rfloor)$.

Following above definitions, a moment of thought reveals that the nonzero entries of ith row of $\big((\mathbf{U}_{\mathbf{W}_k} \otimes \mathbf{U}_{\mathbf{A}})\mathbf{M}_k\big)$ is given by $(\mathbf{U}_{\mathbf{W}_k})_{i_\omega,[m=1,2,\ldots,d]}(\mathbf{U}_{\mathbf{A}})_{i_\alpha,k}$. We therefore investigate $(\tilde{\mathbf{U}}\tilde{\mathbf{U}}^T)_{i,i'}$ i.e., the inner product between ith and i'th rows of $\big((\mathbf{U}_{\mathbf{W}_k} \otimes \mathbf{U}_{\mathbf{A}})\mathbf{M}_k\big)$ summed over all $k = 1, 2, \ldots, n$. To start, the inner product between ith and i'th rows of $\big((\mathbf{U}_{\mathbf{W}_k} \otimes \mathbf{U}_{\mathbf{A}})\mathbf{M}_k\big)$ is as follows

$$\begin{aligned}
&\langle [(\mathbf{U}_{\mathbf{W}_k})_{i_\omega,[m=1,2,\ldots,d]}(\mathbf{U}_{\mathbf{A}})_{i_\alpha,k}], [(\mathbf{U}_{\mathbf{W}_k})_{i'_\omega,[m=1,2,\ldots,d]}(\mathbf{U}_{\mathbf{A}})_{i'_\alpha,k}] \rangle \\
&= \sum_m (\mathbf{U}_{\mathbf{W}_k})_{i_\omega,m}(\mathbf{U}_{\mathbf{A}})_{i_\alpha,k}(\mathbf{U}_{\mathbf{W}_k})_{i'_\omega,m}(\mathbf{U}_{\mathbf{A}})_{i'_\alpha,k} \\
&= \sum_m (\mathbf{U}_{\mathbf{W}_k})_{i_\omega,m}(\mathbf{U}_{\mathbf{W}_k})_{i'_\omega,m}(\mathbf{U}_{\mathbf{A}})_{i_\alpha,k}(\mathbf{U}_{\mathbf{A}})_{i'_\alpha,k} \\
&= (\mathbf{U}_{\mathbf{A}})_{i_\alpha,k}(\mathbf{U}_{\mathbf{A}})_{i'_\alpha,k} \sum_m (\mathbf{U}_{\mathbf{W}_k})_{i_\omega,m}(\mathbf{U}_{\mathbf{W}_k})_{i'_\omega,m}.
\end{aligned} \tag{3.15}$$

Let now analyze the cases when (1) $i \neq i'$, and (2) $i = i'$. Assume (1). If further $i_\omega \neq i'_\omega$, it is immediate that

$$\sum_m (\mathbf{U}_{\mathbf{W}_k})_{i_\omega,m}(\mathbf{U}_{\mathbf{W}_k})_{i'_\omega,m} = 0$$

by the fact that $\mathbf{U}_{\mathbf{W}_k}$ is unitary, hence

$$\langle [(\mathbf{U}_{\mathbf{W}_k})_{i_\omega,[m=1,2,\ldots,d]}(\mathbf{U}_{\mathbf{A}})_{i_\alpha,k}], [(\mathbf{U}_{\mathbf{W}_k})_{i'_\omega,[m=1,2,\ldots,d]}(\mathbf{U}_{\mathbf{A}})_{i'_\alpha,k}]\rangle = 0$$

For (1), if $i_\omega = i'_\omega$, we have $i_\alpha \neq i'_\alpha$. Further, $\sum_m (\mathbf{U}_{\mathbf{W}_k})_{i_\omega,m}(\mathbf{U}_{\mathbf{W}_k})_{i'_\omega,m} = 1$ and hence

$$\begin{aligned}&\langle [(\mathbf{U}_{\mathbf{W}_k})_{i_\omega,[m=1,2,\ldots,d]}(\mathbf{U}_{\mathbf{A}})_{i_\alpha,k}], [(\mathbf{U}_{\mathbf{W}_k})_{i'_\omega,[m=1,2,\ldots,d]}(\mathbf{U}_{\mathbf{A}})_{i'_\alpha,k}]\rangle \\ &\qquad = (\mathbf{U}_{\mathbf{A}})_{i_\alpha,k}(\mathbf{U}_{\mathbf{A}})_{i'_\alpha,k}\sum_m (\mathbf{U}_{\mathbf{W}_k})_{i_\omega,m}(\mathbf{U}_{\mathbf{W}_k})_{i'_\omega,m} \\ &\qquad = (\mathbf{U}_{\mathbf{A}})_{i_\alpha,k}(\mathbf{U}_{\mathbf{A}})_{i'_\alpha,k}.\end{aligned} \tag{3.16}$$

Hence, the inner product between ith and i'th rows of $\big((\mathbf{U}_{\mathbf{W}_k} \otimes \mathbf{U}_{\mathbf{A}})\mathbf{M}_k\big)$ is given by $(\mathbf{U}_{\mathbf{A}})_{i_\alpha,k}(\mathbf{U}_{\mathbf{A}})_{i'_\alpha,k}$. Recalling Equation 3.14, we have $(\tilde{\mathbf{U}}\tilde{\mathbf{U}}^T)_{i,i'} = \sum_k (\mathbf{U}_{\mathbf{A}})_{i_\alpha,k}(\mathbf{U}_{\mathbf{A}})_{i'_\alpha,k}$. As previously mentioned we have $i_\alpha \neq i'_\alpha$. By the unitary property of $\mathbf{U}_{\mathbf{A}}$, we further have $(\tilde{\mathbf{U}}\tilde{\mathbf{U}}^T)_{i,i'} = \sum_k (\mathbf{U}_{\mathbf{A}})_{i_\alpha,k}(\mathbf{U}_{\mathbf{A}})_{i'_\alpha,k} = 0$.

So far we have shown that $(\tilde{\mathbf{U}}\tilde{\mathbf{U}}^T)_{i,i'} = 0$ when $i \neq i'$. Let now $i = i'$, i.e., (2). IT follows from Equation 3.15 that

$$(\tilde{\mathbf{U}}\tilde{\mathbf{U}}^T)_{i,i} \overset{(a)}{=} \sum_k (\mathbf{U}_{\mathbf{A}})^2_{i_\alpha,k}\sum_m (\mathbf{U}_{\mathbf{W}_k})^2_{i_\omega,m}(\tilde{\mathbf{U}}\tilde{\mathbf{U}}^T)_{i,i} \overset{(b)}{=} \sum_k (\mathbf{U}_{\mathbf{A}})^2_{i_\alpha,k}1(\tilde{\mathbf{U}}\tilde{\mathbf{U}}^T)_{i,i} \overset{(c)}{=} 1 \tag{3.17}$$

where (a) results from that $\mathbf{U}_{\mathbf{W}_k}$ is unitary, and (b) follows from that $\mathbf{U}_{\mathbf{A}}$ is unitary. Combining above arguments and Equation 3.17, we have $\tilde{\mathbf{U}}\tilde{\mathbf{U}}^T = \mathbf{I}$.

(**A.2**) Next, we show that $\tilde{\mathbf{U}}^T\tilde{\mathbf{U}} = \mathbf{I}$. We begin with

$$\begin{aligned}\tilde{\mathbf{U}}^T\tilde{\mathbf{U}} &= \sum_k \big((\mathbf{U}_{\mathbf{W}_k} \otimes \mathbf{U}_{\mathbf{A}})\mathbf{M}_k\big)^T\big(\sum_{k'}(\mathbf{U}_{\mathbf{W}_{k'}} \otimes \mathbf{U}_{\mathbf{A}})\mathbf{M}_{k'}\big)\tilde{\mathbf{U}}^T\tilde{\mathbf{U}} \\ &= \sum_{k,k'} \big((\mathbf{U}_{\mathbf{W}_k} \otimes \mathbf{U}_{\mathbf{A}})\mathbf{M}_k\big)^T\big((\mathbf{U}_{\mathbf{W}_{k'}} \otimes \mathbf{U}_{\mathbf{A}})\mathbf{M}_{k'}\big).\end{aligned} \tag{3.18}$$

For $k \neq k'$,

$$\begin{aligned}&\Big(\big((\mathbf{U}_{\mathbf{W}_k} \otimes \mathbf{U}_{\mathbf{A}})\mathbf{M}_k\big)^T\big((\mathbf{U}_{\mathbf{W}_{k'}} \otimes \mathbf{U}_{\mathbf{A}})\mathbf{M}_{k'}\big)\Big)_{i,i'} \\ &\quad = \Big\langle \big((\mathbf{U}_{\mathbf{W}_k} \otimes \mathbf{U}_{\mathbf{A}})\mathbf{M}_k\big)_{i'}, \big((\mathbf{U}_{\mathbf{W}_{k'}} \otimes \mathbf{U}_{\mathbf{A}})\mathbf{M}_{k'}\big)_{i'}\Big\rangle.\end{aligned} \tag{3.19}$$

Note that, due to the orthogonality of $\mathbf{M}_k$ and $\mathbf{M}_k$ for $k \neq k'$, we further have $\langle \big((\mathbf{U}_{\mathbf{W}_k} \otimes \mathbf{U}_{\mathbf{A}})\mathbf{M}_k\big)_{i'}, \big((\mathbf{U}_{\mathbf{W}_{k'}} \otimes \mathbf{U}_{\mathbf{A}})\mathbf{M}_{k'}\big)_{i'} \rangle = 0$ for $i \neq i'$. When $i = i'$, on the other hand, we have

$$\begin{aligned}
&\Big(\big((\mathbf{U}_{\mathbf{W}_k} \otimes \mathbf{U}_{\mathbf{A}})\mathbf{M}_k\big)^T \big((\mathbf{U}_{\mathbf{W}_{k'}} \otimes \mathbf{U}_{\mathbf{A}})\mathbf{M}_{k'}\big)\Big)_{i,i'} = \langle \big((\mathbf{U}_{\mathbf{W}_k} \otimes \mathbf{U}_{\mathbf{A}})\mathbf{M}_k\big)_{i'}, \big((\mathbf{U}_{\mathbf{W}_{k'}} \otimes \mathbf{U}_{\mathbf{A}})\mathbf{M}_{k'}\big)_{i} \rangle \\
&\overset{(a)}{=} \langle (\mathbf{U}_{\mathbf{W}_k})_{[z=1,\cdots,d],i_\omega} (\mathbf{U}_{\mathbf{A}})_{[w=1,\cdots,n],k'} (\mathbf{U}_{\mathbf{W}_{k'}})_{[z=1,\cdots,d],i_\omega} (\mathbf{U}_{\mathbf{A}})_{[w=1,\cdots,n],k'} \rangle \\
&= \sum_w \sum_d (\mathbf{U}_{\mathbf{W}_k})_{z,i_\omega} (\mathbf{U}_{\mathbf{A}})_{w,k} (\mathbf{U}_{\mathbf{W}_{k'}})_{z,i_\omega} (\mathbf{U}_{\mathbf{A}})_{w,k'} \\
&\overset{(b)}{=} \sum_d (\mathbf{U}_{\mathbf{W}_k})_{z,i_\omega} (\mathbf{U}_{\mathbf{W}_{k'}})_{z,i_\omega} \sum_w (\mathbf{U}_{\mathbf{A}})_{w,k} (\mathbf{U}_{\mathbf{A}})_{w,k'} \\
&= 0
\end{aligned} \tag{3.20}$$

where (a) follows from that $\big((\mathbf{U}_{\mathbf{W}_k} \otimes \mathbf{U}_{\mathbf{A}})\mathbf{M}_k\big)_i = (\mathbf{U}_{\mathbf{W}_k})_{[z=1,\cdots,d],i_\omega} (\mathbf{U}_{\mathbf{A}})_{[w=1,\cdots,n],k}$and (b) results from that $\sum_w (\mathbf{U}_{\mathbf{A}})_{w,k} (\mathbf{U}_{\mathbf{A}})_{w,k'} = 0$ for $k \neq k'$ as $\mathbf{U}_{\mathbf{A}}$ is unitary.

Therefore, Equation 3.18 can be resumed as

$$\begin{aligned}
\tilde{\mathbf{U}}^T \tilde{\mathbf{U}} &= \sum_k \big((\mathbf{U}_{\mathbf{W}_k} \otimes \mathbf{U}_{\mathbf{A}})\mathbf{M}_k\big)^T \big((\mathbf{U}_{\mathbf{W}_k} \otimes \mathbf{U}_{\mathbf{A}})\mathbf{M}_k\big) \\
\tilde{\mathbf{U}}^T \tilde{\mathbf{U}} &= \sum_k \mathbf{M}_k (\mathbf{U}_{\mathbf{W}_k} \otimes \mathbf{U}_{\mathbf{A}})^T (\mathbf{U}_{\mathbf{W}_k} \otimes \mathbf{U}_{\mathbf{A}}) \mathbf{M}_k \\
\tilde{\mathbf{U}}^T \tilde{\mathbf{U}} &\overset{(a)}{=} \sum_k \mathbf{M}_k \mathbf{I} \mathbf{M}_k = \sum_k \mathbf{M}_k \overset{(b)}{=} \mathbf{I}
\end{aligned}$$

where (a) follows from that the kronecker product of unitary matrices is also unitary, hence $(\mathbf{U}_{\mathbf{W}_k} \otimes \mathbf{U}_{\mathbf{A}})$ is unitary, and (b) follows from the definition of $\mathbf{M}_k$.

As the last step, recall from Equation 3.13 that $\sum_k \mathbf{W}_k \otimes \mathbf{A}_k = \tilde{\mathbf{U}} \sum_k (\mathbf{S}_{\mathbf{W}_k} \otimes \mathbf{S}_k) \tilde{\mathbf{V}}^T$, and note by the definition of $\mathbf{S}_k$ that $(\mathbf{S}_{\mathbf{W}_k} \otimes \mathbf{S}_k)_{i,i'} = \lambda_k(\mathbf{A})\lambda_j(\mathbf{S}_{\mathbf{W}_k})$ if $i = i'$ and i, hence i', of the form $i = k + (j-1)n$ for $j = 1, 2, \cdots, d$, and $(\mathbf{S}_{\mathbf{W}_k} \otimes \mathbf{S}_k)_{i,i'} = 0$ elsewhere. Therefore, by the fact that $(\mathbf{S}_{\mathbf{W}_k} \otimes \mathbf{S}_k)(\mathbf{S}_{\mathbf{W}_{k'}} \otimes \mathbf{S}_{k'}) = 0$ for $k \neq k'$, it follows that $\sum_k (\mathbf{S}_{\mathbf{W}_k} \otimes \mathbf{S}_k)$ is a diagonal matrix with diagonal entries $\lambda_k(\mathbf{A})\lambda_j(\mathbf{S}_{\mathbf{W}_k})$ where $j = 1, 2, \cdots, d$ and $k = 1, 2, \cdots, n$, which completes the proof. □

For the decomposition of $\mathbf{A}$ such that $\mathbf{A}_k = \mathbf{U_A S}_k \mathbf{V}_\mathbf{A}^T$ where the singular value decomposition of $\mathbf{A}$ is given by $\mathbf{A} = \mathbf{U_A S V}_\mathbf{A}^T$, we recall Theorem 7 and Theorem 8 to conclude Corollary 4 and Corollary 5 as follows.

Proof of Corollary 4. Let $\sigma_{\mathbf{A}_k} = \lambda_k(\mathbf{A})$ and $\sigma_{\mathbf{W}_k} = \sup_i \max_j \lambda_j(\mathbf{W}_k^{(i)})$. By Lemma 7, we have $\max_j \lambda_j(\sum_k(\mathbf{W}_k^{(i)} \otimes \mathbf{A}_k)) \leq \max_k \sigma_{\mathbf{A}_k}\sigma_{\mathbf{W}_k}$. Noting that $\mathbf{P}^{(i)}$ is diagonal with

entries at most 1, we have $\max_j \lambda_j\big(\mathbf{P}^{(l)} \sum_{k_l}(\mathbf{W}_{k_l}^{(l)} \otimes \mathbf{A}_{k_l}) \cdots$ $\mathbf{P}^{(2)} \sum_{k_2}(\mathbf{W}_{k_2}^{(2)} \otimes \mathbf{A}_{k_2})\mathbf{P}^{(1)} \sum_{k_1}(\mathbf{W}_{k_1}^{(1)} \otimes \mathbf{A}_{k_1})\big) \leq (\max_k \sigma_{\mathbf{A}_k}\sigma_{\mathbf{W}_k})^l$. Therefore, if $\underline{\forall k = \{1, 2, \ldots, n\}}$ $\sigma_{\mathbf{A}_k}\sigma_{\mathbf{W}_k} < 1$, then $\lim_{l\to\infty} \max_j \lambda_j\big(\sum_k(\mathbf{W}_k^{(i)} \otimes \mathbf{A}_k)\big) = 0$. Hence $\lim_{l\to\infty} \mathcal{I}(\mathbf{x}; \mathbf{y}^{(l)}) = 0$ results by Lemma 6. □

Proof of Corollary 5. Let $\gamma_{\mathbf{W}_k} = \inf_i \min_j \lambda_j(\mathbf{W}_k^{(i)})$. Note that $\min_j \lambda_j\big(\mathbf{P}^{(i)} \sum_k \mathbf{W}_k^{(i)} \otimes \mathbf{A}_k\big) \geq a \min_k \lambda_k(\mathbf{A})\gamma_{\mathbf{W}_k}$ by Lemma Lemma 7 and that $\min_j \lambda_j(\mathbf{P}^i) = a$. Moreover, $\min_j \lambda_j\big(\mathbf{P}^{(l)} \sum_{k_l}(\mathbf{W}_{k_l}^{(l)} \otimes \mathbf{A}_{k_l}) \cdots \mathbf{P}^{(2)} \sum_{k_2}(\mathbf{W}_{k_2}^{(2)} \otimes \mathbf{A}_{k_2})\mathbf{P}^{(1)} \sum_{k_1}(\mathbf{W}_{k_1}^{(1)} \otimes \mathbf{A}_{k_1})\big) \geq (a \min_k \lambda_k(\mathbf{A})\gamma_{\mathbf{W}_k})^l$. Therefore, if $a\sigma_{\mathbf{A}_k}\gamma_{\mathbf{W}_k} \geq 1$, $\underline{\forall k \in \{1, 2, \ldots, n\}}$, then $\mathcal{I}(\mathbf{x}; \mathbf{y}^{(l)}) = \mathcal{H}(\mathbf{x})\ \forall l \in \mathbb{N}^+$ by Lemma 6, hence $\mathcal{L}(\mathbf{y}^{(l)}) = 0$. □

3.3.6 *Connectivity Aware Graph Decomposition*

The question – *"Is there an optimal decomposition strategy for my graph-structured data?"* has, beyond doubt, a non-trivial answer. In fact, there is probably no universal strategy on how to perform graph decomposition for an arbitrary graph-structured data. Yet, there seems to be common principles that impact the *goodness* of a decomposition. For instance, our theoretical analysis hinted that node feature information is to be preserved along with layers of GNN. In this section, we will briefly study a *connectivity-aware* graph decomposition strategy that utilizes our theoretical analysis and demonstrate its effectiveness through numerical experiments. The Connectivity Aware Graph Decomposition framework is a joint work with the co-authors of (Xupeng Miao, Nezihe Merve Gürel[†],Wentao Zhang et al., 2021). We review the framework below for completeness and refer to the publication for detailed explanations and experiments.

As illustrated in Section 3.3.2 that the random decomposition performs poorly compared to a non-random decomposition. Moreover, the random decomposition of the adjacency matrix $\mathbf{A}$ into K components may result in a) overfitting when K is large and b) graph disconnectivity impeding the information flow as GCN relies on the graph structures to propagate the node features and labels along the edges. Inspired by that, we propose DeGNN to automatically perform decomposition on graph structured data. It takes graph connectivity into account and utilize spanning tree structure for preserving the accessibility of the nodes. Such a connectivity aware decomposition, DeGNN, is shown to outperform existing variants, such as GCN, JK-Net, ResGCN, DenseGCN, on the semi-supervised node classification task and on several datasets including Cora, Citeseer, Pubmed, Flickr and Reddit. We refer to (Xupeng Miao, Nezihe Merve Gürel[†],Wentao Zhang et al., 2021) for further details.

3.4 SUMMARY

We introduced the problem of label-efficient model selection for pretrained models, and presented a stream-based approach to selectively query the labels of instances that are informative for ranking pretrained models and to sequentially predict unseen labels. Our framework is generic, easy to implement, and applies across various classification tasks. We illustrated the effectiveness of our method on several well-studied ML benchmarks. Our efforts in this part is followed by an investigation of the importance of graph decomposition in graph neural networks. We theoretically analyzed how graph decomposition can avoid the information loss problem caused by increasing networks depth. To utilize the information preserving ability of the decomposition in general graph-structured data, we briefly introduced a novel connectivity-aware graph decomposition to balance the trade-off between depth and information loss.

In the next part, we will go beyond accuracy and move our focus from efficiently achieving high accuracy results to efficiently achieving adversarial robustness.

Part III

ROBUSTNESS VIA KNOWLEDGE INTEGRATION

4 KNOWLEDGE ENHANCED ADVERSARIAL ROBUSTNESS

Artificial intelligence systems need the wisdom to know when to take advice from us and when to learn from data.

— Subbarao Kambhampati (Polanyi's Revenge and AI's New Romance with Tacit Knowledge, 2021)

4.1 OVERVIEW

The ability to design efficient adversarial defenses that are effective across all phases of an end-to-end ML pipeline is tied to understanding how adversarial perturbations mislead the ML models to make arbitrarily incorrect predictions even when these perturbations are imperceptibly small. In this spirit, the anatomical analysis of perturbed data has led to the creation of the *robust features* concept (Ilyas et al., 2019), where the features that fail to manipulate the predictions without human-recognition are identified as robust. This human-centric phenomenon has lately spawned some interest in supplying human-perceptible features into the defense mechanisms, including edge feature utilization (Sun et al., 2021) and using domain knowledge to detect adversaries (Melacci et al., 2021).

In this part, motivated by this human-centric perspective of adversarial examples, we aim at improving ML robustness across the entire ML pipeline in an effective and affordable manner by integrating *domain knowledge*. Towards that, we will first take stop sign recognition as a simple example to illustrate the potential role of knowledge in ML prediction. In this example, the **main task** is to predict whether a stop sign appears in the input image. Training a DNN model for this task is known to be vulnerable against a range of adversarial attacks (Bielik et al., 2020; Eykholt et al., 2018; Xiao et al., 2018a). However, if, in addition to such a DNN model, we could (1) build a detector

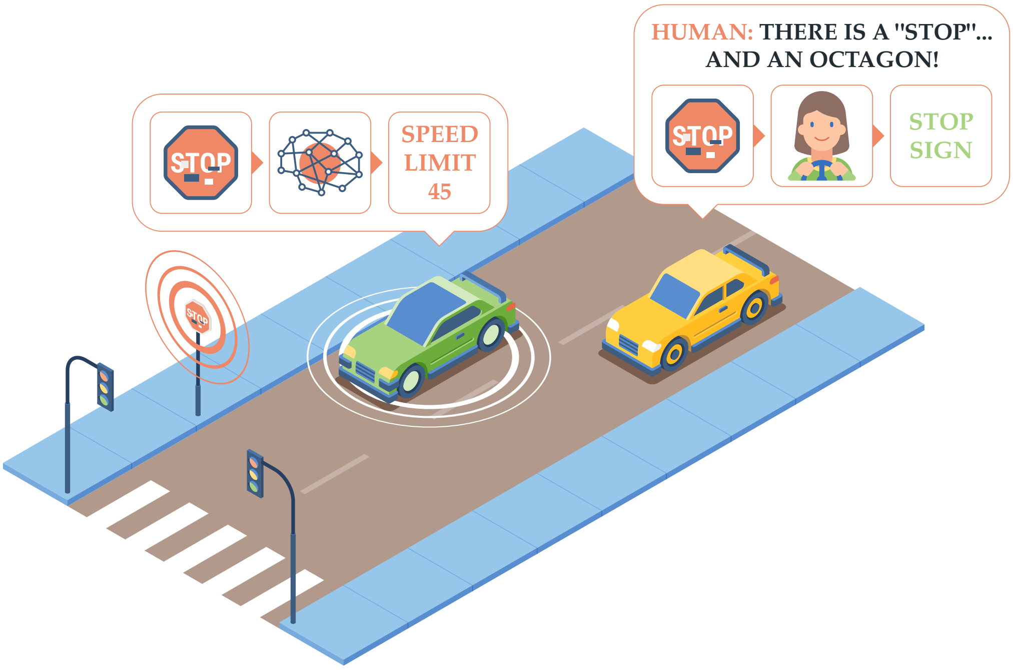


Figure 4.1: Illustration showing how supplying domain knowledge can help improve ML robustness. Training a DNN model for road sign classification is known to be vulnerable against a range of adversarial attacks (Eykholt et al., 2018; Xiao et al., 2018a). If we could build a shape and pattern detector, and integrate the domain knowledge such that *"A stop sign should be of an octagon shape"* and *"There must be a STOP on it"*, it is possible that additional information could enable the ML system to detect or defend against attacks, which lead to conflicts between the DNN prediction and domain knowledge.

for a different **auxiliary task**, for example, detecting whether an octagon appears in the input by using other learning strategies such as traditional computer vision techniques, and (2) integrate the **domain knowledge** such that *"A stop sign should be of an octagon shape"*, it is possible that this additional information could enable the ML system to detect or defend against attacks, which lead to conflicts between the DNN prediction and domain knowledge. For instance, if a speed limit sign with *rectangle* shape is misrecognized as a stop sign, the ML system would identify this conflict and try to correct the prediction (see Figure 4.1 for an illustration).

Inspired by this intuition, we aim to understand how to *enhance the robustness of ML models via domain knowledge integration*. Despite the natural intuition in the previous simple example, providing a technically rigorous treatment to

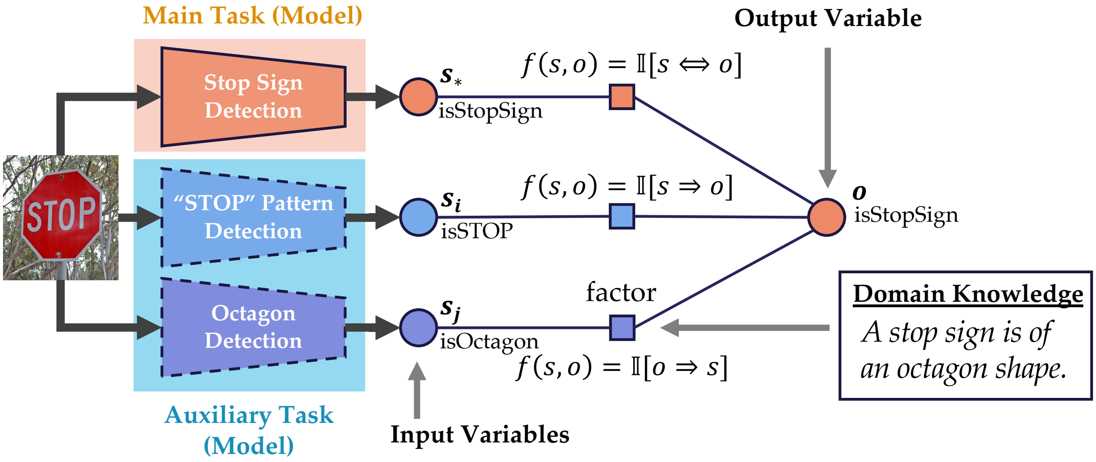


Figure 4.2: An overview of the KEMLP framework. KEMLP constructs a factor graph by modeling the output of ML models as random input variables, and the KEMLP prediction as a random output variable. It integrates domain knowledge via factors connecting different random variables.

this problem is far from trivial, yielding the following questions: *How should we integrate domain knowledge in a principled way? When will integrating domain knowledge help with robustness and will there be a trade-off between robustness and clean accuracy? Can integration of domain knowledge genuinely bring additional robustness benefits against practical attacks when compared with state-of-the-art defenses?*

We propose the Knowledge Enhanced Machine Learning Pipeline (KEMLP), a framework that facilitates the integration of *domain knowledge* in order to improve the robustness of ML models. Figure 4.2 illustrates the KEMLP framework. In KEMLP, the outputs of different ML models are modeled as random input variables, whereas the output of KEMLP is modeled as another variable. To integrate domain knowledge, KEMLP introduces corresponding factors connecting these random variables. For example, as illustrated in Figure 4.2, the knowledge rule "*A stop sign is of an octagon shape*" introduces a factor between the input variable (i.e., the output of the octagon detector) and the output variable (i.e., output of the stop sign detector) with a factor function that *the former implies the latter*. To make predictions, KEMLP runs statistical inference over the factor graph constructed by integrating all such domain knowledge expressed as first-order logic rules, and output the marginal probability of the output variable.

Based on KEMLP, our main goal is to understand two fundamental questions: (1) *What type of knowledge is needed to improve the robustness of the joint inference results from KEMLP, and can we prove it?* (2) *Can we show that knowledge integration in the KEMLP framework can provide significant robustness gain over powerful state-of-the-art models?*

We conduct a theoretical analysis to understand the first question, focusing on two specific types of knowledge rules: (1) *permissive knowledge* of the form "$B \implies A$", and (2) *preventive knowledge* of the form "$A \implies B$", where A represents the main task, B an auxiliary task and $\implies$ denotes logical implication. We focus on the *weighted robust accuracy*, which is a weighted average of accuracies on benign and adversarial examples, respectively, and we derive sufficient conditions under which KEMLP outperforms the main task model alone. Under mild conditions, we show that integrating multiple weak auxiliary models, both in their robustness and quality, together with the permissive and preventive rules, the weighted robust accuracy of KEMLP can be guaranteed to improve over the single main task model. To our best knowledge, this is the first analysis of the proposed form, focusing on the intersection of knowledge integration, joint inference, and robustness.

We then conduct extensive empirical studies to understand the second question. We focus on the road sign classification task and consider the state-of-the-art adversarial training models based on both the $\mathcal{L}_p$ bounded perturbation and occlusion perturbations (Wu, Tong, and Vorobeychik, 2019) as our baselines as well as the main task model. We will show that by training weak auxiliary models for recognizing the shapes and contents of road signs, together with the corresponding knowledge rules as illustrated in Figure 4.2, KEMLP achieves significant improvements on their robustness compared with baseline main task models against a *diverse* set of adversarial attacks while maintaining similar or even higher clean accuracy, given its improvement on the tradeoff between clean accuracy and robustness. In particular, we consider existing physical attacks (Eykholt et al., 2018), $\mathcal{L}_p$ bounded attacks (Madry et al., 2017), unforeseen attacks (Kang et al., 2019), and common corruptions (Hendrycks and Dietterich, 2019), under both whitebox and blackbox settings. To our best knowledge, KEMLP is the first ML model robust to diverse attacks in practice with high clean accuracy.

4.2 RELATED WORK

In the following, we review several bodies of literature that are relevant to the objective of our part.

ADVERSARIAL EXAMPLES are carefully crafted inputs aiming to mislead well-trained ML models (Goodfellow, Shlens, and Szegedy, 2015; Szegedy et al., 2013). A variety of approaches to generate such adversarial examples have also been proposed based on different perturbation measurement metrics, including $\mathcal{L}_p$ bounded, unrestricted, and physical attacks (Bhattad et al., 2020; Eykholt et al., 2018; Wong, Schmidt, and Kolter, 2019; Xiao et al., 2018b,c).

DEFENSE METHODS against such attacks have been proposed. Empirically, *adversarial training* (Madry et al., 2017) has shown to be effective, together with feature quantization (Xu, Evans, and Qi, 2017) and reconstruction approaches (Samangouei, Kabkab, and Chellappa, 2018). Certified robustness has also been studied by propagating the interval bound of a NN (Gowal et al., 2018), by leveraging the differentiable abstract interpretation (Mirman, Gehr, and Vechev, 2018; Singh et al., 2018a), or randomized smoothing of a given model (Cohen, Rosenfeld, and Kolter, 2019). Several approaches have further improved it: by incorporating it with adversarial training (Balunovic and Vechev, 2020), for generative neural networks (Mirman et al., 2021), by choosing different smoothing distributions for different L_p norms (Dvijotham et al., 2020; Yang et al., 2020; Zhang et al., 2020), or training more robust smoothed classifiers via data augmentation (Cohen, Rosenfeld, and Kolter, 2019), unlabeled data (Carmon et al., 2019), adversarial training (Salman et al., 2019), and regularization (Li et al., 2019a; Zhai et al., 2019). While most prior defenses focus on leveraging statistical properties of an ML model to improve its robustness, they can only be robust towards a specific type of attack, such as ℓ_p bounded attacks. Our work aims to explore how to utilize knowledge inference information to improve the robustness of a logically connected ML pipeline against a diverse set of attacks.

JOINT INFERENCE has been studied to take multiple predictions made by different models, together with the relations among them, to make a final prediction (Biba, Ferilli, and Esposito, 2011; Chakrabarti et al., 2014; Chen et al., 2014; Deng et al., 2014; McCallum, 2009; Poon and Domingos, 2007; Xu et al., 2020). These approaches usually use different inference models, such as factor graphs (Wainwright and Jordan, 2008), Markov logic networks (Richardson and Domingos, 2006) and Bayesian networks (Neuberg, 2003), as a way to characterize their relationships. The programmatic weak supervision approaches (Ratner et al., 2017, 2016) also perform joint inference by employing labeling functions and using generative modeling techniques, which aims to create noisy training data. In this work, we take a different perspective on this problem — we explore the potential of using joint inference with the objective of integrating domain knowledge and to eventually improving the ML robustness. As we will see, by integrating domain knowledge, it is possible to improve the learning robustness by a wide margin.

4.3 KNOWLEDGE ENHANCED MACHINE LEARNING PIPELINE

We first present the proposed framework Knowledge Enhanced Machine Learning Pipeline (KEMLP), which aims to improve the robustness of an ML model by integrating a diverse set of domain knowledge. In this section, we formally define the KEMLP framework.

NOTATION We consider a classification problem under a supervised learning setting, defined on a feature space $\mathcal{X}$ and a finite label space $\mathcal{Y}$. We refer to $x \in \mathcal{X}$ as an input and $y \in \mathcal{Y}$ as the target variable. An input x can be a benign example or an adversarial example. To model this, we use $z \in \{0,1\}$, a latent variable that is not exposed to KEMLP. That is, x is an adversarial example with $(x,y) \sim \mathcal{D}_a$ whenever $z = 1$, and $(x,y) \sim \mathcal{D}_b$ otherwise, where $\mathcal{D}_a$ and $\mathcal{D}_b$ represent the adversarial and benign data distributions. We let

$\pi_{\mathcal{D}_a} = \mathbb{P}(z = 1)$ and $\pi_{\mathcal{D}_b} = \mathbb{P}(z = 0)$, implying $\pi_{\mathcal{D}_a} + \pi_{\mathcal{D}_b} = 1$. For convenience, we denote $\mathbb{P}_{\mathcal{D}_a}(x, y) = \mathbb{P}(x, y|z = 1)$ and $\mathbb{P}_{\mathcal{D}_b}(x, y) = \mathbb{P}(x, y|z = 0)$. In the following, to ease the exposition, we slightly abuse the notation and use probability densities for discrete distributions.

Given an input x whose corresponding z is unknown (benign or adversarial), KEMLP aims to predict the target variable y by employing a set of *models*. These predictive models are constructed, say, using ML or some other traditional rule-based methods (e.g., edge detector). For simplicity, we describe the KEMLP framework as a binary classification task, in which case $\mathcal{Y} = \{0, 1\}$, noting that the multi-class scenario is a simple extension of it. We introduce the KEMLP framework as follows.

MODELS Models are a collection of predictive ML models, each of which takes as input x and outputs some predictions. In KEMLP, we distinguish three different type of models.

- *Main task model*: We call the (untrusted) ML model whose robustness users want to enhance as the *main task model*, denoting its predictions by $s_* \in \mathcal{Y}$.

- *Permissive models*: Let $s_{\mathcal{I}} = \{s_i\colon i \in \mathcal{I}\}$ be a set of m permissive models, each of which corresponds to the prediction of one ML model. Conceptually, permissive models are usually designed for specific events which are *sufficient* for inferring $y = 1$: $s_i \implies y$.

- *preventive models*: Similarly, we have n preventive models: $s_{\mathcal{J}} = \{s_j\colon j \in \mathcal{J}\}$, each of which corresponds to the prediction of one ML model. Conceptually, preventive models capture the events that are *necessary* for the event $y = 1$: $y \implies s_j$.

KNOWLEDGE INTEGRATION Given a data example $(x, y) \sim \mathcal{D}_b$ or $(x, y) \sim \mathcal{D}_a$, y is unknown to KEMLP. We create a factor graph to embed the domain knowledge as follows. The outputs of each model over x become *input variables*: $s_*, s_{\mathcal{I}} = \{s_i : i \in \mathcal{I}\}, s_{\mathcal{J}} = \{s_j : j \in \mathcal{J}\}$. KEMLP also has an output variable $o \in \mathcal{Y}$, which corresponds to its prediction. Different models introduce different types of factors connecting these variables:

- Main model: KEMLP introduces a factor between the main model s_* and the output variable o with factor function $f_*(o, s_*) = \mathbb{1}\{o = s_*\}$;

- Permissive model: KEMLP introduces a factor between each permissive model s_i and the output variable o with factor function $f_i(o, s_i) = \mathbb{1}\{s_i \Longrightarrow o\}$.

- preventive model: KEMLP introduces a factor between each preventive model s_j and the output variable o with factor function $f_j(o, s_j) = \mathbb{1}\{o \Longrightarrow s_j\}$.

LEARNING WITH KEMLP To make a prediction, KEMLP outputs the *probability* of the output variable o. KEMLP assigns a weight for each model and constructs the following statistical model:

$$\begin{aligned}&\mathbb{P}[o, s_*, s_\mathcal{I}, s_\mathcal{J}, w_*, w_\mathcal{I}, w_\mathcal{J}, b_o] \quad \propto \\ &\exp\{b_o + w_* f_*(o, s_*)\} \times \exp\Big\{\sum_{i\in\mathcal{I}} w_i f_i(o, s_i)\Big\} \times \exp\Big\{\sum_{j\in\mathcal{J}} w_j f_j(o, s_j)\Big\}\end{aligned}$$

where w_*, w_i, w_j are the corresponding weights for models s_*, s_i, s_j, $w_\mathcal{I} = \{w_i : i \in \mathcal{I}\}, w_\mathcal{J} = \{w_j : j \in \mathcal{J}\}$ and b_o is some bias parameter that depends on o. For the simplicity of exposition, we use an equivalent notation by putting all the weights and outputs of factor functions into vectors using an ordering of models. More precisely, we define

$$\begin{aligned}\mathbf{w} &= [1; w_*; (w_i)_{i\in\mathcal{I}}; (w_j)_{j\in\mathcal{J}}], \\ \mathbf{f}_o(s_*, s_\mathcal{I}, s_\mathcal{J}) &= [b_o; f_*(o, s_*); (f_i(o, s_i))_{i\in\mathcal{I}}; (f_j(o, s_j))_{j\in\mathcal{J}}],\end{aligned}$$

for $o \in \mathcal{Y}$. All concatenated vectors from above are in $\mathbb{R}^{m+n+2}$. Given this, an equivalent form of KEMLP's statistical model is

$$\mathbb{P}[o|s_*, s_\mathcal{I}, s_\mathcal{J}, \mathbf{w}] = \frac{1}{Z_w} \exp(\langle w, \mathbf{f}_o(s_*, s_\mathcal{I}, s_\mathcal{J})\rangle) \qquad (4.1)$$

where Z_w is the normalization constant over $o \in \mathcal{Y}$. With some abuse of notation, **w** is meant to govern all parameters including weights and biases whenever used with probabilities. such that

$$Z_w = \exp\left(\langle w, \mathbf{f}_0(s_*, s_\mathcal{I}, s_\mathcal{J})\rangle\right) + \exp\left(\langle w, \mathbf{f}_1(s_*, s_\mathcal{I}, s_\mathcal{J})\rangle\right).$$

WEIGHT LEARNING During the training phase of KEMLP, we choose parameters w by performing standard maximum likelihood estimation over a training dataset. Given a particular input instance $x^{(n)}$, respective model predictions $s_*^{(n)}, s_\mathcal{I}^{(n)}, s_\mathcal{J}^{(n)}$, and the ground truth label $y^{(n)}$, we minimize the negative log-likelihood function in view of

$$\hat{w} = \arg\min_{w} \left\{ -\sum_n \log\left(\mathbb{P}[o^{(n)} = y^{(n)} | s_*^{(n)}, s_\mathcal{I}^{(n)}, s_\mathcal{J}^{(n)}, \mathbf{w}]\right)\right\}.$$

INFERENCE During the inference phase of KEMLP, given an input example $\hat{x}$, we predict $\hat{y}$ that has the largest probability given the respective model predictions $\hat{s}_*, \hat{s}_\mathcal{I}, \hat{s}_\mathcal{J}$, namely, $\hat{y} = \arg\max_{\tilde{y}\in\mathcal{Y}} \mathbb{P}[o = \tilde{y} | \hat{s}_*, \hat{s}_\mathcal{I}, \hat{s}_\mathcal{J}, \hat{\mathbf{w}}]$.

4.4 THEORETICAL ANALYSIS

How does knowledge integration impact the robustness of KEMLP? In this section, we provide a theoretical analysis about the impact of domain knowledge integration on the robustness of KEMLP. We hope to (1) depict the regime under which knowledge integration can help with robustness; (2) explain how a collection of "weak" (in terms of prediction accuracy) but "robust" auxiliary models, on tasks different from the main one, can be used to boost overall robustness. Here we state the main results, whereas we refer interested readers to Section 4.5 where we provide all relevant details. Furthermore, we are interested in the performance of KEMLP over both adversarial and benign examples. Towards that, we define our performance metric as follows.

WEIGHTED ROBUST ACCURACY Previous theoretical analysis on ML robustness (Javanmard, Soltanolkotabi, and Hassani, 2020; Raghunathan et al., 2020; Xu, Caramanis, and Mannor, 2009) have identified two natural dimensions of model quality: *clean accuracy* and *robust accuracy*, which are the accuracy of a given ML model on inputs x drawn from either the benign distribution $\mathcal{D}_b$ or adversarial distribution $\mathcal{D}_a$. In this work, to balance their trade-off, we use their weighted average as our main metric of interest. That is, given a classifier $h : \mathcal{X} \to \mathcal{Y}$ we define its Weighted Robust Accuracy as

$$\mathcal{A}_h = \pi_{D_a} \mathbb{P}_{\mathcal{D}_a}[h(x) = y] + \pi_{D_b} \mathbb{P}_{\mathcal{D}_b}[h(x) = y].$$

We use $\mathcal{A}^{KEMLP}$ and $\mathcal{A}^{main}$ to denote the weighted robust accuracies of KEMLP and main task model, respectively.

4.4.1 *Weighted Robust Accuracy of* KEMLP

The goal of our analysis is to identify the regime under which $\mathcal{A}^{KEMLP} > \mathcal{A}^{main}$ is guaranteed. The main analysis to achieve this hinges on deriving the weighted robust accuracy $\mathcal{A}^{KEMLP}$ for KEMLP. We first describe the modeling assumptions of our analysis, and then describe two key characteristics of models, culminating in a lower bound of $\mathcal{A}^{KEMLP}$.

MODELING ASSUMPTIONS We assume that for a fixed z, that is, for a fixed $\mathcal{D} \in \{\mathcal{D}_b, \mathcal{D}_a\}$, the models make independent errors given the target variable. Thus, for all $\mathcal{D} \in \{\mathcal{D}_b, \mathcal{D}_a\}$, the class conditional distribution can be decomposed as

$$\mathbb{P}_{\mathcal{D}}[s_*, s_{\mathcal{I}}, s_{\mathcal{J}} | y] = \mathbb{P}_{\mathcal{D}}[s_* | y] \prod_{i \in \mathcal{I}} \mathbb{P}_{\mathcal{D}}[s_i | y] \prod_{j \in \mathcal{J}} \mathbb{P}_{\mathcal{D}}[s_j | y].$$

We also assume for simplicity that the main task model makes symmetric errors given the class of target variable, that is, $\mathbb{P}_{\mathcal{D}}[s_* \neq y | y]$ is fixed with respect to y for all $\mathcal{D} \in \{\mathcal{D}_b, \mathcal{D}_a\}$.

CHARACTERIZING MODELS: TRUTH RATE (α) AND FALSE RATE (ε) Each auxiliary model $k \in \mathcal{I} \cup \mathcal{J}$ is characterized by two values, their truth rate

(α) and false rate (ε) over benign and adversarial distributions. These values measure the *consistency* of the model with the ground truth:

$$PermissiveModels:$$
$$\alpha_{i,\mathcal{D}} := \mathbb{P}_{\mathcal{D}}[s_i = y|y = 1], \;\; \varepsilon_{i,\mathcal{D}} := \mathbb{P}_{\mathcal{D}}[s_i \neq y|y = 0]$$

$$preventiveModels:$$
$$\alpha_{j,\mathcal{D}} := \mathbb{P}_{\mathcal{D}}[s_j = y|y = 0], \;\; \varepsilon_{j,\mathcal{D}} := \mathbb{P}_{\mathcal{D}}[s_j \neq y|y = 1]$$

Note that, given the asymmetric nature of these auxiliary models, we do *not* necessarily have $\varepsilon_{k,\mathcal{D}} = 1 - \alpha_{k,\mathcal{D}}$. In addition, for a high quality permissive model ($k \in \mathcal{I}$), or a high quality preventive model ($k \in \mathcal{J}$) for which the logic rules mostly hold, we expect $\alpha_{k,\mathcal{D}}$ to be large and $\varepsilon_{k,\mathcal{D}}$ to be small.

We define the truth rate of the main model over data examples drawn from $\mathcal{D} \in \{\mathcal{D}_b, \mathcal{D}_a\}$ as $\alpha_{*,\mathcal{D}} := \mathbb{P}_{\mathcal{D}}(s_* = y)$, and its false rate as $\varepsilon_{*,\mathcal{D}} := \mathbb{P}_{\mathcal{D}}(s_* \neq y) = 1 - \alpha_{*,\mathcal{D}}$.

These characteristics are of integral importance to the weighted robust accuracy of KEMLP. To combine all the models together, we define upper and lower bounds to truth rates and false rates. For the main model, we have ${}_{\wedge}\alpha_* := \min_{\mathcal{D}} \alpha_{*,\mathcal{D}}$ and ${}_{\vee}\alpha_* := \max_{\mathcal{D}} \alpha_{*,\mathcal{D}}$. For the auxiliary models, on the other hand, for each model index $k \in \mathcal{I} \cup \mathcal{J}$, we have

$$
\begin{aligned}
{}_{\wedge}\alpha_k &:= \min_{\mathcal{D}} \alpha_{k,\mathcal{D}}, \quad {}_{\wedge}\varepsilon_k := \min_{\mathcal{D}} \varepsilon_{k,\mathcal{D}} \\
{}_{\vee}\alpha_k &:= \max_{\mathcal{D}} \alpha_{k,\mathcal{D}}, \quad {}_{\vee}\varepsilon_k := \max_{\mathcal{D}} \varepsilon_{k,\mathcal{D}}.
\end{aligned}
$$

Intuitively, the difference between ${}_{\wedge}\alpha$ and ${}_{\vee}\alpha$ (resp. ${}_{\wedge}\varepsilon$ and ${}_{\vee}\varepsilon$) indicates the "robustness" of each individual model. If a model performs very similarly when it is given a benign and an adversarial example, we have that ${}_{\wedge}\alpha$ should be similar to ${}_{\vee}\alpha$ (resp. ${}_{\wedge}\varepsilon$ to ${}_{\vee}\varepsilon$).

The truth and false rates of models directly influence the factor weights which govern the influence of models in the main task. In Section 4.5 we prove that the optimal weight of an auxiliary model is bounded by $w_k \geq \log {}_{\wedge}\alpha_k(1 - {}_{\vee}\varepsilon_k)/(1 - {}_{\wedge}\alpha_k){}_{\vee}\varepsilon_k$, for all $k \in \mathcal{I} \cup \mathcal{J}$. That is, the lowest truth rate and highest false rate of an auxiliary model (resp. ${}_{\wedge}\alpha_k$ and ${}_{\vee}\epsilon_k$) are indicative of its influence in the main task. By taking partial derivatives, this lower bound can be shown to be increasing in ${}_{\wedge}\alpha_k$ and decreasing in ${}_{\vee}\epsilon_k$. That is, as the lowest truth rate of a model gets higher, KEMLP increases its influence in the weighted majority voting accordingly – in the above nonlinear fashion. The lowest truth rate is often determined by the *robust accuracy*. As a result, the more "robust" an auxiliary model is, the larger the influence on KEMLP, which naturally contributes to its robustness.

WEIGHTED ROBUST ACCURACY OF KEMLP We now provide a lower bound on the weighted robust accuracy of KEMLP, which can be written as

$$\mathcal{A}^{\text{KEMLP}} = \mathbb{E}_{\mathcal{D}\sim\{\mathcal{D}_a,\mathcal{D}_b\}}\mathbb{E}_{y\sim\mathcal{Y}}\left[\mathbb{P}_{\mathcal{D}}[o = y|y, \mathbf{w}]\right]. \tag{4.2}$$

We first provide one key technical lemma followed by the general theorem.

We see that the key component in $\mathcal{A}^{\text{KEMLP}}$ is $\mathbb{P}_{\mathcal{D}}[o = y|y, \mathbf{w}]$, the conditional probability that a KEMLP pipeline outputs the correct prediction. Using knowledge aggregation rules f_*, f_i and f_j, as well as Equation 4.1, for each $\mathcal{D} \in \{\mathcal{D}_b, \mathcal{D}_a\}$ we have

$$\begin{aligned}\mathbb{P}_{\mathcal{D}}[o = y|y, \mathbf{w}] &= \mathbb{P}_{\mathcal{D}}\Big[\mathbb{P}[o = y|s_*, s_{\mathcal{I}}, s_{\mathcal{J}}, \mathbf{w}] > 1/2\big|y\Big] \\ &= \mathbb{P}_{\mathcal{D}}\left[\langle w, \mathbf{f}_y(s_*, s_{\mathcal{I}}, s_{\mathcal{J}}) - \mathbf{f}_{1-y}(s_*, s_{\mathcal{I}}, s_{\mathcal{J}})\rangle > 0|y\right].\end{aligned}$$

To bound the above value, we need to characterize the concentration behavior of the random variable

$$\Delta_w(y, s_*, s_{\mathcal{I}}, s_{\mathcal{J}}) := \langle w, \mathbf{f}_y(s_*, s_{\mathcal{I}}, s_{\mathcal{J}}) - \mathbf{f}_{1-y}(s_*, s_{\mathcal{I}}, s_{\mathcal{J}})\rangle.$$

That is, we need to bound its left tail below zero. For this purpose, we reason about its expectation, leading to the following lemma.

Lemma 8. *Let Δ_w be a random variable defined above. Suppose that KEMLP uses optimal parameters w such that $\mathbb{P}[y|s_*, s_{\mathcal{I}}, s_{\mathcal{J}}] = \mathbb{P}[o|s_*, s_{\mathcal{I}}, s_{\mathcal{J}}, \mathbf{w}]$. Let also r_y*

denote the log-ratio of class imbalance $\log \frac{\mathbb{P}[y=1]}{\mathbb{P}[y=0]}$. *For a fixed* $y \in \mathcal{Y}$ *and* $\mathcal{D} \in \{\mathcal{D}_b, \mathcal{D}_a\}$, *one has*

$$\begin{aligned}
&\mathbb{E}_{s_*, s_{\mathcal{I}}, s_{\mathcal{J}}}[\Delta_w(y, s_*, s_{\mathcal{I}}, s_{\mathcal{J}}) | y] \\
&\geq \mu_{d_{*,\mathcal{D}}} + y\mu_{d_{\mathcal{I},\mathcal{D}}} + (1-y)\mu_{d_{\mathcal{J},\mathcal{D}}} + (2y-1)r_y := \mu_{y,\mathcal{D}},
\end{aligned}$$

where

$$\mu_{d_{*,\mathcal{D}}} = \alpha_{*,\mathcal{D}} \log \frac{{}_{\wedge}\alpha_*}{1 - {}_{\wedge}\alpha_*} + (1 - \alpha_{*,\mathcal{D}}) \log \frac{1 - {}_{\vee}\alpha_*}{{}_{\vee}\alpha_*},$$

$$\begin{aligned}
\mu_{d_{\mathcal{I},\mathcal{D}}} = &\sum_{i \in \mathcal{I}} \alpha_{i,\mathcal{D}} \log \frac{{}_{\wedge}\alpha_i}{{}_{\vee}\varepsilon_i} + (1 - \alpha_{i,\mathcal{D}}) \log \frac{1 - {}_{\vee}\alpha_i}{1 - {}_{\wedge}\varepsilon_i} \\
&- \sum_{j \in \mathcal{J}} \varepsilon_{j,\mathcal{D}} \log \frac{{}_{\vee}\alpha_j}{{}_{\wedge}\varepsilon_j} - (1 - \varepsilon_{j,\mathcal{D}}) \log \frac{1 - {}_{\wedge}\alpha_j}{1 - {}_{\vee}\varepsilon_j},
\end{aligned}$$

and

$$\begin{aligned}
\mu_{d_{\mathcal{J},\mathcal{D}}} = &\sum_{j \in \mathcal{J}} \alpha_{j,\mathcal{D}} \log \frac{{}_{\wedge}\alpha_j}{{}_{\vee}\varepsilon_j} + (1 - \alpha_{j,\mathcal{D}}) \log \frac{1 - {}_{\vee}\alpha_j}{1 - {}_{\wedge}\varepsilon_j} \\
&- \sum_{i \in \mathcal{I}} \varepsilon_{i,\mathcal{D}} \log \frac{{}_{\vee}\alpha_i}{{}_{\wedge}\varepsilon_i} - (1 - \varepsilon_{i,\mathcal{D}}) \log \frac{1 - {}_{\wedge}\alpha_i}{1 - {}_{\vee}\varepsilon_i}.
\end{aligned}$$

Proof Sketch. This lemma can be derived by first decomposing Δ_w into parts that are relevant for $s_*, s_{\mathcal{I}}, s_{\mathcal{J}}$, namely there exist $d_{*,\mathcal{D}}, d_{\mathcal{I},\mathcal{D}}, d_{\mathcal{J},\mathcal{D}}$ such that

$$\Delta_w(y, s_*, s_{\mathcal{I}}, s_{\mathcal{J}}) = d_{*,\mathcal{D}} + y d_{\mathcal{I},\mathcal{D}} + (1-y) d_{\mathcal{J},\mathcal{D}} + (2y-1) r_y.$$

Then we prove that $\mu_{*,\mathcal{D}} \leq \mathbb{E}[d_{*,\mathcal{D}}]$ for the main model, and $\mu_{d_{\mathcal{K},\mathcal{D}}} \leq \mathbb{E}[d_{\mathcal{K},\mathcal{D}}]$ for $\mathcal{K} \in \{\mathcal{I}, \mathcal{J}\}$, the permissive and preventive models.

discussion The above lemma illustrates the relationship between the models and $\mathcal{A}^{\text{KEMLP}}$. Intuitively, the larger $\mu_{y,\mathcal{D}}$ is, the further away the expectation of $\Delta_w(y, s_*, s_{\mathcal{I}}, s_{\mathcal{J}})$ is from 0, and thus, the larger the probability that $\Delta_w(y, s_*, s_{\mathcal{I}}, s_{\mathcal{J}}) > 0$. We see that $\mu_{y,\mathcal{D}}$ consists of three terms: $\mu_{d_{*,\mathcal{D}}}$, $\mu_{d_{\mathcal{I},\mathcal{D}}}$, $\mu_{d_{\mathcal{J},\mathcal{D}}}$, measuring the contributions from the main model for all y, permissive models and preventive models for $y = 1$ and $y = 0$, respectively. More specifically, $\mu_{y,\mathcal{D}}$ is increasing in terms of a weighted sum of α_i, and decreasing in terms of a weighted sum of ε_j. When $s_i \implies y$ holds (permissive models), it implies a large α_i for $y = 1$, whereas when $y \implies s_j$ holds (preventive model) it implies a small ε_j for $y = 1$. Thus, this lemma connects the property of auxiliary models to the weighted robust accuracy of KEMLP.

4.4.2 *Convergence*

Now we are ready to present our convergence result.

Theorem 9 (Convergence of Pipeline Accuracy). *For $y \in \mathcal{Y}$ and $\mathcal{D} \in \{\mathcal{D}_b, \mathcal{D}_a\}$, let $\mu_{y,\mathcal{D}}$ be defined as in Lemma 8. Suppose that the* modeling assumption *holds, and suppose that $\mu_{d_{\mathcal{K},\mathcal{D}}} > 0$, for all $\mathcal{K} \in \{\mathcal{I}, \mathcal{J}\}$ and $\mathcal{D} \in \{\mathcal{D}_b, \mathcal{D}_a\}$. Then*

$$\mathcal{A}^{KEMLP} \geq 1 - \mathbb{E}_{\mu_{y,\mathcal{D}}}[\exp\left(-2\mu_{y,\mathcal{D}}^2/v^2\right)], \tag{4.3}$$

where v^2 is the variance upper bound to $\mathbb{P}[o = y|y, \mathbf{w}]$ with

$$v^2 = 4\left(\log \frac{{}_{\vee}\alpha_*}{1 - {}_{\wedge}\alpha_*}\right)^2 + \sum_{k \in \mathcal{I} \cup \mathcal{J}} \left(\log \frac{{}_{\vee}\alpha_k(1 - {}_{\wedge}\varepsilon_k)}{{}_{\wedge}\varepsilon_k(1 - {}_{\vee}\alpha_k)}\right)^2.$$

Proof Sketch. We begin by subtracting the term $\mu_{y,\mathcal{D}}$ from $\mathbb{P}_{\mathcal{D}}(o = y|y, \mathbf{w})$, and then decomposing the result into individual summands, where each summand is induced by a single model. We then treat each summand as a bounded increment whose sum is a submartingale. Followed by an application of generalized bounded difference inequality (Geer, 2002), we arrive at the proof, whose full details can be found in Section 4.5.

DISCUSSION In the following, we attempt to understand the scaling of the weighted robust accuracy of KEMLP in terms of the models' characteristics.

Impact of truth rates and false rates: We note that $\mu_{d_{\mathcal{K},\mathcal{D}}}$ for $\mathcal{K} \in \{\mathcal{I}, \mathcal{J}\}$, which is an additive component of $\mu_{y,\mathcal{D}}$, is crucial in understanding the parameters contributing to the performance of KEMLP. Generally, a larger $\mu_{d_{\mathcal{K},\mathcal{D}}}$ (and hence $\mu_{y,\mathcal{D}}$) would increase the right tail probability of $\Delta_{\mathbf{w}}(y, s_*, s_{\mathcal{I}}, s_{\mathcal{J}})$ leading to a larger weighted accuracy for KEMLP. Although exceptions exist in cases where the variance increases disproportionally, here in our discussion we first focus on parameters that increase $\mu_{d_{\mathcal{K},\mathcal{D}}}$. Towards that, we simplify our exposition and let each auxiliary model have the same truth and false rate over both benign and adversarial examples, and within each type, where the exact parameters are given by $\alpha_k := \alpha_{k,\mathcal{D}} = {}_\wedge\alpha_{k,\mathcal{D}} = {}_\vee\alpha_{k,\mathcal{D}}$ and $\epsilon_k := \epsilon_{k,\mathcal{D}} = {}_\wedge\epsilon_{k,\mathcal{D}} = {}_\vee\epsilon_{k,\mathcal{D}}$, for $k \in \mathcal{I} \cup \mathcal{J}$. In this simplified setting where the expected performance improvement by the auxiliary models is given by $\mu_{d_{\mathcal{K},\mathcal{D}}}$ for $\mathcal{K} \in \{\mathcal{I}, \mathcal{J}\}$ and fixed with respect to $\mathcal{D}$, one can observe through partial derivatives that $\mu_{d_{\mathcal{K},\mathcal{D}}}$ is increasing over α_k and decreasing over ϵ_k. This explains why the two types of knowledge rules would help: high-quality permissive models would have high truth rate and low false rate (α_i and ϵ_i), as well as the preventive models (α_j and ϵ_j), yet with different coverages for $y \in \mathcal{Y}$.

Auxiliary models in KEMLP - the more the merrier? Next, we investigate the effect of the number of auxiliary models. To simplify, let $|\mathcal{I}| = |\mathcal{J}|$, and let $\hat{\mu}_{y,\mathcal{D}}$ be a random variable with $\hat{\mu}_{y,\mathcal{D}} = \mu_{y,\mathcal{D}}/(n+1)$, and $\hat{v}^2 = v^2/(n+1)$. The exponent thus becomes $-\mu^2_{y,\mathcal{D}}/v^2 = -(n+1)\hat{\mu}^2_{y,\mathcal{D}}/\hat{v}^2$. One can show that $\hat{\mu}^2_{y,\mathcal{D}}/\hat{v}^2 \geq c$ for some positive constant c, implying that $\mathcal{A}^{\text{KEMLP}} \geq 1 - \exp(-2(n+1)c)$. That is, increasing the number of models generally improves the weighted robust accuracy of KEMLP. To demonstrate this, we now focus on understanding the scaling of weighted robust accuracy on a simplified setting. We assume that the auxiliary models are *homogeneous* for each type: permissive or preventive. For example, α_k is fixed with respect to $k \in \mathcal{I} \cup \mathcal{J}$, hence we drop the subscripts, i.e., $\alpha_{k,\mathcal{D}} = \alpha$ and $\epsilon_{k,\mathcal{D}} = \epsilon$. We assume that the same number of auxiliary models are used, namely $|\mathcal{I}| = |\mathcal{J}| = n$, and that the classes are balanced with $\mathbb{P}_{\mathcal{D}}(y = 1) = \mathbb{P}_{\mathcal{D}}(y = 0)$, for all $\mathcal{D} \in \{\mathcal{D}_b, \mathcal{D}_a\}$. Finally, we let $\alpha_{*,\mathcal{D}_b} = 1$ and $\alpha_{*,\mathcal{D}_a} = 0$, and $\alpha - \epsilon > 0$. Then, the following holds.

Corollary 6 (Homogenous models)**.** *The weighted robust accuracy of KEMLP in the homogeneous setting satisfies*

$$\mathcal{A}^{KEMLP} \geq 1 - \exp\big(-2n(\alpha - \epsilon)^2\big).$$

In particular, one has $\lim_{n\to\infty} \mathcal{A}^{KEMLP} = 1$.

For this particular case, the predicted class for the target variable y is based upon an (unweighted) majority voting decision. The above result suggests that for a setting where the auxiliary models are homogeneous with different coverage, the performance of KEMLP to predict the output variable y robustly is determined by: (a) the difference between the probability of predicting the output variable correctly and that of making an erroneous prediction, that is, $\alpha - \varepsilon$, and (b) the number of auxiliary models. Consequently, $\mathcal{A}^{\text{KEMLP}}$ converges to 1 exponentially fast in the number of auxiliary models as long as $\alpha - \varepsilon > 0$, which is naturally satisfied by the principle KEMLP employs while constructing the logical relations between the output variable and different knowledge.

4.4.3 *Comparison*

Theorem 9 guarantees that the addition of models allows the weighted robust accuracy of KEMLP to converge to 1 exponentially fast. We now introduce a sufficient condition under which $\mathcal{A}^{KEMLP}$ is strictly better than $\mathcal{A}^{main}$.

Theorem 10 (Sufficient condition for $\mathcal{A}^{\text{KEMLP}} > \mathcal{A}^{\text{main}}$)**.** *Let the number of permissive and preventive models be the same and denoted by n such that $n := |\mathcal{I}| = |\mathcal{J}|$. Note that the weighted accuracy of the main model in terms of its truth rate is simply $\alpha_* := \sum_{\mathcal{D}\in\{\mathcal{D}_b,\mathcal{D}_a\}} \pi_{\mathcal{D}}\alpha_{*,\mathcal{D}}$. Moreover, let $\mathcal{K},\mathcal{K}' \in \{\mathcal{I},\mathcal{J}\}$ with $\mathcal{K} \neq \mathcal{K}'$ and for any $\mathcal{D} \in \{\mathcal{D}_b,\mathcal{D}_a\}$, let*

$$\gamma_{\mathcal{D}} := \frac{1}{n+1}\min_{\mathcal{K}}\Big\{\alpha_{*,\mathcal{D}} - 1/2 + \sum_{k\in\mathcal{K}}\alpha_{k,\mathcal{D}} - \sum_{k'\in\mathcal{K}'}\varepsilon_{k',\mathcal{D}}\Big\}.$$

If $\gamma_{\mathcal{D}} > \sqrt{\frac{4}{n+1}\log\frac{1}{1-\alpha_*}}$ for all $\mathcal{D} \in \{\mathcal{D}_b,\mathcal{D}_a\}$, then $\mathcal{A}^{KEMLP} > \mathcal{A}^{main}$.

Proof Sketch. We first approximate $\Delta_{\mathbf{w}}(y,s_*,s_{\mathcal{I}},s_{\mathcal{J}})$ with a Poisson Binomial random variable and apply the relevant Chernoff bound. Imposing a strict bound between the Chernoff result and the true and false rates of main model concludes the proof. We note that this bound is slightly simplified, and our full proof in the Section 4.5 is tighter.

DISCUSSION We start by noting that $\gamma_{\mathcal{D}}$ is a combined truth rate of all models normalized over the number of models. That is, for a fixed distribution $\mathcal{D}$, $\alpha_{*,\mathcal{D}} - 1/2$ indicates the truth rate of main task model over a random classifier and $\sum_{k \in \mathcal{K}} \alpha_{k,\mathcal{D}} - \sum_{k' \in \mathcal{K}'} \varepsilon_{k',\mathcal{D}}$ refers to the improvement by the auxiliary models on top of the main task model. More specifically, in cases where the true class of output variable is positive with $y = 1$, $\sum_{i \in \mathcal{I}} \alpha_{i,\mathcal{D}} - \sum_{j \in \mathcal{J}} \varepsilon_{j,\mathcal{D}}$ account for the total (and unnormalized) success of permissive models in identifying $y = 1$ interfered by the failure of preventive model in identifying $y = 1$ (resp. For $y = 0$, $\mathcal{K} = \mathcal{J}$). Hence, $\gamma_{\mathcal{D}}$ is the "worst-case" combined truth rate of all models, where the worst-case refers to minimization over all possible labels of target variable.

Proposition 10 therefore forms a relationship between the improvement of KEMLP over the main task model and the combined truth rate of models, and theoretically justifies our intuition – larger truth rates and lower false rates of individual auxiliary models result in larger combined truth rate $\gamma_{\mathcal{D}}$, hence making the sufficient condition more likely to hold. Additionally, employing a large number of auxiliary models is found to be beneficial for better KEMLP performance, as we conclude in Corollary 6 as well. Our finding here also confirms that in the extreme scenarios where the main task model has a perfect clean and robust truth rate ($\alpha_* = 1$), it is *not* possible to improve upon the main task model. Conversely, when $\alpha_* = 0$, any improvement by KEMLP would result in absolute improvement over the main model.

4.5 PROOFS

4.5.1 *Preliminaries*

For completeness, here we recall our setup and introduce further remarks.

DATA MODEL We begin by recalling our notation. We consider a classification problem under supervised learning setting, defined on a feature space $\mathcal{X}$ and a finite label space $\mathcal{Y}$. We refer to $x \in \mathcal{X}$ as an input, and $y \in \mathcal{Y}$ as the prediction. An input x can be a benign example or an adversarial example. To model this, we use $z \in \{0, 1\}$, a latent variable which is not exposed to KEMLP. That is, x is an adversarial example with $(x, y) \sim \mathcal{D}_a$ whenever $z = 1$, and $(x, y) \sim \mathcal{D}_b$ otherwise, where $\mathcal{D}_a$ and $\mathcal{D}_b$ represent the adversarial and benign data distribution. We let $\pi_{\mathcal{D}_a} = \mathbb{P}(z = 1)$ and $\pi_{\mathcal{D}_b} = \mathbb{P}(z = 0)$, implying $\pi_{\mathcal{D}_a} + \pi_{\mathcal{D}_b} = 1$. For convenience, we denote $\mathbb{P}_{\mathcal{D}_a}(x, y) = \mathbb{P}(x, y|z = 1)$ and $\mathbb{P}_{\mathcal{D}_b}(x, y) = \mathbb{P}(x, y|z = 0)$.

For simplicity, we describe the KEMLP framework as a binary classification task, in which case $\mathcal{Y} = \{0, 1\}$, noting that the multi-class scenario is a simple extension of it. We introduce the KEMLP framework as follows.

KNOWLEDGE INTEGRATION Given a data example $(x, y) \sim \mathcal{D}_b$ or $(x, y) \sim \mathcal{D}_a$, y is unknown to KEMLP. We create a factor graph to embed the domain knowledge as follows. The outputs of each model over x become *input variables*: $s_*, s_\mathcal{I} = \{s_i : i \in \mathcal{I}\}, s_\mathcal{J} = \{s_j : j \in \mathcal{J}\}$. KEMLP also has an output variable $o \in \mathcal{Y}$, which corresponds to its prediction. Different models introduce different types of factors connecting these variables:

- Main model: KEMLP introduces a factor between the main model s_* and the output variable o with factor function $f_*(o, s_*) = \mathbb{1}\{o = s_*\}$;
- Permissive model: KEMLP introduces a factor between each permissive model s_i and the output variable o with factor function $f_i(o, s_i) = \mathbb{1}\{s_i \Longrightarrow o\}$.
- preventive model: KEMLP introduces a factor between each preventive model s_j and the output variable o with factor function $f_j(o, s_j) = \mathbb{1}\{o \Longrightarrow s_j\}$.

LEARNING WITH KEMLP To make a prediction, KEMLP outputs the *probability* of the output variable o. KEMLP assigns a weight for each model and constructs the following log-linear statistical model:

$$\begin{aligned}&\mathbb{P}[o, s_*, s_\mathcal{I}, s_\mathcal{J}, w_*, w_\mathcal{I}, w_\mathcal{J}] \\ &\qquad \propto \exp\{b_o + w_* f_*(o, s_*)\} \times \exp\Big\{\sum_{i \in \mathcal{I}} w_i f_i(o, s_i)\Big\} \times \exp\Big\{\sum_{j \in \mathcal{J}} w_j f_j(o, s_j)\Big\}\end{aligned}$$

where w_*, w_i, w_j are the corresponding weights for models s_*, s_i, s_j, $w_\mathcal{I} = \{w_i : i \in \mathcal{I}\}, w_\mathcal{J} = \{w_j : j \in \mathcal{J}\}$ and b_o is some bias parameter that depends on o.

For the simplicity of exposition, we use an equivalent notation by putting all the weights and outputs of factor functions into vectors using an ordering of models. More precisely, we define

$$\begin{aligned} \mathbf{w} &= [1; w_*; (w_i)_{i\in\mathcal{I}}; (w_j)_{j\in\mathcal{J}}], \\ \mathbf{f}_o(s_*, s_\mathcal{I}, s_\mathcal{J}) &= [b_o; f_*(o, s_*); (f_i(o, s_i))_{i\in\mathcal{I}}; (f_j(o, s_j))_{j\in\mathcal{J}}], \end{aligned}$$

for $o \in \mathcal{Y}$. All concatenated vectors from above are in $\mathbb{R}^{m+n+2}$. Given this, an equivalent form of KEMLP's statistical model is

$$\mathbb{P}[o|s_*, s_\mathcal{I}, s_\mathcal{J}, \mathbf{w}] = \frac{1}{Z_w}\exp(\langle w, \mathbf{f}_o(s_*, s_\mathcal{I}, s_\mathcal{J})\rangle) \tag{4.4}$$

where Z_w is the normalization constant over $o \in \mathcal{Y}$ such that

$$Z_w = \exp\left(\langle w, \mathbf{f}_0(s_*, s_\mathcal{I}, s_\mathcal{J})\rangle\right) + \exp\left(\langle w, \mathbf{f}_1(s_*, s_\mathcal{I}, s_\mathcal{J})\rangle\right).$$

With some abuse of notation, $\mathbf{w}$ is meant to govern all parameters including weights and biases whenever used with probabilities.

WEIGHT LEARNING During the training phase of KEMLP, we choose parameters w by performing standard maximum likelihood estimation over a training dataset. Given a particular input instance $x^{(n)}$, respective model predictions $s_*^{(n)}, s_\mathcal{I}^{(n)}, s_\mathcal{J}^{(n)}$, and the ground truth label $y^{(n)}$, we minimize the negative log-likelihood function in view of

$$\hat{w} = \arg\min_w \left\{ -\sum_n \log\left(\mathbb{P}[o^{(n)} = y^{(n)}|s_*^{(n)}, s_\mathcal{I}^{(n)}, s_\mathcal{J}^{(n)}, \mathbf{w}]\right)\right\}.$$

INFERENCE During the inference phase of KEMLP, given an input example $\hat{x}$, we predict $\hat{y}$ that has the largest probability given the respective model predictions $\hat{s}_*, \hat{s}_\mathcal{I}, \hat{s}_\mathcal{J}$, namely, $\hat{y} = \arg\max_{\tilde{y}\in\mathcal{Y}} \mathbb{P}[o = \tilde{y}|\hat{s}_*, \hat{s}_\mathcal{I}, \hat{s}_\mathcal{J}, \hat{\mathbf{w}}]$.

WEIGHTED ROBUST ACCURACY Previous theoretical analysis on ML robustness (Javanmard, Soltanolkotabi, and Hassani, 2020; Raghunathan et al., 2020; Xu, Caramanis, and Mannor, 2009) have identified two natural dimensions of model quality: *clean accuracy* and *robust accuracy*, which are the accuracy of a given ML model on inputs x drawn from either the benign distribution $\mathcal{D}_b$ or adversarial distribution $\mathcal{D}_a$. In this work, to balance their trade-off, we use their weighted average as our main metric of interest. That is, given a classifier $h : \mathcal{X} \to \mathcal{Y}$ we define its *Weighted Robust Accuracy* as

$$\mathcal{A}_h = \pi_{D_a}\mathbb{P}_{\mathcal{D}_a}[h(x) = y] + \pi_{D_b}\mathbb{P}_{\mathcal{D}_b}[h(x) = y].$$

We use $\mathcal{A}^{KEMLP}$ and $\mathcal{A}^{main}$ to denote the weighted robust accuracies of KEMLP and main task model, respectively.

MODELING ASSUMPTIONS We assume that for a fixed z, that is, for a fixed $\mathcal{D} \in \{\mathcal{D}_b, \mathcal{D}_a\}$, the models make independent errors given the target variable y. Thus, for all $\mathcal{D} \in \{\mathcal{D}_b, \mathcal{D}_a\}$ the class conditional distribution can be decomposed as

$$\mathbb{P}_{\mathcal{D}}[s_*, s_{\mathcal{I}}, s_{\mathcal{J}}|y] = \mathbb{P}_{\mathcal{D}}[s_*|y] \prod_{i \in \mathcal{I}} \mathbb{P}_{\mathcal{D}}[s_i|y] \prod_{j \in \mathcal{J}} \mathbb{P}_{\mathcal{D}}[s_j|y].$$

We also assume for simplicity that the main task model makes symmetric errors given the class of target variable, that is, $\mathbb{P}_{\mathcal{D}}[s_* \neq y|y]$ is fixed with respect to y for all $\mathcal{D} \in \{\mathcal{D}_b, \mathcal{D}_a\}$.

CHARACTERIZING MODELS: TRUTH RATE (α) AND FALSE RATE (ε) Each auxiliary model $k \in \mathcal{I} \cup \mathcal{J}$ is characterized by two values, their truth rate (α) and false rate (ε) over benign and adversarial distributions. These values measure the *consistency* of the model with the ground truth:

$$PermissiveModels :$$
$$\alpha_{i,\mathcal{D}} := \mathbb{P}_{\mathcal{D}}[s_i = y|y = 1], \quad \varepsilon_{i,\mathcal{D}} := \mathbb{P}_{\mathcal{D}}[s_i \neq y|y = 0]$$

$$\begin{aligned}&\textit{preventiveModels}:\\&\alpha_{j,\mathcal{D}} := \mathbb{P}_{\mathcal{D}}[s_j = y | y = 0], \quad \varepsilon_{j,\mathcal{D}} := \mathbb{P}_{\mathcal{D}}[s_j \neq y | y = 1]\end{aligned}$$

Note that, given the asymmetric nature of these auxiliary models, we do *not* necessarily have $\varepsilon_{k,\mathcal{D}} = 1 - \alpha_{k,\mathcal{D}}$. In addition, for a high quality permissive model ($k \in \mathcal{I}$), or a high quality preventive model ($k \in \mathcal{J}$) for which the logic rules mostly hold, we expect $\alpha_{k,\mathcal{D}}$ to be large and $\varepsilon_{k,\mathcal{D}}$ to be small.

We define the truth rate of main model over data examples drawn from $\mathcal{D} \in \{\mathcal{D}_b, \mathcal{D}_a\}$ as $\alpha_{*,\mathcal{D}} := \mathbb{P}_{\mathcal{D}}(s_* = y)$, and its false rate as $\varepsilon_{*,\mathcal{D}} := \mathbb{P}_{\mathcal{D}}(s_* \neq y) = 1 - \alpha_{*,\mathcal{D}}$.

These characteristics are of integral importance to weighted robust accuracy of KEMLP. To combine all the models together, we define upper and lower bounds to truth rates and false rates. For the main model, we have ${}_{\wedge}\alpha_* := \min_{\mathcal{D}} \alpha_{*,\mathcal{D}}$ and ${}_{\vee}\alpha_* := \max_{\mathcal{D}} \alpha_{*,\mathcal{D}}$. whereas for auxiliary models, for each model index $k \in \mathcal{I} \cup \mathcal{J}$, we have

$$ {}_{\wedge}\alpha_k := \min_{\mathcal{D}} \alpha_{k,\mathcal{D}}, \qquad {}_{\wedge}\varepsilon_k := \min_{\mathcal{D}} \varepsilon_{k,\mathcal{D}} \tag{4.5}$$

$$ {}_{\vee}\alpha_k := \max_{\mathcal{D}} \alpha_{k,\mathcal{D}}, \qquad {}_{\vee}\varepsilon_k := \max_{\mathcal{D}} \varepsilon_{k,\mathcal{D}}. \tag{4.6}$$

4.5.2 *Parameters*

In this section we will derive the closed-form expressions for the parameters based on our generative model, namely, weights and biases.

To make a prediction, KEMLP outputs the *marginal probability* of the output variable o. KEMLP assigns a weight for each model and constructs the following statistical model:

$$\begin{aligned}&\mathbb{P}[o | s_*, s_{\mathcal{I}}, s_{\mathcal{J}}, \mathbf{w}]\\&\quad \propto \exp\{b_o + w_* f_*(o, s_*)\} \times \exp\Big\{\sum_{i \in \mathcal{I}} w_i f_i(o, s_i)\Big\} \times \exp\Big\{\sum_{j \in \mathcal{J}} w_j f_j(o, s_j)\Big\},\end{aligned}$$

where w_*, w_i, w_j are the corresponding weights for models s_*, s_i, s_j, and b_o is some bias parameter that depends on o. For the simplicity of exposition, we use an equivalent notation by putting all the weights and outputs of factor functions into vectors using an ordering of models. More precisely, we define

$$\begin{aligned}\mathbf{w} &= [1; w_*; (w_i)_{i\in\mathcal{I}}; (w_j)_{j\in\mathcal{J}}],\\ \mathbf{f}_o(s_*, s_{\mathcal{I}}, s_{\mathcal{J}}) &= [b_o; f_*(o, s_*); (f_i(o, s_i))_{i\in\mathcal{I}}; (f_j(o, s_j))_{j\in\mathcal{J}}],\end{aligned}$$

for $o \in \mathcal{Y}$. All concatenated vectors from above are in $\mathbb{R}^{m+n+2}$. Given this, an equivalent form of KEMLP's statistical model is

$$\mathbb{P}[o|s_*, s_{\mathcal{I}}, s_{\mathcal{J}}, \mathbf{w}] = \frac{1}{Z_w}\exp(\langle w, \mathbf{f}_o(s_*, s_{\mathcal{I}}, s_{\mathcal{J}})\rangle), \tag{4.7}$$

where Z_w is the normalization constant over $o \in \mathcal{Y}$. We can further show that

$$\begin{aligned}\mathbb{P}[o = \tilde{y}|s_*, s_{\mathcal{I}}, s_{\mathcal{J}}, \mathbf{w}] &= \frac{\mathbb{P}[o = \tilde{y}|s_*, s_{\mathcal{I}}, s_{\mathcal{J}}, \mathbf{w}]}{\mathbb{P}[o = \tilde{y}|s_*, s_{\mathcal{I}}, s_{\mathcal{J}}, \mathbf{w}] + \mathbb{P}[o = 1 - \tilde{y}|s_*, s_{\mathcal{I}}, s_{\mathcal{J}}, \mathbf{w}]}\\ &= \frac{\exp(\langle w, \mathbf{f}_y(s_*, s_{\mathcal{I}}, s_{\mathcal{J}})\rangle)}{\exp(\langle w, \mathbf{f}_{\tilde{y}}(s_*, s_{\mathcal{I}}, s_{\mathcal{J}})\rangle) + \exp(\langle w, \mathbf{f}_{1-\tilde{y}}(s_*, s_{\mathcal{I}}, s_{\mathcal{J}})\rangle)}\\ &= \frac{1}{1 + \exp(-\Delta_w(\tilde{y}, s_*, s_{\mathcal{I}}, s_{\mathcal{J}}))}\end{aligned} \tag{4.8}$$

where $\Delta_w(\tilde{y}, s_*, s_{\mathcal{I}}, s_{\mathcal{J}})$ is previously defined as

$$\Delta_w(\tilde{y}, s_*, s_{\mathcal{I}}, s_{\mathcal{J}}) := \langle w, \mathbf{f}_{\tilde{y}}(s_*, s_{\mathcal{I}}, s_{\mathcal{J}}) - \mathbf{f}_{1-\tilde{y}}(s_*, s_{\mathcal{I}}, s_{\mathcal{J}})\rangle.$$

Therefore, we have

$$\mathbb{P}[o = \tilde{y}|s_*, s_{\mathcal{I}}, s_{\mathcal{J}}, \mathbf{w}] = \sigma(\Delta_w(\tilde{y}, s_*, s_{\mathcal{I}}, s_{\mathcal{J}})) \tag{4.9}$$

where $\sigma : \mathbb{R} \mapsto [0, 1]$ is the Sigmoid function.

Remark 3 (Closed form expression of $\Delta_w(\tilde{y}, s_*, s_{\mathcal{I}}, s_{\mathcal{J}})$)**.** *Recalling our knowledge integration rules, it can be shown that*

$$\begin{aligned}\Delta_w(\tilde{y}, s_*, s_{\mathcal{I}}, s_{\mathcal{J}}) &= \langle w, \mathbf{f}_{\tilde{y}}(s_*, s_{\mathcal{I}}, s_{\mathcal{J}}) - \mathbf{f}_{1-\tilde{y}}(s_*, s_{\mathcal{I}}, s_{\mathcal{J}})\rangle \\ &= b(\tilde{y}) + w_*\big(f_*(\tilde{y}, s_*) - f_*(1-\tilde{y}, s_*)\big) \\ &+ \sum_{i\in\mathcal{I}} w_i\big(f_i(\tilde{y}, s_i) - f_i(1-\tilde{y}, s_i)\big) \\ &+ \sum_{j\in\mathcal{J}} w_i\big(f_j(\tilde{y}, s_j) - f_j(1-\tilde{y}, s_j)\big)\end{aligned}$$

where $b(\tilde{y}) = b_{\tilde{y}} - b_{1-\tilde{y}}$. Let $b := b_1 - b_0$. Then $b(\tilde{y}) = (2\tilde{y}-1)b$.

Using the logical rules, we moreover have

$$\begin{aligned}f_*(\tilde{y}, s_*) - f_*(1-\tilde{y}, s_*) &= \mathbb{1}\{\tilde{y} = s_*\} - \mathbb{1}\{1-\tilde{y} = s_*\} \\ &= (2\tilde{y}-1)(2s_*-1) \\ f_i(\tilde{y}, s_i) - f_i(1-\tilde{y}, s_i) &= \mathbb{1}\{s_i \implies \tilde{y}\} - \mathbb{1}\{s_i \implies 1-\tilde{y}\} \\ &= (2\tilde{y}-1)s_i \\ f_j(\tilde{y}, s_j) - f_j(1-\tilde{y}, s_j) &= \mathbb{1}\{\tilde{y} \implies s_j\} - \mathbb{1}\{1-\tilde{y} \implies s_j\} \\ &= -(2\tilde{y}-1)(1-s_j).\end{aligned}$$

Therefore, the closed form expression for $\Delta_w(\tilde{y}, s_*, s_{\mathcal{I}}, s_{\mathcal{J}})$ is given by

$$\Delta_w(\tilde{y}, s_*, s_{\mathcal{I}}, s_{\mathcal{J}}) = (2\tilde{y}-1)\Big(b + w_*(2s_*-1) + \sum_{i\in\mathcal{I}} w_i s_i - \sum_{j\in\mathcal{J}} w_j(1-s_j)\Big)$$

Remark 4 (Optimal parameters)**.** *We now analyze the class conditional distribution $\mathbb{P}[y|s_*, s_\mathcal{I}, s_\mathcal{J}]$. Optimal set of parameters for our generative model must satisfy:*

$$\begin{aligned}
\mathbb{P}[y = \tilde{y}|s_*, s_\mathcal{I}, s_\mathcal{J}] &= \frac{\mathbb{P}[y = \tilde{y}, s_*, s_\mathcal{I}, s_\mathcal{J}]}{\mathbb{P}[s_*, s_\mathcal{I}, s_\mathcal{J}]} \\
&= \frac{\mathbb{P}[y = \tilde{y}, s_*, s_\mathcal{I}, s_\mathcal{J}]}{\mathbb{P}[y = \tilde{y}, s_*, s_\mathcal{I}, s_\mathcal{J}] + \mathbb{P}[y = 1 - \tilde{y}, s_*, s_\mathcal{I}, s_\mathcal{J}]} \\
&= \frac{1}{1 + \frac{\mathbb{P}[y=1-\tilde{y}, s_*, s_\mathcal{I}, s_\mathcal{J}]}{\mathbb{P}[y=\tilde{y}, s_*, s_\mathcal{I}, s_\mathcal{J}]}} = \frac{1}{1 + \exp\big(\log \frac{\mathbb{P}[y=1-\tilde{y}, s_*, s_\mathcal{I}, s_\mathcal{J}]}{\mathbb{P}[y=\tilde{y}, s_*, s_\mathcal{I}, s_\mathcal{J}]}\big)} \\
&= \frac{1}{1 + \exp\big(-\log \frac{\mathbb{P}[y=\tilde{y}, s_*, s_\mathcal{I}, s_\mathcal{J}]}{\mathbb{P}[y=1-\tilde{y}, s_*, s_\mathcal{I}, s_\mathcal{J}]}\big)}.
\end{aligned} \tag{4.10}$$

Note that, the optimal parameters satisfy

$$\mathbb{P}[o = \tilde{y}|s_*, s_\mathcal{I}, s_\mathcal{J}] = \mathbb{P}[y = \tilde{y}|s_*, s_\mathcal{I}, s_\mathcal{J}].$$

Hence, combining Equation 4.8 and Equation 4.10 as well as Remark 3 we further have

$$\log \frac{\mathbb{P}[y = \tilde{y}, s_*, s_\mathcal{I}, s_\mathcal{J}]}{\mathbb{P}[y = 1 - \tilde{y}, s_*, s_\mathcal{I}, s_\mathcal{J}]} = (2\tilde{y} - 1)\Big(b + w_*(2s_* - 1) + \sum_{i \in \mathcal{I}} w_i s_i - \sum_{j \in \mathcal{J}} w_j (1 - s_j)\Big). \tag{4.11}$$

Above remark indicates the condition that the optimal parameters must satisfy.

4.5.3 *Proof of Lemma 8*

Recall that for each model index $k \in \mathcal{I} \cup \mathcal{J}$ we define upper and lower bounds to truth rates and false rates as

$$\begin{aligned}
{}_\wedge\alpha_k &:= \min_\mathcal{D} \alpha_{k,\mathcal{D}}, \quad {}_\wedge\varepsilon_k := \min_\mathcal{D} \varepsilon_{k,\mathcal{D}} \\
{}_\vee\alpha_k &:= \max_\mathcal{D} \alpha_{k,\mathcal{D}}, \quad {}_\vee\varepsilon_k := \max_\mathcal{D} \varepsilon_{k,\mathcal{D}}.
\end{aligned}$$

Next, we revisit Lemma 8 towards its proof.

Lemma (Recall). *Let Δ_w be a random variable defined above. Suppose that KEMLP uses optimal parameters w such that $\mathbb{P}[y|s_*, s_\mathcal{I}, s_\mathcal{J}] = \mathbb{P}[o|s_*, s_\mathcal{I}, s_\mathcal{J}, \mathbf{w}]$. Let also r_y denote the log-ratio of class imbalance $\log \frac{\mathbb{P}[y=1]}{\mathbb{P}[y=0]}$. For a fixed $y \in \mathcal{Y}$ and $\mathcal{D} \in \{\mathcal{D}_b, \mathcal{D}_a\}$, one has*

$$\mathbb{E}_{s_*, s_\mathcal{I} s_\mathcal{J}}[\Delta_w(y, s_*, s_\mathcal{I}, s_\mathcal{J})|y] \geq \mu_{d_{*,\mathcal{D}}} + y\mu_{d_{\mathcal{I},\mathcal{D}}} + (1-y)\mu_{d_{\mathcal{J},\mathcal{D}}} + (2y-1)r_y := \mu_{y,\mathcal{D}},$$

where

$$\mu_{d_{*,\mathcal{D}}} = \alpha_{*,\mathcal{D}} \log \frac{{}_\wedge\alpha_*}{1 - {}_\wedge\alpha_*} + (1 - \alpha_{*,\mathcal{D}}) \log \frac{1 - {}_\vee\alpha_*}{{}_\vee\alpha_*},$$

$$\mu_{d_{\mathcal{I},\mathcal{D}}} = \sum_{i\in\mathcal{I}} \alpha_{i,\mathcal{D}} \log \frac{{}_\wedge\alpha_i}{{}_\vee\varepsilon_i} + (1-\alpha_{i,\mathcal{D}}) \log \frac{1 - {}_\vee\alpha_i}{1 - {}_\wedge\varepsilon_i} - \sum_{j\in\mathcal{J}} \varepsilon_{j,\mathcal{D}} \log \frac{{}_\vee\alpha_j}{{}_\wedge\varepsilon_j} - (1-\varepsilon_{j,\mathcal{D}}) \log \frac{1 - {}_\wedge\alpha_j}{1 - {}_\vee\varepsilon_j},$$

and

$$\mu_{d_{\mathcal{J},\mathcal{D}}} = \sum_{j\in\mathcal{J}} \alpha_{j,\mathcal{D}} \log \frac{{}_\wedge\alpha_j}{{}_\vee\varepsilon_j} + (1-\alpha_{j,\mathcal{D}}) \log \frac{1 - {}_\vee\alpha_j}{1 - {}_\wedge\varepsilon_j} - \sum_{i\in\mathcal{I}} \varepsilon_{i,\mathcal{D}} \log \frac{{}_\vee\alpha_i}{{}_\wedge\varepsilon_i} - (1-\varepsilon_{i,\mathcal{D}}) \log \frac{1 - {}_\wedge\alpha_i}{1 - {}_\vee\varepsilon_i}.$$

Proof of Lemma 8. We show earlier that the optimal parameters satisfy Equation 4.11. Note that the probabilities on the left hand side of Equation 4.11 are mixtures over both the benign and adversarial distributions. Namely,

$$\mathbb{P}[y = \tilde{y}, s_*, s_\mathcal{I}, s_\mathcal{J}] = \sum_{\mathcal{D}\in\{\mathcal{D}_b, \mathcal{D}_a\}} \pi_\mathcal{D} \mathbb{P}_\mathcal{D}[y = \tilde{y}, s_*, s_\mathcal{I}, s_\mathcal{J}].$$

Recall from our modeling assumptions that models are conditionally independent given y with $\mathbb{P}_\mathcal{D}[s_*, s_\mathcal{I}, s_\mathcal{J}|y = \tilde{y}] = \mathbb{P}_\mathcal{D}[s_*|y = \tilde{y}] \prod_{i\in\mathcal{I}} \mathbb{P}_\mathcal{D}[s_i|y = \tilde{y}] \prod_{j\in\mathcal{J}} \mathbb{P}_\mathcal{D}[s_j|y = \tilde{y}]$. Therefore, without loss of generality, this holds not for $\mathbb{P}[y = \tilde{y}, s_*, s_\mathcal{I}, s_\mathcal{J}]$. That is, each parameter is to encode this dependency structure and must be a function of some set of models. Below we propose a strategy to choose optimal weights to satisfy Equation 4.11.

We start by decomposing $\log \frac{\mathbb{P}[y=\tilde{y}, s_*, s_\mathcal{I}, s_\mathcal{J}]}{\mathbb{P}[y=1-\tilde{y}, s_*, s_\mathcal{I}, s_\mathcal{J}]}$.

$$\begin{aligned} &\log \frac{\mathbb{P}[y = \tilde{y}, s_*, s_\mathcal{I}, s_\mathcal{J}]}{\mathbb{P}[y = 1 - \tilde{y}, s_*, s_\mathcal{I}, s_\mathcal{J}]} \\ &= \log \frac{\mathbb{P}[y = \tilde{y}, s_*]}{\mathbb{P}[y = 1 - \tilde{y}, s_*]} + \sum_{i\in\mathcal{I}} \log \frac{\mathbb{P}[s_i|y = \tilde{y}, s_{I_i}]}{\mathbb{P}[s_i|y = 1 - \tilde{y}, s_{I_i}]} + \sum_{j\in\mathcal{J}} \log \frac{\mathbb{P}[s_j|y = \tilde{y}, s_I, s_{J_j}]}{\mathbb{P}[s_j|y = 1 - \tilde{y}, s_I, s_{J_j}]} \end{aligned}$$

where I_i is the set of i' such that $i' \in \mathcal{I}$ and $i' < i$. Similarly, we let J_j be the set of j' such that $j' \in \mathcal{J}$ and $j' < j$. Note that there are multiple such constructions to satisfy Equation 4.11 to have optimal set of weights.

We split our proof into three main steps as follows.

Step 1: Derivation of bounds for optimal set of parameters

Given our strategy, we then derive the parameters in terms of conditional probabilities of individual models. Towards that, let b be decomposed into its additive components such that $b = b_* + \sum_{i\in\mathcal{I}} b_i - \sum_{j\in\mathcal{J}} b_j$. Let also $r_y = \log \frac{\mathbb{P}[y=1]}{\mathbb{P}[y=0]}$. We derive bounds for each sensor using Equation 4.11 as follows.

- *Main task model*: The parameters for the main model simply satisfies

$$\begin{aligned}(2\tilde{y}-1)\big(w_*(2s_*-1)+b_*\big) &= \log\frac{\mathbb{P}[y=\tilde{y},s_*]}{\mathbb{P}[y=1-\tilde{y},s_*]}\\ &= \log\frac{\sum_{\mathcal{D}\in\{\mathcal{D}_b,\mathcal{D}_a\}}\pi_{\mathcal{D}}\mathbb{P}_{\mathcal{D}}[y=\tilde{y},s_*]}{\sum_{\mathcal{D}\in\{\mathcal{D}_b,\mathcal{D}_a\}}\pi_{\mathcal{D}}\mathbb{P}_{\mathcal{D}}[y=1-\tilde{y},s_*]}.\end{aligned}$$

With a simple algebraic manipulation where $y = 1$ and $s_* = 1$ (resp. for $y = 0$, $s_* = 1$), we have that

$$w_* + b_* = \log\frac{\mathbb{P}[y=1,s_*=1]}{\mathbb{P}[y=0,s_*=1]} \tag{4.12}$$

and for $y = 0$ and $s_* = 0$ (resp. for $y = 1$, $s_* = 0$)

$$w_* - b_* = \log\frac{\mathbb{P}[y=0,s_*=0]}{\mathbb{P}[y=1,s_*=0]}. \tag{4.13}$$

Combining Equation 4.12 and Equation 4.13 we have

$$\begin{aligned}w_* &= \frac{1}{2}\log\frac{\mathbb{P}[y=1,s_*=1]}{\mathbb{P}[y=0,s_*=1]}\frac{\mathbb{P}[y=0,s_*=0]}{\mathbb{P}[y=1,s_*=0]}\\ &\overset{(*)}{=}\frac{1}{2}\log\frac{\big(\sum_{\mathcal{D}\in\{\mathcal{D}_b,\mathcal{D}_a\}}\pi_{\mathcal{D}}\mathbb{P}_{\mathcal{D}}[y=1]\alpha_{*,\mathcal{D}}\big)\big(\sum_{\mathcal{D}\in\{\mathcal{D}_b,\mathcal{D}_a\}}\pi_{\mathcal{D}}\mathbb{P}_{\mathcal{D}}[y=0]\alpha_{*,\mathcal{D}}\big)}{\big(\sum_{\mathcal{D}\in\{\mathcal{D}_b,\mathcal{D}_a\}}\pi_{\mathcal{D}}\mathbb{P}_{\mathcal{D}}[y=1](1-\alpha_{*,\mathcal{D}})\big)\big(\sum_{\mathcal{D}\in\{\mathcal{D}_b,\mathcal{D}_a\}}\pi_{\mathcal{D}}\mathbb{P}_{\mathcal{D}}[y=0](1-\alpha_{*,\mathcal{D}})\big)}\end{aligned} \tag{4.14}$$

where (*) follows from that $\mathbb{P}_{\mathcal{D}}[y = s_*|y] = \alpha_{*,\mathcal{D}}$ and $\mathbb{P}_{\mathcal{D}}[y \neq s_*|y] = 1 - \alpha_{*,\mathcal{D}}$ for $\mathcal{D} \in \{\mathcal{D}_b, \mathcal{D}_a\}$.

Similarly, for b_* we have

$$\begin{aligned} b_* &= \frac{1}{2}\log\frac{\mathbb{P}[y=1,s_*=1]}{\mathbb{P}[y=0,s_*=1]}\frac{\mathbb{P}[y=1,s_*=0]}{\mathbb{P}[y=0,s_*=0]} \\ &= \frac{1}{2}\log\frac{\left(\sum_{\mathcal{D}\in\{\mathcal{D}_b,\mathcal{D}_a\}}\pi_{\mathcal{D}}\mathbb{P}_{\mathcal{D}}[y=1]\alpha_{*,\mathcal{D}}\right)\left(\sum_{\mathcal{D}\in\{\mathcal{D}_b,\mathcal{D}_a\}}\pi_{\mathcal{D}}\mathbb{P}_{\mathcal{D}}[y=1](1-\alpha_{*,\mathcal{D}})\right)}{\left(\sum_{\mathcal{D}\in\{\mathcal{D}_b,\mathcal{D}_a\}}\pi_{\mathcal{D}}\mathbb{P}_{\mathcal{D}}[y=0](1-\alpha_{*,\mathcal{D}})\right)\left(\sum_{\mathcal{D}\in\{\mathcal{D}_b,\mathcal{D}_a\}}\pi_{\mathcal{D}}\mathbb{P}_{\mathcal{D}}[y=0]\alpha_{*,\mathcal{D}}\right)}. \end{aligned} \tag{4.15}$$

Finally, noting that, for all $\tilde{y} \in \mathcal{Y}$, we have

$$\begin{aligned} \sum_{\mathcal{D}\in\{\mathcal{D}_b,\mathcal{D}_a\}}\pi_{\mathcal{D}}\mathbb{P}_{\mathcal{D}}[y=\tilde{y}]\alpha_{*,\mathcal{D}} &\geq {}_{\wedge}\alpha_* \sum_{\mathcal{D}\in\{\mathcal{D}_b,\mathcal{D}_a\}}\pi_{\mathcal{D}}\mathbb{P}_{\mathcal{D}}[y=\tilde{y}] = {}_{\wedge}\alpha_*\mathbb{P}[y=\tilde{y}] \\ \sum_{\mathcal{D}\in\{\mathcal{D}_b,\mathcal{D}_a\}}\pi_{\mathcal{D}}\mathbb{P}_{\mathcal{D}}[y=\tilde{y}]\alpha_{*,\mathcal{D}} &\leq {}_{\vee}\alpha_* \sum_{\mathcal{D}\in\{\mathcal{D}_b,\mathcal{D}_a\}}\pi_{\mathcal{D}}\mathbb{P}_{\mathcal{D}}[y=\tilde{y}] = {}_{\vee}\alpha_*\mathbb{P}[y=\tilde{y}]. \end{aligned}$$

Using the above relation as well as Equation 4.14 and Equation 4.15, the weight and bias of the main task model, w_* and b_*, can therefore be bounded as

$$\log\frac{{}_{\wedge}\alpha_*}{1-{}_{\wedge}\alpha_*} \leq w_* \leq \log\frac{{}_{\vee}\alpha_*}{1-{}_{\vee}\alpha_*} \qquad \text{and} \tag{4.16}$$

$$r_y + \log\frac{{}_{\wedge}\alpha_*(1-{}_{\vee}\alpha_*)}{(1-{}_{\wedge}\alpha_*){}_{\vee}\alpha_*} \leq b_* \leq r_y + \log\frac{{}_{\vee}\alpha_*(1-{}_{\wedge}\alpha_*)}{(1-{}_{\vee}\alpha_*){}_{\wedge}\alpha_*}. \tag{4.17}$$

To distinguish the effect of class imbalance in our analysis, we define $b_{**} := b_* - r_y$.

- *Permissive models*: For permissive model, we have

$$\log\frac{\mathbb{P}[s_i|y=\tilde{y},s_{I_i}]}{\mathbb{P}[s_i|y=1-\tilde{y},s_{I_i}]} = (2\tilde{y}-1)(w_i s_i + b_i).$$

Therefore

$$\log \frac{\mathbb{P}[s_i|y=\tilde{y},s_{I_i}]}{\mathbb{P}[s_i|y=1-\tilde{y},s_{I_i}]} = \log \frac{\frac{\mathbb{P}[s_i,y=\tilde{y},s_{I_i}]}{\mathbb{P}[y=\tilde{y},s_{I_i}]}}{\frac{\mathbb{P}[s_i,y=1-\tilde{y},s_{I_i}]}{\mathbb{P}[y=1-\tilde{y},s_{I_i}]}} \overset{(*)}{=} \log \frac{\frac{\sum_{\mathcal{D}\in\{\mathcal{D}_b,\mathcal{D}_a\}}\pi_{\mathcal{D}}\mathbb{P}_{\mathcal{D}}[y=\tilde{y},s_{I_i}]\mathbb{P}_{\mathcal{D}}[s_i|y=\tilde{y}]}{\sum_{\mathcal{D}\in\{\mathcal{D}_b,\mathcal{D}_a\}}\pi_{\mathcal{D}}\mathbb{P}_{\mathcal{D}}[y=\tilde{y},s_{I_i}]}}{\frac{\sum_{\mathcal{D}\in\{\mathcal{D}_b,\mathcal{D}_a\}}\pi_{\mathcal{D}}\mathbb{P}_{\mathcal{D}}[y=1-\tilde{y},s_{I_i}]\mathbb{P}_{\mathcal{D}}[s_i|y=1-\tilde{y}]}{\sum_{\mathcal{D}\in\{\mathcal{D}_b,\mathcal{D}_a\}}\pi_{\mathcal{D}}\mathbb{P}_{\mathcal{D}}[y=1-\tilde{y},s_{I_i}]}}$$

where (*) follows from the conditional independence assumption.

Let $\tilde{y}=1$. Therefore, for $s_i=1$ we have

$$\min_{\mathcal{D}} \alpha_{i,\mathcal{D}} = {}_{\wedge}\alpha_i \le \frac{\sum_{\mathcal{D}\in\{\mathcal{D}_b,\mathcal{D}_a\}}\pi_{\mathcal{D}}\mathbb{P}_{\mathcal{D}}[y=\tilde{y},s_{I_i}]\mathbb{P}_{\mathcal{D}}[s_i|y=\tilde{y}]}{\sum_{\mathcal{D}\in\{\mathcal{D}_b,\mathcal{D}_a\}}\pi_{\mathcal{D}}\mathbb{P}_{\mathcal{D}}[y=\tilde{y},s_{I_i}]} \le \max_{\mathcal{D}} \alpha_{i,\mathcal{D}} = {}_{\vee}\alpha_i$$

$$\min_{\mathcal{D}} \varepsilon_{i,\mathcal{D}} = {}_{\wedge}\varepsilon_i \le \frac{\sum_{\mathcal{D}\in\{\mathcal{D}_b,\mathcal{D}_a\}}\pi_{\mathcal{D}}\mathbb{P}_{\mathcal{D}}[y=1-\tilde{y},s_{I_i}]\mathbb{P}_{\mathcal{D}}[s_i|y=1-\tilde{y}]}{\sum_{\mathcal{D}\in\{\mathcal{D}_b,\mathcal{D}_a\}}\pi_{\mathcal{D}}\mathbb{P}_{\mathcal{D}}[y=1-\tilde{y},s_{I_i}]} \le \max_{\mathcal{D}} \varepsilon_{i,\mathcal{D}} = {}_{\vee}\varepsilon_i.$$

Above bounds finally lead to

$$\log \frac{{}_{\wedge}\alpha_i}{{}_{\vee}\varepsilon_i} \le \log \frac{\mathbb{P}[s_i|y=\tilde{y},s_{I_i}]}{\mathbb{P}[s_i|y=1-\tilde{y},s_{I_i}]} = w_i + b_i \le \log \frac{{}_{\vee}\alpha_i}{{}_{\wedge}\varepsilon_i}. \tag{4.18}$$

Next, we let $s_i=0$. Repeating the same technique above, we have

$$\frac{\sum_{\mathcal{D}\in\{\mathcal{D}_b,\mathcal{D}_a\}}\pi_{\mathcal{D}}\mathbb{P}_{\mathcal{D}}[y=\tilde{y},s_{I_i}]\mathbb{P}_{\mathcal{D}}[s_i|y=\tilde{y}]}{\sum_{\mathcal{D}\in\{\mathcal{D}_b,\mathcal{D}_a\}}\pi_{\mathcal{D}}\mathbb{P}_{\mathcal{D}}[y=\tilde{y},s_{I_i}]} \ge \min_{\mathcal{D}} 1-\alpha_{i,\mathcal{D}} = 1-{}_{\vee}\alpha_i$$
$$\frac{\sum_{\mathcal{D}\in\{\mathcal{D}_b,\mathcal{D}_a\}}\pi_{\mathcal{D}}\mathbb{P}_{\mathcal{D}}[y=\tilde{y},s_{I_i}]\mathbb{P}_{\mathcal{D}}[s_i|y=\tilde{y}]}{\sum_{\mathcal{D}\in\{\mathcal{D}_b,\mathcal{D}_a\}}\pi_{\mathcal{D}}\mathbb{P}_{\mathcal{D}}[y=\tilde{y},s_{I_i}]} \le \max_{\mathcal{D}} 1-\alpha_{i,\mathcal{D}} = 1-{}_{\wedge}\alpha_i$$

and

$$\frac{\sum_{\mathcal{D}\in\{\mathcal{D}_b,\mathcal{D}_a\}}\pi_{\mathcal{D}}\mathbb{P}_{\mathcal{D}}[y=1-\tilde{y},s_{I_i}]\mathbb{P}_{\mathcal{D}}[s_i|y=1-\tilde{y}]}{\sum_{\mathcal{D}\in\{\mathcal{D}_b,\mathcal{D}_a\}}\pi_{\mathcal{D}}\mathbb{P}_{\mathcal{D}}[y=1-\tilde{y},s_{I_i}]} \ge \min_{\mathcal{D}} 1-\varepsilon_{i,\mathcal{D}} = 1-{}_{\vee}\varepsilon_i$$
$$\frac{\sum_{\mathcal{D}\in\{\mathcal{D}_b,\mathcal{D}_a\}}\pi_{\mathcal{D}}\mathbb{P}_{\mathcal{D}}[y=1-\tilde{y},s_{I_i}]\mathbb{P}_{\mathcal{D}}[s_i|y=1-\tilde{y}]}{\sum_{\mathcal{D}\in\{\mathcal{D}_b,\mathcal{D}_a\}}\pi_{\mathcal{D}}\mathbb{P}_{\mathcal{D}}[y=1-\tilde{y},s_{I_i}]} \le \max_{\mathcal{D}} 1-\varepsilon_{i,\mathcal{D}} = 1-{}_{\wedge}\varepsilon_i.$$

Above bounds finally lead to

$$\log \frac{1 - {}_{\vee}\alpha_i}{1 - {}_{\wedge}\varepsilon_i} \leq \log \frac{\mathbb{P}[s_i|y = \tilde{y}, s_{I_i}]}{\mathbb{P}[s_i|y = 1 - \tilde{y}, s_{I_i}]} = b_i \leq \log \frac{1 - {}_{\wedge}\alpha_i}{1 - {}_{\vee}\varepsilon_i}. \tag{4.19}$$

Note that the same conclusion can be drawn for $\tilde{y} = 0$.

- *preventive models*: For preventive model, we have

$$\log \frac{\mathbb{P}[s_j|y = \tilde{y}, s_I, s_{J_j}]}{\mathbb{P}[s_j|y = 1 - \tilde{y}, s_I, s_{J_j}]} = -(2\tilde{y} - 1)(w_j(1 - s_j) + b_j).$$

Then

$$\log \frac{\mathbb{P}[s_j|y = \tilde{y}, s_I, s_{J_j}]}{\mathbb{P}[s_j|y = 1 - \tilde{y}, s_I, s_{J_j}]} = \log \frac{\frac{\mathbb{P}[s_j, y=\tilde{y}, s_I, s_{J_j}]}{\mathbb{P}[y=\tilde{y}, s_I, s_{J_j}]}}{\frac{\mathbb{P}[s_j, y=1-\tilde{y}, s_I, s_{J_j}]}{\mathbb{P}[y=1-\tilde{y}, s_I, s_{J_j}]}}$$

$$\overset{(*)}{=} \log \frac{\frac{\sum_{\mathcal{D}\in\{\mathcal{D}_b,\mathcal{D}_a\}} \pi_{\mathcal{D}} \mathbb{P}_{\mathcal{D}}[y=\tilde{y}, s_I, s_{J_j}] \mathbb{P}_{\mathcal{D}}[s_j|y=\tilde{y}]}{\sum_{\mathcal{D}\in\{\mathcal{D}_b,\mathcal{D}_a\}} \pi_{\mathcal{D}} \mathbb{P}_{\mathcal{D}}[y=\tilde{y}, s_I, s_{J_j}]}}{\frac{\sum_{\mathcal{D}\in\{\mathcal{D}_b,\mathcal{D}_a\}} \pi_{\mathcal{D}} \mathbb{P}_{\mathcal{D}}[y=1-\tilde{y}, s_I, s_{J_j}] \mathbb{P}_{\mathcal{D}}[s_j|y=1-\tilde{y}]}{\sum_{\mathcal{D}\in\{\mathcal{D}_b,\mathcal{D}_a\}} \pi_{\mathcal{D}} \mathbb{P}_{\mathcal{D}}[y=1-\tilde{y}, s_I, s_{J_j}]}}$$

where (*) follows from the conditional independence assumption.
Let $\tilde{y} = 0$. Therefore, for $s_j = 0$ we have

$$\min_{\mathcal{D}} \alpha_{j,\mathcal{D}} = {}_{\wedge}\alpha_j \leq \frac{\sum_{\mathcal{D}\in\{\mathcal{D}_b,\mathcal{D}_a\}} \pi_{\mathcal{D}} \mathbb{P}_{\mathcal{D}}[y = \tilde{y}, s_I, s_{J_j}] \mathbb{P}_{\mathcal{D}}[s_j|y = \tilde{y}]}{\sum_{\mathcal{D}\in\{\mathcal{D}_b,\mathcal{D}_a\}} \pi_{\mathcal{D}} \mathbb{P}_{\mathcal{D}}[y = \tilde{y}, s_I, s_{J_j}]} \leq \max_{\mathcal{D}} \alpha_{j,\mathcal{D}} = {}_{\vee}\alpha_j$$

and

$$\min_{\mathcal{D}} \varepsilon_{j,\mathcal{D}} = {}_{\wedge}\varepsilon_j \leq \frac{\sum_{\mathcal{D}\in\{\mathcal{D}_b,\mathcal{D}_a\}} \pi_{\mathcal{D}} \mathbb{P}_{\mathcal{D}}[y = 1 - \tilde{y}, s_I, s_{J_j}] \mathbb{P}_{\mathcal{D}}[s_j|y = 1 - \tilde{y}]}{\sum_{\mathcal{D}\in\{\mathcal{D}_b,\mathcal{D}_a\}} \pi_{\mathcal{D}} \mathbb{P}_{\mathcal{D}}[y = 1 - \tilde{y}, s_I, s_{J_j}]} \leq \max_{\mathcal{D}} \varepsilon_{j,\mathcal{D}} = {}_{\vee}\varepsilon_j.$$

Above bounds finally lead to

$$\log \frac{{}_{\wedge}\alpha_j}{{}_{\vee}\varepsilon_j} \leq \log \frac{\mathbb{P}[s_j|y = \tilde{y}, s_I, s_{J_j}]}{\mathbb{P}[s_j|y = 1 - \tilde{y}, s_I, s_{J_j}]} = w_j + b_j \leq \log \frac{{}_{\vee}\alpha_j}{{}_{\wedge}\varepsilon_j}. \tag{4.20}$$

Next, we let $s_j = 1$. Repeating the same technique above, we have

$$\frac{\sum_{\mathcal{D}\in\{\mathcal{D}_b,\mathcal{D}_a\}} \pi_{\mathcal{D}} \mathbb{P}_{\mathcal{D}}[y=\tilde{y}, s_I, s_{J_j}] \mathbb{P}_{\mathcal{D}}[s_j | y = \tilde{y}]}{\sum_{\mathcal{D}\in\{\mathcal{D}_b,\mathcal{D}_a\}} \pi_{\mathcal{D}} \mathbb{P}_{\mathcal{D}}[y=\tilde{y}, s_I, s_{J_j}]} \geq \min_{\mathcal{D}} 1 - \alpha_{j,\mathcal{D}} = 1 - {}_{\vee}\alpha_j$$
$$\frac{\sum_{\mathcal{D}\in\{\mathcal{D}_b,\mathcal{D}_a\}} \pi_{\mathcal{D}} \mathbb{P}_{\mathcal{D}}[y=\tilde{y}, s_I, s_{J_j}] \mathbb{P}_{\mathcal{D}}[s_j | y = \tilde{y}]}{\sum_{\mathcal{D}\in\{\mathcal{D}_b,\mathcal{D}_a\}} \pi_{\mathcal{D}} \mathbb{P}_{\mathcal{D}}[y=\tilde{y}, s_I, s_{J_j}]} \leq \max_{\mathcal{D}} 1 - \alpha_{j,\mathcal{D}} = 1 - {}_{\wedge}\alpha_j$$

and

$$\frac{\sum_{\mathcal{D}\in\{\mathcal{D}_b,\mathcal{D}_a\}} \pi_{\mathcal{D}} \mathbb{P}_{\mathcal{D}}[y=1-\tilde{y}, s_I, s_{J_j}] \mathbb{P}_{\mathcal{D}}[s_j | y = 1-\tilde{y}]}{\sum_{\mathcal{D}\in\{\mathcal{D}_b,\mathcal{D}_a\}} \pi_{\mathcal{D}} \mathbb{P}_{\mathcal{D}}[y=1-\tilde{y}, s_I, s_{J_j}]} \geq \min_{\mathcal{D}} 1 - \varepsilon_{j,\mathcal{D}} = 1 - {}_{\vee}\varepsilon_j$$
$$\frac{\sum_{\mathcal{D}\in\{\mathcal{D}_b,\mathcal{D}_a\}} \pi_{\mathcal{D}} \mathbb{P}_{\mathcal{D}}[y=1-\tilde{y}, s_I, s_{J_j}] \mathbb{P}_{\mathcal{D}}[s_j | y = 1-\tilde{y}]}{\sum_{\mathcal{D}\in\{\mathcal{D}_b,\mathcal{D}_a\}} \pi_{\mathcal{D}} \mathbb{P}_{\mathcal{D}}[y=1-\tilde{y}, s_I, s_{J_j}]} \leq \max_{\mathcal{D}} 1 - \varepsilon_{j,\mathcal{D}} = 1 - {}_{\wedge}\varepsilon_j.$$

Similarly as in permissive models, above bounds lead to

$$\log \frac{1 - {}_{\vee}\alpha_j}{1 - {}_{\wedge}\varepsilon_j} \leq \log \frac{\mathbb{P}[s_j | y = \tilde{y}, s_I, s_{J_j}]}{\mathbb{P}[s_j | y = 1 - \tilde{y}, s_I, s_{J_j}]} = b_j \leq \log \frac{1 - {}_{\wedge}\alpha_j}{1 - {}_{\vee}\varepsilon_j}. \tag{4.21}$$

The same conclusion can be drawn for $\tilde{y} = 1$.

STEP 2: DECOMPOSITION OF $\Delta_w(y, s_*, s_{\mathcal{I}}, s_{\mathcal{J}})$ Next, we recall Remark 3 and present a lower bound for $\Delta_w(y, s_*, s_{\mathcal{I}}, s_{\mathcal{J}})$ that decomposes $\Delta_w(y, s_*, s_{\mathcal{I}}, s_{\mathcal{J}})$ into its additive components such that

$$\begin{aligned}\Delta_w(y, s_*, s_{\mathcal{I}}, s_{\mathcal{J}}) &= (2\tilde{y} - 1)\Big(b + w_*(2s_* - 1) + \sum_{i\in\mathcal{I}} w_i s_i - \sum_{j\in\mathcal{J}} w_j(1 - s_j)\Big) \\ &= (2\tilde{y} - 1)\Big(w_*(2s_* - 1) + \sum_{i\in\mathcal{I}} (w_i s_i + b_i) - \sum_{j\in\mathcal{J}} (w_j(1 - s_j) + b_j)\Big).\end{aligned}$$

Next, we analyze

$$\mathbb{P}_{\mathcal{D}}\big[\langle w, \mathbf{f}_y(s_*, s_{\mathcal{I}}, s_{\mathcal{J}}) - \mathbf{f}_{1-y}(s_*, s_{\mathcal{I}}, s_{\mathcal{J}})\rangle | y\big] = \mathbb{P}_{\mathcal{D}}[\Delta_w(y, s_*, s_{\mathcal{I}}, s_{\mathcal{J}}) | y].$$

Note that $\mathbb{P}_{\mathcal{D}}[s_*|y] = \alpha_{*,\mathcal{D}}$ if $s_* = y$. Therefore, $\mathbb{P}_{\mathcal{D}}[s_* = 1|y = 1] = \alpha_{*,\mathcal{D}}$ and $\mathbb{P}_{\mathcal{D}}[s_* = 0|y = 1] = 1 - \alpha_{*,\mathcal{D}}$. Similarly, $\mathbb{P}_{\mathcal{D}}[s_* = 0|y = 0] = \alpha_{*,\mathcal{D}}$ and $\mathbb{P}_{\mathcal{D}}[s_* = 1|y = 0] = 1 - \alpha_{*,\mathcal{D}}$. Thus

$$\mathbb{P}_{\mathcal{D}}[(2\tilde{y}-1)\big(w_*(2s_*-1)+b_*\big)|y] \overset{(*)}{=} \mathbb{P}_{\mathcal{D}}[w_*(2s_{**}-1)+b_{**}+(2\tilde{y}-1)r_y|y]$$

where s_{**} satisfies $\mathbb{P}_{\mathcal{D}}[s_{**} = 1] = \alpha_{*,\mathcal{D}}$ and $\mathbb{P}_{\mathcal{D}}[s_{**} = 0] = 1 - \alpha_{*,\mathcal{D}}$. Note that (*) stems from the symmetry of s_* and b_{**} with respect to y. To reduce exposition, we will stick to s_* notation and continue to refer to s_{**} as s_*. Hence, we define $d_{*,\mathcal{D}}$ as

$$d_{*,\mathcal{D}} := w_*(2s_*-1) + b_{**} \tag{4.22}$$

where $b_{**} := b_* - r_y$ as defined earlier. Therefore, the contribution of the main task model in the majority voting random variable $\Delta_w(y, s_*, s_{\mathcal{I}}, s_{\mathcal{J}})$ will be

$$d_{*,\mathcal{D}} + (2y-1)r_y. \tag{4.23}$$

Next, we analyze the auxiliary model predictions. For $y = 1$,

$$\begin{aligned}\mathbb{P}_{\mathcal{D}}\big[(2y-1)\big(\sum_{i\in\mathcal{I}}(w_is_i+b_i) - \sum_{j\in\mathcal{J}}(w_j(1-s_j)+b_j)\big)|y\big]\\ = \mathbb{P}_{\mathcal{D}}\big[\sum_{i\in\mathcal{I}}(w_is_i+b_i) - \sum_{j\in\mathcal{J}}(w_j(1-s_j)+b_j)|y=1\big]\end{aligned}$$

where, on the right hand side, we have $\mathbb{P}_{\mathcal{D}}[s_i = 1|y = \tilde{y}] = \alpha_{i,\mathcal{D}}$ and $\mathbb{P}_{\mathcal{D}}[1 - s_j = 1|y = \tilde{y}] = \varepsilon_{j,\mathcal{D}}$ for $\tilde{y} = 1$ over distribution $\mathcal{D} \in \{\mathcal{D}_b, \mathcal{D}_a\}$. Therefore, we define $d_{\mathcal{I},\mathcal{D}}$ as

$$\begin{aligned}&\big(\Delta_w(y, s_*, s_{\mathcal{I}}, s_{\mathcal{J}}) - d_{*,\mathcal{D}} - r_y|y=1\big)\\ &= \sum_{i\in\mathcal{I}}(w_is_i+b_i) - \sum_{j\in\mathcal{J}}(w_j(1-s_j)+b_j) := d_{\mathcal{I},\mathcal{D}}.\end{aligned} \tag{4.24}$$

Using the same strategy for $y = 0$, we define $d_{\mathcal{J},\mathcal{D}}$ as

$$\begin{aligned}&\big(\Delta_w(y, s_*, s_{\mathcal{I}}, s_{\mathcal{J}}) - d_{*,\mathcal{D}} + r_y | y = 0\big)\\&= \sum_{j \in \mathcal{J}} (w_j(1 - s_j) + b_j) - \sum_{i \in \mathcal{I}} (w_i s_i + b_i) := d_{\mathcal{J},\mathcal{D}}\end{aligned} \tag{4.25}$$

where, on the right hand side, we have $\mathbb{P}_{\mathcal{D}}[1 - s_j = 1 | y = \tilde{y}] = \alpha_{j,\mathcal{D}}$ and $\mathbb{P}_{\mathcal{D}}[s_i = 1 | y = \tilde{y}] = \varepsilon_{i,\mathcal{D}}$ for $\tilde{y} = 0$ over $\mathcal{D} \in \{\mathcal{D}_b, \mathcal{D}_a\}$.

Combining Equation 4.23, Equation 4.24 and Equation 4.25, we have

$$\big(\Delta_w(y, s_*, s_{\mathcal{I}}, s_{\mathcal{J}}) | y\big) = d_{*\mathcal{D}} + y d_{\mathcal{I},\mathcal{D}} + (1 - y) d_{\mathcal{J},\mathcal{D}} + (2y - 1) r_y. \tag{4.26}$$

FINAL STEP: $\mathbb{E}_{s_*, s_{\mathcal{I}}, s_{\mathcal{J}}}[\Delta_w(y, s_*, s_{\mathcal{I}}, s_{\mathcal{J}}) | y]$ We express $\Delta_w(y, s_*, s_{\mathcal{I}}, s_{\mathcal{J}}) | y$ in terms of y and a function of model predictions thus far. In this step, using the bounds on the optimal parameters in the first step as well as the decomposition introduced in the second step, we derive a lower bound for the $\mathbb{E}_{s_*, s_{\mathcal{I}}, s_{\mathcal{J}}}[\Delta_w(y, s_*, s_{\mathcal{I}}, s_{\mathcal{J}}) | y]$. Towards that, we lower bound the expected value of $d_{*,\mathcal{D}}$, $d_{\mathcal{I},\mathcal{D}}$ and $d_{\mathcal{J},\mathcal{D}}$ individually.

- $\mathbb{E}_{s_*}[d_{*,\mathcal{D}}]$: For the main task model, we have

$$\mathbb{E}_{s_*}[d_{*,\mathcal{D}}] = \mathbb{E}_{s_*}[w_*(2s_* - 1) + b_{**}] \tag{4.27}$$

over distribution $\mathcal{D} \in \{\mathcal{D}_b, \mathcal{D}_a\}$ and w_*. One can infer from Equation 4.16 and Equation 4.17 for $b_{**} = b_* - r_y$ that

$$\begin{aligned}\mathbb{E}_{s_*}[d_{*,\mathcal{D}}] &= \mathbb{E}_{s_*}[w_*(2s_* - 1) + (2y - 1) b_{**}]\\&\geq \alpha_{*,\mathcal{D}} \log \frac{{}_{\wedge}\alpha_*}{1 - {}_{\wedge}\alpha_*} + (1 - \alpha_{*,\mathcal{D}}) \log \frac{1 - {}_{\vee}\alpha_*}{{}_{\vee}\alpha_*} := \mu_{d_{*,\mathcal{D}}}.\end{aligned} \tag{4.28}$$

- $\mathbb{E}_{s_{\mathcal{I}},s_{\mathcal{J}}}[d_{\mathcal{I},\mathcal{D}}]$: For the permissive models, we have

$$\begin{aligned}\mathbb{E}_{s_{\mathcal{I}},s_{\mathcal{J}}}[d_{\mathcal{I},\mathcal{D}}] &= \mathbb{E}_{s_{\mathcal{I}},s_{\mathcal{J}}}\Big[\sum_{i\in\mathcal{I}}(w_i s_i + b_i) - \sum_{j\in\mathcal{J}}(w_j(1-s_j)+b_j)\Big] \\ &= \mathbb{E}_{s_{\mathcal{I}}}\Big[\sum_{i\in\mathcal{I}}(w_i s_i + b_i)\Big] - \mathbb{E}_{s_{\mathcal{J}}}\Big[\sum_{j\in\mathcal{J}}(w_j(1-s_j)+b_j)\Big].\end{aligned}$$

Note that $w_i s_i + b_i = w_i + b_i$ with probability $\alpha_{i,\mathcal{D}}$ and $w_i s_i + b_i = b_i$ otherwise. Therefore, using Equation 4.18 and Equation 4.19 we lower bound $\mathbb{E}_{s_{\mathcal{I}}}\Big[\sum_{i\in\mathcal{I}}(w_i s_i + b_i)\Big]$ as

$$\mathbb{E}_{s_{\mathcal{I}}}\Big[\sum_{i\in\mathcal{I}}(w_i s_i + b_i)\Big] \geq \sum_{i\in\mathcal{I}} \alpha_{i,\mathcal{D}} \log\frac{{}_{\wedge}\alpha_i}{{}_{\vee}\varepsilon_i} + (1-\alpha_{i,\mathcal{D}})\log\frac{1-{}_{\vee}\alpha_i}{1-{}_{\wedge}\varepsilon_i}.$$

Similarly, $-\mathbb{E}_{s_{\mathcal{J}}}\Big[\sum_{j\in\mathcal{J}}(w_j(1-s_j)+b_j)\Big]$ can be lower bounded as

$$-\mathbb{E}_{s_{\mathcal{J}}}\Big[\sum_{j\in\mathcal{J}}(w_j(1-s_j)+b_j)\Big] \geq -\sum_{j\in\mathcal{J}} \varepsilon_{j,\mathcal{D}} \log\frac{{}_{\vee}\alpha_j}{{}_{\wedge}\varepsilon_j} + (1-\varepsilon_{j,\mathcal{D}})\log\frac{1-{}_{\wedge}\alpha_j}{1-{}_{\vee}\varepsilon_j}.$$

Combining above result, we have

$$\begin{aligned}&\mathbb{E}_{s_{\mathcal{I}},s_{\mathcal{J}}}[d_{\mathcal{I},\mathcal{D}}] \\ &\geq \sum_{i\in\mathcal{I}} \alpha_{i,\mathcal{D}} \log\frac{{}_{\wedge}\alpha_i}{{}_{\vee}\varepsilon_i} + (1-\alpha_{i,\mathcal{D}})\log\frac{1-{}_{\vee}\alpha_i}{1-{}_{\wedge}\varepsilon_i} \\ &\quad - \sum_{j\in\mathcal{J}} \varepsilon_{j,\mathcal{D}} \log\frac{{}_{\vee}\alpha_j}{{}_{\wedge}\varepsilon_j} - (1-\varepsilon_{j,\mathcal{D}})\log\frac{1-{}_{\wedge}\alpha_j}{1-{}_{\vee}\varepsilon_j} := \mu_{\mathcal{I},\mathcal{D}}.\end{aligned} \tag{4.29}$$

- $\mathbb{E}_{s_{\mathcal{I}},s_{\mathcal{J}}}[d_{\mathcal{J},\mathcal{D}}]$: Following the same strategy to that of $\mathbb{E}_{s_{\mathcal{I}},s_{\mathcal{J}}}[d_{\mathcal{I},\mathcal{D}}]$, we have

$$\begin{aligned}&\mathbb{E}_{s_{\mathcal{I}},s_{\mathcal{J}}}[d_{\mathcal{J},\mathcal{D}}] \\ &\geq \sum_{j\in\mathcal{J}} \alpha_{j,\mathcal{D}} \log\frac{{}_{\wedge}\alpha_j}{{}_{\vee}\varepsilon_j} + (1-\alpha_{j,\mathcal{D}})\log\frac{1-{}_{\vee}\alpha_j}{1-{}_{\wedge}\varepsilon_j} \\ &- \sum_{i\in\mathcal{I}} \varepsilon_{i,\mathcal{D}} \log\frac{{}_{\vee}\alpha_i}{{}_{\wedge}\varepsilon_i} - (1-\varepsilon_{i,\mathcal{D}})\log\frac{1-{}_{\wedge}\alpha_i}{1-{}_{\vee}\varepsilon_i} := \mu_{\mathcal{J},\mathcal{D}}.\end{aligned} \tag{4.30}$$

Finally, combining Equation 4.26, Equation 4.27, Equation 4.29, Equation 4.30 we conclude

$$\begin{aligned}\mathbb{E}_{s_*,s_\mathcal{I},s_\mathcal{J}}[\Delta_w(y,s_*,s_\mathcal{I},s_\mathcal{J})|y] &= \mathbb{E}_{s_*,s_\mathcal{I},s_\mathcal{J}}[d_{*,\mathcal{D}} + y d_{\mathcal{I},\mathcal{D}} + (1-y) d_{\mathcal{J},\mathcal{D}} + (2y-1) r_y] \\ &\geq \mu_{*,\mathcal{D}} + y\mu_{\mathcal{I},\mathcal{D}} + (1-y)\mu_{\mathcal{J},\mathcal{D}} + (2y-1) r_y := \mu_{y,\mathcal{D}}.\end{aligned} \tag{4.31}$$

The proof is thus completed. □

4.5.4 *Proof of Theorem 9*

We start by recalling our main theorem.

Theorem (Recall). *For $y \in \mathcal{Y}$ and $\mathcal{D} \in \{\mathcal{D}_b, \mathcal{D}_a\}$, let $\mu_{y,\mathcal{D}}$ be defined as in Lemma 8. Suppose that the* modeling assumption *holds, and suppose that $\mu_{d_{\mathcal{K},\mathcal{D}}} > 0$, for all $\mathcal{K} \in \{\mathcal{I}, \mathcal{J}\}$ and $\mathcal{D} \in \{\mathcal{D}_b, \mathcal{D}_a\}$. Then*

$$\mathcal{A}^{\text{KEMLP}} \geq 1 - \mathbb{E}_{\mu_{y,\mathcal{D}}}[\exp\left(-2\mu_{y,\mathcal{D}}^2/v^2\right)], \tag{4.32}$$

where v^2 is the variance upper bound to $\mathbb{P}[o = y|y]$ with

$$v^2 = 4\left(\log \frac{{}_\vee\alpha_*}{1 - {}_\wedge\alpha_*}\right)^2 + \sum_{k\in\mathcal{I}\cup\mathcal{J}} \left(\log \frac{{}_\vee\alpha_k(1 - {}_\wedge\epsilon_k)}{{}_\wedge\epsilon_k(1 - {}_\vee\alpha_k)}\right)^2.$$

Proof of Theorem 9. Recall that we define weighted robust accuracy of KEMLP as

$$\mathcal{A}^{\text{KEMLP}} = \mathbb{E}_{\mathcal{D}\sim\{\mathcal{D}_a,\mathcal{D}_b\}}\mathbb{E}_{y\sim\mathcal{Y}}\left[\mathbb{P}_\mathcal{D}[o = y|y, \mathbf{w}]\right].$$

The weighted accuracy definition comes from the latent variable z. That is, $\mathcal{A}^{\text{KEMLP}} = \mathbb{P}[o = y|\mathbf{w}] = \sum_{z\in\{0,1\}} \mathbb{P}[o = y|z, \mathbf{w}]$ where $\mathbb{P}[o = y|z = 0, \mathbf{w}] = \mathbb{P}_{\mathcal{D}_b}[o = y|\mathbf{w}]$ and $\mathbb{P}[o = y|z = 1, \mathbf{w}] = \mathbb{P}_{\mathcal{D}_a}[o = y|\mathbf{w}]$. Hence, $\mathcal{A}^{\text{KEMLP}} = \mathbb{E}_{\mathcal{D}\sim\{\mathcal{D}_b,\mathcal{D}_a\}}\left[\mathbb{P}_\mathcal{D}[o = y|\mathbf{w}]\right] = \mathbb{E}_{\mathcal{D}\sim\{\mathcal{D}_b,\mathcal{D}_a\}}\mathbb{E}_{y\sim\mathcal{Y}}\left[\mathbb{P}_\mathcal{D}[o = y|y, \mathbf{w}]\right]$.

Let w be the set of optimal parameters. Using Equation 4.9 and our inference rule, $\mathbb{P}_{\mathcal{D}}[o = y|y, \mathbf{w}]$ can be further expressed as

$$\begin{aligned}
&\mathbb{P}_{\mathcal{D}}[o = y|y, \mathbf{w}] \\
&= \mathbb{P}_{\mathcal{D}}\left[\sigma\left(\Delta_w(y, s_*, s_{\mathcal{I}}, s_{\mathcal{J}})\right) > 1/2|y\right] \\
&= \mathbb{P}_{\mathcal{D}}[\Delta_w(y, s_*, s_{\mathcal{I}}, s_{\mathcal{J}}) > 0|y] = 1 - \mathbb{P}_{\mathcal{D}}[\Delta_w(y, s_*, s_{\mathcal{I}}, s_{\mathcal{J}}) < 0|y]
\end{aligned}$$

For the rest of the proof, we will focus on bounding the term $\mathbb{P}_{\mathcal{D}}[\Delta_w(y, s_*, s_{\mathcal{I}}, s_{\mathcal{J}}) < 0|y]$, and $\mathcal{A}^{\text{KEMLP}}$ will follow from taking expectation of $1 - \mathbb{P}_{\mathcal{D}}[\Delta_w(y, s_*, s_{\mathcal{I}}, s_{\mathcal{J}}) < 0|y]$ over $\mathcal{D} \in \{\mathcal{D}_b, \mathcal{D}_a\}$ and $y \in \mathcal{Y}$.

Next, we recall the generalized bounded difference inequality as well as generalized Hoeffding's inequality (Geer, 2002). Note that the same result can be shown via Azuma's inequality for submartingale sequences (Azuma, 1967).

Theorem 11 (Azuma, 1967; Geer, 2002)**.** *Assume that X_t be a random variable with respect to filtration $\mathcal{F}_t$, and $\mathcal{L}_t$ and $\mathcal{U}_t$ be $\mathcal{F}_{t-1}$ measurable random variables such that*

$$\mathcal{L}_t \leq X_t - X_{t-1} \leq \mathcal{U}_t$$

where $\mathcal{L}_t < \mathcal{U}_t$ and $\mathcal{U}_t - \mathcal{L}_t \leq c_t$ almost surely. Therefore, for some $\varepsilon > 0$, one has

$$\begin{aligned}
\mathbb{P}(X_n - \mathbb{E}[X_n] < -\varepsilon) &\leq \exp\left(-\frac{2\varepsilon^2}{\sum_{t=[n]} c_t^2}\right) \\
\mathbb{P}(X_n - \mathbb{E}[X_n] > \varepsilon) &\leq \exp\left(-\frac{2\varepsilon^2}{\sum_{t=[n]} c_t^2}\right).
\end{aligned} \tag{4.33}$$

We now consider the random variable $\Delta_w(y, s_*, s_{\mathcal{I}}, s_{\mathcal{J}}) = d_{*,\mathcal{D}} + yd_{\mathcal{I},\mathcal{D}} + (1-y)d_{\mathcal{J},\mathcal{D}} + (2y-1)r_y$ that is meant to represent X_n in Theorem 11, where each increment is induced by a single model. We call $\Delta_w(y, s_*, s_{\mathcal{I}}, s_{\mathcal{J}})$ as $X_{1+|\mathcal{I}|+|\mathcal{J}|}$.

To prove compatibility of our setting with the Theorem 11, we present the following remark.

Remark 5 (Measurability of $X_{1+|\mathcal{I}|+|\mathcal{J}|}$ and the bounded differences)**.** *Let $y = 1$. We can write our random variable $X_{1+|\mathcal{I}|+|\mathcal{J}|} = \Delta_w(y, s_*, s_{\mathcal{I}}, s_{\mathcal{J}})$ as*

$$\big(\Delta_w(y, s_*, s_{\mathcal{I}}, s_{\mathcal{J}})|y = 1\big) = w_*(2s_* - 1) + b_* + \sum_{i\in\mathcal{I}}(w_i s_i + b_i) - \sum_{j\in\mathcal{J}}(w_j(1 - s_j) + b_j).$$

That is, we represent $\big(\Delta_w(y, s_*, s_{\mathcal{I}}, s_{\mathcal{J}})|y = 1\big)$ as a random process with a total of $1 + |\mathcal{I}| + |\mathcal{J}|$ increments. Let $X_0 = 0$, we treat the main sensor as the first increment such that

$$X_1 = w_*(2s_* - 1) + b_*.$$

For $t = 1, ..., |\mathcal{I}|$ we let

$$X_{t+1} - X_t = w_i s_i + b_i \ \text{ s.t. } i = t + 1.$$

Finally, for $t = |\mathcal{I}| + 1, ..., |\mathcal{I}| + |\mathcal{J}|$ we let

$$X_{t+1} - X_t = -(w_j(1 - s_j) + b_j) \ \text{ s.t. } j = t + 1.$$

and the similar analysis can be performed for $y = 0$.

Above decomposition shows that $X_{1+|\mathcal{I}|+|\mathcal{J}|}$ is $\mathcal{F}_n$ measurable. Specifically, $X_{t+1} - X_t$ is $\mathcal{F}_t$ measurable for all $t = 1, ..., 1 + |\mathcal{I}| + |\mathcal{J}|$. Moreover, $X_{t+1} - X_t$ and $X_{t'+1} - X_{t'}$ are independent for $t \neq t'$.

Using the increments introduced above, one can further show that the maximum increments c_t for $t = 1, ..., 1 + |\mathcal{I}| + |\mathcal{J}|$ are given by

$$|w_* + b_* - (-w_* + b_*)| = 2w_* \leq 2_\vee w_* := c_1.$$

For $t = 1, ..., |\mathcal{I}|$ we let

$$|X_{t+1} - X_t| = |(w_i + b_i) - b_i| \leq {}_\vee w_i := c_{t+1} \ \text{ s.t. } i = t + 1.$$

Finally, for $t = |\mathcal{I}| + 1, ..., |\mathcal{I}| + |\mathcal{J}|$ we let

$$|X_{t+1} - X_t| = |-(w_j + b_j) - (-b_j)| \leq {}_\vee w_j := c_{t+1} \ \text{ s.t. } i = t + 1.$$

Recalling the bounds in Equation 4.16, Equation 4.18, Equation 4.19, Equation 4.20, Equation 4.21, we have

$$c_1 = 2\log\frac{{}_\vee\alpha_*}{1-{}_\wedge\alpha_*} \quad \mathit{for} t = 1 \mathit{and}\ c_t = \log\frac{{}_\vee\alpha_t(1-{}_\wedge\varepsilon_t)}{{}_\wedge\varepsilon_t(1-{}_\vee\alpha_t)} \quad \mathit{for}\ t \in \mathcal{I}\cup\mathcal{J}. \tag{4.34}$$

Next, for any $y \in \mathcal{Y}$, we derive the following

$$\begin{aligned}
&\mathbb{P}_{\mathcal{D}}[\Delta_w(y,s_*,s_{\mathcal{I}},s_{\mathcal{J}}) < 0|y] \\
&= \mathbb{P}_{\mathcal{D}}\big[\Delta_w(y,s_*,s_{\mathcal{I}},s_{\mathcal{J}}) - \mathbb{E}_{s_*,s_{\mathcal{I}},s_{\mathcal{J}}}[\Delta_w(y,s_*,s_{\mathcal{I}},s_{\mathcal{J}})] \\
&< -\mathbb{E}_{s_*,s_{\mathcal{I}},s_{\mathcal{J}}}[\Delta_w(y,s_*,s_{\mathcal{I}},s_{\mathcal{J}})]\big|y\big] \\
&\overset{(*)}{\leq} \mathbb{P}_{\mathcal{D}}\big[\Delta_w(y,s_*,s_{\mathcal{I}},s_{\mathcal{J}}) - \mathbb{E}_{s_*,s_{\mathcal{I}},s_{\mathcal{J}}}[\Delta_w(y,s_*,s_{\mathcal{I}},s_{\mathcal{J}})|y] < -\mu_{y,\mathcal{D}}\big|y\big]
\end{aligned}$$

where (*) stems from that $\mu_{y,\mathcal{D}}$ is a lower bound to $\mathbb{E}_{s_*,s_{\mathcal{I}},s_{\mathcal{J}}}[\Delta_w(y,s_*,s_{\mathcal{I}},s_{\mathcal{J}})|y]$ as shown in Lemma 8.

Let $\varepsilon = \mu_{y,\mathcal{D}}$. If $\mu_{y,\mathcal{D}} > 0$, using Theorem 11 for $\Psi_2 = \frac{\sum_{t\in\{1\}\cup\mathcal{I}\cup\mathcal{J}} c_t^2}{\mu_{y,\mathcal{D}}^2}$ where c_t is as defined in Equation 4.34 results in

$$\begin{aligned}
&\mathbb{P}_{\mathcal{D}}[\Delta_w(y,s_*,s_{\mathcal{I}},s_{\mathcal{J}}) < 0|y] \\
&\leq \mathbb{P}_{\mathcal{D}}\big[\Delta_w(y,s_*,s_{\mathcal{I}},s_{\mathcal{J}}) - \mathbb{E}_{s_*,s_{\mathcal{I}},s_{\mathcal{J}}}[\Delta_w(y,s_*,s_{\mathcal{I}},s_{\mathcal{J}})|y] < -\mu_{y,\mathcal{D}}\big|y\big] \\
&\leq \exp(-2/\Psi_2).
\end{aligned}$$

By further taking the expectation of $\mathbb{P}_{\mathcal{D}}[\Delta_w(y,s_*,s_{\mathcal{I}},s_{\mathcal{J}}) < 0|y]$ over $\mathcal{D} \in \{\mathcal{D}_b,\mathcal{D}_a\}$ and $y \in \mathcal{Y}$ such that

$$\begin{aligned}
&\mathcal{A}^{\text{KEMLP}} = \mathbb{E}_{\mathcal{D}\sim\{\mathcal{D}_a,\mathcal{D}_b\}}\mathbb{E}_{y\sim\mathcal{Y}}\big[\mathbb{P}_{\mathcal{D}}[o = y|y]\big] \\
&= \mathbb{E}_{\mathcal{D}\sim\{\mathcal{D}_a,\mathcal{D}_b\}}\mathbb{E}_{y\sim\mathcal{Y}}\big[\mathbb{P}_{\mathcal{D}}[\Delta_w(y,s_*,s_{\mathcal{I}},s_{\mathcal{J}}) > 0|y]\big] \\
&= 1 - \mathbb{E}_{\mathcal{D}\sim\{\mathcal{D}_a,\mathcal{D}_b\}}\mathbb{E}_{y\sim\mathcal{Y}}\big[\mathbb{P}_{\mathcal{D}}[\Delta_w(y,s_*,s_{\mathcal{I}},s_{\mathcal{J}}) < 0|y]\big] \\
&\geq 1 - \mathbb{E}_{\mu_{y,\mathcal{D}}}\big[\exp(-2\mu_{y,\mathcal{D}}^2/v^2)\big]
\end{aligned}$$

concludes the proof. □

4.5.5 *Proof of Proposition 10*

We begin with recalling Proposition 10.

Theorem (Recall)**.** *Let the number of permissive and preventive models be the same and denoted by n such that $n := |\mathcal{I}| = |\mathcal{J}|$. Note that the weighted accuracy of the main model in terms of its truth rate is simply $\alpha_* := \sum_{\mathcal{D}\in\{\mathcal{D}_b,\mathcal{D}_a\}} \pi_{\mathcal{D}}\alpha_{*,\mathcal{D}}$. Moreover, let $\mathcal{K}, \mathcal{K}' \in \{\mathcal{I}, \mathcal{J}\}$ with $\mathcal{K} \neq \mathcal{K}'$ and for any $\mathcal{D} \in \{\mathcal{D}_b, \mathcal{D}_a\}$, let*

$$\gamma_{\mathcal{D}} := \frac{1}{n+1} \min_{\mathcal{K}} \left\{ \alpha_{*,\mathcal{D}} - 1/2 + \sum_{k\in\mathcal{K}} \alpha_{k,\mathcal{D}} - \sum_{k'\in\mathcal{K}'} \varepsilon_{k',\mathcal{D}} \right\}.$$

If $\gamma_{\mathcal{D}} > \sqrt{\frac{4}{n+1}\log\frac{1}{1-\alpha_*}}$ for all $\mathcal{D} \in \{\mathcal{D}_b, \mathcal{D}_a\}$, then $\mathcal{A}^{KEMLP} > \mathcal{A}^{main}$.

Proof of Proposition 10. We start by recalling the widely known Chernoff bound for the sum of independent and non-identical random variables.

Lemma 9 (Chernoff Bound for Poisson Binomial Distributions)**.** *Let X be a random variable with Poisson Binomial distribution. For $\delta \in [0, 1]$,*

$$\mathbb{P}[X < (1-\delta)\mu_X] \leq \exp(-\delta^2\mu_X/2).$$

Recall that KEMLP predicts y to be $\hat{o}$ where

$$\hat{o} = \arg\max_{\tilde{y}\in\mathcal{Y}} \mathbb{P}[o = \tilde{y}|\tilde{s}_*, \tilde{s}_{\mathcal{I}}, \tilde{s}_{\mathcal{J}}, w] = \arg\max_{\tilde{y}\in\mathcal{Y}} \sigma(\Delta_w(\tilde{y}, s_*, s_{\mathcal{I}}, s_{\mathcal{J}}))$$

where

$$\Delta_w(\tilde{y}, s_*, s_{\mathcal{I}}, s_{\mathcal{J}}) = (2\tilde{y}-1)\Big(b + w_*(2s_* - 1) + \sum_{i\in\mathcal{I}} w_i s_i - \sum_{j\in\mathcal{J}} w_j(1-s_j)\Big).$$

We showed earlier that there exist a set of parameters w, and call it optimal parameters w^*, where

$$\mathbb{P}[o = \tilde{y}|\tilde{s}_*, \tilde{s}_{\mathcal{I}}, \tilde{s}_{\mathcal{J}}, w^*] = \mathbb{P}[y = \tilde{y}|\tilde{s}_*, \tilde{s}_{\mathcal{I}}, \tilde{s}_{\mathcal{J}}]$$

for all $\tilde{y} \in \mathcal{Y}$.

Note that, due to above equation, $\mathbb{P}[o = \tilde{y}|\tilde{s}_*, \tilde{s}_{\mathcal{I}}, \tilde{s}_{\mathcal{J}}, w^*]$ is Bayes classifier where the error of classifier is minimized over w. Hence,

$$\mathbb{P}[\hat{o} \neq y|w^*] \leq \mathbb{P}[\hat{o} \neq y|w]$$

and

$$\mathbb{P}[\hat{o} = y|w^*] \geq \mathbb{P}[\hat{o} = y|w]$$

for any $w \in \mathbb{R}^{|\mathcal{I}|+|\mathcal{J}|+2}$.

Leveraging above fact, we will bound $\mathbb{P}[\hat{o} = y|w]$ from below where we will use some parameters w that are not optimal. That is, from now on, we will focus on $\mathbb{P}[\hat{o} = y|w]$ where w is not optimal but leads to a close resemblance of $\mathbb{P}[\hat{o} = y|w^*]$. In other words, we will perform a worst-case analysis where $\hat{o}$ will be a result of unweighted majority voting. Hence, we let w be given by $\mathbf{w} = [0; 1/2; (1)_{i\in\mathcal{I}}; (1)_{j\in\mathcal{J}}]$. For this case, $\Delta_w(\tilde{y}, s_*, s_{\mathcal{I}}, s_{\mathcal{J}})$ becomes a random variable with Poisson Binomial distribution and with some bias. That is,

$$\Delta_w(\tilde{y}, s_*, s_{\mathcal{I}}, s_{\mathcal{J}}) = (2\tilde{y} - 1)\Big((s_* - 1/2) + \sum_{i\in\mathcal{I}} s_i - \sum_{j\in\mathcal{J}}(1 - s_j)\Big)$$

where s_*, $s_{i\in\mathcal{I}}$ and $s_{j\in\mathcal{J}}$ are random variables in $\mathcal{Y}$.

Using the weight introduced above, we can now re-write the weighted robust accuracy of KEMLP as

$$\begin{aligned}
&\mathcal{A}^{\text{KEMLP}} = \mathbb{P}[\hat{o} = y|w^*] \geq \mathbb{P}[\hat{o} = y|w] = \pi_{\mathcal{D}_a}\mathbb{P}_{\mathcal{D}_a}[\hat{o} = y|w] + \pi_{\mathcal{D}_b}\mathbb{P}_{\mathcal{D}_b}[\hat{o} = y|w] \\
&= \pi_{\mathcal{D}_a}\big(\mathbb{P}_{\mathcal{D}_a}[\hat{o} = y|w, y = 1]\mathbb{P}_{\mathcal{D}_a}[y = 1] + \mathbb{P}_{\mathcal{D}_a}[\hat{o} = y|w, y = 0]\mathbb{P}_{\mathcal{D}_a}[y = 0]\big) \\
&+ \pi_{\mathcal{D}_b}\big(\mathbb{P}_{\mathcal{D}_b}[\hat{o} = y|w, y = 1]\mathbb{P}_{\mathcal{D}_b}[y = 1] + \mathbb{P}_{\mathcal{D}_b}[\hat{o} = y|w, y = 0]\mathbb{P}_{\mathcal{D}_b}[y = 0]\big).
\end{aligned} \tag{4.35}$$

Next, we will derive a lower bound for $\mathbb{P}_{\mathcal{D}}[\hat{o} = y|y = \tilde{y}, \mathbf{w}]$ for $\mathcal{D} \in \{\mathcal{D}_b, \mathcal{D}_a\}$ and for all $\tilde{y} \in \{0, 1\}$.

FOR $y = 1$: We have

$$\mathbb{P}_{\mathcal{D}}[\hat{o} = y|w, y = 1] = \mathbb{P}_{\mathcal{D}}[s_* + \sum_{i\in\mathcal{I}} s_i + \sum_{j\in\mathcal{J}} s_j - (|\mathcal{J}| + 1/2) \geq 0|y = 1]$$

$$1 - \mathbb{P}_{\mathcal{D}}[s_* + \sum_{i\in\mathcal{I}} s_i + \sum_{j\in\mathcal{J}} s_j - (|\mathcal{J}| + 1/2) < 0|y = 1]$$

$$= 1 - \mathbb{P}_{\mathcal{D}}[s_* + \sum_{i\in\mathcal{I}} s_i + \sum_{j\in\mathcal{J}} s_j < |\mathcal{J}| + 1/2|y = 1]$$

where $\mathbb{P}_{\mathcal{D}}[s_* = 1|y = 1] = \alpha_{*,\mathcal{D}}$ (resp. $\mathbb{P}_{\mathcal{D}}[s_i = 1|y = 1] = \alpha_{i,\mathcal{D}}$ and $\mathbb{P}_{\mathcal{D}}[s_j = 1|y = 1] = 1 - \varepsilon_{j,\mathcal{D}}$).

We let

$$\Psi_{\mathcal{D},y=1} := s_* + \sum_{i\in\mathcal{I}} s_i + \sum_{j\in\mathcal{J}} s_j - (|\mathcal{J}| + 1/2)$$

and

$$\hat{\Psi}_{\mathcal{D},y=1} := s_* + \sum_{i\in\mathcal{I}} s_i + \sum_{j\in\mathcal{J}} s_j = \Psi_{\mathcal{D},y=1} + |\mathcal{J}| + 1/2.$$

Similarly, the expected values of $\Psi_{\mathcal{D},y=1}$ and $\hat{\Psi}_{\mathcal{D},y=1}$ over s_*, s_i and s_j are given by $\mu_{\Psi_{\mathcal{D},y=1}}$ and $\mu_{\hat{\Psi}_{\mathcal{D},y=1}}$, respectively. Precisely,

$$\mu_{\Psi_{\mathcal{D},y=1}} = \alpha_{*,\mathcal{D}} - 1/2 + \sum_{i\in\mathcal{I}} \alpha_{i,\mathcal{D}} - \sum_{j\in\mathcal{J}} \varepsilon_{j,\mathcal{D}}$$

and

$$\mu_{\hat{\Psi}_{\mathcal{D},y=1}} = \alpha_{*,\mathcal{D}} + \sum_{i\in\mathcal{I}} \alpha_{i,\mathcal{D}} + \sum_{j\in\mathcal{J}} (1 - \varepsilon_{j,\mathcal{D}}) = \mu_{\Psi_{\mathcal{D},y=1}} + |\mathcal{J}| + 1/2$$

We then write $\mathbb{P}_{\mathcal{D}}[\hat{o} \neq y|w, y = 1]$ as

$$\mathbb{P}_{\mathcal{D}}[\hat{o} \neq y|w, y = 1] = \mathbb{P}[\Psi_{\mathcal{D},y=1} < 0] \leq \exp(-\delta^2_{\mathcal{D},y=1}\mu_{\hat{\Psi}_{\mathcal{D},y=1}}/2)$$

where

$$\delta_{\mathcal{D},y=1} = 1 - \frac{|\mathcal{J}| + 1/2}{\mu_{\hat{\Psi}_{\mathcal{D},y=1}}} = \frac{\mu_{\Psi_{\mathcal{D},y=1}}}{\mu_{\hat{\Psi}_{\mathcal{D},y=1}}}.$$

Let now $\gamma_{\mathcal{D},y=1}$ be the difference between true and false rates of sensors normalized over preventive models when $y = 1$ such that

$$\gamma_{\mathcal{D},y=1} := \frac{1}{|\mathcal{J}|+1}(\alpha_{*,\mathcal{D}} - 1/2 + \sum_{i\in\mathcal{I}} \alpha_{i,\mathcal{D}} - \sum_{j\in\mathcal{J}} \varepsilon_{j,\mathcal{D}}).$$

Noting that $\mu_{\Psi_{\mathcal{D},y=1}} = (|\mathcal{J}|+1)\gamma_{\mathcal{D},y=1}$, we have $\delta_{\mathcal{D},y=1} = \frac{(|\mathcal{J}|+1)\gamma_{\mathcal{D},y=1}}{(|\mathcal{J}|+1)\gamma_{\mathcal{D}}+|\mathcal{J}|+1/2}$ and $\mu_{\hat{\Psi}_{y=1}} = (|\mathcal{J}|+1)\gamma_{\mathcal{D},y=1} + |\mathcal{J}| + 1/2$. Using Lemma 9 for a Poisson random variable $\hat{\Psi}_{y=1}$, we bound $\mathbb{P}_{\mathcal{D}}[\hat{o} \neq y|w, y=1]$ as

$$\begin{aligned}
&\mathbb{P}_{\mathcal{D}}[\hat{o} \neq y|w, Y=1] = \mathbb{P}[\Psi_{\mathcal{D},y=1} < 0] = \mathbb{P}[\hat{\Psi}_{\mathcal{D},y=1} < |\mathcal{J}| + 1/2] \\
&\leq \exp(-\delta^2_{\mathcal{D},y=1}\mu_{\hat{\Psi}_{\mathcal{D},y=1}}/2) \\
&= \exp\left(-\frac{(|\mathcal{J}|+1)^2\gamma^2_{\mathcal{D},y=1}}{2\Big((|\mathcal{J}|+1)\gamma_{\mathcal{D},y=1} + |\mathcal{J}| + 1/2\Big)}\right) \\
&\leq \exp\left(-\frac{(|\mathcal{J}|+1)^2\gamma^2_{\mathcal{D},y=1}}{2\Big((|\mathcal{J}|+1)\gamma_{\mathcal{D},y=1} + |\mathcal{J}| + 1\Big)}\right) \\
&= \exp\left(-(|\mathcal{J}|+1)\frac{\gamma^2_{\mathcal{D},y=1}}{2(\gamma_{\mathcal{D},y=1}+1)}\right)
\end{aligned} \tag{4.36}$$

FOR $y = 0$: We have

$$\begin{aligned}
&\mathbb{P}_{\mathcal{D}}[\hat{o} = y|w, y=0] = \mathbb{P}_{\mathcal{D}}[s_* - 1/2 + \sum_{i\in\mathcal{I}} s_i - \sum_{j\in\mathcal{J}} 1 - s_j \leq 0|y=0] \\
&1 - \mathbb{P}_{\mathcal{D}}[s_* - 1/2 + \sum_{i\in\mathcal{I}} s_i - \sum_{j\in\mathcal{J}} 1 - s_j > 0|y=0] \\
&= 1 - \mathbb{P}_{\mathcal{D}}[-s_* + 1/2 - \sum_{i\in\mathcal{I}} s_i + \sum_{j\in\mathcal{J}} 1 - s_j < 0|y=0] \\
&= 1 - \mathbb{P}_{\mathcal{D}}[-s_* + 1 - 1/2 + \sum_{i\in\mathcal{I}} 1 - s_i - |\mathcal{I}| + \sum_{j\in\mathcal{J}} 1 - s_j < 0|y=0] \\
&= 1 - \mathbb{P}_{\mathcal{D}}[-s_* + 1 + \sum_{i\in\mathcal{I}} 1 - s_i + \sum_{j\in\mathcal{J}} 1 - s_j < |\mathcal{I}| + 1/2|y=0]
\end{aligned}$$

where $\mathbb{P}_{\mathcal{D}}[s_* = 1|y=0] = 1 - \alpha_{*,\mathcal{D}}$ (resp. $\mathbb{P}_{\mathcal{D}}[s_i = 1|y=0] = \varepsilon_{i,\mathcal{D}}$ and $\mathbb{P}_{\mathcal{D}}[s_j = 1|y=0] = 1 - \alpha_{j,\mathcal{D}}$).

We let

$$\Psi_{\mathcal{D},y=0} := 1 - s_* + \sum_{i \in \mathcal{I}} 1 - s_i + \sum_{j \in \mathcal{J}} 1 - s_j - (|\mathcal{I}| + 1/2)$$

and

$$\hat{\Psi}_{\mathcal{D},y=0} := 1 - s_* + \sum_{i \in \mathcal{I}} 1 - s_i + \sum_{j \in \mathcal{J}} 1 - s_j = \Psi_{\mathcal{D},y=0} + |\mathcal{I}| + 1/2.$$

Similarly, the expected values of $\Psi_{\mathcal{D},y=0}$ and $\hat{\Psi}_{\mathcal{D},y=0}$ over s_*, s_i and s_j are given by $\mu_{\Psi_{\mathcal{D},y=0}}$ and $\mu_{\hat{\Psi}_{\mathcal{D},y=0}}$, respectively. Precisely,

$$\mu_{\Psi_{\mathcal{D},y=0}} = \alpha_{*,\mathcal{D}} - 1/2 - \sum_{i \in \mathcal{I}} \varepsilon_{i,\mathcal{D}} + \sum_{j \in \mathcal{J}} \alpha_{j,\mathcal{D}}$$

and

$$\mu_{\hat{\Psi}_{\mathcal{D},y=0}} = \alpha_{*,\mathcal{D}} + \sum_{i \in \mathcal{I}} 1 - \varepsilon_{i,\mathcal{D}} + \sum_{j \in \mathcal{J}} \alpha_{j,\mathcal{D}} = \mu_{\Psi_{\mathcal{D},y=0}} + |\mathcal{I}| + 1/2$$

We then write $\mathbb{P}_{\mathcal{D}}[\hat{o} \neq y | w, y = 0]$ as

$$\mathbb{P}_{\mathcal{D}}[\hat{o} \neq y | w, y = 0] = \mathbb{P}[\Psi_{\mathcal{D},y=0} < 0] \leq \exp(-\delta^2_{\mathcal{D},y=0} \mu_{\hat{\Psi}_{\mathcal{D},y=0}}/2)$$

where

$$\delta_{\mathcal{D},y=0} = 1 - \frac{|\mathcal{I}| + 1/2}{\mu_{\hat{\Psi}_{\mathcal{D},y=0}}} = \frac{\mu_{\Psi_{\mathcal{D},y=0}}}{\mu_{\hat{\Psi}_{\mathcal{D},y=0}}}.$$

Let now $\gamma_{\mathcal{D},y=0}$ be the difference between true and false rates of sensors normalized over permissive models when $y = 0$ such that

$$\gamma_{\mathcal{D},y=0} := \frac{1}{|\mathcal{I}| + 1} (\alpha_{*,\mathcal{D}} - 1/2 + \sum_{j \in \mathcal{J}} \alpha_{j,\mathcal{D}} - \sum_{i \in \mathcal{I}} \varepsilon_{i,\mathcal{D}}).$$

Noting that $\mu_{\Psi_{\mathcal{D},y=0}} = (|\mathcal{I}|+1)\gamma_{\mathcal{D},y=0}$, we have $\delta_{\mathcal{D},y=0} = \frac{(|\mathcal{I}|+1)\gamma_{\mathcal{D}}}{(|\mathcal{I}|+1)\gamma_{\mathcal{D}}+|\mathcal{I}|+1/2}$ and $\mu_{\hat{\Psi}_{y=0}} = (|\mathcal{I}|+1)\gamma_{\mathcal{D},y=0} + |\mathcal{I}| + 1/2$. Using Lemma 9 for a Poisson random variable $\hat{\Psi}_{y=0}$, we bound $\mathbb{P}_{\mathcal{D}}[\hat{o} \neq y | w, y=0]$ as

$$\begin{aligned}
&\mathbb{P}_{\mathcal{D}}[\hat{o} \neq y|w, y=0] = \mathbb{P}[\Psi_{\mathcal{D},y=0} < 0] \\
&= \mathbb{P}[\hat{\Psi}_{\mathcal{D},y=0} < |\mathcal{I}| + 1/2] \leq \exp(-\delta^2_{\mathcal{D},y=0}\mu_{\hat{\Psi}_{\mathcal{D},y=0}}/2) \\
&= \exp\left(-\frac{(|\mathcal{I}|+1)^2\gamma^2_{\mathcal{D},y=0}}{2\Big((|\mathcal{I}|+1)\gamma_{\mathcal{D},y=0} + |\mathcal{I}| + 1/2\Big)}\right) \\
&\leq \exp\left(-\frac{(|\mathcal{I}|+1)^2\gamma^2_{\mathcal{D},y=0}}{2\Big((|\mathcal{I}|+1)\gamma_{\mathcal{D},y=0} + |\mathcal{I}| + 1\Big)}\right) \\
&= \exp\left(-(|\mathcal{I}|+1)\frac{\gamma^2_{\mathcal{D},y=0}}{2(\gamma_{\mathcal{D},y=0}+1)}\right)
\end{aligned} \tag{4.37}$$

LAST STEP: For convenience, let $n := |\mathcal{I}| = |\mathcal{J}|$ and

$$\gamma_{\mathcal{D}} := \min(\gamma_{\mathcal{D},y=1}, \gamma_{\mathcal{D},y=0}).$$

Using Equation 4.36 and Equation 4.37, we bound the pipeline accuracy in Equation 4.35 such that

$$\begin{aligned}
\mathcal{A}^{\text{KEMLP}} &\geq 1 - \sum_{\mathcal{D}\in\{\mathcal{D}_b,\mathcal{D}_a\}} \pi_{\mathcal{D}} \exp\left(-(n+1)\frac{\gamma^2_{\mathcal{D}}}{2(\gamma_{\mathcal{D}}+1)}\right) \\
&\geq 1 - \sum_{\mathcal{D}\in\{\mathcal{D}_b,\mathcal{D}_a\}} \pi_{\mathcal{D}} \exp\left(-(n+1)\frac{\gamma^2_{\mathcal{D}}}{4}\right).
\end{aligned} \tag{4.38}$$

Hence, if $1 - \exp\left(-(n+1)\frac{\gamma^2_{\mathcal{D}}}{4}\right) > \mathcal{A}^{\text{main}}$ for all $\mathcal{D} \in \{\mathcal{D}_b, \mathcal{D}_a\}$, then we have $\mathcal{A}^{\text{KEMLP}} > \mathcal{A}^{\text{main}}$. Manipulating it further for all $\mathcal{D} \in \{\mathcal{D}_b, \mathcal{D}_a\}$ concludes the proof. □

4.5.6 *Proof of Corollary 6*

We recall the respective setting as follows. We assume that the auxiliary models are *homogeneous* for each type: permissive or preventive. For example, α_k is fixed with respect to $k \in \mathcal{I} \cup \mathcal{J}$, hence we drop the subscripts, i.e., $\alpha_{k,\mathcal{D}} = \alpha$ and $\varepsilon_{k,\mathcal{D}} = \varepsilon$. We assume that the same number of auxiliary models are used, namely $|\mathcal{I}| = |\mathcal{J}| = n$, and that the classes are balanced with $\mathbb{P}_{\mathcal{D}}(y = 1) = \mathbb{P}_{\mathcal{D}}(y = 0)$, for all $\mathcal{D} \in \{\mathcal{D}_b, \mathcal{D}_a\}$. Finally, we let $\alpha_{*,\mathcal{D}_b} = 1$ and $\alpha_{*,\mathcal{D}_a} = 0$, and $\alpha - \varepsilon > 0$. Then, the following holds.

Corollary (Recall). *The weighted robust accuracy of KEMLP in the homogeneous setting satisfies*

$$\mathcal{A}^{KEMLP} \geq 1 - \exp\left(-2n(\alpha - \varepsilon)^2\right).$$

In particular, one has $\lim_{n\to\infty} \mathcal{A}^{KEMLP} = 1$.

Proof of Corollary 6. First, for $\alpha_{*,\mathcal{D}_b} = 1$ and $\alpha_{*,\mathcal{D}_a} = 0$, using Equation 4.12 and Equation 4.13, we note that

$$w_* = b_* = 0.$$

Secondly, in the homogeneous case, the conditional independence reflects to the mixture model and models become conditionally independent in the mixture model as well. That is, the condition on the other models in Equation 4.18, Equation 4.19, Equation 4.20, Equation 4.21 drops and we have closed form expression for all optimal parameters. Namely, for $\alpha_{i,\mathcal{D}} = \alpha_{j,\mathcal{D}} = \alpha$ and $\varepsilon_{i,\mathcal{D}} = \varepsilon_{j,\mathcal{D}} = \varepsilon$ with $\alpha > \varepsilon$, once can deduce from Equation 4.18, Equation 4.19, Equation 4.20, Equation 4.21 that the optimal weight of auxiliary sensors are given by $w_i = w_j = \log \frac{\alpha}{\varepsilon}$ and $b = \sum_{i\in\mathcal{I}} b_i - \sum_{j\in\mathcal{J}} b_j = \sum_{i\in\mathcal{I}} \log \frac{1-\alpha}{1-\varepsilon} - \sum_{j\in\mathcal{J}} \log \frac{1-\alpha}{1-\varepsilon} = 0$. Also, $w_i = w_j > 0$ for $\alpha > \varepsilon$. For this setting, we can write out $\mathcal{A}^{\text{KEMLP}}$ as follows.

$$\begin{aligned}\mathcal{A}^{\text{KEMLP}} &= \mathbb{E}_{\mathcal{D}\in\{\mathcal{D}_b,\mathcal{D}_a\}}\mathbb{E}_{y\sim\mathcal{Y}}\left[\mathbb{P}[d_{*,\mathcal{D}} + yd_{\mathcal{I},\mathcal{D}} + (1-y)d_{\mathcal{I},\mathcal{D}} > 0|y]\right] \\ &\overset{(*)}{=} \mathbb{E}_{y\sim\mathcal{Y}}\left[\mathbb{P}[yd_{\mathcal{I},\mathcal{D}} + (1-y)d_{\mathcal{I},\mathcal{D}} > 0|y]\right] \\ &\overset{(**)}{=} \frac{1}{2}\left(\mathbb{P}[d_{\mathcal{I},\mathcal{D}} > 0|y=1] + \mathbb{P}[d_{\mathcal{J},\mathcal{D}} > 0|y=0]\right) \overset{(***)}{=} \mathbb{P}[d_{\mathcal{I},\mathcal{D}} > 0|y=1]\end{aligned}$$

where (*) follows from the homogeneity of models over both benign and adversarial distributions as well as that $d_{*,\mathcal{D}} = w_*(2s_* - 1) = 0$, (**) follows from the class balance, and finally (***) stems from the symmetry.

Let $B(n,p)$ denote the Binomial distribution with count parameter n and success probability p. Let also that d_α and d_ε be random variables with Binomial distributions such that $d_\alpha \sim B(n,\alpha)$ and $d_\varepsilon \sim B(n,\varepsilon)$. We then rewrite the Weighted Robust Accuracy of KEMLP as follows.

$$\begin{aligned}\mathcal{A}^{\text{KEMLP}} &= \mathbb{P}[d_{\mathcal{I},\mathcal{D}} > 0|y=1] = 1 - \mathbb{P}[d_{\mathcal{I},\mathcal{D}} < 0|y=1] \\ &= 1 - \mathbb{P}[w(d_\alpha - d_\varepsilon) < 0|y=1] = 1 - \mathbb{P}[d_\alpha - d_\varepsilon < 0]\end{aligned}$$

where the last equality follows from that $w = \log\frac{\alpha}{\varepsilon} > 0$.

We then review the Bounded Differences Inequality which will enable us to bound the tail probability $\mathbb{P}[d_\alpha - d_\varepsilon < 0|y=1]$.

Theorem 12 (Bounded Differences Inequality (Boucheron, Lugosi, and Massart, 2013)). *Assume that a function $\phi : \mathcal{X}^n \to \mathbb{R}$ of independent random variables $X_1, ..., X_n \in \mathcal{X}$ satisfies the bounded differences property with constants $c_1, ..., c_n$. Denote $v^2 = \sum_{i=[n]} c_i^2$ and $Z = \phi(X_1, ..., X_n)$. Z satisfies:*

$$\mathbb{P}(Z - \mathbb{E}(Z) > t) \le \exp\left(-\frac{2t^2}{v^2}\right) \quad \textit{and} \quad \mathbb{P}(Z - \mathbb{E}(Z) < -t) \le \exp\left(-\frac{2t^2}{v^2}\right).$$

We refer to, for example, (Boucheron, Lugosi, and Massart, 2013) for a proof of Theorem 12.

Using Theorem 12 for $Z = d_\alpha - d_\varepsilon$, $\mathcal{A}^{\text{KEMLP}}$ can be bounded as:

$$\begin{aligned}\mathcal{A}^{\text{KEMLP}} &= 1 - \mathbb{P}[d_\alpha - d_\varepsilon < 0] = 1 - \mathbb{P}[d_\alpha - d_\varepsilon - \mathbb{E}[d_\alpha - d_\varepsilon] < -\mathbb{E}[d_\alpha - d_\varepsilon]] \\ &= 1 - \mathbb{P}[d_\alpha - d_\varepsilon - n(\alpha - \varepsilon) < -n(\alpha - \varepsilon)].\end{aligned}$$

Moreover, for $t = n(\alpha - \varepsilon)$ and $v^2 = n$ we finally have

$$\begin{aligned}\mathcal{A}^{\text{KEMLP}} &= 1 - \mathbb{P}[d_\alpha - d_\varepsilon - n(\alpha - \varepsilon) < -n(\alpha - \varepsilon)] \\ &\geq 1 - \exp\big(- 2(n^2(\alpha - \varepsilon)^2)/n\big) \\ &= 1 - \exp\big(- 2n(\alpha - \varepsilon)^2\big)\end{aligned}$$

concludes the proof for the lower bound.

As the final step, we will prove that $\mathcal{A}^{\text{KEMLP}} > \mathcal{A}^{\text{main}}$. Note that $\mathcal{A}^{\text{main}} = \mathbb{E}_{\mathcal{D}\in\{\mathcal{D}_b,\mathcal{D}_a\}}\mathbb{E}_{y\sim\mathcal{Y}}\big[\mathbb{P}[d_{*,\mathcal{D}} > 0|y]\big] = \pi_{\mathcal{D}_b}\alpha_{*,\mathcal{D}_b} + \pi_{\mathcal{D}_a}\alpha_{*,\mathcal{D}_a} = 1/2 \cdot 1 + 1/2 \cdot 0 = 1/2$. Therefore, it only remains to analyze whether $\mathcal{A}^{\text{KEMLP}} > 1/2$ or not. Towards that, we state the following result.

Lemma 10 (On the comparison of two binomial random variables)**.** *Let $p, q \in [0, 1]$ denote the success probabilities for two Binomial random variables. If $p > q$, then $\mathbb{P}[X > Y] > \frac{1}{2}$.*

Proof. Let X and Y be random variables such that $X \sim B(n, p)$ and $Y \sim B(n, q)$. $Z := X - Y$ can be shown to have the following probability mass function

$$\mathbb{P}(Z = z) = \begin{cases} \sum_{k\in\{0\}\cup[n]} f(k+z, n, p) f(k, n, q) & \text{if } x \geq 0 \\ \sum_{k\in\{0\}\cup[n]} f(k, n, p) f(k+z, n, q) & \text{elsewhere}\end{cases}$$

where $f(k, n, p) = \binom{n}{k} p^k (1-p)^{n-k}$ for $k \leq n$. Moreover, we have

$$\mathbb{P}(Z > 0) = \mathbb{P}(X - Y > 0) = \sum_{\substack{z\in[n] \\ k\in\{0\}\cup[n]}} f(k+z, n, p) f(k, n, q),$$

$$\mathbb{P}(Z \leq 0) = \sum_{\substack{z\in[n] \\ k\in\{0\}\cup[n]}} f(k, n, p) f(k+z, n, q).$$

Note that if $p > q$, then $f(k+z, n, p) f(k, n, q) > f(k, n, p) f(k+z, n, q)$ for fixed $n, k \geq 0$. Hence, the summation over $z \in [n], k \in \{0\} \cup [n]$ leads to $\mathbb{P}(Z > 0) > \mathbb{P}(Z \leq 0)$. It is further implied by $\mathbb{P}(Z > 0) + \mathbb{P}(Z \leq 0) = 1$ that $\mathbb{P}(Z > 0) > \frac{1}{2}$. □

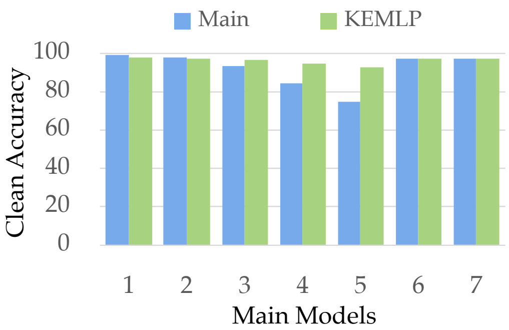


Figure 4.3: Comparison of clean accuracies: KEMLP vs. different main task models

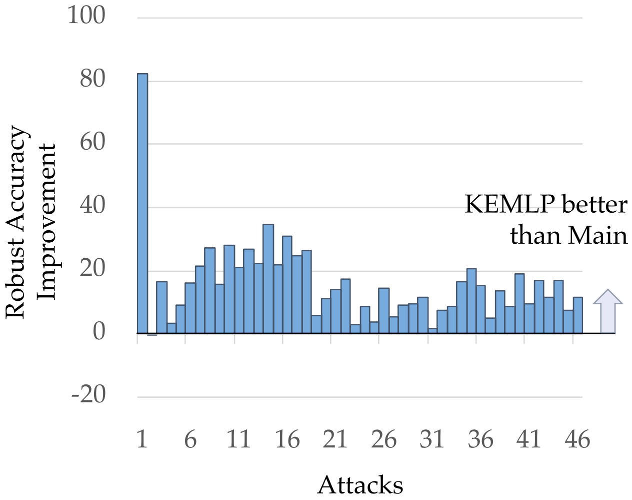


Figure 4.4: Robust accuracy improvement of KEMLP over GTSRB-CNN

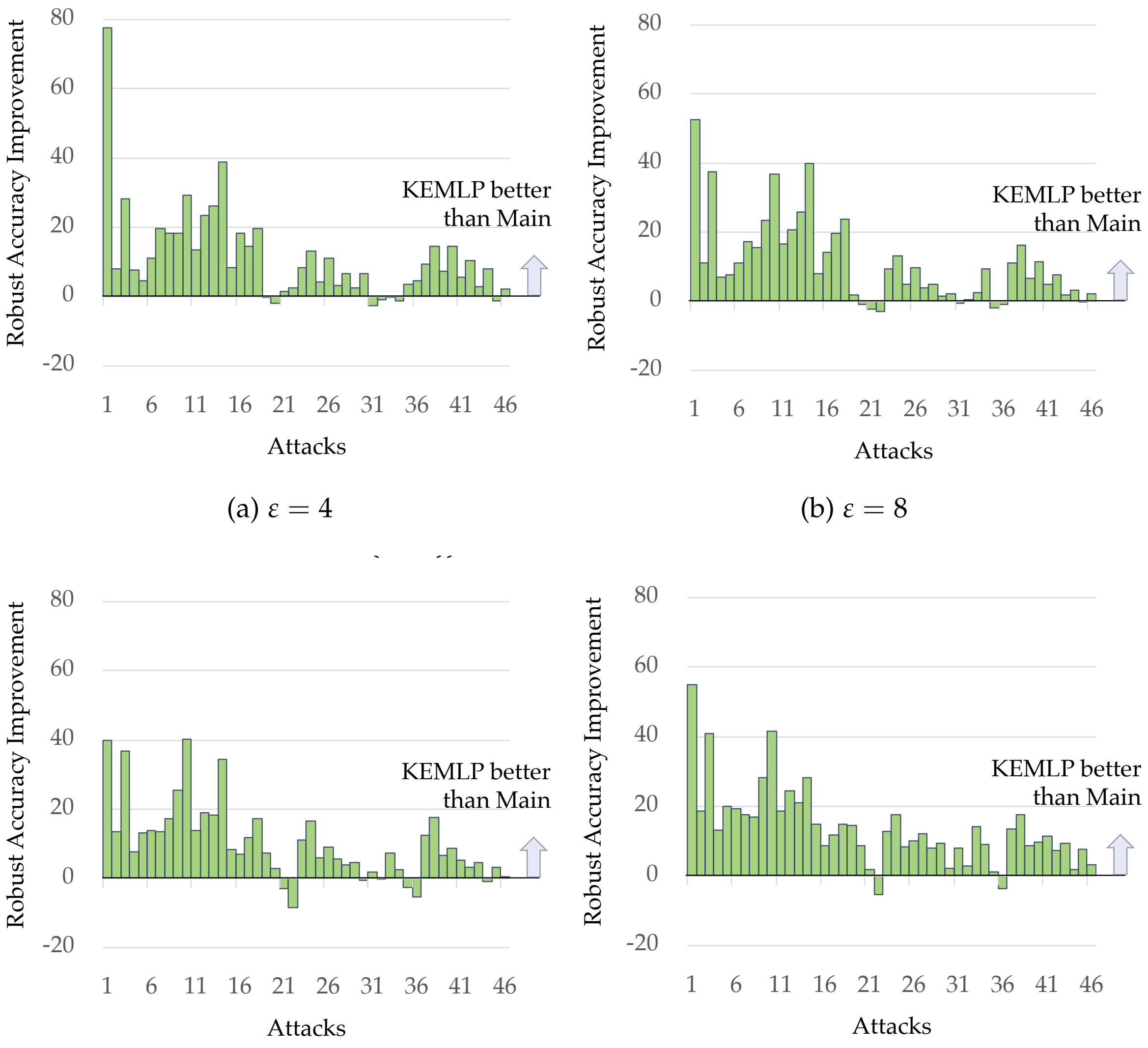
Using Lemma 10 for $X = d_\alpha$ and $Y = d_\varepsilon$ as well as that $\alpha > \varepsilon$, we have

$$\mathcal{A}^{\text{KEMLP}} = \mathbb{P}[d_\alpha - d_\varepsilon > 0] = \mathbb{P}[d_\alpha > d_\varepsilon] > 1/2 = \mathcal{A}^{\text{main}}.$$

Hence the proof results. □

4.6 EXPERIMENTAL EVALUATION

In this section, we evaluate KEMLP based on the traffic sign recognition task against different adversarial attacks and corruptions, including the physical attacks (Eykholt et al., 2018), $\mathcal{L}_\infty$ bounded attacks, unforeseen attacks (Kang



(a) $\varepsilon = 4$

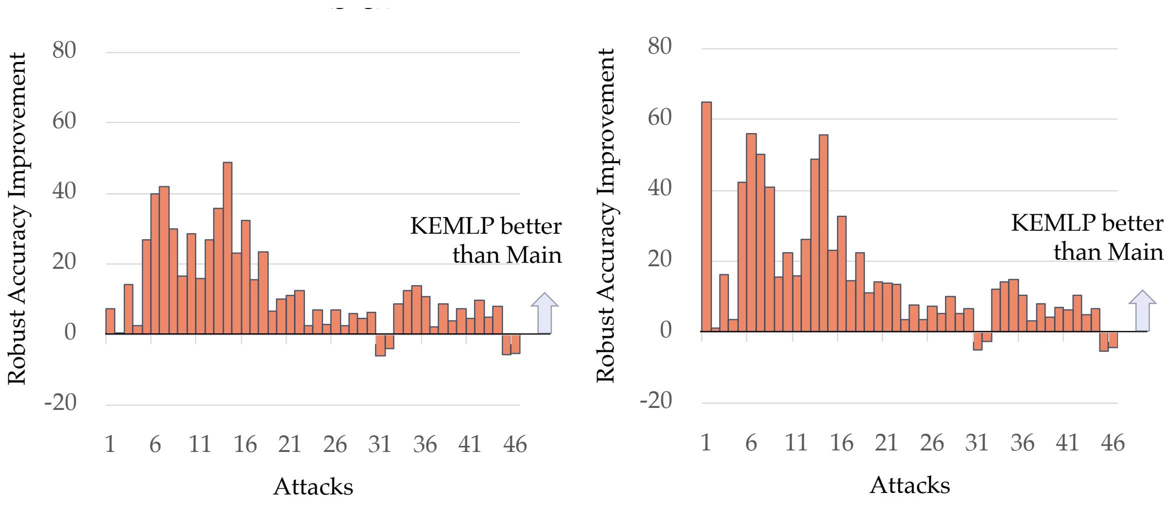
(b) $\varepsilon = 8$

(c) $\varepsilon = 16$

(d) $\varepsilon = 32$

Figure 4.5: Robust accuracy improvement of KEMLP over Adversarial Training with various ε

et al., 2019), and common corruptions (Hendrycks and Dietterich, 2019). We show that under both whitebox and blackbox settings against a *diverse* set of attacks, 1) KEMLP achieves significantly higher robustness than baselines, 2) KEMLP maintains similar clean accuracy with a strong main task model whose clean accuracy is originally high (e.g., vanilla CNN), 3) KEMLP even achieves higher clean accuracy than a relatively weak main task model whose clean accuracy is originally low as a tradeoff for its robustness (e.g., adversarially trained models).



(a) 5 × 5

(b) 7 × 7

Figure 4.6: Robust accuracy improvement of KEMLP over DOA

Table 4.1: Model performances (%) under physical attacks ($\beta = 0.4$). Performance gain and loss of KEMLP over baselines are put in the parentheses.

	Main			KEMLP		
	Clean Acc	Robust Acc	W-Robust Acc	Clean Acc	Robust Acc	W-Robust Acc
GTSRB-CNN	100	5	52.5	100(0)	87.5(+82.5)	93.75(+41.25)
AdvTrain ($\varepsilon = 4$)	100	12.5	56.25	100(0)	90(+77.5)	95(+38.75)
AdvTrain ($\varepsilon = 8$)	97.5	37.5	67.5	100(+2.5)	90(+52.5)	95(+27.5)
AdvTrain ($\varepsilon = 16$)	87.5	50	68.75	100(+12.5)	90(+40)	95(+26.25)
AdvTrain ($\varepsilon = 32$)	62.5	32.5	47.5	100(+37.5)	90(+57.5)	95(+47.5)
DOA (5x5)	95	90	92.5	100(+5)	100(+10)	100(+7.5)
DOA (7x7)	57.5	32.5	45	100(+42.5)	100(+67.5)	100(+55)

Besides the worst-case adversarial attacks, we also consider the robustness against common corruptions (Hendrycks and Dietterich, 2019).

4.6.1 *Experimental Setup*

DATASET Following existing work (Eykholt et al., 2018; Wu, Tong, and Vorobeychik, 2019) that evaluate ML robustness on traffic sign data, we adopt LISA (Mogelmose, Trivedi, and Moeslund, 2012) and GTSRB (Stallkamp et al., 2012) for training and evaluation. All data are processed by standard crop-and-resize to 32 × 32 as described in (Sermanet and LeCun, 2011). In

Table 4.2: Accuracy (%) under whitebox $\mathcal{L}_\infty$ attacks ($\beta = 0.8$)

Models		$\varepsilon = 0$	$\varepsilon = 4$	$\varepsilon = 8$	$\varepsilon = 16$	$\varepsilon = 32$
GTSRB-CNN	Main	99.38	67.31	43.13	13.50	3.63
	KEMLP	98.28(−1.10)	85.39(+18.08)	71.76(+28.63)	48.89(+35.39)	26.13(+22.50)
AdvTrain ($\varepsilon = 4$)	Main	97.94	87.94	68.85	38.66	8.77
	KEMLP	97.89(−0.05)	92.80(+4.86)	79.58(+10.73)	57.48(+18.82)	28.58(+19.81)
AdvTrain ($\varepsilon = 8$)	Main	93.72	84.21	71.76	43.16	13.01
	KEMLP	96.79(+3.07)	92.08(+7.87)	81.58(+9.82)	59.18(+16.02)	30.61(+17.60)
AdvTrain ($\varepsilon = 16$)	Main	84.54	78.58	71.89	55.99	19.55
	KEMLP	94.68(+10.14)	91.64(+13.06)	85.55(+13.66)	67.98(+11.99)	32.61(+13.06)
AdvTrain ($\varepsilon = 32$)	Main	74.74	70.24	65.61	56.22	29.04
	KEMLP	91.46(+16.72)	88.58(+18.34)	83.23(+17.62)	72.02(+15.80)	41.90(+12.86)
DOA (5x5)	Main	97.43	57.46	28.76	5.81	0.85
	KEMLP	97.45(+0.02)	83.85(+26.39)	67.98(+39.22)	45.27(+39.46)	24.28(+23.43)
DOA (7x7)	Main	97.27	38.50	9.75	2.83	0.67
	KEMLP	97.22(−0.05)	80.89(+42.39)	63.40(+53.65)	49.20(+46.37)	31.04(+30.37)

Table 4.3: Correspondence between id numbers and attacks/corruptions

1	2	3
Physical Attack	Fog Corruption	Contrast Corruption
4	5	6
Brightness Corruption	$\mathcal{L}_\infty$ Attack ($\varepsilon = 4$, whitebox sensor)	$\mathcal{L}_\infty$ Attack ($\varepsilon = 8$, whitebox sensor)
7	8	9
$\mathcal{L}_\infty$ Attack ($\varepsilon = 16$, whitebox sensor)	$\mathcal{L}_\infty$ Attack ($\varepsilon = 32$, whitebox sensor)	Fog Attack ($\varepsilon = 256$, whitebox sensor)
10	11	12
Fog Attack ($\varepsilon = 512$, whitebox sensor)	Snow Attack ($\varepsilon = 0.25$, whitebox sensor)	Snow Attack ($\varepsilon = 0.75$, whitebox sensor)
13	14	15
Jpeg Attack ($\varepsilon = 0.125$, whitebox sensor)	Jpeg Attack ($\varepsilon = 0.25$, whitebox sensor)	Gabor Attack ($\varepsilon = 20$, whitebox sensor)
16	17	18
Gabor Attack ($\varepsilon = 40$, whitebox sensor)	Elastic Attack ($\varepsilon = 1.5$, whitebox sensor)	Elastic Attack ($\varepsilon = 2.0$, whitebox sensor)
19	20	21
$\mathcal{L}_\infty$ Attack ($\varepsilon = 4$, blackbox sensor)	$\mathcal{L}_\infty$ Attack ($\varepsilon = 8$, blackbox sensor)	$\mathcal{L}_\infty$ Attack ($\varepsilon = 16$, blackbox sensor)
22	23	24
$\mathcal{L}_\infty$ Attack ($\varepsilon = 32$, blackbox sensor)	Fog Attack ($\varepsilon = 256$, blackbox sensor)	Fog Attack ($\varepsilon = 512$, blackbox sensor)
25	26	27
Snow Attack ($\varepsilon = 0.25$, blackbox sensor)	Snow Attack ($\varepsilon = 0.75$, blackbox sensor)	Jpeg Attack ($\varepsilon = 0.125$, blackbox sensor)
28	29	30
Jpeg Attack ($\varepsilon = 0.25$, blackbox sensor)	Gabor Attack ($\varepsilon = 20$, blackbox sensor)	Gabor Attack ($\varepsilon = 40$, blackbox sensor)
31	32	33
Elastic Attack ($\varepsilon = 1.5$, blackbox sensor)	Elastic Attack ($\varepsilon = 2.0$, blackbox sensor)	$\mathcal{L}_\infty$ Attack ($\varepsilon = 4$, blackbox pipeline)
34	35	36
$\mathcal{L}_\infty$ Attack ($\varepsilon = 8$, blackbox pipeline)	$\mathcal{L}_\infty$ Attack ($\varepsilon = 16$, blackbox pipeline)	$\mathcal{L}_\infty$ Attack ($\varepsilon = 32$, blackbox pipeline)
37	38	39
Fog Attack ($\varepsilon = 256$, blackbox pipeline)	Fog Attack ($\varepsilon = 512$, blackbox pipeline)	Snow Attack ($\varepsilon = 0.25$, blackbox pipeline)
40	41	42
Snow Attack ($\varepsilon = 0.75$, blackbox pipeline)	Jpeg Attack ($\varepsilon = 0.125$, blackbox pipeline)	Jpeg Attack ($\varepsilon = 0.25$, blackbox pipeline)
43	44	45
Gabor Attack ($\varepsilon = 20$, blackbox pipeline)	Gabor Attack ($\varepsilon = 40$, blackbox pipeline)	Elastic Attack ($\varepsilon = 1.5$, blackbox pipeline)
46		
Elastic Attack ($\varepsilon = 2.0$, blackbox pipeline)		

this work, we conduct the evaluation on two dataset settings: *1) Setting-A:* a subset of GTSRB, which contains 12 types of German traffic signs. In total, there are 14880 samples in the training set, 972 samples in the validation set, and 3888 samples in the test set; *2) Setting-B:* a modified version of Setting-A, where the German stop signs are replaced with the U.S. stop signs from LISA, following (Eykholt et al., 2018).

MODELS We adopt the GTSRB-CNN architecture (Eykholt et al., 2018) as the main task model. KEMLP is constructed based on the main task model together with a set of auxiliary task models (e.g., color, shape, and content detectors). To train the weights of factors in KEMLP, we use β to denote the prior belief on balance between benign and adversarial distributions. More details on implementation are provided in Chapter A.

BASELINES To demonstrate the superiority of KEMLP, we compare it with two state-of-the-art baselines: **adversarial training** (Madry et al., 2017) and **DOA** (Wu, Tong, and Vorobeychik, 2019), which are strong defenses against $\mathcal{L}_p$ bounded attacks and physically attacks respectively.

For adversarial training, we adopt $\mathcal{L}_\infty$ bound $\varepsilon \in \{4, 8, 16, 32\}$ during training phase. Since adversarial training failed to make progress for $\varepsilon \in \{16, 32\}$, we use the curriculum training version (Cai, Liu, and Song, 2018), where the model is firstly trained on smaller ε with ε gradually increasing to the largest bound. For all versions of adversarial training in our implementation, we adopt 40 iterations of PGD attack with a step size of 1/255. In all cases, pixels are in $0 \sim 255$ range and the retraining takes 3000 training iterations with a batch size of 200 for each random iteration.

For DOA, we consider adversarial patches with the size of 5×5 and 7×7 respectively for rectangle occlusion during retraining. For both cases, we use an exhaustive search to pick the attack location and perform 30 iterations PGD inside the adversarial patch to generate noise. The retraining takes 5000 training iterations and the batch size is 200.

Thus, in total, we have 7 baseline CNN models (1 standard CNN model, 4 adversarially trained CNN models, 2 DOA trained CNN models), and we use id numbers $1 \sim 7$ to denote “GTSRB-CNN", “AdvTrain ($\varepsilon = 4$)", “AdvTrain ($\varepsilon = 8$)", “AdvTrain ($\varepsilon = 16$)", “AdvTrain ($\varepsilon = 32$)", “DOA (5x5)", “DOA (7x7)", respectively in Figure 4.3.

EVALUATED ATTACKS AND CORRUPTIONS We consider four types of attacks for thorough evaluation: *1) physical attacks* on stop signs (Eykholt et al., 2018); *2) $\mathcal{L}_\infty$ bounded attacks* (Madry et al., 2017) with $\varepsilon \in \{4, 8, 16, 32\}$; *3) Unforeseen attacks*, which produce a diverse set of unforeseen test distributions (e.g. Elastic, JPEG, Fog) distinct from $\mathcal{L}_p$ bounded perturbation (Kang et al., 2019); *4) common corruptions* (Hendrycks and Dietterich, 2019). For each attack, we consider both the *whitebox attack* against the main task model and *blackbox attack* by distilling either the main task model or the whole KEMLP pipeline.

Since our constructed KEMLP pipeline is a compound model consisting of multiple sub-models, some of which are not differentiable, we can not directly generate adversarial examples via the standard end-to-end white-box attack. Alternatively, we further propose three different attack settings to evaluate the robustness of our KEMLP pipeline: **1)White-box sensor attack,** where adversarial examples are generated by directly applying gradient methods to the main task model of the KEMLP pipeline in a white-box fashion; **2)Black-box sensor attack.** In this setting, we train substitute model of the main task model using the same model architecture and the same standard training data, and generate adversarial examples with this substitute model; **3)Black-box pipeline attack,** in which we generate adversarial examples with a substitute model, which is obtained via distilling the whole KEMLP pipeline. For this setting, a substitute model with the same GTSRB-CNN architecture is trained on a synthetic training set, where all the images are from the original training set, while the labels are generated by the pipeline model. Then all the models are evaluated on the same set of adversarial test samples crafted on the trained substitute.

Specifically, **1) For $\mathcal{L}_\infty$ attack,** we consider the strength of $\varepsilon \in \{4, 8, 16, 32\}$ in our evaluation. 1000 iterations of standard PGD (Madry et al., 2017) with a step size of $1/255$ is used to craft the adversarial examples, and all the three attack settings introduced above are respectively applied; **2) For unforeseen attacks**, we consider the Fog, Snow, JPEG, Gabor and Elastic attacks suggested in (Kang et al., 2019), which are all gradient-based worst-case adversarial attacks, generating diverse test distributions distinct from the common $\mathcal{L}_p$ bounded attacks. For Fog attack, we consider $\varepsilon \in \{256, 512\}$. For Snow attack, we evaluate for $\varepsilon \in \{0.25, 0.75\}$ respectively. For JPEG attack, we adopt the parameters $\varepsilon \in \{0.125, 0.25\}$. For Gabor attack, $\varepsilon \in \{20, 40\}$ are tested. Finally, $\varepsilon \in \{1.5, 2.0\}$ are considered for Elastic attack. Since all of these attacks are gradient based, we also apply the three different settings above to generate adversarial examples respectively; **3) For physical attacks on stop signs,** we directly use the same stickers (i.e., the same color and mask) generated in (Eykholt et al., 2018) to attack the same 40 stop sign samples, and we also adopt the same end-to-end classification model used in (Eykholt et al., 2018) to construct KEMLP model. Since our ultimate goal is defense, we follow the same practice in (Wu, Tong, and Vorobeychik, 2019), where we only consider the digital representation of the attack instead of the real physical implementation, ignoring issues like the attack's robustness to different viewpoints and environments. Thus, we implement the physical stop sign attack by directly placing the stickers on the stop sign samples in digital space; **4) For common corruptions,** we evaluate our models with 15 categories of corruptions suggested in (Hendrycks and Dietterich, 2019). Empirically, in our traffic sign identification task, only 3 types of corruptions out of the 15 categories effectively reduce the accuracy (with a margin over 10%) of our standard GTSRB-CNN model. Thus, we only present the evaluation results of our models against the three most successful corruption — Fog, Contrast, Brightness. (Note that, here we use Fog corruption which is similar to the Fog attack in unforeseen attacks. However, they are different in that the Fog corruption here is not adversarially generated like that in Fog attack.)

Thus, based on different attack/corruption methods and attack settings, in total, we have 46 different attacks/corruptions. In Figure 4.5 and Figure 4.6, we use id numbers $1 \sim 46$ to denote all the attacks we evaluate on, and we present the correspondence between id numbers and attacks in Table 4.3.

Table 4.4: Accuracy (%) under whitebox unforeseen attacks ($\beta = 0.8$)

		Clean	Fog-256	Fog-512	Snow-0.25	Snow-0.75	Jpeg-0.125
GTSRB-CNN	Main	99.38	59.65	34.18	56.58	24.54	55.74
	KEMLP	98.28(−1.10)	76.95(+17.30)	62.83(+28.65)	78.94(+22.36)	53.22(+28.68)	79.63(+23.89)
AdvTrain ($\varepsilon = 4$)	Main	97.94	55.53	29.50	66.31	32.61	56.58
	KEMLP	97.89(−0.05)	76.08(+20.55)	61.96(+32.46)	80.45(+14.14)	57.84(+25.23)	84.23(+27.65)
AdvTrain ($\varepsilon = 8$)	Main	93.72	50.03	23.56	63.71	34.93	57.56
	KEMLP	96.79(+3.07)	76.59(+26.56)	63.97(+40.41)	81.40(+17.69)	57.07(+22.14)	85.11(+27.55)
AdvTrain ($\varepsilon = 16$)	Main	84.54	47.92	19.75	66.46	37.60	66.56
	KEMLP	94.68(+10.14)	77.13(+29.21)	64.38(+44.63)	81.64(+15.18)	58.20(+20.60)	86.99(+20.43)
AdvTrain ($\varepsilon = 32$)	Main	74.74	48.71	22.84	61.78	38.91	63.58
	KEMLP	91.46(+16.72)	79.22(+30.51)	66.33(+43.49)	81.20(+19.42)	64.53(+25.62)	86.70(+23.12)
DOA (5x5)	Main	97.43	58.00	32.69	61.19	28.34	41.13
	KEMLP	97.45(+0.02)	76.85(+18.85)	63.07(+30.38)	78.78(+17.59)	56.76(+28.42)	78.60(+37.47)
DOA (7x7)	Main	97.27	59.88	38.01	62.47	30.17	23.46
	KEMLP	97.22(−0.05)	78.09(+18.21)	62.76(+24.75)	79.68(+17.21)	58.26(+28.09)	74.25(+50.79)

Table 4.5: Accuracy (%) under whitebox unforeseen attacks ($\beta = 0.8$)

		Jpeg-0.25	Gabor-20	Gabor-40	Elastic-1.5	Elastic-2.0
GTSRB-CNN	Main	27.01	57.25	32.41	44.78	24.31
	KEMLP	63.40(+36.39)	80.17(+22.92)	65.20(+32.79)	69.34(+24.56)	52.37(+28.06)
AdvTrain ($\varepsilon = 4$)	Main	28.11	73.30	46.76	57.25	30.09
	KEMLP	68.57(+40.46)	81.48(+8.18)	65.77(+19.01)	71.19(+13.94)	50.33(+20.24)
AdvTrain ($\varepsilon = 8$)	Main	26.16	76.72	53.76	48.25	24.46
	KEMLP	68.70(+42.54)	85.29(+8.57)	68.90(+15.14)	68.78(+20.53)	49.31(+24.85)
AdvTrain ($\varepsilon = 16$)	Main	34.23	78.01	64.33	55.48	32.28
	KEMLP	70.40(+36.17)	87.42(+9.41)	72.61(+8.28)	67.31(+11.83)	50.28(+18.00)
AdvTrain ($\varepsilon = 32$)	Main	43.49	70.37	65.20	54.58	39.45
	KEMLP	73.38(+29.89)	87.04(+16.67)	74.92(+9.72)	66.38(+11.80)	54.76(+15.31)
DOA (5x5)	Main	11.29	55.43	29.55	58.02	32.74
	KEMLP	61.78(+50.49)	80.25(+24.82)	63.89(+34.34)	72.69(+14.67)	57.51(+24.77)
DOA (7x7)	Main	3.65	54.58	27.29	56.33	30.97
	KEMLP	61.39(+57.74)	79.06(+24.48)	62.29(+35.00)	71.27(+14.94)	55.09(+24.12)

4.6.2 *Evaluation Results*

Here we compare the clean accuracy, robust accuracy, and weighted robustness (W-Robust Accuracy) for baselines and KEMLP under different attacks and settings.

CLEAN ACCURACY OF KEMLP First, we present the clean accuracy of KEMLP and baselines in Figure 4.3 and Table 4.1–Table 4.6. As demonstrated, the clean accuracy of KEMLP is generally high (over 90%), by either main-

Table 4.6: Accuracy (%) under common corruptions ($\beta = 0.2$)

		Clean	Fog	Contrast	Brightness
GTSRB-CNN	Main	99.38	76.23	57.61	85.52
	KEMLP	98.28(−1.10)	78.14(+1.91)	72.43(+14.82)	89.58(+4.06)
AdvTrain ($\varepsilon = 4$)	Main	97.94	63.81	42.31	78.47
	KEMLP	97.89(−0.05)	70.29(+6.48)	67.46(+25.16)	86.70(+8.23)
AdvTrain ($\varepsilon = 8$)	Main	93.72	59.05	31.97	78.47
	KEMLP	96.79(+3.07)	67.41(+8.36)	66.69(+34.72)	85.91(+7.44)
AdvTrain ($\varepsilon = 16$)	Main	84.54	56.58	34.31	78.01
	KEMLP	94.68(+10.14)	66.80(+10.22)	68.39(+34.08)	86.14(+8.13)
AdvTrain ($\varepsilon = 32$)	Main	74.74	50.87	30.45	71.30
	KEMLP	91.46(+16.72)	64.94(+14.07)	68.31(+37.86)	83.20(+11.90)
DOA (5x5)	Main	97.43	73.95	62.24	83.92
	KEMLP	97.45(+0.02)	76.08(+2.13)	74.38(+12.14)	87.60(+3.68)
DOA (7x7)	Main	97.27	73.41	57.54	83.56
	KEMLP	97.22(−0.05)	76.00(+2.59)	72.40(+14.86)	87.78(+4.22)

Table 4.7: Adversarial accuracy under black-box sensor $\mathcal{L}_\infty$ attack, $\beta = 0.2$ (Accuracy %)

		$\varepsilon = 0$	$\varepsilon = 4$	$\varepsilon = 8$	$\varepsilon = 16$	$\varepsilon = 32$
GTSRB-CNN	Main	99.38	85.16	67.98	47.56	25.69
	KEMLP	98.28(−1.10)	91.36(+6.20)	79.53(+11.55)	61.21(+13.65)	41.85(+16.16)
AdvTrain ($\varepsilon = 4$)	Main	97.94	94.88	90.23	72.99	50.75
	KEMLP	97.89(−0.05)	95.88(+1.00)	90.66(+0.43)	77.01(+4.02)	55.56(+4.81)
AdvTrain ($\varepsilon = 8$)	Main	93.72	91.49	89.02	80.56	64.76
	KEMLP	96.79(+3.07)	94.29(+2.80)	90.23(+1.21)	81.40(+0.84)	65.92(+1.16)
AdvTrain ($\varepsilon = 16$)	Main	84.54	83.05	82.00	79.76	73.20
	KEMLP	94.68(+10.14)	90.72(+7.67)	86.52(+4.52)	80.02(+0.26)	70.47(−2.73)
AdvTrain ($\varepsilon = 32$)	Main	74.74	73.64	72.79	71.91	67.77
	KEMLP	91.46(+16.72)	86.60(+12.96)	81.66(+8.87)	75.69(+3.78)	66.77(−1.00)
DOA (5x5)	Main	97.43	84.93	70.70	52.44	33.15
	KEMLP	97.45(+0.02)	92.21(+7.28)	81.56(+10.86)	64.07(+11.63)	45.70(+12.55)
DOA (7x7)	Main	97.27	79.48	65.77	48.71	30.99
	KEMLP	97.22(−0.05)	90.56(+11.08)	80.20(+14.43)	62.55(+13.84)	44.24(+13.25)

taining the high clean accuracy of strong main task models (e.g., vanilla DNN) or improving upon the weak main task models with relatively low clean accuracy (e.g., adversarially trained models). It is clear that KEMLP can relax the tradeoff between benign and robust accuracy and maintain the high performance for both via knowledge integration.

Table 4.8: Adversarial accuracy under black-box pipeline $\mathcal{L}_\infty$ attack, $\beta = 0.2$ (Accuracy %)

		$\varepsilon = 0$	$\varepsilon = 4$	$\varepsilon = 8$	$\varepsilon = 16$	$\varepsilon = 32$
GTSRB-CNN	Main	99.38	81.17	60.52	37.60	24.28
	KEMLP	98.28(−1.10)	89.76(+8.59)	76.18(+15.66)	56.07(+18.47)	37.50(+13.22)
AdvTrain ($\varepsilon = 4$)	Main	97.94	94.42	88.32	66.08	46.60
	KEMLP	97.89(−0.05)	95.88(+1.46)	89.61(+1.29)	71.91(+5.83)	51.57(+4.97)
AdvTrain ($\varepsilon = 8$)	Main	93.72	90.72	87.11	75.49	58.64
	KEMLP	96.79(+3.07)	94.16(+3.44)	89.40(+2.29)	77.31(+1.82)	60.26(+1.62)
AdvTrain ($\varepsilon = 16$)	Main	84.54	82.87	81.46	77.13	70.09
	KEMLP	94.68(+10.14)	90.87(+8.00)	86.37(+4.91)	78.06(+0.93)	68.44(−1.65)
AdvTrain ($\varepsilon = 32$)	Main	74.74	73.66	72.35	70.16	66.08
	KEMLP	91.46(+16.72)	86.70(+13.04)	81.74(+9.39)	73.46(+3.30)	65.23(−0.85)
DOA (5x5)	Main	97.43	81.94	66.13	48.28	33.26
	KEMLP	97.45(+0.02)	91.13(+9.19)	78.88(+12.75)	61.42(+13.14)	42.36(+9.10)
DOA (7x7)	Main	97.27	77.85	63.68	46.55	31.79
	KEMLP	97.22(−0.05)	89.84(+11.99)	77.78(+14.10)	60.39(+13.84)	40.90(+9.11)

Table 4.9: Adversarial accuracy under black-box sensor unforeseen attack, $\beta = 0.2$ (Accuracy %)

		clean	fog-256	fog-512	snow-0.25	snow-0.75	jpeg-0.125
GTSRB-CNN	Main	99.38	77.55	59.93	78.50	45.34	83.10
	KEMLP	98.28(−1.10)	84.03(+6.48)	68.54(+8.61)	83.08(+4.58)	57.77(+12.43)	88.97(+5.87)
AdvTrain ($\varepsilon = 4$)	Main	97.94	70.68	54.06	77.70	49.67	87.45
	KEMLP	97.89(−0.05)	79.37(+8.69)	64.38(+10.32)	82.38(+4.68)	59.21(+9.54)	91.80(+4.35)
AdvTrain ($\varepsilon = 8$)	Main	93.72	67.70	53.73	76.13	51.75	86.27
	KEMLP	96.79(+3.07)	76.70(+9.00)	64.97(+11.24)	80.99(+4.86)	60.39(+8.64)	91.02(+4.75)
AdvTrain ($\varepsilon = 16$)	Main	84.54	66.44	49.64	75.15	52.73	81.58
	KEMLP	94.68(+10.14)	77.11(+10.67)	63.84(+14.20)	81.58(+6.43)	60.73(+8.00)	87.68(+6.10)
AdvTrain ($\varepsilon = 32$)	Main	74.74	65.82	50.18	71.97	52.37	72.61
	KEMLP	91.46(+16.72)	77.62(+11.80)	64.56(+14.38)	79.60(+7.63)	61.09(+8.72)	83.85(+11.24)
DOA (5x5)	Main	97.43	78.24	62.32	79.55	56.69	86.55
	KEMLP	97.41(−0.02)	84.26(+6.02)	69.08(+6.76)	83.36(+3.81)	62.58(+5.89)	90.41(+3.86)
DOA (7x7)	Main	97.27	76.34	61.32	79.30	55.94	83.20
	KEMLP	97.22(−0.05)	82.74(+6.40)	68.52(+7.20)	83.74(+4.44)	62.47(+6.53)	89.04(+5.84)

ROBUSTNESS AGAINST DIVERSE ATTACKS We then present the robustness of KEMLP based on different main task models against the physical attacks, which is very challenging to defend currently (Table 4.1), ℓ_p bounded attacks (Table 4.2), unseen attacks (Table 4.4 and Table 4.5), and common corruptions (Table 4.6) under whitebox attack setting. The corresponding results for blackbox setting can be found in Appendix. From the tables, we observe that KEMLP achieves significant *robustness gain* over baselines. Note that although adversarial training improves the robustness against $\mathcal{L}_\infty$ attacks and DOA helps to defend against physical attacks, they are not robust to other

Table 4.10: Adversarial accuracy under black-box sensor unforeseen attack, $\beta = 0.2$ (Accuracy %)

		jpeg-0.25	gabor-20	gabor-40	elastic-1.5	elastic-2.0
GTSRB-CNN	Main	65.90	75.36	59.26	77.16	57.64
	KEMLP	74.90(+9.00)	84.88(+9.52)	70.04(+10.78)	82.10(+4.94)	66.69(+9.05)
AdvTrain ($\varepsilon = 4$)	Main	72.84	88.14	68.21	83.38	70.09
	KEMLP	80.09(+7.25)	91.51(+3.37)	75.05(+6.84)	84.80(+1.42)	73.12(+3.03)
AdvTrain ($\varepsilon = 8$)	Main	76.75	89.25	76.47	80.71	67.85
	KEMLP	82.54(+5.79)	91.56(+2.31)	79.45(+2.98)	83.26(+2.55)	71.37(+3.52)
AdvTrain ($\varepsilon = 16$)	Main	77.78	83.90	82.48	76.23	68.26
	KEMLP	82.77(+4.99)	89.27(+5.37)	83.44(+0.96)	81.07(+4.84)	71.55(+3.29)
AdvTrain ($\varepsilon = 32$)	Main	71.09	76.26	77.16	68.03	64.38
	KEMLP	79.30(+8.21)	85.60(+9.34)	80.09(+2.93)	77.67(+9.64)	70.81(+6.43)
DOA (5x5)	Main	71.32	82.23	67.28	87.96	75.75
	KEMLP	77.98(+6.66)	87.06(+4.83)	73.69(+6.41)	86.09(−1.87)	75.90(+0.15)
DOA (7x7)	Main	66.10	82.25	67.54	86.73	73.77
	KEMLP	76.44(+10.34)	87.60(+5.35)	74.51(+6.97)	85.91(−0.82)	75.49(+1.72)

Table 4.11: Adversarial accuracy under black-box pipeline unforeseen attack, $\beta = 0.2$ (Accuracy %)

		clean	fog-256	fog-512	snow-0.25	snow-0.75	jpeg-0.125
GTSRB-CNN	Main	99.38	71.17	49.13	70.73	36.45	75.44
	KEMLP	98.28(−1.10)	78.96(+7.79)	60.65(+11.52)	80.02(+9.29)	52.16(+15.71)	85.31(+9.87)
AdvTrain ($\varepsilon = 4$)	Main	97.94	66.23	47.33	73.46	42.10	84.23
	KEMLP	97.89(−0.05)	74.97(+8.74)	58.62(+11.29)	80.63(+7.17)	54.09(+11.99)	90.84(+6.61)
AdvTrain ($\varepsilon = 8$)	Main	93.72	63.14	45.14	72.87	46.66	84.59
	KEMLP	96.79(+3.07)	72.89(+9.75)	58.02(+12.88)	79.73(+6.86)	55.86(+9.20)	90.59(+6.00)
AdvTrain ($\varepsilon = 16$)	Main	84.54	62.32	42.98	73.23	50.08	80.97
	KEMLP	94.68(+10.14)	73.48(+11.16)	58.18(+15.20)	80.45(+7.22)	57.54(+7.46)	86.99(+6.02)
AdvTrain ($\varepsilon = 32$)	Main	74.74	61.86	45.01	70.47	50.57	72.38
	KEMLP	91.46(+16.72)	73.33(+11.47)	58.49(+13.48)	78.94(+8.47)	58.67(+8.10)	83.33(+10.95)
DOA (5x5)	Main	97.43	75.01	56.97	77.67	53.14	83.15
	KEMLP	97.41(−0.02)	80.40(+5.39)	64.40(+7.43)	82.28(+4.61)	59.52(+6.38)	77.88(−1.00)
DOA (7x7)	Main	97.27	73.97	57.05	77.21	53.55	81.40
	KEMLP	97.22(−0.05)	80.04(+6.07)	64.17(+7.12)	82.46(+5.25)	59.75(+6.20)	88.30(+6.90)

types of attacks or corruptions. In contrast, KEMLP presents general robustness against a range of attacks and corruptions without further adaptation.

performance stability of kemlp We conduct additional ablation studies on β, representing the prior belief on the benign and adversarial distribution balance. We set $\beta = 0.5$ for KEMLP indicating a balanced random guess for the distribution tradeoff. We show the clean accuracy and robustness

Table 4.12: Adversarial accuracy under black-box pipeline unforeseen attack, $\beta = 0.2$ (Accuracy %)

		jpeg-0.25	gabor-20	gabor-40	elastic-1.5	elastic-2.0
GTSRB-CNN	Main	51.98	72.61	53.47	70.88	54.53
	KEMLP	67.64(+15.66)	84.13(+11.52)	69.24(+15.77)	80.66(+9.78)	67.80(+13.27)
AdvTrain ($\varepsilon = 4$)	Main	65.07	87.29	66.95	82.10	68.80
	KEMLP	76.00(+10.93)	90.61(+3.32)	74.77(+7.82)	84.85(+2.75)	74.95(+6.15)
AdvTrain ($\varepsilon = 8$)	Main	71.35	88.86	73.74	80.30	67.88
	KEMLP	80.02(+8.67)	90.92(+2.06)	77.93(+4.19)	83.80(+3.50)	73.77(+5.89)
AdvTrain ($\varepsilon = 16$)	Main	76.26	83.51	81.22	75.80	68.75
	KEMLP	80.92(+4.66)	88.30(+4.79)	82.23(+1.01)	81.71(+5.91)	72.69(+3.94)
AdvTrain ($\varepsilon = 32$)	Main	69.70	76.16	76.39	68.65	64.99
	KEMLP	77.42(+7.72)	84.95(+8.79)	79.09(+2.70)	78.37(+9.72)	71.45(+6.46)
DOA (5x5)	Main	63.79	82.07	65.77	88.17	78.88
	KEMLP	73.69(+9.90)	87.04(+4.97)	73.43(+7.66)	86.99(−1.18)	77.88(−1.00)
DOA (7x7)	Main	62.68	82.15	67.28	87.42	78.27
	KEMLP	73.48(+10.80)	87.09(+4.94)	73.95(+6.67)	86.42(−1.00)	78.58(+0.31)

of KEMLP and baselines under diverse 46 attacks in Figure 4.3. We can see that KEMLP consistently and significantly outperforms the baselines, which indicates the performance stability of KEMLP regarding different distribution ratio β.

We now present the evaluation results of $\mathcal{L}_\infty$ attack and unforeseen attacks under blackbox sensor and blackbox pipeline attack settings. Specifically, we present the two blackbox results for $\mathcal{L}_\infty$ attack in Table 4.7 and Table 4.8, and accordingly the two blackbox results for unforeseen attacks in Table 4.9, Table 4.9, Table 4.11 and Table 4.11.

As shown, similar trends in whitebox sensor attack setting can also be observed in these two blackbox attack settings, which indicates that the robustness does not just come from gradient masking (Athalye, Carlini, and Wagner, 2018; Carlini and Wagner, 2017b).

4.7 SUMMARY

In this part, we introduced KEMLP, which integrates *domain knowledge* with a set of weak auxiliary models to enhance the ML robustness against a diverse set of adversarial attacks and corruptions. We developed theory identifying how and when knowledge help with ML robustness. In particular, we observed that introducing the knowledge via first-order logic functions helps KEMLP to achieve strong guarantees as, in a way, it employs the knowledge where it is powerful, as opposed to encode all relations as identity knowledge. This enabled KEMLP to bring not only robust but also clean accuracy improvement over the main task model.

We conducted several experiments to demonstrate the capabilities of such knowledge-based framework. Under almost all evaluated settings, our KEMLP framework significantly improved the robustness of a single neural network model. Moreover, since our KEMLP framework based defense is orthogonal to the neural network training based defenses (e.g. adversarial training), we can always combine these two defense methodologies and get the best models.

Overall, KEMLP framework offers general robustness (that is, robustness against a diverse set of attacks or corruptions). As shown by our results, although adversarial training improves the robustness against $\mathcal{L}_\infty$ attack and DOA helps to defend against stop sign attacks, they are neither generally robust to other types of attacks or corruptions. In contrast, our KEMLP models exhibits general robustness against a diverse collection of attacks and corruptions. We thus believe that such knowledge-based methods can have a substantial impact in bringing general robustness to ML pipelines.

5 CONCLUSION

Science advances one funeral at a time.

— Max Planck (1950)

5.1 SUMMARY

Designing cost-efficient and robust ML pipelines for data-intensive applications has been one of the core challenges of artificial intelligence. In this dissertation, we took a view through the lens of data and addressed several open questions in regards to hardware efficiency, label efficiency and robustness of ML systems without much dependency on the learner. We found that curations of the cost problem through data extend the generalizability of repeatability of the proposed techniques. Our findings were accompanied by theoretical guarantees on the quality of learning, enabling practitioners to optimize their budget according to their needs on learning quality while still maintaining robustness of the results. In particular, this dissertation encompassed the following contributions:

In Part i of this thesis, we extended the usability of low precision training to compressive sensing solvers. Among all sparsity-constrained minimization methods, we specifically focused on the normalized IHT as otherwise methods often require strong constraints on the measurement matrix for provable guarantees as well as practical performance. In particular, we presented a low precision variant IHT, which we named QIHT. For provable guarantees and practical performance, we benefited from the scale invariant property of measurement matrix that is inherited from the normalized IHT. This allowed us to have a high confidence on the practical performance in regimes of non-symmetric RIP, and rescale the measurement matrix without amplifying the additive thermal noise in electronics. Our theoretical analysis indicated that the data quantization has only a mild effect on the signal recovery error while requiring a slightly tighter RIP condition in exchange. As the adaptive step size enables strong practical performance also outside of RIP regime, the performance of QIHT seems not to be highly bounded by the RIP condition. Inspired by our theoretical findings on the potential of low precision compressive sensing, we formulated the full sky imaging of radio telescopes as a compressive sensing problem. We then observed through application of QIHT to radio astronomy and MRI that lowering the precision of the data can significantly accelerate image recovery with negligible loss of quality. From the hardware point of view, with a small set of operators that are newly introduced, we were able to exploit the existing designs to implement our quantization framework on hardware.

In Part ii of this thesis, we studied label-efficient model selection. The driving force behind our motivation was the constant need to adapt downstream model in the ML pipeline for the distribution shift of production data from disparate sources. To cope with it, we proposed the idea of model selection by labeling only a small portion of freshly arriving data. We investigated the potential of the state-of-the-art active learners, and found out that tailoring them for "model testing" is needed. Upon that, we developed Model Picker a novel, principled and efficient model selection approach. We assumed a stream-based setting, where the data examples arrive in a stream, and the learner actively decides to query the label per each example. As the first framework exploring data sampling only for the task of pretrained model selection, we introduced a novel and fair evaluation framework, which is nontrivial for the stream-based setting. We collected several model collections on well-studied ML benchmarks, each of which has different properties. We then conducted extensive experiments, comparing our algorithm with a range of other adapted active learners. To reach the same accuracy, competing methods often required significantly more labels. Apart from the relative performance, on the dataset with thousands of labels, Model Picker was demonstrated to be capable of identifying the best model by querying a mere amount of labels with high confidence, and with low regret across the stream. We established Model Picker as the state-of-the-art for this problem. We also made the Model Picker framework publicly available[1] and the experimental results reproducible.[2].

Next in Part ii, we took the first step towards the theoretical analysis on the causes of oversmoothing in graph neural networks, and the impact of graph decomposition that alleviates it. We took an information theoretical view and analyzed the infinite-sample behaviour of Shannon's mutual information between the input and output of lth layer. We found the information theoretical perspective to provide a much simpler but equally tight analysis for GCN compared to the existing analysis, but more importantly, our analysis made it possible to analyze more complex cases for GraphCNN with the presence of decomposition.

In Part iii, we developed a general defense technique, KEMLP, that can be employed at every stage of the ML pipeline and against any kind of attack and corruption. We achieved this by integrating a diverse set of weak auxiliary models based on their logical relationships to the main DNN model that performs the target task. Theoretically, we provided convergence results and prove that, under mild conditions, the prediction of KEMLP is more robust than that of the main DNN model. We took road sign recognition as the example use case and leveraged the relationships between road signs and their shapes and contents as domain knowledge. We showed that compared with adversarial training and other baselines, KEMLP achieved higher robustness against physical attacks, L_p bounded attacks, unforeseen attacks, and natural corruptions under both whitebox and blackbox settings, while still maintaining high clean accuracy.

5.2 IMPACT

We believe that curating cost efficiency and robustness through the manipulations and interventions of data can enable novel techniques on many fronts. In what follows, we briefly mention what we believe the impact of each

[1] We demonstrate the Model Picker framework here: https://github.com/easeml/modelpicker

[2] Our results are reproducible here: https://github.com/DS3Lab/online-active-model-selection

contribution is. It must be noted that the broad range of impacts we refer below is thanks to this data-centric view we adopt, which inherently brings benefits in many dimensions.

First of all, learning with quantized data can bring substantial benefits to processing high volumes of data, from efficient data communication and computation to storage. In this thesis, we mainly studied its benefit for accelerated computation[3]. In fact, in many scientific applications, the data must be processed in a speedy manner such that the anomalies can be detected in real-time, upon which more resources can be spent to investigate them. For example, this is often encountered in medical imaging applications and astronomy, where the resources are used to "zoom in" particular areas by fusing the stream of waves to target the direction where the anomaly occurs. In addition to its computational benefits, data compression naturally reduces the physical layer communication overhead for far-field sensor systems. Take the Square Kilometre Array, the largest radio telescope ever built with thousands of dishes, as an example. By mid-2020, the Square Kilometre Array is expected to achieve raw data throughput of 62 Exabytes (Mattmann et al., 2014). Compressing the raw data, as practiced in this thesis, can significantly reduce the amount of data sent from antennas to a central signal processor. Another byproduct of low precision training is the efficient storage of data. This is in particular crucial for medical applications, which can highly benefit from reduced storage needs to keep patients' raw data, as discussed in (Langer, 2011; Poldrack, Mumford, and Nichols, 2011) for the case of MRI.

Secondly, the ability to choose the most *suitable* model to use is becoming an indispensable component of modern ML eco-systems as ML is being applied to more and more critical applications with increasing impact on society. In this thesis, we focus on a specific instance of this challenge where we tackle the distribution shift and propose a label-efficient way of picking the best model from a set of candidate models in terms of prediction accuracy. Performing this in a label-efficient manner is particularly severe for applications where data scarcity is acute such as personalized healthcare, clinical trials, robot reinforcement learning, and cognitively inspired natural language

[3] which has made it to a new library for efficient computation using low-precision data on state-of-the-art CPUs (Stojanov et al., 2018)

processing. Potentially, our method can have an impact on the deployment and maintenance of ML models in many application domains. Our framework is agnostic as it imposes no assumption on neither the models nor the data, which further extends its usability to many other scenarios such as choosing the best pretrained model for transfer learning or assessing the relative bias of models for a given task in a label-efficient fashion.

As the following contribution, our exploration of graph decomposition and its role in overcoming oversmoothing is inherently triggered by the interest to design decomposition strategies that improve the performance of graph neural networks. Understanding and utilizing information preserving ability of the decomposition in general graph-structured data can enable the design of practical algorithms that takes advantage of graph decomposition. One such example is the novel technique called connectivity-aware graph decomposition, which strikes a balance between information loss and model performance trade-off. We believe this theoretical framework can further enable novel graph decomposition strategies that are powerful.

Finally, from the robust ML perspective, we incorporated knowledge to attain better adversarial robustness. Unlike end-to-end neural network models, knowledge fusion tackles sensory information processing and decision making separately. First, it captures sensory information in the sensing domain via observations. It then performs decision making in the logic domain via logical reasoning. Such design brings two benefits. First, the sensory features extracted in the sensing domain are in a human-understandable format with clear semantic meaning, and the relations between features and candidate decisions in the logical domain also have clear human knowledge as the basis. For instance, in our implementation for traffic sign identification, the extracted sensory features output by the sensors are all Boolean variables representing meaningful concepts, such that *whether the given sign is of the shape of an octagon*, *whether the content on the sign is the characters "S", "T", "O", "P"*. In the similar spirit, the decision rules as *"Stop sign must be of the shape of octagon"* have a clear logical basis. As a result, in our framework, the vulnerability of non-robust features (Ilyas et al., 2019; Tsipras et al., 2019) is restricted to the sensing domain and tackled by the logic domain via knowledge rules. We then only need to account for the sensory errors, as the sensors may still be attacked. Nevertheless, they are technically tractable and can be well controlled, as we illustrated in Part iii via making robust decisions despite the weakness of sensors. We believe that these benefits are what make our framework inherently robust to combat perturbations and corruptions not only in adversarial setting but also in changing environments with shifting distributions. Moreover, despite that adversarial robustness is often at the cost of clean accuracy (Mohapatra et al., 2020; Tsipras et al., 2019), our knowledge-based framework achieved robustness without harming the clean accuracy, as demonstrated in Part iii. This observation indicates the potential of knowledge fusion via logic rules not only for the adversarial robustness but the prediction accuracy of ML models in general, moreover, in an interpretable way for humans.

5.3 FUTURE WORK

There are several future directions for each of the contributions we presented. We first separately visit them below.

The framework of low precision training for scientific instruments we studied in this thesis facilitates several orthogonal research dimensions to pursue. Firstly, QIHT can be further devised to work with end-to-end low precision data representation where the precision of gradient vector is also reduced. This can unlock the full potential of low precision training in accelerated computation. The end-to-end reduced precision representations studied in the case of deep learning (Hubara et al., 2017; Miyashita, Daisuke and Lee, Edward H and Murmann, Boris, 2016; Miyashita, Lee, and Murmann, 2016; Rastegari et al., 2016) and linear models (Zhang et al., 2017) are shown to bring an order of magnitude acceleration to computation, holding a promise also for accelerated compressive sensing. However, learning with end-to-end low precision data requires rigorous treatment and a deep understanding of the limitations when applied to mission-critical scientific applications such as medical imaging, mass-spectrometers, and microscopes due to potentially high-stake signal loss. We thus believe that this extension could be a challenging yet intriguing next direction to explore. Secondly, the tools we developed in the context of low precision compressive sensing (such as stochastic quantization and optimization of compression ratio) are general enough to extend to other greedy recovery algorithms. In this thesis, we scope ourselves to iterative thresholding, but our fundamental findings are worth validating for other sparse reconstruction or ℓ_1 minimization problems, which also demand high processing and storage capabilities from the hardware when handling large amounts of data. Finally, as a third potential direction, low precision compressive sensing framework can be highly beneficial to other real-world applications which, despite having a sparse underlying representation, employ more complex algorithms than compressive sensing. An immediate example of this is compressive sensing with *beamforming* (Edelmann and Gaumond, 2011; Gurbuz, McClellan, and Cevher, 2008; Han and Wang, 2015). Low precision compressive sensing with beamforming is nontrivial. To exemplify this, take the phased-array example, in which beamforming is achieved by sending the raw signal to the central processor in a hierarchical way to reduce the amount of data that needs to be transferred. Hence, the mapping from the sparse coefficients to the measurements contains several measurement matrices that are combined in a non-linear fashion. Investigating the potential of reduced precision training to function with such strategies is also an exciting research direction and can bring substantial improvement to the efficiency of data transmission over hundreds of kilometers. Lately, (Corda et al., 2022) demonstrated such a possibility where complex algorithms for radio telescope imaging with data at low precision achieve significant speed-up on FPGA.

Our work on label-efficient model selection can be further extended towards a setting in which the user at once has access to a pool of production data examples. In such a pool-based sampling case, one can rank the entire collection of data samples to select the most informative example instead of scanning through the data sequentially to decide whether to query a label. Despite the applicability of our approach to such a scenario where one can form a stream by sampling i.i.d. from the pool of samples, the availability of entire data collection can be exploited to further reduce the annotation costs with a more specialized strategy for pool-based scenarios. We believe the greedy information maximization strategies (Chen et al., 2015) can be utilized to rank and select informative samples. The scalability problem can be further addressed via sampling and labeling in batches of production data.

In our theoretical framework investigating oversmoothing in graph neural networks and the impact of graph decomposition on it, we assumed an infinite sample regime. Our findings can be explored in the finite sample regime via the information bottleneck principle (Saxe et al., 2019; Shamir, Sabato, and Tishby, 2010; Tishby, Pereira, and Bialek, 2000) to understand the role of graph decomposition further when combined with the regularization effect of the finite sample regime and to derive design principles for graph decomposition that improves the performance of GraphCNN.

As the future directions on the robustness front, there are several points to consider. In this thesis, we take the first step in designing a principled framework where we perform knowledge fusion by forming logical constraints based on domain knowledge. The logical constraints we introduced leverage logical relations among a set of semantic concepts induced by a predefined knowledge base. While our framework can be extended to other applications, for any knowledge system, one naturally needs domain experts to design the knowledge rules specific to that application. There is probably no universal strategy for aggregating knowledge for any arbitrary application; instead, application-specific constructions will be needed. We believe that our framework as a prototype demonstrated the benefit of such construction. Once the principled framework of knowledge fusion is ready, application-specific developments of knowledge rules will naturally follow, similar to what happened previously for knowledge-enriched joint inference. We think that a good starting point could be to utilize the existing knowledge APIs such that high-quality knowledge can be formed towards combining learning and explicit knowledge.

Ultimately, this dissertation addressed several open problems in cost efficiency and robustness of ML and proposed repeatable solutions that can be employed throughout the iterations of the ML lifecycle. We believe that it is the additional data-focused view we adopted that brought a higher degree of cost-efficiency, robustness, programmability, and repeatability to the ML process compared to what is achieved via solely model-focused strategies. As the data-centric ML research has been more and more pursued by the community, we hope that the insights of this dissertation contribute and bring impact to the attributes mentioned above. Moreover, the strategies we developed in this thesis are agnostic to the type and specifics of the ML model employed by the pipeline and hence can be accompanied by other model-centric approaches. We believe this is important to note as the success of artificial intelligence hinges jointly on high-quality data and well-conceived models.

AUTHOR'S PUBLICATIONS

This dissertation is largely based on the following publications. † indicates co-first authorship.

Nezihe Merve Gürel†, Xiangyu Qi, Luka Rimanic, Ce Zhang, and Bo Li (2021). "Knowledge Enhanced Machine Learning Pipeline against Diverse Adversarial Attacks." In: *International Conference on Machine Learning* 139, pp. 3976–3987.

Nezihe Merve Gürel, Kaan Kara, Alen Stojanov, Tyler Smith, Thomas Lemmin, Dan Alistarh, Markus Püschel, and Ce Zhang (2020). "Compressive Sensing Using Iterative Hard Thresholding with Low Precision Data Representation: Theory and Applications." In: *IEEE Transactions on Signal Processing* 68, pp. 4268–4282.

Nezihe Merve Gürel, Hansheng Ren, Yujing Wang, Hui Xue, Yaming Yang, and Ce Zhang (2020). "An Anatomy of Graph Neural Networks Going Deep via the Lens of Mutual Information: Exponential Decay vs. Full Preservation." In: *Deep Learning on Graphs, AAAI Conference on Artificial Intelligence*.

Nezihe Merve Gürel and Ce Zhang (2022). "Calibration of Data Precision for Interferometric Radio Astronomy." In: *submission*.

Xupeng Miao, Nezihe Merve Gürel†,Wentao Zhang, Wentao Zhang, Zhichao Han, Bo Li, Wei Min, Susie Xi Rao, Hansheng Ren, Yinan Shan, Yingxia Shao, et al. (2021). "DeGNN: Improving Graph Neural Networks with Graph Decomposition." In: *ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pp. 1223–1233.

Mohammad Reza Karimi, Nezihe Merve Gürel†, Bojan Karlaš, Johannes Rausch, Ce Zhang, and Andreas Krause (2021). "Online active model selection for pre-trained classifiers." In: *International Conference on Artificial Intelligence and Statistics*, pp. 307–315.

Nezihe Merve Gürel (2018). "Towards More Accurate Radio Telescope Images." In: *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 1983–1986.

Cedric Renggli, Luka Rimanic, Nezihe Merve Gürel, Bojan Karlaš, Wentao Wu, and Ce Zhang (2021). "A Data Quality-Driven View of MLOps." In: *IEEE Data Engineering Bulletin*, pp. 11–20.

Leonel Aguilar Melgar, ... Nezihe Merve Gürel†, ..., Ce Zhang (2021). "Ease. ML: A Lifecycle Management System for Machine Learning." In: *Proceedings of the Annual Conference on Innovative Data Systems Research*.

Further publications that are outside the scope of this thesis:

Bojan Karlaš, Peng Li, Renzhi Wu, Nezihe Merve Gürel, Xu Chu, Wentao Wu, and Ce Zhang (2020). "Nearest Neighbor Classifiers over Incomplete Information: From Certain Answers to Certain Predictions." In: *Very Large Data Bases*, pp. 255–267.

Nezihe Merve Gürel, Paul Hurley and Matthieu Simeoni (2017). "Denoising Radio Interferometric Images by Subspace Clustering." In: *2017 IEEE International Conference on Image Processing*, pp. 2134–2138.

Ruoxi Jia, David Dao, Boxin Wang, Frances Ann Hubis, Nezihe Merve Gürel, Bo Li, Ce Zhang, Costas J Spanos, and Dawn Song (2019a). "Efficient Task-specific Data Valuation for Nearest Neighbor Algorithms." In: *Very Large Data Bases*, pp. 1610–1623.

Ruoxi Jia, David Dao, Boxin Wang, Frances Ann Hubis, Nick Hynes, Nezihe Merve Gürel, Bo Li, Ce Zhang, Dawn Song, and Costas J Spanos (2019b). "Towards Efficient Data Valuation based on the Shapley Value." In: *International Conference on Artificial Intelligence and Statistics*, pp. 1167–1176.

Yuexiang Xie, Zhen Wang, Yaliang Li, Bolin Ding, Nezihe Merve Gürel, Ce Zhang, Minlie Huang, Wei Lin, and Jingren Zhou (2021). "Fives: Feature Interaction via Edge Search for Large-scale Tabular Data." In: *ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 3795–3805.

BIBLIOGRAPHY

Abe, Naoki and Hiroshi Mamitsuka (1998). "Query learning strategies using boosting and bagging." In: *Proceedings of the International Conference on Machine Learning*, pp. 1–9.

Ai, Albert, Alex Lapanowski, Yaniv Plan, and Roman Vershynin (2014). "One-bit compressed sensing with non-Gaussian measurements." In: *Linear Algebra and its Applications: Special Issue on Sparse Approximate Solution of Linear Systems* 441, pp. 222–239.

Albericio, Jorge, Patrick Judd, Tayler Hetherington, Tor Aamodt, Natalie Enright Jerger, and Andreas Moshovos (2016). "Ineffectual-neuron-free deep neural network computing." In: *ACM SIGARCH Computer Architecture News* 44.3, pp. 1–13.

Ali, Alnur, Rich Caruana, and Ashish Kapoor (2014). "Active learning with model selection." In: *Proceedings of the AAAI conference on artificial intelligence* 28.1, pp. 1673–1679.

Alina, Beygelzimer, Hsu Daniel, Karampatziakis Nikos, Langford John, and Zhang Tong (2011). "Efficient active learning." In: *ICML Workshop on On-line Trading of Exploration and Exploitation*.

Alina, Beygelzimer, Dasgupta Sanjoy, and John Langford (2008). "Importance weighted active learning." In: *ACM International Conference Proceeding Series* 382.

Alistarh, Dan, Demjan Grubic, Jerry Li, Ryota Tomioka, and Milan Vojnovic (2017). "QSGD: communication-efficient SGD via gradient quantization and encoding." In: *Advances in Neural Information Processing Systems 30*, pp. 1709–1720.

Altschuler, Jason and Kunal Talwar (2018). "Online learning over a finite action set with limited switching." In: *Conference On Learning Theory*, pp. 1569–1573.

Athalye, Anish, Nicholas Carlini, and David Wagner (2018). "Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples." In: *International Conference on Machine Learning*, pp. 274–283.

Auer, Peter, Nicolo Cesa-Bianchi, and Paul Fischer (2002). "Finite-time analysis of the multiarmed bandit problem." In: *Machine learning* 47.2-3, pp. 235–256.

Azuma, Kazuoki (1967). "Weighted sums of certain dependent random variables." In: *Tohoku Mathematical Journal, Second Series* 19.3, pp. 357–367.

Bach, Francis R (2007). "Active learning for misspecified generalized linear M=models." In: *Advances in Neural Information Processing Systems 19*, pp. 65–72.

Balcan, Maria Florina, Alina Beygelzimer, and John Langford (2009). "Agnostic active learning." In: *Journal of Computer and System Sciences* 75.1, pp. 78–89.

Balunovic, Mislav and Martin Vechev (2020). "Adversarial training and provable defenses: bridging the gap." In: *International Conference on Learning Representations*.

Baraniuk, Richard, Mark Davenport, Ronald DeVore, and Michael Wakin (2008). "A simple proof of the Restricted Isometry Property for random matrices." In: *Constructive Approximation* 28, pp. 253–263.

Beck, Daniel, Gholamreza Haffari, and Trevor Cohn (2018). "Graph-to-sequence learning using gated graph neural networks." In: *arXiv preprint arXiv:1806.09835*.

Belghazi, Mohamed Ishmael, Aristide Baratin, Sai Rajeshwar, Sherjil Ozair, Yoshua Bengio, Aaron Courville, and Devon Hjelm (2018). "Mutual information neural estimation." In: *Proceedings of the 35th International Conference on Machine Learning*. Proceedings of Machine Learning Research 80, pp. 531–540.

Berinde, Radu and Piotr Indyk (2010). "Sparse recovery using sparse random matrices." In: *LATIN 2010: Theoretical Informatics*. Ed. by Alejandro Lopez-Ortiz, pp. 157–157.

Beygelzimer, Alina, Daniel J Hsu, John Langford, and Tong Zhang (2010). "Agnostic active learning without constraints." In: *Advances in neural information processing systems* 23.

Bhattad, Anand, Min Jin Chong, Kaizhao Liang, Bo Li, and David A Forsyth (2020). "Unrestricted adversarial examples via semantic manipulation." In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=Sye_OgHFwH.

Biba, Marenglen, Stefano Ferilli, and Floriana Esposito (2011). "Protein fold recognition using markov logic networks." In: Springer, pp. 69–85.

Bielik, Pavol, Peter Tsankov, Andreas Krause, and Martin Vechev (2020). "Reliability assessment of traffic sign classifiers." In: *Federal Office for Information Security, Jul*.

Blanchard, Jeffrey D, Jared Tanner, and Ke Wei (2013). "Conjugate gradient iterative hard thresholding: observed noise dtability for compressed censing." In: *Oxford Numerical Analysis Group Preprint*.

Blumensath, Thomas (2012). "Accelerated iterative hard thresholding." In: *The Journal of Fourier Analysis and Applications* 92.3, pp. 265–274.

– (2013). "Compressed sensing with nonlinear observations and related nonlinear optimization problems." In: *IEEE Transactions of Information Theory* 59.6, pp. 3466–3474.

Blumensath, Thomas and Michael E Davies (2008). "Iterative thresholding for sparse approximations." In: *The Journal of Fourier Analysis and Applications* 14.5–6, pp. 629–654.

– (2009). "Iterative thresholding for compressed sensing." In: *Applied and Computational Harmonic Analysis* 27.3, pp. 265–274.

– (2010). "Normalized iterative hard thresholding: guaranteed stability and performance." In: *IEEE Selected Topics in Signal Processing* 4.2, pp. 298–309.

Blumensath, Thomas, Michael E Davies, Gabriel Rilling, YC Eldar, and G Kutyniok (2012). "Greedy algorithms for compressed sensing." In: *Compressed Sensing: Theory and Applications*. Ed. by Y. C. Eldar and G. Kutyniok, pp. 348–393.

Boucheron, Stéphane, Gábor Lugosi, and Pascal Massart (2013). *Concentration inequalities: A nonasymptotic theory of independence*. Oxford university press.

Boufounos, Petros T and Richard G Baraniuk (2008). "1-Bit compressive sensing." In: *42nd Annual Conference on Information Sciences and Systems*, pp. 16–21.

Breiman, Leo (1996). "Bagging predictors." In: *Machine learning*, pp. 123–140.

Bruna, Joan, Wojciech Zaremba, Arthur Szlam, and Yann LeCun (2013). "Spectral networks and locally connected networks on graphs." In: *arXiv preprint arXiv:1312.6203*.

Cai, Qi-Zhi Cai, Chang Liu, and Dawn Song (July 2018). "Curriculum adversarial training." In: *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pp. 3740–3747. DOI: 10.24963/ijcai.2018/520. URL: https://doi.org/10.24963/ijcai.2018/520.

Candes, Emmanuel J, Justin K Romberg, and Terence Tao (2006a). "Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information." In: *IEEE Transactions of Information Theory* 52.2, pp. 489–509.

– (2006b). "Stable signal recovery from incomplete and inaccurate measurements." In: *Communications on Pure and Applied Mathematics* 59.8, pp. 1207–1223.

Candes, Emmanuel (2008). "The restricted isometry property and its implications for compressed sensing." In: *Comptes Rendus Mathematique* 346.9–10, pp. 589–592.

Carlini, Nicholas and David Wagner (2017a). "Adversarial examples are not easily detected: Bypassing ten detection methods." In: *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, pp. 3–14.

– (2017b). "Towards evaluating the robustness of neural networks." In: *IEEE Symposium on Security and Privacy (SP)*, pp. 39–57.

Carmon, Yair, Aditi Raghunathan, Ludwig Schmidt, John C Duchi, and Percy S Liang (2019). "Unlabeled data improves adversarial robustness." In: *Advances in Neural Information Processing Systems*, pp. 11190–11201.

Cesa-Bianchi, Nicolo, Gábor Lugosi, and Gilles Stoltz (2005). "Minimizing regret with label efficient prediction." In: *IEEE Transactions on Information Theory* 51.6, pp. 2152–2162.

Cevher, Volkan (2011). "On sccelerated hard thresholding methods for sparse approximation." In: *Wavelets And Sparsity Xiv* 8138.

Chakrabarti, Deepayan, Stanislav Funiak, Jonathan Chang, and Sofus Macskassy (2014). "Joint inference of multiple label types in large networks." In: *International Conference on Machine Learning*, pp. 874–882.

Chang, Ting-Jui, Yukun He, and Peng Li (2018). "Efficient two-step adversarial defense for deep neural networks." In: *arXiv preprint arXiv:1810.03739*.

Chartrand, Rick and Valentina Staneva (2008). "Restricted isometry properties and nonconvex compressive sensing." In: *Inverse Problems* 24.

Chen, Liwei, Yansong Feng, Jinghui Mo, Songfang Huang, and Dongyan Zhao (2014). "Joint inference for knowledge base population." In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1912–1923.

Chen, Yu-Hsin, Joel Emer, and Vivienne Sze (2016). "Eyeriss: A spatial architecture for energy-efficient dataflow for convolutional neural networks." In: *ACM SIGARCH computer architecture news* 44.3, pp. 367–379.

Chen, Yuxin, S Hamed Hassani, Amin Karbasi, and Andreas Krause (2015). "Sequential information maximization: When is greedy near-optimal?" In: *Conference on Learning Theory*, pp. 338–363.

Cohen, Jeremy, Elan Rosenfeld, and Zico Kolter (2019). "Certified adversarial robustness via randomized smoothing." In: *International Conference on Machine Learning*, pp. 1310–1320.

Cohn, David, Les Atlas, and Richard Ladner (1994). "Improving generalization with active learning." In: *Machine Learning* 15.2, pp. 201–221.

Cong, Jason and Bingjun Xiao (2014). "Minimizing computation in convolutional neural networks." In: *International conference on artificial neural networks*, pp. 281–290.

Corda, Stefano, Bram Veenboer, Ahsan Javed Awan, John W Romein, Roel Jordans, Akash Kumar, Albert-Jan Boonstra, and Henk Corporaal (2022). "Reduced-precision acceleration of radio-astronomical imaging on reconfigurable hardware." In: *IEEE Access* 10, pp. 22819–22843.

Courbariaux, Matthieu, Yoshua Bengio, and Jean-Pierre David (2015). "Binaryconnect: training deep neural networks with binary weights during propagations." In: *Advances in neural information processing systems* 28.

Dasgupta, Sanjoy, Daniel J Hsu, and Claire Monteleoni (2008). "A general agnostic active learning algorithm." In: *Advances in Neural Information Processing Systems 20*, pp. 353–360.

Davenport, M. A., Y. Plan, E. can den Berg, and M. Wooters (2012). "1-Bit matrix completion." In: *Information and Inference* 3.

De Sa, Christopher M, Ce Zhang, Kunle Olukotun, and Christopher Ré (2015). "Taming the wild: A unified analysis of Hogwild!-style algorihms." In: *Advances in Neural Information Processing Systems*, pp. 2674–2682.

Defferrard, Michaël, Xavier Bresson, and Pierre Vandergheynst (2016). "Convolutional neural networks on graphs with fast localized spectral filtering." In: *Advances in neural information processing systems*, pp. 3844–3852.

Deng, Jia, Nan Ding, Yangqing Jia, Andrea Frome, Kevin Murphy, Samy Bengio, Yuan Li, Hartmut Neven, and Hartwig Adam (2014). "Large-scale object classification using label relation graphs." In: *European conference on computer vision*, pp. 48–64.

Dolatabadi, Hadi M, Sarah Erfani, and Christopher Leckie (2021). "*ell*_∞-robustness and beyond: unleashing efficient adversarial training." In: *arXiv preprint arXiv:2112.00378*.

Donoho, David L (2006). "Compressed sensing." In: *IEEE Transactions of Information Theory* 52.4, pp. 1289–1306.

Dubout, Charles and François Fleuret (2012). "Exact acceleration of linear object detectors." In: *European Conference on Computer Vision*, pp. 301–311.

Dvijotham, Krishnamurthy Dj, Jamie Hayes, Borja Balle, Zico Kolter, Chongli Qin, Andras Gyorgy, Kai Xiao, Sven Gowal, and Pushmeet Kohli (2020). "A framework for robustness certification of smoothed classifiers using f-divergences." In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=SJlKrkSFPH.

Edelmann, Geoffrey F and Charles F Gaumond (2011). "Beamforming using compressive sensing." In: *The Journal of the Acoustical Society of America* 130.4, pp. 232–237.

Eldar, Yonina C and Gitta Kutyniok, eds. (2012). *Compressed sensing: theory and applications*. Cambridge University Press.

Eykholt, Kevin, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno, and Dawn Song (2018). "Robust physical-world attacks on deep learning visual classification." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1625–1634.

Freund, Yoav and Robert E Schapire (1995). "A desicion-theoretic generalization of online learning and an application to boosting." In: *European conference on computational learning theory*, pp. 23–37.

Gardner, Jacob, Gustavo Malkomes, Roman Garnett, Kilian Q Weinberger, Dennis Barbour, and John P Cunningham (2015). "Bayesian active model Selection with an application to automated audiometry." In: *Advances in Neural Information Processing Systems 28*, pp. 2386–2394.

Geer, Sara A van de (2002). "On Hoeffding's inequality for dependent random variables." In: Springer, pp. 161–169.

Gehr, Timon, Matthew Mirman, Dana Drachsler-Cohen, Petar Tsankov, Swarat Chaudhuri, and Martin Vechev (2018). "AI2: safety and robustness certification of neural networks with abstract interpretation." In: *IEEE Symposium on Security and Privacy (SP)*, pp. 3–18.

Goodfellow, Ian J, Yaroslav Bulatov, Julian Ibarz, Sacha Arnoud, and Vinay Shet (2013). "Multi-digit number recognition from street view imagery using deep convolutional neural networks." In: *arXiv preprint arXiv:1312.6082*.

Goodfellow, Ian J., Jonathon Shlens, and Christian Szegedy (2015). "Explaining and harnessing adversarial examples." In: *International Conference on Learning Representations*.

Gopi, Sivakant, Praneeth Netrapalli, Prateek Jain, and Aditya Nori (2013). "One-bit compressed sensing: provable support and vector recover." In: *Proceedings of the 30th International Conference on Machine Learning* 28.3, pp. 154–162.

Gowal, Sven, Krishnamurthy Dvijotham, Robert Stanforth, Rudy Bunel, Chongli Qin, Jonathan Uesato, Relja Arandjelovic, Timothy Mann, and Pushmeet Kohli (2018). "On the effectiveness of interval bound propagation for training verifiably robust models." In: *arXiv preprint arXiv:1810.12715*.

Gupta, Ankit, Robert Nowak, and Benjamin Recht (2010). "Sample complexity for 1-bit compressed sensing and sparse classification." In: *IEEE International Symposium on Information Theory*, pp. 1553–1557.

Gupta, Suyog, Ankur Agrawal, Kailash Gopalakrishnan, and Pritish Narayanan (2013). "Deep learning with limited numerical precision." In: *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, pp. 1737–1746.

Gurbuz, Ali Cafer, James H McClellan, and Volkan Cevher (2008). "A compressive beamforming method." In: *2008 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 2617–2620.

Han, Song, Xingyu Liu, Huizi Mao, Jing Pu, Ardavan Pedram, Mark A Horowitz, and William J Dally (2016). "EIE: Efficient inference engine on compressed deep neural network." In: *ACM SIGARCH Computer Architecture News* 44.3, pp. 243–254.

Han, Song, Huizi Mao, and William J Dally (2015). "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding." In: *arXiv preprint arXiv:1510.00149*.

Han, Song, Jeff Pool, John Tran, and William Dally (2015). "Learning both weights and connections for efficient neural network." In: *Advances in neural information processing systems*, pp. 1135–1143.

Han, Yubing and Jian Wang (2015). "Adaptive beamforming based on compressed sensing with smoothed norm." In: *International Journal of Antennas and Propagation* 2015.

Hendrycks, Dan and Thomas Dietterich (2019). "Benchmarking neural network robustness to common corruptions and perturbations." In: *International Conference on Learning Representations*.

Hinton, Geoffrey, Oriol Vinyals, Jeff Dean, et al. (2015). "Distilling the knowledge in a neural network." In: *arXiv preprint arXiv:1503.02531* 2.7.

Högbom, Jan A (1974). "Aperture synthesis with a non-regular distribution of interferometer baselines." In: *Astronomy and Astrophysics Supplement* 15, p. 417.

Hubara, I., M. Courbariaux, D. Soudry, R. Al-Yaniv, and Y. Bengio (2017). "Quantized neural networks: training neural networks with low precision weights and activations." In: *The Journal of Machine Learning Research* 18.1, pp. 6869–6898.

Ido, Dagan and Engelson Sean (1995). "Committee-based sampling for training probabilistic classifiers." In: *Machine Learning Proceedings 1995*, pp. 150–157.

Ilyas, Andrew, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry (2019). "Adversarial examples are not bugs, they are features." In: *Advances in Neural Information Processing Systems*, pp. 125–136.

Izhikevich, Eugene M (2004). "Which model to use for cortical spiking neurons?" In: *IEEE transactions on neural networks* 15.5, pp. 1063–1070.

Jacques, Laurent, Jason N Laska, Petros T Boufounos, and Richard G Baraniuk (2013). "Robust 1-bit compressive sensing via binary stable embeddings of sparse vectors." In: *IEEE Transactions on Information Theory* 59.4, pp. 2082–2102.

Janocha, Katarzyna and Wojciech Marian Czarnecki (2017). "On loss functions for deep neural networks in classification." In: *arXiv preprint arXiv:1702.05659*.

Javanmard, Adel, Mahdi Soltanolkotabi, and Hamed Hassani (2020). "Precise tradeoffs in adversarial training for linear regression." In: *Conference on Learning Theory*, pp. 2034–2078.

Kaiming, He, Zhang Xiangyu, Ren Shaoqing, and Sun Jian (2016). "Deep residual learning for image recognition." In: *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778.

Kang, Daniel, Yi Sun, Dan Hendrycks, Tom Brown, and Jacob Steinhardt (2019). "Testing robustness against unforeseen adversaries." In: *arXiv preprint arXiv:1908.08016*.

Kara, Kaan, Dan Alistarh, Gustavo Alonso, Onur Mutlu, and Ce Zhang (2017). "FPGA-accelerated dense linear machine learning: A precision-convergence trade-off." In: *IEEE 25th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM)*, pp. 160–167.

Karimi, Mohammad Reza, Andreas Krause, Silvio Lattanzi, and Sergei Vassilvtiskii (2019). "Consistent online optimization: Convex and submodular." In: *Proceedings of Machine Learning Research* 89, pp. 2241–2250.

Katariya, Namit, Arun Iyer, and Sunita Sarawagi (2012). "Active evaluation of classifiers on large datasets." In: *2012 IEEE 12th International Conference on Data Mining*, pp. 329–338.

Kipf, Thomas N and Max Welling (2016a). "Semi-supervised classification with graph convolutional networks." In: *arXiv preprint arXiv:1609.02907*.

– (2016b). "Semi-supervised classification with graph convolutional networks." In: *arXiv preprint arXiv:1609.02907*.

Kipf, Thomas N and Max Welling (2017). “Semi-supervised classification with graph convolutional networks.” In: *ICLR*. URL: https://openreview.net/forum?id=SJU4ayYgl.

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E Hinton (2012). “Imagenet classification with deep convolutional neural networks.” In: *Advances in neural information processing systems*, pp. 1097–1105.

Kumar, Anurag and Bhiksha Raj (2018). “Classifier risk estimation under limited labeling resources.” In: *Advances in Knowledge Discovery and Data Mining*, pp. 3–15.

Langer, Steve G (2011). “Challenges for data storage in medical imaging research.” In: *Journal of Digital Imaging* 24.2, pp. 203–207.

Laska, Jason N, Zaiwen Wen, Wotao Yin, and Richard G Baraniuk (2011). “Fast and accurate signal recovery from 1-bit compressive measurements.” In: *IEEE Transactions on Signal Processing* 59.11, pp. 5289–5301.

Lavin, Andrew and Scott Gray (2016). “Fast algorithms for convolutional neural networks.” In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4013–4021.

LeCun, Yann, John Denker, and Sara Solla (1989). “Optimal brain damage.” In: *Advances in neural information processing systems* 2.

Lee, JunKyu, Lev Mukhanov, Amir Sabbagh Molahosseini, Umar Minhas, Yang Hua, Jesus Martinez del Rincon, Kiril Dichev, Cheol-Ho Hong, and Hans Vandierendonck (2021). “Resource-efficient deep learning: a survey on model-, arithmetic-, and implementation-level techniques.” In: *arXiv preprint arXiv:2112.15131*.

Leite, Rui and Pavel Brazdil (2010). “Active testing strategy to predict the best classification algorithm via sampling and metalearning.” In: *Proceedings of the 2010 Conference on ECAI 2010: 19th European Conference on Artificial Intelligence*, pp. 309–314.

Li, Bai, Changyou Chen, Wenlin Wang, and Lawrence Carin (2019a). “Certified adversarial robustness with additive noise.” In: *Advances in Neural Information Processing Systems*, pp. 9464–9474.

Li, Feng, Tim J Cornwell, and Frank de Hoog (2011). “The application of compressive sampling to radio astronomy 1: deconvolution.” In: *Astronomy and Astrophysics Manuscript* 15, p. 417.

Li, Guohao, Matthias Müller, Ali Thabet, and Bernard Ghanem (2019b). "Can GCNs go as deep as CNNs?" In: *arXiv preprint arXiv:1904.03751*.

Li, Qimai, Zhichao Han, and Xiao-Ming Wu (2018). "Deeper insights into graph convolutional networks for semi-supervised learning." In: *Thirty-Second AAAI Conference on Artificial Intelligence*.

Li, Yaguang, Rose Yu, Cyrus Shahabi, and Yan Liu (2017). "Graph convolutional recurrent neural network: Data-driven traffic forecasting." In: *arXiv preprint arXiv:1707.01926*.

LiKamWa, Robert, Yunhui Hou, Julian Gao, Mia Polansky, and Lin Zhong (2016). "Redeye: analog convnet image sensor architecture for continuous mobile vision." In: *ACM SIGARCH Computer Architecture News* 44.3, pp. 255–266.

Littlestone, Nick and Manfred K Warmuth (1994). "The weighted majority algorithm." In: *Information and Computation*, pp. 212 –261.

Liu, Bo, Xiao-Tong Yuan, Lezi Wang, Qingshan Liu, and Dimitris N Metaxas (2017). "Dual iterative hard thresholding: From non-convex sparse minimization to non-smooth concave maximization." In: *Proceedings of the 34th International Conference on Machine Learning*, pp. 2179–2187.

Liu, Ji and Ce Zhang (2020). "Distributed learning systems with first-order methods." In: *Databases* 9.1, pp. 1–100.

Liu, Xiaogeng, Haoyu Wang, Yechao Zhang, Fangzhou Wu, and Shengshan Hu (2022). "Towards efficient data-centric robust machine learning with noise-based augmentation." In: *arXiv preprint arXiv:2203.03810*.

Lustig, Michael, David Donoho, and John M Pauly (2007). "Sparse MRI: The application of compressed sensing for rapid MR imaging." In: *Magnetic resonance in medicine* 58 6, pp. 1182–95.

– (2008). "Compressed sensing MRI." In: *IEEE Signal Processing Magazine* 25, pp. 72–82.

Madani, Omid, Daniel Lizotte, and Russell Greiner (2012). "Active model selection." In: *Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence*, pp. 357–365.

Madry, Aleksander, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu (2017). "Towards deep learning models resistant to adversarial attacks." In: *arXiv preprint arXiv:1706.06083*.

Mathieu, Michael, Mikael Henaff, and Yann LeCun (2014). "Fast training of convolutional networks through FFTs." In: *ICLR*.

Mattmann, Chris A, Andrew Hart, Luca Cinquini, Joseph Lazio, Shakeh Khudikyan, Dayton Jones, Robert Preston, Thomas Bennett, Bryan Butler, David Harland, et al. (2014). "Scalable data mining, archiving, and big data management for the next generation astronomical telescopes." In: *Big Data Management, Technologies, and Applications*.

McCallum, Andrew (2009). "Joint inference for natural language processing." In: *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, pp. 1–1.

McCallum, Andrew and Kamal Nigam (1998). "Employing EM and pool-based active learning for text classification." In: *Proceedings of the Fifteenth International Conference on Machine Learning*, pp. 350–358.

Melacci, Stefano, Gabriele Ciravegna, Angelo Sotgiu, Ambra Demontis, Battista Biggio, Marco Gori, and Fabio Roli (2021). "Domain knowledge alleviates adversarial attacks in multi-Label classifiers." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Melville, Prem and Raymond J. Mooney (2004a). "Diverse ensembles for active learning." In: *Proceedings of the Twenty-First International Conference on Machine Learning*, p. 74.

Melville, Prem and Raymond J Mooney (2004b). "Diverse ensembles for active learning." In: *Proceedings of the twenty-first international conference on Machine learning*, p. 74.

Menghani, Gaurav (2021). "Efficient deep learning: A survey on making deep learning models smaller, faster, and better." In: *arXiv preprint arXiv:2106.08962*.

Mirman, Matthew, Timon Gehr, and Martin Vechev (2018). "Differentiable abstract interpretation for provably robust neural networks." In: *International Conference on Machine Learning*, pp. 3578–3586.

Mirman, Matthew, Alexander Hägele, Timon Gehr, Pavol Bielik, and Martin Vechev (2021). "Robustness certification with generative models." In: *Proceedings of the 42nd ACM SIGPLAN Conference on Programming Language Design and Implementation*.

Miyashita, Daisuke and Lee, Edward H and Murmann, Boris (2016). “Convolutional neural networks using logarithmic data representation.” In: *arXiv:1603.01025*.

Miyashita, Daisuke, Edward H Lee, and Borisu Murmann (2016). “Dorefa-net: training low bitwidth convolutional neural networks with low bitwidth gradients.” In: *arXiv:1606.06160*.

Mogelmose, Andreas, Mohan Manubhai Trivedi, and Thomas B Moeslund (2012). “Vision-based traffic sign detection and analysis for intelligent driver assistance systems: Perspectives and survey.” In: *IEEE Transactions on Intelligent Transportation Systems* 13.4, pp. 1484–1497.

Mohammad Reza Karimi, Nezihe Merve Gürel[†], Bojan Karlaš, Johannes Rausch, Ce Zhang, and Andreas Krause (2021). “Online active model selection for pre-trained classifiers.” In: *International Conference on Artificial Intelligence and Statistics*, pp. 307–315.

Mohapatra, Jeet, Ching-Yun Ko, Sijia Liu, Pin-Yu Chen, Luca Daniel, et al. (2020). “Rethinking randomized smoothing for adversarial robustness.” In: *arXiv preprint arXiv:2003.01249*.

Murmann, Boris, Daniel Bankman, Elaina Chai, Daisuke Miyashita, and Lita Yang (2015). “Mixed-signal circuits for embedded machine-learning applications.” In: *2015 49th Asilomar conference on signals, systems and computers*, pp. 1341–1345.

Nayak, Gaurav Kumar, Ruchit Rawal, and Anirban Chakraborty (2022). “DAD: data-free adversarial defense at test time.” In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 3562–3571.

Needell, Deanna and Joel A Tropp (2008). “CoSaMP: Iterative signal recovery from incomplete and inaccurate samples.” In: *Applied and Computational Harmonic Analysis* 26.3, pp. 301–321.

Neuberg, Leland Gerson (2003). *Causality: models, reasoning, and inference*.

Nezihe Merve Gürel, Kaan Kara, Alen Stojanov, Tyler Smith, Thomas Lemmin, Dan Alistarh, Markus Püschel, and Ce Zhang (2020). “Compressive Sensing Using Iterative Hard Thresholding with Low Precision Data Representation: Theory and Applications.” In: *IEEE Transactions on Signal Processing* 68, pp. 4268–4282.

Noel, Cyprien and Simon Osindero (2014). "Dogwild!-distributed hogwild for CPU & GPU." In: *NIPS Workshop on Distributed Machine Learning and Matrix Computations*, pp. 693–701.

Ocal, Orhan, Paul Hurley, Giovanni Cherubini, and Sanaz Kazemi (2015). "Collaborative randomized beamforming for phased array radio interferometers." In: *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5654–5658.

Oono, Kenta and Taiji Suzuki (2019). "On asymptotic behaviors of graph CNNs from dynamical systems perspective." In: *arXiv preprint arXiv:1905.10947*.

Pernkopf, Franz, Wolfgang Roth, Matthias Zoehrer, Lukas Pfeifenberger, Guenther Schindler, Holger Froening, Sebastian Tschiatschek, Robert Peharz, Matthew Mattina, and Zoubin Ghahramani (2018). "Efficient and robust machine learning for real-world systems." In: *arXiv preprint arXiv:1812.02240*.

Plan, Yaniv and Roman Vershynin (2013a). "One-bit compressed sensing by linear programming." In: *Communications on Pure and Applied Mathematics* 66.8, pp. 1275–1297.

– (2013b). "Robust 1-bit compressed sensing and sparse logistic regression: a convex programming approach." In: *IEEE Transactions on Information Theory* 59.1, pp. 482–494.

Poldrack, Russell A, Jeanette A Mumford, and Thomas E Nichols (2011). "Handbook of functional MRI data analysis." In:

Poon, Hoifung and Pedro Domingos (2007). "Joint inference in information extraction." In: *AAAI* 7, pp. 913–918.

Raghunathan, Aditi, Sang Michael Xie, Fanny Yang, John Duchi, and Percy Liang (2020). "Understanding and mitigating the tradeoff between robustness and accuracy." In: *arXiv preprint arXiv:2002.10716*.

Rastegari, M., V. Ordonez, J. Redmon, and A. Farhadi (2016). "Xnor-net: Imagenet classification using binary convolutional neural networks." In: *European Conference on Computer Vision*, pp. 525–542.

Ratner, Alexander, Stephen H Bach, Henry Ehrenberg, Jason Fries, Sen Wu, and Christopher Ré (2017). "Snorkel: Rapid training data creation with weak supervision." In: *Proceedings of the VLDB Endowment. International Conference on Very Large Data Bases* 11.3, p. 269.

Ratner, Alexander, Christopher De Sa, Sen Wu, Daniel Selsam, and Christopher Ré (2016). "Data programming: Creating large training sets, quickly." In: *Advances in neural information processing systems* 29, p. 3567.

Richardson, Matthew and Pedro Domingos (2006). "Markov logic networks." In: *Machine learning* 62.1-2, pp. 107–136.

Rong, Yu, Wenbing Huang, Tingyang Xu, and Junzhou Huang (2019). "DropEdge: towards deep graph convolutional networks on node classification." In: *International Conference on Learning Representations*.

Ross, Andrew and Finale Doshi-Velez (2018). "Improving the adversarial robustness and interpretability of deep neural networks by regularizing their input gradients." In: *Proceedings of the AAAI Conference on Artificial Intelligence* 32.1.

Rother, Carsten, Vladimir Kolmogorov, and Andrew Blake (2004). "" GrabCut" interactive foreground extraction using iterated graph cuts." In: *ACM transactions on graphics (TOG)* 23.3, pp. 309–314.

Salman, Hadi, Jerry Li, Ilya Razenshteyn, Pengchuan Zhang, Huan Zhang, Sebastien Bubeck, and Greg Yang (2019). "Provably robust deep learning via adversarially trained smoothed classifiers." In: *Advances in Neural Information Processing Systems*, pp. 11292–11303.

Samangouei, Pouya, Maya Kabkab, and Rama Chellappa (2018). "Defense-gan: Protecting classifiers against adversarial attacks using generative models." In: *arXiv preprint arXiv:1805.06605*.

Sandler, Mark, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen (2018). "Mobilenetv2: Inverted residuals and linear bottlenecks." In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4510–4520.

Sawade, Christoph, Niels Landwehr, Steffen Bickel, and Tobias Scheffer (2010). "Active risk estimation." In: *Proceedings of the 27th International Conference on International Conference on Machine Learning*, pp. 951–958.

Sawade, Christoph, Niels Landwehr, and Tobias Scheffer (2012). "Active comparison of prediction models." In: *Advances in Neural Information Processing Systems* 25.

Saxe, Andrew M, Yamini Bansal, Joel Dapello, Madhu Advani, Artemy Kolchinsky, Brendan D Tracey, and David D Cox (2019). "On the information bottleneck theory of deep learning." In: *Journal of Statistical Mechanics: Theory and Experiment* 2019.12, p. 124020.

Schott, Lukas, Jonas Rauber, Matthias Bethge, and Wieland Brendel (2018). "Towards the first adversarially robust neural network model on MNIST." In: *International Conference on Learning Representations*.

Seide, Frank, Hao Fu, Jasha Droppo, Gang Li, and Dong Yu (2014). "1-bit stochastic gradient descent and application to data-parallel distributed training of speech DNNs." In: *Interspeech*, pp. 1058–1062.

Semeval-EmoContext (2019). URL: https://www.humanizing-ai.com/emocontext.html.

Sermanet, Pierre and Yann LeCun (2011). "Traffic sign recognition with multi-scale convolutional networks." In: *The 2011 International Joint Conference on Neural Networks*, pp. 2809–2813.

Settles, Burr (2009). "Active learning literature survey." In:

Settles, Burr and Mark Craven (2008a). "An analysis of active learning strategies for sequence labeling tasks." In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 1070–1079.

– (2008b). "An analysis of active learning strategies for sequence labeling tasks." In: *proceedings of the 2008 conference on empirical methods in natural language processing*, pp. 1070–1079.

Seung, Sebastian, Manfred Opper, and Haim Sompolinsky (1992). "Query by committee." In: *Proceedings of the fifth annual workshop on Computational learning theory*, pp. 287–294.

Shamir, Ohad, Sivan Sabato, and Naftali Tishby (2010). "Learning and generalization with the information bottleneck." In: *Theoretical Computer Science* 411.29-30.

Simeoni, Matthieu Martin Jean-Andre (2015). "Towards more accurate and efficient beamformed radio interferometry imaging." In:

Singh, Gagandeep, Timon Gehr, Matthew Mirman, Markus Püschel, and Martin Vechev (2018a). "Fast and effective robustness certification." In: *Advances in neural information processing systems* 31.

Singh, Gagandeep, Timon Gehr, Markus Püschel, and Martin Vechev (2018b). "Boosting robustness certification of neural networks." In: *International conference on learning representations*.

Sriramanan, Gaurang, Sravanti Addepalli, Arya Baburaj, et al. (2021). "Towards efficient and effective adversarial training." In: *Advances in Neural Information Processing Systems* 34.

Stallkamp, Johannes, Marc Schlipsing, Jan Salmen, and Christian Igel (2012). "Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition." In: *Neural networks* 32, pp. 323–332.

Stewart, Gilbert W (1990). "Perturbation theory for the singular value decomposition." In: *SVD and Signal Processing II: Algorithms, Analysis and Applications*.

Stewart, M. (2006). "Perturbation of the SVD in the presence of small singular values." In: *Linear Algebra and its Applications* 419.1, pp. 53 –77.

Stojanov, Alen, Taylor M Smith, Dan Alistarh, and Markus Püschel (2018). "Fast quantized arithmetic on x86: Trading compute for data movement." In: *IEEE International Workshop on Signal Processing Systems (SiPS)*, pp. 349–354.

Such, Felipe Petroski, Shagan Sah, Miguel Alexander Dominguez, Suhas Pillai, Chao Zhang, Andrew Michael, Nathan D Cahill, and Raymond Ptucha (2017). "Robust spatial filtering with graph convolutional neural networks." In: *IEEE Journal of Selected Topics in Signal Processing* 11.6, pp. 884–896.

Sugiyama, Masashi (2006). "Active learning for misspecified models." In: *Advances in Neural Information Processing Systems 18*, pp. 1305–1312.

Sun, Mingjie, Zichao Li, Chaowei Xiao, Haonan Qiu, Bhavya Kailkhura, Mingyan Liu, and Bo Li (2021). "Can shape structure features improve model robustness under diverse adversarial settings?" In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 7526–7535.

Sze, Vivienne, Yu-Hsin Chen, Joel Emer, Amr Suleiman, and Zhengdong Zhang (2017a). "Hardware for machine learning: Challenges and opportunities." In: *2017 IEEE Custom Integrated Circuits Conference (CICC)*, pp. 1–8.

Sze, Vivienne, Yu-Hsin Chen, Tien-Ju Yang, and Joel S Emer (2017b). "Efficient processing of deep neural networks: A tutorial and survey." In: *Proceedings of the IEEE* 105.12, pp. 2295–2329.

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

Taylor, G. B. and C. L. Carilli (1999). "Synthesis imaging in radio astronomy ii." In: *In ASP Conf. Series*.

Taylor, Greg B, Chris Luke Carilli, and Richard A Perley (1999). "Synthesis imaging in radio astronomy II." In: *A Collection of Lectures from the Sixth NRAO-NMIMT Synthesis Imaging Summer School, ASP Conf. Series* 180.

Telatar, Emre (1999). "Capacity of multi-antenna Gaussian channels." In: *European transactions on telecommunications* 10 (6), pp. 585–595.

Tishby, Naftali, Fernando C Pereira, and William Bialek (2000). "The information bottleneck method." In: *arXiv preprint physics/0004057*.

Tosh, Christopher and Sanjoy Dasgupta (2018). "Interactive structure learning with structural query-by-committee." In: *Advances in Neural Information Processing Systems 31*, pp. 1121–1131.

Tsipras, Dimitris, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry (2019). "Robustness may be at odds with accuracy." In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=SyxAb30cY7.

Vaccaro, R. and A. Kot (1987). "A perturbation theory for the analysis of SVD-based algorithms." In: *IEEE International Conference on Acoustics, Speech, and Signal Processing* 12, pp. 1613–1616.

Vanhoucke, Vincent, Andrew Senior, and Mark Z Mao (2011). "Improving the speed of neural networks on CPUs." In:

Vergara, Alexander (2012). *UCI Machine Learning Repository*. URL: http://archive.ics.uci.edu/ml/datasets/Gas+Sensor+Array+Drift+Dataset.

Vergara, Alexander, Shankar Vembu, Tuba Ayhan, Margaret A Ryan, Margie L Homer, and Ramón Huerta (2012). "Chemical gas sensor drift compensation using classifier ensembles." In: *Sensors and Actuators B: Chemical* 166, pp. 320–329.

Wainwright, Martin J and Michael Irwin Jordan (2008). *Graphical models, exponential families, and variational inference*. Now Publishers Inc.

Wang, Zhuo, Robert Schapire, and Naveen Verma (2014). "Error-adaptive classifier boosting (EACB): Exploiting data-driven training for highly fault-tolerant hardware." In: *IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 3884–3888.

Wei, K. (2015). “Fast iterative hard thresholding for compressed sensing.” In: *IEEE Signal Processing Letters* 22.5.

Wenger, S., S. Darabi, P. Sen, K. H. Glassmeier, and M. Magnor (2010). “Compressed sensing for aperture synthesis imaging.” In: *IEEE International Conference on Image Processing*, pp. 1381–1384.

Wiaux, Yves, Laurent Jacques, Gilles Puy, Anna MM Scaife, and Pierre Vandergheynst (May 2009). “Compressed sensing imaging techniques for radio interferometry.” In: *Monthly Notices of the Royal Astronomical Society* 395.3, pp. 1733–1742.

Wong, Eric, Leslie Rice, and J. Zico Kolter (2020). “Fast is better than free: Revisiting adversarial training.” In: *International Conference on Learning Representations*.

Wong, Eric, Frank R Schmidt, and J Zico Kolter (2019). “Wasserstein adversarial examples via projected sinkhorn iterations.” In: *arXiv preprint arXiv:1902.07906*.

Wong, Sebastien C., Adam Gatt, Victor Stamatescu, and Mark D. McDonnell (2016). “Understanding data augmentation for dlassification: when to warp?” In: *2016 International Conference on Digital Image Computing: Techniques and Applications (DICTA)*, pp. 1–6.

Wu, Felix, Tianyi Zhang, Amauri Holanda de Souza Jr, Christopher Fifty, Tao Yu, and Kilian Q Weinberger (2019a). “Simplifying graph convolutional networks.” In: *arXiv preprint arXiv:1902.07153*.

Wu, Tong, Liang Tong, and Yevgeniy Vorobeychik (2019). “Defending against physically realizable attacks on image classification.” In: *arXiv preprint arXiv:1909.09552*.

Wu, Zonghan, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Philip S Yu (2019b). “A comprehensive survey on graph neural networks.” In: *arXiv preprint arXiv:1901.00596*.

Xiao, Chaowei, Ruizhi Deng, Bo Li, Fisher Yu, Mingyan Liu, and Dawn Song (2018a). “Characterizing adversarial examples based on spatial consistency information for semantic segmentation.” In: *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 217–234.

Xiao, Chaowei, Bo Li, Jun-Yan Zhu, Warren He, Mingyan Liu, and Dawn Song (2018b). "Generating adversarial examples with adversarial networks." In: *IJCAI*.

Xiao, Chaowei, Jun-Yan Zhu, Bo Li, Warren He, Mingyan Liu, and Dawn Song (2018c). "Spatially transformed adversarial examples." In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=HyydRMZC-.

Xu, Danfei, Yuke Zhu, Christopher B Choy, and Li Fei-Fei (2017). "Scene graph generation by iterative message passing." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5410–5419.

Xu, Huan, Constantine Caramanis, and Shie Mannor (2009). "Robustness and regularization of support vector machines." In: *Journal of machine learning research* 10.7.

Xu, Keyulu, Chengtao Li, Yonglong Tian, Tomohiro Sonobe, Ken-ichi Kawarabayashi, and Stefanie Jegelka (2018). "Representation learning on graphs with jumping knowledge networks." In: *ICML*, pp. 5449–5458. URL: http://proceedings.mlr.press/v80/xu18c.html.

Xu, Kuo, Jian Wang, and Byonghyo Shim (2014). "The RIP for random matrices with complex Gaussian entries." In: *Future Information Technology, Lecture Notes in Electrical Engineering* 276.

Xu, Weilin, David Evans, and Yanjun Qi (2017). "Feature squeezing: Detecting adversarial examples in deep neural networks." In: *arXiv preprint arXiv:1704.01155*.

Xu, Zhe, Ivan Gavran, Yousef Ahmad, Rupak Majumdar, Daniel Neider, Ufuk Topcu, and Bo Wu (2020). "Joint inference of reward machines and policies for reinforcement learning." In: *Proceedings of the International Conference on Automated Planning and Scheduling* 30, pp. 590–598.

Xupeng Miao, Nezihe Merve Gürel[†],Wentao Zhang, Wentao Zhang, Zhichao Han, Bo Li, Wei Min, Susie Xi Rao, Hansheng Ren, Yinan Shan, Yingxia Shao, et al. (2021). "DeGNN: Improving Graph Neural Networks with Graph Decomposition." In: *ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pp. 1223–1233.

Yang, Greg, Tony Duan, J Edward Hu, Hadi Salman, Ilya Razenshteyn, and Jerry Li (2020). "Randomized smoothing of all shapes and sizes." In: *International Conference on Machine Learning*, pp. 10693–10705.

Yang, Tien-Ju, Yu-Hsin Chen, and Vivienne Sze (2017). "Designing energy-efficient convolutional neural networks using energy-aware pruning." In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5687–5695.

Yuan, Xiaotong, Ping Li, and Tong Zhang (2014). "Gradient hard thresholding pursuit for sparsity-constrained optimization." In: *Proceedings of the 31st International Conference on Machine Learning* 32.2, pp. 127–135.

– (2016). "Exact recovery of hard thresholding pursuit." In: *Advances in Neural Information Processing Systems*, pp. 3558–3566.

Zhai, Runtian, Chen Dan, Di He, Huan Zhang, Boqing Gong, Pradeep Ravikumar, Cho-Jui Hsieh, and Liwei Wang (2019). "MACER: Attack-free and scalable robust training via maximizing certified radius." In: *International Conference on Learning Representations*.

Zhang, Chicheng and Kamalika Chaudhuri (2014). "Beyond disagreement-based agnostic active learning." In: *Advances in Neural Information Processing Systems 27*, pp. 442–450.

Zhang, Dinghuai, Mao Ye, Chengyue Gong, Zhanxing Zhu, and Qiang Liu (2020). "Black-box certification with randomized smoothing: A functional optimization based framework." In: *arXiv preprint arXiv:2002.09169*.

Zhang, Hantian, Jerry Li, Kaan Kara, Dan Alistarh, Ji Liu, and Ce Zhang (2017). "ZipML: Training linear models with end-to-end low precision, and a little bit of deep learning." In: *International Conference on Machine Learning*, pp. 4035–4043.

Zhang, Jintao, Zhuo Wang, and Naveen Verma (2015). "18.4 A matrix-multiplying ADC implementing a machine-learning classifier directly with data conversion." In: *2015 IEEE International Solid-State Circuits Conference-(ISSCC) Digest of Technical Papers*, pp. 1–3.

– (2016). "A machine-learning classifier implemented in a standard 6T SRAM array." In: *2016 IEEE symposium on vlsi circuits (vlsi-circuits)*, pp. 1–2.

Zhao, Jiawei, Steve Dai, Rangharajan Venkatesan, Ming-Yu Liu, Brucek Khailany, Bill Dally, and Anima Anandkumar (2021). "Low-precision training in logarithmic number system using multiplicative weight update." In: *arXiv preprint arXiv:2106.13914*.

Zhen, Peng, Huang Wenbing, Luo Minnan, Zheng Qinghua, Rong Yu, Xu Tingyang, and Huang Junzhou (2020). "Graph representation learning via graphical mutual information maximization." In: *WWW '20: The Web Conference 2020, Taipei, Taiwan, April 20-24, 2020*, pp. 259–270. URL: https://doi.org/10.1145/3366423.3380112.

Zhou, Jie, Ganqu Cui, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, and Maosong Sun (2018). "Graph neural networks: A review of methods and applications." In: *arXiv preprint arXiv:1812.08434*.

Zhu, Xingquan, Peng Zhang, Xiaodong Lin, and Yong Shi (2007). "Active learning from data streams." In: *Seventh IEEE International Conference on Data Mining (ICDM 2007)*, pp. 757–762.

Zhu, Yao, Xiao Wei, and Yue Zhu (2021). "Efficient adversarial defense without adversarial training: A Batch normalization approach." In: *2021 International Joint Conference on Neural Networks*, pp. 1–8.

A APPENDIX

A.1 LABEL-EFFICIENT MODEL SELECTION

A.1.1 *Hyperparameter Setting*

The hyperparameter tuning is performed via grid search. For each grid point, we run the experiment for 100 realizations and compute the average number of requests. The grid search was performed over the following search space:

- CIFAR-10: Model Picker: `[0, 3000]`, Entropy: `[0, 20]`, S-QBC: `[0, 10]`, Importance: `[0, 0.9]`, Efal: `[0, 1.5e-2]`
- ImageNet: Model Picker: `[0, 135]`, Entropy: `[0, 22]`, S-QBC: `[0, 20]`, Importance: `[0, 1]`
- Drift: Model Picker: `[0, 60]`, Entropy: `[0, 4]`, S-QBC: `[0, 4]`, Importance: `[0, 05]`
- EmoContext: Model Picker: `[0, 60]`, Entropy: `[0, 4]`, S-QBC: `[0, 4]`, Importance: `[0, 05]`, Efal: `[0, 1e-2]`
- CIFAR-10 V2: Model Picker: `[0, 1000]`, Entropy: `[0, 3]`, S-QBC: `[0, 10]`, Importance: `[0, 0.9]`, Efal: `[0, 1e-1]`

with grid size of 250 where grid points are equally spaced. The respective number of requests for each grid point can be found in our publicly available repository[1].

Remark that the amount of requests by Model Picker saturates when Model Picker reaches at a high identification probability. Therefore, the update probability is upscaled with a very high value such that Model Picker queries large number of labels, and thus comparison to other methods

[1] https://github.com/DS3Lab/online-active-model-selection

for large budget constraints are made possible. Practically, this would not be required as Model Picker itself decides when to stop requesting labels. For example, when the update probability is upscaled by a factor of 11 for CIFAR-10 V2 dataset, the number of requests made by Model Picker is 3 800 labels, whereas an upscaling of 835 is used to enable Model Picker requests nearly 4 800 labels.

A.1.2 *Extended Results*

We conduct another numerical analysis on the performance of Model Picker when pretrained models have relatively lower accuracies. Towards that, we train 80 models on CIFAR-10 varying in machine learning models and parameters. The accuracy of pretrained models line in 40-70% over a test set of of size 10 000. We compare the model selection methods over this new model collection by following the exact same procedure as in the Section 3.2.5. We use a stream size of 5 000 and average the results over 500 realizations. Figure A.2, Figure A.3 and Figure A.4 encompass the comparison. When the accuracy of pretrained models are low, the query by committee algorithm expectedly underperforms as the disagreement measure becomes noisy under the existence of models with low accuracies. Model Picker, on the other hand, noticeably outperforms in returning the true best model as well as the ranking of the models. The regret analysis in Figure A.5 suggests that the structural query by committee method maintains a low regret throughout the streaming process as well as for different labeling budgets, and very closely followed by Model Picker.

A.2 KNOWLEDGE ENHANCED ADVERSARIAL ROBUSTNESS

A.2.1 *Implementation Details for Traffic Sign Identification*

To implement a nontrivial KEMLP pipeline for traffic sign identification, we need to design informative knowledge rules, connecting useful sensory information to each type of traffic sign. The full GTSRB dataset contains 43 types

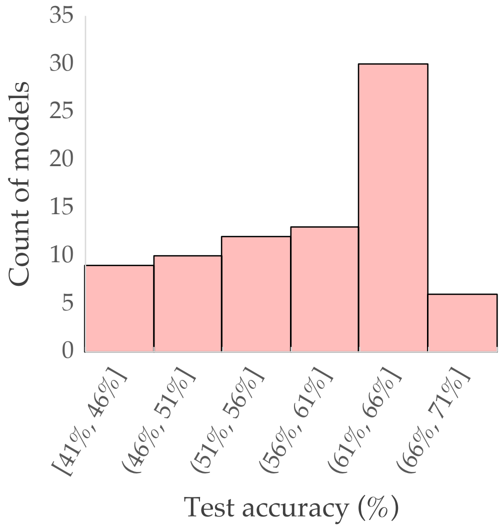
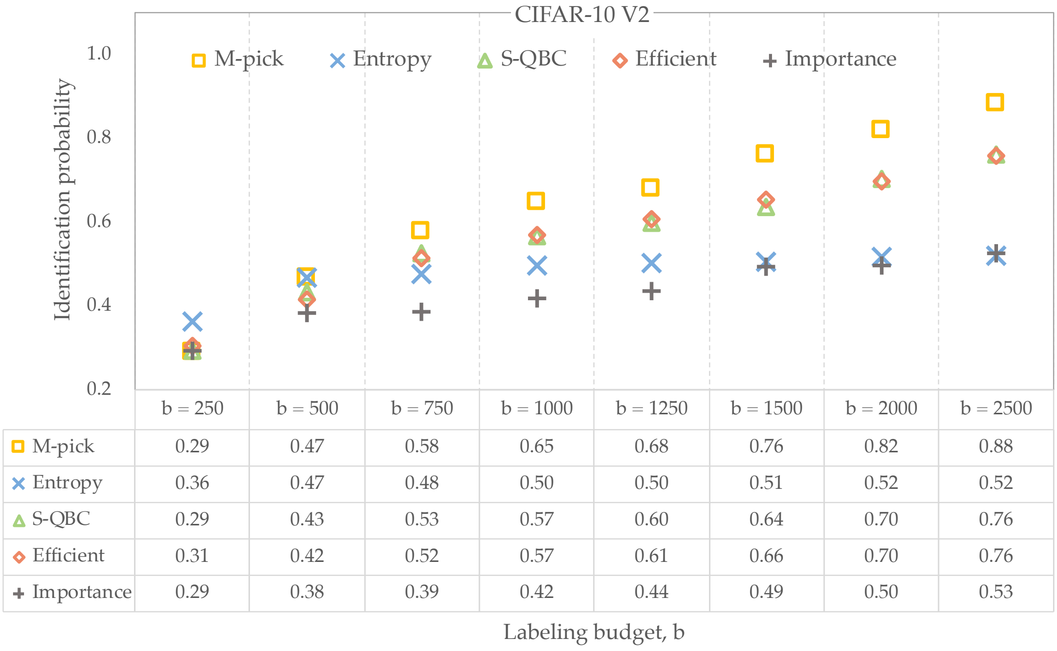


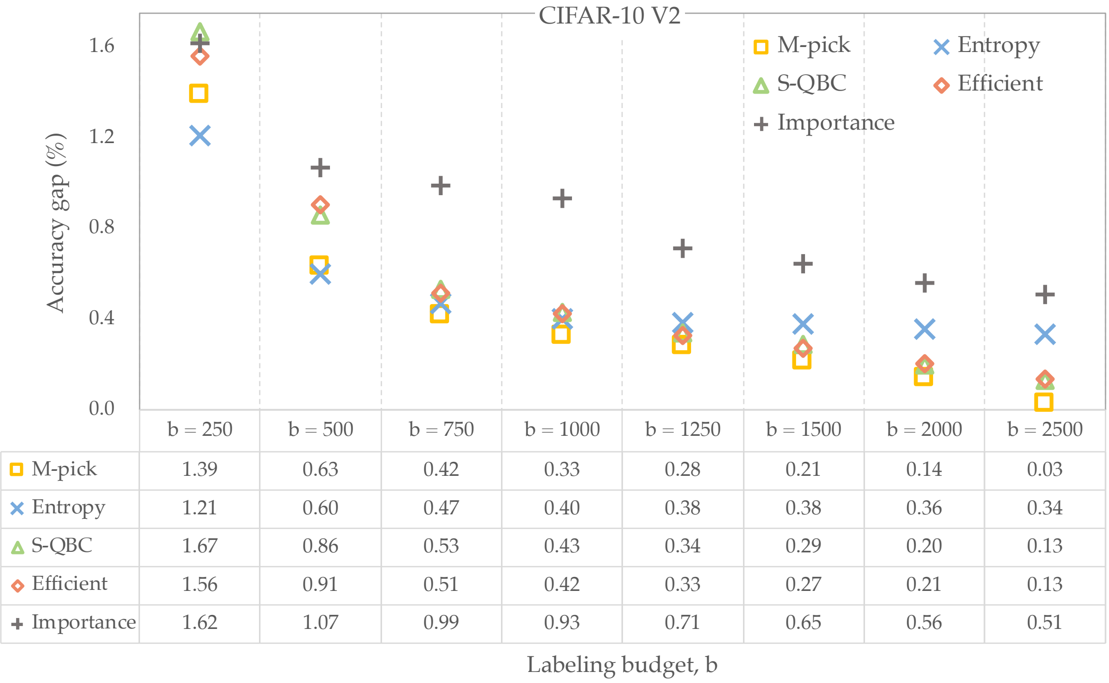
Figure A.1: Histogram of test accuracies for CIFAR-10 V2 dataset



	b = 250	b = 500	b = 750	b = 1000	b = 1250	b = 1500	b = 2000	b = 2500
□ M-pick	0.29	0.47	0.58	0.65	0.68	0.76	0.82	0.88
× Entropy	0.36	0.47	0.48	0.50	0.50	0.51	0.52	0.52
△ S-QBC	0.29	0.43	0.53	0.57	0.60	0.64	0.70	0.76
◇ Efficient	0.31	0.42	0.52	0.57	0.61	0.66	0.70	0.76
+ Importance	0.29	0.38	0.39	0.42	0.44	0.49	0.50	0.53

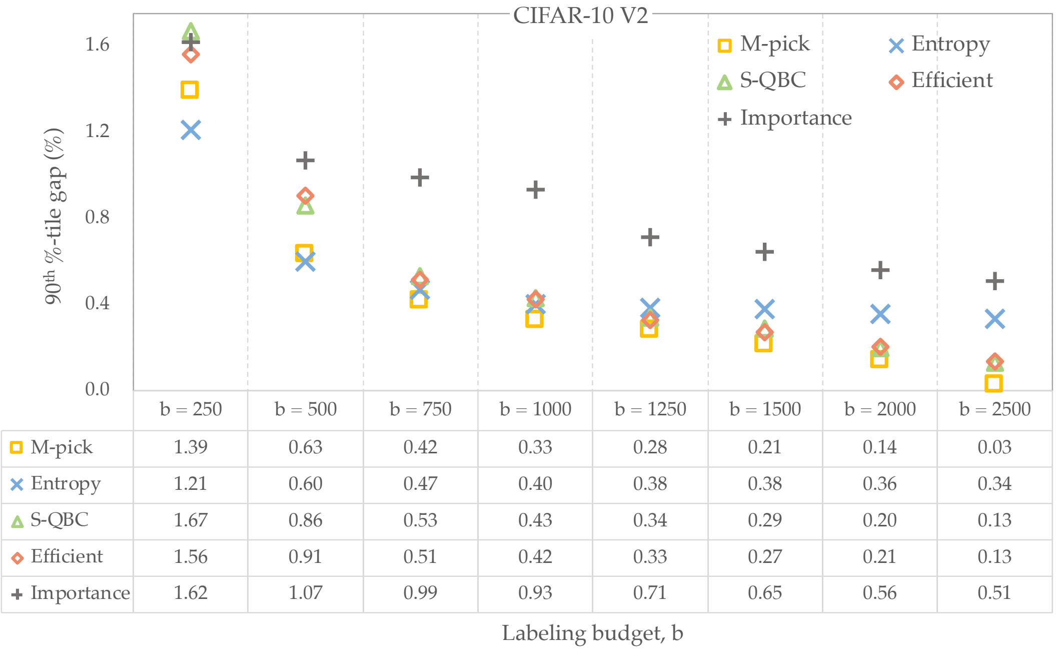
Figure A.2: Identification probability for CIFAR-10 V2 dataset

of signs, thus it requires a large amount of fine-grained sensory information and corresponding knowledge rules to distinguish between different signs, which requires a heavy engineering workload. Since the main purpose of this work is to illustrate the knowledge enhancement methodology rather than engineering practice, alternatively, we only consider a 12-class subset (as



	b = 250	b = 500	b = 750	b = 1000	b = 1250	b = 1500	b = 2000	b = 2500
□ M-pick	1.39	0.63	0.42	0.33	0.28	0.21	0.14	0.03
× Entropy	1.21	0.60	0.47	0.40	0.38	0.38	0.36	0.34
△ S-QBC	1.67	0.86	0.53	0.43	0.34	0.29	0.20	0.13
◇ Efficient	1.56	0.91	0.51	0.42	0.33	0.27	0.21	0.13
+ Importance	1.62	1.07	0.99	0.93	0.71	0.65	0.56	0.51

Figure A.3: Accuracy gap for CIFAR-10 V2 dataset



	b = 250	b = 500	b = 750	b = 1000	b = 1250	b = 1500	b = 2000	b = 2500
□ M-pick	1.39	0.63	0.42	0.33	0.28	0.21	0.14	0.03
× Entropy	1.21	0.60	0.47	0.40	0.38	0.38	0.36	0.34
△ S-QBC	1.67	0.86	0.53	0.43	0.34	0.29	0.20	0.13
◇ Efficient	1.56	0.91	0.51	0.42	0.33	0.27	0.21	0.13
+ Importance	1.62	1.07	0.99	0.93	0.71	0.65	0.56	0.51

Figure A.4: 90th %-tile gap for CIFAR-10 V2 dataset

shown in Figure A.6) in our experiment, where the selected signs have diverse appearance and high frequencies.

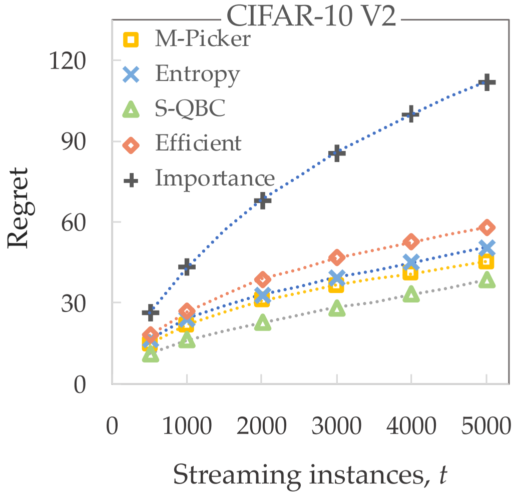


Figure A.5: Regret for CIFAR-10 V2 dataset

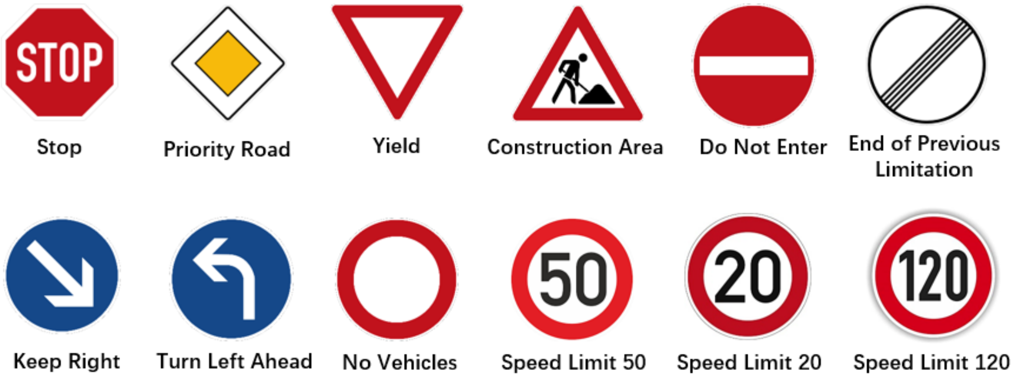


Figure A.6: The selected 12 types of signs from the full GTSRB.

For detailed KEMLP pipeline implementation, we consider two orthogonal domains — logic domain and sensing domain, respectively.

In the logic domain, based on the specific tasks we need to deal with, we design a set of knowledge rules, which determine the basic logical structure of the predefined reasoning model. Specifically, for our task of traffic sign identification on the 12-class dataset, in total, we have designed 12 pieces of permissive knowledge rules and 12 pieces of preventative knowledge rules for the selected 12 types of signs. Each type of sign shares exactly one permissive knowledge rule and one preventative knowledge rule, respectively.

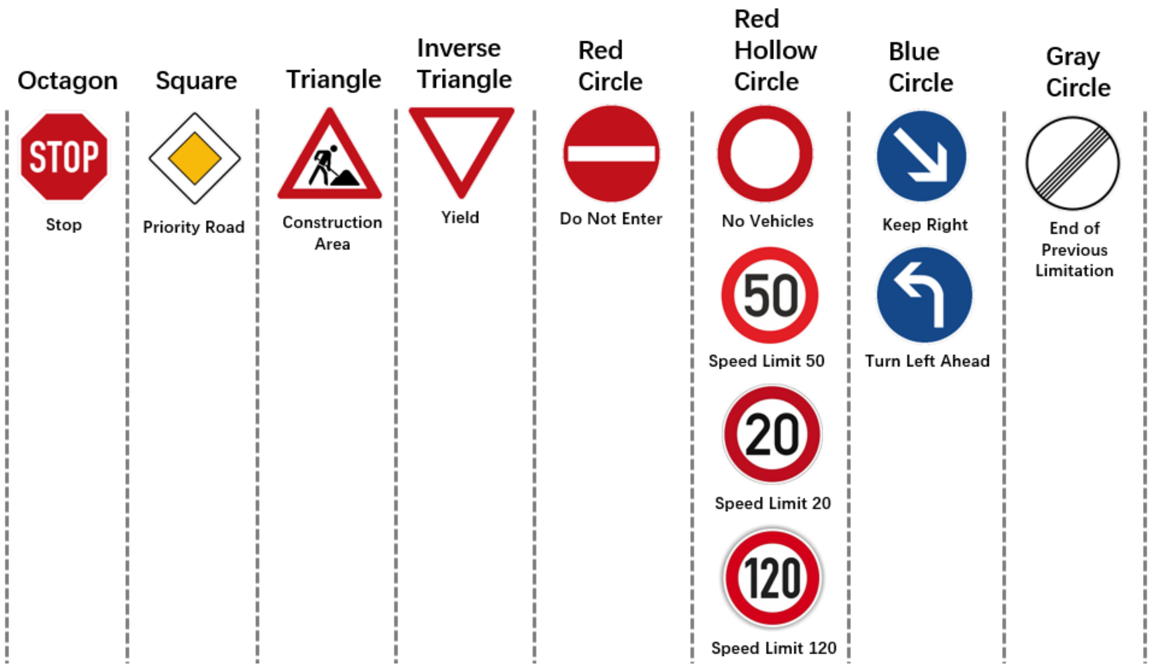


Figure A.7: Border patterns of the selected signs.

In our design, we take *border patterns* and *sign contents* of the traffic signs as the sensory information to construct knowledge rules. As shown in Figure A.7, based on the border pattern, we can always construct a preventative knowledge rule for each sign based on its border in the form as *if it is a stop sign, it should be of the shape of octagon*. In our 12-class set, since there are six types of signs (“Stop", “Priority Road", “Construction Area", “Yield", “Do Not Enter", “End of Previous Limitation") sharing the unique border pattern, we also design an permissive rule for each of the six classes based on their borders, e.g. *if the sign is of the shape of octagon, it must be a stop sign*. Then, for the rest of the six types (“No Vehicles", “Speed Limit 50", “Speed Limit 20", “Speed Limit 120", “Keep Right", “Turn Left Ahead"), whose borders can not uniquely determine their identity, we use their unique sign content to design permissive rules for them. Specifically, we define the content pattern *Blank Circle*, *Digits-20*, *Digits-50*, *Digits-120*, *Arrow-Right-Down*, *Arrow-Left-Ahead* to distinguish between these signs. We present the permissive relations in Figure A.8, Figure A.9 and Figure A.10.

In the sensing domain, the principal task is to design a set of reliable auxiliary models to identify those sensory information required by the knowledge

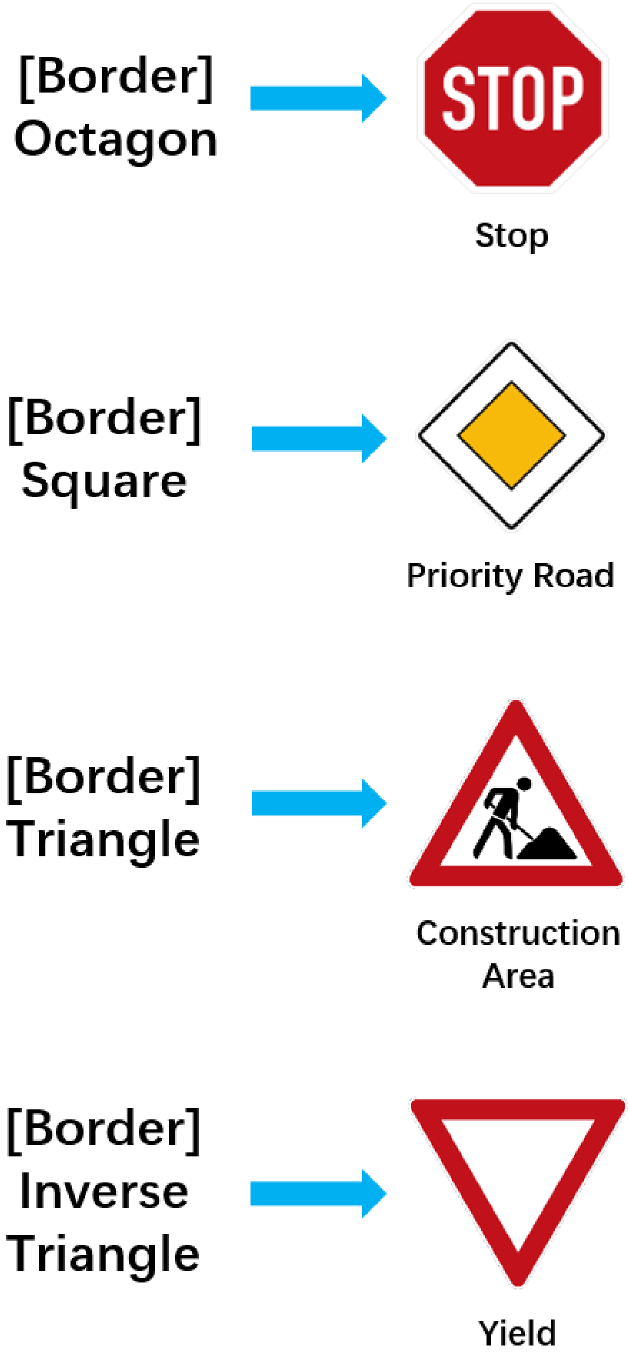


Figure A.8: permissive relations for each sign.

rules defined in the logic domain. For traffic sign identification, we adopt a non-neural pre-processing plus neural identification workflow to identify the border and content of each type. Specifically, to identify the border type (e.g. shape and color), we first use GrabCut (Rother, Kolmogorov, and Blake, 2004)

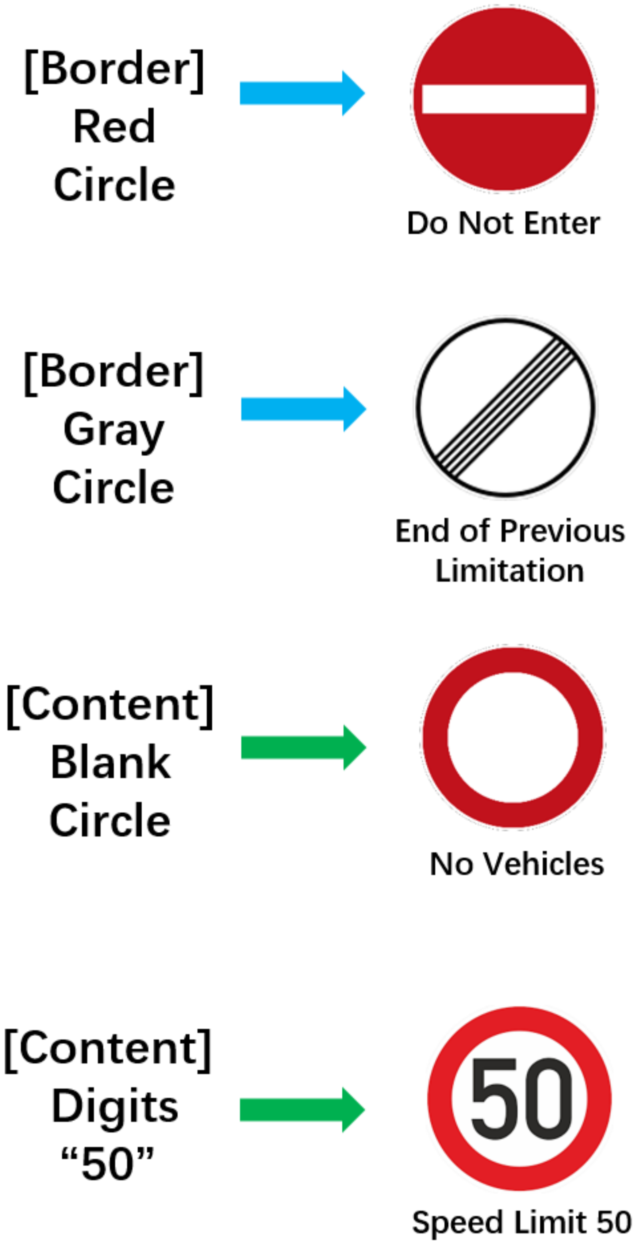


Figure A.9: permissive relations for each sign.

to get the mask of the sign and then discard all pixels of sign content and background, only retaining the border pixels, and finally a binary CNN classifier is used to make the statistical prediction (e.g. predict whether the shape is octagon only based on the border pixels). For sign content, similarly, we first use GrabCut to filter out all irrelevant pixels except for the sign content, and then the edge operator will extract the contour of the content, finally

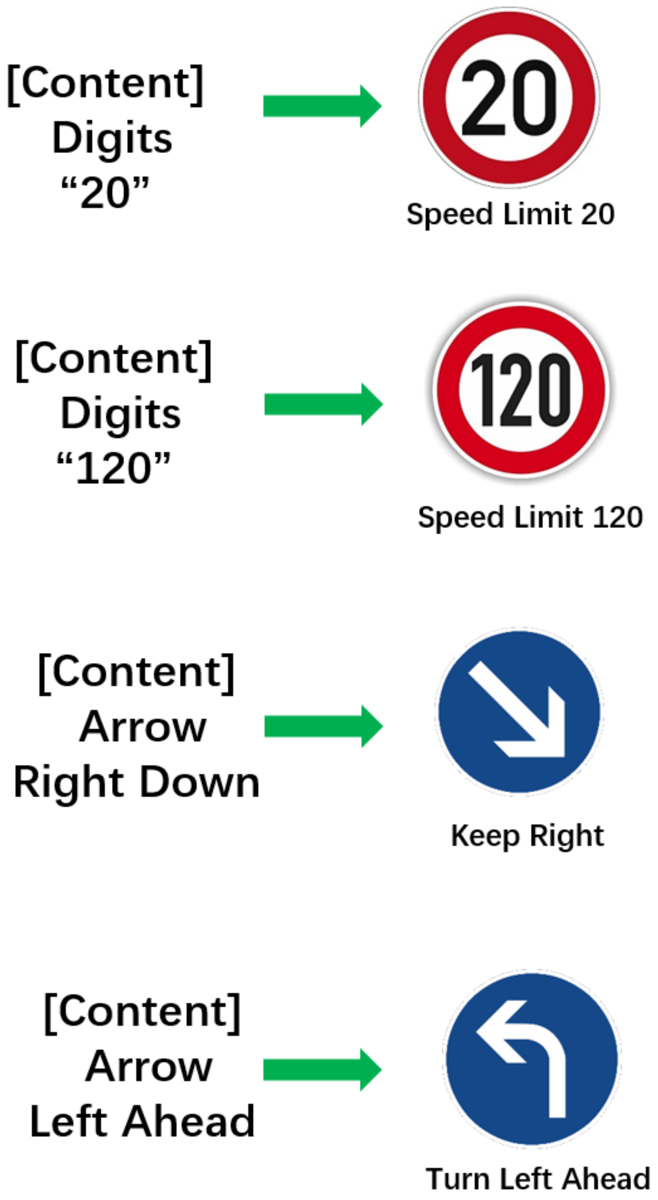


Figure A.10: permissive relations for each sign.

CNN models are applied to recognize specific features like digits, arrows and characters. In Figure A.11, we provide an overview of the workflow of our implemented auxiliary models.

In total, in our KEMLP pipeline, we implement 19 submodels — 1) One end-to-end GTSRB-CNN classifier (Eykholt et al., 2018) as the main task model; 2) 8 binary preventative models for all 8 types of borders; 3) 6 binary permissive models for the 6 border types, each of which is shared only by a

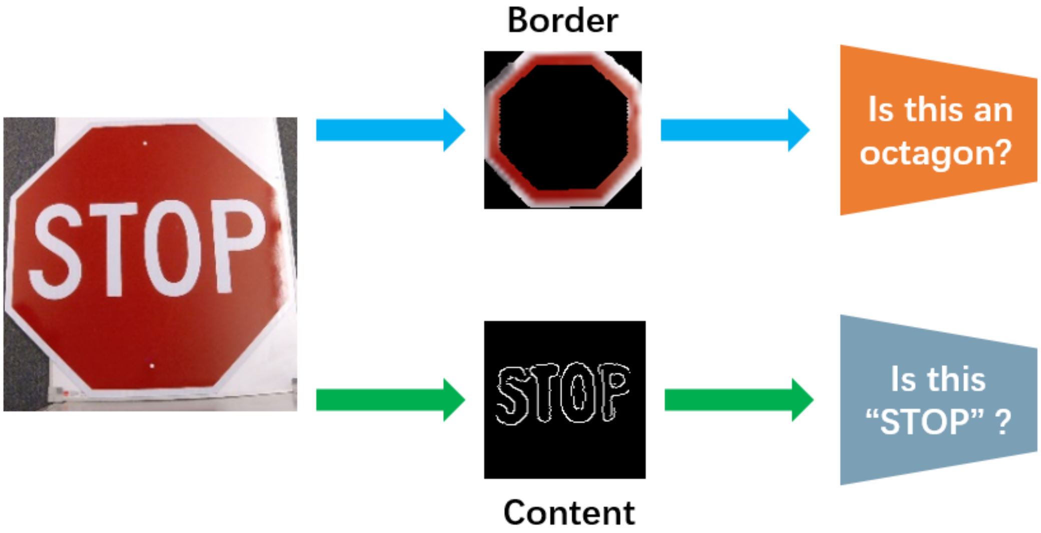


Figure A.11: Overview: workflow of the auxiliary models.

unique class of sign; 4) 3 binary permissive models based on edge map of sign content (*Blank Circle*, *Arrow-Right-Down*, *Arrow-Left-Ahead*); 5) A single permissive model for digit recognition, which is used to identify *Digits-20*, *Digits-50*, *Digits-120*. All of the 17 binary classification neural models adopt the same backbone architecture in GTSRB-CNN and the rest digit recognition model adopts the architecture proposed in (Goodfellow et al., 2013).

Training Details. To make our KEMLP pipeline function normally as the way we expect, next, we consider the training issues of the overall model.

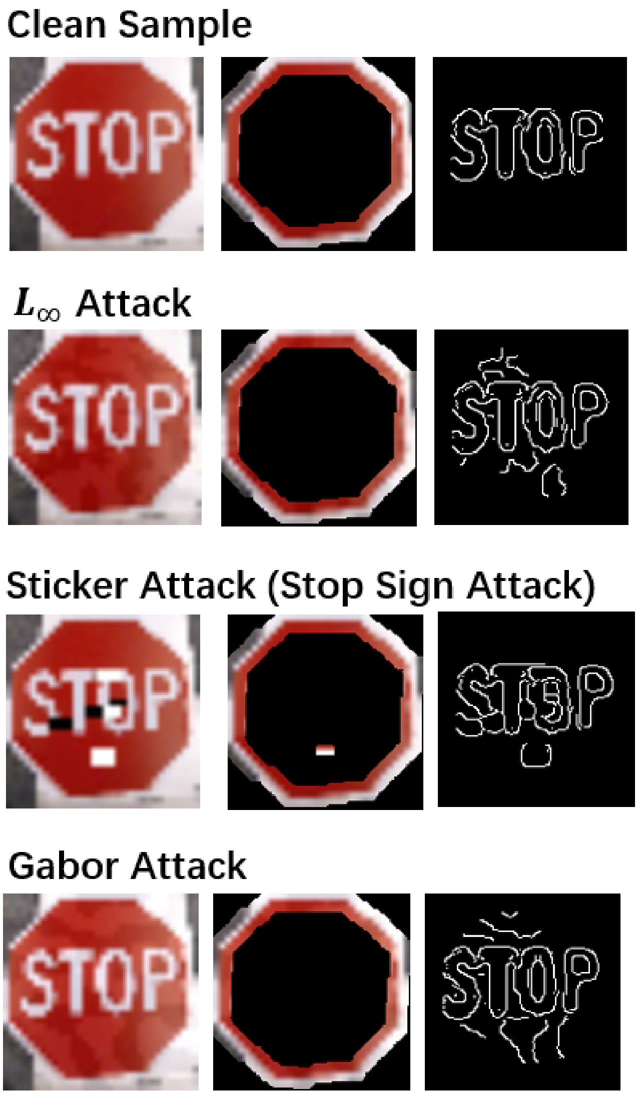
Given the definition of permissive and preventative models, ideally, the permissive models should have low false rate and nontrivial truth rate, while the preventative models should have high truth rate and nontrivial false rate. These conditions are very critical for auxiliary models to bring accuracy improvement into the KEMLP pipeline. We guarantee the conditions to hold by assigning biased weights to classification loss on positive samples and negative samples during the training stage. Specifically, we train all of our binary auxiliary models with the following loss function:

$$L(\mathcal{D}, f) = a\mathbb{E}_{x\sim\mathcal{D}^+}[CE(f(x), 1)] + b\mathbb{E}_{x\sim\mathcal{D}^-}[CE(f(x), 0)],$$

where $\mathcal{D} = \{D^+, D^-\}$ is the dataset, D^+ is the subset containing positive samples, D^- is the subset containing negative samples, f is the classifier and CE is the crossentroy loss. For permissive model, we set $a << b$, so that low false rate will be encouraged at the cost of truth rate; while for preventative sensors, we set $a >> b$, then we can expect a high truth rate at the cost of some false rate.

Besides the performance of each individual model, we also need to get proper weights for the reasoning graphical model in the KEMLP pipeline. Empirically, in our traffic sign identification task, since the end-to-end main task model has almost perfect accuracy on clean data, directly training on clean data will always give the main task model a dominant weight, leading to a trivial pipeline model. Thus, during training, we augment the training set with artificial adversarial samples, where the sensing signal from the main task model is randomly flipped. As a result, during training, to make correct predictions on these artificial adversarial samples, the optimizer must also assign nontrivial weights to other auxiliary models. We call the ratio of such artificial adversarial samples in the training set the "adversarial ratio" in our context, indicating prior belief on the balance between benign and adversarial distributions, and use β to denote it. In our evaluation, we test different settings of $\beta \in \{0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0\}$ and report the best results in Table 4.2, Table 4.7,Table 4.8,Table 4.1,Table 4.4, Table 4.5, Table 4.9, Table 4.9, Table 4.11, Table 4.11, Table 4.6. In particular, we use $\beta = 0.8$ in Table 4.2, Table 4.4 and Table 4.5 $\beta = 0.2$ in Table 4.7,Table 4.8, Table 4.9, Table 4.9, Table 4.11, Table 4.12, Table 4.6 and $\beta = 0.4$ in Table 4.1.

For all the neural models, we use the standard Stochastic Gradient Descent Optimizer for training. The optimizer adopts a learning rate of 10^{-2}, momentum of 0.9 and weight decay of 10^{-4}. In all the training cases, we use 50000 training iterations with a batch size of 200 for each random training iteration. To train the weights of the graphical model in the pipeline, we perform Maximum Likelihood Estimate (MLE) with the standard gradient descent algorithm, and we use a learning rate of 10^{-1} and run 4000 training iterations with a batch size of 50 for each random iterations.



Clean Sample

L_∞ Attack

Sticker Attack (Stop Sign Attack)

Gabor Attack

Figure A.12: Visualization of adversarial examples and corrupted samples.

A.2.2 *Visualization of Adversarial Examples and Corrupted Samples*

From Figure A.12 to Figure A.14, we provide a visualization of the generated adversarial examples (corrupted samples) that are used for robustness evaluation in our work. For each type of attack (corruption), we present the generated example (the first image in each block), the extracted border (the second image in each block), and the sign content (the third image in each block) from the sample.

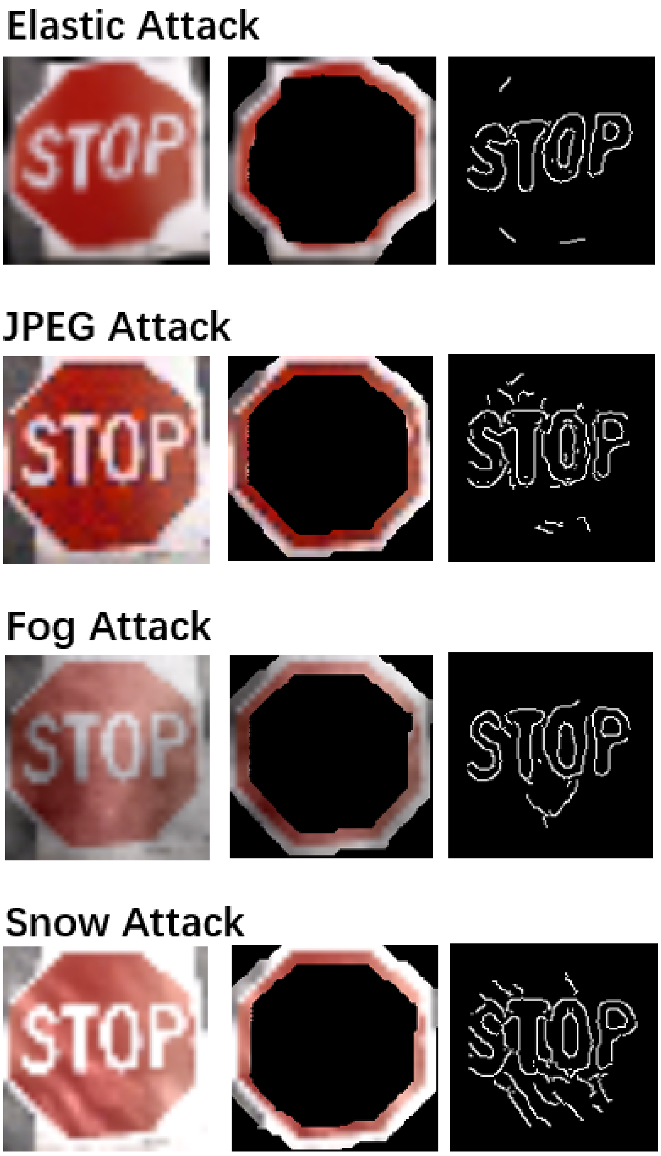
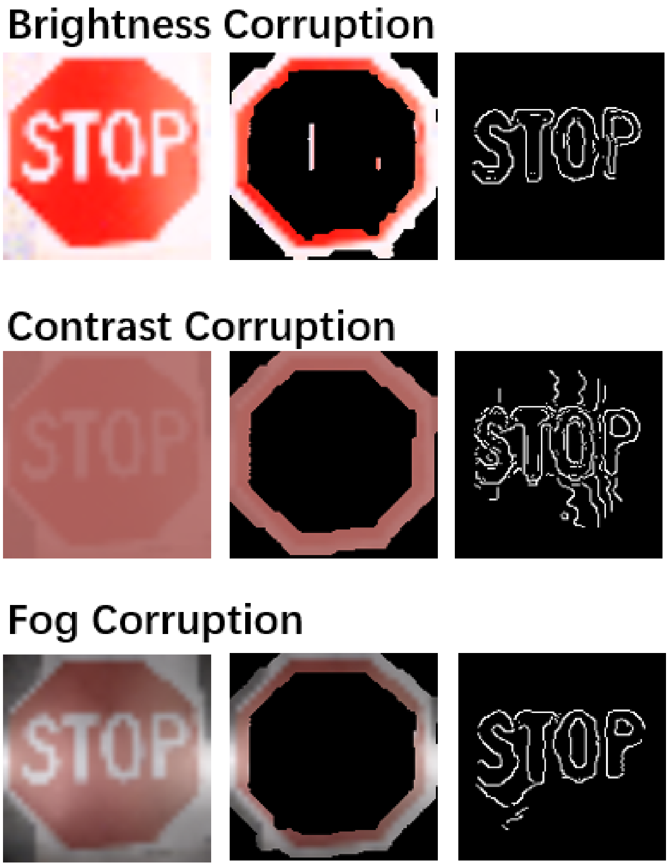


Figure A.13: Visualization of adversarial examples and corrupted samples.

As we can see, although the adversarial examples can easily fool an end-to-end neural network based main task model, the non-neural GrabCut algorithm and edge operator can still correctly extract the border and sign content from them. This allows other auxiliary models help to rectify the mistakes made by the main task model.



Brightness Corruption

Contrast Corruption

Fog Corruption

Figure A.14: Visualization of adversarial examples and corrupted samples.